Poh Leong Hien
ceeplh@nus.edu.sg

4 Numerical Initial Value Problems

4.1 General First Order IVP

Differential equations are often used to model physical problems, e.g. structural dynamics, seepage, consolidation and heat transfer. The aim here is to determine physical variable y at some future time t, given the initial value. This is known as Initial Value Problem (IVP).

Let the initial time be $t=a$ (quite often this is taken as 0 as an arbitrary reference). We consider first-order ordinary differential equations (ODE) as follows

$$IVP \begin{cases} \dfrac{dy}{dt} = f(t,y) & \text{for } a \le t \le b \\ y(a) = y_0 \end{cases} \tag{4.1}$$

In many books, it is common to denote dy/dt by y' or $\dot{y}$. Thus, in short

$$y' = f \tag{4.2}$$

Higher-order ODEs can be converted into a system of first-order ODEs by defining

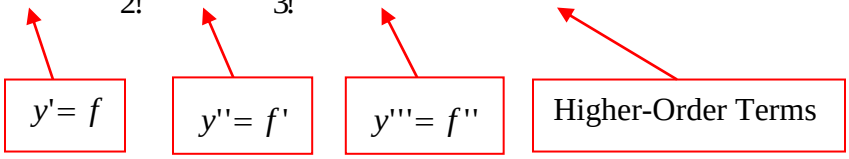
$$y_1 = y \qquad y_2 = y' \qquad y_3 = y'' \qquad \ldots$$

Consider only single ODE here for ease of discussion. The numerical methods are applicable to system of ODEs.

4.1.1 Taylor Series Expansion

This forms the basis for most numerical methods used to solve differential equations. Expansion about solution at t_i with a time step h (forward) gives

$$y(t_{i+1}) = y(t_i) + h\,y'(t_i) + \frac{h^2}{2!}y''(t_i) + \frac{h^3}{3!}y'''(t_i) + \text{H.O.T.}$$



where $t_{i+1} = t_i + h$. In the above illustration, H.O.T. include terms containing h^4, h^5, etc, and they can be represented by just one term called the remainder.

$$R_4 = \frac{h^4}{4!}y''''(\xi) \qquad \text{where } \xi \text{ lies in the interval from } t_i \text{ to } t_{i+1}.$$

This is sometimes simply denoted as $O(h^4)$, meaning a term of order of h^4. (But strictly speaking, we have to worry about the magnitude of y'''' in the error term.)

Poh Leong Hien
ceeplh@nus.edu.sg

When using numerical methods, we normally obtain solutions at only some discrete time points (not continuous), i.e. at t_i where i=1, 2, 3, … , N (finite).

Let y_i be the numerical solution for $y(t_i)$, i.e. variable with subscript i denotes the numerical (approx.) solution for y at a specific discrete time t_i. Hence,

$$y_{i+1} = y_i + h\ f(t_i, y_i) + \frac{h^2}{2!} f'(t_i, y_i) + \frac{h^3}{3!} f''(t_i, y_i) + O(h^4) \tag{4.3}$$

4.1.2 First-Order Methods

4.1.2.1 Euler Explicit (or Forward) Method

Consider

$$y_{i+1} = y_i + h\ f(t_i, y_i) + O(h^2) \tag{4.4}$$

Let's assume we already have the solution at t_i and want to compute the solution at the next step $t_{i+1} = t_i + h$ where h is the time step (increment). A first-order method, known as Euler Explicit (or Forward) Method is obtained by ignoring $O(h^2)$ in the above:

$$\boxed{y_{i+1} = y_i + h\ f(t_i, y_i)} \tag{4.5}$$

Eq (4.5) is an approximation of Taylor (infinite) series. $O(h^2)$ term ignored is called the *local truncation error* (e_i) which results from the approximation in using the truncated (incomplete) Taylor series.

But this error is only for one time step (from i to i+1). If we are interested in the solution for the range [a, b] and use constant time step h, then the total number of steps required is $N = (b-a)/h$. The error will accumulate and approximately the order of error is

$$E_n \sim O(h^2) \times N \sim O(h) \quad \leftarrow \textit{Global truncation error}$$

This error decreases as h decreases. Hence, the method is "consistent". In this case, the method has consistency of order 1. Since this reflects accuracy, this method is said to be first-order accurate.

Generally, we refer to the **global** truncation error when talking about the order of accuracy for a numerical scheme. If a numerical method gives $E_n \sim O(1)$, the error cannot be reduced by using smaller h, and the method is inconsistent.

Truncation error arises from our approximation, e.g. in truncating Taylor's series. This error exists in general even if we had an "exact computer" which could do exact calculations with infinite number of significant figures (i.e. no round-off error).

Round-off error is a different source of error due to the generally inexact representation of floating-point numbers on digital computers.

