Unplotting Tools

Manual Interpolation from Printed Plots

This worksheet introduces a component to have a tool inside Mathcad to manually extract points from a scan of a printed plot.

A lot of times engineers must to extract data from printed diagrams or similar sources; each time that this happen calculations must to stop and wait for the interpolated data: simulation and modeling process can't be done well with a runtime travel to the diagram.

With Mathcad is possible automatic reading information from a picture and write a program that extract the curves (one or more): the tools are the smoothing techniques.

But this automation procedure works fine under some restrictions:

- The reader have a good background on image processing.
- The image is "good".

For example, the selected diagram isn't very good in the sense of image processing and requires a hard setup to sucess; it is the variation of the thermal conductivity of some solids with temperature, taked from Çengel, Yunus A. [2003], *Heat transfer: a practical approach*, McGraw-Hill, online **here**. Wanted: the *Iron curve*.

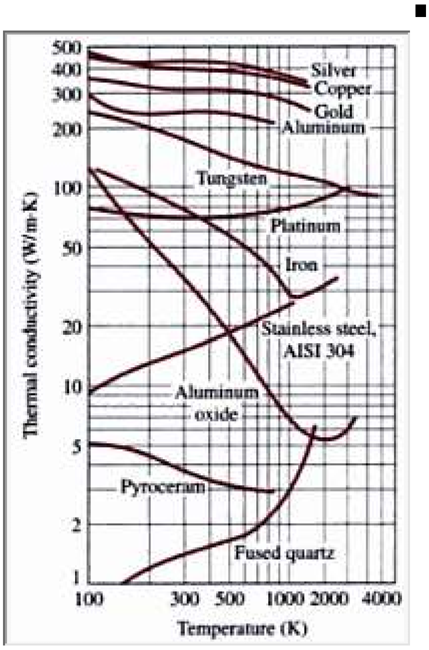
Procedure

Enable the following component, and to manually extract the data do

- Click to select a point
- Alt+Click to delete the last entry
- Ctrl+Click to set a first known value
- Shift+Click to set a second known value

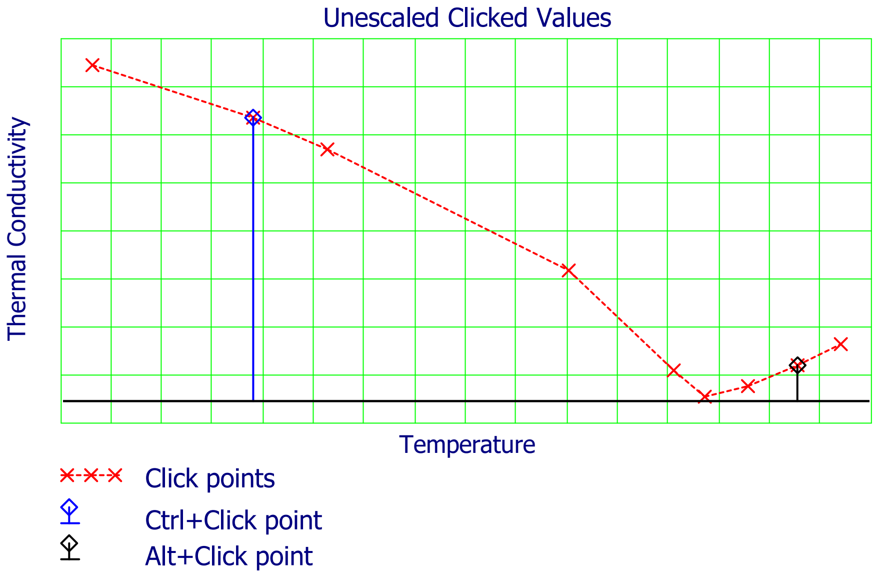
–

R :=



Selected points can be viewed in the following table and plot

$$R^T = \begin{pmatrix} 28 & 116 & 2 & 40 & 79 & 96 & 101 & 108 & 123 \\ 54 & 7 & 64 & 48 & 25 & 6 & 1 & 3 & 11 \end{pmatrix}$$



