

MATH 335: Numerical Analysis

Problem Set 20, Solutions

★42 (a) Here is my code:

```
function [y, t] = abm_sys(f, t0, tlast, y0, order, N)
% function [y, t] = abm_sys(f, t0, tlast, y0, order, N)
%
% This routine calculates an approximate solution at mesh points
% t0, t1, ..., tN to the initial-value problem
%
%      y' = f(t, y),    subject to y(t0) = y0,
%
% using the Adams-Bashforth-Moulton predictor-corrector of specified
% order. We assume evenly-spaced mesh points, with stepsize
%
%      h = (tlast - t0) / N.
%
% INPUTS:
%   f      function handle to the RHS in the ODE giving y'
%          It is necessary f accept inputs
%          t  scalar (time)
%          y  vector (state)
%          and output a vector of the same size and shape as y.
%   t0     initial "time"
%   tlast  last "time" at which to approximate the solution
%   y0     initial value, the value of the solution at the initial time
%   order  order of the Adams-Bashforth-Moulton method to use
%   N      number of steps to take to go from t0 to tN
%
% OUTPUTS:
%   y      matrix of approximate value of solution at mesh points
%          The jth column corresponds to the jth time.
%   t      mesh points

h = (tlast - t0) / N;
t = [t0:h:tlast]';
y = y0(:);
prev_f = [];

% Compute first few steps using RK4 method
```

```

for jj = 1:min(order-1, N)
    k1 = f(t(jj), y(:, jj));
    k2 = f(t(jj) + h/2, y(:, jj) + h*k1/2);
    k3 = f(t(jj) + h/2, y(:, jj) + h*k2/2);
    k4 = f(t(jj) + 1, y(:, jj) + h*k3);
    y(:, jj + 1) = y(:, jj) + h*(k1 + 2*k2 + 2*k3 + k4) / 6;
    prev_f = [k1 prev_f];
end
prev_f = [f(t(jj+1), y(:, jj+1)) prev_f];

[wtsAB, wtsAM] = adamsWts(order);
for jj = order:N
    y_pred = y(:, jj) + prev_f * wtsAB * h;
    prev_f = [f(t(jj+1), y_pred) prev_f(:, 1:length(wtsAM)-1)];
    y(:, jj+1) = y(:, jj) + prev_f * wtsAM * h;
    prev_f(:, 1) = f(t(jj+1), y(:, jj+1));
end
end

```

★43 (a)

```

function [y, t] = impEulerLinearODE(A, t0, tfin, y0, N)
% function [y, t] = impEulerLinearODE(A, t0, tfin, y0, N)
%
% This routine calculates an approximate solution at mesh points
% t0, t1, ..., tN to the linear initial-value problem
%
%          y' = A y,    subject to y(t0) = y0,
%
% using the implicit Euler method. We assume evenly-spaced
% mesh points, with stepsize
%
%          h = (tlast - t0) / N.
%
% INPUTS:
%   A      coefficient matrix
%   t0     initial "time"
%   tlast  last "time" at which to approximate the solution
%   y0     initial value, the value of the solution at the initial time
%   N     number of steps to take to go from t0 to tN
%
% OUTPUTS:

```

```
%      y      matrix of approximate value of solution at mesh points
%              The jth column corresponds to the jth time.
%      t      mesh points

h = (tfin - t0) / N;
t = [t0:h:tfin]';
y = y0(:);
Id = eye(size(A));
B = (Id - h*A) \ Id;
for jj = 1:N
    y = [y B*y(:,jj)];
end
end
```