How To Use FitNumDifSR.m

1. Verify that FitNumDifSR.m is the appropriate template file to use
   a. The data points must be of the form $\left(x_1, x_2, \cdots, x_{n_s}, \hat{y}\right)$
   b. The model being fit to those data must be of the form $y = f\left(u_1, u_2, \cdots, u_{n_u}, x_1, x_2, \cdots, x_{n_s}\right)$ where the u_i are found by solving a set of differential model equations of the following form:
      i.
$$\begin{gathered}
\frac{du_1}{dv} = g_1\left(v, u_1, u_2, \cdots, u_{n_u}, \theta_1, \theta_2, \cdots, \theta_{n_p}; x_1, x_2, \cdots, x_{n_s}\right); \quad u_1(0) = u_1^0 \\
\frac{du_2}{dv} = g_2\left(v, u_1, u_2, \cdots, u_{n_u}, \theta_1, \theta_2, \cdots, \theta_{n_p}; x_1, x_2, \cdots, x_{n_s}\right); \quad u_2(0) = u_2^0 \\
\vdots \\
\frac{du_{n_u}}{dv} = g_{n_u}\left(v, u_1, u_2, \cdots, u_{n_u}, \theta_1, \theta_2, \cdots, \theta_{n_p}; x_1, x_2, \cdots, x_{n_s}\right); \quad u_{n_u}(0) = u_{n_u}^0
\end{gathered}$$
      ii. and $\underline{u}^0$ may include the set variables, $\underline{x}$.
2. Save a copy of FitNumDifSR.m as *newname*.m in the current MATLAB working directory or in a directory that is in the MATLAB search path ("*newname*" should be some meaningful file name)
3. Change the function declaration statement to match the filename from step 2
   a. from: `function FitNumDifSR(p_guess)`
   b. to: `function newname(p_guess)`
4. Find the comment indicating the location of the first required file modification
   a. Replace
      i.
```
% EDIT HERE (Required modification 1 of 4):
        % define universal and experimental constants here
```
   b. With statements defining variables and assigning their values for each constant that appears in the problem being solved. Universal constants like the ideal gas constant should be defined here, as well, and all the values should have consistent units.
5. Find the comment indicating the location of the second required modification and replace it with a statement defining a matrix named x
   a. There should be one row in the matrix x for each data point in the data set being fit
   b. There should be one column in the matrix x for each set variable in the data set
   c. The matrix x should contain the values of the corresponding set variables and data points
6. Find the comment indicating the location of the third required modification and replace it with a statement defining a column vector named y_hat
   a. There should be one row in the y_hat for each data point in the data set being fit and it should contain the corresponding value of the measured response for that data point

7. Find the comment indicating the location of the fourth required file modification and change the lines that follows the comment
   a. from:
```
dudv = [
    % Evaluate du1/dv = g1(v,u(1),...,u(n);p(1),...,p(n_par)) here
    % Evaluate du2/dv = g2(v,u(1),...,u(n);p(1),...,p(n_par)) here
    % and so on through gn, one per line
];
```
   b. so that the first line within the square brackets evaluates the function g_1 in step 1.b.i, the second line evaluates the function g_2 in step 1.b.i, and so on.
   c. If the parameters or set variables are needed in order to evaluate the functions, g, they are available in the column vectors `p` and `x_set`, respectively.
8. Find the comment indicating the location of the fifth required file modification and change the line that follows the comment
   a. from:
```
v0 = ; % insert the independent variable initial value here
u0 = [
    % insert the initial values of dependent variables u1 here
    % insert initial values for u2, u3, ..., one per line
];
vf = ; % insert the independent variable final value here
```
   b. so that the initial and final values of the independent variable for data point i are assigned to `v0` and `vf`, respectively, and
   c. so that the first line within the square brackets calculates $u_1(0)$ from the equations in step 1.b.i, the second line calculates $u_2(0)$, and so on.
   d. If the set variables or response variables are needed in order to calculate the guesses they are available in the column vectors `x_set` and `y_hat`, respectively.
9. Find the comment indicating the location of the sixth and final required file modification and change the line that follows the comment
   a. from: `y(i) = ; % insert statement(s) to calculate y for data point i`
   b. so that it evaluates the function f in step 1b and sets `y(i)` equal to the result
   c. If the set variables are needed in order to evaluate the function, f, they are available in the column vectors `x_set`.
10. If the ODEs in step 1.b.i are stiff, change the solver from ode45 to ode15s
11. Save the modified version of newname.m (where newname is the filename chosen in step 2)
12. Create a matrix named x in the MATLAB workspace

   a. The first column of `x` should contain the values of x_1 for each of the data points, one per row; the second column should similarly contain the values of x_2, and so on for all set variables
13. Create a column vector named `y_hat` in the MATLAB workspace; it should contain the values of $\hat{y}$ for each of the data points, one per row
14. Create a column vector named `p_guess` in the MATLAB workspace; it should contain guesses for the values of the parameters, $\underline{\theta}$, one per row
15. Execute the file by typing the following at the MATLAB command prompt (again using "newname" to represent the filename chosen in step 2): `newname(p_guess)`
16. The following quantity will be listed in the MATLAB command window
    a. `r_squared` - the correlation coefficient for the fit
17. The following quantities will be returned
    a. `pf` - a column vector containing the fitted parameters
    b. `pf_u` - a column vector containing the ± 95% confidence limits for the fitted parameters
    c. y - a column vector containing the values of the response variable predicted by the fitted model
18. The following figures will be displayed
    a. If there is one set variable per data point
        i. A model plot
    b. If there are two or more set variables per data point
        i. A parity plot
        ii. A set of residuals plots with each of the set variables as the abscissa
19. Copy the values of `pf` to `p_guess` and repeat step 15
    a. Repeat this step until the values returned as `pf` equal the values in `p_guess` and none of the other returned quantities have changed indicating a converged minimization
20. To search for a different minimum of the objective function, repeat steps 14 through 19 using a significantly different `p_guess` in step 9