MA 580 Assignment 6

Kyle Hansen

Due 24 Nov. 2025
Turned in 9 Dec. 2025

AI Use Statement: No AI used for this assignment

Question 1 Newton Iteration

The results for $c = 1, 10, 100$ using $n_x = 2^7$ are shown in Figure 1. The red circles in these plots show $\|r_k\|_2 / \|r_{k-1}\|_2^2$, which converges to a constant with successive iterations for quadratic convergence.

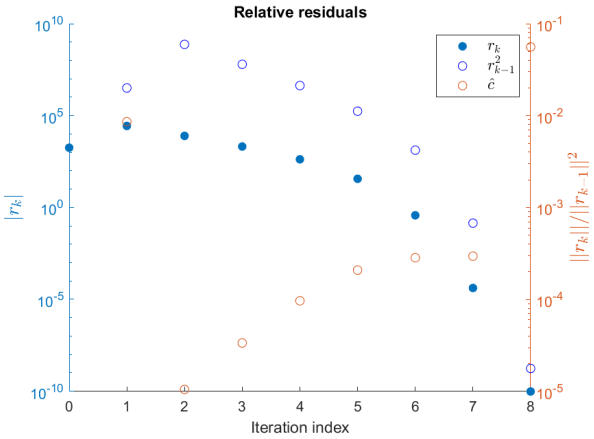
The results for different grids are show in Table 1. The plots clearly demonstrate quadratic convergence, with the number of required iterations showing relative independence from the spatial resolution.

Though the total iterative error $r_0\tau_{rel} + \tau_{abs}$ converged in a similar number of steps for each case, the relative residual of the coarser grids was lower after completing iterations (suggesting that, for refined grids, absolute residual dominates).

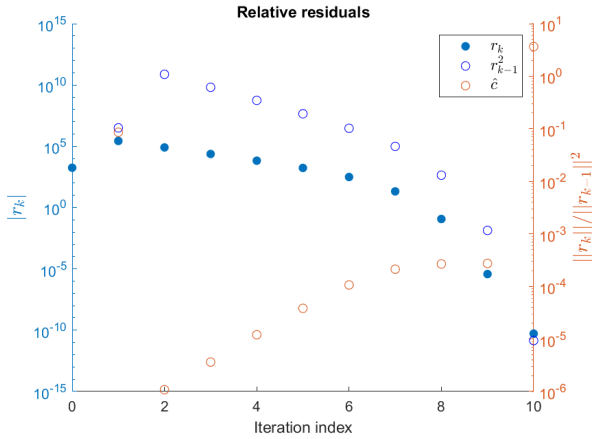
The cost of solution could be further improved with iterative methods of inverting J. The code is listed in the Appendix.

Table 1: Number of required iterations for congergence with $\tau = 10^{-10}$

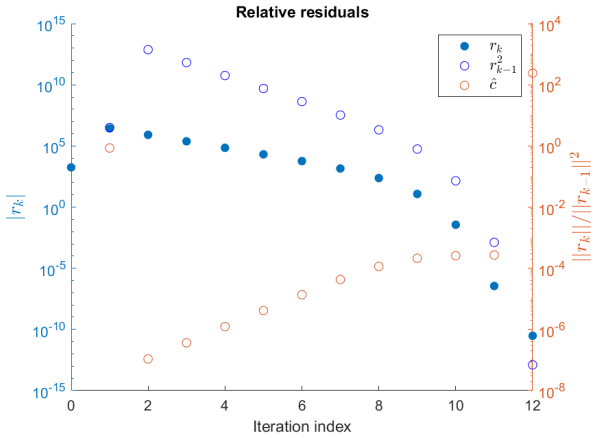
n_x	k (iterations)	$\|r_{final}\|_2$
2^3	12	2.22×10^{-16}
2^4	12	7.19×10^{-13}
2^5	12	2.41×10^{-11}
2^6	13	4.23×10^{-15}
2^7	13	1.71×10^{-14}
2^8	13	6.74×10^{-14}



