Section 5.6 Multistep Methods

Motivation

- How to design a high-order accurate method without the need to compute intermediate (stage) values as Runge-Kutta methods.

General idea: Consider IVP: $y' = f(t, y), \quad a \le t \le b, \; y(a) = \alpha.$

Let the approximate solutions at mesh points $t_0, t_1, t_2, \dots\ t_i$ be already obtained. Since in general error $|y(t_{i+1}) - w_{i+1}|$ grows with respect to time t, it then makes sense to use more previously computed approximate solution $w_i,\ w_{i-1}, w_{i-2}, \dots$ when computing w_{i+1}.

Definition 5.14 An **m-step** multistep method for solving the IVP:

$$y' = f(t,y), \quad a \le t \le b, \; y(a) = \alpha$$

has a difference equation for computing w_{i+1} at the mesh point t_{i+1} represented by:

$$w_{i+1} = a_{m-1}w_i + a_{m-2}w_{i-1} + \cdots + a_0 w_{i+1-m}$$
$$+h[\boldsymbol{b_m} f(t_{i+1}, w_{i+1}) + b_{m-1} f(t_i, w_i)$$
$$+ \cdots + b_0 f(t_{i+1-m}, w_{i+1-m})]$$

for $i = m-1, m, \ldots, N-1$, where $h = (b-a)/N$, the a_0, $a_1, \ldots, a_{m-1}$ and $b_0, b_1, \ldots, b_m$ are constants, and the starting values

$w_0 = \alpha, w_1 = \alpha_1, \ldots, w_{m-1} = \alpha_{m-1}$ are specified.

Remark. 1. When $b_m = 0$, the method is called **explicit**;

2. When $b_m \neq 0$, the method is called **implicit**

Adams-Bashforth Two-step Explicit Method

1) $y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_{i+1}} y'(t)dt = y(t_i) + \int_{t_i}^{t_{i+1}} f(t, y(t))dt$

2) Treat f as a function of t. Do **linear** Lagrange interpolation by using $(t_i, f(t_i, y_i))$ and $(t_{i-1}, f(t_{i-1}, y_{i-1}))$:

$$f(t, y) = f(t_{i-1}, y_{i-1}) \frac{t - t_i}{-h} + f(t_i, y_i) \frac{t - t_{i-1}}{h} + O(h^2)$$

$$y(t_{i+1}) - y(t_i) = \int_{t_i}^{t_{i+1}} f\big(t, y(t)\big)dt$$

$$= \int_{t_i}^{t_{i+1}} \left[f\big(t_{i-1}, y(t_{i-1})\big) \frac{t - t_i}{-h} + f\big(t_i, y(t_i)\big) \frac{t - t_{i-1}}{h} + O(h^2) \right] dt$$

$$= -f\big(t_{i-1}, y(t_{i-1})\big) \frac{h}{2} + f\big(t_i, y(t_i)\big) \frac{3h}{2} + O(h^3)$$

Adams-Bashforth two-step *explicit* method (AB2):

$$w_0 = \alpha_0, \qquad w_1 = \alpha_1$$

$$w_{i+1} = w_i + \frac{h}{2}[3f(t_i, w_i) - f(t_{i-1}, w_{i-1})],$$

$$\text{where } i = 1, 2, \ldots N - 1.$$

Remark:

a) AB2 method has local truncation error of order *two*.

b) AB2 method needs two starting values. w_1 can be computed by a Runge-Kutta method of the same order.

Adams-Moulton Two-step Implicit Method

$$w_0 = \alpha, \qquad w_1 = \alpha_1,$$

$$w_{i+1} = w_i + \frac{h}{12}[5f(t_{i+1}, w_{i+1}) + 8f(t_i, w_i) - f(t_{i-1}, w_{i-1})],$$

$$\text{where } i = 1, 2, \ldots N-1.$$

Remark:

a) AM2 method has local truncation error of order *three*.

b) AM2 method is obtained by treating f as a function of t, and applying **quadratic** Lagrange interpolation using data points at t_{i+1}, t_i and t_{i-1}.

c) AM2 method needs two starting values. w_1 can be computed by a Runge-Kutta method of the same order.

