Cure Kinetics (28 points)

As it was explained during the lectures, a significant part of research in composites manufacturing is involved in analyzing curing processes of thermoset polymers.Thermoset resins, such as epoxy, undergo a curing process driven by chemical reactions. These reactions result in the creation of covalent bonds among monomers, ultimately forming polymer chains. This curing process is quantified by the parameter known as the 'degree of cure,' denoted as α. Initially, in the absence of any covalent bonds, the material consists solely of individual monomers. In this scenario, it can be asserted that no bonding has occurred, resulting in $\alpha = 0$. However, upon the addition of a hardener, the monomers initiate the formation of polymer chains through the establishment of covalent bonds. When all the monomers have become part of polymer chains through these covalent bonds, it is deemed that the degree of cure has reached its maximum, $\alpha = 1$

Chemical reactions occur through contact of two active groups belonging to different molecules (for example of epoxy and of a hardener). Elevated cure temperatures are essential to trigger and maintain the chemical reactions responsible for converting the thermoset into a fully cured state. Whenever resin and hardener molecules (after mixing) are provided with more kinetic energy, they are more likely to shift and collide with neighbouring molecules, increasing the chances of curing reactions. Additionally, when more time is given to two reactants, the probability of a necessary collision for bonding increases. As a result, these phenomena are highly influenced by resin chemistry, catalyst reactivity, cure temperature, and the presence of inhibitors or accelerators.

Therefore, we can say that the degree of cure α is a function of both temperature and time.

$$\alpha = f(T, t)$$

Cure kinetics is concerned with the rates of the chemical reactions in a curing process. This is relevant, since the cure kinetics can help predict the thermoset cure and therefore also determines the manufacturing process and (partially) the final material properties. When cure kinetics are understood, they can be used to predict the degree of cure α of a certain process. To make this type of prediction, DSC measurement data can be used.

Before starting to work on the questions and to achieve a better understanding of the topic, it is highly recommended for you to read the

paper provided with this notebook (Kailong Jin, William H. Heath, John M. Torkelson, Kinetics of multifunctional thiol-epoxy click reactions studied by differential scanning calorimetry: Effects of catalysis and functionality. Polymer. 2015; 81: 70-78).

Question 1 (1 point)

Differential Scanning Calorimetry (DSC) experiments are frequently used to evaluate the cure kinetics parameters involved in the curing of thermosets. To show this correlation, three tests were performed in isothermal conditions for the polymerization of a thermoset at three different temperatures. The .txt files which contain these raw isothermal DSC measurements were attached to this notebook.

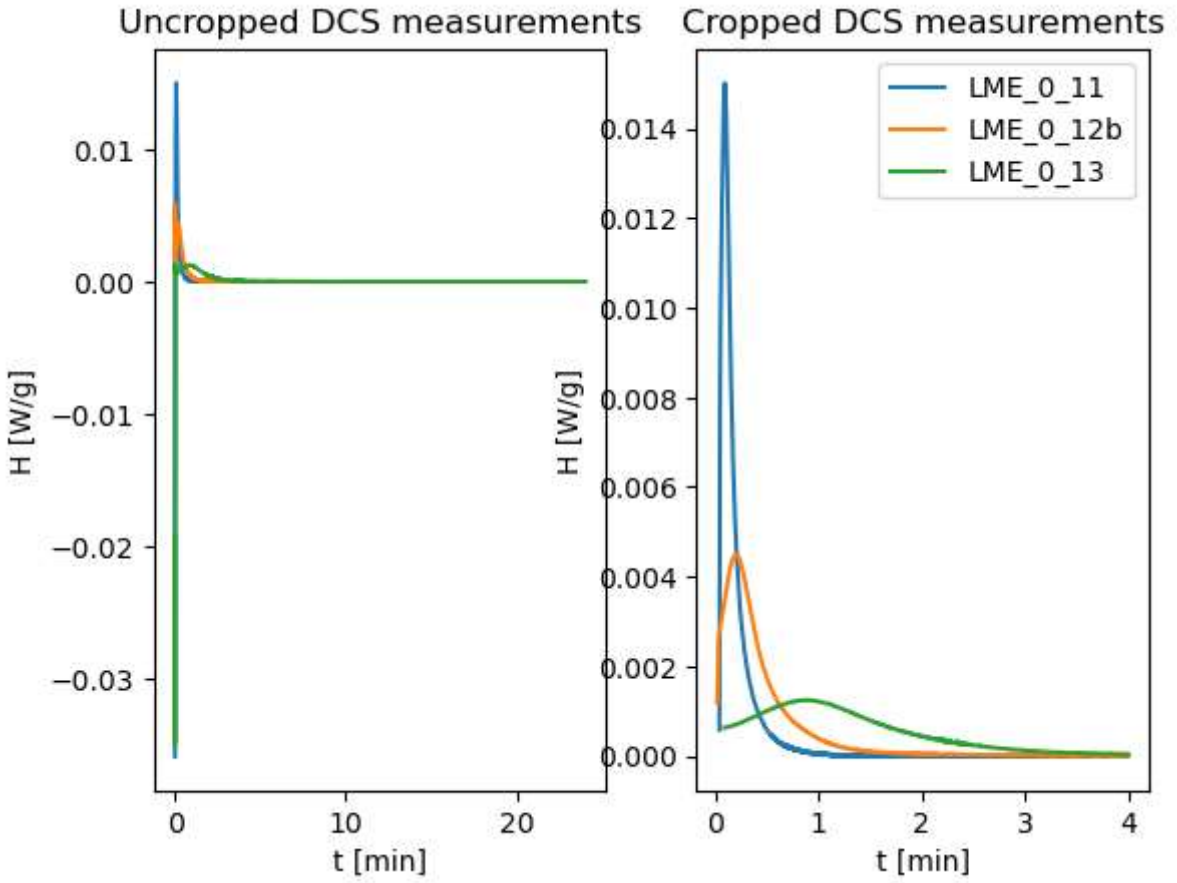
In the cell below, plot the relevant DSC measurement data for all of the three cases in a single figure. Don't forget to label the axis (ylabel = H[W/g], xlabel=t[min]).

NOTE: The unit used for heat flow data in the .txt files is [mW/s]

In [1]:
```
#import packages here
import numpy as np
import matplotlib.pyplot as plt
from math import e
from scipy.optimize import leastsq
import pandas as pd
from scipy.optimize import curve_fit
```

In [2]:
```
mass11 = 0.0144
mass12 = 0.0104
mass13 = 0.0137


File_data_1 = np.loadtxt("new_lme_0_11.txt", dtype=np.float64)
offset_1 = np.mean(File_data_1[14300:14399,1])
x_axis_1 = File_data_1[:,0] / 60
y_axis_1 = (File_data_1[:,1]-offset_1) * 10**(-3) / mass11 * 0.0166667


File_data_2 = np.loadtxt("LME_0_12b.txt", dtype=np.float64, skiprows=66, max_rows =
offset_2 = np.mean(File_data_2[28000:28800,1])
x_axis_2 = File_data_2[:,0] / 60
y_axis_2 = (File_data_2[:,1]-offset_2) * 10**(-3) / mass12 * 0.0166667


File_data_3 = np.loadtxt("LME_0_13.txt", dtype=np.float64, skiprows=90, max_rows =
offset_3 = np.mean(File_data_3[-70000:,1])
x_axis_3 = File_data_3[:,0] / 60
y_axis_3 = (File_data_3[:,1]-offset_3) * 10**(-3) / mass13 * 0.0166667

#Plotting
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.plot(x_axis_1, y_axis_1)
ax1.plot(x_axis_2, y_axis_2)
ax1.plot(x_axis_3, y_axis_3)
ax1.set_xlabel('t [min]')
ax1.set_ylabel('H [W/g]')
ax1.set_title('Uncropped DCS measurements')

ax2.plot(x_axis_1[59:14310], y_axis_1[59:14310], label='LME_0_11')
ax2.plot(x_axis_2[43:14371], y_axis_2[43:14371], label='LME_0_12b')
```

```
ax2.plot(x_axis_3[271:14353], y_axis_3[271:14353], label='LME_0_13')
ax2.set_xlabel('t [min]')
ax2.set_ylabel('H [W/g]')
ax2.set_title('Cropped DCS measurements')
ax2.legend()
plt.show()
```



Question 2 (2 points)

After visualizing the heat generation curves, explain why and how isothermal DSC measurement data can be related to the curing rate of a thermoset resin. Why is there a peak and what can it be related to?