Poh Leong Hien
ceeplh@nus.edu.sg

Example: Use Euler explicit method to solve

$$IVP \begin{cases} y' = y\left(\frac{2}{t}+1\right) = f(t,y) \quad for\ 1 \le t \le 3 \\ y(1) = 0.37 \text{ (given)} \end{cases} \tag{4.6}$$

Use 10 steps, i.e. $N = 10$, $h = (3-1)/10 = 0.2$, initial: $y_0 = 0.37$ and $t_i = 1 + 0.2i$

Program: Euler Explicit Method

```
function E=EulerExp(f,a,b,y0,N)
% f is the function entered as a string 'f'
% a and b are the left and right limits
% y0 is the initial condition
% N is the number of points
% h is the step size
% E = [T' Y'] where T is the time vector and Y is the solution vector

h=(b-a)/N;
T=zeros(1,N+1);
Y=zeros(1,N+1);
T=a:h:b;
Y(1)=y0;
for j=1:N
    Y(j+1)=Y(j)+h*feval(f, T(j),Y(j));
end
E=[T'  Y'];
```

```
function F1=f(t,y)
F1=[y*(2/t+1)];
```

Output:

```
>> E=EulerExp('F1',1,3,0.37,10)
E =
   1.0000    0.3700
   1.2000    0.5920
   1.4000    0.9077
        ...
   3.0000   13.3681
```

For this problem, the exact solution is $y = t^2 \exp(t\text{-}2)$. Comparison is tabulated below.

t_i	Exact	Euler explicit	Absolute error	Relative error %
1.0	0.3679	0.3700	0.002	0.58
1.2	0.6470	0.5920	0.055	8.51
1.4	1.0757	0.9077	0.168	15.61
1.6	1.7160	1.3486	0.367	21.41
1.8	2.6527	1.9555	0.697	26.28
2.0	4.0000	2.7812	1.219	30.47
2.2	5.9116	3.8937	2.018	34.14
2.4	8.5929	5.3803	3.213	37.39
2.6	12.3175	7.3531	4.964	40.30
2.8	17.4482	9.9550	7.493	42.95
3.0	24.4645	13.3681	11.096	45.36

The numerical solution can be improved by using a smaller h or larger N.

E.g. at t=3: y_i is 17.5066 for N=20, 22.8227 for N=100, 24.4167 for N=1000.

4.1.2.2 Euler Implicit (or Backward) Method

Taylor series can be applied by expanding about solution at t_{i+1} with step $-h$:

$$y_i = y_{i+1} - h f(t_{i+1}, y_{i+1}) + \frac{h^2}{2!} f'(t_{i+1}, y_{i+1}) - \frac{h^3}{3!} f''(t_{i+1}, y_{i+1}) + O(h^4)$$

Thus, another first-order method can be obtained by this *backward* expansion about i+1:

$$y_i = y_{i+1} - h\, f(t_{i+1}, y_{i+1}) + O(h^2) \tag{4.7}$$

This leads to Euler Implicit (or Backward) Method by ignoring $O(h^2)$ term:

$$\boxed{y_{i+1} = y_i + h\, f(t_{i+1}, y_{i+1})} \tag{4.8}$$

Like Euler explicit method, the global truncation error is $E_n = O(h^2) \times N \sim O(h)$

→ First-order accuracy

But the unknown y_{i+1} appears (generally) in the function evaluation on the right-hand side, and thus cannot be computed explicitly.

→ Implicit method: Need to solve iteratively in general and thus need more computational time. But stability is improved compared to explicit method.

Poh Leong Hien
ceeplh@nus.edu.sg

Question: For a given t_{i+1}, how do we solve for y_{i+1} in (4.8)?

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

(a) Newton Raphson / Bisection iteration
(b) Successive Substitution method
   i. Assume a trial value y_{i+1}^{*} and substitute into RHS of (4.8).
   ii. Compute the predicted value y_{i+1}^{**} from LHS of (4.8).
   iii. Check for convergence, e.g. $\text{relative error} = \left| \dfrac{y_{i+1}^{**} - y_{i+1}^{*}}{y_{i+1}^{*}} \right| \le \text{tol}$
   iv. If convergence not achieved, update $y_{i+1}^{*} = y_{i+1}^{**}$. Repeat step (i) – (iv).
   v. Else, update $y_{i+1} = y_{i+1}^{**}$.

4.1.3 Higher-Order Methods

4.1.3.1 Trapezoidal Method

Both versions of Euler method can be interpreted as

$$\textit{New value} = \textit{Current value} + \textit{Time step} \times \textit{Slope}$$

The question is where the slope (derivative) should be computed.

Use "current" slope at i → Euler explicit method
Use "new" slope at i+1 → Euler implicit method

How about using the average of current and new slopes? That is,

$$y_{i+1} = y_i + \frac{h}{2}\left[f(t_i, y_i) + f(t_{i+1}, y_{i+1}) \right] \tag{4.9}$$

This is known as the Trapezoidal Method. Its accuracy order can be shown by Taylor series as follows. Let the local truncation error be:

$$e_i = y_{i+1} - y_i - \frac{h}{2}\left[f(t_i, y_i) + f(t_{i+1}, y_{i+1}) \right] \tag{4.10}$$

Need to include up to second-order (or higher) terms in Taylor series for error analysis.

Forward expansion for y_{i+1} (recall: $f = y'$):

$$y_{i+1} = y_i + h\, f_i + \frac{h^2}{2} f_i' + \frac{h^3}{6} f_i'' + O(h^4) \tag{4.11}$$

Poh Leong Hien
ceeplh@nus.edu.sg

Forward expansion for f_{i+1}:

$$f_{i+1} = f_i + h\ f_i' + \frac{h^2}{2} f_i'' + O(h^3) \tag{4.12}$$

Substitute (4.11) and (4.12) into (4.10):

$$e_i = \left[y_i + h\ f_i + \frac{h^2}{2} f_i' + \frac{h^3}{6} f_i'' + O(h^4) \right] - y_i - \frac{h}{2} \left\{ f_i + \left[f_i + h\ f_i' + \frac{h^2}{2} f_i'' + O(h^3) \right] \right\} \tag{4.13}$$

Local truncation error is $e_i = -\frac{h^3}{12} f_i'' + O(h^4) = O(h^3)$

Global truncation error is $E_N = O(h^2)$ → Second-order accuracy

Thus, the trapezoidal method is second-order and implicit method.

