L2 Computational Physics
Week 3 – 1st order ODEs

- To view slide-by-slide animations in this PDF, do not view it in your web browser
    - Save it to disk
    - View “Full Screen” on Acrobat / Windows - [CTRL+L]
    - “View / Single page” then optionally “View /Enter Full Screen” in OS X/Previews

Differential Equations

Numerical Solvers

Background

- What sets differential equations apart from numerical integration?

- Function depends upon itself

- Often a DEQ is not solvable by analytical techniques
  - Use numerical approximations

DEQ's everywhere

- Physics
  - N-body problem (Newton's second law)
  - Fluid Dynamics (Navier-Stokes equation)
  - Wave Equation
  - Electromagnetism (Maxwell's equations)
  - General Relativity (Einstien's Field Equation)
  - Ballistics (Newton, friction)
- Economics
- Biology
- Everywhere!

Differential equations

- Equation for variable 'x' depends on independent variable 't'

$$\frac{dx}{dt} = f(t)$$

- An ordinary differential equation (ODE) for variable 'x' depends on the value of 'x' itself as well as 't'

$$\frac{dx}{dt} = f(x(t),t)$$

- Some simple DEQs depend upon only the differentials and no independent variables

$$\frac{dx}{dt} = f(x(t))$$

Numerical solvers

- Convert numerical integration methods into differential equation solvers
  - Rectangle Rule → Euler Method
  - Trapezium Rule → Heun's Rule RK2
  - Simpson's Rule → Runge Kutta RK4

Explicit Methods

- This week we are looking at *explicit* methods of solving DEQs.

- This means that we use a formula to explicitly derive the quantity $X_{t+\Delta t}$ as a function of X_t, t

- There are also *implicit* methods, but we'll not talk about them for now…

Euler's Method

Euler's method

- Given a DEQ for x:

$$\frac{dx}{dt} = f(x_t, t)$$

- The gradient is defined by the following limit

$$\frac{dx}{dt} = f(x_t, t) \underset{\Delta t \to 0}{\overset{\lim}{=}} \frac{x_{t+\Delta t} - x_t}{\Delta t}$$

- So for a small value of Δt we can rearrange to solve the equation:

$$x_{t+\Delta t} \approx x_t + f(x_t, t)\Delta t$$

Radioactive Decay

Radioactive Decay

- A population of unstable nuclei undergoes radioactive decay
- A single nuclei decays at a random time
- The ‘continuum behaviour’ of this random process can be described by an ODE

Radioactive Decay

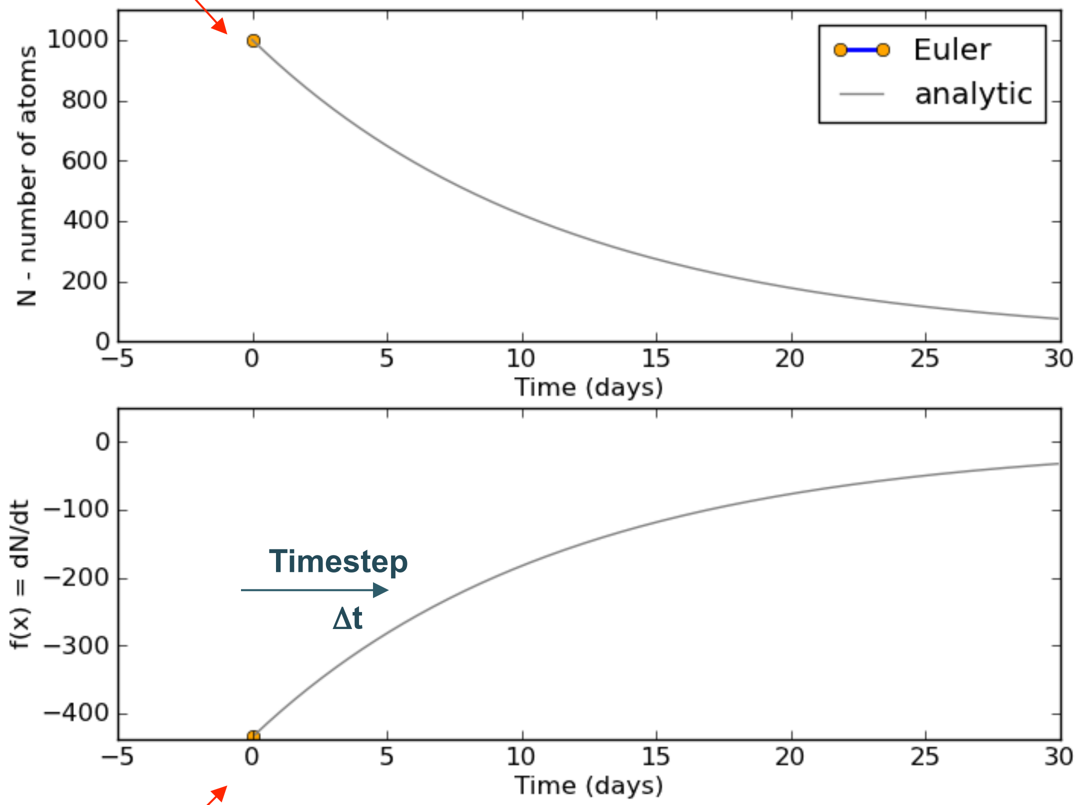
- total population of nuclei: N
- Half-life decay process $t_{1/2}$
- Define mean lifetime of a nuclei:

  $\tau = t_{1/2} / \ln(2)$

- DEQ $f(n,t) = dN/dt = - N / \tau$
- Analytical solution: $N(t) = N_0 e^{-t/\tau}$

Start at T_0

1. Initial conditions – N_0

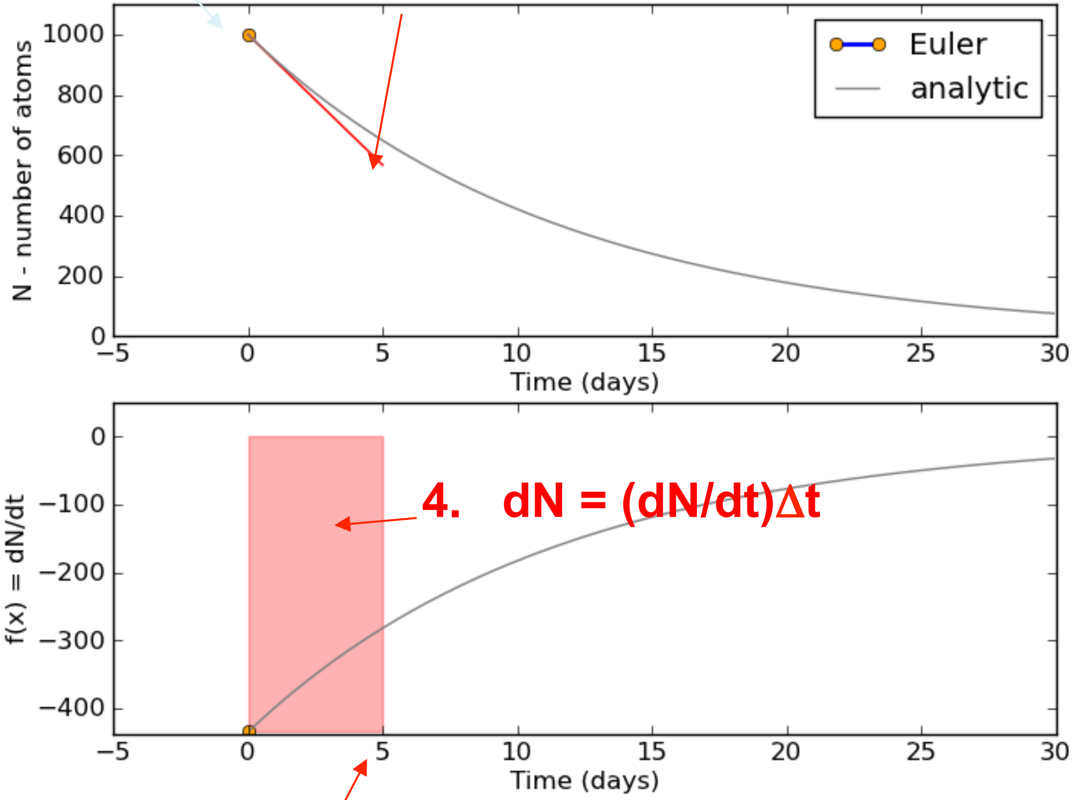


2. Calculate $dN/dt = -N_0/\tau$

Timestep

1. Initial conditions – N_0

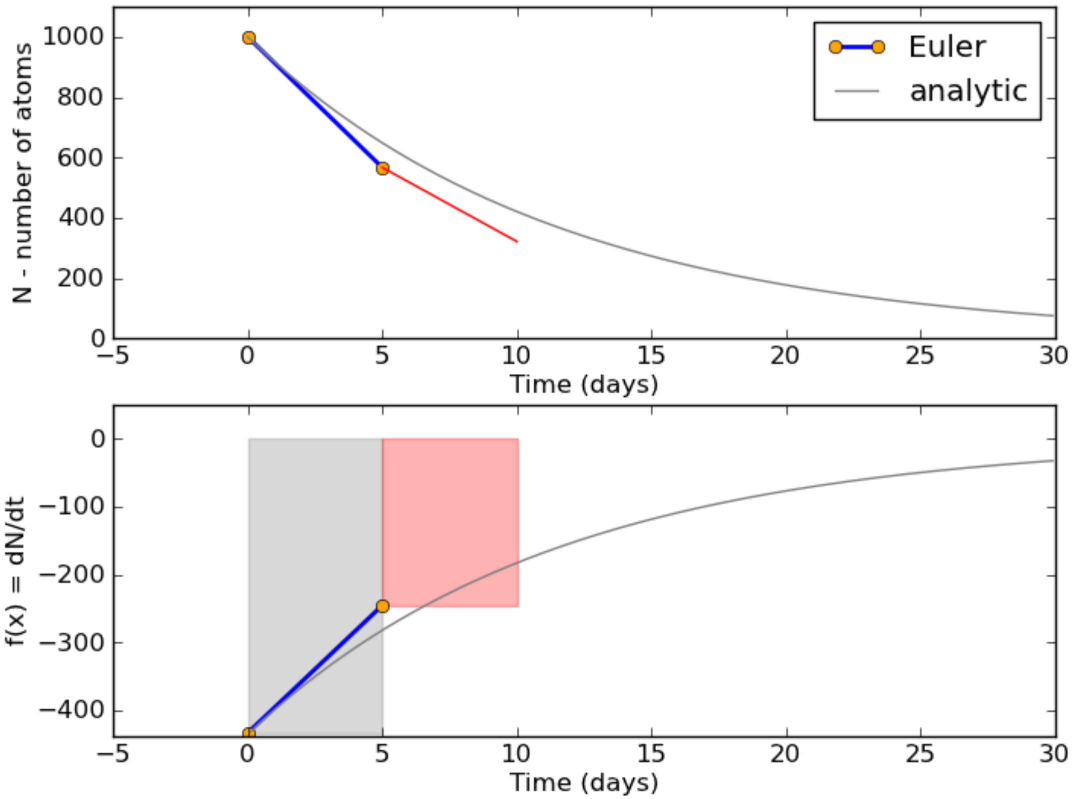
5. Perform the timestep
$N_1 = N_0 + (dN/dt)\Delta t$



