MAT 581 HOMEWORK 27 (DUE MONDAY 11/07/16)

WRITTEN PART: turn in at the beginning of class on Monday 11/07/16.

Suppose we want to fit a function of the form $\beta_1 e^{\beta_2 t}$ to the data

t	0	1	2
y	2	5	13

Write down the function $S(\beta_1, \beta_2)$ which gives the sum of squares of residuals. By equating its partial derivatives to 0, form a system of equations for β_1, β_2. (Do not solve the system).

COMPUTATIONAL PART: turn in on Blackboard by the end of Monday 11/07/16.

Goal: Fit a curve to data using nonlinear least squares; compare the performance of two models. The data are posted at `http://goo.gl/5ksf3i` (clickable link is on Blackboard). The models are:

- Model A: $y = \beta_1 e^{\beta_2 t}$
- Model B: $y = \beta_1 e^{\beta_2 t} + \beta_3$

Method: Define a function for training each model; it returns the squared norm of the residual vector.

```
t = (0:2:10)';
y = [...]';         % training set
training_A = @(beta) norm(beta(1)*exp(beta(2)*t)-y)^2;
training_B = @(beta) norm(beta(1)*exp(beta(2)*t)+beta(3)-y)^2;
```

Then find the minimal beta parameter for each model, by minimizing its training function.

```
beta_A = fminsearch(training_A, [0; 0]);
beta_B = fminsearch(training_B, [0; 0; 0]);
fprintf('Model A has residual %f in training\n', training_A(beta_A));
fprintf('Model B has residual %f in training\n', training_B(beta_B));
```

Since Model B includes all functions used in Model A, it is guaranteed that Model B will have a smaller residual in training. If this isn't the case, `fminsearch` failed to obtain the real minimum. Also, the given data sets are such that the residual should be less than 1; if you get residual much greater than 1, that also means that `fminsearch` failed.

What to do if minimization fails? Try another starting point. For example, replace the first coordinate of the starting point by the first value of the y-vector; this should be a more reasonable starting point than 0. If this also fails, try using 1/2 of the first value of y-vector.

Note for Scilab users: instead of anonymous functions, you'd have things like

```
function S = training_A(beta)
    S = norm(beta(1)*exp(beta(2)*t)-y)^2;
endfunction
```

After you get reasonable values (both residuals are small, and the one for Model B is smaller), proceed to testing.

```
tt = (1:2:9)';
yy = [...]';    % testing data set
testing_A = @(beta) norm(beta(1)*exp(beta(2)*tt)-yy)^2;
testing_B = @(beta) norm(beta(1)*exp(beta(2)*tt)+beta(3)-yy)^2;
fprintf('Model A has residual %f in testing\n', testing_A(beta_A));
fprintf('Model B has residual %f in testing\n', testing_B(beta_B));
```

Notice that no minimization happens here: we just plug the beta-values found during training into the testing data set. The model with smaller residual on the testing set wins the competition. (Which one it is depends on your specific data set.)

As a final step, plot the curve given by the winning model.

```
plot([t;tt], [y;yy], 'r*')
hold on
s = linspace(0,10);
plot(s, ... )              % use the winning model here
hold off
```