jmakfit

March 1, 2019

1 Fitting phase transformation data

Phase transformations often involve change of density. Using dilatometry tests, one can measure the change in dimensions of a sample and thus quantify the extent of phase transformation that took place in the sample. The data is usually given as a function of temperature.

This data should be fit using the so called Johnson-Mehl-Avrami-Kolmogorov model. The equation for this model is given as below.

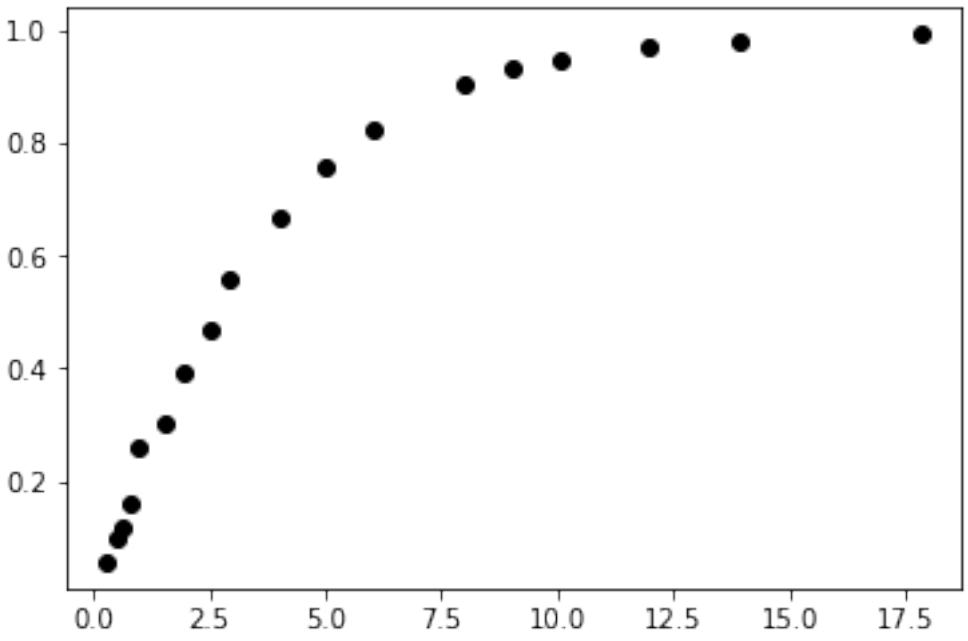
$f = 1 - \exp(-Kt^n)$

Here, f is the fraction transformed, t is time elapsed, K is the Avrami coefficient and n is the Avrami exponent.

We pick example data from Figure 6 of the journal article titled "Effect of cold deformation on the recrystallization behavior of FePd alloy at the ordering temperature using electron backscatter diffraction" by Hung-Pin Lin et al., Materials Characterization 94 (2014) 138-148. The article is available online at doi: 10.1016/j.matchar.2014.05.018. The data is entered in an array as follows.

```
In [1]: %matplotlib inline
        import numpy as np
        import matplotlib.pyplot as plt

        fdata = np.array([[0.2971562455462992, 5.6349544384787436],
        [0.4912199565971146, 9.791607726338654],
        [0.6070174158731206, 11.712932881086147],
        [0.812323106931194, 16.06202957845879],
        [0.987706134287128, 26.1157209788188],
        [1.5486828002791662, 30.117238141724172],
        [1.9331199126044512, 39.19161332243134],
        [2.5210160521584095, 46.958989967449405],
        [2.9326992033811985, 55.88071617550592],
        [3.998900521755244, 66.73122899522158],
        [4.991567172913602, 75.80560417592875],
        [6.0124394559244765, 82.44598993325107],
        [7.980689192649033, 90.53607458969762],
        [9.0318210269715, 93.12022583342922],
        [10.041534309538438, 94.89387632013383],
        [11.979925312058725, 97.14537094322145],
        [13.917111418627282, 98.26241322170168],
        [17.811810301100348, 99.36577616251056]], np.double);
```

```
In [2]: plt.figure(figsize=(6, 4))
        plt.plot(fdata[:,0],fdata[:,1]/100,'ko',label='data')
        plt.show()
```



We need to use a generic curve fitting tool and not a polynomial regression as the function we are fitting is an exponential one. For this, we use the optimization module from scipy. We need to define the function we want to fit.

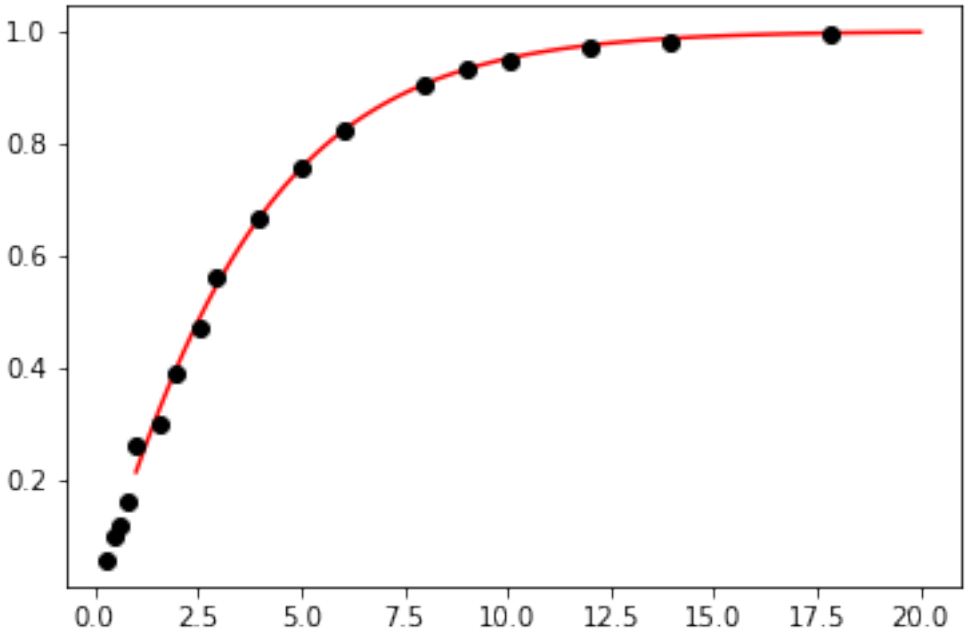
```
In [3]: from scipy.optimize import curve_fit
        from math import exp
        def jmak(x, k, n):
            return 1-exp(-k*x**n)

        x = fdata[:,0]
        y = fdata[:,1]/100
        p0 = [0.01, 3]
        B = [[0,1],[1,4]]
        jmak2 = np.vectorize(jmak)
        popt, pcov = curve_fit(jmak2, x, y, p0, bounds=B)
```

```
In [4]: print(popt)
```

```
[0.24021247 1.09949226]
```

The JAMK exponent can be said to be about 1.1 for this data and the fit.

```
In [5]: xx = np.linspace(1,20,100)
        yy = jmak2(xx, *popt)
        plt.plot(xx, yy, 'r-', label='JMAK fit')
        plt.plot(fdata[:,0],fdata[:,1]/100,'ko')
        plt.show()
```



The paper report the JMAK exponent to be 1.14 for this data set signifying heterogeneous site saturated nucleation of recrystallized grains. If this exponent were around 3, it signifies ideal recrystallization for homogeneous, continuous rate nucleation.

Ref: For further info on JAMK model, refer to Principles of Physical Metallurgy by Reed-Hill