AMME2000 - Assingment 1, 2024

Riley Smith

March 2024

Contents

List of Figures

1 Analytical Solution

1.1 Boundary Conditions and Initial Conditions

Below are the initial condition and boundary conditions of the problem stated mathematically.

$$C(x,0) = 0 \tag{1}$$

$$C(0,t) = 1 \tag{2}$$

$$C(L,t) = 0 \tag{3}$$

Some parameters were also given in the Introduction to the problem:

$$L = 2000m$$

$$D = 5 * 10^{-1} m^2/s$$

1.2 Non-Homogeneous Problem

The solution to $C(x,t)$ will comprise of a steady-state solution $C_s(x)$ and a homogeneous solution $C_H(x,t)$ due to its boundary conditions. The boundary conditions as seen in equations 2 and 3, are non-homogeneous. This is because $C(0,t)$ is not equal to zero, meaning there is a constant CO mass fraction at $x = 0$. This means that to solve for $C(x,t)$ there must be a steady-state solution $C_s(x)$ and a homogeneous solution $C_H(x,t)$.

$$C(x,t) = C_s(x) + C_H(x,t) \tag{4}$$

1.3 Steady State Solution

The diffusion equation is as follows:

$$\frac{\partial}{\partial t} C(x,t) = D \frac{\partial^2}{\partial x^2} C(x,t) \tag{5}$$

At a steady state, $\frac{\partial}{\partial t} C(x,t) = 0$, so the equation simplifies to:

$$D \frac{\partial^2}{\partial x^2} C_s(x,t) = 0 \tag{6}$$

$$\frac{\partial^2}{\partial x^2} C_s(x,t) = 0 \tag{7}$$

Integrating this equation gives:

$$\frac{\partial}{\partial x} C_s(x,t) = A \tag{8}$$

Integrating this equation again gives the general solution to the steady-state:

$$C_s(x,t) = Ax + B \tag{9}$$

$$C_s(x) = Ax + B \tag{10}$$

We can now use the boundary conditions to derive values for both A and B.

$$C_s(0) = A(0) + B = 1 \tag{11}$$

$$B = 1 \tag{12}$$

$$C_s(L) = A(L) + 1 = 0 \tag{13}$$

$$A = -\frac{1}{L} \tag{14}$$

Thus giving us the steady-state solution for this problem.

$$C_s(x) = -\frac{1}{L}(x) + 1 \tag{15}$$

A sketch of this solution can be seen in Figure 1

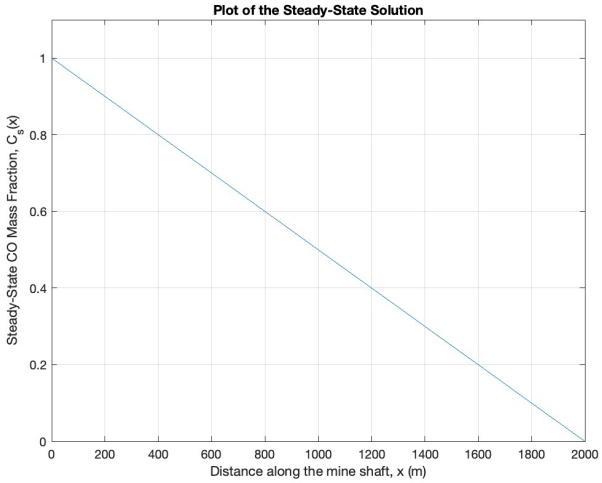

Figure 1: Sketch of the Steady State Solution

1.4 Modified Initial Condition

The modified initial condition for the homogeneous problem is derived from the equations as seen below:

$$C(x,t) = C_H(x,t) + C_s(x) \tag{16}$$

$$C(x,0) = C_H(x,0) + C_s(x) \tag{17}$$

Rearranging for $C_H(x,0)$ and substituting in the known values derives the value for the modified initial condition for the homogeneous problem.

$$C_H(x,0) = C(x,0) - C_s(x) = 0 - (-\frac{x}{L} + 1) \tag{18}$$

$$C_H(x,0) = \frac{x}{L} - 1 \tag{19}$$

A plot of this modified initial condition can be seen in Figure 2 over the domain $[0, L]$.