Answer: The heat generation curves relate to the curing rate of the thermoset resin as the heat generated is directly proportional to the amount of resin reacting. As the reaction is typically exothermic, it generates heat as a by-product. The amount of heat generated gives an indication on the amount of reaction occuring. The peaks in the heat flow plot correspond to the time period where the reaction is occuring at its' fastest. When the heat flow returns to the 'zero' value, the reaction has been comleted, and the thermoset resin has reached a degree of cure of 1.

As mentioned, the just plotted isothermal DSC data can be used to obtain an estimation of degree of cure α vs. time t. To do so, it first has to be assumed that the heat flow is proportional to the degree of cure.

$$\Delta H_{max} \equiv \alpha = 1$$

Where ΔH_{max} identifies the maximum total heat of reaction found for the three DSC measurements.

Overall, the formula above expresses how the total generated heat flow while curing corresponds to a complete degree of cure for the reaction (100%).

Then, this assumption is used to normalize the heat flow measurement. This normalized heat flow can now be related to the curing rate.

$$\frac{d\alpha}{dt} = \frac{1}{\Delta Hmax}\frac{dH(t)}{dt}$$

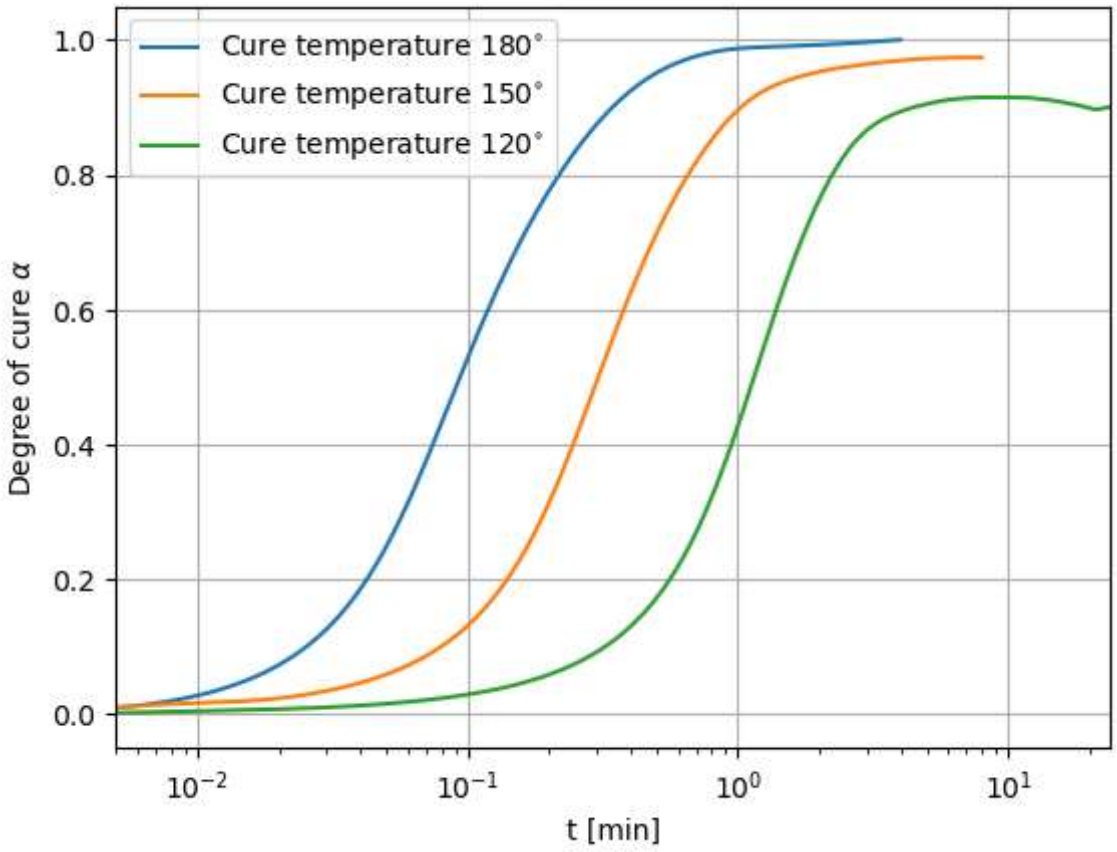
Finally, integrating this result leads to the relationship of the degree of cure α vs. time t.

Question 3 (4 points)

Use the assumptions and information provided above to plot the degree of cure α vs. time t, using the provided data. Plot all cases in a single figure. Make sure to plot over a time interval which suits the results, include a legend and axes labels with units.

In [3]:
```
H11 = y_axis_1[58:]
H12 = y_axis_2[0:]
H13 = y_axis_3[80:]

dhdt11 = (File_data_1[58:,1]-offset_1)/mass11/1000
dhdt12 = (File_data_2[0:,1]-offset_2)/mass12/1000
dhdt13 = (File_data_3[80:,1]-offset_3)/mass13/1000

Hmax11 = np.sum(H11)
Hmax12 = np.sum(H11)
Hmax13 = np.sum(H11)

dt = File_data_1[4,0] - File_data_1[3,0]

alpha11 = np.zeros(len(H11))
alpha12 = np.zeros(len(H12))
alpha13 = np.zeros(len(H13))

timeaxis11 = np.zeros(len(alpha11))
timeaxis12 = np.zeros(len(alpha12))
timeaxis13 = np.zeros(len(alpha13))

dadt11 = np.zeros(len(H11))
dadt12 = np.zeros(len(H12))
dadt13 = np.zeros(len(H13))

for i in range(len(H11)-1):
    dadt11[i+1] = dhdt11[i+1]/Hmax11
    alpha11[i+1] = alpha11[i] + dadt11[i+1]*dt
    timeaxis11[i+1] = timeaxis11[i] + dt


for i in range(len(H12)-1):
    dadt12[i+1] = dhdt12[i+1]/Hmax12
    alpha12[i+1] = alpha12[i] + dadt12[i+1]*dt
    timeaxis12[i+1] = timeaxis12[i] + dt


for i in range(len(H13)-1):
    dadt13[i+1] = dhdt13[i+1]/Hmax13
    alpha13[i+1] = alpha13[i] + dadt13[i+1]*dt
```

```
    timeaxis13[i+1] = timeaxis13[i] + dt



f1 = plt.figure()
plt.plot(timeaxis11/60,alpha11,label=r"Cure temperature $180^{\circ}$")
plt.plot(timeaxis12/60,alpha12,label=r"Cure temperature $150^{\circ}$")
plt.plot(timeaxis13/60,alpha13,label=r"Cure temperature $120^{\circ}$")
plt.legend()
plt.grid()
plt.xlabel('t [min]')
plt.ylabel(r"Degree of cure $\alpha$")
plt.xscale('log')
plt.xlim([0.005,np.max((timeaxis13/60))])
plt.show()
```



Question 4 (4 points)

Question 4.1: In the previous question you have generated graphs illustrating the relationship between the degree of cure α and time t for three isothermal DSC measurements carried out at different temperatures. Provide your analysis regarding the maximum degree of cure visible in the generated graphs. Can you explain the correlation between the final degree of cure and cure temperature?

