# How To Use SolvBVDif.m

1. Verify that SolvBVDif.m is the appropriate template file to use
   a. First order ordinary differential equations of the following form are to be solved

$$\frac{dy_1}{dx} = f_1\left(x, y_1, \cdots, y_n\right); \qquad a \le x \le b$$
$$\vdots$$
$$\frac{dy_n}{dx} = f_n\left(x, y_1, \cdots, y_n\right); \qquad a \le x \le b$$

   b. The boundary conditions are specified in terms of the values of the dependent variables at both boundaries, $x = a$ and $x = b$:

$$g_1\left(y_1(a), \cdots, y_n(a), y_1(b), \cdots, y_n(b)\right) = 0$$
$$\vdots$$
$$g_n\left(y_1(a), \cdots, y_n(a), y_1(b), \cdots, y_n(b)\right) = 0$$

   c. The functions, $f$, do not have singularities in the range $a \le x \le b$
2. Save a copy of SolvBVDif.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 result = SolvBVDif`
   b. to: `function result = newname`
4. Find the comment indicating the location of the first required file modification
   a. Replace

```
i. % EDIT HERE (Required modification 1 of 6):
         % define universal and experimental constants here
```

   b. With statements entering values for each constant that appears in the problem being solved
5. Find the comment indicating the location of the second required file modification and change the lines that follow the comment
   a. from:

```
dydx = [
    % Evaluate dy1/dx here
    % Evaluate dy2/dx here
    % and so on, one dyi/dx per line
];
```

   b. so that the first line within the square brackets evaluates the function $f_1$ in step 1.a, the second line evaluates the function $f_2$ in step 1.a, and so on

6. Find the comment indicating the location of the third required file modification and change the lines that follow the comment

   a. from:

```
res = [
    % Calculate the error in the first boundary condition here
    % Calculate the error in second boundary condition here
    % and so on, one boundary condition per line
];
```

   b. so that the first line within the square brackets evaluates the function $g_1$ from step 1.b, the second line evaluates the function $g_2$ from step 1.b, and so on

      i. The column vector y_at_start contains values of $\underline{y}$ at $x = a$

      ii. The column vector y_at_end contains the values of $\underline{y}$ at $x = b$

7. Find the comment indicating the location of the fourth required modification and change the lines that follow

   a. from

```
x_range_low = % insert starting value of the valid range of x here
x_range_high = % insert ending value of the valid range of x here
n_mesh_points = % insert the number of mesh points here
```

   b. so that x_range_low is set equal to $a$, the lower end point of the valid range of $x$

   c. so that x_range_high is set equal to $b$, the lower end point of the valid range of $x$

   d. and n_mesh_points is set equal to the desired number of mesh points

8. Find the comment indicating the location of the fifth required modification and change the lines that follow

   a. from

```
yinit = [
    % Enter a guess for y1 (one value for the whole x range) here
    % Enter a guess for y2 (one value for the whole x range) here
    % and so on, one guess per line
];
```

   b. so that the first line within the square brackets provides a guess for the average value of $y_1$, the second line provides a guess for the average value of $y_2$, and so on

9. Find the comment indicating the location of the sixth and final required file modification and change the lines that follow the comment

   a. from:

```
% Calculate any other desired quantities from the results, noting
% result.x(i) contains the final x value of mesh point i
% result.y(i,j) contains the final value of yj at mesh point i
```

   b. so that any additionally needed quantities that depend upon the unknowns are calculated

   i. Do not use semicolons at the ends of these statements; if you do, they will not appear in the output

10. Save the modified version of newname.m (where newname is the filename chosen in step 2)
11. Execute the file by typing the following at the MATLAB command prompt (again using “newname” to represent the filename chosen in step 2): `result = newname`
12. The results of the code entered in step 9.b will be listed in the MATLAB command window
13. The structure `result` will be returned and available within the MATLAB workspace
    a. `result.x` is a column vector containing the values of x at each of the final mesh points
    b. `result.y` is a matrix; `result.y(i,j)` contains the final value of $y_j$ at mesh point $i$