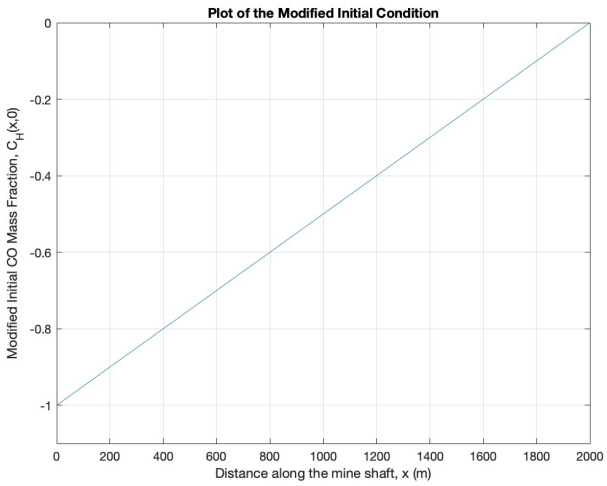

Figure 2: Sketch of the Modified Initial Condition

1.5 Separation of Variables and General Solution

1.5.1 Spatial ODE

The Spatial ODE is given in the form of a second-order linear homogeneous differential equation.

$$F_{xx}(x) + p^2 F(x) = 0 \tag{20}$$

The general solution to this differential equation is:

$$F(x) = aCos(px) + bSin(px) \tag{21}$$

We can then substitute the modified boundary conditions to solve for $F(x)$. By definition, the boundary conditions for the homogeneous problem are zero.

$$F(0) = a(1) + b(0) = 0,\ a = 0 \tag{22}$$

$$F(L) = bSin(pL) = 0 \tag{23}$$

b cannot be equal to 0 as that would mean $F(x) = 0$, thus the $Sin(pL)$ term must equal to 0.

$$b \neq 0 \quad pL = n\pi \tag{24}$$

For this to be true $pL = n\pi$ where $n = 1, 2, 3, ...$ as $Sin(n\pi)$ is always equal to 0. Rearranging for p gives:

$$p = \frac{n\pi}{L} \tag{25}$$

Plugging this into $F(x)$ gives the Spatial ODE (b will be found later):

$$F_n(x) = bSin(\frac{n\pi x}{L}) \tag{26}$$

1.5.2 Temporal ODE

The Spatial ODE is given in the form of a first-order linear homogeneous differential equation.

$$G_t(t) + Dp^2G(t) = 0 \tag{27}$$

The general solution to this differential equation is:

$$G(t) = Be^{-at} \tag{28}$$

$$a = Dp^2 \tag{29}$$

Plugging the value for a into the equation gives the Temporal ODE:

$$G_n(t) = B_n e^{-Dp^2 t} \tag{30}$$

1.5.3 General Solution for Homogeneous Problem

The general solution for the homogeneous problem is given by:

$$C_H(x,t) = F_n(x)G_n(t) \tag{31}$$

Substituting $F_n(x)$ and $G_n(t)$ into the above equation gives:

$$C_H(x,t) = bSin(\frac{n\pi}{L})B_n e^{-Dp^2 t} \tag{32}$$

$$C_H(x,t) = \sum_{n=1}^{\infty} B_n Sin(\frac{n\pi}{L})e^{-Dp^2 t} \tag{33}$$

Where constant b is absorbed into B_n, this is possible as they are both unknown coefficients. We will keep p in the equation, instead of substituting in $p = \frac{n\pi}{L}$.

1.6 Solving for B_n

To calculate B_n we can use a general Fourier Sine series formula where $f(x)$ is the modified initial condition:

$$B_n = \frac{2}{L}\int_0^L f(x)Sin(\frac{n\pi x}{L})dx \tag{34}$$

$$B_n = \frac{2}{L}\int_0^L (\frac{x}{L}-1)Sin(\frac{n\pi x}{L})dx \tag{35}$$

We will solve this integral using integration by parts:

$$\int u * dv = uv - \int v * du \tag{36}$$

$$u = \frac{x}{L} - 1 \quad dv = Sin(\frac{n\pi x}{L})dx \tag{37}$$

$$du = \frac{dx}{L} \quad v = -\frac{L}{n\pi}Cos(\frac{n\pi x}{L}) \tag{38}$$

This gives us the integral equation for which we can solve B_n:

$$B_n = \frac{2}{L}\left[(\frac{x}{L}-1)(-\frac{L}{n\pi}Cos(\frac{n\pi x}{L}))\Big|_0^L - \int_0^L -\frac{L}{n\pi}Cos(\frac{n\pi x}{L})\frac{dx}{L}\right] \tag{39}$$

$$B_n = \frac{2}{L}\left[-\frac{L}{n\pi} + \frac{1}{n\pi}\int_0^L Cos(\frac{n\pi x}{L})dx\right] \tag{40}$$

$$B_n = \frac{2}{L}\left[-\frac{L}{n\pi} + \frac{1}{n\pi}\left(\frac{LSin(\frac{n\pi x}{L})}{n\pi}\Big|_0^L\right)\right] \tag{41}$$

$$B_n = \frac{2}{L}\left[-\frac{L}{n\pi}\right] \tag{42}$$

$$B_n = -\frac{2}{n\pi} \tag{43}$$

Now that we have B_n we can state the complete the homogeneous problem:

$$C_H(x,t) = \sum_{n=1}^{\infty} -\frac{2}{n\pi}Sin(\frac{n\pi}{L})e^{-Dp^2t} \tag{44}$$

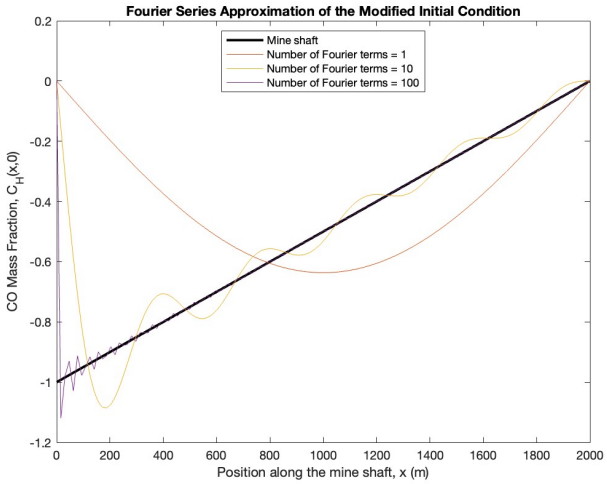

Figure 3: Sketch of Fourier Series Approximations of the Modified Initial Condition

1.7 MATLAB Homogeneous Solution Implementation

The code from the implementation of the homogeneous solution can be found in the Appendix in section A.1.1. The plot obtained from the implementation can be seen in Figure 3.

As the number of summed eigenfunctions (terms in the Fourier Series) increases, the approximation of the modified initial condition by the Fourier series increases in accuracy. This can be seen in Figure 3. As the number of Fourier Series terms increases, the approximation converges with the exact value faster.

In Figure 3, when the number of Fourier terms is equal to 1, the approximation is not accurate at all. Whereas when the number of Fourier terms is 100, the approximation is quite accurate, converging quickly on the exact value.

1.8 Final Analytical Solution

The final analytical solution $C(x,t)$, is given by summing $C_H(x,t)$ and $C_s(x)$.

$$C(x,t) = C_s(x) + C_H(x,t) = -\frac{1}{L}x + 1 + \sum_{n=1}^{\infty} B_n Sin(\frac{n\pi}{L})e^{-Dp^2 t} \tag{45}$$

$$C(x,t) = -\frac{x}{2000} + 1 + \sum_{n=1}^{\infty} B_n Sin(\frac{n\pi}{2000})e^{-D\frac{n^2\pi^2}{2000^2}t} \tag{46}$$

2 Numerical Solution

2.1 FTCS Numerical Scheme

A FTCS scheme can be derived from the governing equation ($\frac{\partial}{\partial t}C(x,t) = D\frac{\partial^2}{\partial x^2}C(x,t)$) using finite difference equations.

