# Unit Graph Formatting V0

Erdem Canaz

2020, December 30

## Contents

# 1 Introduction

Some researches are not expected to result in practical use. Thus they may not deal with non-ideal components. However, this project is directly linked to real-life components such as water pumps, photovoltaic panels, or inverters. As a consequence, the datasheets have a crucial role in modeling the system. They will allow us to compare different system responses and go for the most optimal one. Thus simplifying the algorithm developing process between the project contributors and using the datasheets' characteristics is very important. In this White paper, I will suggest a method of using the datasheets in our algorithms. I will also share the software I coded. It extracts the graph data from datasheets in the described format. Each contributor different from the author can easily understand and troubleshoot it rather than dealing with the variable names.

# 2 Unit Graph formatting version 0

## 2.1 Squeezing the characteristic graphs to 1-1 graph


Figure 1: Pump 1

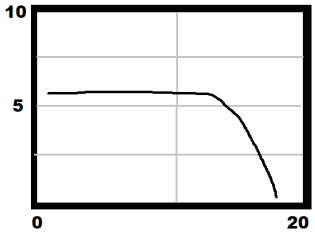


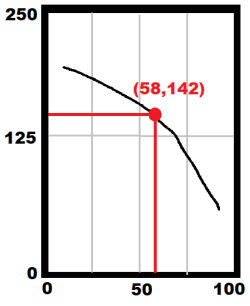

Figure 2: Pump 2

Let's say we have two different pumps. Their characteristics are shown in the figures 2.1 and 2. The vertical axis corresponds to pressure, whereas the horizontal one corresponds to the flow. If those pumps are the ones we may buy for the project, we have to upload their characteristics to the program. Then our algorithm can know those characteristic graphs and it can analyze which one suits better for the given case. Uploading data requires methodology. We have to create a data format. The aim of the **Unit Graph Formatting V0** is to squeeze the characteristic graphs to 1-1 plot. And we can achieve it by using transformation.
Observe that for both graphs, fundumental base is

$$B_f = \{\begin{bmatrix}1\\0\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}\}$$

For figure 2.1 we may pick a base such as;

$$B_1 = \{\begin{bmatrix}100\\0\end{bmatrix}, \begin{bmatrix}0\\250\end{bmatrix}\}$$




$$\overrightarrow{V_{real}} = \begin{bmatrix}58\\142\end{bmatrix}$$

$$\left[\overrightarrow{V_{formatted}}\right]_{B_1} = \left[\left[\overrightarrow{B_{f_1}}\right]_{B_1} | \left[\overrightarrow{B_{f_2}}\right]_{B_1}\right] \begin{bmatrix}58\\142\end{bmatrix}$$

$$\begin{bmatrix}\frac{58}{100}\\ \\ \frac{142}{250}\end{bmatrix} = \begin{bmatrix}\frac{1}{100} & 0\\ 0 & \frac{1}{250}\end{bmatrix} \begin{bmatrix}58\\142\end{bmatrix}$$

We have formatted the data $(58, 142)$ to $(0.58, 0.568)$. Note that we can guarantee uniqueness. For a particular point in figure 2.1, a particular point exists in the transformed data. To get the original data, we may perform:

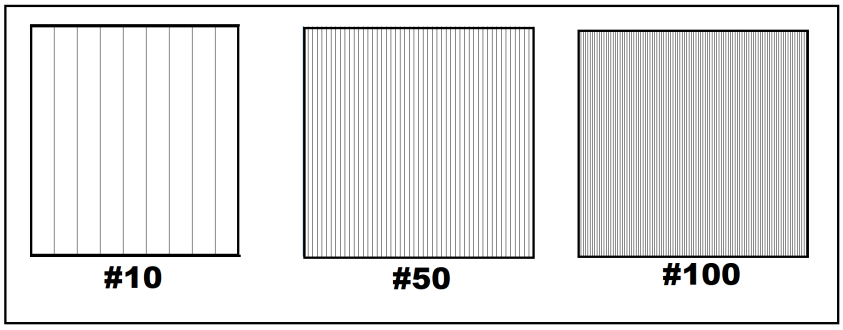
$$\begin{bmatrix} 58 \\ 142 \end{bmatrix} = \begin{bmatrix} \frac{1}{100} & 0 \\ 0 & \frac{1}{250} \end{bmatrix}^{(-1)} \begin{bmatrix} 0.58 \\ 0.568 \end{bmatrix}$$

