Problem 1: Checking if a method is statistically better than another.

Finding efficient algorithms to describe, measure and compare shapes is a central problem in image processing. This problem arises in numerous disciplines that generate extensive quantitative and visual information. Among these, biology occupies a central place. For example, registration of brain anatomy is essential to many studies in neurobiology.

Here we consider the problem of aligning 38 brains (defined by their cortical surface) onto a template brain. Each brain consists of two independent hemispheres, Left, and Right. We have two methods for aligning the hemisphere of a brain onto the corresponding hemisphere of the template, Method1 and Method2. For each method, we align the hemisphere onto the template and check the quality of the alignment by comparing the positions of 35 regions between the aligned brain and the template. Figure 1 illustrates the positions of these regions, while Figure 2 shows one example of an alignment.

We will only consider the left hemisphere in this exercise. The files Left1.dat and Left2.dat (available on the web page) give the results of the alignments for Method1 and Method2, respectively. Each file includes a large table with 35 rows, for the 35 regions of interest, and 38 columns, for the 38 brains considered. The value for a row I and a column J gives the distance between the region I in brain J and the same region in the template brain. Note that the lower the distance, the better.
For each region, compare the two distributions of distance values obtained over all 38 brains for the two alignment methods, indicate if these distributions are significantly different (at 95% confidence level), and which of the two methods works best. How would you present these results?

Script for generating results:

```matlab
% Read in data from both methods
load Left1.dat
load Left2.dat

% Find number of regions
s = size(Left1);
nregions = s(1);
nbrains = s(2);

% Build rtable table
rtable = cell(36,6);

% Analyse all regions
for i = 1:nregions
    val1 = Left1(i,:);
    val2 = Left2(i,:);
    m1   = mean(val1);
    m2   = mean(val2);
    rtable{i+1,1}=i;
    rtable{i+1,2}=m1;
    rtable{i+1,3}=m2;
    if m1 > m2
        rtable{i+1,4} = 'Method 2';
    else
        rtable{i+1,4} = 'Method 1';
    end
    [h,p]=ttest2(val1,val2);
    rtable{i+1,6} = p;
```
if p < 0.05
    rtable{i+1,5} = 'Y';
else
    rtable{i+1,5} = 'N';
end
end

% Generate plot as a bar graph that shows the difference of the mean, with significant % differences in blue, and non significant differences in red %
n_significant = 0
n_nonsignificant = 0
for i = 1:35
    if rtable{i+1,6} < 0.05
        n_significant = n_significant + 1;
        region_sig(n_significant) = i;
        diff_sig(n_significant) = rtable{i+1,2}-rtable{i+1,3};
    else
        n_nonsignificant = n_nonsignificant + 1;
        region_nonsig(n_nonsignificant) = i;
        diff_nonsig(n_nonsignificant) = rtable{i+1,2}-rtable{i+1,3};
    end
end
% bar(region_sig,diff_sig,’b’);
hold on
bar(region_nonsig,diff_nonsig,’r’)
%
xlabel(‘Region #’);
ylabel(‘Mean (method1) – Mean(method2)’);
legend(’Significant’,’Not significant’);
title(‘Comparing two methods for brain matching’);

Problem 2: Polynomial fitting

(based on Chapter 5: Least squares, from Moler’s book, Numerical computing with Matlab)

NIST, the National Institute of Standards and Technology, is the branch of the U.S. Department of Commerce responsible for setting national and international standards. NIST maintains Statistical Reference Datasets, StRD, for use in testing and certifying statistical software (see http://www.itl.nist.gov/div898/strd/). This lab involves one of these datasets, Filip. There dataset is available on the class web page as filip.dat, or directly from the web page: http://www.itl.nist.gov/div898/strd/lls/data/Filip.shtml.
The data consist of several dozen observations of a variable \( y \) at different values of \( x \) (see figure below). The task is to model \( y \) by a polynomial of degree 10 in \( x \). Let \( N \) be the number of data points and \( P \) the number of parameters in the polynomial model (\( P=11 \)):

\[
y_{10} = a_{10} x^{10} + a_9 x^9 + a_8 x^8 + a_7 x^7 + a_6 x^6 + a_5 x^5 + a_4 x^4 + a_3 x^3 + a_2 x^2 + a_1 x + a_0
\]

This data set is controversial. Some mathematical and statistical packages are able to reproduce the polynomial coefficients that NIST has decreed to be the “certified values.” Other packages give warning or error messages that the problem is too badly conditioned to solve. A few packages give different coefficients without warning. The Web offers several opinions about whether or not this is a reasonable problem. Let us see what Matlab does with it.