Answer: From the graphs generated it can be seen that the degree of cure, α, slowly increases in the beginning and also near the completion of the reaction. It is also visible that it depends on time as well as on cure temperature. As the cure temperature increases there is more energy involved in the reaction which makes it faster for the cross-links to be formed. Therefore, higher cure temperatures increases the rate of cure and produce the maximum degree of cure in shorter periods of time. In some circumstances, if the cure

temperature is too low, α may not reach a 100% level in the time interval given to the reaction to occur. That's what happens to the reactions at 150 degrees and 120 degrees.

Question 4.2 : A graph correlating degree of cure α, sometimes also called conversion, and cure time for DSC isothermal scans of an epoxy-amine system is reported below. Conversions of the system cured at six different temperatures for 166 hours are shown.

Use this data to compare your graph from question 3 and the given one. You may highlight similarities and provide your reasons for any possible relevant difference.

Conversion vs. ln(time) curves for an epoxy-amine system. From Wisanrakkit and Gillham, J. Appl. Poly. Sci. 42, 2453 (1991)

Answer By comparing the graph obtained in question 3 with the given one, we can conclude that for both materials, a raise in the cure temperature results in a higher curing rate. A relevant difference between them is that it takes longer for the epoxy-amine system to cure. This might suggest different situations such as: that it requires a higher degree of cross-linking, that it includes inhibitors that slow down the curing process, that it has a higher volume which requires more time for the curing agent to penetrate and react throughout the entire material, or simply that its chemical reactions proceed more slowly due to their intrinsic kinectics.

Several cure kinetics models were formulated to predict and simulate experimental cure profiles of thermosetting resins in terms of curing rate. You will be putting to test one of the most widely used ones: the Kamal-Sourour model (shown below)

$$\frac{d\alpha}{dt} = (k_1 + k_2 \cdot \alpha^m)(1-\alpha)^n$$

Where k_1 and k_2 are rate constants, and m and n are reaction orders.

The reaction rate constants k_1 and k_2 strongly depend on cure temperature and follow an Arrhenius type relation, as shown by the

equation:

$$k_i = A_i \cdot exp(-\frac{E_i}{RT})$$

$$i = 1, 2$$

Where the pre-exponential factor A_i represents a constant, E_i is the activation energy (mol/J), R is the molar gas constant and T is the cure temperature.

Question 5 (7 points)

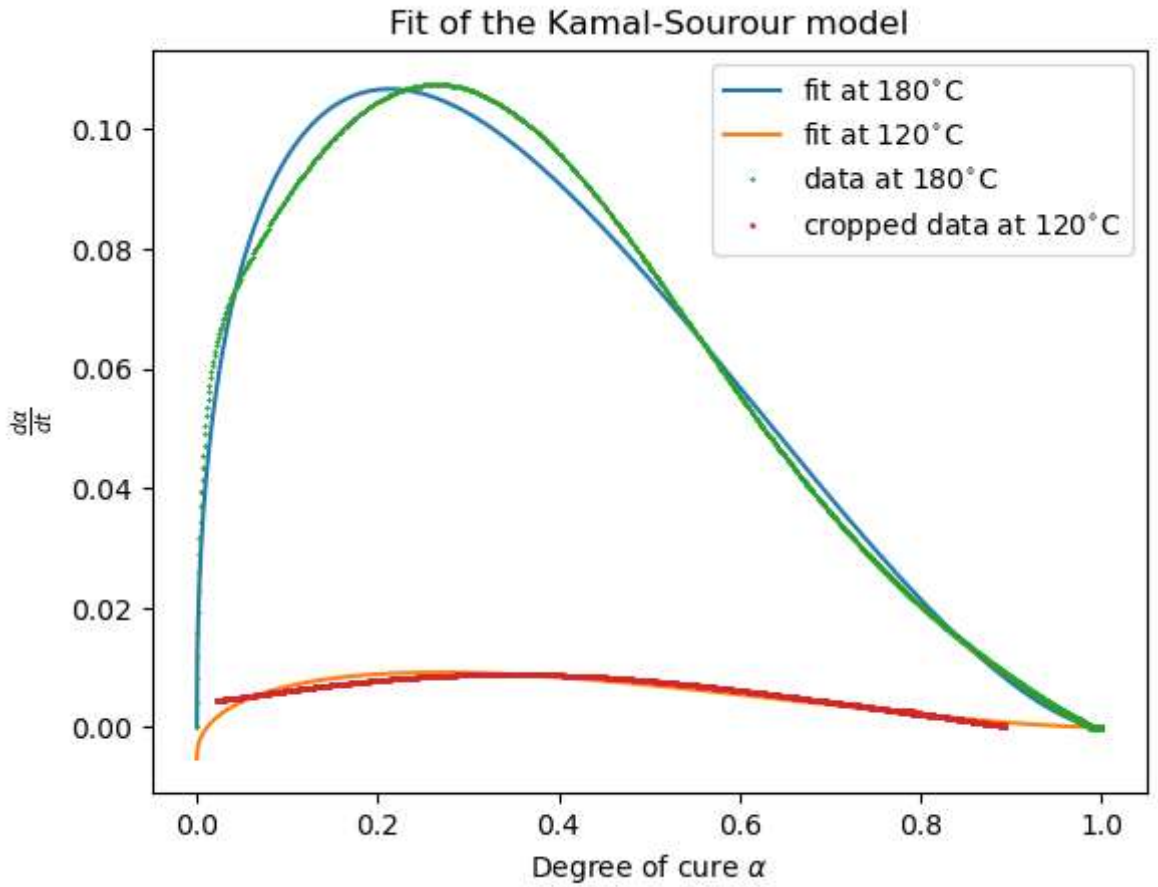
Question 5.1: Using the experimental isothermal DSC data (already used in question 1) for curing at 180°C and 120°C, estimate the parameters of the Kamal model (k_1, k_2, m, n, E_i, A_i) and report them below. After obtaining these parameters, use them to create plots of conversion rate $d\alpha/dt$ vs. degree of conversion α for these two cure temperatures. Compare these curves with the experimental results of $d\alpha/dt$ vs. α.

Tip: Parameters can be estimated by: 1. making an initial guess of the parameters. 2. Minimizing the sum of squared errors between guess and data (fit to data), where m, n, k_1, k_2, A_1, A_2 should be the same for the two datasets.

