Curve fitting

Prof. Anil Kokaram : Assignment 2

1 Jan 2021

This assignment requires you to fit a curve to some biological data. You will experiment with polynomial and non-linear models. Please read this entire document before attempting the assignment as there is useful information that will help you in the NOTES section.

This assignment counts for 10% of your final mark. All your submitted code must obey the Matlab style rules as per the document in the MatlabGuides area of Blackboard. See blackboard for the assignment due date. When submitting please use the filename LASTNAME_FIRSTNAME_STUDENTID.m

The compressive properties of human muscle tissue are significant for impact biomechanics, surgical simulation and rehabilitation engineering. The most important loading direction is transverse to the muscle fibre direction, and tests performed at TCD on pig muscle samples using a universal testing machine showed nonlinear anisotropic behavior under quasi-static loading conditions [1]. In this assignment you will fit regression curves to data from a compression test on a sample performed in the transverse (or cross fibre XF) direction, i.e. corresponding to the blue curve in the figure below. You will see that the relationship between stress and strain is not linear, and therefore defining a constant value for slope of the stress strain curve (ie assuming that there is a Young's Modulus) will not give a very good fit to the data.

[1] M. Van Loocke, C.G. Lyons, and C.K. Simms. A validated model of passive muscle in compression. *Journal of Biomechanics*, pages 2999–3009, October 2005

The data is stored in a file muscle_data_2017.xls. Use the following Matlab code to read in the data.

```
raw_data = xlsread('muscle_data_2017.xlsx');
strain = raw_data(:,3);
stress = raw_data(:,4);
```

You will now explore two different models for the data. We will use the model formulation $\hat{y_k} = f(\phi_k)$ where $\hat{y_k}$ is our estimated stress given the measured strain, ϕ_k, for the kth measurement. The error between the observed stress y_k and the modelled (estimated) stress $\hat{y}_k$ is therefore $e_k = y_k - \hat{y_k}$.

1. Plot in figure 1, a plot of the observed stress (on the y-axis) y_k Vs strain (on the x-axis) (ϕ_k). Use a blue line of thickness 3.

2. You are required to fit the data to a polynomial model. The problem is to estimate the best or appropriate model order m. Test 5 model orders $m = [1 : 5]$. For each polynomial model with order m use the `polyfit` function in Matlab to estimate the coefficients of the polynomial fit to the data. Calculate the **sum squared error** for each polynomial fit and assign the resulting data to the vector `sse_per_m`. That vector should be 5 elements long because you will have tested 5 model orders. In figure 2, plot `sse_per_m` versus model order and use that to select the optimal model order $\hat{m}$ for this problem. Use a red line of thickness 3. Label the figure and axes appropriately. Assign your selected model order to the variable `my_m`.

3. Using your selected model order $\hat{m}$, calculate the estimated signal $\hat{y}_k$ and superimpose that line in red on the plot in figure 1. Assign your estimated signal $\hat{y}_k$ to the variable `est_stress`.

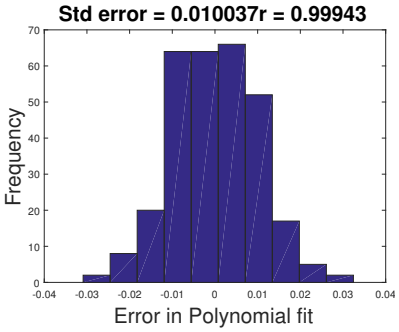
4. Calculate the error between your estimate and the observed stress, plot a histogram in figure 3 showing the distribution of errors. Your plot should look something like the plot shown on the side panel (no need for the title though). Note that the values shown in that plot may be different from your estimates. Calculate the coefficient of determination for this polynomial fit and assign that to the variable `ccoef_p`.

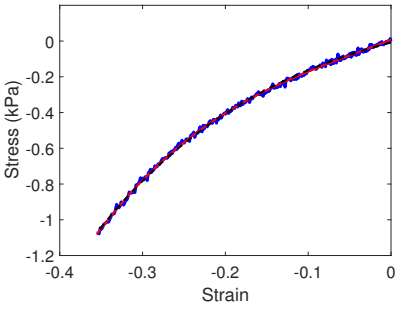
5. Now fit the data to a different nonlinear model as follows. (You will have to use `fminsearch` see the notes below).

$$\hat{y_k} = 1 - a_2 - (a_0 \exp(a_1 \phi_k)) \tag{1}$$

where $\hat{y}$ is the estimated stress given a strain measurement ϕ. You must start from an initial estimate $[0.3, 0.3, 0.3]$. Using your estimated values for a_0, a_1, a_2 calculate the estimated signal $\hat{y}_k$ and superimpose that line in black/dashed on the plot in figure 1. Your final figure 1 should look something like the plot in the side panel shown here.

You will have to use your notes or the book by Chapra to find the definition for the coefficient of determination.





6. Calculate the error between your estimate and the observed stress, and assign it to the variable `err_nl` . Plot a histogram in figure 4 showing the distribution of errors. (Just use the `hist` command in Matlab e.g. `hist(err_nl)`). Hence calculate the coefficient of determination for this non-linear fit, and assign that to the variable ccoef_nl.

Notes

Use » `help fminsearch` and » `help polyfit` from the matlab command line (») to find out more about these matlab functions and how to use them. For fminsearch you will need to pass arguments to your optimisation function which are not being estimated i.e. the data itself `stress, strain`. You may do this using the syntax `coefs = fminsearch(@(x) func(x, stress, strain),[0.3;0.3;0.3]);`. In this invocation you are declaring to fminsearch that the error measurement is in the function `func()`, and the `@(x)` is declaring that only the variable `x` in this function is being optimised. You will have to define the function `func()` yourself. It will be based on the model you are testing. In this case `x` should be a vector with your three model parameters $x(1) = a_0, x(2) = a_1, x(3) = a_2$. Hence `func()` then should calculate the sum squared error between the estimated stress using the model given (using your parameters `x`) and the actual measured stress, given the measured strain.

You might find it useful to know that Matlab string handling uses vector syntax like this.

```
str = ['This variable =', num2str(number), 'while that = ', num2str(value)];
```

The `num2str` function turns a number into a string for use in string handling like this. The string itself is in `str` and it just concatenates all the strings in the vector. You can then use this string for auto-composition of a title. For instance e.g. `title(str, 'fontsize' , 20)` will create a title that has the text as follows.

```
This variable = 78.2 while that = 7.1
```

When submitting your code for this assignment, please

note that it must obey the Matlab style rules as per the document in the MatlabGuides area of Blackboard.

References

[1] M. Van Loocke, C.G. Lyons, and C.K. Simms. A validated model of passive muscle in compression. *Journal of Biomechanics*, pages 2999–3009, October 2005.