Solving differential equations (DE) Berry Boessenkool, berry-b@gmx.de, April 2013

In the model of **exponential growth of populations**, population change $\left(\frac{\delta N}{\delta t}\right)$ is considered to be dependent on the current size of population (N) as follows: N multiplied by a simple reproduction rate (r), the number of offspring per individual per time step (δt). N might be the number of individuals in a habitat, mg of bacteria or number of yeast cells in a Petri dish, wood volume in a forest, count of infected people, …

$$\boldsymbol{\frac{\delta N}{\delta t} = r * N} \quad \Rightarrow \quad N_t = N_{t-\delta t} + \delta N \quad \Rightarrow \quad N_t = N_{t-\delta t} + r * \delta t * N_{t-\delta t}$$

Numerical approach to solution

Given an initial starting population size $N(0) = 18$ with a reproduction rate of $r = 0.21\frac{1}{h}$ and a time step of $\delta t = 2h$, the estimated population size after two hours can be calculated as

$N(2) = N(0) + r * \delta t * N(0) = 18 + 0.21 * 2 * 18 = 25.56$. Please calculate N after 4 and 6 hours.

You will see that this discretization is too rough, when the results are compared with the **analytical solution**. Please calculate N(6) for the abovementioned scenario with the equation below.

$$\boxed{\boldsymbol{\frac{\delta N}{\delta t} = r * N}}$$

$$\frac{1}{N}\delta N = r * \delta t$$

$$\int \frac{1}{N}\delta N = \int r * \delta t$$

$$\ln(N) + c_1 = r * t + c_2 \quad , \quad c_i \in \mathbb{R}$$

$$\ln(N) = r * t + c_3$$

$$e^{\ln(N)} = e^{r*t+c_3}$$

$$N = e^{rt} * e^{c_3} = e^{rt} * c_4$$

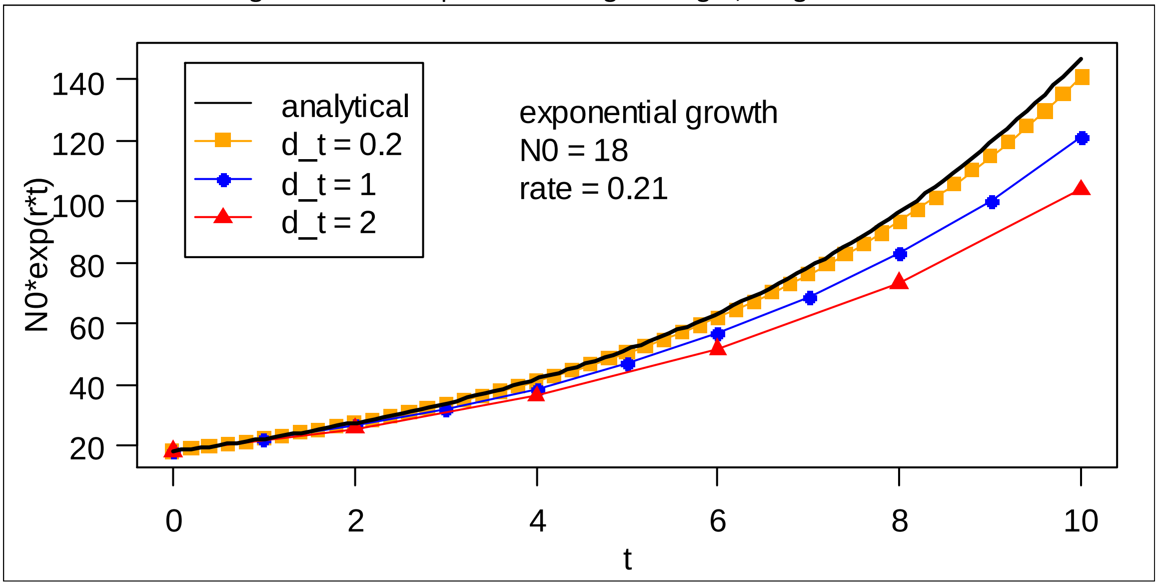
$$N(t) = c * e^{rt}$$

$$Initial\ condition\ (Anfangsbedingung): N(0) = N_0$$

$$N(0) = N_0 = c * e^{r*0} = c * 1 \quad \Rightarrow \quad c = N_0$$

$$\boxed{\boldsymbol{N(t) = N_0 * e^{rt}}}$$

A better numerical approach would be to **refine the time step**, as shown in the image below. Beware of the additional **computing time** this may consume in more complex DE-systems with several (interdependent) variables. Also, very small time steps carry the danger of rounding errors (due to the arithmetic imprecision of the computer).Usually in automated routines, a locally variable time/space discretization is used on the base of relative change – at times or places with high changes, a high resolution is used.



This was a relatively simple equation, where the analytical solution was easily obtained. Consider the model of **logistic growth** towards the (ecosystem-) capacity K, where the rate is adjusted, so that populations don't keep growing exponentially, but reach a steady state at K.

$$\boldsymbol{\frac{\delta N}{\delta t} = r * N * \frac{K-N}{K}}$$

Relationship N to K	$\boldsymbol{\frac{K-N}{K}}$	Population change
$\boldsymbol{N < K}$	> 0	growth
$\boldsymbol{N = K}$	= 0	equilibrium
$\boldsymbol{N > K}$	< 0	decline

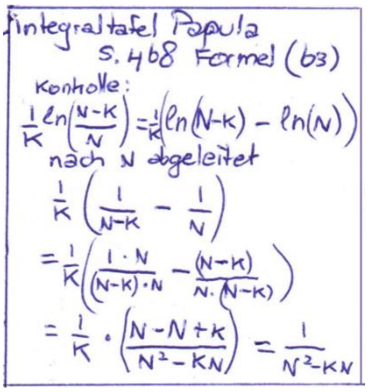
It is still possible to find an analytical solution, but it becomes increasingly more demanding of our mathematical capacities.

$$\boldsymbol{\frac{\delta N}{\delta t} = r * N * \frac{K-N}{K}}$$

$$\frac{1}{N} * \frac{K}{K-N} \delta N = r\, \delta t$$

$$-K * \int \frac{1}{N^2 - KN} \delta N = \int r\, \delta t$$

$$\underbrace{-K * \frac{1}{K} * \ln\left(1 - \frac{K}{N}\right)} = r * t + c_3$$

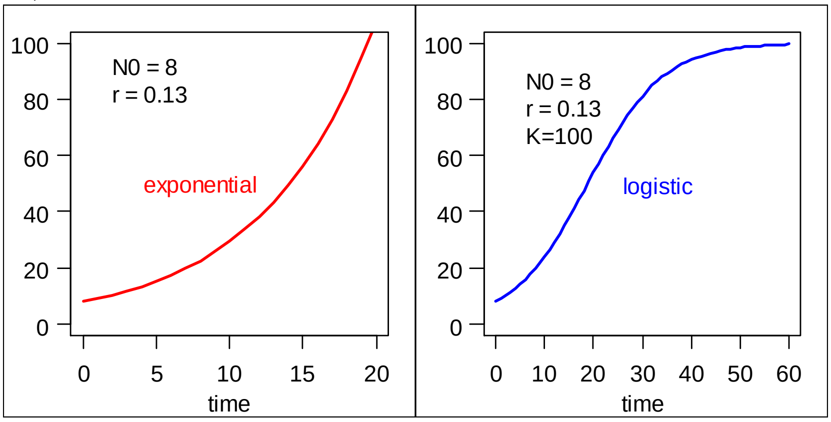


$$1 - \frac{K}{N} = e^{-rt} * c_4$$

$$N = \frac{K}{1 - c_4 * e^{-rt}}$$

$$initial\ condition: N(0) = N_0 = \frac{K}{1-c_4} \Rightarrow c_4 = 1 - \frac{K}{N_0}$$

$$\boxed{\boldsymbol{N(t) = \frac{K}{1 - \left(1 - \frac{K}{N_0}\right) * e^{-rt}}}}$$



This calls for numerical solutions for models with complex differential equations, interdependencies (partial DE) or second degree derivatives. Moreover, for nonlinear DE there often are **no analytical solutions**.

We have so far used the **explicit** solving scheme called **Euler method**, a first-order numerical procedure for solving ordinary differential equations (ODEs) with a given initial value by numerical integration. The Euler method is a first-order method, which means that the local error (error per step) is proportional to the square of the step size, and the global error (error at a given time) is proportional to the step size. It also suffers from stability problems. For these reasons, the Euler method is not often used in practice. It serves as the basis to construct more complicated methods.[1]

The **backward Euler method** is **implicit**. Here's Wikipedia on the difference: Explicit methods calculate the state of a system at a later time from the state of the system at the current time, while implicit methods find a solution by solving an equation involving both the current state of the system and the later one.[2]

Euler forward: $y(t_1) = y(t_0) + y'(t_0)\,\Delta t$

Euler backward: $y(t_1) = y(t_0) + y'(t_1)\,\Delta t$

Trapezoid: $y(t_1) = y(t_0) + \frac{1}{2}\left(y'(t_0) + y'(t_1)\right)\Delta t$

Problem: we need slope y' at points for which y is not known yet. Usually, a pre-estimation with Euler forward is made.

[1] http://en.wikipedia.org/wiki/Euler_method

[2] http://en.wikipedia.org/wiki/Explicit_and_implicit_methods