5. Newton's law of cooling

a) $$\frac{dT(t)}{dt} = -K\,(T(t) - T_e)$$

$$\int_0^t \frac{dT(t)}{T(t) - T_e} = \int_0^t -K\,dt$$

$$\left[\ln\,(T(t) - T_e)\right]_0^t = \left[-Kt\right]_0^t$$

$$\ln\left(\frac{T(t) - T_e}{T(0) - T_e}\right) = -Kt$$

$$\frac{T(t) - T_e}{T(0) - T_e} = e^{-Kt}$$

$$T(t) = T_e + (T(0) - T_e)\,e^{-Kt}$$

b) (1) $$\frac{dT(t)}{dt} = -K\,(T(t) - T_e)$$

EF: (2) $$\frac{dT(t)}{dt} = \frac{T(t+\Delta t) - T(t)}{\Delta t}$$

$\xrightarrow{(1)=(2)}$ $$-K\,u(t) = \frac{T(t+\Delta t) - T(t)}{\Delta t}$$

$$-K\,u(t) = \frac{u(t+\Delta t) - u(t)}{\Delta t}$$

$$u(t+\Delta t) = u(t)\,(1 - K\Delta t)$$

$$u(m\,\Delta t) = u(0)\,(1 - K\Delta t)^m$$

$\xrightarrow{\text{re substitution}}$ $$\boxed{T(m\,\Delta t) = T_e + (T_0 - T_e)\,(1 - K\Delta t)^m}$$

c) $T(m\Delta t) = T_e + (T_0 - T_e)\,(1 - K\Delta t)^m$

Stability cond. $|1 - K\Delta t| \leq 1$

$0 \leq 2 - K\Delta t \leq 2$

$\rightarrow K\Delta t \geq 0 \;\wedge\; K\Delta t \leq 2$

$\rightarrow$ conditionally stable !

d) see at the last part of the pdf

e) $$-K\,u(t+\Delta t) = \frac{T(t+\Delta t) - T(t)}{\Delta t}$$

$$-K\,u(t+\Delta t) = \frac{u(t+\Delta t) - u(t)}{\Delta t}$$

$$u(t) = u(t+\Delta t)\,(1 + K\Delta t)$$

$$u(t+\Delta t) = \frac{u(t)}{1 + K\Delta t}$$

$$u(m\Delta t) = \frac{u(0)}{(1 + K\Delta t)^m}$$

resub. $\rightarrow$ $$\boxed{T(m\Delta t) = T_e + \frac{T(0) - T_e}{(1 + K\Delta t)^m}}$$

stability cond. $\underbrace{|1 + K\Delta t|}_{\text{both positive}} \geq 1$

$\rightarrow$ always fullfilled $\rightarrow$ unconditionally stable !

f) Taylor expansion

$$f(t+\Delta t) = f(t) + \frac{d}{dt} f(t)\Delta t + \frac{1}{2}\frac{d^2}{dt^2} f(t)\,\Delta t^2 + O(\Delta t^3)$$

$$f(t) = T_e + (T_0 - T_e)e^{-Kt}$$

$$\frac{d}{dt} f(t) = (T_0 - T_e)(-K)e^{-Kt}$$

$$\frac{d^2}{dt^2} f(t) = (T_0 - T_e)K^2 e^{-Kt}$$

$$\boxed{\begin{aligned} P_1(t) &= (T_0 - T_e)e^{-Kt}(1 - K\Delta t) + T_e \\ P_2(t) &= (T_0 - T_e)e^{-Kt}(1 - K\Delta t + K^2\Delta t^2) + T_e \end{aligned}}$$

As we see in the plot $P_2(t)$ is sligthly closer to the analytical solution (higher order = higher accuracy)