**Basic fitting**

Load the data into Matlab, plot it with ‘.’ as the line type, and then invoke the Basic Fitting tool available under the Tools menu on the figure window. Select the 10th-degree polynomial fit. You will be warned that the polynomial is badly conditioned, but ignore that for now. How do the computed coefficients compare with the certified values on the NIST Web page? How does the plotted fit compare with the graphic on the NIST Web page? The basic fitting tool also displays the norm of the residuals, \(|r|\). Compare this with the NIST quantity “Residual Standard Deviation,” which is

\[
\frac{|r|}{\sqrt{N - P}}
\]

Build the following table:
<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Polyfit</th>
<th>Certified coefficients (NIST)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a10</td>
<td>-4.029 E-05</td>
<td>-4.029 E-05</td>
</tr>
<tr>
<td>a9</td>
<td>-0.00247</td>
<td>-2.47 E-03</td>
</tr>
<tr>
<td>a8</td>
<td>-0.067</td>
<td>-0.067</td>
</tr>
<tr>
<td>a7</td>
<td>-1.622</td>
<td>-1.062</td>
</tr>
<tr>
<td>a6</td>
<td>-10.875</td>
<td>-10.87</td>
</tr>
<tr>
<td>a5</td>
<td>-75.124</td>
<td>-75.12</td>
</tr>
<tr>
<td>a4</td>
<td>-354.48</td>
<td>-354.47</td>
</tr>
<tr>
<td>a3</td>
<td>-1128</td>
<td>-1127.97</td>
</tr>
<tr>
<td>a2</td>
<td>-2316.4</td>
<td>-2316.37</td>
</tr>
<tr>
<td>a1</td>
<td>-2772.2</td>
<td>-2772.18</td>
</tr>
<tr>
<td>a0</td>
<td>-1467.5</td>
<td>-1467.5</td>
</tr>
<tr>
<td>Residual standard deviation</td>
<td>0.0033</td>
<td>0.0033</td>
</tr>
</tbody>
</table>

2) Statistics on estimated parameters

Use the Matlab function `fit` to compute the coefficients and the corresponding confidence intervals. How do these values compare with the NIST certified values?

*Note: you will need to define the function for “fit”:*

\[
g = fittype({'x^10', 'x^9', 'x^8', 'x^7', 'x^6', 'x^5', 'x^4', 'x^3', 'x^2', 'x', '1'})
\]

*Then you will use something like:*

\[
[p,good]=fit(x,y,g)
\]
>> g=fitype(['x^10', 'x^9', 'x^8', 'x^7', 'x^6', 'x^5', 'x^4', 'x^3', 'x^2', 'x', '1'])

g =

Linear model:
\[ g(a,b,c,d,e,f,g,h,k,l,m,x) = a \cdot x^{10} + b \cdot x^9 + c \cdot x^8 + d \cdot x^7 + e \cdot x^6 + f \cdot x^5 + g \cdot x^4 + h \cdot x^3 + k \cdot x^2 + l \cdot x + m \]

>> [p4,good]=fit(x,y,g)
Warning: Equation is badly conditioned. Remove repeated data points
or try centering and scaling.
> In fit>handlewarn at 792
In fit at 520

p4 =

Linear model:
\[ p4(x) = a \cdot x^{10} + b \cdot x^9 + c \cdot x^8 + d \cdot x^7 + e \cdot x^6 + f \cdot x^5 + g \cdot x^4 + h \cdot x^3 + k \cdot x^2 + l \cdot x + m \]

Coefficients (with 95% confidence bounds):
\[
\begin{align*}
a &= -4.03e-05 \quad (-5.807e-05, -2.242e-05) \\
b &= -0.002468 \quad (-0.003536, -0.0014) \\
c &= -0.06702 \quad (-0.09541, -0.03863) \\
d &= -1.062 \quad (-1.504, -0.6203) \\
e &= -10.88 \quad (-15.34, -6.415) \\
f &= -75.12 \quad (-105.6, -44.64) \\
g &= -354.5 \quad (-497.3, -211.6) \\
h &= -1128 \quad (-1581, -674.9) \\
k &= -2316 \quad (-3247, -1386) \\
l &= -2772 \quad (-3888, -1656) \\
m &= -1467 \quad (-2062, -873.1)
\end{align*}
\]

good =

\[
\begin{align*}
\text{sse:} & \quad 7.958513852749903e-04 \\
\text{rsq:} & \quad 0.996727416172864 \\
\text{dfe:} & \quad 71 \\
\text{adjrsq:} & \quad 0.996266488873268 \\
\text{rmse:} & \quad 0.003348010519770
\end{align*}
\]