4.1.3.2 Modified Trapezoidal Method / Huen's Method

The above implicit method can be modified to become an explicit method:

(a) First use Euler explicit method to estimate y_i^* (* denotes temporary or intermediate).

(b) Then use this estimate to approximate y_{i+1} as required on the R.H.S. of Eq (4.9).

$$\boxed{\begin{aligned} &(a) \quad y_{i+1}^* = y_i + h\ f(t_i, y_i) \\ &(b) \quad y_{i+1} = y_i + h \times \frac{1}{2} \left[f(t_i, y_i) + f(t_{i+1}, y_{i+1}^*) \right] \end{aligned}} \tag{4.14}$$

This explicit method is known as the Modified Trapezoidal Method or Huen's Method. Its accuracy can be shown to be also second order.

4.1.3.3 Midpoint Method

Another idea:

(a) Use Euler explicit method to estimate the solution at "midpoint", i.e. $t_{i+0.5} = t_i + 0.5h$.

(b) Then go back to i. Re-apply Euler explicit method based on the midpoint slope.

$$\boxed{\begin{aligned} &(a) \quad y_{i+0.5}^* = y_i + 0.5h\ f(t_i, y_i) \\ &(b) \quad y_{i+1} = y_i + h\ f(t_{i+0.5}, y_{i+0.5}^*) \end{aligned}} \tag{4.15}$$

This explicit second-order method is known as the Midpoint Method.

In Sections 4.1.3.2 and 4.1.3.3, the first step is called the predictor – to get an estimate first but this is not final. The second step is called the corrector – to give an improved (corrected) estimate making use of the predictor result (usually for computing slope f).

Poh Leong Hien
ceeplh@nus.edu.sg

4.1.3.4 Error analysis of second-order methods

The Taylor expansion of a function with 2 variables is given by

$$\begin{aligned} f(t+\Delta t\,,\, y+\Delta y) = {} & f(t,y) + \left[\, \Delta t\ f_t(t,y) + \Delta y\ f_y(t,y)\right] \\ & + \tfrac{1}{2!}\left[\, (\Delta t)^2\ f_{tt}(t,y) + 2\Delta t\Delta y\ f_{ty}(t,y) + (\Delta y)^2\ f_{yy}(t,y)\right] \\ & + \tfrac{1}{3!}\left[\, (\Delta t)^3\ f_{ttt}(t,y) + 3(\Delta t)^2\Delta y\ f_{tty}(t,y) + 3\Delta t(\Delta y)^2\ f_{tyy}(t,y) + (\Delta y)^3\ f_{yy}(t,y)\right] \\ & + \ldots \end{aligned} \tag{4.16}$$

Let's consider an approximation scheme which uses the linear combination of two gradients to find $y(t+h)$

$$y(t+h) = y(t) + Ah\ f_0 + Bh\ f_1 \tag{4.17}$$

where $f_0 = f(t,y)$, $f_1 = f(t+Ph, y+Qhf_0)$

A, B, P, Q are constants.

Referring to (4.16), the Taylor expansion for f_1 is given by

$$\begin{aligned} f_1 &= f(t+Ph, y+Qhf_0) \\ &= f(t,y) + Ph\ f_t(t,y) + Qh\ f_y(t,y)\ f(t,y) + O(h^2) \end{aligned} \tag{4.18}$$

Sub (4.18) into (4.17), the Taylor expansion for the approximation scheme becomes

$$\begin{aligned} y(t+h) &= y(t) + Ahf(t,y) + Bh\left[f(t,y) + Ph\ f_t(t,y) + Qh\ f_y(t,y)\ f(t,y)\right] + O(h^3) \\ &= y(t) + (A+B)h\ f(t,y) + BPh^2 f_t(t,y) + BQh^2 f_y(t,y)\ f(t,y) + O(h^3) \end{aligned} \tag{4.19}$$

Now, consider a general function $y(t)$. We can obtain $y(t+h)$ using Taylor expansion. Suppose that we like to have <u>second order</u> accuracy. This implies that the *local* truncation error is $O(h^3)$.

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2}y''(t) + O(h^3) \tag{4.20}$$

Recall that $y'(t) = f(t,y)$. Hence,

$$y''(t) = f_t(t,y) + f_y(t,y)\ y'(t) = f_t(t,y) + f_y(t,y)\ f(t,y) \tag{4.21}$$

Sub (4.21) into (4.20),

$$y(t+h) = y(t) + hf(t,y) + \frac{h^2}{2}\left[f_t(t,y) + f_y(t,y)\ f(t,y)\right] + O(h^3) \tag{4.22}$$

which is the Taylor expansion of $y(t+h)$ with a second order accuracy for the global error.

Poh Leong Hien
ceeplh@nus.edu.sg

For the proposed approximation scheme to have second order global accuracy, its Taylor expansion in (4.19) has to be equal to (4.22). Comparing the terms, we have

$$\begin{aligned} A + B &= 1 \\ BP &= 0.5 \\ BQ &= 0.5 \end{aligned} \tag{4.23}$$

There are only three equations for four unknowns. The system of equations can only be solved if we make certain choices. Two of the more common choices are:

Case 1:

Let $A = 0.5$. This will result in $B = 0.5$, $P = Q = 1$. The approximation scheme in (4.17) becomes

$$y(t+h) = y(t) + 0.5h\left[f(t,y) + f(t+h, y+hf(t,y))\right] \tag{4.24}$$

which is the Modified Trapezoidal method in (4.14).