6. a) $f(N(t)) = \frac{N(t+\Delta t) - N(t)}{\Delta t}$

$N(t) \to N(t) + \delta N(t)$

$$f(N(t) + \delta N(t)) = \frac{N(t+\Delta t) + \delta N(t+\Delta t) - N(t) - \delta N(t)}{\Delta t}$$

$$f(N(t) + \delta N(t)) = \underbrace{\frac{N(t+\Delta t) - N(t)}{\Delta t}}_{= f(N(t))} + \frac{\delta N(t+\Delta t) - \delta N(t)}{\Delta t}$$

$$f(N(t) + \delta N(t)) - f(N(t)) = \frac{\delta N(t+\Delta t) - \delta N(t)}{\Delta t}$$

$$\Delta t \quad \underbrace{\frac{f(N(t) + \delta N(t)) - f(N(t))}{\delta N(t)}}_{= \frac{df}{dN}} = \frac{\delta N(t+\Delta t)}{\delta N(t)} - 1$$

$$N(t+\Delta t) = N(t)\left(\Delta t \frac{df}{dN} + 1\right)$$

---

b) $\cancel{N(t+\Delta t) = N(t)\left(\Delta t\, 2N + 1\right)}$

Stability cond $\quad |1 + \Delta t \; 2N| < 1$

(1) $\quad 1 + \Delta t\, 2N < 1$

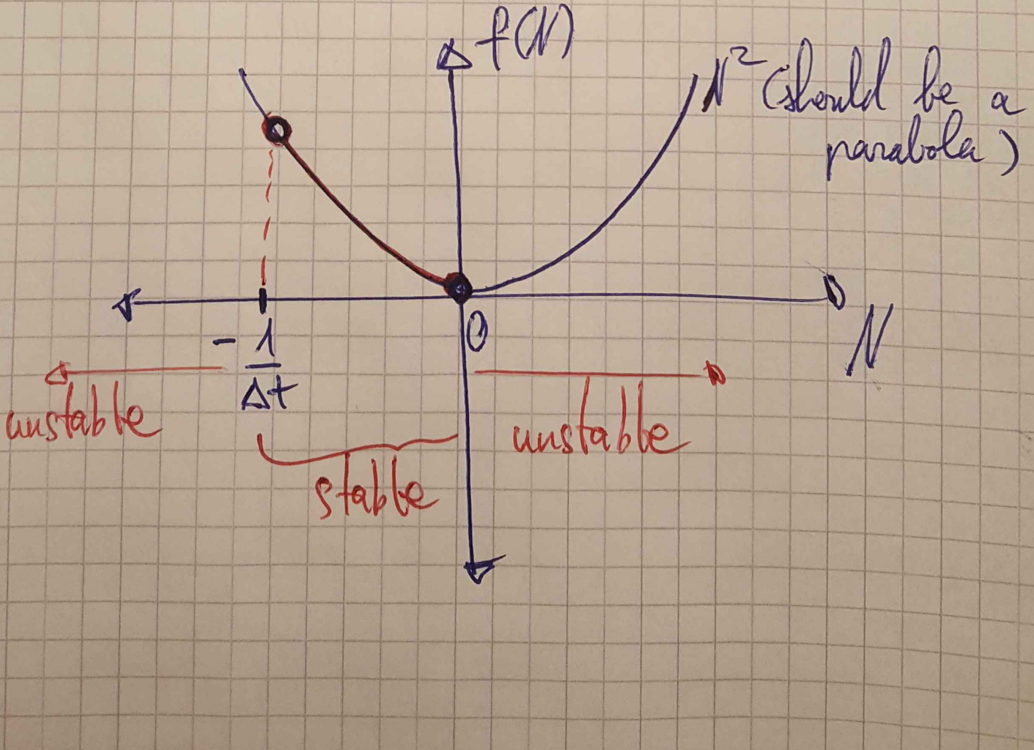
$\Delta t\, 2N < 0$

$\underline{N < 0}$

(2) $\quad -1 - \Delta t\, 2N < 1$

$-\Delta t\, 2N < 2$

$\underline{N > -\frac{1}{\Delta t}}$



sheet4

September 21, 2017

1 5.

1.1 d)

```
In [1]: using PyPlot
```

```
In [2]: #ef marching eq
        ef(x,t0,te,dt,k)=te+(t0-te)*(1.-k*dt)^x
        #analytical sol
        analytical(x,t0,te,k)=te+(t0-te)*exp(-k*x)
```

