Numerical Solution of Differential Equations I

Endre Süli

Mathematical Institute
University of Oxford
2020

Lecture 2

One-step methods

Consider the initial-value problem

$$y' = f(x, y), \qquad x \in [x_0, X_M], \tag{1}$$

$$y(x_0) = y_0. \tag{2}$$

[1] Leonard Euler (1707–1783)

One-step methods

Consider the initial-value problem

$$y' = f(x, y), \qquad x \in [x_0, X_M], \tag{1}$$

$$y(x_0) = y_0. \tag{2}$$

The simplest example of a one-step method for the numerical solution of the initial-value problem (1), (2) is Euler's method.[1]

[1] Leonard Euler (1707–1783)

One-step methods

Consider the initial-value problem

$$y' = f(x, y), \qquad x \in [x_0, X_M], \tag{1}$$
$$y(x_0) = y_0. \tag{2}$$

The simplest example of a one-step method for the numerical solution of the initial-value problem (1), (2) is Euler's method.[1]

Euler's method. Suppose that the initial-value problem (1), (2) is to be solved on the interval $[x_0, X_M]$. We divide this interval by the **mesh-points** $x_n = x_0 + nh$, $n = 0, \ldots, N$, where $h = (X_M - x_0)/N$ and N is a positive integer; h is called the **step size**.

[1] Leonard Euler (1707–1783)

Suppose that, for each n, we seek a numerical approximation y_n to $y(x_n)$, the value of the analytical solution at the mesh point x_n.

Suppose that, for each n, we seek a numerical approximation y_n to $y(x_n)$, the value of the analytical solution at the mesh point x_n.

As $y(x_0) = y_0$ is known, suppose that we have already computed y_n, up to some n, $0 \leq n \leq N-1$; we define

$$y_{n+1} = y_n + hf(x_n, y_n), \qquad n = 0, \ldots, N-1.$$

Suppose that, for each n, we seek a numerical approximation y_n to $y(x_n)$, the value of the analytical solution at the mesh point x_n.

As $y(x_0) = y_0$ is known, suppose that we have already computed y_n, up to some n, $0 \leq n \leq N-1$; we define

$$y_{n+1} = y_n + hf(x_n, y_n), \qquad n = 0, \ldots, N-1.$$

Thus, taking in succession $n = 0, 1, \ldots, N-1$, one step at a time, the approximate values y_n at the mesh points x_n can be easily obtained. This numerical method is known as **Euler's method**.

A simple derivation of Euler's method proceeds by first integrating the differential equation (1) between two consecutive mesh points x_n and x_{n+1} to deduce that, for $n = 0, \ldots, N-1$,

$$y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(x, y(x))\,\mathrm{d}x.$$

Next,

A simple derivation of Euler's method proceeds by first integrating the differential equation (1) between two consecutive mesh points x_n and x_{n+1} to deduce that, for $n = 0, \ldots, N-1$,

$$y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(x, y(x))\,\mathrm{d}x.$$

Next, apply the numerical integration rule

$$\int_{x_n}^{x_{n+1}} g(x)\,\mathrm{d}x \approx hg(x_n),$$

called the **rectangle rule**, with $g(x) = f(x, y(x))$, to get

$$y(x_{n+1}) \approx y(x_n) + hf(x_n, y(x_n)), \qquad n = 0, \ldots N-1, \qquad y(x_0) = y_0.$$

This then motivates the definition of Euler's method.

This can be generalised by replacing the rectangle rule with a one-parameter family of integration rules of the form

$$\int_{x_n}^{x_{n+1}} g(x)\,\mathrm{d}x \approx h\left[(1-\theta)g(x_n) + \theta g(x_{n+1})\right],$$

with $\theta \in [0,1]$ a parameter.

