Using MATLAB for Laboratory Data Reduction

Data Collected

Munson, Young, and Okiishi [1] provide laboratory data for the measurement of the viscosity of water with a capillary tube viscometer. The viscometer consists of a vertically oriented capillary tube with a reservoir attached to the upper end. The lower end of the viscometer is open, and fluid flowing out of the bottom of the tube is collected in a device for measuring the fluid volume in a measured time interval. For a Newtonian fluid, the viscosity is linearly related to the volumetric flow rate through the small diameter tube.

Measurements with water at different temperatures yield the following data.

V (mL)	Δt (s)	T (°C)
9.2	19.8	15.6
9.7	15.8	26.3
9.2	16.8	21.3
9.1	21.3	12.3
9.2	13.1	34.3
9.4	10.1	50.4
9.1	8.9	58.1

The first column is the volume of water collected during the time interval listed in the second column. The third column is the temperature of the water during the experiment.

Data Reduction

The relationship between volumetric flow rate Q and kinematic viscosity ν for the viscometer is

$$Q = \frac{K}{\nu} \tag{1}$$

where K is the calibration constant for the viscometer. Equation (1) is consistent with common sense: increasing the viscosity will decrease the flow rate for a fixed pressure head. The first step in the data reduction is to convert the volume and time measurements to volumetric flow rate

$$Q = \frac{V}{\Delta t}. \tag{2}$$

The measured data at $T = 15.6\,°\mathrm{C}$ is used to find K for the viscometer. Using the measured Q and the value of $\nu(15.6\,°\mathrm{C})$ from a reference table, the calibration constant is

$$K = Q_{\text{measured}}\nu_{\text{reference}} \tag{3}$$

With K known, the rest of the measured data is converted to $\nu = f(T)$ by rearranging Equation (1), i.e.

$$\nu = \frac{K}{Q} \tag{4}$$

where K is determined from Equation (3) and Q is from the measured data with Equation (2).

Matlab Solution

The viscometerData function listed on the next page performs the data reduction. We will examine the code in viscometerData a few lines at a time.

The first task is to store the raw data. This is achieved by manually assigning the data to three row vectors (one-dimensional arrays), V, t, and T.

```
V = [ 9.2  9.7  9.2  9.1  9.2  9.4  9.1];   %  volume (mL)
t = [19.8 15.8 16.8 21.3 13.1 10.1  8.9];   %  time (s)
T = [15.6 26.3 21.3 12.3 34.3 50.4 58.1];   %  temperature (C)
```

Note that the units of each vector are indicated by the end-of-line comment statements. Also note that t and T are different variables: the case of variable names is significant in Matlab.

With the volume and time data stored in V and t, respectively, the volumetric flow rate is calculated with

```
Q = (V/1.0e6) ./ t;   %  volumetric flow rate (m^3/s)
```

The (V/1.0e6) subexpression converts the volume in mL to m^3. The *array operator* ./ is necessary because both V and t are vectors. The expression (V/1.0e6) ./ t produces a row vector with the same number of elements as V and t. The expression is equivalent to, but much more compact than, the explicit *for loop*

```
for i=1:length(V)
  Q(i) = V(i)/1.0e-6 / t(i);
end
```

The reference value of viscosity is computed by linear interpolation in a table of $\nu = f(T)$ data. Data from Table B.2 on page 831 of the book by Munson, Young, and Okiishi is stored in the Tnu and nuw vectors.

```
Tnu = [0      5      10     20     30     40     50     60];
nuw = [1.787  1.519  1.307  1.004  0.8009  0.6580  0.5534  0.4745] * 1e-6;
```

The first V and t data is obtained at $T = 15.6\,^\circ$C. Thus, the interpolation involves the third and fourth elements of the Tnu and nuw arrays.

```
nuref = nuw(3) + (T(1) - Tnu(3)) * (nuw(4) - nuw(3))/(Tnu(4) - Tnu(3))
```

A more general procedure for interpolating with Tnu and nuw would involve the built-in interp1 function which performs interpolation in a one-dimensional table. This expression

```
nuref = interp1(Tnu,nuw,T(1))
```

is equivalent to the preceding expression for nuref as long as T(1) lies between T(3) and T(4).

With the reference value of $\nu(15.6)$ known, the calibration constant for the viscometer is obtained by applying Equation (3).

```
K = nuref*Q(1)
```

Only the first element of the Q array is used because that is the value corresponding to the $T = 15.6\,^\circ$C data used to compute nuref. (Note the use of T(1) in the computation of nuref.)