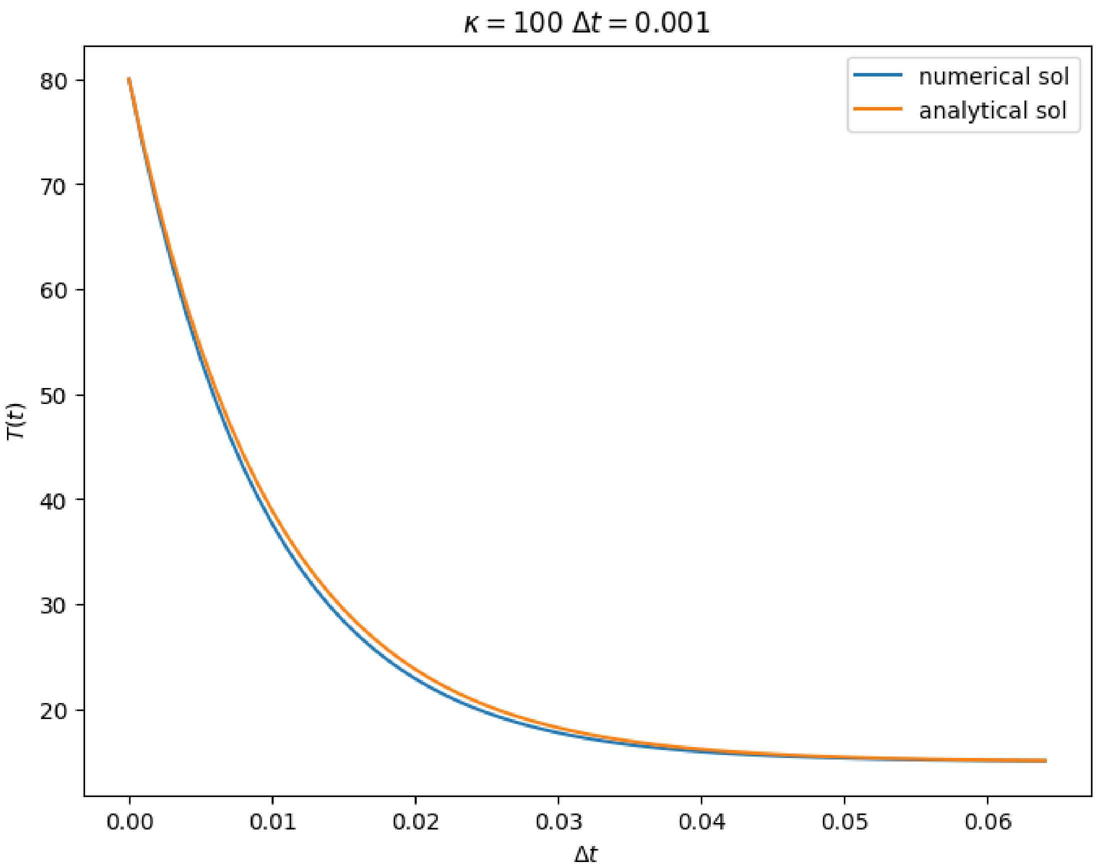
```
Out[2]: analytical (generic function with 1 method)
```

$\kappa\Delta t \geq 0 \wedge \kappa\Delta \leq 2$
e.g. $\rightarrow \kappa = 100 \wedge \Delta t = 0.001$

```
In [3]: #parameters
        k=100.
        dt=0.001
        t0=80
        te=15
        tend=te+0.1
        tvalues=Float64[]
        xvalues=Float64[]
        anavalues=Float64[]
        #x-step, corresponds to m
        x=0.
        #time out of step
        t(x)=x*dt
        #numerical temp
        tnow=ef(x,t0,te,dt,k)
        #analytical
        ana=analytical(x,t0,te,k)
        println(ana)
        #until the analytical is aboth the wished
        while(ana>=tend)
            push!(tvalues,tnow)
            push!(anavalues,ana)
```

```
    push!(xvalues,x)
    #increases timestep
    x+=1
    tnow=ef(x,t0,te,dt,k)
    ana=analytical(t(x),t0,te,k)
end

#plot
figure(1,figsize=(8,6))
plot(xvalues*dt,tvalues,label="numerical sol")
plot(xvalues*dt,anavalues,label="analytical sol")
title(L"$\kappa = 100$ $\Delta t=0.001$")
ylabel(L"$T(t)$")
xlabel(L"$\Delta t$")
legend()
```




```
80.0
```