This can be generalised by replacing the rectangle rule with a one-parameter family of integration rules of the form

$$\int_{x_n}^{x_{n+1}} g(x)\,\mathrm{d}x \approx h\left[(1-\theta)g(x_n) + \theta g(x_{n+1})\right],$$

with $\theta \in [0,1]$ a parameter. Applying this with $g(x) = f(x, y(x))$ we find that, for $n = 0, \ldots, N-1$,

$$\begin{aligned} y(x_{n+1}) &\approx y(x_n) + h\left[(1-\theta)f(x_n, y(x_n)) + \theta f(x_{n+1}, y(x_{n+1}))\right], \\ y(x_0) &= y_0. \end{aligned}$$

This can be generalised by replacing the rectangle rule with a one-parameter family of integration rules of the form

$$\int_{x_n}^{x_{n+1}} g(x)\,\mathrm{d}x \approx h\left[(1-\theta)g(x_n) + \theta g(x_{n+1})\right],$$

with $\theta \in [0,1]$ a parameter. Applying this with $g(x) = f(x, y(x))$ we find that, for $n = 0, \ldots, N-1$,

$$\begin{aligned} y(x_{n+1}) &\approx y(x_n) + h\left[(1-\theta)f(x_n, y(x_n)) + \theta f(x_{n+1}, y(x_{n+1}))\right], \\ y(x_0) &= y_0. \end{aligned}$$

This motivates the definition of the following one-parameter family of methods: with y_0 given, define

$$y_{n+1} = y_n + h\left[(1-\theta)f(x_n, y_n) + \theta f(x_{n+1}, y_{n+1})\right], \quad n = 0, \ldots, N-1,$$

parametrised by $\theta \in [0,1]$, called the θ-**method**.

Now, for $\theta = 0$ we recover Euler's method. For $\theta = 1$, and y_0 specified by (2), we get

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1}), \quad n = 0, \dots, N-1,$$

referred to as the **implicit Euler method** since,

Now, for $\theta = 0$ we recover Euler's method. For $\theta = 1$, and y_0 specified by (2), we get

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1}), \quad n = 0, \ldots, N-1,$$

referred to as the **implicit Euler method** since, unlike Euler's method considered above, it requires the solution of an implicit equation in order to determine y_{n+1}, given y_n.

Now, for $\theta = 0$ we recover Euler's method. For $\theta = 1$, and y_0 specified by (2), we get

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1}), \quad n = 0, \ldots, N-1,$$

referred to as the **implicit Euler method** since, unlike Euler's method considered above, it requires the solution of an implicit equation in order to determine y_{n+1}, given y_n.

Euler's method is sometimes called the **explicit Euler method**.

Now, for $\theta = 0$ we recover Euler's method. For $\theta = 1$, and y_0 specified by (2), we get

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1}), \quad n = 0, \ldots, N-1,$$

referred to as the **implicit Euler method** since, unlike Euler's method considered above, it requires the solution of an implicit equation in order to determine y_{n+1}, given y_n.

Euler's method is sometimes called the **explicit Euler method**.

The scheme for $\theta = \frac{1}{2}$ is also of interest: y_0 is supplied by (2) and subsequent values y_{n+1} are computed from

$$y_{n+1} = y_n + \frac{1}{2}h\left[f(x_n, y_n) + f(x_{n+1}, y_{n+1})\right], \qquad n = 0, \ldots, N-1;$$

this is called the **trapezium rule method**.

A further possibility, instead of the **trapezium rule method**,

$$y_{n+1} = y_n + \frac{1}{2} h \left[f(x_n, y_n) + f(x_{n+1}, y_{n+1}) \right], \qquad n = 0, \ldots, N-1;$$

is the following implicit one-step method

$$y_{n+1} = y_n + h\, f\left(\frac{x_n + x_{n+1}}{2}, \frac{y_n + y_{n+1}}{2} \right), \qquad n = 0, \ldots, N-1;$$

called the **implicit midpoint rule**.