2.1.1 Forward-in-Time approximation for Time Derivative

The Forward-in-Time approximation for the time derivative can be given by the forward finite difference equation.

$$C_t\Big|_i^n \approx \frac{C_i^{n+1} - C_i^n}{\Delta t} \tag{47}$$

2.1.2 Central-in-Space approximation for Spatial Derivative

As the spatial derivative is a double derivative, there are extra steps to solving the approximation.

$$C_x\Big|_{i-\frac{1}{2}}^n \approx \frac{C_i^n - C_{i-1}^n}{\Delta x} \qquad C_x\Big|_{i+\frac{1}{2}}^n \approx \frac{C_{i+1}^n - C_i^n}{\Delta x} \tag{48}$$

$$C_{xx}\Big|_i^n \approx \frac{C_x\Big|_{i+\frac{1}{2}}^n - C_x\Big|_{i-\frac{1}{2}}^n}{\Delta x} \approx \frac{(C_{i+1}^n - C_i^n) - (C_i^n - C_{i-1}^n)}{(\Delta x)^2} \tag{49}$$

$$C_{xx}\Big|_i^n \approx \frac{C_{i+1}^n - 2C_i^n + C_{i-1}^n}{(\Delta x)^2} \tag{50}$$

2.1.3 FTCS Approximation Equation

We can then combine these equations to make C_i^{n+1} the subject, forming an explicit FTCS approximation equation.

$$\frac{C_i^{n+1} - C_i^n}{\Delta t} = D\frac{C_{i+1}^n - 2C_i^n + C_{i-1}^n}{(\Delta x)^2} \tag{51}$$

$$C_i^{n+1} = \sigma C_{i-1}^n + (1-2\sigma)C_i^n + \sigma C_{i+1}^n \qquad \sigma = \frac{D\Delta t}{(\Delta x)^2} \tag{52}$$

2.2 Explicit FTCS

The FTCS numerical scheme is called an 'explicit' scheme because it calculates the future state of a system, in this case concentration, explicitly in terms of its current state. That is, the value of the function at a future time is calculated using the known values at the current time.

2.3 MATLAB Numerical Solution Implementation

In Figure 4, Numerical Solutions with different values of n_x are overlaid with the Analytical Solution. The plot demonstrates that the precision of the numerical solutions improves as n_x increases, aligning more closely with the analytical solution.

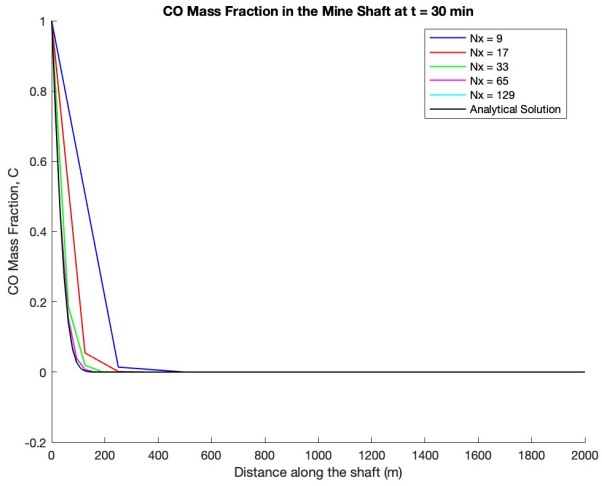

Figure 4: Comparison of the Analytical Solution with Numerical Solutions for Various n_x values

2.4 Order of Accuracy and Error Norms

The calculated error norms and order of accuracies can be seen in Table 1, the MATLAB code that generated these values can be found in A.2.2. We expect a convergence towards 2 for the Order of Accuracy values. This is because the FTCS Numerical scheme used is second-order accurate in space. The convergence towards 2 as n_x increases, indicates that the accuracy of the solved equation increases with n_x.

n_x	L^1	L^2	L^∞	$\mathcal{O}(L^1)$	$\mathcal{O}(L^2)$	$\mathcal{O}(L^\infty)$
9	$1.799 * 10^1$	$1.007 * 10^0$	$6.316 * 10^{-2}$	-	-	-
17	$4.689 * 10^0$	$2.668 * 10^{-1}$	$2.139 * 10^{-2}$	1.940	1.917	1.562
33	$1.302 * 10^0$	$7.233 * 10^{-2}$	$5.787 * 10^{-3}$	1.849	1.883	1.886
65	$3.142 * 10^{-1}$	$1.763 * 10^{-3}$	$1.415 * 10^{-3}$	2.051	2.036	2.032

Table 1: Error Norms and Order of Accuracies

2.5 Change in CO Mass Fraction over Time

In Figure 5, sketches of the Numerical Solution at different times can be seen. The code that sketched these results can be found in A.2.3. As time increases, the length of the mineshaft with a CO mass fraction greater than 0 increases. However, the gradient of the CO mass fraction along the mineshaft from $x =$ 0m decreases, indicating that the CO concentration spreads more evenly over time. This gradual equalisation results from the diffusion process, where CO spreads from high to low concentration areas, reducing the rate of change in concentration along the mineshaft and leading towards a more uniform CO distribution.

