1 Some background on ODEs.

In this section some useful notation on ordinary differential equations will be presented. We will also give existence and uniqueness results, but without proofs.

A system of m first order ordinary differential equation is given by

$$y' = f(t, y) \tag{1}$$

or, written out, as

$$\begin{aligned} y_1' &= f_1(t, y_1, \cdots, y_m), \\ y_2' &= f_2(t, y_1, \cdots, y_m), \\ &\vdots \\ y_m' &= f_m(t, y_1, \cdots, y_m). \end{aligned}$$

This is an *initial value problem* (IVP) if the solution is given at some point t_0, thus

$$y_1(t_0) = y_{1,0},\ y(t_0) = y_{2,0},\ \cdots\ y_m(t_0) = y_{m,0}.$$

Example 1.1. *The following equation is an example of the Lotka-Volterra equation:*

$$\begin{aligned} y_1' &= y_1 - y_1 y_2, \\ y_2' &= y_1 y_2 - 2y_2. \end{aligned}$$

An ODE is called *autonomous* if f is not a function of t, but only of y. The Lotka-Volterra equation is an example of an autonomous ODE. A nonautonomous system can be made autonomous by a simple trick, just add the equation

$$y_{m+1}' = 1, \qquad y_{m+1}(t_0) = t_0,$$

and replace t with y_{m+1}. Also higher order ODE/IVPs

$$u^{(m)} = f(t, u, u', \cdots, u^{(m-1)}), \qquad u(t_0) = u_0,\ u'(t_0) = u_0',\ \cdots, u^{(m-1)}(t_0) = u_0^{(m-1)},$$

where $u^{(m)} = d^m u/dt^m$, can be written as a system of first order equations, again by a simple trick: Let

$$y_1 = u,\ y_2 = u',\ \cdots y_m = u^{(m-1)},$$

and we get the system

$$\begin{aligned} y_1' &= y_2, & y_1(t_0) &= u_0, \\ y_2' &= y_3, & y_2(t_0) &= u_0', \\ &\vdots & &\vdots \\ y_{m-1}' &= y_m, & y_{m-1}(t_0) &= u_0^{(m-2)}, \\ y_m' &= f(t, y_1, y_2, \cdots, y_m), & y_m(t_0) &= u_0^{(m-1)}. \end{aligned}$$

Example 1.2. *Van der Pol's equation is given by*

$$u'' + \mu(u^2 - 1)u' + u = 0.$$

Using $y_1 = u$ *and* $y_2 = u'$ *this equation can be rewritten as*

$$\begin{aligned} y_1' &= y_2, \\ y_2' &= \mu(1 - y_1^2)y_2 - y_1. \end{aligned}$$

This problem was first introduced by Van der Pol in 1926 in the study of an electronic oscillator.

Before concluding this section, we present some existence and uniqueness results for solution of ODEs.

Definition 1.3. *A function* $f : \mathbb{R} \times \mathbb{R}^m \to \mathbb{R}^m$ *satisfies the* Lipschitz condition *with respect to* y *on a domain* $(a, b) \times D$ *where* $D \subset \mathbb{R}^m$ *if there exist a constant* L *so that*

$$\|f(t, y) - f(t, \tilde{y})\| \leq L\|y - \tilde{y}\|, \qquad \text{for all} \quad t \in (a, b),\ y, \tilde{y} \in D.$$

The constant L *is called* the Lipschitz constant.

It is not hard to show that the function f satisfies the Lipschitz condition if $\partial f_i/\partial y_j,\ i, j = 1, \cdots, m$ are continuous and bounded on the domain.

Theorem 1.4. *Consider the initial value problem*

$$y' = f(t, y), \qquad y(t_0) = y_0. \tag{2}$$

If

1. $f(t, y)$ *is continuous in* $(a, b) \times D$,
2. $f(t, y)$ *satisfies the Lipschitz condition with respect to* y *in* $(a, b) \times D$.

with given initial values $t_0 \in (a, b)$ *and* $y_0 \in D$, *then* (5) *has one and only one solution in* $(a, b) \times D$.

2 Numerical solution of ODEs.

In this section we develop some simple methods for the solution of initial value problems. In both cases, let us assume that we somehow have found solutions $y_l \approx y(t_l)$, for $l = 0, 1, \cdots, n$, and we want to find an approximation $y_{n+1} \approx y(t_{n+1})$ where $t_{n+1} = t_n + h$, where h is the stepsize. Basically, there are two different classes of methods in practical use.

1. *One-step methods.* Only y_n is used to find the approximation y_{n+1}. One-step methods usually require more than one function evaluation pr. step. They can all be put in a general abstract form

$$y_{n+1} = y_n + h\Phi(t_n, y_n; h).$$

2. *Linear multistep methods:* y_{n+1} is approximated from $y_{n-k+1}, \cdots, y_n$.

2.1 Some examples of one-step methods.

Assume that t_n, y_n is known. The exact solution $y(t_{n+1})$ with $t_{n+1} = t_n + h$ of (4) passing through this point is given by

$$y(t_n + h) = y_n + \int_{t_n}^{t_{n+1}} y'(\tau)d\tau = y_n + \int_{t_n}^{t_{n+1}} f(\tau, y(\tau))d\tau. \tag{3}$$

The idea is to find approximations to the last integral. The simplest idea is to use $f(\tau, y(\tau)) \approx f(t_n, y_n)$, in which case we get the Euler method again:

$$y_{n+1} = y_n + hf(t_n, y_n).$$

The integral can also be approximated by the trapezoidal rule

$$\int_{t_n}^{t_{n+1}} f(\tau, y(\tau)) = \frac{h}{2}(f(t_n, y_n) + f(t_{n+1}, y(t_{n+1})).$$

By replacing the unknown solution $y(t_{n+1})$ by y_{n+1} we get the *trapezoidal method*

$$y_{n+1} = y_n + \frac{h}{2}\left(f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right).$$

Here y_{n+1} is available by solving a (usually) nonlinear system of equations. Such methods are called implicit. To avoid this extra difficulty, we could replace y_{n+1} on the right hand side by the approximation from Eulers method, thus

$$\begin{aligned}
\tilde{y}_{n+1} &= y_n + hf(t_n, y_n); \\
y_{n+1} &= y_n + \frac{h}{2}\left(f(t_n, y_n) + f(t_{n+1}, \tilde{y}_{n+1})\right).
\end{aligned}$$

This method is called the *improved Euler method*. Similarly, we could have used the midpoint rule for the integral,

$$\int_{t_n}^{t_{n+1}} f(\tau, y(\tau)) = \left(f(t_n + \frac{h}{2}, y(t_n + \frac{h}{2})\right),$$

and replaced $y(t_n + \frac{h}{2})$ by one half Euler step. The result is the *modified Euler method*:

$$\begin{aligned}
\tilde{y}_{n+\frac{1}{2}} &= y_n + \frac{h}{2}f(t_n, y_n), \\
y_{n+1} &= y_n + hf(t_n + \frac{h}{2}, \tilde{y}_{n+\frac{1}{2}}).
\end{aligned}$$

Do we gain anything by constructing these methods? Let us solve the problem from Example 1.1 using improved/modified Euler with $h = 0.1$. For each step, also the global error $e_n = y(t_n) - y_n$ is computed. For comparison, also the

result for the Euler method is included.

	Euler		improved Euler		modified Euler	
t_n	y_n	e_n	y_n	e_n	y_n	e_n
0.0	1.000000	0	1.000000	0	1.000000	0
0.1	1.000000	$-9.95 \cdot 10^{-3}$	0.990000	$4.98 \cdot 10^{-5}$	0.990000	$4.98 \cdot 10^{-5}$
0.2	0.980000	$-1.92 \cdot 10^{-2}$	0.960696	$9.34 \cdot 10^{-5}$	0.960597	$1.92 \cdot 10^{-4}$
0.3	0.940800	$-2.69 \cdot 10^{-2}$	0.913814	$1.17 \cdot 10^{-4}$	0.913528	$4.03 \cdot 10^{-4}$
0.4	0.884352	$-3.22 \cdot 10^{-2}$	0.852040	$1.04 \cdot 10^{-4}$	0.851499	$6.45 \cdot 10^{-4}$
0.5	0.813604	$-3.48 \cdot 10^{-2}$	0.778765	$3.60 \cdot 10^{-5}$	0.777930	$8.71 \cdot 10^{-4}$
0.6	0.732243	$-3.46 \cdot 10^{-2}$	0.697773	$-9.69 \cdot 10^{-5}$	0.696636	$1.04 \cdot 10^{-3}$
0.7	0.644374	$-3.17 \cdot 10^{-2}$	0.612924	$-2.98 \cdot 10^{-4}$	0.611507	$1.12 \cdot 10^{-3}$
0.8	0.554162	$-2.69 \cdot 10^{-2}$	0.527850	$-5.58 \cdot 10^{-4}$	0.526202	$1.09 \cdot 10^{-3}$
0.9	0.465496	$-2.06 \cdot 10^{-2}$	0.445717	$-8.59 \cdot 10^{-4}$	0.443904	$9.54 \cdot 10^{-4}$
1.0	0.381707	$-1.38 \cdot 10^{-2}$	0.369053	$-1.17 \cdot 10^{-3}$	0.367153	$7.27 \cdot 10^{-4}$

As we can see, there is a significant improvement in accuracy, compared with the Euler method.