Numerical Solutions to Differential Equations

Lecture Notes #15 — Example: Variable Order LMM Schemes

Peter Blomgren,
⟨blomgren.peter@gmail.com⟩

Department of Mathematics and Statistics
Dynamical Systems Group
Computational Sciences Research Center
San Diego State University
San Diego, CA 92182-7720

http://terminus.sdsu.edu/

Spring 2015

Outline

Closing out the Initial Value Problem

To finish up the discussion on numerical methods for the *initial value problem*, as we know it (so far), we will develop a variable order predictor-corrector method.

For simplicity, we will look at the following problem

$$y'(t) = f(t, y(t))$$

where

$$y(0) = 0, \quad f(t, y(t)) = 0, \quad t < T_{\text{start}} = 1.$$

This way we will not have to worry about generating starting values (the forcing turns on after a while).

Our first forcing function

For our first example we choose the following forcing function...

$$f(t, y(t)) = \begin{cases} 0 & t \in [0, 1] \\ -10\, y(t) + \sin(2\pi t))e^{-(t-1)^{-2}} & t \in (1, 8] \\ (-10\, y(t) + \sin(2\pi t))e^{-(t-1)^{-2}} e^{-(t-8)^2} & t \in (8, 12] \end{cases}$$

The Code Matlab


```
% Variable order Predictor-Corrector method
% in P-(EC)^2-E mode.
%
clear all

% The right-hand side of y'(t) = f(t,y(t))
%
f = @(t,y) ...
    ( (t>=1) * (-10*y + sin(2*pi*t)) * exp(-1/((t-1)*(t-1))) * ...
          ((t>8)*exp(-(t-8)*(t-8)) + (t<=8)) );

% Coefficients for Adams-Bashforth Predictors
%
AB = [...
      1 0 0 0 0 0; ...
      [3 -1 0 0 0 0]/2; ...
      [23 -16 5 0 0 0]/12; ...
      [55 -59 37 -9 0 0]/24; ...
      [1901 -2774 2616 -1274 251 0]/720; ...
      [4277 -7923 9982 -7298 2877 -475]/1440];
```

The Code


```
% Error Coefficients for Adams-Bashforth Predictors
%
ABE = [1/2 5/12 9/24 251/720 475/1440 19087/60480];

% Coefficients for Adams-Moulton Correctors
%
AM = [...
      1 0 0 0 0 0; ...
      [1 1 0 0 0 0]/2; ...
      [5 8 -1 0 0 0]/12; ...
      [9 19 -5 1 0 0]/24; ...
      [251 646 -264 106 -19 0]/720; ...
      [475 1427 -798 482 -173 27]/1440];

% Error Coefficients for Adams-Moulton Correctors
%
AME = [-1/2 -1/12 -1/24 -19/720 -27/1440 -863/60480];

% Currently, we cannot go beyond sixth order
%
MAX_ORDER = 6;
```

The Code


```
% Step size
%
h = 1/12;
% Starting order
%
order = 1;
% Number of corrections
%
mu = 2;
% Tolerance
%
TOL = 0.00007;
%
Tmax = 12;
tv   = 0:h:Tmax;
y    = zeros(size(tv));
fv   = zeros(size(tv));
Errv = zeros(size(tv));
Ov   = ones(size(tv));
```

The Code


```
ix = 2; while( ix <= length(tv) )
    % Extract the time
    %
    t = tv(ix-1);

    % Extract the Adams-Bashforth and Adams-Moulton Coefficients
    %
    ab = AB(order,(1:order));
    am = AM(order,(1:order));

    % Predictor Step --- P
    %
    f_history = fv((ix-1):(-1):(ix-order));
    y_pred = y(ix-1) + h * sum(ab.*f_history);

    % Evaluate+Correct, (EC)-loop
    %
    f_history = fv((ix):(-1):(ix-order+1));
    f_history(1) = f(t,y_pred);
    for nu = 1:mu
        y_corr = y(ix-1) + h * sum(am.*f_history);
        f_history(1) = f(t,y_corr);
    end
```