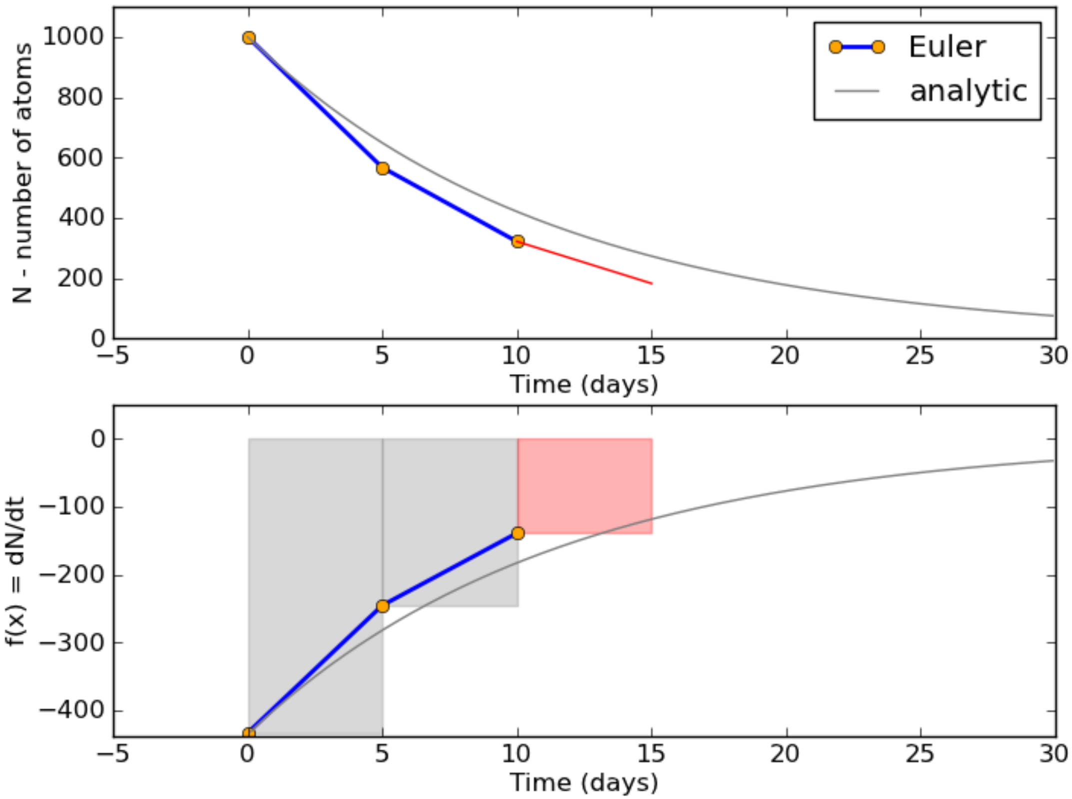
4. $dN = (dN/dt)\Delta t$

3. Assume dN/dt constant

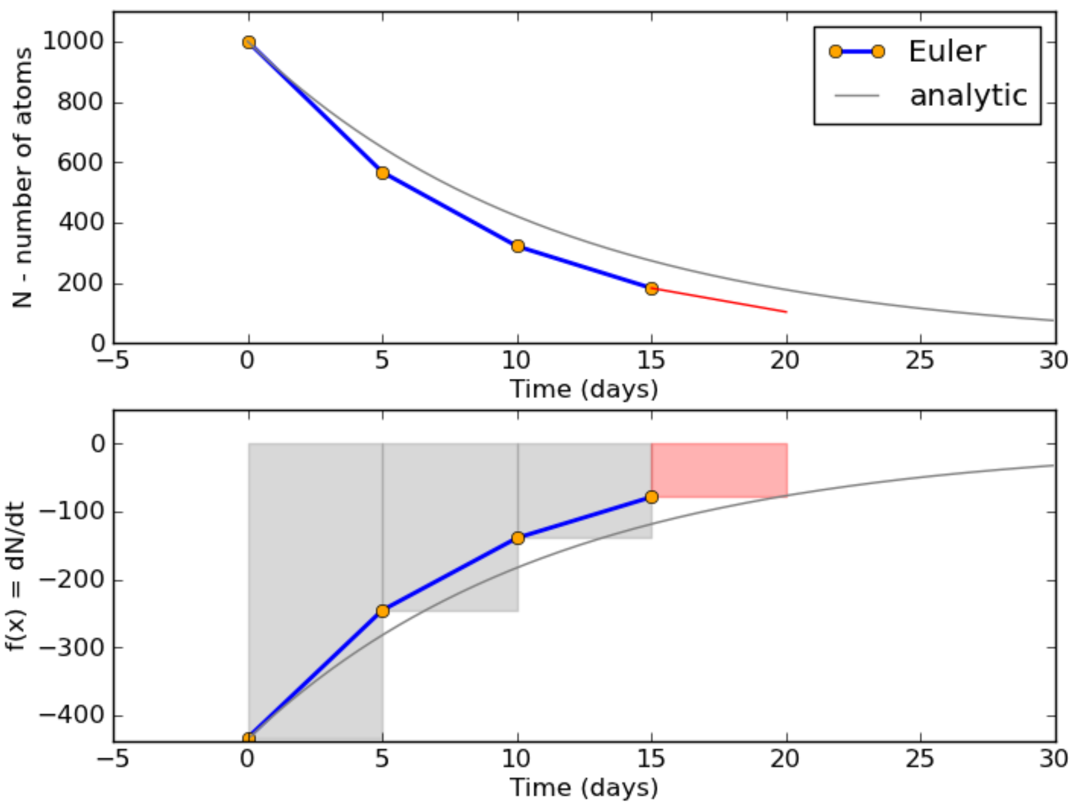
- rectangle rule



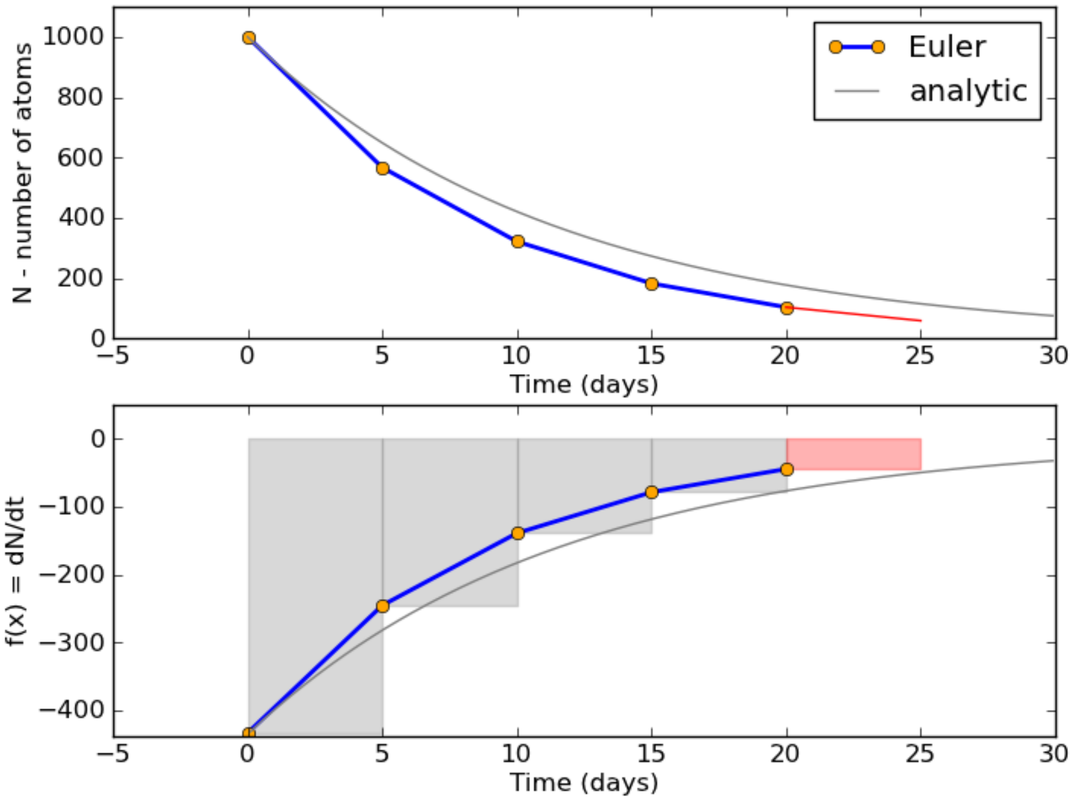
Euler
analytic
N - number of atoms
Time (days)
f(x) = dN/dt
Time (days)



Euler
analytic
N - number of atoms
Time (days)
f(x) = dN/dt
Time (days)



Euler
analytic
N - number of atoms
Time (days)
f(x) = dN/dt
Time (days)

