Decay kinetics for peroxides to estimate the residual amount

Dieter Schuster
SAFE+ Algorithmics GmbH

30. Januar 2023

This document[1] describes how a residual amount of a peroxide used as a polymer initiator can be estimated from the temperature-induced decay. The residual amount can then be used for migration simulation.

1 Example decomposition kinetics for peroxides (as polymerization initiators)

For decay kinetics with decay rate k the Arrhenius approach can be used

$$k = Ae^{-\frac{E_A}{RT}}, \tag{1}$$

with the Arrehnius-Parameters A and E_A and the absolute temperature T and the universal gas constant R. The decay at time t is determined by the first order kinetics:

$$c(t) = e^{-kt}\, c_0$$

for the used concentration c_0. Then the following applies to the half-life time $t_{\frac{1}{2}}$:

$$t_{\frac{1}{2}} = \frac{\ln 2}{k}. \tag{2}$$

Data sheets often list half-live times for the temperature induced decay. From two half-live times t_1 and t_2 at the corresponding temperatures T_1 and T_2 the Arrhenius parameters can be determined by: Substitute (2) into (1):

$$t_{\frac{1}{2}} = \frac{\ln 2}{Ae^{-\frac{E_A}{RT}}} \tag{3}$$

If we substitute t_1 and T_1 into (3) and t_2 and T_2 into (3), we get a system of two equations and with the unknowns E_A and A. The solution is:

$$E_A = R\frac{T_1T_2}{T_2 - T_1}\ln\frac{t_1}{t_2} \tag{4}$$

$$\text{and} \quad A = \frac{\ln 2}{t_2}\, e^{\frac{E_A}{RT_2}}. \tag{5}$$

Thus, for a further temperature T the decay coefficient or the half-life time can be determined from (1) or (2), by using the values A and E_A computed from (4) and (5).

[1]:
```
from math import log, exp  # log is here the natural logarithm
import matplotlib.pyplot as plt
```

universal gas constant

[2]:
```
R = 8.314 # J/(K*mol)
```

[1] This document is based on a *Jupyter notebook* with Python code, the calculation results are direct output of the Python code.

1.1 Example

For an example, the peroxide United Initiators BU-50-AL, which is based on 2,2-Di(tert.butylperoxy)butane (CAS 2167-23-9) is taken. The Safety Data Sheet can be found at: https://saunitedinitiatorsprod.blob.core.windows.net/productdata/BU-50-AL/United_Initiators_BU-50-AL_MSDS_DE_DE.pdf

The Technical data sheet can be found at: https://saunitedinitiatorsprod.blob.core.windows.net/productdata/BU-50-AL/United_Initiators_BU-50-AL_TDS_AL_EN.pdf

A chemical description is found at PubChem (https://pubchem.ncbi.nlm.nih.gov/compound/221260).

The Technical data sheet (see Revision number: 1.0. Date: HW/14.08.2015 Device M: TDS) list the following Half life Data:

10h / 1h / 1 min: 104 °C / 124 °C / 165 °C.

In the following are:

- T temperature
- th half-life time

[3]:
```
T_1 = 104 + 273.15 # K
T_2 = 165 + 273.15 # K
th_1 = 36000 # s
th_2 = 60 # s
```

Calculation of the activation energy E_A and the pre-exponential factor A

[4]:
```
E_A = R*(T_1*T_2)/(T_2-T_1)*log(th_1/th_2)
A = log(2)/th_2*exp(E_A/(R*T_2))

print(f"E_A = {E_A/1000:5.3f} kJ/mol")
print(f"A   = {A:.3g} 1/s")
```

```
E_A = 144.075 kJ/mol
A   = 1.74e+15 1/s
```

Control for mean value with 124 °C from half life time.

[5]:
```
T_3 = 124 + 273.15 # K
k_3 = A*exp(-E_A/(R*T_3))
th = log(2)/k_3
print(f"th_{T_3-273.15:.0f}C = {th:.0f} s")
```

```
th_124C = 3560 s
```

The data sheet lists 1 h = 3600 s.

Decay rate during a process for given temperatures.

[7]:
```
T = {"Polymerisation": 100 + 273.15, # K
     "Drying": 60 + 273.15, # K
     "Sinter": 360 + 273.15} # K

k = {process: A*exp(-E_A/(R*temperature)) for process, temperature in T.items()}

for process, temperature in k.items():
    print(f"{process:15}: k = {temperature:0.2g} 1/s")
```

```
Polymerisation : k = 1.2e-05 1/s
Drying         : k = 4.5e-08 1/s
Sinter         : k = 2.3e+03 1/s
```