In [4]:
```
R  = 8.314
# model works better when using degrees celcuis than with Kelvin
T1 = 180
T3 = 120

def model(alpha, A1, A2, E1, E2, m, n,T):
    k1 = A1*np.exp(-E1/(R*T))
    k2 = A2*np.exp(-E2/(R*T))
    dadt = (k1 + k2*alpha**m)*(1-alpha)**n
    return dadt

def test_function(alpha11, alpha13, A1, A2, E1, E2, m, n):
    dadt1 = (A1*e**(-E1/(R*(T1))) + A2*e**(-E2/(R*(T1)))*alpha11**m)*(1-alpha11)**r
    dadt3 = (A1*e**(-E1/(R*(T3))) + A2*e**(-E2/(R*(T3)))*alpha13**m)*(1-alpha13)**r
    return dadt1,dadt3

def reduce_resolution(original_dataset, target_length=3500):
    original_length = len(original_dataset)
    interval = max(original_length // target_length, 1)  # Calculate the interval

    # Sample the dataset at regular intervals
    reduced_data = original_dataset[::interval]

    # Trim to achieve the exact target length if needed
    reduced_data = reduced_data[:target_length]

    return reduced_data

# Crop to reduce some of the long tail at the end, and remove the small peak at the
alpha13_sampeled = reduce_resolution(alpha13[271:15000])
dadt13_sampeled = reduce_resolution(dadt13[271:15000])

indata = list()
for i in range(3500):
    a = alpha11[i]
```

```
    b = alpha13_sampeled[i]
    indata.append( [a,b] )


outdata = list()
for i in range(3500):
    s = dadt11[i]
    t = dadt13_sampeled[i]
    outdata.append( [s, t ] )

indata = np.array( indata)
outdata = np.array( outdata)

###############################################################################
### define the residuals function for fitting This is the important part!
###############################################################################

def residuals( params, xdata, ydata, weightA=1, weightB=1 ):
    x0, x1, x2, x3, x4, x5 = params
    diff = list()
    for ab, st in zip( indata, outdata ):
        a, b = ab
        s, t = st
        sf, tf = test_function( a, b, x0, x1, x2, x3, x4, x5)
        diff.append( weightA * ( s - sf ) )
        diff.append( weightB * ( t - tf ) )
    return diff

### Fit
solx, cov, info, msg, ier = leastsq(
    residuals, [1,1,1,1,1,1],
    args=( indata, outdata ), full_output=True
)
#print (solx)

A1, A2, E1, E2, m, n = solx

alpha = np.linspace(0,1,3500)
dadt11_fit = model(alpha, A1, A2, E1, E2, m, n, T1)
dadt13_fit = model(alpha, A1, A2, E1, E2, m, n, T3)

plt.plot(alpha, dadt11_fit, label='fit at 180$^{\circ}$C')
plt.plot(alpha, dadt13_fit,  label='fit at 120$^{\circ}$C')
plt.plot(alpha11, dadt11, 'x', markersize=1, label='data at 180$^{\circ}$C')
# plt.plot(alpha13, dadt13, 'x', markersize=1, label='data at 120$^{\circ}$C')
plt.plot(alpha13_sampeled, dadt13_sampeled, 'o', markersize=1, label='cropped data
plt.title('Fit of the Kamal-Sourour model')
plt.legend()
plt.xlabel(r"Degree of cure $\alpha$")
plt.ylabel(r"$\frac{d\alpha}{dt}$")
plt.show()

print('A1 =',solx[0])
print('A2 =',solx[1])
print('E1 =',solx[2])
print('E2 =',solx[3])
print('m =',solx[4])
print('n =',solx[5])
```

```
C:\Users\coenh\AppData\Local\Temp\ipykernel_8632\1334968296.py:13: RuntimeWarning:
divide by zero encountered in scalar power
  dadt1 = (A1*e**(-E1/(R*(T1))) + A2*e**(-E2/(R*(T1)))*alpha11**m)*(1-alpha11)**n
```




```
A1 = -4.194653926187208e-11
A2 = 21.934545040124082
E1 = -18591.074809935155
E2 = 6408.1677835087185
m = 0.429345879756128
n = 1.5851814931939467
```

The model with your estimated parameters (m, n, k_1, k_2, A_1, A_2) that you have created in Question 5.1 should be a model that describes the cure behaviour of the resin system at different temperatures. To check whether that is the case, you will be using your model for a 150°C cure temperature.

Question 5.2: The goal now is to validate the parameters you found in question 5.1 (m, n, k_1, k_2, A_1, A_2) by applying the Kamal model to a third cure temperature (150°C).

Therefore, plot in a single graph the conversion rate $d\alpha/dt$ vs. degree of conversion α:

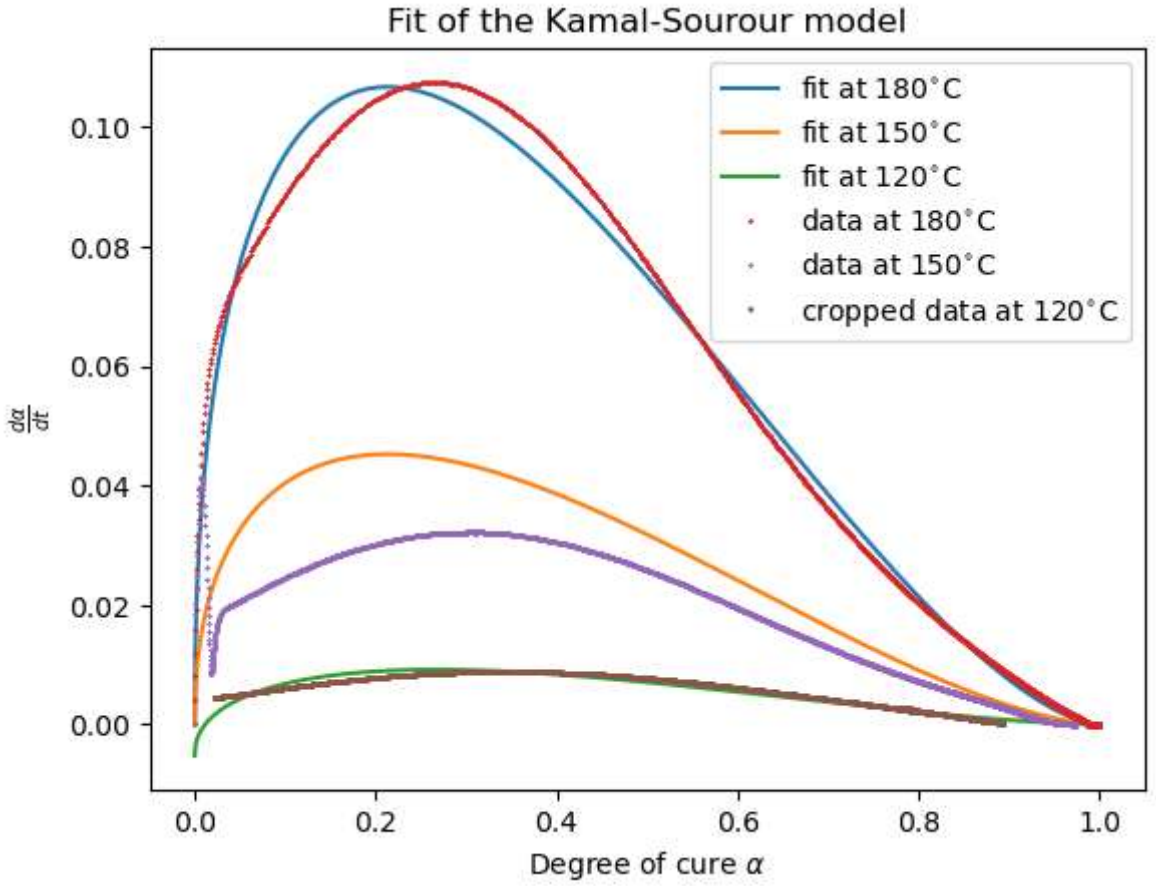
- from the DSC experimental data for the three cure temperatures provided in Question 1 provided in the spreadsheet

- from the Kamal model, using parameters estimated in 5.1 (for all three cure temperatures)

Finally, comment on the plot you obtained for 150°C based on your model compared to the experimental data. Also comment on the validity and accuracy of your model.