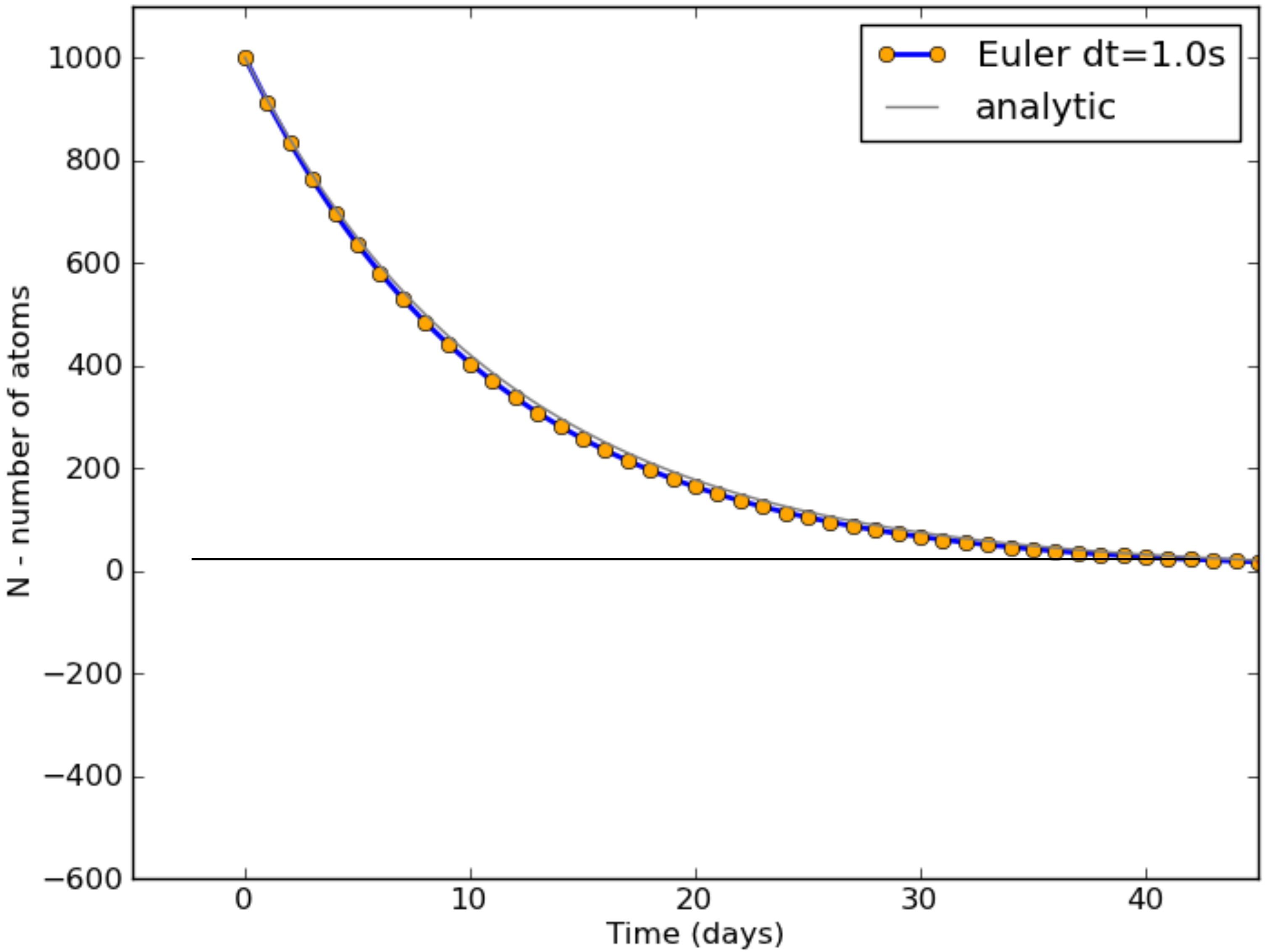


Euler
analytic
N - number of atoms
1000
800
600
400
200
0
−5
0
5
10
15
20
25
30
Time (days)
f(x) = dN/dt
0
−100
−200
−300
−400
Time (days)

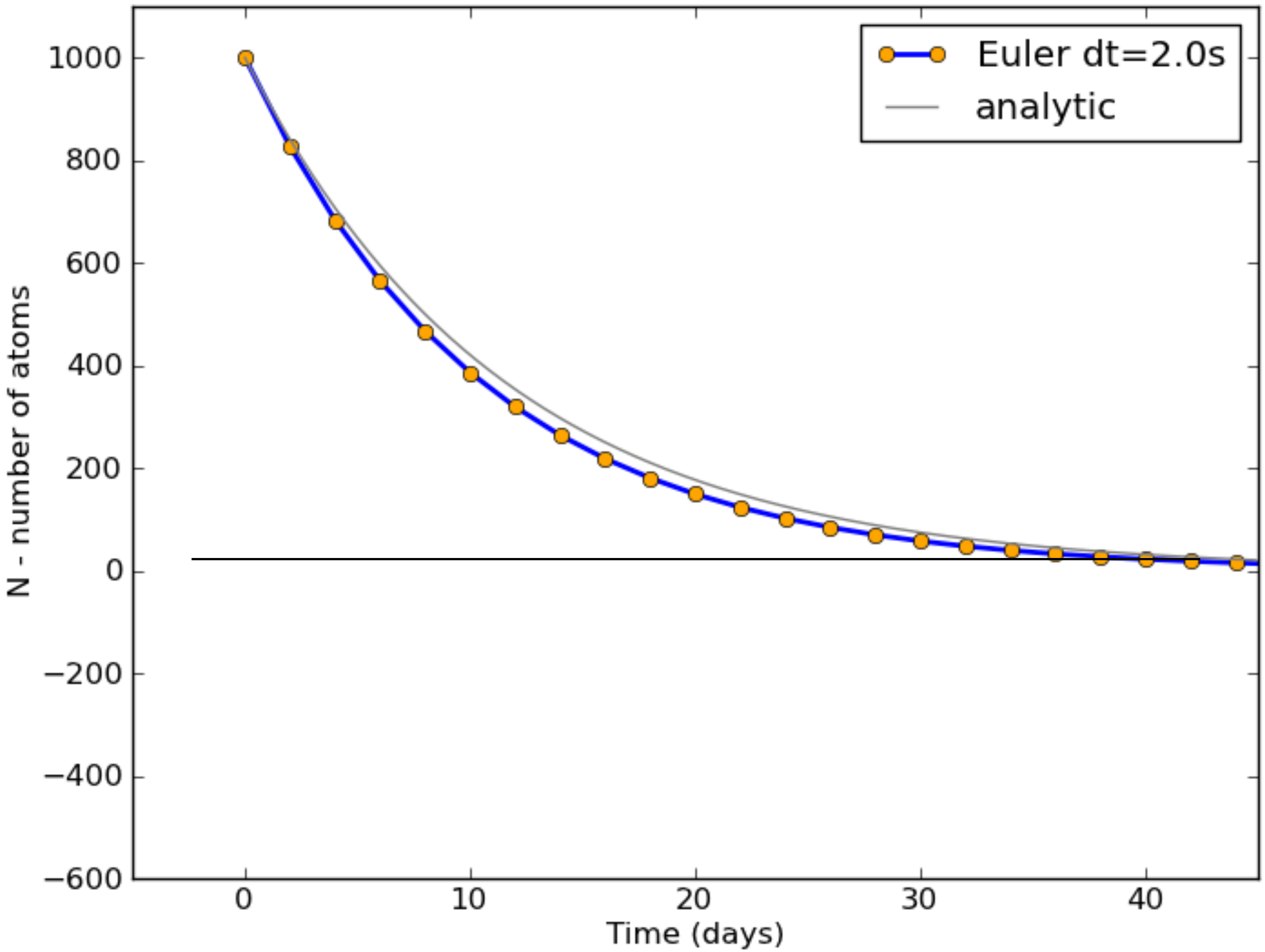
Instability

Instability

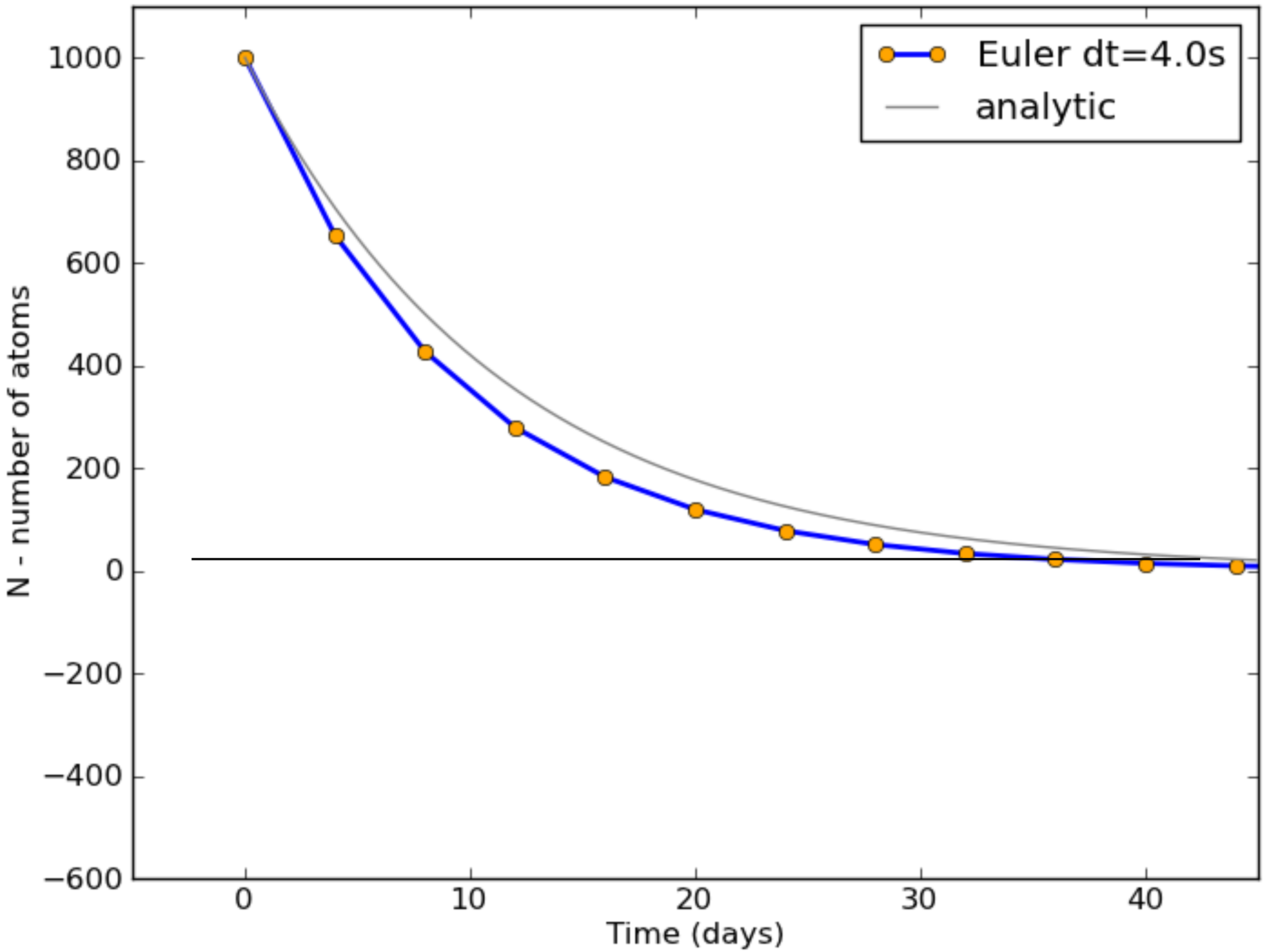
- If your step size (Δt) is to large, the assumptions we made break down and the numerical solution becomes *numerically unstable*
- If an equation is susceptible to numerical instability, it is said to be *stiff*