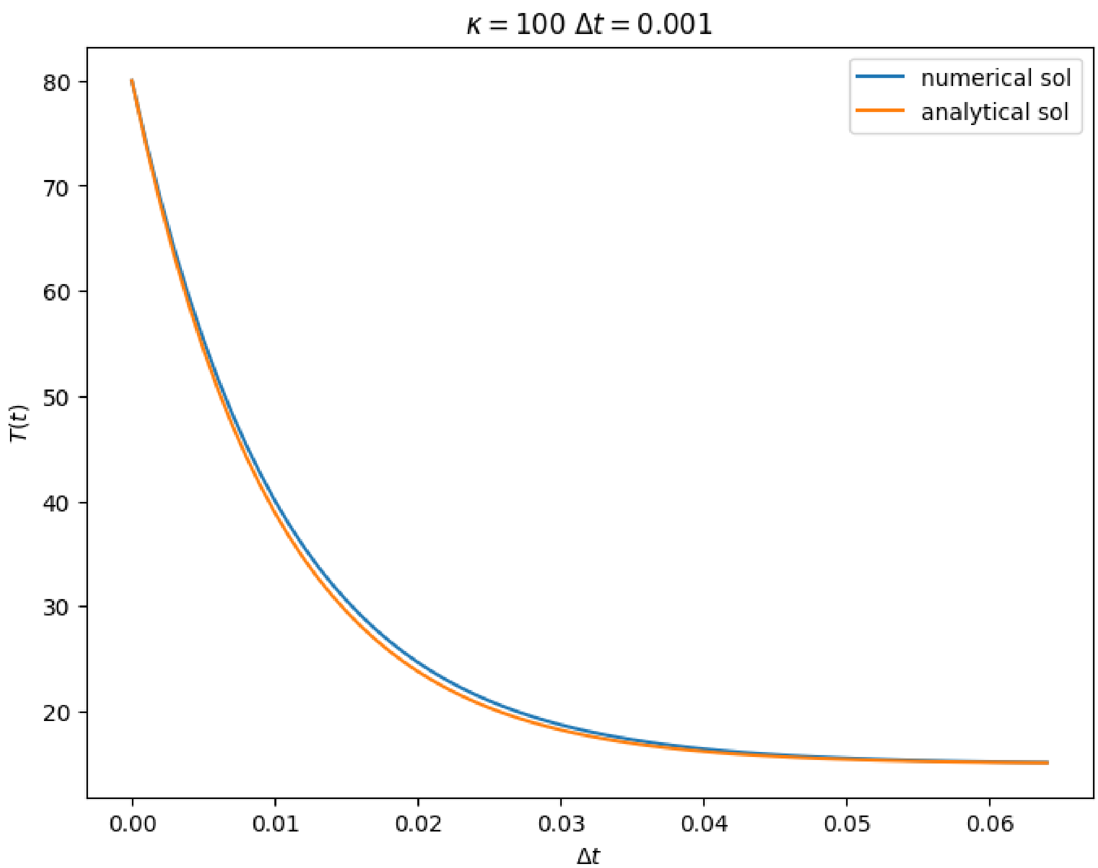
```
Out[3]: PyObject <matplotlib.legend.Legend object at 0x7f4729944890>
```

1.2 e)

```
In [4]: #eb-scheme
        eb(x,t0,te,dt,k)=te+(t0-te)/(1+k*dt)^x

        #parameters
        k=100.
        dt=0.001
        t0=80
        te=15
        tend=te+0.1
        tvalues=Float64[]
        xvalues=Float64[]
        anavalues=Float64[]
        #x-step, corresponds to m
        x=0.
        t(x)=x*dt
        tnow=eb(x,t0,te,dt,k)
        ana=analytical(x,t0,te,k)
        println(ana)
        #until the analytical is aboth the wished
        while(ana>=tend)
            push!(tvalues,tnow)
            push!(anavalues,ana)
            push!(xvalues,x)
            x+=1
            tnow=eb(x,t0,te,dt,k)
            ana=analytical(t(x),t0,te,k)
        end

        #plot
        figure(1,figsize=(8,6))
        plot(xvalues*dt,tvalues,label="numerical sol")
        plot(xvalues*dt,anavalues,label="analytical sol")
        title(L"$\kappa = 100$ $\Delta t=0.001$")
        ylabel(L"$T(t)$")
        xlabel(L"$\Delta t$")
        legend()
```




```
80.0
```

```
WARNING: Method definition t(Any) in module Main at In[3]:13 overwritten at In[4]:15.
```