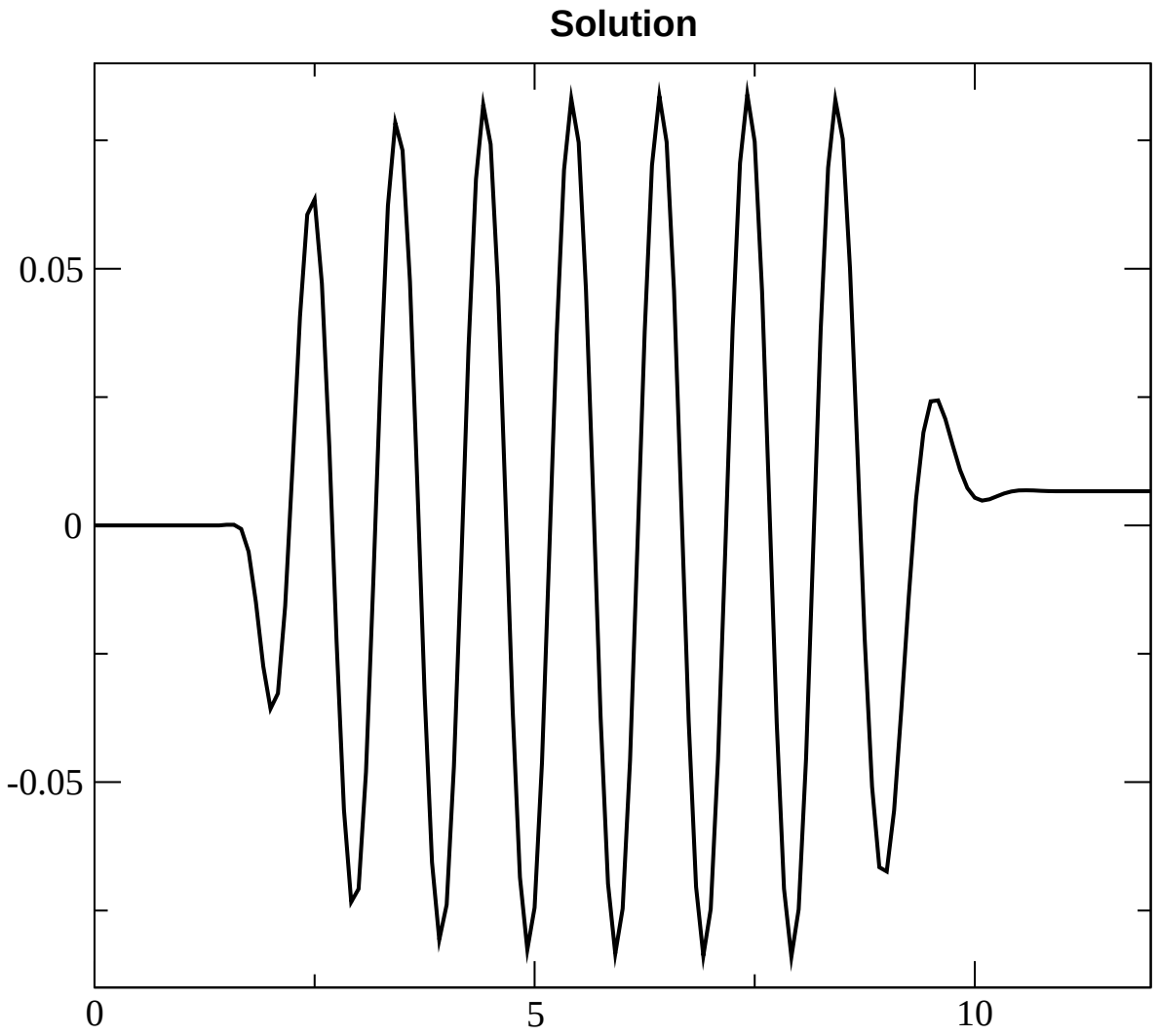
The Code


```
% Evaluate, --- E
%
f_eval = f_history(1);

% Milne's Error Estimate
%
LTE_est = AME(order)/(ABE(order)-AME(order))*(y_corr-y_pred);
Errv(ix) = LTE_est;

if( abs(LTE_est) < TOL )
    y(ix) = y_corr;
    fv(ix) = f_eval;
    Ov(ix) = order;
    ix = ix + 1;
    if( order > 1 )
      Cp = AME(order) / (ABE(order)-AME(order));
      Cpm1 = AME(order-1) / (ABE(order-1)-AME(order-1));
      if( abs(LTE_est * Cpm1 / Cp / h) < 0.5*TOL )
        order = order - 1;
        fprintf('Decreasing order to %d at time %f\n',order,t);
      end
    end
```