In [5]:
```
T2 = 150
dadt12_fit = model(alpha, A1, A2, E1, E2, m, n, T2)
```

```
plt.plot(alpha, dadt11_fit, label='fit at 180$^{\circ}$C')
plt.plot(alpha, dadt12_fit, label='fit at 150$^{\circ}$C')
plt.plot(alpha, dadt13_fit,  label='fit at 120$^{\circ}$C')
plt.plot(alpha11, dadt11, 'x', markersize=1, label='data at 180$^{\circ}$C')
plt.plot(alpha12, dadt12, 'x', markersize=1, label='data at 150$^{\circ}$C')
# plt.plot(alpha13, dadt13, 'x', markersize=1, label='data at 120$^{\circ}$C')
plt.plot(alpha13_sampeled, dadt13_sampeled, 'o', markersize=1, label='cropped data
plt.title('Fit of the Kamal-Sourour model')
plt.legend()
plt.xlabel(r"Degree of cure $\alpha$")
plt.ylabel(r"$\frac{d\alpha}{dt}$")
plt.show()
```



Comment: As expected, the fit of the model is much better for the 120 and 180 degree curves, than the 150 degree. This is because although the 150 degrees lies between 120 and 180, the model wasn't fitted for this data set. We can see however, that the fit gets progressively better towards the end of the curve. This may be because the data used to create the model for the 120 degree curve was cropped to not include the initial peak. This decision was made as the initial fittingwas not very neat for the remainder of the data with this peak remaining. Havng not performed the measurementsthemselves, it is hard to tell whether this small peak, and the one for 150 degrees, is due to measurement errors, or due to a physical effect. However, we believe that the better fit for the remainder for ther curves is more important than modelling this small peak. However, this means the model is not valid for approximately the first 0-0.05 degree of cure.

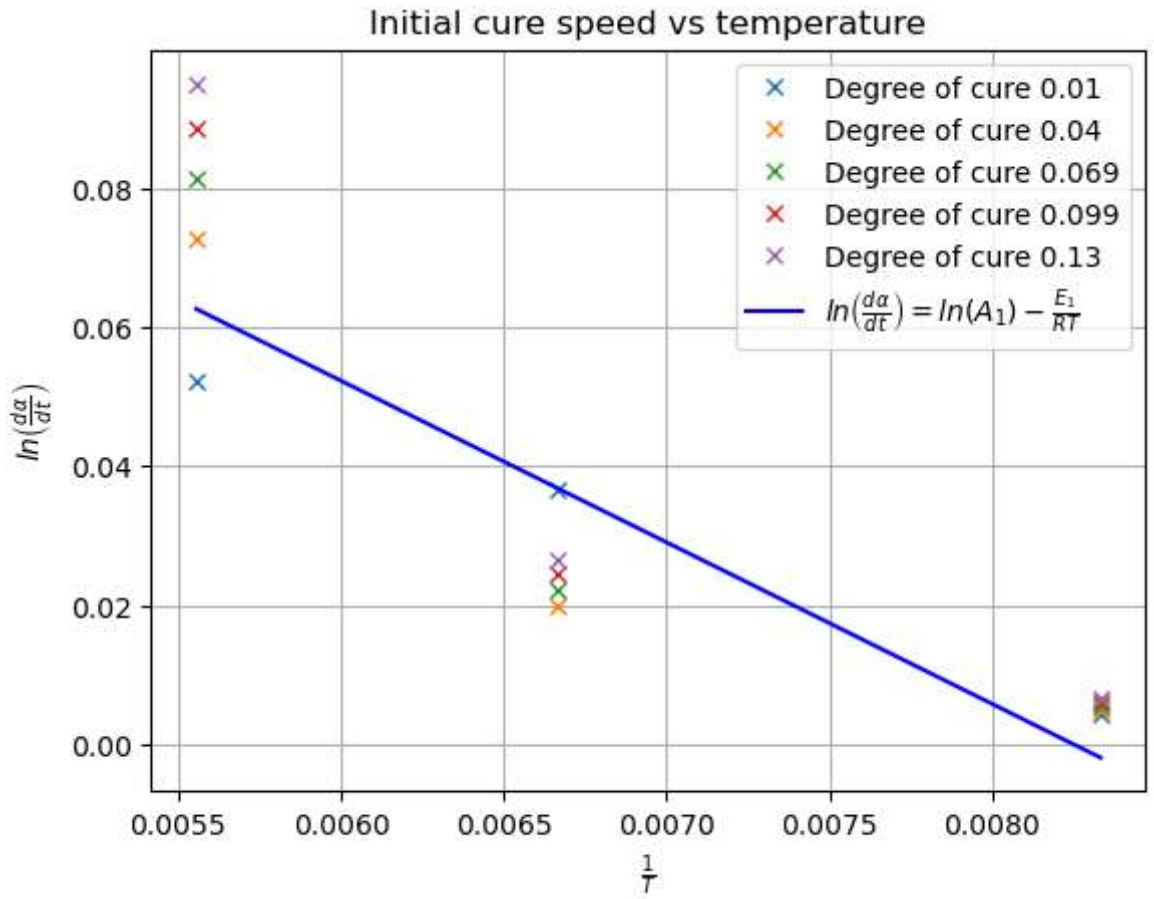
What is also interesting to note is that from alpha is 0.05 to approx. alpha is 6.0, the fit for the 180 degree data is shifted forward by approx. 0.03 alpha. A smimlar thing can also be seen for the 150 and 120, although less pronounced. As it occurs for all three curves, it is not

likely that this is caused by the fitting parameters. It may also be that the actual shape of the data curve is too complex for the model to recreate.

Question 5.3: In order to verify one of the parameters you found, plot in a single graph $ln(d\alpha/dt)$ vs. 1/T for different degrees of cure α of the three DSC measurements. Comment on the obtained graph and on the activation energy of the epoxy system.

Hint: The plot should confirm the Arrhenius type relation existing between $\frac{E}{RT}$ and $ln(d\alpha/dt)$

In [6]:
```
#degcure = np.arange(0.0005,0.0035,0.0005)
degcure = np.arange(0.01,0.16,0.03)

def find_nearest(array, value):
    array = np.asarray(array)
    idx = (np.abs(array - value)).argmin()
    return idx

lndadt = np.zeros([len(degcure),3])

overT = np.array([1/(180),1/(150),1/(120)])

for i in range(len(degcure)):
    lndadt[i,0] = dadt11[find_nearest(alpha11,degcure[i])]
    lndadt[i,1] = dadt12[find_nearest(alpha12,degcure[i])]
    lndadt[i,2] = dadt13[find_nearest(alpha13,degcure[i])]


def Logdadt(overT,a,b):
    R = 8.314
    out = np.log(a) - b/R*overT
    return out

cfit, ccov = curve_fit(Logdadt,overT,lndadt[len(overT)//2,:])
a = cfit[0]
b = cfit[1]

linefit = Logdadt(overT,a,b)

f3 = plt.figure()
for i in range(len(degcure)):
    plt.plot(overT,lndadt[i,:],label="Degree of cure "+str(degcure[i])[:5],marker='
plt.plot(overT,linefit,'b',label=r"$ln \left( \frac{d\alpha}{dt} \right) = ln(A_1)
plt.grid()
plt.xlabel(r"$\frac{1}{T}$")
plt.ylabel(r"$ln \left( \frac{d\alpha}{dt} \right)$")
plt.legend()
plt.title('Initial cure speed vs temperature')
plt.show()

print("A1 =",a)
print("E1 =",b)
```




```
A1 = 1.2110505441247694
E1 = 192.91493341552874
```