A further possibility, instead of the **trapezium rule method**,

$$y_{n+1} = y_n + \frac{1}{2}h\left[f(x_n, y_n) + f(x_{n+1}, y_{n+1})\right], \qquad n = 0, \ldots, N-1;$$

is the following implicit one-step method

$$y_{n+1} = y_n + h\, f\left(\frac{x_n + x_{n+1}}{2}, \frac{y_n + y_{n+1}}{2}\right), \qquad n = 0, \ldots, N-1;$$

called the **implicit midpoint rule**.

Remark

All of these methods can be easily extended to initial-value problems for systems of differential equations of the form

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}),$$
$$\mathbf{y}(x_0) = \mathbf{y}_0,$$

by replacing y *with* $\mathbf{y}$ *and* f *with* $\mathbf{f}$ *throughout.*

Example (MATLAB)

Compare the implicit midpoint rule with the explicit and implicit Euler methods for the following initial-value problem:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \qquad \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Example (MATLAB)

Compare the implicit midpoint rule with the explicit and implicit Euler methods for the following initial-value problem:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \qquad \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Exact solution: $y_1(t) = \sin t$, $y_2(t) = \cos t$.

Example (MATLAB)

Compare the implicit midpoint rule with the explicit and implicit Euler methods for the following initial-value problem:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \qquad \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Exact solution: $y_1(t) = \sin t$, $y_2(t) = \cos t$.

Clearly,

$$Q(t) := \sqrt{y_1^2(t) + y_2^2(t)} = 1 \qquad \text{for all } t \geq 0.$$

Example (MATLAB)

Compare the implicit midpoint rule with the explicit and implicit Euler methods for the following initial-value problem:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \qquad \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Exact solution: $y_1(t) = \sin t$, $y_2(t) = \cos t$.

Clearly,

$$Q(t) := \sqrt{y_1^2(t) + y_2^2(t)} = 1 \qquad \text{for all } t \geq 0.$$

[Run the MATLAB code: testcase2a.m]

Example

Given the initial-value problem $y' = x - y^2$, $y(0) = 0$, on the interval of $x \in [0, 0.4]$, we compute an approximate solution using the θ-method, for $\theta = 0, \frac{1}{2}, 1$, with step size $h = 0.1$.

For the two implicit methods, corresponding to $\theta = \frac{1}{2}$ and $\theta = 1$, the nonlinear equations were solved by using a fixed-point iteration.

Example

Given the initial-value problem $y' = x - y^2$, $y(0) = 0$, on the interval of $x \in [0, 0.4]$, we compute an approximate solution using the θ-method, for $\theta = 0, \frac{1}{2}, 1$, with step size $h = 0.1$.

For the two implicit methods, corresponding to $\theta = \frac{1}{2}$ and $\theta = 1$, the nonlinear equations were solved by using a fixed-point iteration.

k	x_k	y_k for $\theta = 0$	y_k for $\theta = \frac{1}{2}$	y_k for $\theta = 1$
0	0	0	0	0
1	0.1	0	0.00500	0.00999
2	0.2	0.01000	0.01998	0.02990
3	0.3	0.02999	0.04486	0.05955
4	0.4	0.05990	0.07944	0.09857

Table: The values of the numerical solution at the mesh points

For comparison, we also compute the values of the analytical solution $y(x)$ at the mesh points $x_n = 0.1 * n$, $n = 0, \ldots, 4$.

For comparison, we also compute the values of the analytical solution $y(x)$ at the mesh points $x_n = 0.1 * n$, $n = 0, \ldots, 4$. As the solution is not available in closed form, we use a Picard iteration to calculate an accurate approximation to the analytical solution on the interval $[0, 0.4]$ and call this the "exact solution":

$$y_0(x) \equiv 0, \qquad y_k(x) = \int_0^x \left(\xi - y_{k-1}^2(\xi)\right) \mathrm{d}\xi, \quad k = 1, 2, \ldots.$$

