Numeral Analysis Course Project

Chen Ziyi   2021011633

June 27, 2023

1 Programming Problem 1

Given a grid on the interval $[a, c]$, say a partition of the interval into subintervals

$$[a, c] = \bigcup_{i=1}^{n} [a_i, c_i].$$

On each of the subinterval, k Chebyshev nodes are given. Discretize the integral equation to obtain a linear system $Ax = b$. Solve the linear system by calling a standard linear algebra solver. (This forms a direct solver of the integral equation/ODE, which works on an adaptive grid but is not automatically adaptive.)

Numerical analysis:

The Matlab codes please see the website https://github.com/chenziyiTHU/Numerical-Analysis-project-2023.git.

Now we take the example 1 in Lee and Greengard as an numerical experiment of our codes. For convenience, we directly take equidistent partition with number n of the interval $[-1, 1]$.For $n = 4, 8, 12, 16, 20, 24, 28, 32, 35$, the numerical solution is shown below.

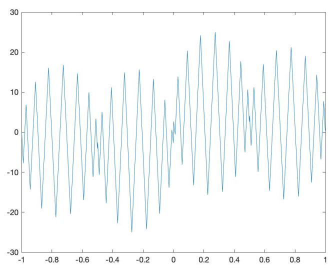
From this numerical result, we see that the result highly depends on the choice of the partition. Although the case $n = 24$ has less intervals than $n = 32$, but it performs better. The perfomance of $n = 35$ is even much worse, which is because $n = 35$ is odd and the origin, which achieves large derivative of the exact solution, lies in the interior of an interval. Thus, we can conclude that the error of the direct solver does not always decreases as n grows larger.

2 Programming Problem 2

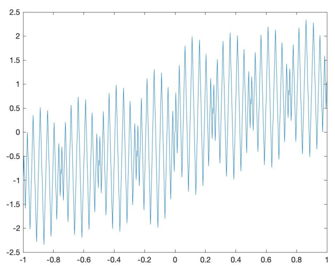
Implement the full algorithm discussed in Lee and Greengard.

Numerical analysis:

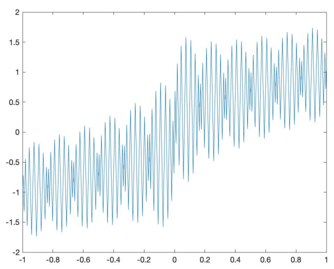
n=4



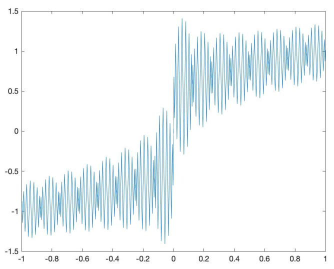
n=8



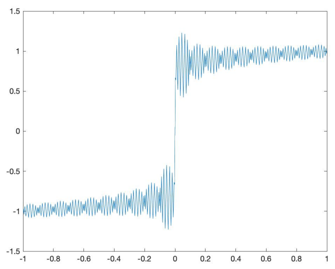
n=12



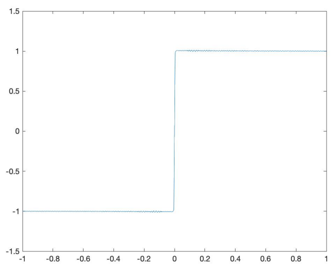
n=16



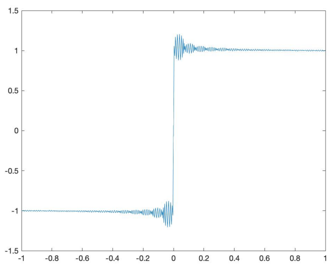
n=20



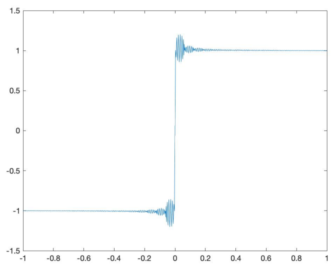
n=24



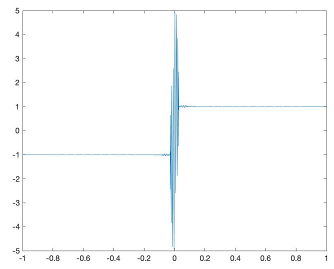
n=28



n=32



n=35

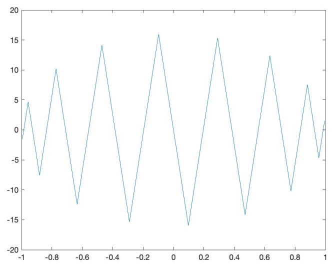


The Matlab codes please see the website https://github.com/chenziyiTHU/Numerical-Analysis-project-2023.git.

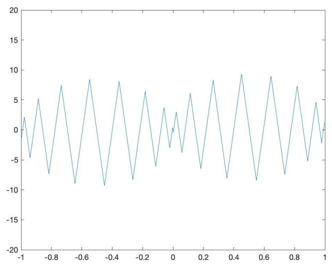
We first take the example 1 in Lee and Greengard as an numerical experiment of our codes. Just following the paper, the order of the method is fixed to be 16 and the initial mesh is just a single interval.

For $\epsilon = 10^{-5}$, we plot the numerical solution at refinement step from 1 to 9 as below.

