BM 593 Numerical Methods & C Programming

7th week Differential Equations and Random Numbers

Solution of Differential Equations

Ordinary Differential Equations

$dx/dt = -x$

$dx/x = -t \rightarrow \int_{x_0}^{x} dx/x = -\int_{t_0}^{t} dt \rightarrow ln(x)|_{x_0}^{x} = -(t - t_0) \rightarrow x = x_0 e^{t-t_0}$

One-Step Methods

$dy/dx = f(x, y)$

$y_{i+1} = y_i + \phi h$ $\phi : slope = dy/dx$

Euler's Method

$y_{i+1} = y_i + f(x_i, y_i)h$

Ex: $f(x, y) = -2x^3 + x^2 - 20x + 8.5$

$h = 0.5$ $x(0) = 0$ and $y(0) = 1$

$y = -x^4/2 + 4x^3 - 10x^2 + 8.5x + 1$

$y_e(0.5) = y(0) + f(0, 1)0.5$ and $y(0) = 1$

$f(0, 1) = -2(0)^3 + 12(0)^2 - 20(0) + 8.5$

$y_e(0.5) = 5.25$

$y(0.5) = -(0.5)^4/2 + 4(0.5)^3 - 10(0.5)^2 + 8.5(0.5) + 1 = 3.21$

$Error = (3.21 - 5.25)/3.21 = -63.1\%$

Two Sources of error from truncation

i) local error

ii) propagated error

Error can be reduced by decreasing step size.

Modifications of step size

Heun's Method

$y_{i+1}^{o} = y_i + f(x_i, y_i)h$

$y_{i+1}{}' = f(x_{i+1}, y_{i+1}^{o})$

$y' = [f(x_i, y_i) + f(x_{i+1}, y_{i+1}^{o})]/2$

$y' = y_i + y'h$

Runge-Kutta (RK) Methods

$y_{i+1} = y_i + \phi(x_i, y_i, h)$

$\phi = a_1 k_1 + a_2 k_2 + \ldots + a_n k_n$

$k_1 = f(x_i, y_i)$

$k_2 = f(x_i + p_1 h, y_i + q_{1,1} k_1 h)$

$k_3 = f(x_i + p_2 h, y_i + q_{2,1} k_1 h + q_{2,2} k_2 h) \vdots$

$k_n = f(x_i + p_{n-1} h, y_i + q_{n-1,1} k_1 h + q_{n-1,2} k_2 h + \ldots + q_{n-1,n-1} k_{n-1} h)$

1st order RK $\Rightarrow$ Euler's Method

2nd order RK

$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2)h$

$k_1 = f(x_i, y_i)$

$k_2 = f(x_i + p_1 h, y_i + q_{1,1} k_1 h)$

By Taylor Expansion:

$y_{i+1} = y_i + f(x_i, y_i)h + f'(x_i, y_i)h^2/2$

$f'(x, y) = \partial f/\partial x + \partial f/\partial y \; dy/dx$

$y_{i+1} = y_i + f(x_i, y_i)h + (\partial f/\partial x + \partial f/\partial y \; dy/dx)h^2/2$

$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2)h$

$\quad = y_i + a_1 f(x_i, y_i)h + a_2[f(x_i, y_i) + p_1 h \partial f/\partial x + +q_{1,1} k_1 h f \partial/\partial y \; dy/dx]h$

$a_1 + a_2 = 1$

$a_2 p_1 = 1/2$

$a_2 q_{1,1} = 1/2$

3 Equations. 4 unknowns

$a_2 = 1/2$ Heun's Method

$a_2 = 1$ Improved Polygon Method

$a_2 = 2/3$ Ralston's Method

4th Order RK

$k_1 = f(x_n, y_n)$

$k_2 = f(x_n + h/2, y_n + k_1/2)$

$k_3 = f(x_n + h/2, y_n + k_2/2)$

$k_4 = f(x_n + h, y_n + k_3)$

$y_{n+1} = y_n + (k_1/6 + k_2/3 + k_3/3 + k_4/6)h + O(h^5)$

Multi-step (Predictor-Corrector) Methods

$y(t_{k+1}) = y_k + \int_{t_k}^{t_{k+1}} f(t, y(t))dt$

When the 3rd order Lagrange polynomial approximation for f based on t_{k-3}, f_{k-3},t_{k-2}, f_{k-2}, t_{k-1}, f_{k-1} and t_k, f_k is integrated over $[t_k, t_{k+1}]$

The predictor equation becomes:

$p_{k+1} = y_k + \frac{h}{25}(-9f_{k-3} + 37f_{k-2} - 59f_{k-1} + 55f_k)$

Then a 3rd order Lagrange polynomial approximation for f based on t_{k-2}, f_{k-2},t_{k-1}, f_{k-1}, t_{k-1}, f_{k-1} and t_k, f_k is integrated over $[t_{k+1}, p_{k+1}]$ and

the corrector equation becomes:

$y_{k+1} = y_k + \frac{h}{24}(f_{k-2} - 5f_{k-1} + 19f_k + 9f_{k+1})$

This is called Adams-Bashford-Moulton Method

Higher Order Differential Equations

$d^2h(t)/dt^2 = (5000 - 0.1(dh(t)/dt)^2/(300 - 10t) - g$

$y_1(t) = h(t)$

$y_2(t) = y_1'(t)$

$\vec{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}$

$d\vec{y}(t)/dt = \vec{f}(t, \vec{y}(t)) = \begin{bmatrix} y_2 \\ (5000 - 0.1y_2^2)/(300 - 10t) - g \end{bmatrix}$

$\partial \vec{f}/\partial t = \vec{f}_t = \begin{bmatrix} 0 \\ 10(5000 - 0.1y_2^2)/(300 - 10t)^2 \end{bmatrix}$

$\partial \vec{f}/\partial \vec{y}(t) = \vec{f}_{\vec{y}} = \begin{bmatrix} \partial f_1/\partial y_1 & \partial f_1/\partial y_2 \\ \partial f_2/\partial y_1 & \partial f_2/\partial y_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-0.2y_2}{(300-10t)} \end{bmatrix}$

$\vec{y}_{i+1} = \vec{y}_i + \vec{f}(t_i, \vec{y}_i)\Delta t + [\vec{f}_t(t_i, \vec{y}_i) + f_{\vec{y}}(t_i, \vec{y}_i) f(t_i, \vec{y}_i)]\Delta^2 t/2$

Modified Euler Method

$\vec{k}_1 = f(t_i, \vec{y}_i)$

$\vec{k}_2 = f(t_i + \Delta t, \vec{y}_i + \Delta t \vec{k}_1)$

$\vec{y}_{i+1} = \vec{y}_i + (\vec{k}_1 + \vec{k}_2)\Delta t/2$

Random Number Generation

Generation of Uniform Deviates

```
#include <stdlib.h>
void srand (int iseed)
int rand()
```

Linear Congruential pseudo random number generator in C. Integer numbers are uniformly distributed in the interval $[0 RAND_M AX - 1]$

To convert them into float numbers distributed in the interval [01.]:

```
x= rand()/(RAND_MAX+1.)
```

To generate an integer number distributed in the interval [010]:

```
j= (int) (10.*rand())/(RAND_MAX+1.)
```

To make the random numbers portable (i.e. to be regenerated anywhere, at any time) use the `srand()` to set the seed of the generator.

To generate a uniform random sequence with arbitrary mean and variance scale the number with the desired standard deviation and add the desired mean after normalizing them with their standard deviation and subtracting from them their mean.

If $y = g(x)$ the $p(y)|dy| = p(x)|dx|$ where p is the probability density function.

If x is uniform distributed $p(x)$ is constant.

$|dx/dy| = p(y) \rightarrow x = \int p(y)dy = P(y) \rightarrow y = P^{-1}(x)$

P is also called cumulative density function.

Hence, to generate an exponentially distributed y i.e. $p(y) = e^{-y}$

$P(y) = \int e^{-y}dy = -e^{-y} \rightarrow P^{-1}(y) = -ln(-y)$

The uniform deviates x are transformed as

$y = -ln(x)$

Rejection Method

1. For a desired random distribution $p(x)$ define an encapsulating sampling distribution $f(x)$ whose integral is $F(x)$,
2. Generate a uniform random number r_0 in $[0, 1]$,
3. Find $x_0 = F^{-1}(r_0)$,
4. Generate a second random number r_1 in $[0, f(x_0)]$,
5. Accept if $r_1 \leq p(x_0)$.

Monte Carlo Estimation

To find the volume of sphere with radius r, encapsulate the boundary of the sphere function $f(x, y, z) = x^2 + y^2 + z^2 - r^2 = 0$, with a suitable function i.e. a cube with edge $2r$,

1. Generate N uniform random number triplets $\{x_i, y_i, z_i\}$ in $[-r, r]$,
2. Accept those if $\sqrt{x_i^2 + y_i^2 + z_i^2} \leq r$.

If N_0 is the number of points falling within the sphere then

$V_{sphere} \approx 8r^3 N_0/N$