Comment: As can be noted, the parameters found in this regression are not the same as those found in the model itself. They vary quite widely. This can be for two reasons; firstly it is noted that the linear fit only has three x-points to pass through, and each of these have a large y-range. This means the choice of line is ambigous, especially with regards to the ast x-point which the line doesn't go through. Changing this line can cause the parameters to vary. Secondly, the parametrs found in the model are not very physically accurate. The size of A1 implied that the first term is neglected. Anmd the activation energy should not be negative. In order to fix this, and potentially obtain a better fit, limiuts can be set on the parameters within the model. This may give model parameters more similar to what is found here. 4

As mentioned at the beginning of this Notebook, thermoset polymers undergo a curing cycle to achieve hardening. This process involves the formation of covalent bonds between monomers, ultimately resulting in the creation of the polymer network. The amount of formed bonds, therefore the degree of cross-linking, directly impacts the glass transition temperature (Tg) of the material. In the initial stages of the curing cycle, when relatively few bonds or crosslinks have formed, the material exhibits a relatively low Tg. However, as the degree of cure increases, the Tg of the material increases.

The correlation between Tg and the degree of cure is particularly crucial in the design and engineering of polymer-based products, as it allows for the precise tuning of material properties by controlling the curing parameters. By adjusting the degree of cross-linking through cure temperature and time, engineers can achieve specific material characteristics, such as

stiffness, strength, resistance to heat, and dimensional stability, tailored to the requirements of a particular application.

Question 6 (2 points)

Describe the glass transition temperature and explain what happens (on a microscale and chemical level) when a material transitions through this value. Please include an explanation on how this value depends on the degree of cure.

Answer: The glass transition temperature is the temperature (range) at which the amorphus parts of the polymer start to melt. The weaker intramolecular bonds (Hydrogen and van de Waals) start to break down, while the stronger covalent bonds forming the cross-links stay. At the glass transition temperature, the elastic modulus of the material starts to decrease, while the loss modulus starts to increase. This signifies the transition the material is undergoing, where additional energy applied will go into the deformation of the material, instead of energy storage. This glass transition depends on the cross-link density, the crystallinity, and the molar mass (the length of the polymer chains). Due to the nature of the curing reaction, when the degree of cure increases, the cross-link density and the length of the polymer chains increase. Both these factors mean that more energy is required to loosen the bonds between the polymer chains, thereby increasing the glass transition temperature.

For a thermosetting polymer, it is particularly relevant to have a model able to describe the physical changes encountered during the cure cycle.

The following empirical relation (Di Benedetto equation) between glass transition temperature T_g and conversion α has this goal and identifies the vitrification limit of a polymer:

$$\frac{T_g - T_{g0}}{T_{g\infty} - T_{g0}} = \frac{\lambda\alpha}{1-(1-\lambda)\alpha}$$

In which $\lambda = \frac{\Delta C_p}{\Delta C_{p0}}$ is the ratio of the heat capacities of the fully reacted system and the initial system (therefore $\lambda < 1$), $T_{g\infty}$ is the Tg of the fully reacted system, and T_{g0} is the Tg of the initial system.

Question 7 (2 points)

A dataset containing glass transition temperature T_g and degree of cure α values for Airstone 780E was provided with this notebook.

Use the T_g values to fit the model to the experimental data, find λ and report its value below.

Generate a graph for the glass transition temperature T_g vs. degree of cure α to compare experimental values with the analytical model.

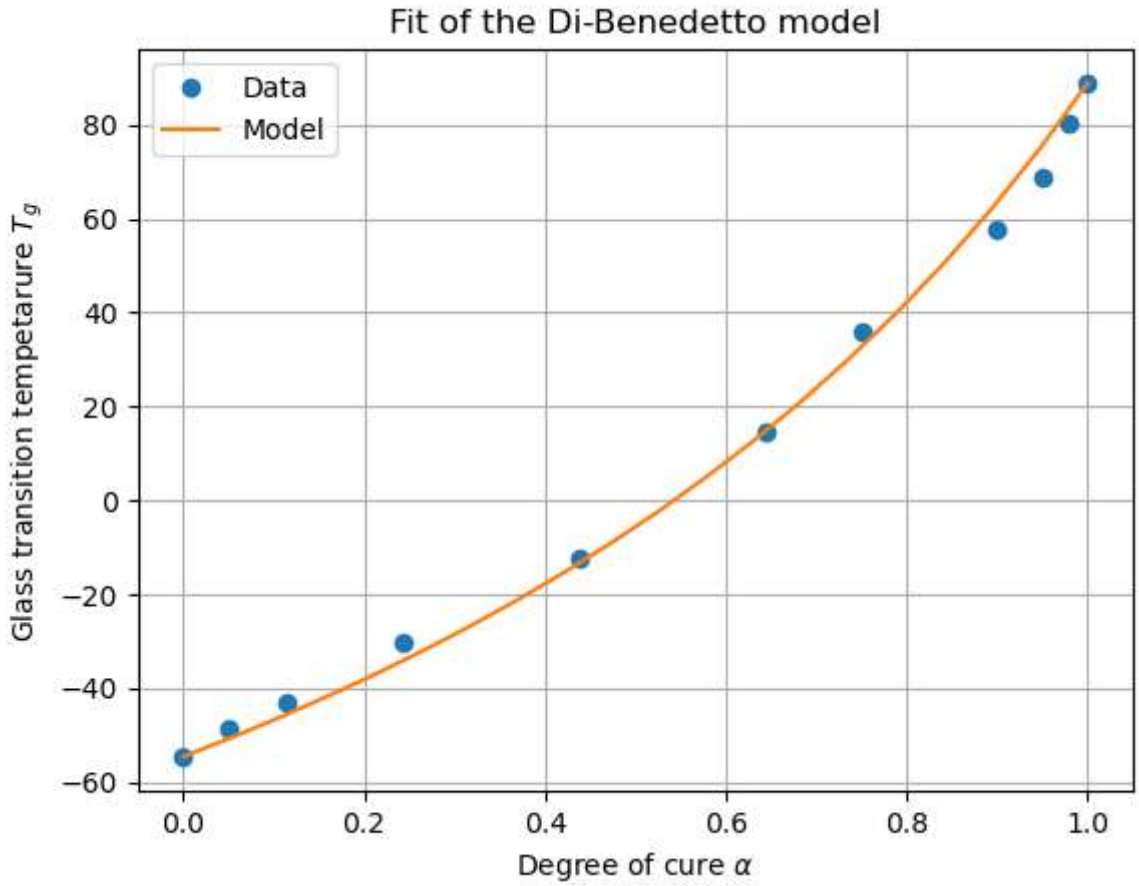
NOTE: T_{g0} = -54.577, $T_{g\infty}$ = 88.853,

In [7]:
```
tg0 = -54.577
tginf = 88.853

df = pd.read_excel('Tg data.xlsx')
Tg_data = np.array(df.loc[:,'Unnamed: 8'])[1:-3]
alpha_data = np.array(df.loc[:,'Unnamed: 7'])[1:-3]

def di_benedetto(alpha, lamda):
    Tg = lamda*alpha*(tginf-tg0)/(1-(1-lamda)*alpha) + tg0
    return Tg
```