Out[4]: PyObject <matplotlib.legend.Legend object at 0x7f47297c4e50>

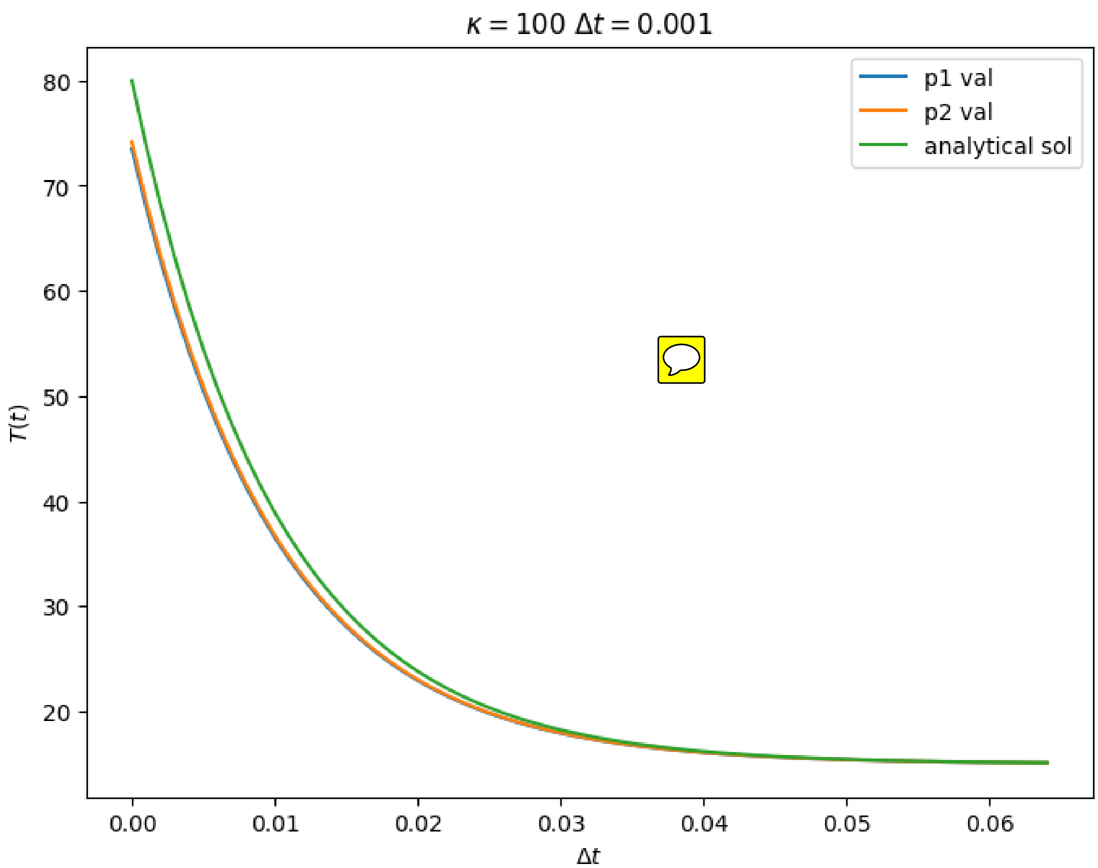
1.3 f)

In [5]:
```
#the 2 polynomials
p1(x,t0,te,dt,k)=(t0-te)*exp(-k*x)*(1-k*dt)+te
p2(x,t0,te,dt,k)=(t0-te)*exp(-k*x)*(1-k*dt+k^2*dt^2)+te

#parameters
k=100.
dt=0.001
t0=80
te=15
tend=te+0.1
```

```
#values
xvalues=Float64[]
p1values=Float64[]
p2values=Float64[]
anavalues=Float64[]
x=0.
t(x)=x*dt
t1=p1(t(x),t0,te,dt,k)
t2=p2(t(x),t0,te,dt,k)
ana=analytical(x,t0,te,k)
println(ana)
while(ana>=tend)
    push!(p1values,t1)
    push!(p2values,t2)
    push!(anavalues,ana)
    push!(xvalues,x)
    x+=1
    t1=p1(t(x),t0,te,dt,k)
    t2=p2(t(x),t0,te,dt,k)
    ana=analytical(t(x),t0,te,k)
end

#plot
figure(1,figsize=(8,6))
plot(xvalues*dt,p1values,label="p1 val")
plot(xvalues*dt,p2values,label="p2 val")
plot(xvalues*dt,anavalues,label="analytical sol")
title(L"$\kappa = 100$ $\Delta t=0.001$")
ylabel(L"$T(t)$")
xlabel(L"$\Delta t$")
legend()
```