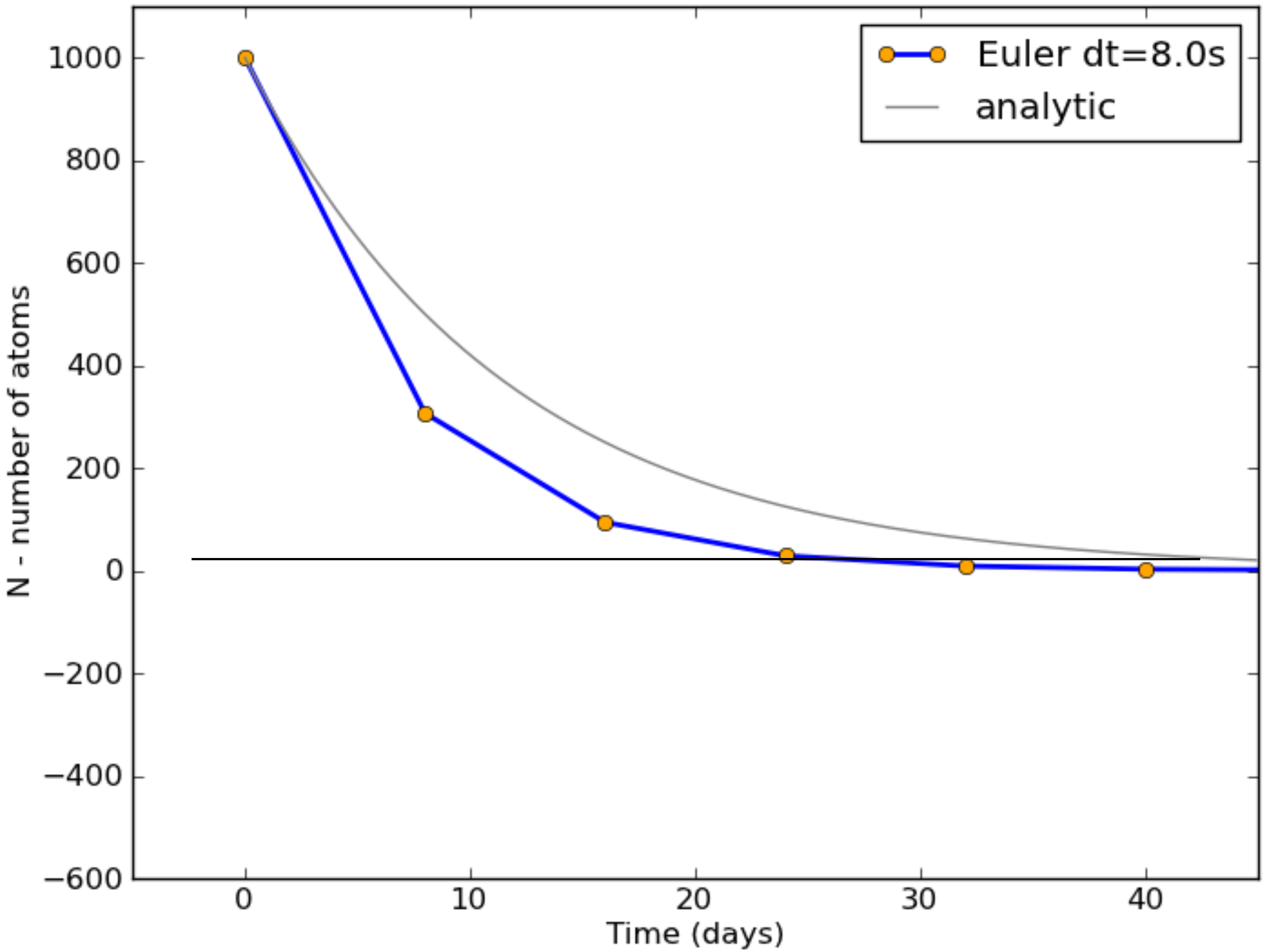
Euler dt=1.0s
analytic
N - number of atoms
Time (days)
1000
800
600
400
200
0
−200
−400
−600
0
10
20
30
40



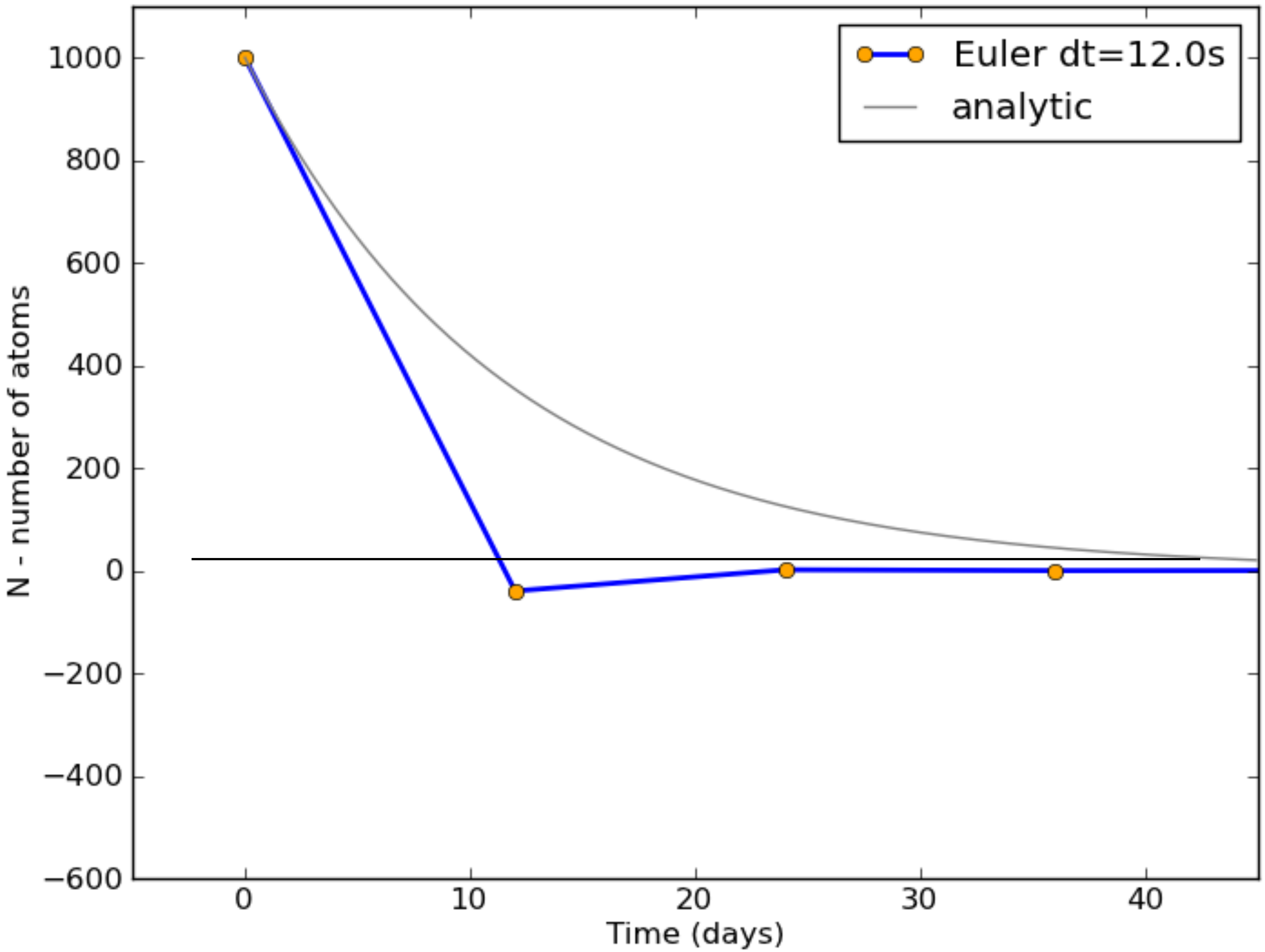
Euler dt=2.0s
analytic
1000
800
600
400
200
0
−200
−400
−600
N - number of atoms
0
10
20
30
40
Time (days)

