

# Dynamic Graphs in Python and MatDeck

- Level: Medium

Given arrays `vec1` and dynamic `vec2`. The task is to write a Python program to visualize the result, by plotting line graphs using `vec1` as x axis and `vec2` as y axis.

## Examples:

```
Input1: vec1 = [1, 2, 3, 4, 5] vec2 = [10, 12, 9, 11, 13] with increment of 2 per iteration
```

```
Output: Dynamic 2D graph - line plot which is changing as vec2 changes
```

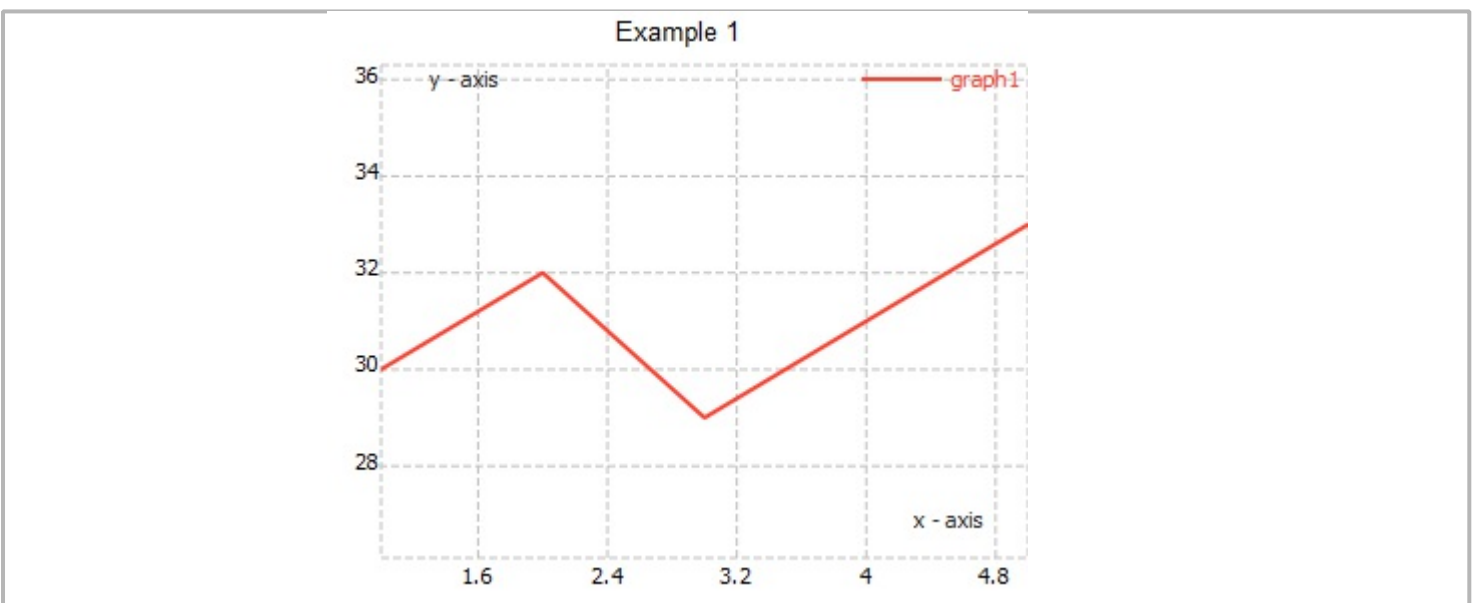
Two vectors are defined in MatDeck script:

```
1 vec1 := [1, 2, 3, 4, 5]
2 vec2 := [10, 12, 9, 11, 13]
3 graph1 := join_mat_rows(vec1, vec2)
```

## MatDeck Script Code

Here, we illustrate dynamic graph in MatDeck Script. The use of 2D graph is supported by the most MatDeck licenses, there is no need for additional installation of any package. Further, the code is rather simple and graph can be embedded within MatDeck document. The main advantage of MatDeck graph compared to Python graph, is in the fact that the graph properties are easy set using directly graph GUI, while Python graph is configured by using script code. We change variable `vec2`, by incrementing it by two in every iteration.

```
4 for(i := 0; i < 10; i += 1)
5 {
6   vec2 = vec2 + 2
7   graph1 = join_mat_rows(vec1, vec2)
8 }
9
```



# Python Code

In the program below, the Python library matplotlib is used. Therefore, it is necessary to install matplotlib, which can be done using the following command `python -m pip install -U matplotlib`. Matplotlib is available for Windows.

In Python we change variable in for loop, and we plot graph .

```
10 #py
11 import matplotlib.pyplot as plt
12 import time
13 vec1=[1, 2, 3, 4, 5]
14 vec2py=[10, 12, 9, 11, 13]
15 plt.show()
16 axes = plt.gca()
17 axes.set_xlim(0, 6)
18 axes.set_ylim(5, 50)
19 plt.xlabel('x - axis')
20 plt.ylabel('y - axis')
21 plt.title('Example 1')
22 plt.grid()
23 line, = axes.plot(vec1,vec2py,color='red',lw=1)
24
25
26 for x in range(0,10):
27     vec2py = [x + 2 for x in vec2py]
28     line.set_ydata(vec2py)
29     plt.draw()
30     plt.pause(1e-17)
31     time.sleep(0.5)
32 plt.show()
33 ###
34
```