Enter the actual values for the two points showed as steams

Point to set by Ctrl+Click

$$A := \begin{pmatrix} 200 \\ 91.5 \end{pmatrix}$$

Point to set by Shift+Click

$$B := \begin{pmatrix} 2000 \\ 30.6 \end{pmatrix}$$

Following is for scale to the logarithmic axis leaving the ability to a minor modification if the axis are not scaled (setting $\varphi(x) = x$)

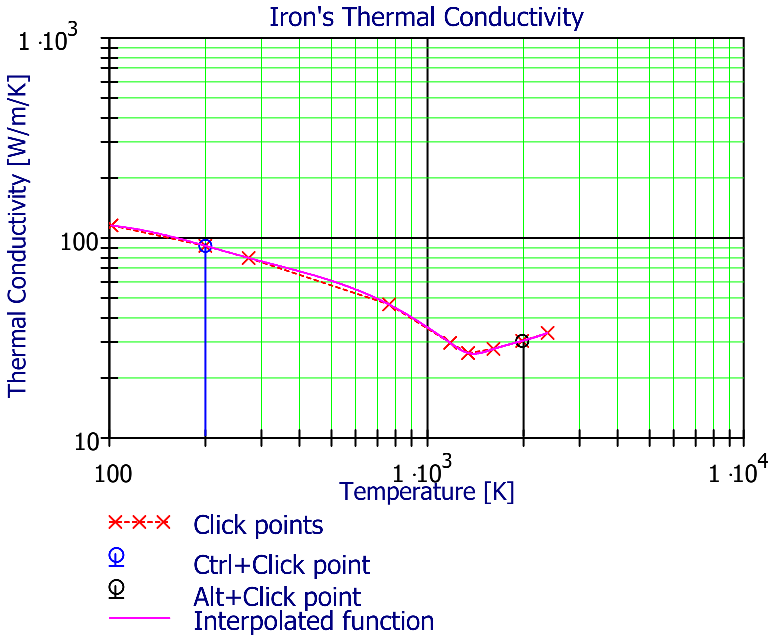
$$\varphi(x) := \frac{\ln(x)}{\ln(10)} \qquad \vartheta(y) := y = \varphi(x) \text{ solve}, x \ \rightarrow \exp(y \cdot \ln(10))$$

With this information we can scale, order and interpolate the clicked points

```
$(X\ \ Y) :=$ | $s \leftarrow \text{stack}\left(\frac{\varphi(B_1) - \varphi(A_1)}{R_{2,1} - R_{1,1}}, \frac{\varphi(B_2) - \varphi(A_2)}{R_{2,2} - R_{1,2}}\right)$
              | for $k \in 1 .. \text{rows}(R)$
              |     | $M_{k,1} \leftarrow s_1 \cdot (R_{k,1} - R_{1,1}) + \varphi(A_1)$
              |     | $M_{k,2} \leftarrow s_2 \cdot (R_{k,2} - R_{1,2}) + \varphi(A_2)$
              | $\left(\text{csort}(M,1)^{\langle 1 \rangle}\ \ \text{csort}(M,1)^{\langle 2 \rangle}\right)$
```

$$vs := \text{cspline}(X, Y) \qquad f(x) := \vartheta(\text{interp}(vs, X, Y, \varphi(x)))$$

Resulting



Finally copy and paste the clicked values here. Why? Because if not you **lost** collected points when closing this worksheet (remember that this is a manual procedure)

$$R \equiv \begin{pmatrix} 28 & 116 & 2 & 40 & 79 & 96 & 101 & 108 & 123 \\ 54 & 7 & 64 & 48 & 25 & 6 & 1 & 3 & 11 \end{pmatrix}^{T}$$