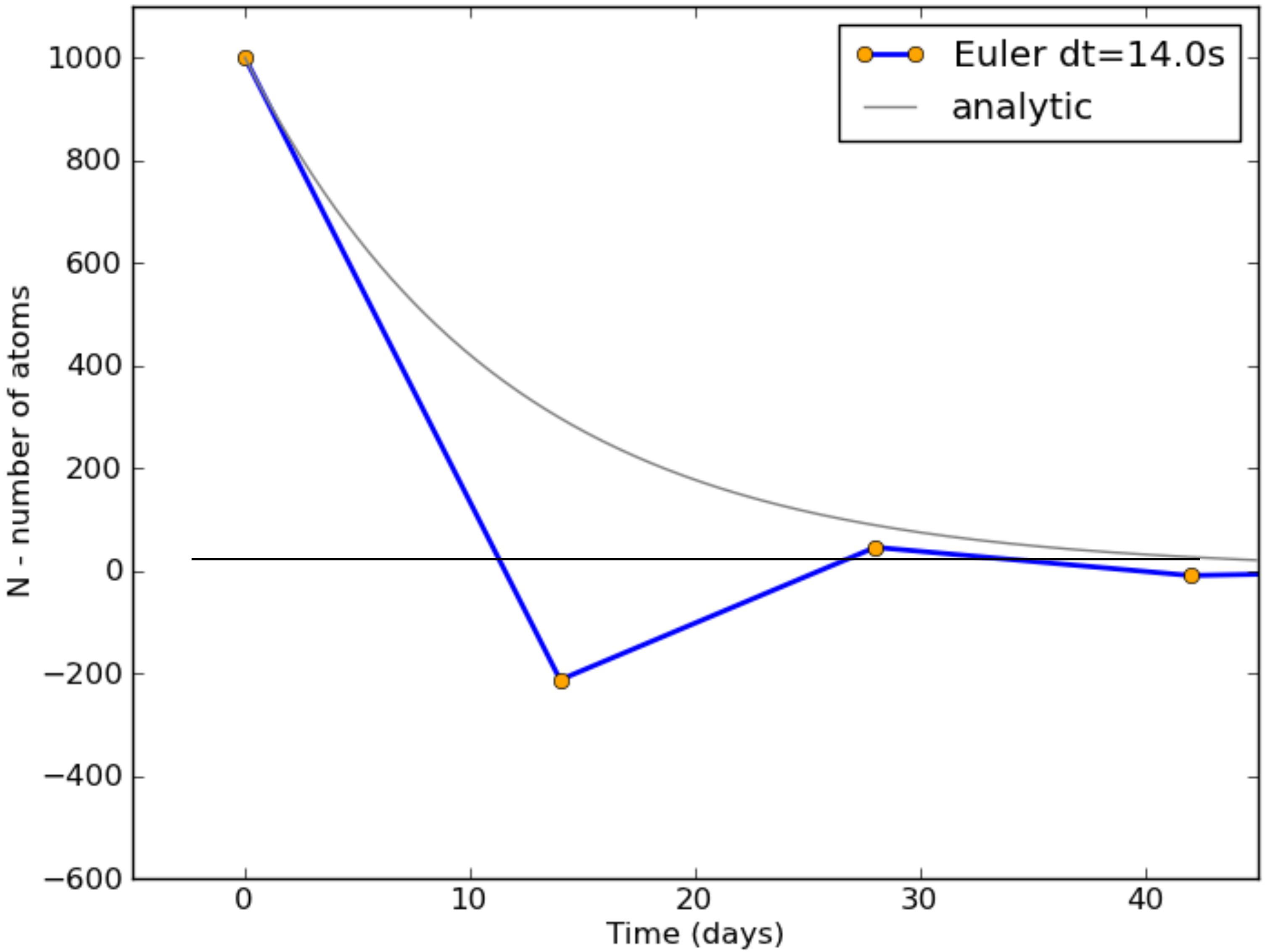


Euler dt=4.0s
analytic
N - number of atoms
Time (days)
1000
800
600
400
200
0
−200
−400
−600
0
10
20
30
40

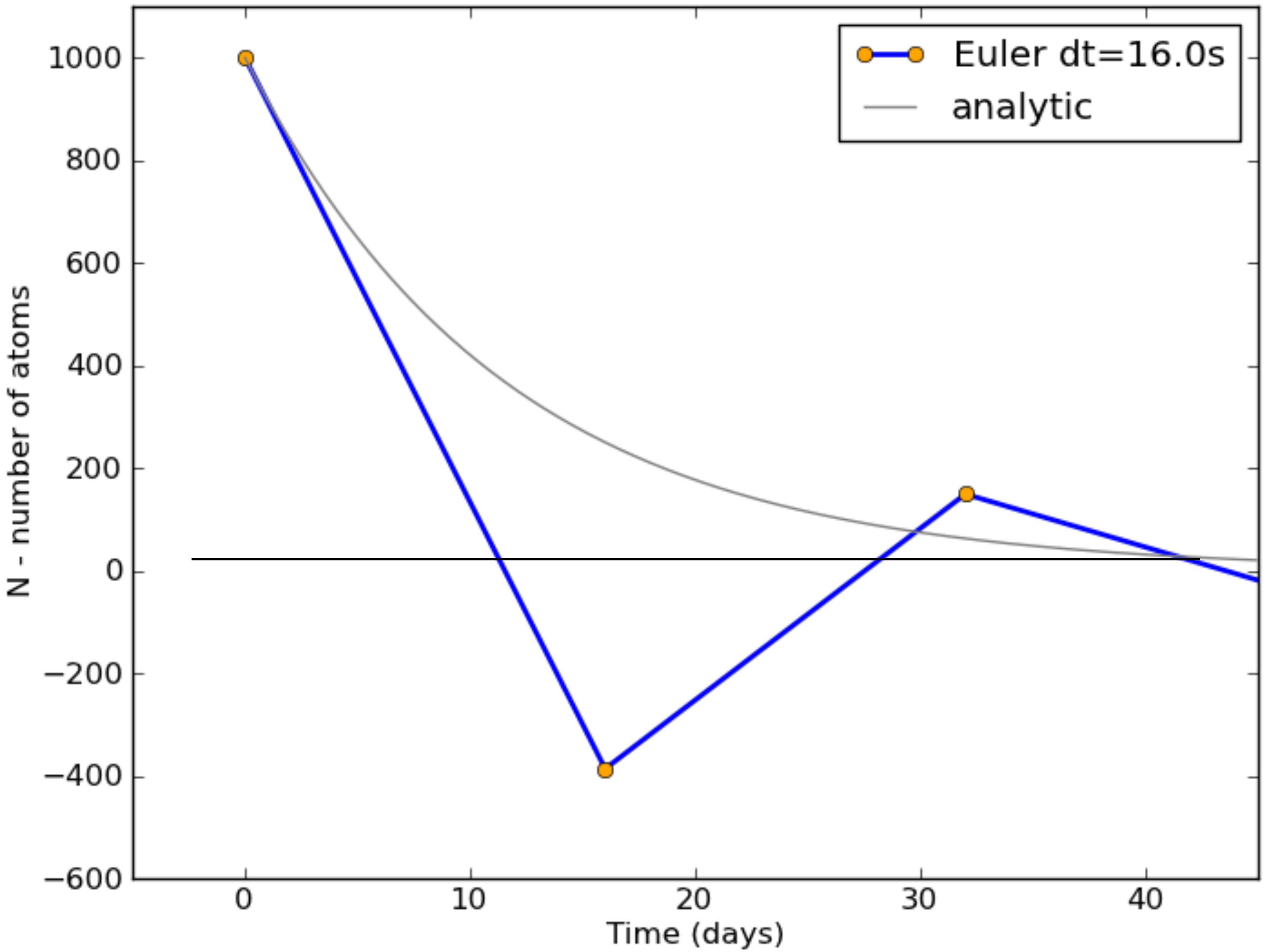


Euler dt=8.0s
analytic
N - number of atoms
Time (days)
1000
800
600
400
200
0
−200
−400
−600
0
10
20
30
40

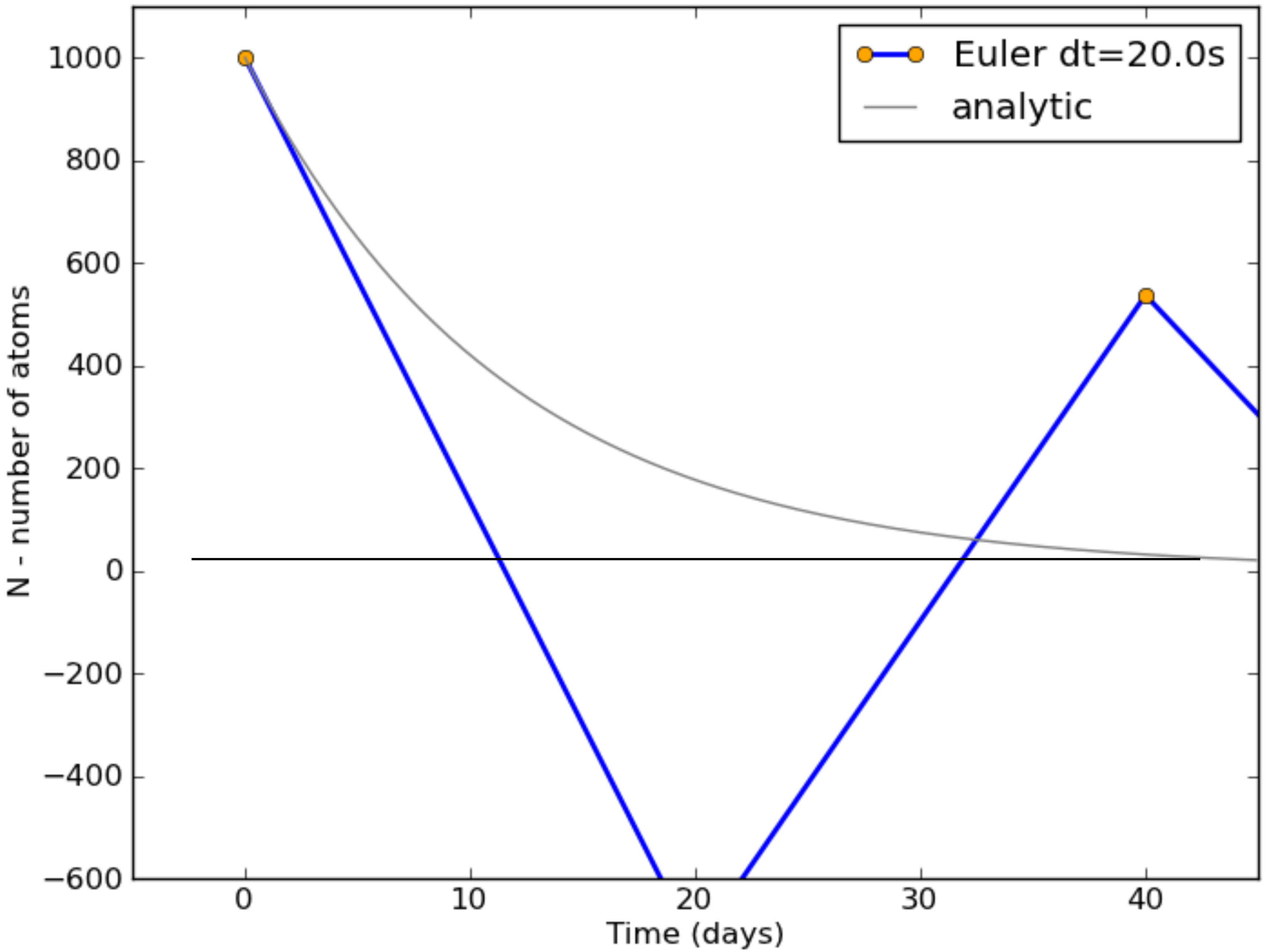

Euler dt=12.0s
analytic
N - number of atoms
Time (days)
1000
800
600
400
200
0
−200
−400
−600
0
10
20
30
40