In [8]:
```
[lamda, i] = curve_fit(di_benedetto, alpha_data, Tg_data)
print('Lambda is ', lamda)
```

```
Lambda is  [0.51726821]
```

In [9]:
```
alpha = np.linspace(0,1,20)
Tg = di_benedetto(alpha, lamda)

plt.plot(alpha_data, Tg_data, label='Data',marker='o',linewidth=0)
plt.plot(alpha, Tg, label='Model')
plt.title('Fit of the Di-Benedetto model')
plt.legend()
plt.grid()
plt.xlabel(r"Degree of cure $\alpha$")
plt.ylabel(r"Glass transition tempetarure $T_g$")
plt.show()
```



The glass transition temperature is specifically relevant when trying to establish a range of temperatures over which a polymer is able to retain its mechanical properties. This range is commonly identified as "Service temperature" and in these conditions the polymer can perform its intended function without experiencing significant degradation or detrimental changes in its properties.

Question 8 (2 points)

Two composite parts (A and B) have been previously manufactured by infusion using Airstone 780E, cured isothermally at 80°C. It is known that the reached degree of cure at the end of the cycle for part A is α= 0.9, while part B was consolidated up to α= 1.

Use the DiBenedetto equation and the parameters given in question 7 to obtain the glass transition temperatures Tg for the two parts. Please, comment on the obtained values in relation to the service temperatures for these parts.

In [10]:
```
alpha_A = 0.9
alpha_B = 1.0

Tg_A = di_benedetto(alpha_A,lamda)
Tg_B = di_benedetto(alpha_B,lamda)

print('Tg for part A is', f'{Tg_A[0]:.2f}','deg C')
print('Tg for part B is', f'{Tg_B[0]:.2f}','deg C')
```

```
Tg for part A is 63.49 deg C
Tg for part B is 88.85 deg C
```

Comment: The glass transition temperature T_g of part A is more than 15° Celsius lower than the T_g of part B, even though there is 'only' a 10% difference in reached cure degree. This means that one can only reliable use part A in temperatures well below 63° Celsius in order to have reliable mechanical properties. For use of the part in the temperature range 63° and 88° one needs to reach higher degree of cure, close to that of part B.

Question 9 (4 points)

The cure kinetics model applicable to Airstone A780 is known and given below:

$$\frac{d\alpha}{dt} = \frac{A \cdot exp(-\frac{E}{RT})}{1 + exp(C(\alpha - \alpha_c - \alpha_T T))}(1-\alpha)^n \cdot \alpha^m$$

Where:

A = 681085 1/s

E = 59291 J/mol

n = 1.67

m = 0.12

C = 47.7

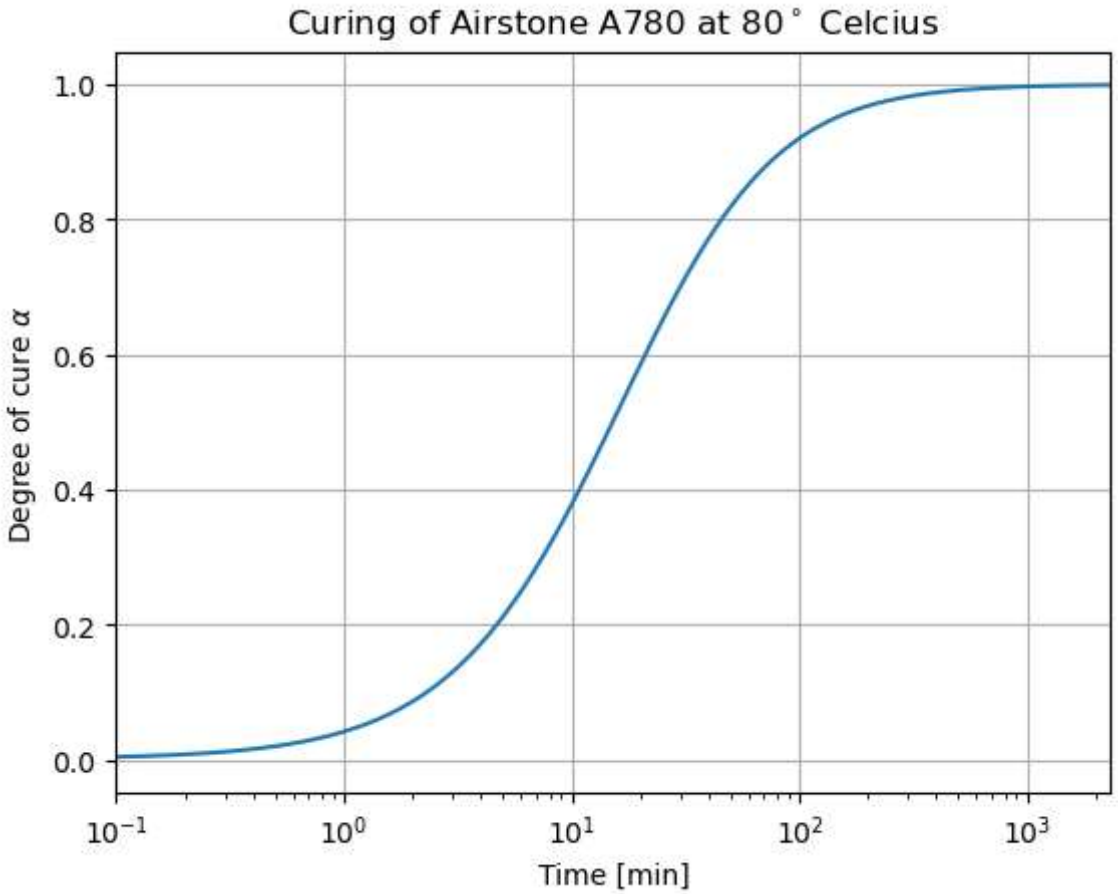
α_c= 0.77

α_T = 0.0016

You are now required to use the provided equation model to improve the cure cycle selected for these parts.

Question 9.1: Using the cure kinetics model and the curing rate, now please calculate what was the curing time for part A (up to α= 0.9) and what additional curing time would be necessary to obtain the highest possible glass transition temperature. Why is this consideration necessary in terms of both processing and service temperatures?