With the calibration constant K known, the measured Q data is converted to viscosity with the expression

```
nu = K ./ Q;
```

Once again an array operator is necessary because we need to divide K by each element of Q to get nu. The preceding expression is equivalent to the explicit for loop

```
for i=1:length(Q)
  nu(i) = K / Q(i);
end
```

```
function dataReduction
% dataReduction  Convert viscometer data from MYO, Lab # 1-90
%
%  This m-file shows how to (1) store data in arrays, (2) perform array
%  calculations with .* and ./ operators, (3) plot data and label axes,
%  (4) sort data stored in different arrays, and (5) print data in a
%  nicely formatted table.

% --- Store data
V = [ 9.2  9.7  9.2  9.1  9.2  9.4  9.1];   %  volume (mL)
t = [19.8 15.8 16.8 21.3 13.1 10.1  8.9];   %  time (s)
T = [15.6 26.3 21.3 12.3 34.3 50.4 58.1];   %  temperature (C)

% Store reference data for viscosity of water: MYO, Table B.1, p. 831
% Tnu is temperature (C), nuw is kinematic viscosity (m^2/s)
Tnu = [0      5      10     20     30      40      50      60];
nuw = [1.787  1.519  1.307  1.004  0.8009  0.6580  0.5534  0.4745] * 1e-6;

% --- Begin data reduction
% Compute volumetric flow rate:  V/1.0e6 is volume in m^3
% Divide each volume measurement by time to fill volume:  Q = volume/time
% The ./ operator does element-by-element division of the V and t arrays.
Q = (V/1.0e6) ./ t;   %  volumetric flow rate (m^3/s)

% --- Use one operating condition to determine viscometer calibration K
%     and compute nu at remaining points
% Linear interpolation to find viscosity of water at 15.6 (C)
%  Manual:
%     nuref = nuw(3) + (T(1) - Tnu(3)) * (nuw(4) - nuw(3))/(Tnu(4) - Tnu(3))
nuref = interp1(Tnu,nuw,T(1))  %   works for any T(1) in range of Tnu data
K = nuref*Q(1)                 % Use reference viscosity to determine K
nu = K ./ Q;                   % Convert data.  ./ does vectorized division

% --- Plot results
plot(T,nu,'o',T(1),nuref,'+',Tnu,nuw,'r--');
xlabel('T ({}^\circ C)');
ylabel('\nu  (m^2/s)');
axis([0 60 1e-7 20e-7]);
grid on
legend('Measured','Calibration','Reference')

% --- Print results
%  Prepare by sorting data in order of increasing temperature
[T,is] = sort(T);   %  "is" is the sort order
nu = nu(is);        %  nu data is now in same order as T data
%  Interpolate to find reference viscosity at *all* measured temperatures
nur = interp1(Tnu,nuw,T);
fprintf('  T (C)    nu (m^2/s)  nu_ref (m^2/s) Diff (m^2/s)  %% Diff\n');
for i=1:length(T)
  Dabs = nu(i) - nur(i);
  Drel = 100*Dabs/nur(i);
  fprintf(' %6.1f  %12.3e  %12.3e  %12.3e  %6.2f\n',T(i),nu(i),nur(i),Dabs,Drel);
end
```

Listing 1: The `dataReduction` m-file performs all calculations necessary to convert the measured values to viscosity. It plots a comparison of the measured viscosity with data from reference [1], and prints the results in a table..

The next block of code in `viscometerData` plots the data. Three sets of data are plotted with

```
plot(T,nu,'o',T(1),nuref,'+',Tnu,nuw,'r--');
```

The first set is the reduced data `T,nu,'o'`, which is plotted with open circles at each point in the (`T`, `nu`) data pairs. The second set is the single point `T(1),nuref,'+'` used to obtain the calibration constant K. A plus sign identifies this point on the plot, and it should be coincident with one of the data pairs in the (`T`, `nu`) set. The third set of data is from the reference table `Tnu,nuw,'r--'`. This data is plotted with a dashed red line connecting each pair of (`Tnu`, `nuw`) points.