We can perform the exact approach on figure 2 by chosing $B_2$ as;

$$B_2 = \{ \begin{bmatrix} 20 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 10 \end{bmatrix} \}$$

## 2.2 Applying transformation on an example

In section 2.1, we have analyzed only one particular point. Obviously, it is not enough to express the character of the component at different operating points. But, how many nodes is sufficient?




In this formatting, the 100 data point is determined to be enough. No matter what the graph looks like, we will have equally spaced (horizontal axis) 100 data points. We will apply transformation described in section 2.1 on them.

If we want to store both horizontal and vertical axis data, we must store 200 values. However, it is not necessary to store the horizontal axis. If we provide the horizontal axis's maximum value, our algorithm can match any of those 100 vertical data with the corresponding horizontal value. Let's analyze a real example.

$$\eta = \begin{bmatrix} 0 & 0.08 & 0.11 & ... & 0.78 & 0.74 & 0.73 \end{bmatrix}$$

The matrix above is the data extracted from figure 3 for the version 90G/75. It stores efficiency [vertical] & flow[horizontal] data. The maximum efficiency is 0.59, and the maximum flow is 300L/s. The following example illustrates how

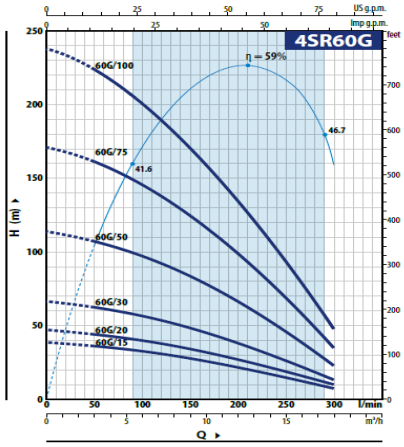



Figure 3: Water pump datasheet

it can be used practically in programming:

$$\eta\left(1+(round)\left[\frac{0L}{s}\cdot\frac{s}{300L}\cdot 99\right]\right)=(Corresponding\,entry)\cdot(Maximum\,efficiency)$$

when the flows are 0L/s and 293L/s the efficiencies are:

$$\eta\left(1+(round)\left[\frac{0L}{s}\cdot\frac{s}{300L}\cdot 99\right]\right)=\eta\,(1)=(0)\cdot(0.59)=0$$

$$\eta\left(1+(round)\left[\frac{293L}{s}\cdot\frac{s}{300L}\cdot 99\right]\right)=\eta\,(99)=(0.74)\cdot(0.59)=0.44$$

It was a rough demonstration. One should consider some details while using it.

- MATLAB indexes starts with 1, not with zero
- the input horizantal value should satisfty $0 \leq input \leq Maximum$
- **If there is no vertical data is given in the datasheet for a specific value (horizontal) , *'E'* is assigned to that transformed entry (vertical).**
- **101. term is reserved for horizontal maximum whereas 102. term is reserved for vertical maximum**
- **All the values are positive**
- **...**

In the upcoming section, a function written in MATLAB that handles this process carefully, safely and correctly will be given.