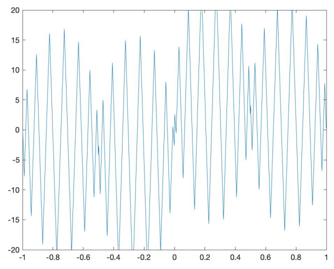
refinement step=1



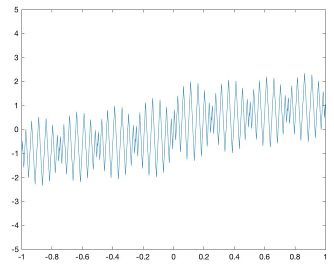
refinement step=2



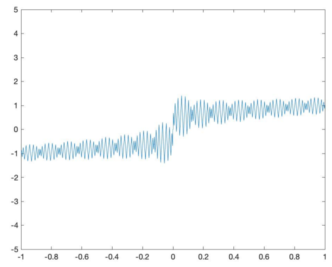
refinement step=3



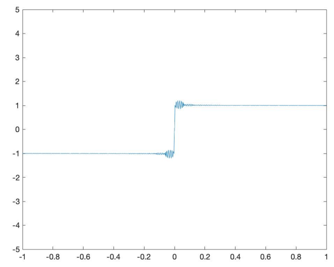
refinement step=4



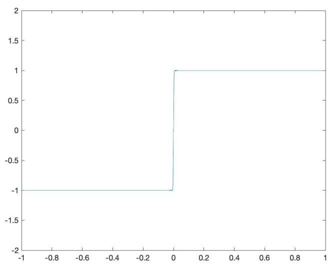
refinement step=5



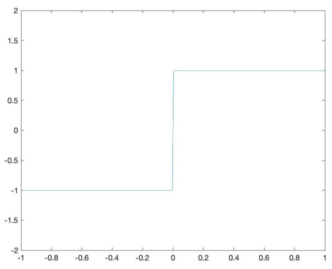
refinement step=6



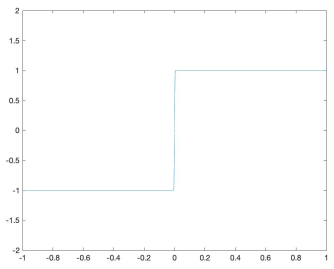
refinement step=7



refinement step=8



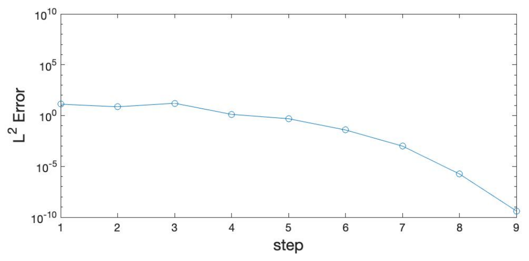
refinement step=9

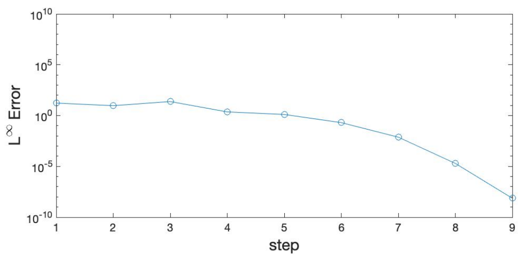


The L^2 norm and L^∞ norm of the error as a function of the refinement step is plotted in the graph below. This result highly coincide with the result in the paper, which assures that our codes are quite convincible.

We also examined the behavior of the algorithm over a wide range of ϵ, following the paper.

Solution Convergence





The amount of time required for these cases when runned on my commputer, the number of loops and the number of intervals are summarized in the table below.

ϵ	10^{-12}	10^{-6}	10^{-8}	10^{-10}
R	9	13	15	18
M_R	18	26	56	152
L^2 Error	$3.75 \cdot 10^{-12}$	$5.11 \cdot 10^{-13}$	$3.17 \cdot 10^{-12}$	$3.11 \cdot 10^{-11}$
Time	0.7	3.0	11.1	56.0

From this table, we can see that the result performs much better then the direct solver in Problem 1, while it still has some gap between the result in the paper. This gap may comes from the efficiency of the coding structure and the computer.

3 Programming Problem 3

Use Newton's iteration on top of the linear solver to form a nonlinear solver (which is also fast and adaptive), for the equation

$$u''(x) = f(x, u, u').$$

Numerical analysis:

The Matlab codes please see the website https://github.com/chenziyiTHU/Numerical-Analysis-project-2023.git.

We consider the following Newton iteration:

Step 1. Pick a initial function u_0.

Step 2. We want to find a function δ such that $u_0 + \delta$ satisfies the nonlinear ordinary differential equation, i.e.

$$u_0'' + \delta'' = f(x, u_0 + \delta, u_0' + \delta').$$

Approximating the right-hand side with Taylor expansion to linear term, the equation becomes

$$u_0'' + \delta'' = f(x, u_0, u_0') + \partial_2 f(x, u_0, u_0') \cdot \delta + \partial_3 f(x, u_0, u_0') \cdot \delta'.$$

Step 3. Now we can use the linear solver constructed in Problem 2 to obtain δ, and let $u_1 = u_0 + \delta$.

We consider example of nonlinear ordinary differential equation:

$$y''(x) = -\frac{1}{x^2}\frac{y}{y+1} + \frac{1}{y}\frac{y'}{(y+1)^2},$$

with boundary conditions

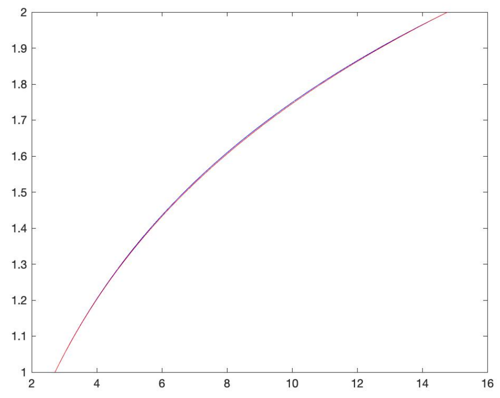
$$y(e) = 1, \quad y(2e^2) = 2,$$

whose exact solution is given by the well-known Lambert W function, which is the inverse function of

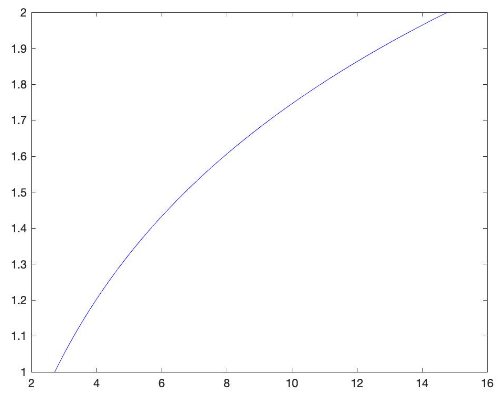
$$y(x) = xe^x.$$

The numerical result with iteration step 1 and 3 and the L^∞ error as a function of iteration step are shown in the figure below. The blue line represents the numerical solution, and the red line is the exact solution, i.e. Lambert W function. As we can see, this Newton's iteration method for nonlinear ordinary equations converges quite efficiently.