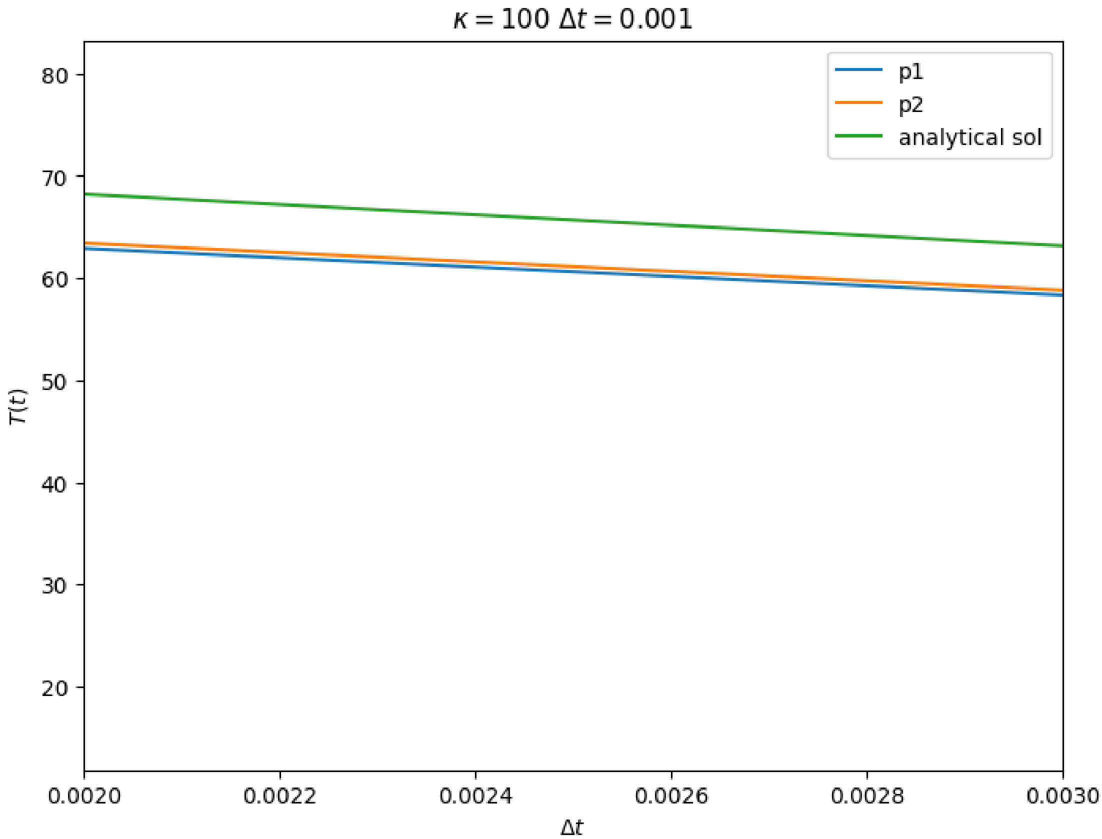

```
80.0
```

```
WARNING: Method definition t(Any) in module Main at In[4]:15 overwritten at In[5]:17.
```

Out[5]: PyObject <matplotlib.legend.Legend object at 0x7f472971c990>

I want to see, which polynomial is closer to the analytical result, therefore i zoom in a bit

In [12]:
```
#plot
figure(1,figsize=(8,6))
plot(xvalues*dt,p1values,label="p1")
plot(xvalues*dt,p2values,label="p2")
plot(xvalues*dt,anavalues,label="analytical sol")
title(L"$\kappa = 100$ $\Delta t=0.001$")
ylabel(L"$T(t)$")
xlabel(L"$\Delta t$")
legend()
#rescalement of the plot
ax=gca()
ax[:set_xlim]([.002,.003])
```



Out[12]: (0.002,0.003)

In [ ]: