Substituting Polynomials in Place of Lookup Tables

There are times when it may be necessary to track sequences of points. Some examples of this include flow profiles in rivers, liquid volume in non-uniform tanks, sensor calibration or non-linear temperature profile testing. The traditional method for tracking sequences is with a lookup table, which consists of a series of points that correlate to measurements and between which points are linearly interpolated. Instead of using lookup tables, the dataTaker can use polynomials to represent point sequences. This technical note will describe how to do this using a voltage/temperature profile as an example.

1 Prerequisites

- TN-0023 - Thermocouple Calibration (required only to familiarise with the process of creating polynomials within Microsoft Excel)

2 Required Equipment

- *dataTaker* DT80 range data logger
- Sensor of choice
- PC with Microsoft Excel

3 Process

This process assumes that you already have a list of point values which you would like the logger to use as a lookup table. It is also assumed that the list is in a format which can be opened or imported into Microsoft Excel. In this example the data is stored in two columns, the first being the test time in seconds and the second for the corresponding point value.

	A	B	C
1	TIME	VALUE	
2	1	149.512	
3	2	148.8788	
4	3	148.1762	
5	4	147.4242	
6	5	146.6333	
7	6	145.81	
8	7	144.9586	
9	8	144.0824	
10	9	143.1841	
11	10	142.2654	
12	11	141.3283	
13	12	140.3738	

Figure 1 - Data format in Microsoft Excel

3.1 Converting the profile into its polynomial equivalent

Load the profile data into Microsoft Excel, select the time and data columns and create a X-Y Scatter Chart. In this example we have a profile created from 600 samples over a period of 10 minutes. The profile appears to be made of three separate trends.

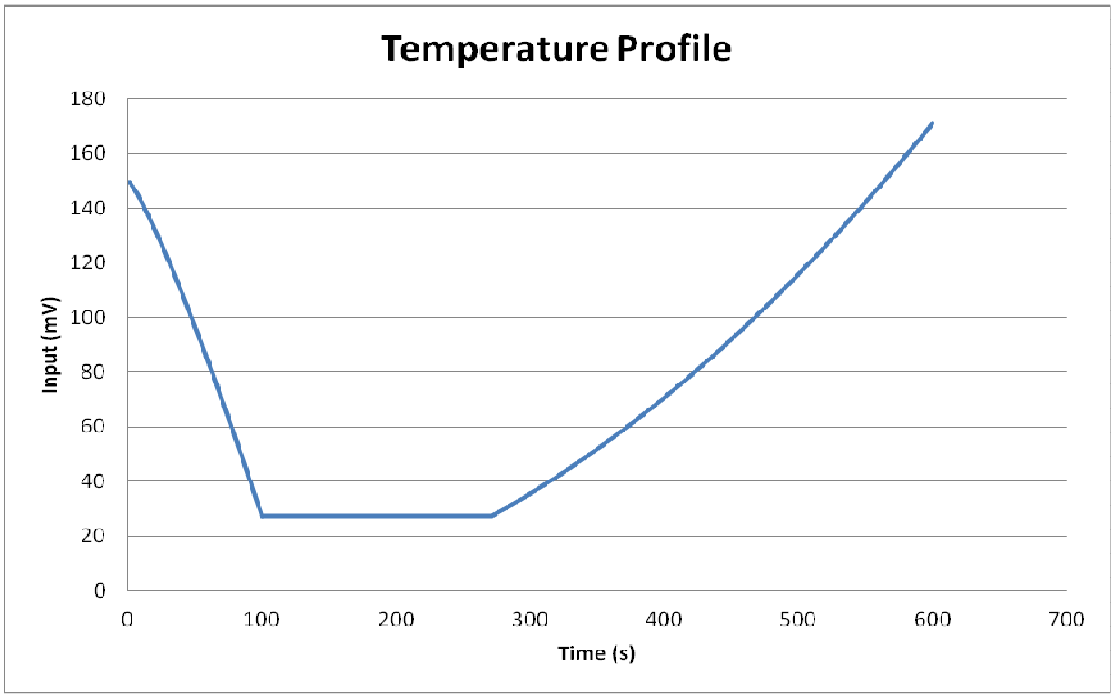

Figure 2 – Example temperature profile

Using those instructions provided in TN-0023, create one or more polynomials which best represent the temperature profile. If the profile has sharp discontinuities or defined regions similar to the example above, then it is best to separate the data into regions and create individual polynomials. To do this, cut and paste the values for each range into a different column whilst maintaining the same X values as shown in Figure 3, then plot them all together on the same chart as shown in Figure 4 before creating your polynomials.

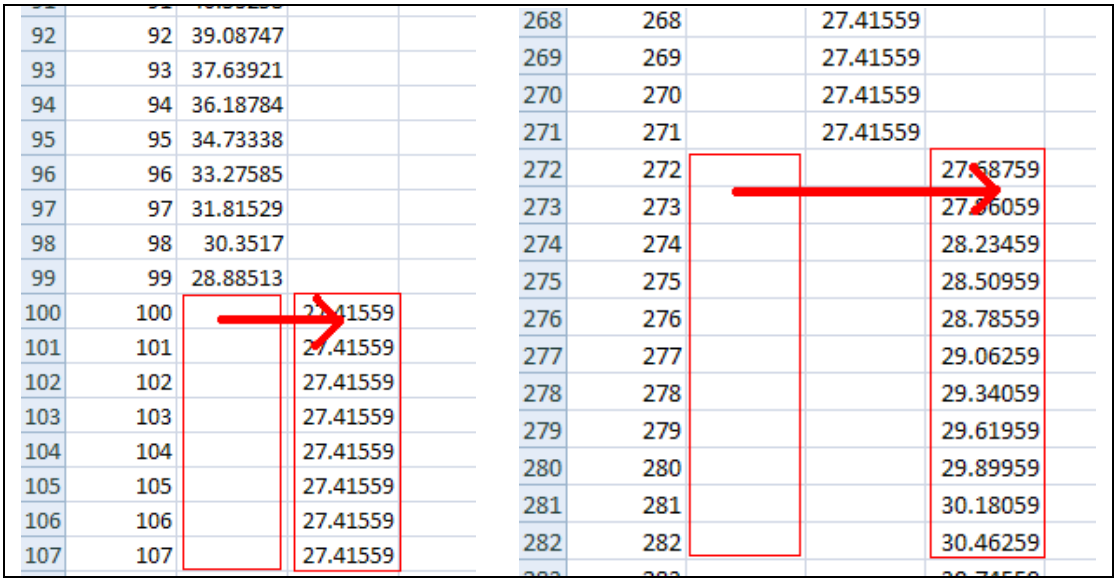

Figure 3 - Splitting the dataset into separate columns

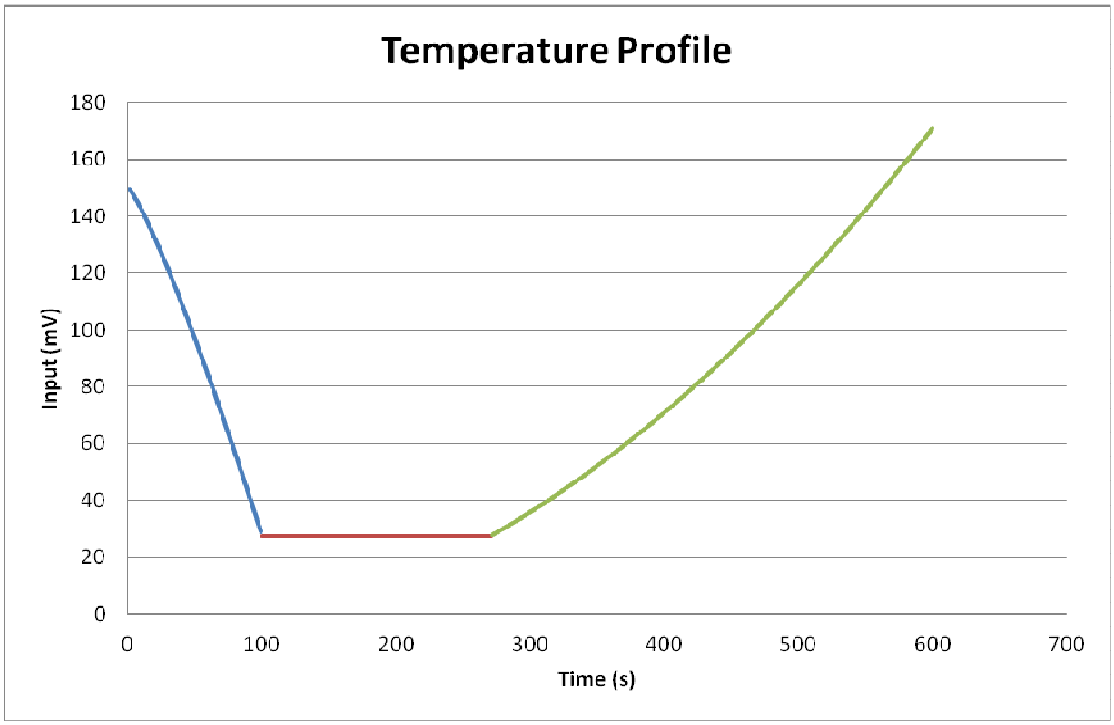

Figure 4 - Split temperature profile

As shown in Figure 4, the temperature profile has been split into three discrete profiles, each of which can be easily replicated using separate polynomials. Figure 5 shows the same chart with the polynomial traces and equations overlaid.

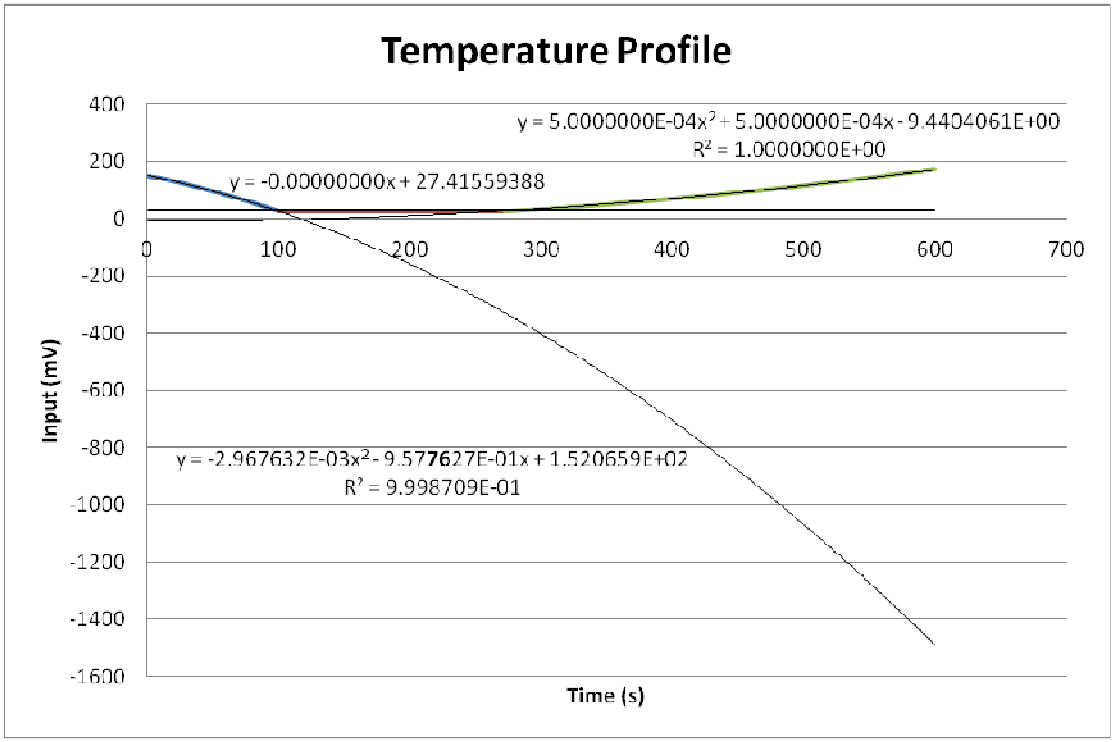

Figure 5 - Split temperature profile showing polynomial equations

Converting the Excel-generated polynomials into the dataTaker version requires reversing. These are the three polynomials for our dataset represented in their dataTaker equivalent format:

```
Y1=1.520659E2,-9.577627E-1,-2.967632E-03   'Between 000 and 100
Y2=27.41559388                             'Between 101 and 271
Y3=-9.4404061E0,5E-4,5E-4                  'Between 272 and 600
```

3.2 Correlating the test time to the polynomial values

It is necessary to track the total test time because the polynomials relate the time to the measured values. The best way is to create a channel and use the **DT** channel option, which returns the exact time between readings. This time difference is regularly measured and summed to obtain the total test time. For simplicity this example uses a schedule that runs at one-second intervals because the original data had periods set to one second, however the timing of the schedule is not critical.

The simple code for this is:

```
RA1S 'run schedule A at 1 second intervals
T(W,DT,+=1CV) 'store total test time into 1CV
```

Applying the polynomial to the variable containing the current test time retrieves the corresponding profile value. This can also be saved to another variable for use in alarms using the =nCV channel option.

```
1CV("PROFILE",Y1,=2CV)
```

If the profile is made up of multiple polynomials then it is also necessary to identify which profile to be use for different times throughout the test. To do this, create a series of **IF** conditions as follows. Note that the greater than symbol “>” actually means “greater than or equal to” so for the first line below if the time (1CV) is greater than or equal to 0 seconds but less than 101 seconds then apply polynomial Y1 and save the value into 2CV.

```
IF(1CV><000,101){1CV("PROFILE-A",Y1,=2CV)}
IF(1CV><101,272){1CV("PROFILE-B",Y2,=2CV)}
IF(1CV><272,601){1CV("PROFILE-C",Y3,=2CV)}
```

3.3 Letting the logger start the test

The logger needs to know exactly when the test begins in order to synchronise the time to the profile. It is therefore recommended to have the logger trigger the start of the test. In this example the logger asserts digital output #1 to signify the start of the test. This could be used to switch a relay to power up the test equipment. Note that this action should occur within a timed schedule and not within the “immediate” or ”on logger activation” schedule, otherwise the test and profile may be out of sync. One could expand on this functionality and have the logger reset the test via a switch or through the function buttons.

3.4 Define the boundary conditions

In this example we trigger an alarm when the temperature falls outside boundary conditions. These conditions can be determined by a percentage, a predetermined amount or other polynomials. In this example a +/- 10mV acceptance criteria will be sufficient and is easy to implement. When a measurement falls outside this boundary the warning lamp on the dataTaker will alight. The output could alternatively be changed to a signal on a digital channel or a schedule trigger, email alarm or other.

3.5 The full *dataTaker* program code

```
BEGIN"PROFILE"
'=====================================================
'Profile Program
'    - Uses a set of polynomials as a form of profile "look up table"
'    - This example constructs a profile from three polynomials
'    - If the measured voltage does follow lie within the boundaries
'      set by the profile then an alarm will be triggered
'=====================================================
'=====================================================
'Initialise variables
       LOGON
       'set up Polynomial for predicted profile path
       Y1=1.520659E2,-9.577627E-1,-2.967632E-03  'Between 000 and 100
       Y2=27.41559388                            'Between 101 and 271
       Y3=-9.4404061E0,5E-4,5E-4                 'Between 272 and 600
       1CV(W)=0                                  'reset the test timer
'=====================================================
'=====================================================
' Schedule A (MAIN PROFILE SCHEDULE)
'    - Runs every 1 second (matches the x axis scale on the profile)
'=====================================================
RA("B:",ALARMS:OV:10KB,DATA:OV:1MB)1S LOGONA
       1DSO(W)=1 'test in progress signal (or turn on equipment).
       '1CV = sum of the time since the program started
       T(W,DT,+=1CV)
       1CV("TOTAL TIME")
       'determine which profile to use, based on the current time
       'if time between 000 and 100, use Y1
       IF(1CV><000,101){1CV("PROFILE-A",Y1,=2CV)}
       'if time between 101 and 271, use Y2
       IF(1CV><101,272){1CV("PROFILE-B",Y2,=2CV)}
       'if time between 272 and 600, use Y3
       IF(1CV><272,601){1CV("PROFILE-C",Y3,=2CV)}
       '2CV = profile value that corresponds with current time
       2CV("PROFILE")
       'set the boundaries for this point in time
       3CV("UPPER")=2CV+10
       4CV("LOWER")=2CV-10
       'take the measurement (in this case a voltage), save into 5CV
       1V("MEASUREMENT~mV",=5CV)
       'if the value is outside the boundaries, turn on warning light
       IF(5CV<>4CV,3CV)1WARN
END
```