Local Truncation Error of Multistep Methods

Definition 5.15 Local Truncation Error. If $y(t)$ solves the IVP $y' = f(t, y),\ a \le t \le b,\ \ y(a) = \alpha$ and

$$w_{i+1} = a_{m-1}w_i + a_{m-2}w_{i-1} + \cdots + a_0 w_{i+1-m}$$
$$h[b_m f(t_{i+1}, w_{i+1}) + b_{m-1} f(t_i, w_i) + \cdots$$
$$+ b_0 f(t_{i+1-m}, w_{i+1-m})],$$

the local truncation error is:

$$\tau_{i+1}(h) = \frac{y(t_{i+1}) - a_{m-1}y(t_i) - a_{m-2}y(t_{i-1}) - \cdots - a_0 y(t_{i+1-m})}{h}$$
$$-[b_m f\big(t_{i+1}, y(t_{i+1})\big) + \cdots + b_0 f(t_{i+1-m}, y(t_i))]$$

for each $i = m-1, m, \ldots, N-1$.

NOTE:

a) the local truncation error of a *m*-step *explicit* step is $O(h^m)$;

b) the local truncation error of a *m*-step *implicit* step is $O(h^{m+1})$.

explicit method vs. implicit method

1. Explicit method is cheaper to calculate *per step*. **(no equations to solve)**
2. Implicit method is more stable, but is also more expensive to compute.

- Explicit methods are usually preferred over implicit methods due to their computational efficiency, but there are some exceptions in favor of implicit methods (stiff problems).

3rd order methods

- **Adams-Bashforth Three-step Explicit Method:**

$$w_{i+1} = w_i + \frac{h}{12}[23f(t_i, w_i) - 16f(t_{i-1}, w_{i-1}) + 5f(t_{i-2}, w_{i-2})]$$

- **Adams-Moulton Two-step Implicit Method:**

$$w_{i+1} = w_i + \frac{h}{12}[5f(t_{i+1}, w_{i+1}) + 8f(t_i, w_i) - f(t_{i-1}, w_{i-1})]$$

4th order methods

- **Adams-Bashforth Four-step Explicit Method:**

$$w_{i+1} = w_i + \frac{h}{24}[55f(t_i, w_i) - 59f(t_{i-1}, w_{i-1}) + 37f(t_{i-2}, w_{i-2}) - 9f(t_{i-3}, w_{i-3})]$$

- **Adams-Moulton Three-step Implicit Method:**

$$w_{i+1} = w_i + \frac{h}{24}[9f(t_{i+1}, w_{i+1}) + 19f(t_i, w_i) - 5f(t_{i-1}, w_{i-1}) + f(t_{i-2}, w_{i-2})]$$

Predictor-Corrector Method

- *Motivation*:

Consider to solve the IVP $y' = e^y, \quad 0 \le t \le 0.25, \quad y(0) = 1$ by the two-step Adams-Moulton method.

Solution: The two-step Adams-Moulton method is

$$w_{i+1} = w_i + \frac{h}{12}\left[5e^{w_{i+1}} + 8e^{w_i} - e^{w_{i-1}}\right] \qquad Eq.(1)$$

$Eq.(1)$ can be solved (for w_{i+1}) by Newton's method. However, this can be quite computationally expensive.

- To avoid solving nonlinear equations, we can combine explicit and implicit methods to form a *predictor-corrector method*.

4th order Predictor-Corrector Method

Step 1: Use 4th order Runge-Kutta method to compute w_0, w_1, w_2 and w_3.

Step 2: For $i = 3,4,5, \ldots N$

a) Predictor sub-step. Use 4th order 4-step explicit Adams-Bashforth method to compute a predicated value $w_{i+1,p}$

$$w_{i+1,p} = w_i + \frac{h}{24}[55f(t_i, w_i) - 59f(t_{i-1}, w_{i-1}) + 37f(t_{i-2}, w_{i-2}) - 9f(t_{i-3}, w_{i-3})]$$

b) Correction sub-step. Use 4th order three-step Adams-Moulton implicit method to compute a correction w_{i+1} (the approximation at $i+1$ time step)

$$w_{i+1} = w_i + \frac{h}{24}[9f\left(t_{i+1}, w_{i+1,p}\right) + 19f(t_i, w_i) - 5f(t_{i-1}, w_{i-1}) + f(t_{i-2}, w_{i-2})]$$

The epidemics (S-I-R) model (revisit)

$$\begin{cases} \dfrac{dS(t)}{dt} = -\beta S(t)I(t) \\ \dfrac{dI(t)}{dt} = \beta\, S(t)I(t) - \gamma\, I(t) \\ \dfrac{dR(t)}{dt} = \gamma\, I(t) \end{cases}$$

- Variable for the S-I-R Model:

t	$=$	the time in days with t=0 at the start of observation
S	$=$	the number of susceptible individuals
I	$=$	the number of infectious individuals
R	$=$	the number of removed individuals

- Parameters in the S-I-R Model:

β: the daily rate of contacts per infective

γ: 1/(the average number of days infectious)

S-I-R **(epidemics)** Model is a 3-component ODE system.