## 2.3 MATLAB function that calculates y value for a specific x using formatted data

**getVerticalValue_UGF0(DATA, inputhorizontalvalue)** is designed to handle this formatting.

```
60G100WaterPump_5500W_3_flow_efficiency_SI_UGFV0 = [0
    0.08532439 0.11947282 0.13654703 0.17313462 0.20972224
    0.23167479 0.25362736 0.29021496 0.31948504 0.34631592
    0.37314683 0.39753857 0.40973443 0.43656534 0.46339625
    0.48047045 0.49754468 0.52193636 0.53169304 0.5536456
    0.5780374 0.5926724 0.614625 0.6341383 0.65609086
    0.66828674 0.69511765 0.7097527 0.72194856 0.7365836
    0.7487795 0.75853616 0.77292734 0.7851232 0.7924407
    0.80707574 0.81683207 0.824149 0.8387829 0.8509779
    0.86073387 0.86805075 0.88024575 0.8900017 0.9021966
    0.91195256 0.91926956 0.92414755 0.9290255 0.93390346
    0.9412204 0.9485374 0.9558543 0.9631713 0.97048825
    0.97780526 0.98024416 0.9826832 0.98756117 0.9900001
    0.99243915 0.99731714 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
    0.99731714 0.99731714 0.9948781 0.99243915 0.98756117
    0.98512214 0.9826832 0.97780526 0.9729273 0.9680493
    0.9631713 0.9607323 0.9509763 0.9436594 0.93878144
    0.9290255 0.92170846 0.9095136 0.8948797 0.8851237
    0.86805075 0.8509779 0.836344 0.81683207 0.8021974
    0.782684 0.7487795 0.7341444 300 59 ];

FLOW = 5;
efficiency_at_FLOW=getVerticalValue_UGF0( 60
    G100WaterPump_5500W_3_flow_efficiency_SI_UGFV0, FLOW )

function f1 = getVerticalValue_UGF0(DATA,
    input_horizontal_value)
    %% 102 entry, 101-> horizontal max, 102-> vertical max, 'E
        ' if no vertical data exists

    %% ERRORS
    if(input_horizontal_value<0) %1
        error("FUNCTION: getVerticalValue_UGF0(..) ERROR1:
            input horizontal value  is negative");
    end
    if(length(DATA)~= 102)%2
        error("FUNCTION: getVerticalValue_UGF0(..) ERROR2:
            number of elements is not 102");
    end
    if(DATA(101) == 0 || DATA(102)==0 ) %3
        error("FUNCTION: getVerticalValue_UGF0(..) ERROR3:
            Maximum value(s) of the data is zero");
    end
```

```
19      if(DATA(101) < 0 || DATA(102)< 0 ) %4
20          error("FUNCTION: getVerticalValue_UGF0(..) ERROR4:
                Maximum value(s) of the data is negative" );
21      end
22      if(DATA(101)< input_horizontal_value) %5
23          error("FUNCTION: getVerticalValue_UGF0(..) ERROR5:
                input horizontal value  is greater than the limit"
                );
24      end
25      for index = 1:1:100 %6
26       if(DATA(index) =='E')
27           continue
28       elseif(DATA(index)>1 || DATA(index)<0)
29           error("FUNCTION: getVerticalValue_UGF0(..) ERROR6:
                corrupted vertical value. 0<= val <= 1 is not
                satisfied" );
30       end
31      end
32
33      %%%%%%%
34
35      %% index_corresponds_to_input      [1,100], float
36      index_corresponds_to_input= 1 + 99*(input_horizontal_value
            /DATA(101));
37      %% lower_index      [1,100], integer
38      lower_index = floor(index_corresponds_to_input);
39      %% upper_index      [1,100], integer
40      upper_index = ceil(index_corresponds_to_input);
41
42      %% linear approximation
43      if( isnumeric(DATA(lower_index)) && isnumeric(DATA(
            upper_index)) )
44          dy = DATA(102)*( DATA(upper_index)-DATA(lower_index) )
                ;
45          dx = mod(index_corresponds_to_input,1);
46          f1 = DATA(102)*DATA(lower_index)+ dy*dx;
47      elseif(isnumeric(DATA(lower_index)) && ~isnumeric(DATA(
            upper_index)))
48          f1 = DATA(102)*DATA(lower_index);
49      elseif(~isnumeric(DATA(lower_index)) && isnumeric(DATA(
            upper_index)))
50          f1 = DATA(102)*DATA(upper_index);
51      elseif(~isnumeric(DATA(lower_index)) && ~isnumeric(DATA(
            upper_index)))
52          f1 = 'E';
53      end
54
55
56  end
```

## 2.4 Naming the variable

$$name,\ sub\_name,\ no,\ x\_name,\ y\_name,\ isSI,\ UGFV0$$

$$60G100WaterPump\_5500W\_3\_flow\_head\_\_SI\_UGFV0$$

- **name:** Model (and or or) Name of the component
- **sub_name:** Any handy information that describes the component
- **no:** Any
- **x_name:** Which value does horizontal axiss represents
- **y_name:** Which value does vertical axiss represents
- **isSI:** Whether the units of the axes are in SI
- **UGFV0:** Unit Graph Formatting V0

101. term is reserved for **x_maximum** whereas 102. term is reserved for **y_maximum**

$$x\_max,\ y\_max$$

$$0.005,\ 250$$

- **x_max:** Max value horizontal axiss can take
- **y_max:** Max value vertical axiss can take

# 3 Graph Analyzing Software V1

Outputs any graphical data in **UGFV0** for **MATLAB** use.

## 3.1 UI

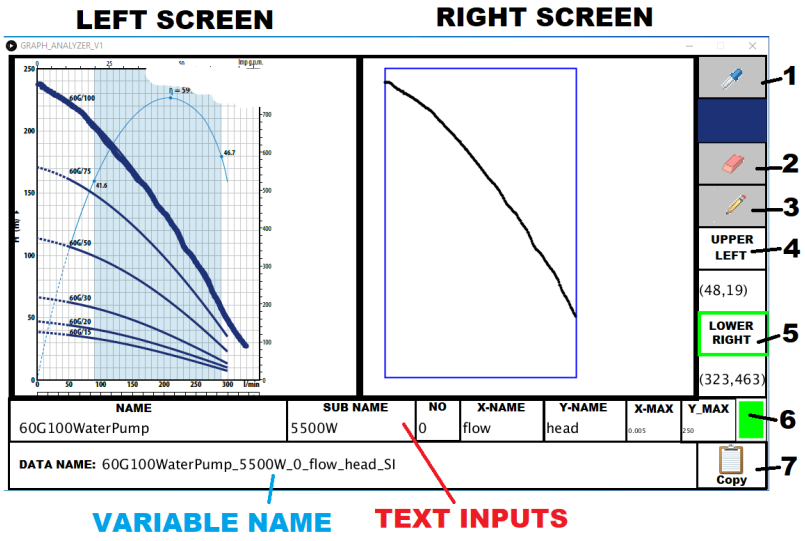



Figure 4: ScreenShot of the software

- **button 1:** color sampling
- **button 2:** eraser
- **button 3:** pencil
- **button 4:** click to the upper left point of the related data
- **button 5:** click to the lower right point of the related data
- **button 6:** if you entered X_MAX & Y_MAX in SI units, make it green, otherwise make it red.
- **button 7:** the formatted variable is copied to you clipboard. Open MATLAB and paste it directly
- **LEFT SCREEN:** Data to extract. You can use buttons [1-5] in this region only.
- **RIGHT SCREEN:** Virtual representation of the output data

- **TEXT INPUT:** You can type in any of them. Just move your cursor on them and start typing.
- **VARIABLE NAME:** the final variable name is shown here

## 3.2 Keyboard shortcuts

- **-:** effective diameter of the eraser and pencil is set to 3px
- **+:** effective diameter of the eraser and pencil is set to 20px
- **ğ:** refreshes the left screen

## 3.3 How to use?

`https://youtu.be/IZbgYxvjC-0`

## 3.4 Github

`https://github.com/erdemcanaz/COMSA-ERDEM`

## 3.5 Drive (As executable)

`https://drive.google.com/drive/folders/1o78OXFs0mPMpJdmVrg6vG0juIDCpY0vc?usp=sharing`

# 4 Conclusion

"Make agriculture great again!"

# References