Case 2:

Let $A = 0$. This will result in $B = 1$, $P = Q = 0.5$. The approximation scheme in (4.17) becomes

$$y(t+h) = y(t) + h\ f(t+0.5h, y+0.5h\ f(t,y)) \tag{4.25}$$

which is the Midpoint in (4.15).

4.1.3.5 Runge-Kutta Method

Extending the above ideas, a more general class of methods is obtained, i.e. Runge-Kutta Methods. The most common one is Runge-Kutta fourth-order (RK4) method:

(a) Euler explicit at half-step

$$y^{*}_{i+0.5} = y_i + 0.5h\ f\left(t_i, y_i\right)$$

(b) Euler "implicit" at half-step

$$y^{**}_{i+0.5} = y_i + 0.5h\ f\left(t_{i+0.5}, y^{*}_{i+0.5}\right)$$

(c) Midpoint method at full step

$$y^{***}_{i+1} = y_i + h\ f\left(t_{i+0.5}, y^{**}_{i+0.5}\right)$$

(d) Combine multiple estimates to get an improved weighted-average slope

$$y_{i+1} = y_i + h \times \frac{1}{6}\left[f(t_i, y_i) + 2f\left(t_{i+0.5}, y^{*}_{i+0.5}\right) + 2f\left(t_{i+0.5}, y^{**}_{i+0.5}\right) + f\left(t_{i+1}, y^{***}_{i+1}\right)\right] \tag{4.26}$$

The weights $(\frac{1}{6}, \frac{2}{6}, \frac{2}{6}, \frac{1}{6})$ are chosen to obtain 4-th order accuracy of (proof by Taylor series again, but very tedious).

Poh Leong Hien
ceeplh@nus.edu.sg

Example: Solve (4.6) with RK-4.

Program: Runge-Kutta fourth-order (explicit) method

```
function RK4=rk4(f,a,b,y0,N)
% f is the function
% a and b are the left and right end points
% y0 is the initial condition y(0)
% N is the mesh point; h is the step size
% RK4 = [T'   Y'] where T is time vector and Y is solution vector

h=(b-a)/N;
T=zeros(1,N+1);
Y=zeros(1,N+1);
T=a:h:b;
Y(1)=y0;
for j=1:N
    k1=h*feval(f, T(j), Y(j));
    k2=h*feval(f, T(j)+h/2, Y(j)+k1/2);
    k3=h*feval(f, T(j)+h/2, Y(j)+k2/2);
    k4=h*feval(f, T(j)+h, Y(j)+k3);

    Y(j+1)=Y(j)+(k1+2*k2+2*k3+k4)/6;
end
RK4=[T' Y'];
```

```
function F1=f(t,y)
F1=[y*(2/t+1)];
```

Output:
>> RK4=rk4('F1',1,3,0.37,10) → Use 10 steps, h = 0.2

RK4 =

1.0000	0.3700
1.2000	0.6506
…	
3.0000	24.5819

→ Compare with exact = 24.465 (0.5% error)

Notes:

- For the same number of steps, RK4 is much more accurate than Euler explicit method (which gave 45% error). But this is achieved at the expense of more f evaluations and longer computer time. There is a trade-off between accuracy and computational cost.
- Many second-order methods (e.g. Huen's method and midpoint method) are actually special cases of RK2 methods.
- There are also implicit Runge-Kutta methods which are useful for solving "stiff" problems, but need iterations (higher computational cost).

Poh Leong Hien
ceeplh@nus.edu.sg

4.1.4 Stability Analysis – Test problem

Accuracy, as reflected by the truncation error order, is clearly an important attribute of a numerical method.

Another important attribute is its stability – whether there is any possibility for the numerical solution to "blow up" (go unbounded) when the exact solution is bounded.

Idea:

A general problem

$$y'(t) = f(t, y) \quad , \quad y(t_0) = y_0 \tag{4.27}$$

can be expand about $(t_0 \, , \, y_0)$ to give

$$y'(t) \approx f(t_0 \, , \, y_0) + \left.\frac{\partial f}{\partial t}\right|_{(t_0 \, , \, y_0)} (t - t_0) + \left.\frac{\partial f}{\partial y}\right|_{(t_0 \, , \, y_0)} [y(t) - y_0] \tag{4.28}$$

The solution $y(t)$ can thus be obtained by solving,

$$y'(t) = k \; y(t) + g(t) \quad , \quad y(0) = y_0 \tag{4.29}$$

where $k = \left.\frac{\partial f}{\partial y}\right|_{(t_0 \, , \, y_0)}$ and $g(t) = f(t_0 \, , \, y_0) + \left.\frac{\partial f}{\partial t}\right|_{(t_0 \, , \, y_0)} (t - t_0) - \left.\frac{\partial f}{\partial y}\right|_{(t_0 \, , \, y_0)} y_0$. This will be a reasonable approximation if $|t - t_0|$ is sufficiently small.

Suppose there is now a small perturbation ε in the initial condition to (4.29), and $\tilde{y}(t)$ is the solution to

$$\tilde{y}'(t) = k \; \tilde{y}(t) + g(t) \quad , \quad \tilde{y}(0) = y_0 + \varepsilon \tag{4.30}$$

The difference in solution between (4.30) and (4.29) is given by

$$z(t) = \tilde{y}(t) - y(t) \tag{4.31}$$

Subtracting (4.30) from (4.29), we obtain

$$z'(t) = k \; z(t) \quad , \quad z(0) = \varepsilon \tag{4.32}$$

The solution to (4.32) is given by

$$z(t) = \varepsilon \, e^{kt} \tag{4.33}$$

Poh Leong Hien
ceeplh@nus.edu.sg

In most applications, we are interested in cases where the real component of k is negative, such that the effect of perturbation ε dies out as $t \to \infty$.

