9.5 Exercise 5

In this exercise we will translate the concepts of the previous exercise to real measured data, stored in a text file. We already used this data in exercise 1 as an example for plotting data from a text file. The curves are measured within a diffusion experiment where a polymer is filled with a diffusing gas. M_t is the amount (mass) of migrant loss within a polymer layer of thickness l at time t and M_∞ the final amount of migrant loss until equilibrium is reached. Assuming a constant (e.g. concentration independent) diffusion coefficient D the solution of the 1D diffusion equation can be written as an infinite series

$$\frac{M_t}{M_\infty} = 1 - \sum_{n=0}^{\infty} \frac{8}{(2n+1)^2\pi^2} \exp\left[-D(2n+1)^2\pi^2 t/l^2\right] \tag{9.6}$$

Luckily this equation can be simplified for different regions:

- For $M_t/M_\infty > 0.6$ it can be accurately replaced by

$$\frac{M_t}{M_\infty} = 1 - \frac{8}{\pi^2} \exp\left(\frac{-\pi^2 D t}{l^2}\right) \tag{9.7}$$

- For $M_t/M_\infty < 0.6$ it can be approximated with very little error by

$$\frac{M_t}{M_\infty} = \left(\frac{16D}{\pi l^2}\right)^{1/2} * t^{1/2} \tag{9.8}$$

The data stored in the text file represents the overall weight of the membrane as a function of time, i.e.

$$M(t) = M_{membrane} + M_\infty - M_\infty * \frac{M_t}{M_\infty} \tag{9.9}$$

So introducing one none-linear fitting parameter D/l^2 and two linear fitting parameter $M_{start} = M_{membrane} + M_\infty$ and M_∞ we can fit the whole curve to extract D/l^2 and thus by taking into account the known thickness of the membrane l the diffusion coefficient D.

- Discuss for the function `tga_load_plot` the mathematical transformation from the measured data to the plotted data
- What is the meaning of `M_infinite`?
- We will use the equation for $M_t/M_\infty < 0.6$ to find a good guess for the slope, i.e. the diffusion coefficient (Just as for finding zeros the numerical stability of minimization strategies can drastically be improved by using good/approximate starting points). So we will fit data to a straight line, for which in MATLAB the `polyfit` function can be used.
- Check the MATLAB HELP for `polyfit`. Fit the corresponding part of the measured curve and add the fit line in the plot.

```
function slope = tga_load_calc_slope
    fid = fopen('m202_thin.txt');
    A = textscan(fid,'%f %f %f %f %f %f','delimiter','space','headerlines',31);
    %31 headerlines are typical
    M_infinite = (A{2}(1)-A{2}(end));
    sqrt_t = A{1}.^(0.5);
    y = (A{2}(1)-A{2})./M_infinite;
    plot(sqrt_t,y);
    hold on;
    [~, index] = min(abs(0.6-y));
    [p,~] = polyfit(sqrt_t(1:index),y(1:index),1);
    slope = p(1);
    plot(sqrt_t(1:index),p(2)+slope.*sqrt_t(1:index))
end
```

- Write a nested function representing $M_t/M_\infty(x)$ with $x = D/l^2$, i.e. the output of this function is a column vector with as many components as the the measured data.

```
function slope = tga_load_calc_slope_draw_full
    fid = fopen('m202_thin.txt');
    A = textscan(fid,'%f %f %f %f %f %f','delimiter','space','headerlines',31);
    %31 headerlines are typical
    M_infinite = (A{2}(1)-A{2}(end));
    t = A{1};
    del_t = t(2)-t(1);
    sqrt_t = t.^(0.5);
    y = (A{2}(1)-A{2})./M_infinite;
    plot(sqrt_t,y);
    hold on;
    [~, index] = min(abs(0.6-y));
    [p,~] = polyfit(sqrt_t(1:index),y(1:index),1);
    slope = p(1);
    Ddlq = (pi ./ 16) * (slope.^2);
    plot(sqrt_t,non_scaled_fu(Ddlq));

        function y = non_scaled_fu(Ddlq)
           h1 = 16.*Ddlq./pi;
           index_new = round(0.36/h1/del_t);
           y= h1.^0.5.*sqrt_t;
           y(index_new:end) = 1 - (8./(pi.^2)) .* exp (-pi.^2 .* Ddlq .* t(index_new:end));
        end
end
```

- Adapt the `fit_test_non_linear` of Exercise 4 to perform the full fit to the measured data
- Discuss the quality of the full fit to the approximate results used as input for the full fitting.

```
function [D, M_start, DelM_inf] = tga_load_calc_full_fit
    fid = fopen('m202_thin.txt');
    A = textscan(fid,'%f %f %f %f %f %f','delimiter','space','headerlines',31);
    %31 headerlines are typical
    M_infinite = (A{2}(1)-A{2}(end));
    t = A{1};
    del_t = t(2)-t(1);
    sqrt_t = t.^(0.5);
    y_meas = -A{2};
    y = (A{2}(1)-A{2})./M_infinite;
    [~, index] = min(abs(0.6-y));
    [p,~] = polyfit(sqrt_t(1:index),y(1:index),1);
    slope = p(1);
    l = 50E-6;
    c = [1.0 2.0]';
    Ddlq = (pi ./ 16) * (slope.^2);
    Ddlq = fminbnd(@chi_sqr,Ddlq.*0.3,Ddlq.*3);
    M_start = c(1);
    DelM_inf = c(2);
    D = Ddlq.*l.^2;
    y_fit = DelM_inf.*non_scaled_fu(Ddlq);
    plot(sqrt_t,y_meas-M_start,sqrt_t,y_fit)

        function y = non_scaled_fu(Ddlq)
```

```
            h1 = 16.*Ddlq./pi;
            index_new = round(0.36/h1/del_t);
            y= h1.^0.5.*sqrt_t;
            y(index_new:end) = 1 - (8./(pi.^2)) .* exp (-pi.^2 .* Ddlq .* t(index_new:end));
         end

         function y=chi_sqr(Ddlq)
            E = [ones(size(t)) non_scaled_fu(Ddlq)];
            c = E\y_meas;
            y_fit = E*c;
            y_diff=y_fit-y_meas;
            y = y_diff'*y_diff;
         end
end
```

HOMEWORK 6

Discuss for the above function

- Which are the linear and which are the non-linear fitting parameter?
- What is the meaning of the variable `c` ?
- What is calculated by `y = y_diff'*y_diff` ?
- What is the meaning of `y(index_new:end)` ?