Example 1. Use Adams 4th order predictor-corrector method to solve the SIR model with initial condition

$S(0) = 0.89, \; I(0) = 0.01, \;\; R(0) = 0.10,$ (add up to one)

and the following parameters:

$a) \; \beta = 0.61818, \gamma = 0.09091$ (Rubella)

$b) \; \beta = 1.875, \gamma = 0.125.$ (Measles)

$c) \; \beta = 0.46667, \gamma = 0.33333$ (Influenza)

Computing starting values using RK4 method.

Take final time $tend = 120$, take step size $h = 0.5$.

Reformulation the ODE system

- Set $S(t) = y_1(t), I(t) = y_2(t), R(t) = y_3(t).$
- ODE:

$$\begin{aligned} y_1'(t) &= -\beta y_1 y_2, \\ y_2'(t) &= \beta y_1 y_2 - \gamma y_2, \\ y_3'(t) &= \gamma y_2. \end{aligned}$$

- Or, equivalently

$$\boxed{\frac{d}{dt}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} -\beta y_1 y_2, \\ \beta y_1 y_2 - \gamma y_2, \\ \gamma y_2. \end{bmatrix}}$$

We walk through the code sir.m step-by-step

STEP 0: set-up

```
% inputs
%beta = 0.61818; gamma = 0.09091; % rubella
%beta = 1.875; gamma = 0.125; % measles
beta = 0.46667; gamma = 0.33333; % Influenza

f = @(t, y) [-beta*y(1)*y(2);...
             beta*y(1)*y(2)-gamma*y(2);...
             gamma*y(2)];
tend = 120; % final time (~4 month)
y0 = [0.89;0.01; 0.1]; % initial data
h = 0.5; % half-day step size
```

STEP 1: RK4 for w1, w2, w3

```
tGrid = [0:h:tend]; N = length(tGrid)-1;
wGrid = zeros(3,N+1); wGrid(:,1) = y0;
% RK4 for w1,w2,w3
for i = 1:3
    ti = tGrid(i); wi = wGrid(:,i);
    k1 = f(ti,wi);
    k2 = f(ti+h/2, wi+h/2*k1);
    k3 = f(ti+h/2, wi+h/2*k2);
    k4 = f(ti+h, wi+h*k3);
    wGrid(:,i+1) = wi + h/6*(k1+2*k2+2*k3+k4);
end
```

STEP 2: predictor-corrector for w4, … ,wN

```
for i = 4:N
    ti = tGrid(i); wi = wGrid(:,i);
    wi1 = wGrid(:,i-1);
    wi2 = wGrid(:,i-2);
    wi3 = wGrid(:,i-3);
    % predictor: AB4
    wip = wi + h/24*(55*f(ti,wi)...
        -59*f(ti-h, wi1)...
        +37*f(ti-2*h, wi2)...
        -9*f(ti-3*h, wi3));
    % corrector AM3
    wGrid(:,i+1) = wi + h/24*(9*f(ti+h,wip)...
        +19*f(ti, wi)...
        -5*f(ti-h, wi1)...
        +f(ti-2*h, wi2));
end
```

STEP 3: outputs and plots

```
fprintf('\n at time t=%i',tend)
fprintf('\n Susceptible = %.2f%%',  wGrid(1,N+1)*100)
fprintf('\n Infected    = %.2f%%',  wGrid(2,N+1)*100)
fprintf('\n Removed     = %.2f%%\n',wGrid(3,N+1)*100)

figure(1)
plot(tGrid, wGrid(1,:),'r')
title('Susceptibles')
set(gca, 'FontSize',18)
```

```
at time t=120
Susceptible = 53.08%
Infected    = 0.00%
Removed     = 46.92%
```