In [11]:
```
alpha_A = 0.9
alpha_B = 0.999


curetemp = 80+273.15

time = np.arange(0,140000,1)
dtAirstone = time[1] - time[0]

Airstonealpha = np.zeros(len(time))
Airstonealpha[0] = 0.001

def AirstoneDaDt(alpha,curetemp):
    A = 681085
    E = 59291
    n = 1.67
    m = 0.12
    C = 47.7
    alphac = 0.77
    alphaT = 0.0016
    R = 8.314

    dadt = ((A*np.exp(-E/R/curetemp))/(1+np.exp(C*(alpha-alphac-(alphaT*curetemp))))
    return dadt

for i in range(1,len(time)):
    Airstonealpha[i] = Airstonealpha[i-1] + AirstoneDaDt(Airstonealpha[i-1], curete

f9 = plt.figure()
plt.plot(time/60,Airstonealpha)
plt.xlabel('Time [min]')
plt.ylabel(r"Degree of cure $\alpha$")
plt.grid()
plt.title(r"Curing of Airstone A780 at $80^\circ$ Celcius")
plt.xscale('log')
plt.xlim([0.1,time[-1]/60])
plt.show()

timedoneA = time[np.where(Airstonealpha > alpha_A)]
timedoneB = time[np.where(Airstonealpha > alpha_B)]

Acuretime = timedoneA[0]/60
Bcuretime = timedoneB[0]/60

print('Time to reach degree of cure 0.900:',f'{Acuretime:.2f}','minutes')
print('Time to reach degree of cure 0.999:',f'{Bcuretime:.2f}','minutes')
```




```
Time to reach degree of cure 0.900: 83.58 minutes
Time to reach degree of cure 0.999: 2186.05 minutes
```

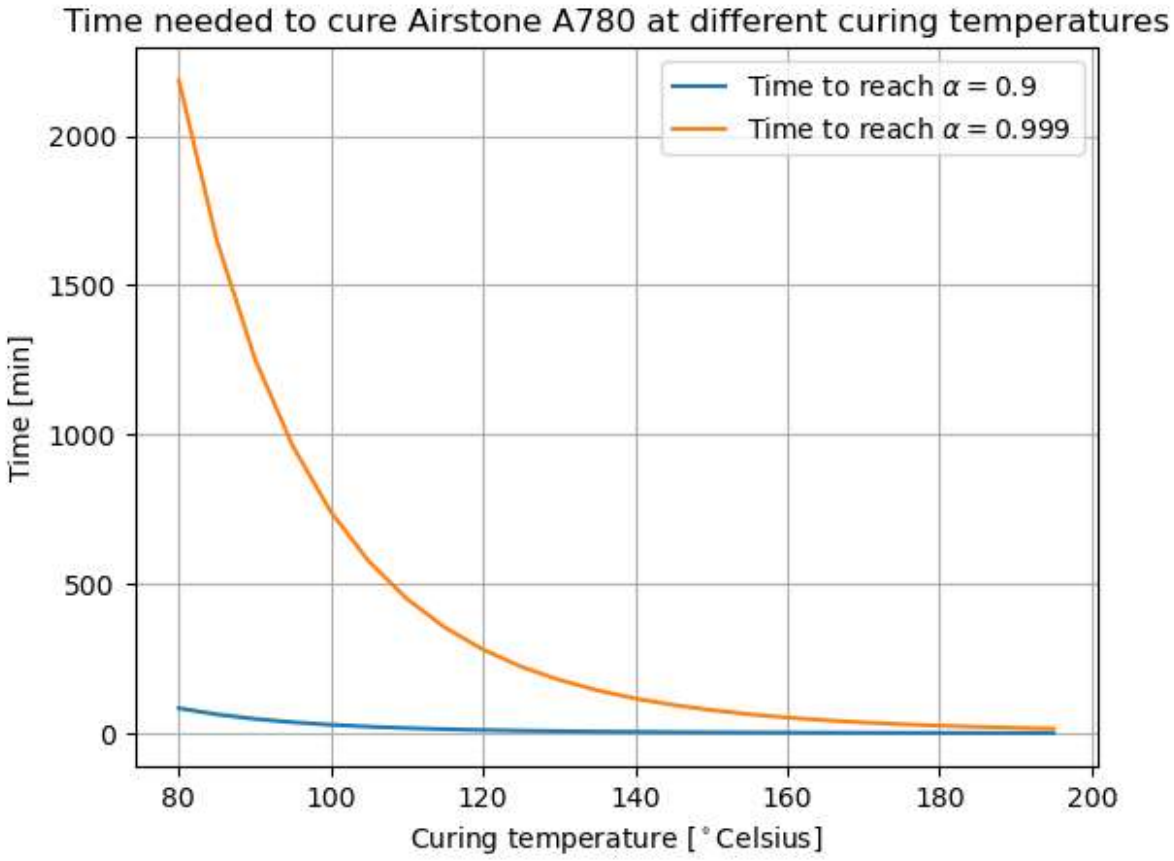
Reflection: As can be seen from the model above, the time needed to reach a full cure (or 0.999 as the model will never reach 1.0) is orders of magnitudes higher than the time needed to reach a degree of cure of 90%. 1 hour and 23 mins vs 36+ hours.

It is therefore highly impractical to cure a part to a full degree of cure and have the highest possible T_g at this curing temperature for anything other than bespoke 'one off' prototyping. If a T_g higher than that of part A is required from a service temperature standpoint, one should look at either using a different resin system which has an higher T_g at a lower degree of cure, or curing at higher temperatures, as will be discussed below in question 9.2

Question 9.2: The production team needs to speed up the production cycle of these parts. One way of achieving that would be to change the cure temperature. Plot a Figure to show the cure time needed for different cure temperatures to reach a degree of cure of 0.9 and 1. Comment on which cure temperature & cure time you would suggest to the production team and why (in your answer also think about the different implications a chosen cure temperature might have during the cure cycle and on the final product).

In [12]:
```
temprange = np.arange(80,200,5)
alpha09 = np.zeros(len(temprange))
alpha99 = np.zeros(len(temprange))

for j in range(len(temprange)):
    curetemp = temprange[j]+273.15

    time = np.arange(0,140000,1)
```

```
    dtAirstone = time[1] - time[0]

    Airstonealpha = np.zeros(len(time))
    Airstonealpha[0] = 0.001



    for i in range(1,len(time)):
        Airstonealpha[i] = Airstonealpha[i-1] + AirstoneDaDt(Airstonealpha[i-1], cu

    timedoneA = time[np.where(Airstonealpha > alpha_A)]
    timedoneB = time[np.where(Airstonealpha > alpha_B)]

    alpha09[j] = timedoneA[0]/60
    alpha99[j] = timedoneB[0]/60

f92 = plt.figure()
plt.plot(temprange,alpha09,label=r"Time to reach $\alpha = 0.9$")
plt.plot(temprange,alpha99,label=r"Time to reach $\alpha = 0.999$")
plt.grid()
plt.legend()
plt.title('Time needed to cure Airstone A780 at different curing temperatures')
plt.ylabel(r"Time [min]")
plt.xlabel(r"Curing temperature [$^\circ$Celsius]")
plt.show()
```



Comment: From the figure above, one can see that the time needed to reach a certain degree of cure comes down exponentially with increasing cure temperature. From 9.1, we saw that in order to reach a degree of cure around 0.999, one would need more than 36 hours at 80 degrees Celsius. If the curing temperature is increased to around 150 degrees Celsius, the time to reach α 0.999 is 83 min. This is the same time as was needed to reach α = 0.9 at 80 degrees Celsius curing temperature.

Thus by increasing the curing time, one can end up with a product that has a significantly higher T_g (and thus can be suitable for operating conditions at higher service temps), in the same processing time.

If one's goal is to cut down the curing time significantly without improving the final T_g of the part, one can look at increasing the curing temperature to around 125 degrees Celsius. This improves the curing time needed to reach $\alpha = 0.9$ to roughly 8.3 minutes, an order of magnitude quicker than curing at 80 degrees Celsius.

Both suggestions show that is very dependent on the paraemeters the team whishes to optimize as to which curing temperature would be a suitable option for them. However, with increasing curing temperatures, one must make sure that there will still be enough time left to properly infuse the part with the resin system, as the viscosity of a resin will also start to increase once higher degrees of cure are reached. Another consideration must be the materials used during the infusion and curing of the part. These must be able to withstand these higher curing temperatures. Think about mould materials, potential infusion consumables, etc. Also means to increase the curing temperatures can be costly, for example if an autoclave with higher curing temperatures is required.