We would like the numerical schemes to exhibit the same behavior, so that any perturbations (errors) will not become unbounded with time. Drawing reference to the above argument, the stability of a numerical method is usually examined by considering the following "test" problem.

$$\begin{aligned} y' &= -ky = f(t,y) \quad , \quad k > 0 \\ y(0) &= 1 \end{aligned} \tag{4.34}$$

The exact solution is $y(t) = \exp(-kt)$ which decays with t and is bounded (for $t \geq 0$).
Note: k = decay constant (unit: 1/time), hence larger k means faster decay.

4.1.4.1 Stability analysis of Euler Explicit

Consider **Euler Explicit Method** in (4.5): $y_{i+1} = y_i + h\, f(t_i, y_i)$
Substitute $f(t,y) = -ky$ to give $y_{i+1} = y_i - h\,k\,y_i$

Recursive solution at i-th step:

$$\begin{aligned} y_i &= (1 - h\,k)\, y_{i-1} \\ &= (1 - h\,k)^2\, y_{i-2} \\ &= (1 - h\,k)^3\, y_{i-3} \\ &= \ \ldots \\ &= (1 - h\,k)^i\, y_0 \end{aligned}$$

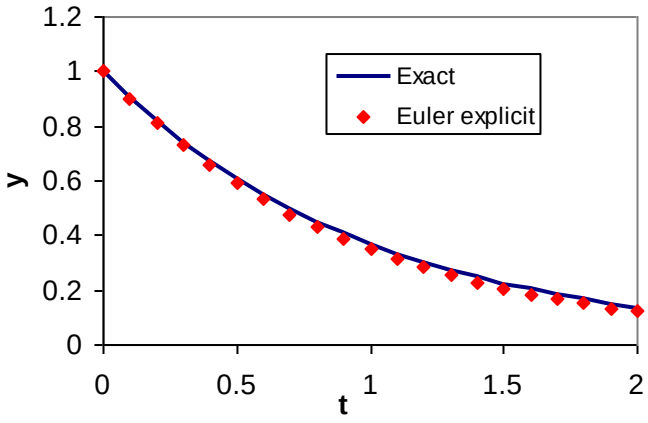
$y_0 \to 1$

The solution is bounded if $|1 - h\,k| \leq 1$ → $0 \leq h\,k \leq 2$
The stability range of h is thus $0 \leq h \leq 2/k$

Consider $k = 1$. The exact solution is given by $y(t) = \exp(-t) = \exp(-ih)$.

Use h = 0.1:

i	t	Exact	Numerical	% Error
0	0	1	1	0.0
1	0.1	0.904837	0.9	-0.5
2	0.2	0.818731	0.81	-1.1
3	0.3	0.740818	0.729	-1.6
4	0.4	0.67032	0.6561	-2.1
5	0.5	0.606531	0.59049	-2.6
…				

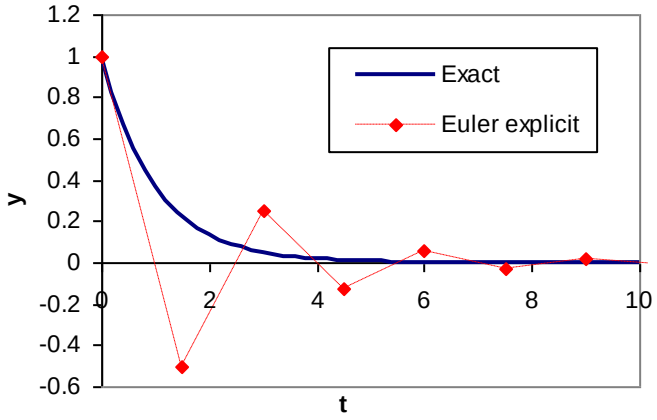


Poh Leong Hien
ceeplh@nus.edu.sg

The solution is stable (does not go unbounded). But stable solution does not necessarily mean accurate solution which will depend on whether the step used is small enough.

Use h = 1.5 (k=1 as before):

i	t	Exact	Numerical	% Error
0	0	1	1	0
1	1.5	0.22313	-0.5	-324
2	3	0.049787	0.25	402
3	4.5	0.011109	-0.125	-1225
4	6	0.002479	0.0625	2421
5	7.5	0.000553	-0.03125	-5750
6	9	0.000123	0.015625	12561
7	10.5	2.75E-05	-0.00781	-28471
8	12	6.14E-06	0.003906	63476
9	13.5	1.37E-06	-0.00195	-142564
10	15	3.06E-07	0.000977	319140



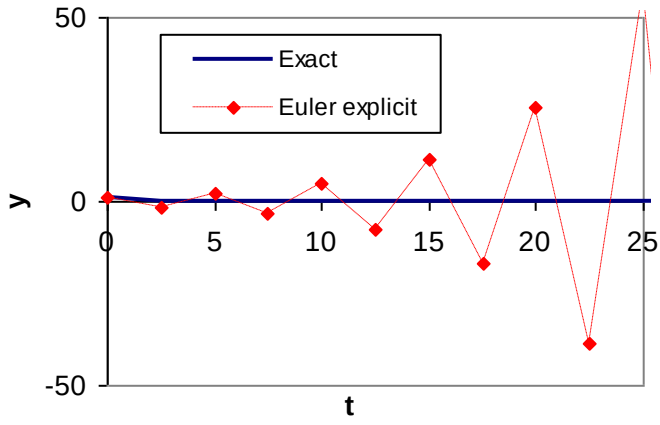
This solution is stable (bounded) but not accurate.