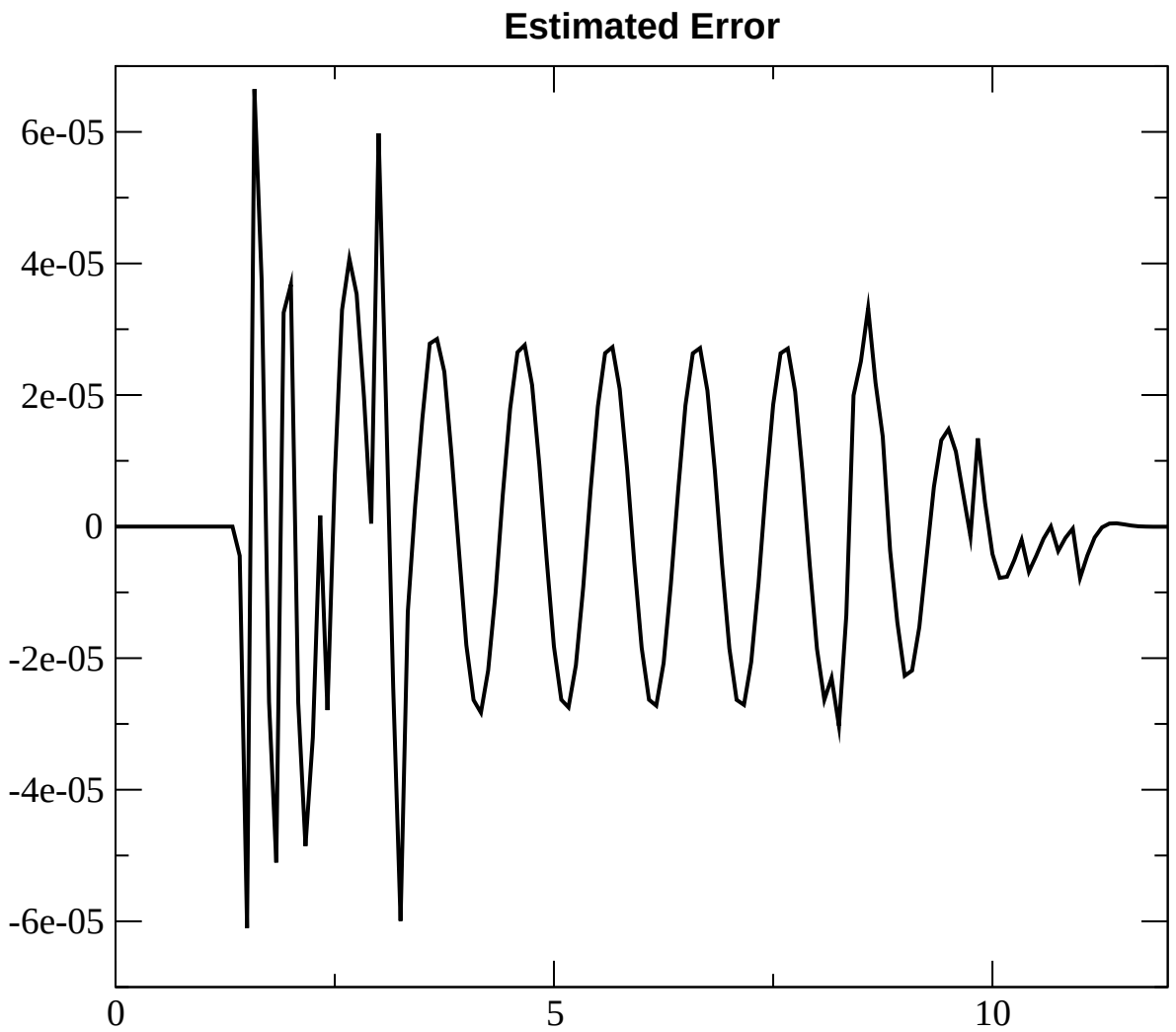
The Code


```
    else
        fv(ix) = 0;
        order = order + 1;
        if( order <= MAX_ORDER )
          fprintf('Increasing order to %d at time %f\n',order,t);
        else
          fprintf('Cannot increase order beyond %d\n',MAX_ORDER);
          fprintf('Program failed at time %f\n\n\n',t);
          fprintf('Try a smaller step-size!\n');
          error('Execution stopped');
        end
    end
end

%
% Dump data to files for plotting.
%
D = [tv' y'];      save('-ascii','var_soln.dat','D');
D = [tv' Ov'];     save('-ascii','var_order.dat','D');
D = [tv' Errv'];   save('-ascii','var_error.dat','D');
D = [tv' fv'];     save('-ascii','var_f.dat','D');
```