The plot is annotated by adding titles for the axes, slightly resizing the axes limits, adding a grid, and using a legend to label the three different sets of data.

```
xlabel('T ({}^\circ C)');
ylabel('\nu  (m^2/s)');
axis([0 60 1e-7 20e-7]);
grid on
legend('Measured','Calibration','Reference')
```

The last block of code prepares the reduced data for printing, and then prints the data in a nicely formatted table. First the data is sorted in order of increasing temperature

```
[T,is] = sort(T);   %  "is" is the sort order
nu = nu(is);        %  nu data is now in same order as T data
%  Interpolate to find reference viscosity at *all* measured temperatures
```

This is not necessary, but it is a nice convenience for someone reading the results. Next, to prepare for a quantitative comparison between the measured and published data, interpolation is used to determine viscosity values from the reference table

```
nur = interp1(Tnu,nuw,T);
```

The third argument to the `interp1` function is the entire vector `T`. The `interp1` function automatically finds the correct location in the (`Tnu`, `nuw`) table, and then performs the interpolation for each element in `T`.

Finally, the results are printed in a formatted table.

```
fprintf('  T (C)    nu (m^2/s)  nu_ref (m^2/s) Diff (m^2/s)  %% Diff\n');
for i=1:length(T)
  Dabs = nu(i) - nur(i);
  Drel = 100*Dabs/nur(i);
  fprintf(' %6.1f  %12.3e  %12.3e  %12.3e  %6.2f\n',T(i),nu(i),nur(i),Dabs,Drel);
end
```

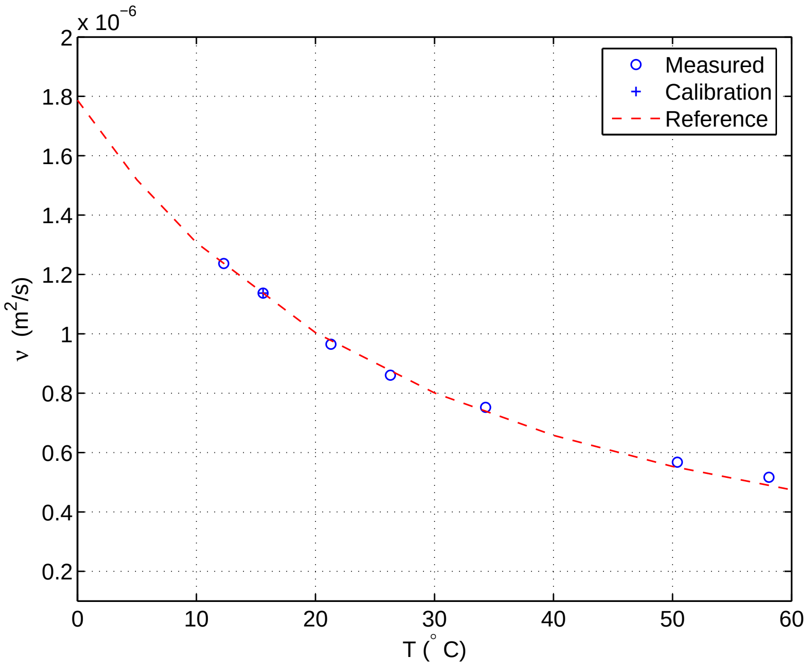
The `Dabs` and `Drel` variables are the absolute and relative differences (respectively) between the measured viscosity and the interpolated values from the reference table.
Running `viscometerData` produces the following text output and the plot below

```
>> viscometerData
nuref =
   1.1373e-06

K =
   5.2845e-13

  T (C)    nu (m^2/s)  nu_ref (m^2/s) Diff (m^2/s)  % Diff
   12.3    1.237e-006    1.237e-006   -3.846e-010   -0.03
   15.6    1.137e-006    1.137e-006    0.000e+000    0.00
   21.3    9.650e-007    9.776e-007   -1.260e-008   -1.29
   26.3    8.608e-007    8.760e-007   -1.527e-008   -1.74
   34.3    7.525e-007    7.395e-007    1.302e-008    1.76
   50.4    5.678e-007    5.502e-007    1.756e-008    3.19
   58.1    5.168e-007    4.895e-007    2.735e-008    5.59
```

The values of `nuref` and `K` are printed because the expressions that assign these variables are not terminated with a semicolon. Alternative methods for displaying the values of `nuref` and `K` involve using either the built-in `disp` function or the `fprintf` function.



References

[1] B. R. Munson, D. F. Young, and T. H. Okiishi. *Fundamentals of Fluid Mechanics*. Wiley, New York, fourth edition, 2002.