An extreme illustration is to use h = 1 which satisfies the stability range → $y_i = y_{i-1}$
→ Numerical solution is always zero -- not accurate at all but stable!

Outside the stability range $0 \le h\,k \le 2$, the solution magnitude will not decay (as it should), and will eventually blow up when i becomes very large.

Use h = 2.5 (k=1 as before):

i	t	Exact	Numerical	% Error
0	0	1	1	0
1	2.5	0.082085	-1.5	-1927
2	5	0.006738	2.25	33293
3	7.5	0.000553	-3.375	-610314
4	10	4.54E-05	5.0625	11150798
5	12.5	3.73E-06	-7.59375	-203768727
6	15	3.06E-07	11.39063	3723615001
7	17.5	2.51E-08	-17.0859	##########
8	20	2.06E-09	25.62891	##########
9	22.5	1.69E-10	-38.4434	##########
10	25	1.39E-11	57.66504	##########



Obviously, this unstable solution is not acceptable.

Conclusion: Euler explicit method is stable provided that the time step is sufficiently small to satisfy a stability range. This method is said to be conditionally stable.

Poh Leong Hien
ceeplh@nus.edu.sg

4.1.4.2 Stability analysis of Euler Implicit

Consider **Euler Implicit Method** in (4.8): $y_{i+1} = y_i + h\, f(t_{i+1}, y_{i+1})$

Substitute $f(t, y) = -ky$ to give $y_{i+1} = y_i - h\,k\,y_{i+1}$

Recursive solution at *i*-th step:

$$\begin{aligned} y_i &= (1 + h\,k)^{-1}\, y_{i-1} \\ &= (1 + h\,k)^{-2}\, y_{i-2} \\ &= \ \ ... \\ &= (1 + h\,k)^{-i}\, y_0 \end{aligned}$$

Solution is bounded if $|(1 + h\,k)^{-1}| \le 1$. Since this condition is satisfied given that $h > 0$ and $k > 0$, the Euler implicit method is thus unconditionally stable.

In general, for stability analysis, we can try solution of the form

$$y_i = \rho^i \tag{4.35}$$

and plug it into the difference equation.

For Euler explicit method, (4.5) and (4.35) give

$$\rho^{i+1} = (1 - hk)\rho^i \quad \rightarrow \quad \rho = 1 - hk$$

For stable solution, the absolute amplitude of ρ must not be larger than one, i.e. $|\rho| \le 1$.
This would give the same stability range as before.

Similarly for Euler implicit method, (4.8) and (4.35) give

$$\rho^{i+1} = (1 + hk)^{-1}\rho^i \quad \rightarrow \quad \rho = 1/(1 + hk)$$

Poh Leong Hien
ceeplh@nus.edu.sg

4.2 IVP in Dynamic Analysis

(Reference: K-J Bathe "Finite Element Procedures" Chapter 9)

Equations of motion for structural system:

$$\mathbf{M}\ddot{\mathbf{U}}(t)+\mathbf{C}\dot{\mathbf{U}}(t)+\mathbf{K}\mathbf{U}(t)=\mathbf{P}(t) \tag{4.36}$$

M = mass matrix, **C** = damping matrix, **K** = stiffness matrix, **P** = load vector
$\ddot{\mathbf{U}}$ = acceleration vector, $\dot{\mathbf{U}}$ = velocity vector, **U** = displacement vector, t = time.

Initial displacement and velocity at t=0 are given: $\mathbf{U}(0)=\mathbf{U}_0$ and $\dot{\mathbf{U}}(0)=\dot{\mathbf{U}}_0$.

If required, initial acceleration $\ddot{\mathbf{U}}(0)$ can be computed by imposing t=0 in (4.36).

Note: We are now dealing with a system of second-order ODEs. We could transform them into a system of 1st-order ODEs and apply previous methods. Alternative methods have also been developed to solve these 2nd-order ODEs directly. Two popular approaches are discussed below.

4.2.1 Central Difference Method

As before, we can approximate differentials (time derivatives) by differences. Velocity at the i-th step is

$$\dot{\mathbf{U}}_i=\frac{1}{2h}\left(\mathbf{U}_{i+1}-\mathbf{U}_{i-1}\right) \tag{4.37}$$

Acceleration at the i-th step is

$$\ddot{\mathbf{U}}_i=\frac{1}{h}\left(\dot{\mathbf{U}}_{i+0.5}-\dot{\mathbf{U}}_{i-0.5}\right) \tag{4.38}$$

where the two velocity terms in (4.38) can be written similarly (half-step forward and backward) in terms of $\mathbf{U}_{i+1}$, $\mathbf{U}_i$ and $\mathbf{U}_{i-1}$.