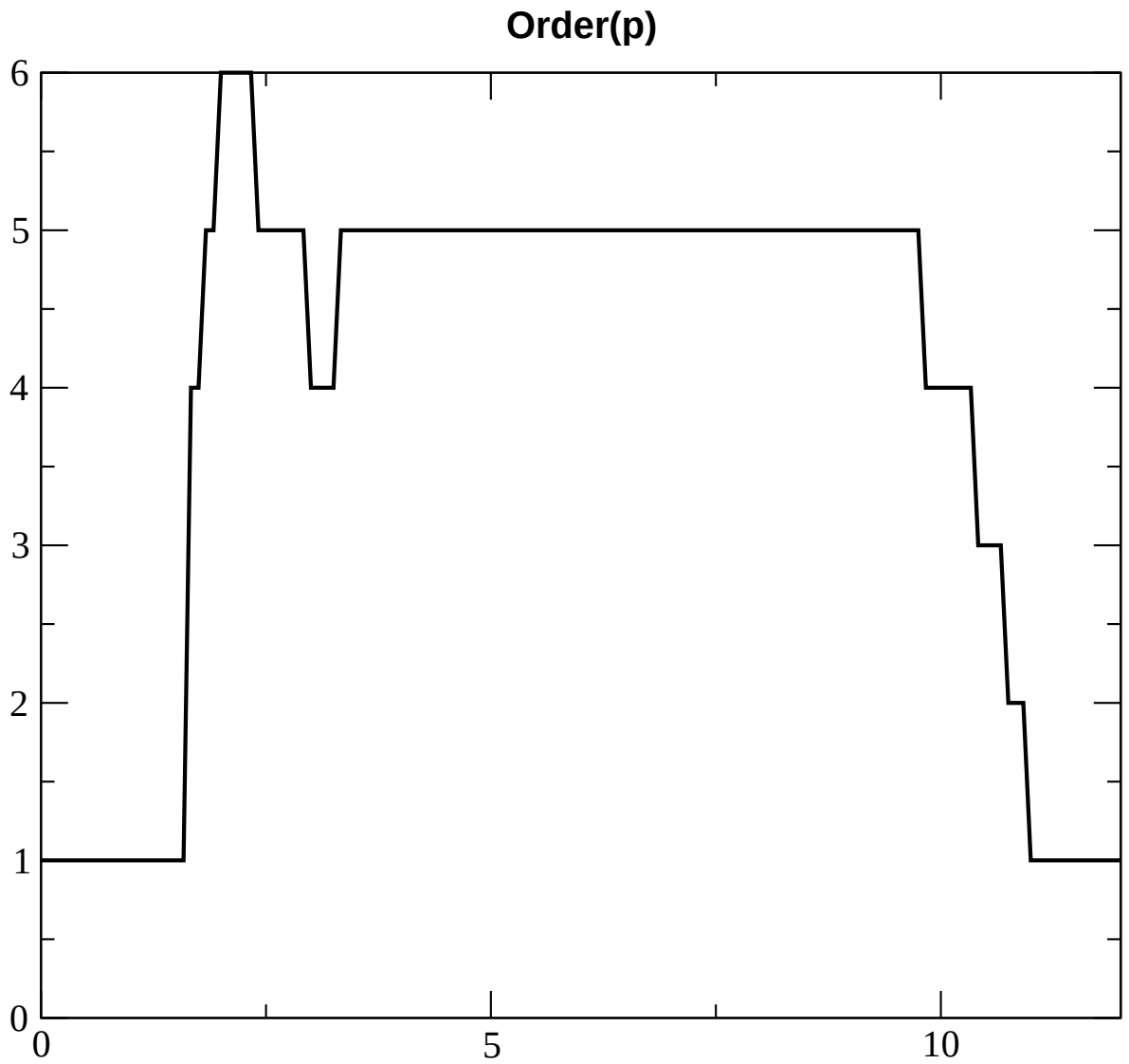
Plots of Solution, Error, Order and $f(t, y(t))$