(a) $c = 1$



(b) $c = 10$



(c) $c = 100$

Figure 1: Residuals, $c = 1, 10, 100$

Question 2 Crank-Nicolson

2(a) Fixed-point Iteration The fixed point iterations are given by

$$\boldsymbol{y}_{n+1}^{(s+1)} = \Phi\left(\boldsymbol{y}_{n+1}^{(s)}\right) = \boldsymbol{y}_n + \frac{h}{2}\left[\boldsymbol{f}(t_n, \boldsymbol{y}_n) + \boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1}^{(s)})\right] \tag{1}$$

starting from initial iterate $\boldsymbol{y}_{n+1}^{(0)}$. To converge, Φ must be a contraction, so that each successive iterate reduces the residual. Using

$$\begin{aligned}&\left\|\Phi\left(\boldsymbol{y}_{n+1}^{\star}\right) - \Phi\left(\boldsymbol{y}_{n+1}\right)\right\|_2 \\ &\quad= \left\| \cancel{\boldsymbol{y}_n} + \frac{h}{2}\left[\cancel{\boldsymbol{f}(t_n, \boldsymbol{y}_n)} + \boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1}^{\star})\right] - \cancel{\boldsymbol{y}_n} - \frac{h}{2}\left[\cancel{\boldsymbol{f}(t_n, \boldsymbol{y}_n)} + \boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1})\right]\right\|_2,\end{aligned} \tag{2}$$

and given the given Lipschitz condtion,

$$\left\|\Phi\left(\boldsymbol{y}_{n+1}^{\star}\right) - \Phi\left(\boldsymbol{y}_{n+1}\right)\right\|_2 = \frac{h}{2}\left[\boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1}^{\star}) - \boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1})\right] \leq L\left\|\boldsymbol{y}_{n+1}^{\star} - \boldsymbol{y}_{n+1}\right\|_2, \tag{3}$$

so,

$$\left\|\Phi\left(\boldsymbol{y}_{n+1}^{\star}\right) - \Phi\left(\boldsymbol{y}_{n+1}\right)\right\|_2 \leq \frac{2L}{h}\left\|\boldsymbol{y}_{n+1}^{\star} - \boldsymbol{y}_{n+1}\right\|_2. \tag{4}$$

This is a contraction only if

$$\alpha = \frac{2L}{h} < 1, \tag{5}$$

which gives the convergence condition for the fixed-point iteration to the solution $\boldsymbol{y}_{n+1}^{\star}$.

2(b) Newton Iteration The zero-finding problem for Crank-Nicolson using Newton's method uses the form

$$F(u) = 0 = \boldsymbol{y}_n + \frac{h}{2}\left[\boldsymbol{f}(t_n, \boldsymbol{y}_n) + \boldsymbol{f}(t_{n+1}, u)\right] - u. \tag{6}$$

The Newton iterates are found with

$$u_{i+1} = u_i + J_F^{-1} F(u_i), \tag{7}$$

where the Jacobian is given by

$$J_F = \frac{h}{2}\begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_d}{\partial u_1} & \cdots & \frac{\partial f_d}{\partial u_d} \end{bmatrix} - \mathbf{I} \tag{8}$$

since the Jacobian of u is $\mathbf{I}$. The initial state could be set to the state at the previous time $\boldsymbol{y}_n$, or an extrapolation from multiple solutions at previous times. If h is small, the previous state is a good guess and should converge in few iterations.

Question 3 Gershgorin Disks

Assuming $\mathbf{A}$ is strictly diagonally dominant, and it has zero as an eigenvalue, then by Gershgorin's theorem therem, zero must lie in one of the Gershgorin disks. That is, there is some i for which

$$|a_{ii} - 0| \leq \sum_{j \neq i} |a_{ij}|. \tag{9}$$

Then,

$$|a_{ii}| \not> \sum_{j \neq i} |a_{ij}| \tag{10}$$

and $\mathbf{A}$ is not strictly diagonally dominant. Thus, if a matrix is strictly diagonally dominant, it cannot have zero as an eigenvalue.