Euler dt=14.0s
analytic
1000
800
600
400
200
0
−200
−400
−600
N - number of atoms
0
10
20
30
40
Time (days)



Euler dt=16.0s
analytic
1000
800
600
400
200
0
−200
−400
−600
N - number of atoms
0
10
20
30
40
Time (days)


Euler dt=20.0s
analytic
1000
800
600
400
200
0
−200
−400
−600
N - number of atoms
0
10
20
30
40
Time (days)

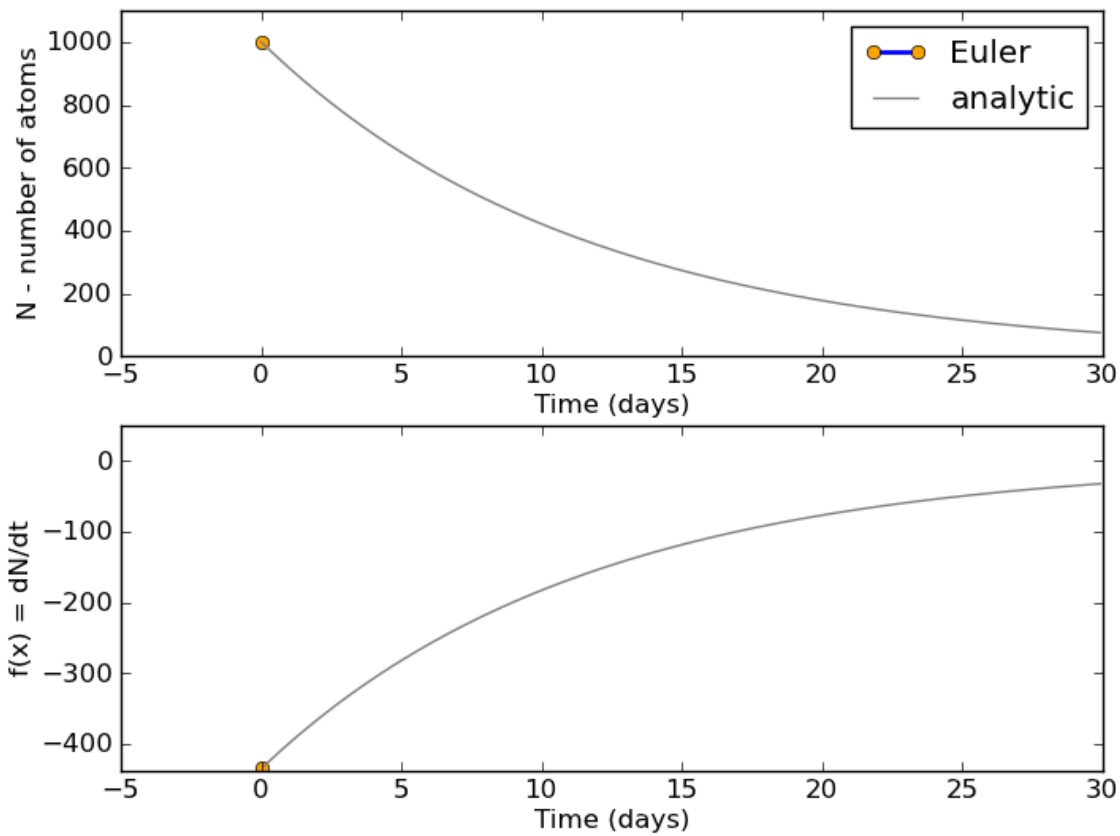
Runge-Kutta

Runge-Kutta

- The Runge-Kutta methods are a *family* of techniques for solving ODEs
- Often people refer to the specific 'RK4' as 'the Runge Kutta'
- These methods are 'predictor-corrector' methods
  - *Predictor – a* first rough estimate of the next timestep
  - *Corrector* – refine the approximation
- They offer better scaling of error vs computational cost than Euler

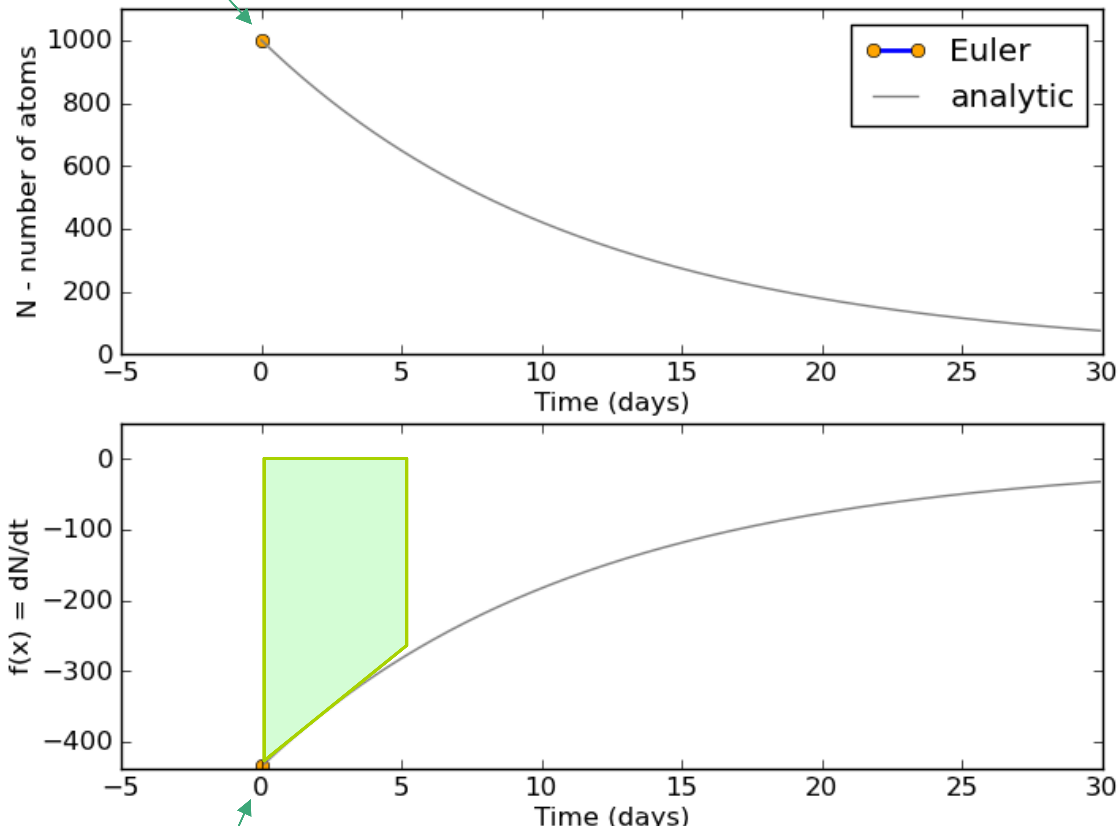
Heun's method – an RK2

- We want to improve upon Euler's method
- If we could integrate the DEQ with trapeziums instead of rectangles we would get more accurate results
- But we only know X(t=0) and dx/dt at time t=0 (left of trapezium) and not t=dt (right of trapezium)



Euler
analytic
N - number of atoms
Time (days)
f(x) = dN/dt
Time (days)

