**SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY**

**MATHEMATICAL SCIENCES DEPARTMENT**

Math 373 HQ 2 Nov 7, 2014

Do not waste time on repetitive computations until each problem you can work is set up and its solution method clearly demonstrated.

3. Set up the following Root Finding procedures:

a) Show how to use One-Point Iteration to find a root to x3 + 2x - ln(x) = 2.5.

b) Describe the criterion for convergence in a sentence or mathematically for part 3 a) above.

1. Circle the bracketing methods.

 False Position, Bisection

 Secant, One-Point Iteration, Newton’s

d) Write the Jacobian for the following set of equations.

2. Show the Gauss-Seidel method of solving the following set of linear equations:

 -x+2y+4z = 10

 x+3y+ 2z = 11

 2x+ y = 5

3. Use Newton's method to find a root of the following equation. Start at x=1. Show two iterations and the starting values for the third iteration.

 x3 - 4 = 0

4. Describe the following “fast” methods for solving a 1D USS HT problem using

1. DuFort-Frankel Method. Use the grid and label.

 b) The Saul’yev Method. Use the grid and label.

5. Below are some regularly-spaced data from a polynomial function.



1. Show the best ***second-order*** (three terms) approximation of f(3.3).
2. What is the order of the polynomial from which the data appear to have been generated? **Why?**
3. Show how to reconstruct the polynomial function that fits the data exactly assuming it is from a polynomial.

6. Circle four of the numerous errors in the boxed area**s** on the following MATLAB code. (Only the first four circles will be graded.) Make clear what the correct code is. There is no need to find errors in both boxes but you may.

% Crank Nicolson

% Define parameters

E=29; % Number of time steps

Dt=10; % time step size

alpha=0.125; % Thermal Diffusivity

Dx=1; % x step size

f=alpha\*Dt/Dx^2/2; % Constant in numerical solution

Ti=0; % initial T

Tb1=1; % BC1

Tb2=1; % BC2

I=10; % Number of x increments

% prealocate arrays

d=zeros(I-1,1);

Tn=zeros(I-1);

T=zeros(E,I+1);

%set IC

for x=2:I

 T(1,x)=Ti;

end

%Set BC's

for t=1:E ;

 T(t,1)=Tb1;

 T(t,I+1)=Tb2;

end

% Set the coef array

a=-f;

b=1+2\*f;

c=-f;

for t=1:E % start time loop

% set RHS's

for n=1:I-1

 d(n,1)=f\*T(t-1,n)+(1-2\*f)\*T(t-1,n+1)+f\*T(t-1,n+2);

end

d(1,1)=d(1,1)+f\*T(t-1,1); % Adjust first RHS for BC#1 term

d(I-1,1)=d(I-1,1)+f\*T(t-1,I+1); % Adjust last RHS for BC#2 term

% Solve

Tn = Tridiag(10,a,b,c,d); % Solve by the "Tridiag" method

for i=1:I-1 % Move the solved Tn values to the T array for plot

 Tn(t,i+1)=T(i);

end

T(t,1)=Tb1; % Write the BC’s in T array so complete T matrix is available for plot

T(t,I)=Tb2;

end

figure;

mesh(T)

%Tridiagonal Matrix Algorithm for constant a, b, and c terms.

function x=Tridiag(a,b,c,d,n)

Beta(1)=b;

Gamma(1)=d(1)/Beta(1);

for i=1:n

 Beta(i)=b-a\*c/Beta(i-1);

 Gamma(i)=d(i)-a\*Gamma(i))/Beta(i-1);

end

x(n)=Gamma(n);

for i=n-1:1

 x(i)=Gamma(i)-c\*x(i+1)/Beta(i+1);

end

**See the next page for Tridiagonal Matrix Algorithm equations.****Please detach this sheet. Do not turn it in with your exam.**

**Crank-Nicolson Method Equations for nine unknown temperatures**

 

where    

The complete enumerations for all grid points where the temperatures are unknown are

1. 
2. 
3. 
4. 
5. 
6. 
7. 
8. 
9. 

# Tridiagonal Matrix Algorithm

Crank-Nicholson has all the same a’s, b’s, and c’s and are not subscripted. The general Tridagonal Algorithm allows unique values of a, b, and c in each equation and are, therefore, subscripted.





Where the ’s and ’s are determined from the recursion formulas



Scratch Paper - **Please detach this sheet. Do not turn it in with your exam.**