Question A MATLAB Code

```
% HW 6, problem 1.
% nonlinear PDE, -kLu + cu^3 = f
    % L = laplacian
    % k, c = constants
    % f = -2pi[cos(2 pi x_1)sin^2(pi x_2) + sin^2(pi x_1) cos(2 pi x_2)]

% Solves using Newton Iteration

close all

% physical constants
kap = 0.1;
c = 100;

% iteration controls
trel = 1e-10;
tabs = 1e-10;
k_max = 20

iters = [];
end_r = [];

% m = log_2(n_x)
% perform iteration and record results for each case
for m = 3:8
    n = 2^m; % number of points

    % problem domain
    x1 = linspace(0, 1, n);
    x2 = linspace(0, 1, n);
    [X1,  X2] = meshgrid(x1, x2);

    % width between points
    h = 1/n;

    % discretize differential operator, 1d
    v = ones(n, 1)/h^2;
    lapl_1d = spdiags([v -2*v v], -1:1, n, n);
    I = speye(n);
    %discrete laplacian in 2d
    lapl_2d = kron(lapl_1d, I) + kron(I, lapl_1d);
    A = -kap*lapl_2d;

```

```
    % initial guess, zeros
    v0 = zeros(n^2, 1);
    v_iter = v0;
    r = norm(func_eval(A, v0, c, X1, X2, n), 2);
    r0 = r;

    residuals = r;
    tol = trel*r + tabs;


    F  = func_eval(A, v_iter, c, X1, X2, n);
    k = 1;

    %figure()
    while r > tol && k < k_max
        k = k + 1;
        J = jacobian(A, c, v_iter, n);

        s = -J\F;
        v_iter = v_iter + s;
        F  = func_eval(A, v_iter, c, X1, X2, n);

        r = norm(F, 2);
        residuals = [residuals r/r0];
        %surf(x1, x2, reshape(v_prev, n, n), LineStyle="none")
        %zlim(gca, [-1, 3.5])
        %pause(0.06)
    end
    end_r = [end_r r/r0];
    iters = [iters k];
end

%%
figure()

yyaxis("left")
scatter(0:length(residuals)-1, residuals, "filled")
hold on
scatter(1:length(residuals)-1, residuals(1:end-1).^2, "blue")
set(gca, 'YScale', 'log')
ylabel("$$|r_k|$$", "interpreter", "latex", FontSize=13.5)

yyaxis("right")
scatter(1:length(residuals)-1,residuals(2:end)./(residuals(1:end-1).^2))
set(gca, "YScale", "log")
```

```
ylabel("$$||r_k|| / ||r_{k-1}||^2$$", ...
    Interpreter="latex", FontSize=13.5)
legend("$$r_k$$", "$$r_{k-1}^2$$", "$$\hat{c}$$", ...
    "interpreter", "latex", fontsize=11)

title("Relative residuals")
xlabel("Iteration index")
saveas(gcf, sprintf("c%i_res_m%i.png", c, m))

function phi = nonlin(U)
    phi = U.^3;
end

function dphi = dnonlin(U)
    dphi = 3*U.^2;
end

function f = src(x1, x2, n)
    f = -2*pi^2*(cos(2*pi*x1).*sin(pi*x2).^2 ...
        + sin(pi*x1).^2.*cos(2*pi*x2));
    f = reshape(f, n^2, 1);
end

function F = func_eval(A, u, c, X1, X2, n)
    F = (A*u) + (c* nonlin(u)) - src(X1, X2, n);
end

function J = jacobian(A, c, u, n)
    J = A + c*spdiags(dnonlin(u), 0, n^2, n^2);
end
```