1. Initial conditions – N_0



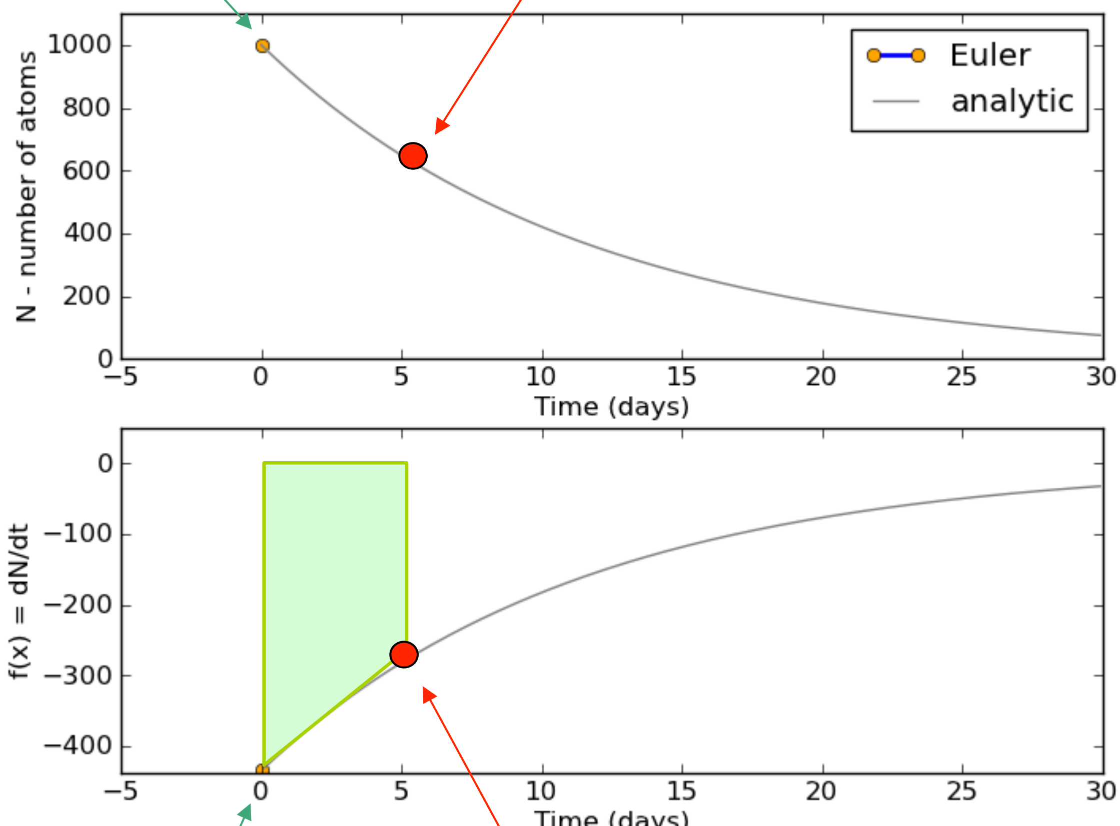
$-N_0/\tau$

At t=0 we know N_0 at and can therefore find dN/dT

The Problem

1. Initial conditions – N_0

$N_{\Delta t}$



$-N_0/\tau$

$-N_{\Delta t} / \tau$

At t=0 we know N_0 at and can therefore find dN/dT

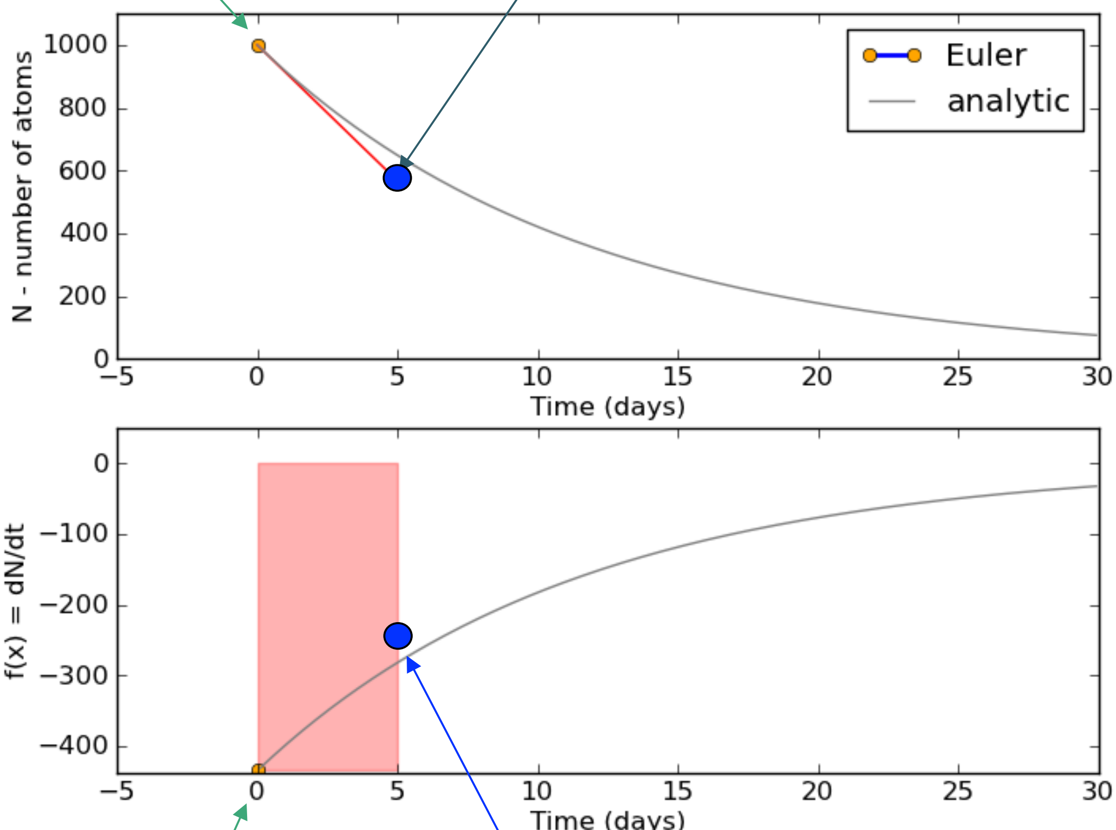
However, we don't know N at t=Δt (that's what we're trying to find!) and therefore we don't know dN/dt at t=Δt.

So we don't know the height of the right hand side of our trapezium!

The Solution

1. Initial conditions – N_0

$N'_{\Delta t}$



$-N_0/\tau$ **$-N'_{\Delta t} / \tau$**

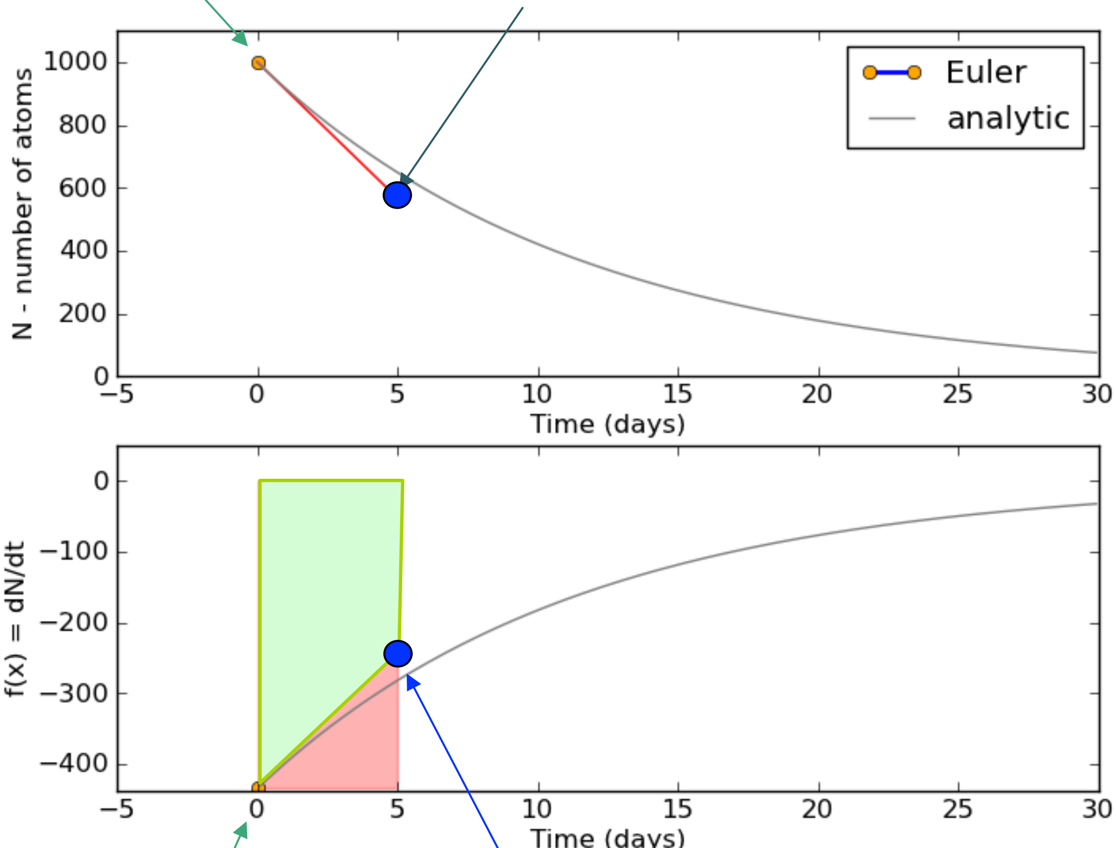
We use Euler's method to *predict* a value for N at t=Δt, which we call N'