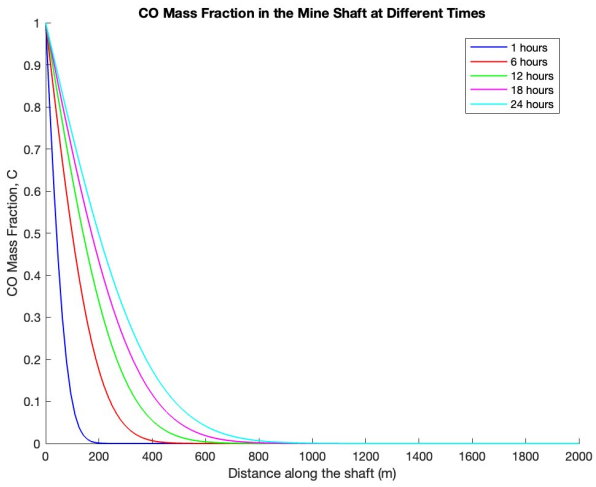

Figure 5: Sketch of the Numerical Solution with changes in time

2.6 Survival of Miners with CO Alarm

The Miners are initially at $x = 750m$ with an alarm at $x = 1000m$ that will sound when the CO mass fraction $C(x,t)$ reaches 0.001. The times at which the CO mass fraction reaches 0.001 can be found using the MATLAB code found in A.2.4. The time at which $C(750,t) = 0.001$ is 14.42hrs (51900s). The time at which the alarm sounds ($C(1000,t) = 0.001$) is 25.63hrs (92280s). This means that by the time the CO alarm sounds, the Miners have already been exposed to a CO mass fraction greater than 0.001 for 11.21hrs (40380s). The Miners will die if exposed to a CO mass fraction of 0.001 for greater than 12hrs. Once the alarm sounds, the Miners evacuate at a speed of $360m/hr$. The Miners will have a total of 0.79hrs (2844s) to travel to a point in the mineshaft where the CO mass fraction is less than 0.001, otherwise they will die. The distance they travel during this time is 284.4m, giving an x-value of 1034.4 ($C(1034.4m, 95112s)$. At

this value of x the mass fraction of CO reaches 0.001 at 27.27hrs (98160s). The miners pass this point on the mine shaft before it reaches a CO mass fraction of 0.001, which means that the miners proceed to the top of the mine shaft. At no point do the Miners experience a CO mass fraction of 0.02. The Miners successfully exit the mine without dying.

3 Numerical Solution Extended

3.1 Implementation of Source Term in Numerical Solution

The MATLAB lines of code that specifically implement and define the source term in the numerical solution can be found below.

```
% Generate the spatial grid
x = linspace(0, L, Nx);
% Discretize the source term f(x,t) = -0.001e
    ^(-0.01x^(1.5))
f = -0.001 * exp(-0.01 * x.^1.5);
...
% FTCS scheme with source term
c_new(i) = c_numerical(i) + sigma * (c_numerical(i
    -1) - 2 * c_numerical(i) + c_numerical(i+1)) +
    dt * f(i);
```

As the source term decreases exponentially with distance (due to $e^{-0.01x^{1.5}}$, the impact it has on the CO mass fraction will be the largest closest to $x = 0$. This impact decreases as the x-value increases.

3.2 Comparison of Ventilated and Unventilated Mineshaft

In Figure 6, there is a clear relationship between ventilation and safety in the mineshaft. As seen in the plot, the ventilated sketch reaches lower CO mass fractions faster compared to the unventilated plot. This increases the safety of the mine as the Miners are exposed to less CO for longer, allowing more time for them to evacuate should they need to. The code that sketched Figure 6 can be found in A.3.1.