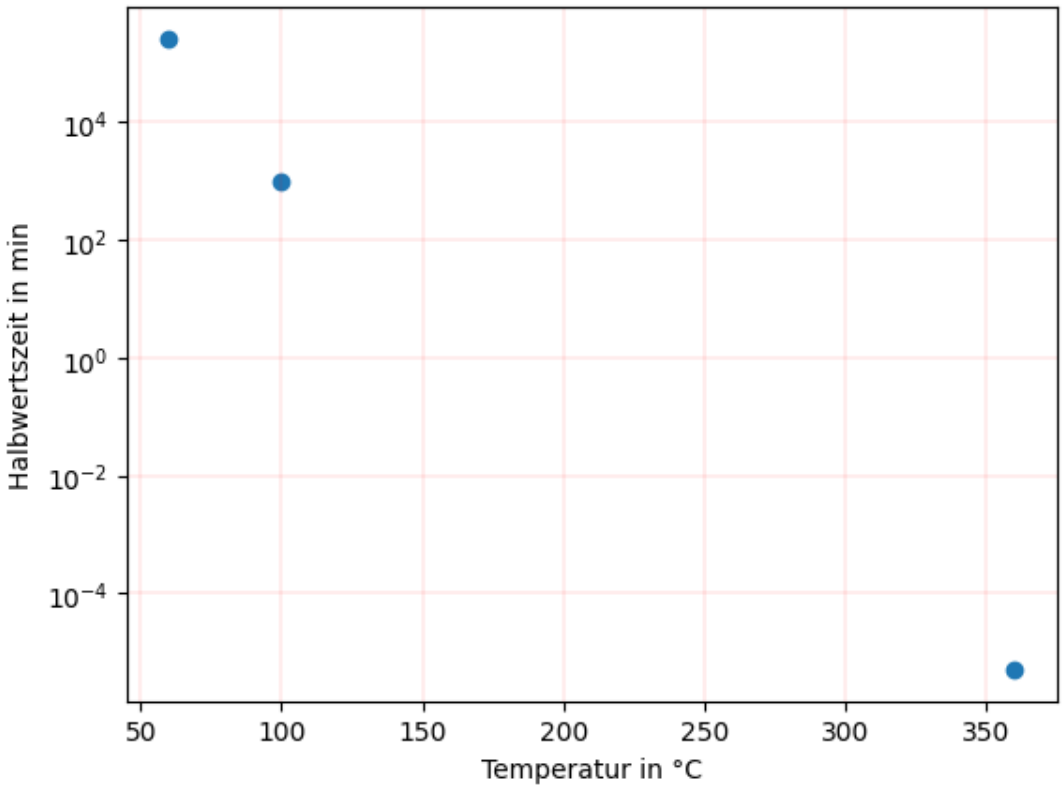
Half-life for T

[13]:
```
th = {process: log(2)/rate for process, rate in k.items()}
for process, time in th.items():
    print(f"{process:15}: ",
          f"th({T[process]-273.15:3.0f} °C) = {time:.2g} s  = {time/60:10.2f} min = ␣
 ↪{time/(3600*24):5.1f} days")
```

```
Polymerisation :  th(100 °C) = 5.9e+04 s  =     981.88 min =   0.7 days
Drying         :  th( 60 °C) = 1.6e+07 s  =  259205.01 min = 180.0 days
Sinter         :  th(360 °C) = 0.00031 s  =       0.00 min =   0.0 days
```

[24]:
```
x = [v-273.15 for v in T.values()]
y = [v/60 for v in th.values()]
plt.grid(axis='both', linewidth=0.1, color='red')
plt.scatter(x,y)
plt.yscale('log')
plt.xlabel('Temperatur in °C')
plt.ylabel('half-life time in min')
```

[24]:
```
Text(0, 0.5, 'half-life time in min')
```



Concentration progression over input concentration c_0 and process times t

[15]:
```
c_0 = 110 # mg/kg
t = {"Polymerisation": 3600, # s
     "Drying": 3600, # s
     "Sinter": 3600} # s
```

Concentration after polymerisation, drying and extrusion.

[23]:
```
c = {"Initial concentration": c_0}
c["Polymerisation"] = exp(-k["Polymerisation"] * t["Polymerisation"]) * c["Initial␣
 ↪concentration"]
c["Drying"]      = exp(-k["Drying"]      * t["Drying"])      * c["Polymerisation"]
c["Sinter"]         = exp(-k["Sinter"]         * t["Sinter"])         * c["Drying"]

process = "Initial concentration"
print(f"{process:21}:                                c = {c[process]:8.1f} mg/kg   ␣
 ↪{c[process]/c_0*100:5.1f} %")
for process in ["Polymerisation", "Drying", "Sinter"]:
    print(f"{process:21}: ",
          f"t = {t[process]:4} s,  T = {T[process]-273.15:3.0f} °C,    c =␣
 ↪{c[process]:8.4g} mg/kg   {c[process]/c_0*100:6.1f} %")
```

```
Initial concentration:                          c =    110.0 mg/kg    100.0 %
Polymerisation       :  t = 3600 s,  T = 100 °C,    c =    105.4 mg/kg     95.9 %
Drying               :  t = 3600 s,  T =  60 °C,    c =    105.4 mg/kg     95.8 %
Sinter               :  t = 3600 s,  T = 360 °C,    c =        0 mg/kg      0.0 %
```

The perxoid decays completely according to the calculation. In order to safely achieve an overestimation a lower limit of 1 µg/kg is defined. In this case, the migration simulation would start with this lower limit.