We then approximate dN/dt at t=Δt with dN' / dt

The Solution

1. Initial conditions – N_0

$N'_{\Delta t}$

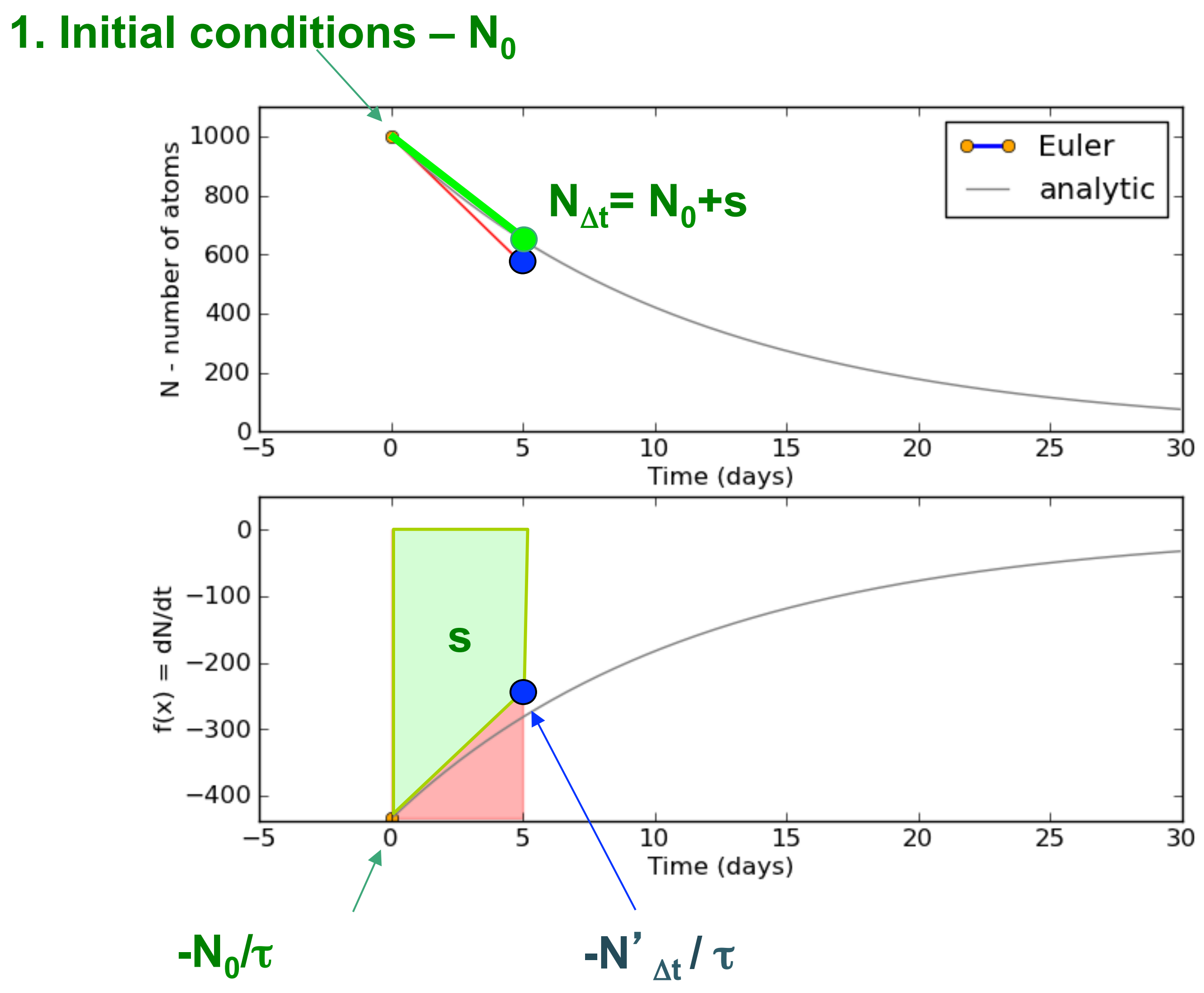


$-N_0/\tau$

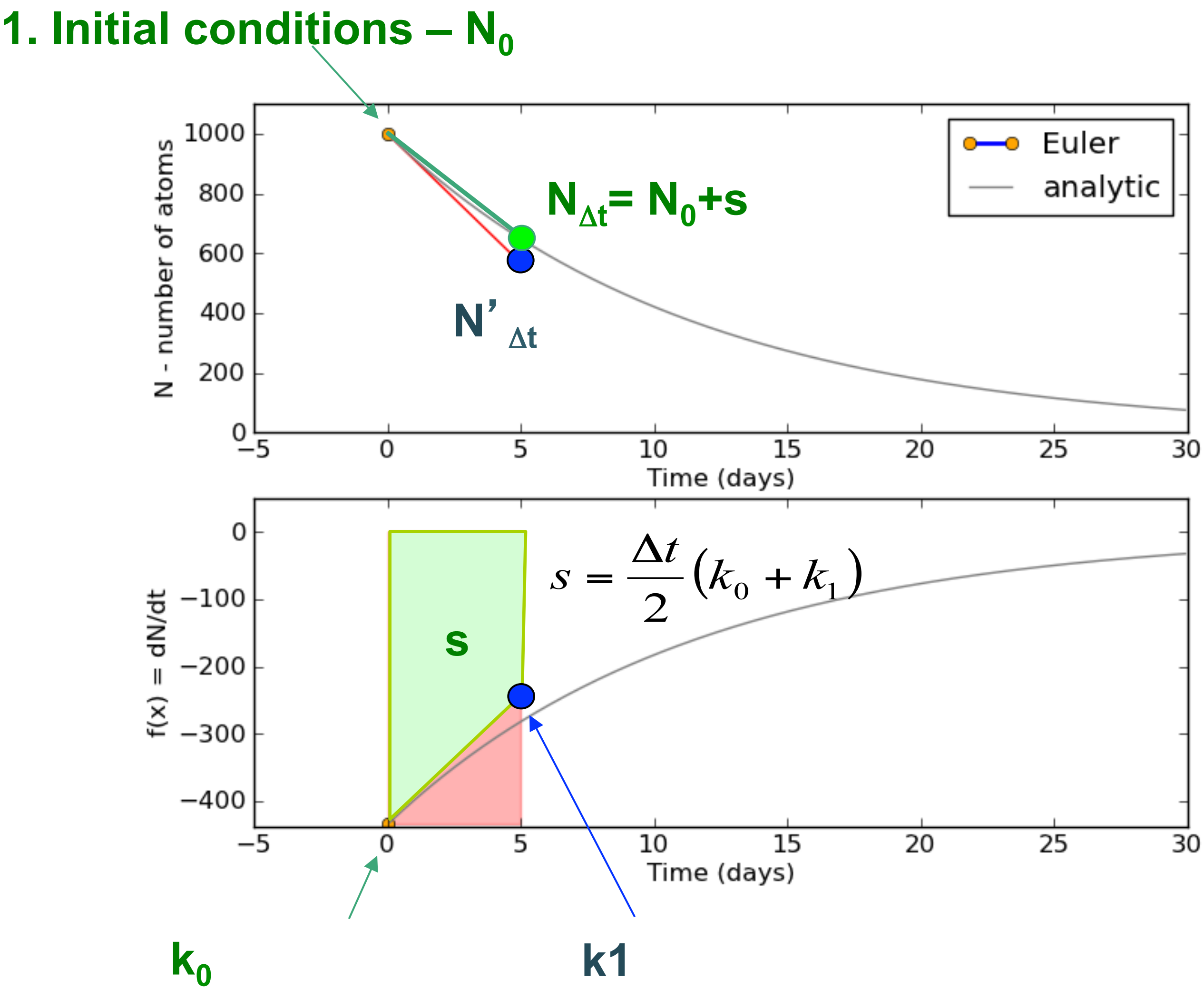
$-N'_{\Delta t} / \tau$

This first order estimate allows us to define the trapezium which we then use to refine our estimate of N at $t=\Delta t$

The Solution



The Solution



$$k_0 = f(N_0)$$

$$k_1 = f(N_0 + k_0\Delta t)$$

$$N_{\Delta t} = N_0 + \frac{\Delta t}{2}(k_0 + k_1)$$

Heun / RK2 Recap

- $dx/dt=f(x,t)$
- $k0=f(x(t),t)$ estimate rate at time t, xt
- Use this estimate to find x at t+dt
- $dx=k0\ dt$ so $x(t+dt)= x(t)+k0\ dt$
- Use this to estimate rate at time t+dt
- $k1=f(x(t+dt),t+dt)$
- $x(t+dt) = x(t) + (k0+k1)\ dt/2$

Runge Kutta RK4

RK4

- RK4 is commonly used
- It is often referred to as just 'the Runge Kutta'
- This is broadly analogous to Simpson's rule as applied to DEQs
- If you use RK4 with a standard equation – f(t) – it simplifies down to Simpson's rule

Given the DEQ:

$$\frac{dx}{dt} = f(x_t, t)$$

Calculate the following intermediate variables:

$$k_1 = f(x_t, t)$$

$$k_2 = f\left(x_t + \frac{\Delta t}{2} k_1, t + \frac{\Delta t}{2}\right)$$

$$k_3 = f\left(x_t + \frac{\Delta t}{2} k_2, t + \frac{\Delta t}{2}\right)$$

$$k_4 = f(x_t + \Delta t.k_3, t + \Delta t)$$

Finally apply the timestep:

$$x_{t+\Delta t} = x_t + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Accuracy

- Two types of error
  - Error per step (step size of Δt)
  - Total error

- Smaller steps means
  - Less error per step
  - but worse total error because more steps!

Method name	Function evaluations	Error per step (order)	Total error (order)
Euler	1	Δt^2	Δt
Heun	2	Δt^3	Δt^2
RK4	4	Δt^5	Δt^4

Weekly Problem 3

Weekly Assessments...

- Computational Physics has:
  - No exam
  - No ‘lab report’

- Just weekly assessments
  - So far they are going well

- You should spend perhaps
  - 0.5-2 hours/week on the problem as homework
  - 1 hour/week in your allotted workshop
  - BUT MORE CHALLENGING AS WE GO ON

Weekly Problem 3

- Implementing Euler's method and Heun's method solvers for radioactive decay
- Comparing analytical and numerical models
- Plotting
  - Decay curves
  - Error of numerical methods

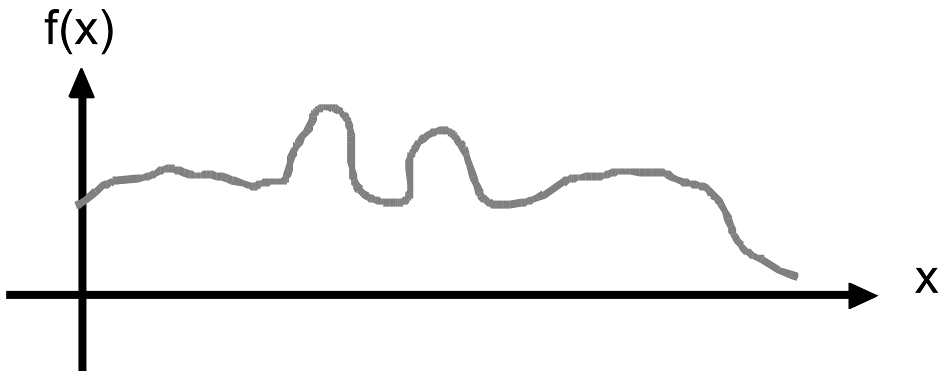
Preparation

- Put some time into the problem before your workshop session
  - Then you will know where you need help at the start of the session...
  - ... so you are much more likely to get that help and benefit from the session
- Read the whole problem
  - Look at the example code before you start
  - Draw parallels to the work you have already done
  - Look at past model solutions
- Get Euler working first!
- WORKSHOPS GIVE YOU HELP!

Adaptive Step Size

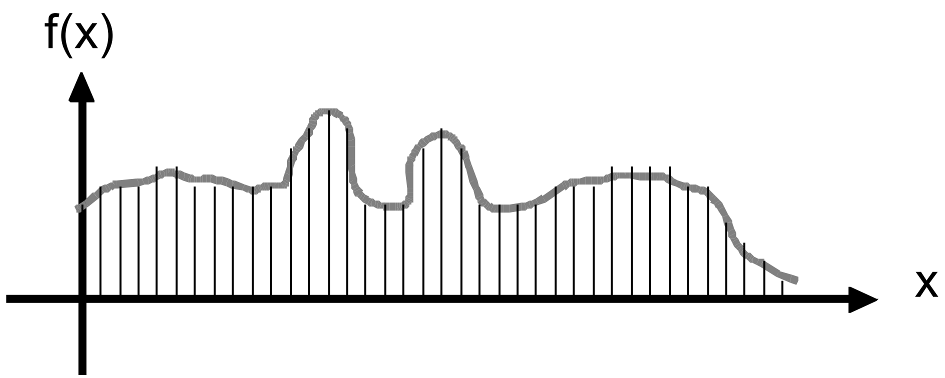
The problem

- Imagine a function to be integrated or an ODE to be solved
  - Some areas vary rapidly
  - Some vary slowly
- We want to integrate to some specified accuracy



The problem

- Rapidly changing areas need small panels to be integrated accurately
- These small panels aren't needed for slowly varying areas
  - These unneeded panels slow us down (Speed is king!)



The Solution

- *Adaptive Step Size*
- We adjust the size/density of panels to match the function
  - Tolerance driven
  - Examine derivatives – more curvature => smaller panels