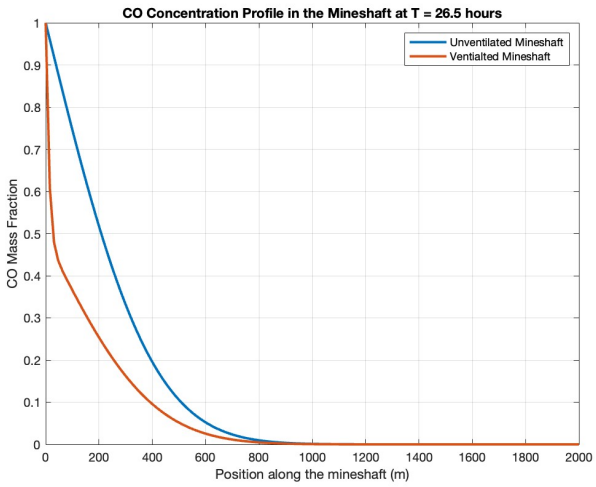

Figure 6: Comparison of Ventilated and Unventilated Mineshaft at $t = 26.5$hrs

A MATLAB Code

A.1 Analytical Solution

A.1.1 MATLAB Homogeneous Solution Implementation

```
% Parameters
L = 2000; % Length of the mine shaft in meters
D = 5e-1; % Diffusivity of CO in m^2/s
t = 0; % t=0 as specified in Q1.7
nfArray = [1, 10, 100]; % Three different nf values

% Spatial domain (129 Spatial grid points)
x = linspace(0, L, 129);

% Modified initial condition
Y_initial = @(x) x/L - 1;

% Plot the initial condition
figure;
plot(x, Y_initial(x), 'k', 'LineWidth', 2, '
    DisplayName', 'Mine shaft');
hold on;

for nf = nfArray
    % Fourier series approximation
    Y_approx = zeros(size(x));
    for n = 1:nf
        % Calculating Bn
        Bn = -2 / (n * pi);
        % Fourier series term contribution at tEnd
        Y_approx = Y_approx + Bn * sin(n * pi * x / L)
            * exp(-D * (n * pi / L)^2 * t);
    end

    plot(x, Y_approx, 'DisplayName', sprintf('Number
        of Fourier terms = %d', nf));
end

% Customize the plot
title('Fourier Series Approximation of the Modified
    Initial Condition');
xlabel('Position along the mine shaft, x (m)');
ylabel('CO Mass Fraction, C_H(x,0)');
legend('show', 'Location', 'north');
```

A.2 Numerical Solution

A.2.1 MATLAB Implemented Numerical Solution

```
function C = analyticalSolution(x, t, L, D, N_terms)
    C = 0; % Initialize C
    SteadyState = -(x / L) + 1; % Steady-state
       solution
    for n = 1:N_terms
        C = C + (-2 / (n * pi)) * sin(n * pi * x / L)
           * exp(-D * (n * pi / L)^2 * t);
    end
    C = C + SteadyState; % Add Steady State to the
       final solution
end

function c_numerical = numericalSolutionFTCS(Nx, dt,
   dx, D, t_final)
    % Solution vector
    c_numerical = zeros(Nx, 1);
    % CO mass fraction fixed at 1.0 at x = 0
    c_numerical(1) = 1;
    % Coefficient for the FTCS scheme
    sigma = D * dt / dx^2;
    % Time-stepping loop
    for t = dt:dt:t_final
        % Initialize a temporary vector for the new
           time step
        c_new = c_numerical;
        % Loop over space
        for i = 2:Nx-1
            % FTCS scheme
            c_new(i) = sigma*c_numerical(i-1) + (1-2*
               sigma)*c_numerical(i) + sigma*
               c_numerical(i+1);
        end

        % Boundary conditions
        c_new(1) = 1;        % CO mass fraction fixed
           at 1.0 at x = 0
        c_new(end) = 0;      % CO mass fraction fixed
           at 0 at x = L

        % Update the concentration profile
        c_numerical = c_new;
    end
```

```
end

% Constants and Parameters
L = 2000; % Length of the domain in meters
D = 5e-1; % Diffusivity of CO in m^2/s
dt = 60; % Time step in seconds
t_final = 30 * 60; % Final time (30 minutes converted
    to seconds)
Nx_values = [9, 17, 33, 65, 129]; % Different choices
    of spatial grid points
N_terms = 500; % Number of Fourier terms for
    analytical solution

hold on;

% Plot numerical solutions for different Nx values
for i = 1:length(Nx_values)
    Nx = Nx_values(i);
    dx = L / (Nx - 1);
    x_numerical = linspace(0, L, Nx);
    c_numerical = numericalSolutionFTCS(Nx, dt, dx, D,
        t_final);
    plot(x_numerical, c_numerical, colors(i), '
       LineWidth', 1);
end

% Plot the analytical solution
x_analytical = linspace(0, L, 129);
C_analytical = analyticalSolution(x_analytical,
    t_final, L, D, N_terms);
plot(x_analytical, C_analytical, 'k-', 'LineWidth', 1)
    ;

% Formatting the plot
title('CO Mass Fraction in the Mine Shaft at t = 30
    min');
xlabel('Distance along the shaft (m)');
ylabel('CO Mass Fraction');
legend('Nx = 9', 'Nx = 17', 'Nx = 33', 'Nx = 65', 'Nx
    = 129', 'Analytical Solution', 'Location', 'best');
hold off;
```

A.2.2 Order of Accuracy and Error Norms

```
% Constants and Parameters
```

```
L = 2000; % Length of the domain in meters
D = 5e-1; % Diffusivity of CO in m^2/s
dt = 60; % Time step in seconds
t_final = 5 * 60 * 60; % Final time (5 hours converted
    to seconds)
Nx_values = [9, 17, 33, 65]; % Different choices of
   spatial grid points, excluding 129
N_terms = 500; % Number of Fourier terms


% Main loop for error norms
for i = 1:length(Nx_values)
    Nx = Nx_values(i);
    dx = L / (Nx - 1);
    x_numerical = linspace(0, L, Nx);

    c_numerical = numericalSolutionFTCS(Nx, dt, dx, D,
        t_final);

    % Calculate analytical solution for each point in
       x_numerical at t_final
    C_analytical = zeros(Nx, 1); % Initialize
    for j = 1:Nx
        C_analytical(j) = analyticalSolution(
           x_numerical(j), t_final, L, D, N_terms);
    end

    % Calculate error norms using the provided
       definition and MATLAB's norm function
    error_norms_L1(i) = norm(c_numerical(:) -
       C_analytical(:), 1) * dx;
    error_norms_L2(i) = norm(c_numerical(:) -
       C_analytical(:), 2) * sqrt(dx);
    error_norms_LInf(i) = norm(c_numerical(:) -
       C_analytical(:), Inf);
end

% Compute orders of accuracy, starting from the second
    entry
for i = 2:length(Nx_values)
    dx_i = L / (Nx_values(i-1) - 1);
    dx_ip1 = L / (Nx_values(i) - 1);

    orders_L1(i-1) = log(error_norms_L1(i-1) /
       error_norms_L1(i)) / log(dx_i / dx_ip1);
    orders_L2(i-1) = log(error_norms_L2(i-1) /
```

```
        error_norms_L2(i)) / log(dx_i / dx_ip1);
    orders_LInf(i-1) = log(error_norms_LInf(i-1) /
        error_norms_LInf(i)) / log(dx_i / dx_ip1);
end
% Display the results, including the calculated orders
    of accuracy
disp('Nx     L1 Norm         L2 Norm         L(
   infinity) Norm        Order L1     Order L2
   Order L(infinity)');
for i = 1:length(Nx_values)
    if i == 1
        % No order of accuracy for the first entry
        fprintf('%-7d %-15.5e %-15.5e %-15.5e\n', ...
            Nx_values(i), error_norms_L1(i),
                error_norms_L2(i), error_norms_LInf(i))
                ;
    else
        % Display the calculated orders of accuracy
            for Nx_values except for 129
        fprintf('%-7d %-15.5e %-15.5e %-15.5e %-12.5f
            %-12.5f %-12.5f\n', ...
            Nx_values(i), error_norms_L1(i),
                error_norms_L2(i), error_norms_LInf(i),
                 ...
            orders_L1(i-1), orders_L2(i-1),
                orders_LInf(i-1));
    end
end
```

A.2.3 Change in CO Mass Fraction over Time

```
    % Constants and Parameters
L = 2000; % Length of the domain in meters
D = 5e-1; % Diffusivity of CO in m^2/s
dt = 60; % Time step in seconds
time_hours = [1, 6, 12, 18, 24]; % Time points in
   hours to plot
Nx = 129; % Choice of spatial grid points as per
   question 2.5
dx = L / (Nx - 1);
x_numerical = linspace(0, L, Nx);

hold on;

% Loop over specified time points
```

```
for i = 1:length(time_hours)
    t_final = time_hours(i) * 3600; % Convert hours to
        seconds
    c_numerical = numericalSolutionFTCS(Nx, dt, dx, D,
        t_final);
    plot(x_numerical, c_numerical, colors(i), '
       LineWidth', 1, 'DisplayName', sprintf('%d hours
       ', time_hours(i)));
end

% Formatting the plot
title('CO Mass Fraction in the Mine Shaft at Different
    Times');
xlabel('Distance along the shaft (m)');
ylabel('CO Mass Fraction, C');
legend('Location', 'best');
hold off;
```

A.2.4 Survival of Miners with CO Alarm

```
% Constants and Parameters
L = 2000; % Domain length in meters
D = 5e-1; % Diffusivity of CO in m^2/s
dt = 60; % Time step in seconds
Nx = 129; % Spatial grid points
dx = L / (Nx - 1);
t = 0; % Start time
target_x = %Target position in meters;
target_C = 0.001; % Target CO concentration

% Find the index for target_x
x_numerical = linspace(0, L, Nx);
target_index = round(target_x / dx) + 1;

% Simulation loop
while true
    t = t + dt; % Increment time
    c_numerical = numericalSolutionFTCS(Nx, dt, dx, D,
        t);

    if c_numerical(target_index) >= target_C
        break; % Condition met, exit loop
    end
end
```

```
fprintf('CO concentration reaches %4.3f at x = %4.2f m
    after %d seconds (%.2f hours).\n',target_C,
   target_x, t, t/3600);
```

A.3 Numerical Solution Extended

A.3.1 Comparison of Ventilated and Unventilated Mineshaft

```
function c_numerical =
    ventilated_numericalSolutionFTCS(Nx, dt, dx, D,
     t_final,x)
% Initialize the concentration profile
c_numerical = zeros(Nx, 1); % Solution vector
c_numerical(1) = 1; % CO mass fraction fixed at
    1.0 at x = 0

% Coefficient for the FTCS scheme
sigma = D * dt / dx^2;

% Discretize the source term f(x,t) = -0.001e
    ^(-0.01x^(1.5))
f = -0.001 * exp(-0.01 * x.^1.5);

% Time-stepping loop
for t = dt:dt:t_final
    % Initialize a temporary vector for the new
        time step
    c_new = c_numerical;
    % Loop over space
    for i = 2:Nx-1
        % FTCS scheme with source term
        c_new(i) = c_numerical(i) + sigma * (
            c_numerical(i-1) - 2 * c_numerical(i) +
             c_numerical(i+1)) + dt * f(i);
    end

    % Boundary conditions
    c_new(1) = 1; % CO mass fraction fixed at 1.0
        at x = 0
    c_new(end) = 0; % CO mass fraction fixed at 0
        at x = L

    % Update the concentration profile
    c_numerical = c_new;
end
```

```
end

% Parameters
Nx = 129; % Number of spatial grid points
L = 2000; % Length of the mineshaft in meters
dx = L / (Nx - 1); % Spatial step size
D = 5e-1; % Diffusivity of CO in m^2/s
dt = 60; % Time step in seconds
t_final = 26.5 * 3600; % Final time in seconds (26.5
    hours converted to seconds)

% Generate the spatial grid
x = linspace(0, L, Nx);

% Call the function without source term for comparison
c_numerical = numericalSolutionFTCS(Nx, dt, dx, D,
    t_final);

% Call the function with source term
c_numerical_ventilated =
    ventilated_numericalSolutionFTCS(Nx, dt, dx, D,
    t_final,x);


% Plotting
figure;
plot(x, c_numerical, 'LineWidth', 2, 'DisplayName', '
    Unventilated Mineshaft');
hold on;
plot(x, c_numerical_ventilated, 'LineWidth', 2, '
    DisplayName', 'Ventialted Mineshaft');
hold off;

xlabel('Position along the mineshaft (m)');
ylabel('CO Mass Fraction');
title('CO Concentration Profile in the Mineshaft at T
    = 26.5 hours');
legend('show');
grid on;
```