4 Program output/results

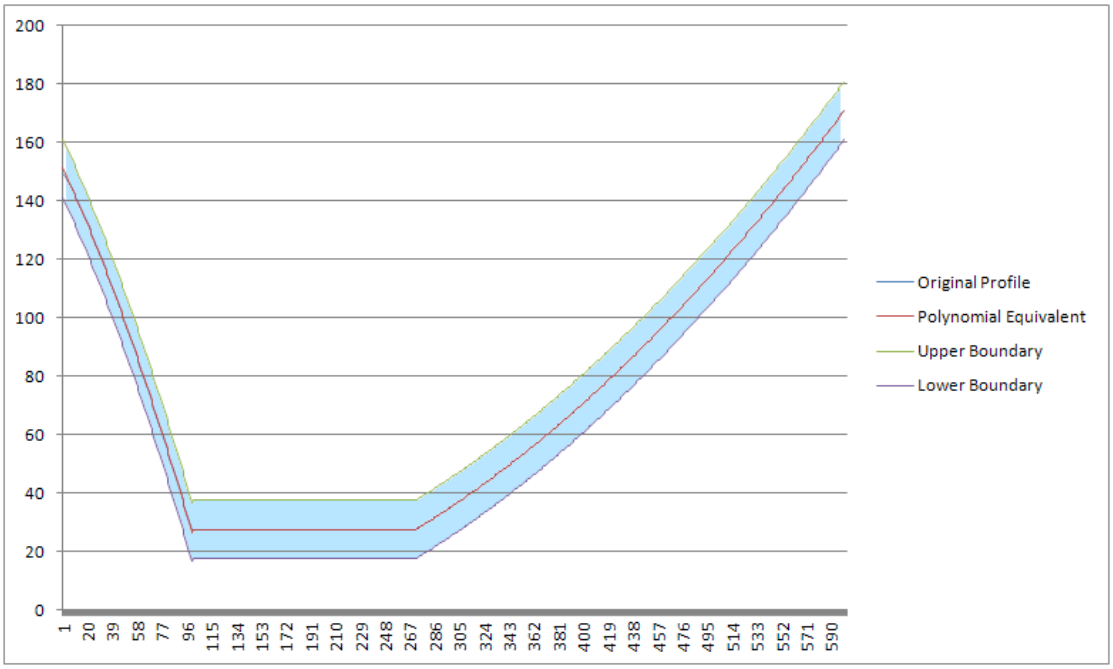

Figure 6 - Temperature profile output

The profiles shown in Figure 6 were created using the program above. Notice that the original trace and the polynomial are virtually identical (the Original Profile trace lies directly underneath the Polynomial Equivalent trace). This demonstrates how well polynomials can be used to replicate a series of points. The shaded area represents the values which lie within the bounded region.

5 Another example: Using polynomial boundaries

Sometimes boundary conditions may not be as basic as a percentage or fixed variance. In this case it may be necessary to use polynomials as the boundaries. The method is the same, but the program looks a little different. This example uses two polynomial boundaries which have been derived using the same methods above.

```
Y1=3.21350969E1,4.99543761E-2,-1.59111120E-3,3.88481564E-5,-
5.25900172E-7,2.83660353E-9 'UPPER

Y2=2.97572717E1,6.14184163E-2,7.93111882E-4,-7.86249890E-
5,2.02403753E-6,-2.33944000E-8,1.02547894E-10 'LOWER
```

The polynomials and acceptable range can be seen in Figure 7. Note once more that the polynomials fit the data almost perfectly..