Hence, it can be shown that

$$\ddot{\mathbf{U}}_i=\frac{1}{h^2}\left(\mathbf{U}_{i+1}-2\mathbf{U}_i+\mathbf{U}_{i-1}\right) \tag{4.39}$$

Poh Leong Hien
ceeplh@nus.edu.sg

Substitute (4.37) and (4.39) into (4.36),

$$\mathbf{M}\ddot{\mathbf{U}}_i + \mathbf{C}\dot{\mathbf{U}}_i + \mathbf{K}\,\mathbf{U}_i = \mathbf{P}_i$$

$$\mathbf{M}\left[\frac{1}{h^2}\left(\mathbf{U}_{i+1} - 2\mathbf{U}_i + \mathbf{U}_{i-1}\right)\right] + \mathbf{C}\left[\frac{1}{2h}\left(\mathbf{U}_{i+1} - \mathbf{U}_{i-1}\right)\right] + \mathbf{K}\mathbf{U}_i = \mathbf{P}_i \tag{4.40}$$

Rearrange the terms,

$$\underbrace{\left[\frac{1}{h^2}\mathbf{M} + \frac{1}{2h}\mathbf{C}\right]}_{\hat{\mathbf{K}}}\mathbf{U}_{i+1} = \underbrace{\mathbf{P}_i - \left[\mathbf{K} - \frac{2}{h^2}\mathbf{M}\right]\mathbf{U}_i - \left[\frac{1}{h^2}\mathbf{M} - \frac{1}{2h}\mathbf{C}\right]\mathbf{U}_{i-1}}_{\hat{\mathbf{P}}_i} \tag{4.41}$$

$\hat{\mathbf{K}}$ Effective stiffness matrix **$\hat{\mathbf{P}}_i$ Effective load**

➔ Solve $\hat{\mathbf{K}}\,\mathbf{U}_{i+1} = \hat{\mathbf{P}}_i$ -- like solving "static" equations but repeatedly at every step.

Remarks

- Does not require factorization of stiffness matrix **K** (usually the main computational cost).
- If lumped mass approach is used, **M** is diagonal. Furthermore, if we assume mass-proportional damping, **C** = *a* **M** is diagonal too (or simply neglect damping). Then, (4.41) can be solved without equation solver.
- Solution at the (*i*+1) step depends on solutions of previous two steps → Two- step method.
- Solution at $i = 1$ needs $\mathbf{U}_0$ and $\mathbf{U}_{-1}$. A special starting procedure is needed to get $\mathbf{U}_{-1}$ by imposing (4.37) and (4.39) at $i = 0$

$$\mathbf{U}_{-1} = \mathbf{U}_0 - h\dot{\mathbf{U}}_0 + \frac{h^2}{2}\ddot{\mathbf{U}}_0 \tag{4.42}$$

- Equilibrium is imposed at the "current" step, i.e. at step (*i*) where solution has been obtained. The RHS of (4.41) does not involve unknown response. Even if the system is nonlinear (e.g. stiffness is not constant but rather it is a function of displacement), no iteration is required. Hence, this is an **explicit** method.
- Conditionally stable -- need small time step. Suitable for explicit dynamic analysis of short-duration problems (e.g. blast, impact).

Poh Leong Hien
ceeplh@nus.edu.sg

4.2.2 Newmark Method

Newmark originally proposed a constant acceleration method, i.e. the acceleration is assumed to be constant in each interval. Thus, velocity is linear and displacement quadratic:

$$\dot{\mathbf{U}}_{i+1} = \dot{\mathbf{U}}_i + \left(\tfrac{1}{2}\ddot{\mathbf{U}}_i + \tfrac{1}{2}\ddot{\mathbf{U}}_{i+1}\right)h \tag{4.43}$$

$$\mathbf{U}_{i+1} = \mathbf{U}_i + \dot{\mathbf{U}}_i\, h + \left(\tfrac{1}{4}\ddot{\mathbf{U}}_i + \tfrac{1}{4}\ddot{\mathbf{U}}_{i+1}\right)h^2 \tag{4.44}$$

Newmark method was later generalized by involving two numerical parameters (α , δ)

$$\dot{\mathbf{U}}_{i+1} = \dot{\mathbf{U}}_i + \left((1-\delta)\ddot{\mathbf{U}}_i + \delta\,\ddot{\mathbf{U}}_{i+1}\right)h \tag{4.45}$$

$$\mathbf{U}_{i+1} = \mathbf{U}_i + \dot{\mathbf{U}}_i\, h + \left(\left(\tfrac{1}{2}-\alpha\right)\ddot{\mathbf{U}}_i + \alpha\,\ddot{\mathbf{U}}_{i+1}\right)h^2 \tag{4.46}$$

The two parameters can be determined to obtain desired accuracy and stability.
$\alpha = \frac{1}{4}$ and $\delta = \frac{1}{2}$ → Constant acceleration method
$\alpha = \frac{1}{6}$ and $\delta = \frac{1}{2}$. → Linear acceleration method

Consider the generalized form. From (4.46),

$$\ddot{\mathbf{U}}_{i+1} = \frac{1}{\alpha h^2}\left(\mathbf{U}_{i+1} - \mathbf{U}_i\right) - \frac{1}{\alpha h}\dot{\mathbf{U}}_i - \left(\frac{1}{2\alpha} - 1\right)\ddot{\mathbf{U}}_i \tag{4.47}$$

Sub (4.47) into (4.45),

$$\dot{\mathbf{U}}_{i+1} = \frac{\delta}{\alpha h}\left(\mathbf{U}_{i+1} - \mathbf{U}_i\right) + \left(1 - \frac{\delta}{\alpha}\right)\dot{\mathbf{U}}_i + h\left(1 - \frac{\delta}{2\alpha}\right)\ddot{\mathbf{U}}_i \tag{4.48}$$

To solve the problem, equilibrium is imposed at step i+1. Sub (4.47), (4.48) into (4.36),

$$\mathbf{M}\ddot{\mathbf{U}}_{i+1} + \mathbf{C}\dot{\mathbf{U}}_{i+1} + \mathbf{K}\,\mathbf{U}_{i+1} = \mathbf{P}_{i+1}$$