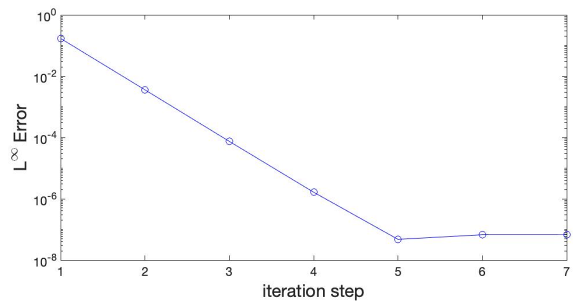
iteration step=1



iteration step=3



Solution Convergence



4 Theoretical Problem 1

Suppose $[a,c] = \bigcup_{i=1}^{n}[a_i, c_i]$, and the Chebyshev nodes on interval $[a_i, c_i]$ are $\{\tau_j^i\}_{j=1}^{K}$. So we may regard the operator

$$P : \mathbb{R}^{nK} \to \mathbb{R}^{nK}, (\sigma(\tau_i^j)) \mapsto P(\sigma(\tau_i^j))$$

as a block matrix with $n \times n$ blocks, each block is a submatrix of size $K \times K$.

For each $i \in \{1, \cdots, n\}$, define the discrete left integral operator on interval $[a_i, c_i]$ by

$$L_i : \mathbb{R}^K \to \mathbb{R}^K.$$

Take $(\sigma_j) \in \mathbb{R}^K$, let $\phi(x)$ be the Chebyshev interpolant with value σ_j at $x = \tau_j^i$. Let

$$(L_i\sigma)_j = \int_{a_i}^{\tau_j^i} \phi(t)dt.$$

Similarly we can define the discrete right integral operator R_i. We may regard these operators as matrices natrually.

By definition,

$$P\sigma = \sigma + \Psi_l L(g_l \circ \sigma) + \Psi_r R(g_l \circ \sigma),$$

where Ψ_l is a diagonal matrix with entries $\psi_l(\tau_j^i)$ and so is Ψ_r, and the operation $\circ$ represents multiplying two vectors by entries.

In form of matrix, we can write

$$L = \begin{pmatrix} L_{11} & & & \\ L_{21} & L_{22} & & \\ \vdots & & \ddots & \\ L_{n1} & \cdots & \cdots & L_{nn} \end{pmatrix}$$

where L_ii is exactly L_i. Since all rows of

$$\begin{pmatrix} L_{i+1,i} \\ L_{i+2,i} \\ \vdots \\ L_{n,i} \end{pmatrix}$$

gives the same contribution to the integral, so the rows are the same, which means this submatrix has rank 1.

Similarly,

$$\begin{pmatrix} R_{1,i} \\ R_{2,i} \\ \vdots \\ R_{i-1,i} \end{pmatrix}$$

has rank 1. If we choose $c_i - a_i$ small enough, then the norm of the diagonal blocks of matrix $P - I$ tends to zero. So the rank of P is nK if we choose the partition fine enough.

Therefore, we obtain the rank structure of the matrix P, i.e. A in the problem.

5 Theoretical Problem 2

No idea.

6 Theoretical Problem 3

No idea as well.

7 Theoretical Problem 4

Yes. We consider the mesh with dimension changing from 1 to 2. Locally, for a small square, we can solve the linear ordinary differential equation using discrete fourier transformation of dimension 2. Then we can construct a 'quad-tree' to optimize the method to an adaptive one, with similar algorithm of upward sweep and downward sweep to construct the coefficients.