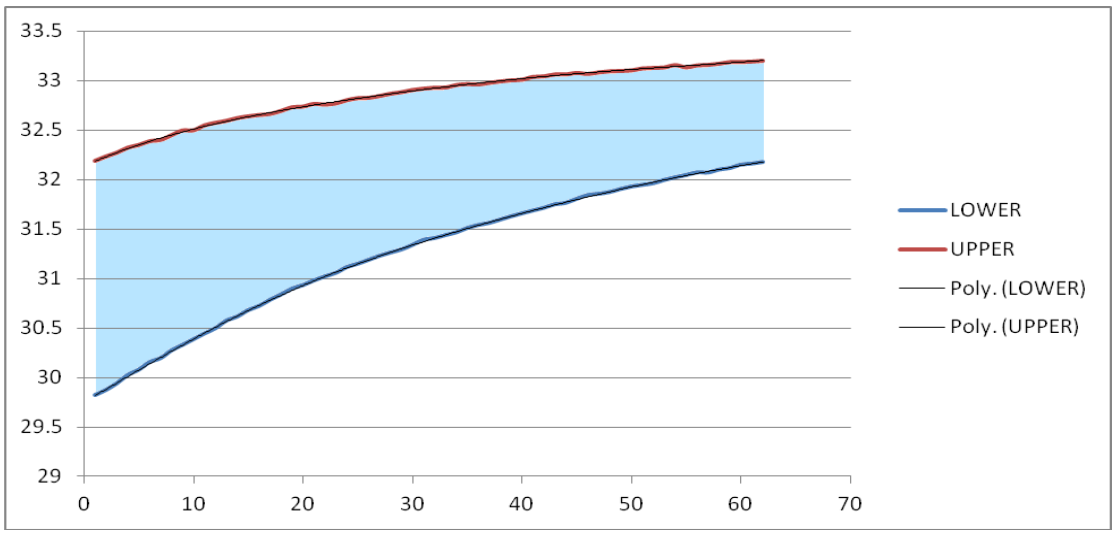

Figure 7 - Polynomial boundaries (shaded area = acceptable values)

5.1 The *dataTaker* program code

```
BEGIN"PROFILE2"
'=====================================================
'Initialise variables
LOGON
'set up Polynomial for the boundaries
Y1=3.21350969E1,4.99543761E-2,-1.59111120E-3,3.88481564E-5,-
5.25900172E-7,2.83660353E-9 'UPPER

Y2=2.97572717E1,6.14184163E-2,7.93111882E-4,-7.86249890E-
5,2.02403753E-6,-2.33944000E-8,1.02547894E-10 'LOWER

1CV(W)=0  'reset the test timer
'=====================================================

'=====================================================
' Schedule A (MAIN PROFILE SCHEDULE)
'   - Runs every 1 second (matches the x axis scale on the profile)
'=====================================================
RA("B:",ALARMS:OV:10KB,DATA:OV:1MB)1S LOGONA
      1DSO(W)=1 'test in progress signal (or turn on equipment).
      '1CV = sum of the time since the program started
      T("PREVIOUS PERIOD",DT,+=1CV)
      1CV("TOTAL TIME")
      1CV("UPPER",Y1,=2CV) 'store corresponding Y1 value into 2CV
      1CV("LOWER",Y2,=3CV) 'store corresponding Y2 value into 3CV
      'take the measurement (in this case a voltage), save into 4CV
      1V("MEASUREMENT~mV",=4CV)
      'if the value is outside the boundaries, turn on warning light
      IF(4CV<>3CV,2CV)1WARN
END
```


©2012 Thermo Fisher Scientific Australia Pty Ltd. All rights reserved. A.B.N. 52 058 390 917