Rearrange the terms to give

$$\hat{\mathbf{K}}\mathbf{U}_{i+1} = \hat{\mathbf{P}}_{i+1} \tag{4.49}$$

where

$$\hat{\mathbf{K}} = \mathbf{K} + \frac{1}{\alpha h^2}\mathbf{M} + \frac{\delta}{\alpha h}\mathbf{C}$$

$$\hat{\mathbf{P}}_{i+1} = \mathbf{P}_{i+1} + \mathbf{M}\left[\frac{1}{\alpha h^2}\mathbf{U}_i + \frac{1}{\alpha h}\dot{\mathbf{U}}_i + \left(\frac{1}{2\alpha} - 1\right)\ddot{\mathbf{U}}_i\right] + \mathbf{C}\left[\frac{\delta}{\alpha h}\mathbf{U}_i + \left(\frac{\delta}{\alpha} - 1\right)\dot{\mathbf{U}}_i + h\left(\frac{\delta}{2\alpha} - 1\right)\ddot{\mathbf{U}}_i\right]$$

Poh Leong Hien
ceeplh@nus.edu.sg

Remarks

- The effective stiffness matrix $\hat{\mathbf{K}}$ involves all three matrices, **M**, **C** and **K**.
- This is an implicit integration method because equilibrium is imposed at a "future" step *i+1*. However, for linear problems with constant **M, C** and **K**, there is no need to solve the equations iteratively -- analogous to using implicit method for $y' = f(t)$ without *y* in *f*.
- In fact, for constant $\hat{\mathbf{K}}$, LU factorization can be done just once.
- "Self-start": No special starting procedure for the one-step method.

Algorithm at each time step:

- Compute effect load $\hat{\mathbf{P}}_{i+1}$.
- Determine displacements $\mathbf{U}_{i+1}$ from $\hat{\mathbf{K}}\mathbf{U}_{i+1} = \hat{\mathbf{P}}_{i+1}$.
- Calculate $\dot{\mathbf{U}}_{i+1}$ from (4.45), and $\ddot{\mathbf{U}}_{i+1}$ from (4.47).

4.2.3 Stability Analysis

This is carried out by considering free vibration of an undamped single-DOF system, i.e.

$$M\ddot{U} + KU = 0 \tag{4.50}$$

The exact solution is harmonic: $A\sin(\omega t + \phi)$ where *A* is the amplitude, ϕ is the phase angle, and $\omega = \sqrt{K/M}$ is the natural frequency.

The numerical solution can be written as difference equations: $\mathbf{Y}_{i+1} = \mathbf{A}\mathbf{Y}_i$

Whether the solution will blow up will depend on the eigenvalues of **A**. In fact, the largest eigenvalue in magnitude will govern.

It is necessary to consider the magnitude in the complex plane, since **A** is not necessarily symmetric and has complex eigenvalues in general.

Define the spectral radius of **A** as $\rho(\mathbf{A}) = \max_{j=1,2...} |\lambda_j|$

Numerical stability requires $\rho(\mathbf{A}) \le 1$

The eigenvalues of **A** are typically dependent on ω, *h* and other numerical parameters (such as α and δ for Newmark method).

Poh Leong Hien
ceeplh@nus.edu.sg

Example: Central Difference Method

Consider:

$$\ddot{U}_i + \omega^2 U_i = 0 \tag{4.51}$$

From (4.39), sub $\ddot{U}_i = \frac{1}{h^2}\left(U_{i+1} - 2U_i + U_{i-1}\right)$ into (4.51) and rearrange it to

$$\begin{bmatrix} U_{i+1} \\ U_i \end{bmatrix} = \underbrace{\begin{bmatrix} 2-\omega^2 h^2 & -1 \\ 1 & 0 \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} U_i \\ U_{i-1} \end{bmatrix} \tag{4.52}$$

Eigenvalues of **A** are

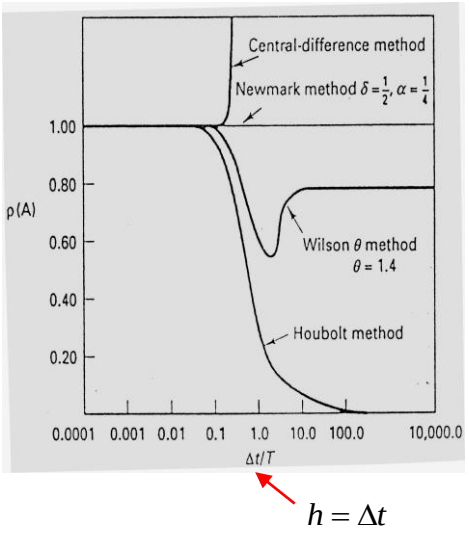
$$\det(\mathbf{A} - \lambda \mathbf{I}) = 0 \tag{4.53}$$

$$\lambda = \frac{2-\omega^2 h^2}{2} \pm \sqrt{\frac{\left(2-\omega^2 h^2\right)^2}{4} - 1}$$

For stability, the spectral radius $\rho(\mathbf{A}) = \max_{j=1,2} |\lambda_j| \le 1$.

This will give rise to a condition that $h \le 2/\omega = T/\pi$

For a generic problem, the central difference method is stable provided $h \le T_n/\pi$, where T_n is the smallest period in the numerical model. Remember: stability does not imply accuracy!



K-J Bathe "Finite Element Procedures", Prentice-Hall, 1996.