For comparison, we also compute the values of the analytical solution $y(x)$ at the mesh points $x_n = 0.1 * n$, $n = 0, \ldots, 4$. As the solution is not available in closed form, we use a Picard iteration to calculate an accurate approximation to the analytical solution on the interval $[0, 0.4]$ and call this the "exact solution":

$$y_0(x) \equiv 0, \qquad y_k(x) = \int_0^x \left(\xi - y_{k-1}^2(\xi)\right) \mathrm{d}\xi, \quad k = 1, 2, \ldots.$$

Hence,

$$\begin{aligned} y_0(x) &\equiv 0, \\ y_1(x) &= \frac{1}{2}x^2, \\ y_2(x) &= \frac{1}{2}x^2 - \frac{1}{20}x^5, \\ y_3(x) &= \frac{1}{2}x^2 - \frac{1}{20}x^5 + \frac{1}{160}x^8 - \frac{1}{4400}x^{11}. \end{aligned}$$

It is easy to prove by induction that

$$y(x) = \frac{1}{2}x^2 - \frac{1}{20}x^5 + \frac{1}{160}x^8 - \frac{1}{4400}x^{11} + O\left(x^{14}\right).$$

Tabulating $y_3(x)$ for $x \in [0, 0.4]$ with step size $h = 0.1$, we get the values of the "exact solution" at the mesh points:

k	x_k	$y(x_k)$
0	0	0
1	0.1	0.00500
2	0.2	0.01998
3	0.3	0.04488
4	0.4	0.07949

Table: Values of the "exact solution" at the mesh points

Tabulating $y_3(x)$ for $x \in [0, 0.4]$ with step size $h = 0.1$, we get the values of the "exact solution" at the mesh points:

k	x_k	$y(x_k)$
0	0	0
1	0.1	0.00500
2	0.2	0.01998
3	0.3	0.04488
4	0.4	0.07949

Table: Values of the "exact solution" at the mesh points

The "exact solution" is in good agreement with the numerical results obtained with $\theta = \frac{1}{2}$: the error is $\leq 5 * 10^{-5}$.

Tabulating $y_3(x)$ for $x \in [0, 0.4]$ with step size $h = 0.1$, we get the values of the "exact solution" at the mesh points:

k	x_k	$y(x_k)$
0	0	0
1	0.1	0.00500
2	0.2	0.01998
3	0.3	0.04488
4	0.4	0.07949

Table: Values of the "exact solution" at the mesh points

The "exact solution" is in good agreement with the numerical results obtained with $\theta = \frac{1}{2}$: the error is $\leq 5 * 10^{-5}$.

For $\theta = 0$ and $\theta = 1$ the mismatch between y_k and $y(x_k)$ is larger: it is $\leq 3 * 10^{-2}$. Question: WHY?

Error analysis of the θ-method

First we have to explain what we mean by *error*.

Error analysis of the θ-method

First we have to explain what we mean by *error*.

The exact solution of the initial-value problem (1), (2) is a function of a continuously varying argument $x \in [x_0, X_M]$, while the numerical solution y_n is only defined at the mesh points x_n, $n = 0, \ldots, N$, so it is a function of a "discrete" argument.

Error analysis of the θ-method

First we have to explain what we mean by *error*.

The exact solution of the initial-value problem (1), (2) is a function of a continuously varying argument $x \in [x_0, X_M]$, while the numerical solution y_n is only defined at the mesh points x_n, $n = 0, \ldots, N$, so it is a function of a "discrete" argument.

We shall compare these two functions by restricting $y(x)$ to the mesh points and comparing $y(x_n)$ with y_n for $n = 0, \ldots, N$.

We define the **global error** e by

$$e_n = y(x_n) - y_n, \qquad n = 0, \ldots, N.$$

So let us consider Euler's explicit method:

$$y_{n+1} = y_n + hf(x_n, y_n), \qquad n = 0, \ldots, N-1, \qquad y_0 = \text{given}.$$

So let us consider Euler's explicit method:

$$y_{n+1} = y_n + hf(x_n, y_n), \qquad n = 0, \ldots, N-1, \qquad y_0 = \text{given}.$$

The quantity

$$T_n := \frac{y(x_{n+1}) - y(x_n)}{h} - f(x_n, y(x_n)),$$

obtained by inserting the analytical solution $y(x)$ into Euler's explicit method and dividing by h is called the **consistency error** (or **truncation error**) of Euler's explicit method.

So let us consider Euler's explicit method:

$$y_{n+1} = y_n + hf(x_n, y_n), \qquad n = 0, \ldots, N-1, \qquad y_0 = \text{given}.$$

The quantity

$$T_n := \frac{y(x_{n+1}) - y(x_n)}{h} - f(x_n, y(x_n)),$$

obtained by inserting the analytical solution $y(x)$ into Euler's explicit method and dividing by h is called the **consistency error** (or **truncation error**) of Euler's explicit method.

It measures the extent to which the analytical solution fails to satisfy the difference equation for Euler's method.

As $f(x_n, y(x_n)) = y'(x_n)$, by applying Taylor's Theorem, it follows that there exists a $\xi_n \in (x_n, x_{n+1})$ such that

$$T_n = \frac{1}{2} h y''(\xi_n),$$

where we have assumed that that f is a sufficiently smooth function of two variables to ensure that y'' exists and is bounded on the interval $[x_0, X_M]$.

As $f(x_n, y(x_n)) = y'(x_n)$, by applying Taylor's Theorem, it follows that there exists a $\xi_n \in (x_n, x_{n+1})$ such that

$$T_n = \frac{1}{2} h y''(\xi_n),$$

where we have assumed that that f is a sufficiently smooth function of two variables to ensure that y'' exists and is bounded on the interval $[x_0, X_M]$. Since from the definition of Euler's method

$$0 = \frac{y_{n+1} - y_n}{h} - f(x_n, y_n),$$

subtracting this from the definition of the consistency error we get

$$e_{n+1} = e_n + h[f(x_n, y(x_n)) - f(x_n, y_n)] + hT_n.$$

Assuming that $|y_n - y_0| \leq Y_M$ the Lipschitz condition implies that

$$|e_{n+1}| \leq (1 + hL)|e_n| + h|T_n|, \qquad n = 0, \ldots, N-1.$$

Assuming that $|y_n - y_0| \leq Y_M$ the Lipschitz condition implies that

$$|e_{n+1}| \leq (1 + hL)|e_n| + h|T_n|, \qquad n = 0, \ldots, N-1.$$

Now, let $T = \max_{0 \leq n \leq N-1} |T_n|$; then,

$$|e_{n+1}| \leq (1 + hL)|e_n| + hT, \qquad n = 0, \ldots, N-1.$$

Assuming that $|y_n - y_0| \leq Y_M$ the Lipschitz condition implies that

$$|e_{n+1}| \leq (1 + hL)|e_n| + h|T_n|, \qquad n = 0, \dots, N-1.$$

Now, let $T = \max_{0 \leq n \leq N-1} |T_n|$; then,

$$|e_{n+1}| \leq (1 + hL)|e_n| + hT, \qquad n = 0, \dots, N-1.$$

By induction, and noting that $1 + hL \leq \mathrm{e}^{hL}$,

$$|e_n| \;\leq\; \mathrm{e}^{L(x_n - x_0)}|e_0| + \frac{T}{L}\left(\mathrm{e}^{L(x_n - x_0)} - 1\right), \qquad n = 1, \dots, N.$$