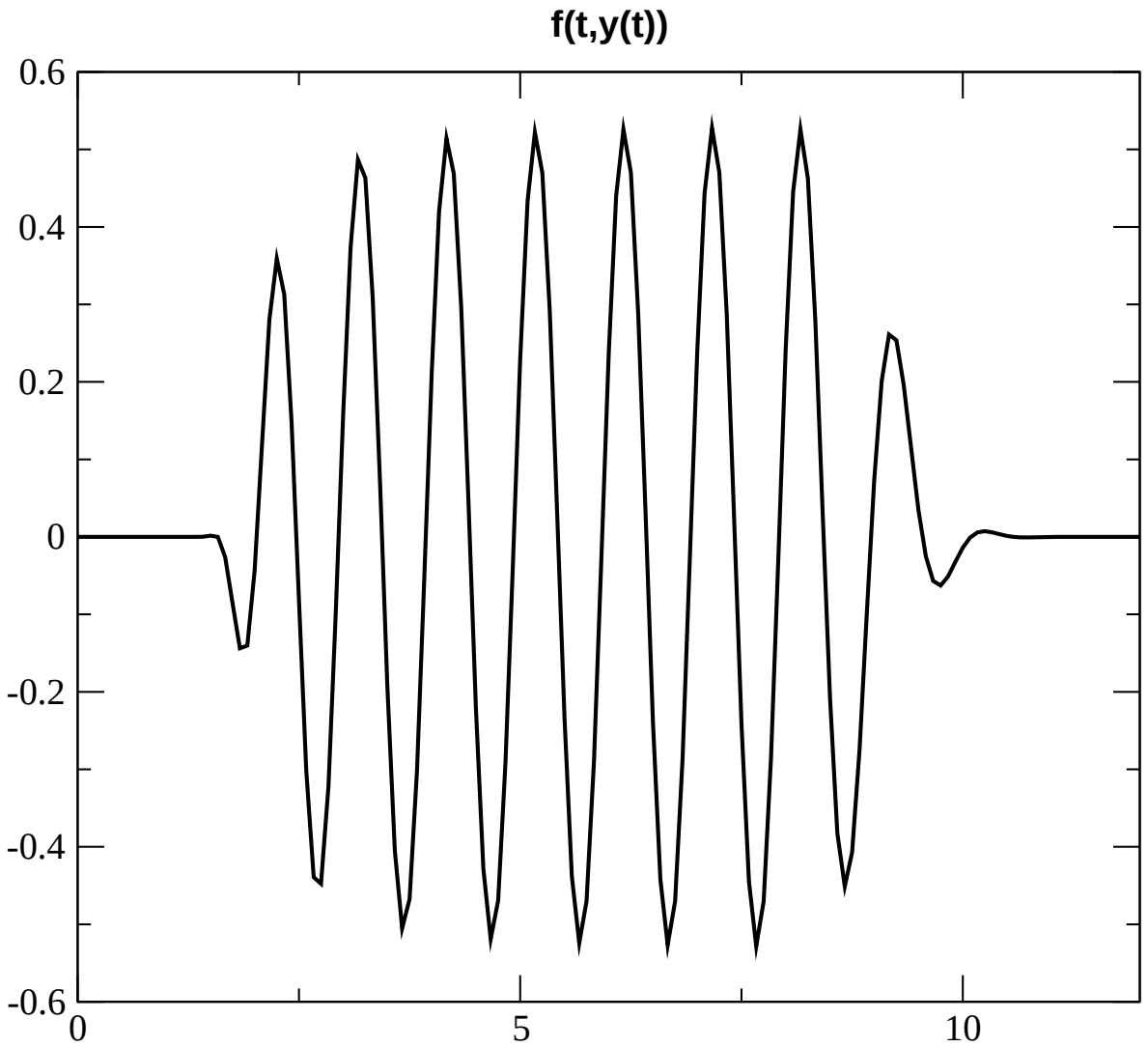
Plots of Solution, Error, Order and $f(t, y(t))$



Plots of Solution, Error, Order and $f(t, y(t))$



Plots of Solution, Error, Order and $f(t, y(t))$



Comments

The example is carefully "cooked" in order to trigger order changes up to 6, but not break.

The forcing function is "somewhat" synthetic (to put it kindly!)

We have completely ignored the stability analysis... But since there is no strange blow-up, everything seems to be OK.

Implementing a variable order BDF scheme (as described in lecture #12) may be a good final project.

Looking ahead — Boundary Value Problems

Next, we will open up a some new cans of worms —

- A discussion of hybrid methods; combining some of the ideas introduced so far.
- A look at Boundary Value Problems (BVP) for ODEs.

Final Project Presentations — Last Week(s) of Class + Final's Week

12-15-minute presentations.

[1] Background — describe the physical problem and equations.

[2] Approach — describe the numerical scheme selected and analysis performed to get a "feel" for the parameters; *e.g.* stability analysis (eigenvalues of the system, if applicable) to get a working step size h.

[3] Results — document how well your approach solves the problem (also report on approaches that went wrong!)

[4] Conclusions — comment on possible improvements which may be beyond what is reasonable for a small class project.

Do not hesitate to use office hours and email to get help!!! Ask early and ask often! — Hand in a draft of #1 and #2 by 4/10/2015.