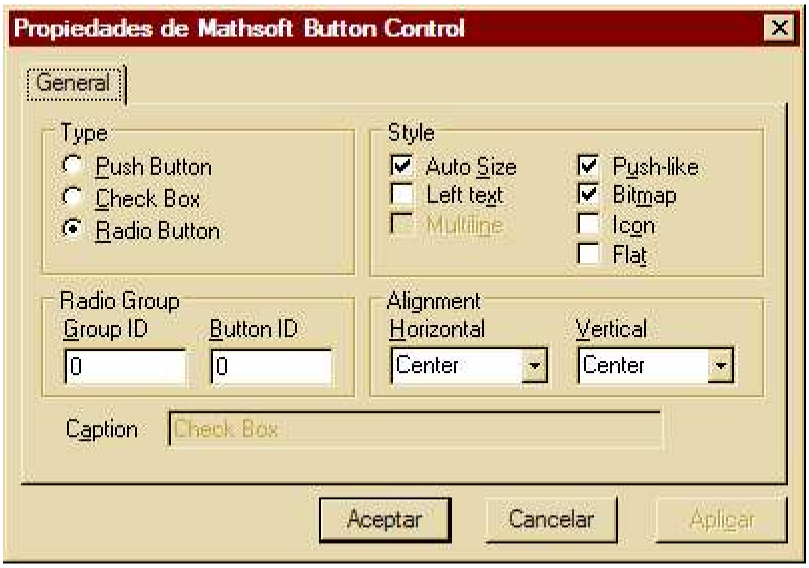
One disadvantage of this method is that not provide good learning for traceability, for this reason preserving original clicked values is important.

How to make the component

Insert a Radio Button component, like the following

R := ○ Radio Button

Select their properties as shown the figure



Propiedades de Mathsoft Button Control

General

Type
○ Push Button
○ Check Box
● Radio Button

Style
☑ Auto Size
☐ Left text
☐ Multiline
☑ Push-like
☑ Bitmap
☐ Icon
☐ Flat

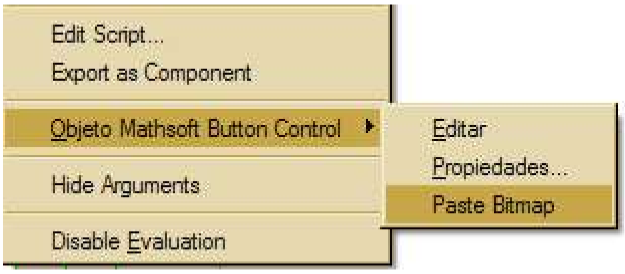
Radio Group
Group ID: 0
Button ID: 0

Alignment
Horizontal: Center
Vertical: Center

Caption: Check Box

Aceptar | Cancelar | Aplicar

To work with your own graph and with it in the clipboard, right click over the component and select "Paste Bitmap"



Change the default name for the component (RadioBtn) by xyPlot and paste the following code

```
Dim x(), y()      ' Click Values
Dim c : c = 0     ' Counter
Dim a, b, sel     ' MouseMove event
Dim xmin, ymax    ' Screen to axis coords conversion

Sub xyPlotEvent_Exec(Inputs,Outputs)
   On error resume Next ' Preventing hangs
   Select Case sel
   Case 0 ' Add point
      c = c + 1
      If c < 1 Then c = 1
      ReDim Preserve x(c), y(c)
      x(c) = a : y(c) = b
   Case 4 ' Alt = delete last point
      If c > 1 Then
         c = c - 1
         ReDim Preserve x(c), y(c)
      End If
   Case 2 ' Ctrl = 1st point
      x(0) = a : y(0) = b
   Case 1 ' Shift = 2nd point
      x(1) = a : y(1) = b
   End Select
   xmin = x(0) : ymax = y(0) ' Find xmin, ymax
   For k = 0 to c
      If xmin > x(k) Then xmin = x(k)
      If ymax < y(k) Then ymax = y(k)
   Next
   For k = 0 To c ' output
      Outputs(0).Value(k,0) = x(k) - xmin
      Outputs(0).Value(k,1) = ymax - y(k)
   Next
End Sub

Sub xyPlot_Click()
   xyPlot.Recalculate()
End Sub

Private Sub xyPlot_MouseMove(Button, Shift, xc, yc)
   a = xc + 1 : b = yc + 1 : sel = Shift
End Sub
```

Further developments

First ideas are:

- The interpolation probably needs a stage to prevent duplicated x-vales, but not at the component level for not excluding plane curves.
- The component's code can be modified to include the scaling, parsing as parameters the points A and B.
- Interpolation by splines can work with few click points, but for a better scan, with a lot of click points, the same Mathcad's smoothing techniques that automates the curve recognition can be applied here also but to the clicked points.
- Could be a good idea search an analytic expression.
- For iso-level curves (or for the exposed example) the component can be modified to handle two variables functions, knowing which curve the user is clicking at any time.