Assuming that $|y_n - y_0| \leq Y_M$ the Lipschitz condition implies that

$$|e_{n+1}| \leq (1 + hL)|e_n| + h|T_n|, \qquad n = 0, \ldots, N-1.$$

Now, let $T = \max_{0 \leq n \leq N-1} |T_n|$; then,

$$|e_{n+1}| \leq (1 + hL)|e_n| + hT, \qquad n = 0, \ldots, N-1.$$

By induction, and noting that $1 + hL \leq \mathrm{e}^{hL}$,

$$|e_n| \;\; \leq \;\; \mathrm{e}^{L(x_n - x_0)}|e_0| + \frac{T}{L}\left(\mathrm{e}^{L(x_n - x_0)} - 1\right), \qquad n = 1, \ldots, N.$$

This estimate, together with the bound

$$|T| \leq \frac{1}{2}hM_2, \qquad M_2 = \max_{x \in [x_0, X_M]} |y''(x)|,$$

yields

$$|e_n| \leq \mathrm{e}^{L(x_n - x_0)}|e_0| + \frac{M_2 h}{2L}\left(\mathrm{e}^{L(x_n - x_0)} - 1\right), \qquad n = 0, \ldots, N.$$

By a similar argument one can show that, for the θ-method,

$$\begin{aligned}|e_n| &\leq |e_0| \exp\left(L\frac{x_n - x_0}{1-\theta Lh}\right) \\ &\quad + \frac{h}{L}\left\{\left|\frac{1}{2}-\theta\right| M_2 + \frac{1}{6}(1+3\theta)hM_3\right\}\left[\exp\left(L\frac{x_n - x_0}{1-\theta Lh}\right) - 1\right],\end{aligned}$$

for $n = 0, \ldots, N$, where now $M_3 = \max_{x\in[x_0, x_M]} |y'''(x)|$.

By a similar argument one can show that, for the θ-method,

$$
\begin{aligned}
|e_n| \quad &\leq \quad |e_0| \exp\left(L\frac{x_n - x_0}{1 - \theta L h}\right) \\
&\qquad + \frac{h}{L}\left\{\left|\frac{1}{2} - \theta\right| M_2 + \frac{1}{6}(1 + 3\theta) h M_3\right\}\left[\exp\left(L\frac{x_n - x_0}{1 - \theta L h}\right) - 1\right],
\end{aligned}
$$

for $n = 0, \ldots, N$, where now $M_3 = \max_{x \in [x_0, x_M]} |y'''(x)|$.

In the absence of rounding errors in the imposition of the initial condition (2) we can suppose that $e_0 = y(x_0) - y_0 = 0$. Assuming that this is the case, we see that $|e_n| = \mathcal{O}(h^2)$ for $\theta = \frac{1}{2}$, while for $\theta = 0$ and $\theta = 1$, and indeed for any $\theta \neq \frac{1}{2}$, $|e_n| = \mathcal{O}(h)$ only.

By a similar argument one can show that, for the θ-method,

$$\begin{aligned} |e_n| &\leq |e_0| \exp\left(L\frac{x_n - x_0}{1 - \theta L h}\right) \\ &\quad + \frac{h}{L}\left\{\left|\frac{1}{2} - \theta\right| M_2 + \frac{1}{6}(1 + 3\theta) h M_3\right\}\left[\exp\left(L\frac{x_n - x_0}{1 - \theta L h}\right) - 1\right], \end{aligned}$$

for $n = 0, \ldots, N$, where now $M_3 = \max_{x \in [x_0, x_M]} |y'''(x)|$.

In the absence of rounding errors in the imposition of the initial condition (2) we can suppose that $e_0 = y(x_0) - y_0 = 0$. Assuming that this is the case, we see that $|e_n| = \mathcal{O}(h^2)$ for $\theta = \frac{1}{2}$, while for $\theta = 0$ and $\theta = 1$, and indeed for any $\theta \neq \frac{1}{2}$, $|e_n| = \mathcal{O}(h)$ only.

This explains why in the tables the values y_n of the numerical solution computed with the trapezium-rule method ($\theta = \frac{1}{2}$) were closer to the analytical solution $y(x_n)$ at the mesh points than those obtained with the explicit and the implicit Euler methods.