Jupyter notebook custom conversion

Romain Madar

August 2018

Overview

nbconvert latex test

Printing using python

Pyout (and Text Wrapping)

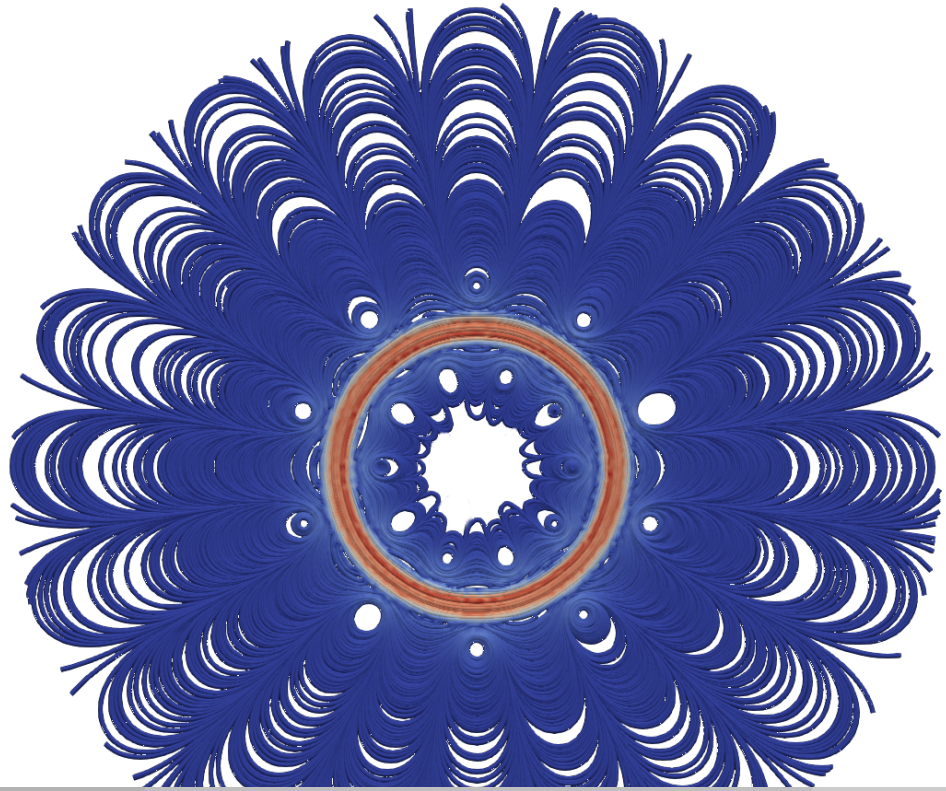
Image and plots

As plain text using markdown

Once exported as markdown and converted to latex/pdf with pandoc, the `{width=60%}` will fix the width of the picture and the `My legend` will appear as caption:

```
![My legend](figures/magnetostatics_field.png){width=50% #figlabel}
```

gives the result showns in this figure.



Plots produced by the code I

```
import numpy as np
x = np.linspace(-10,10,300)
y = np.sin(x)
plt.figure(figsize=(4,3),dpi=100)
p=plt.plot(x,y)
```

Plots produced by the code II

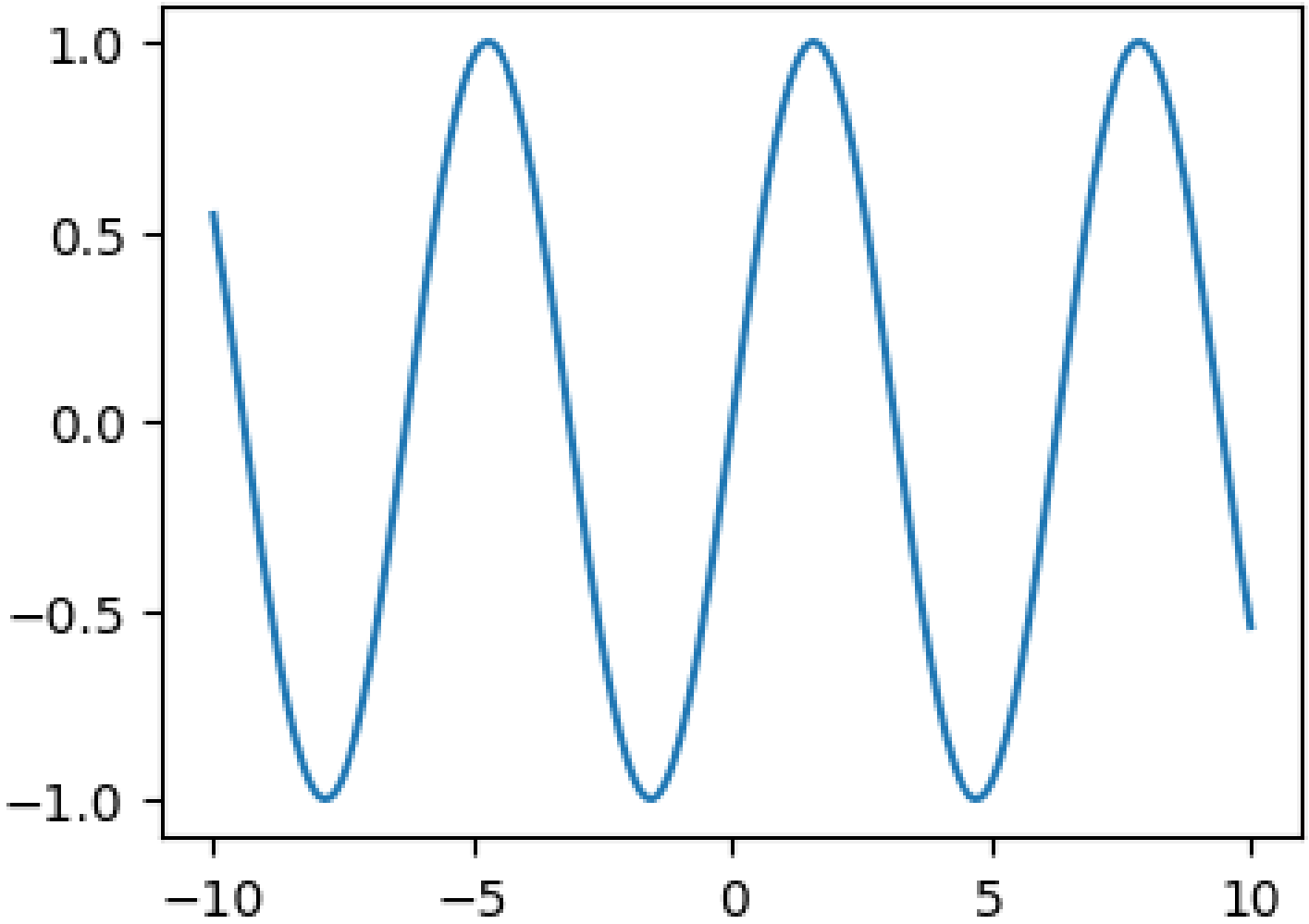
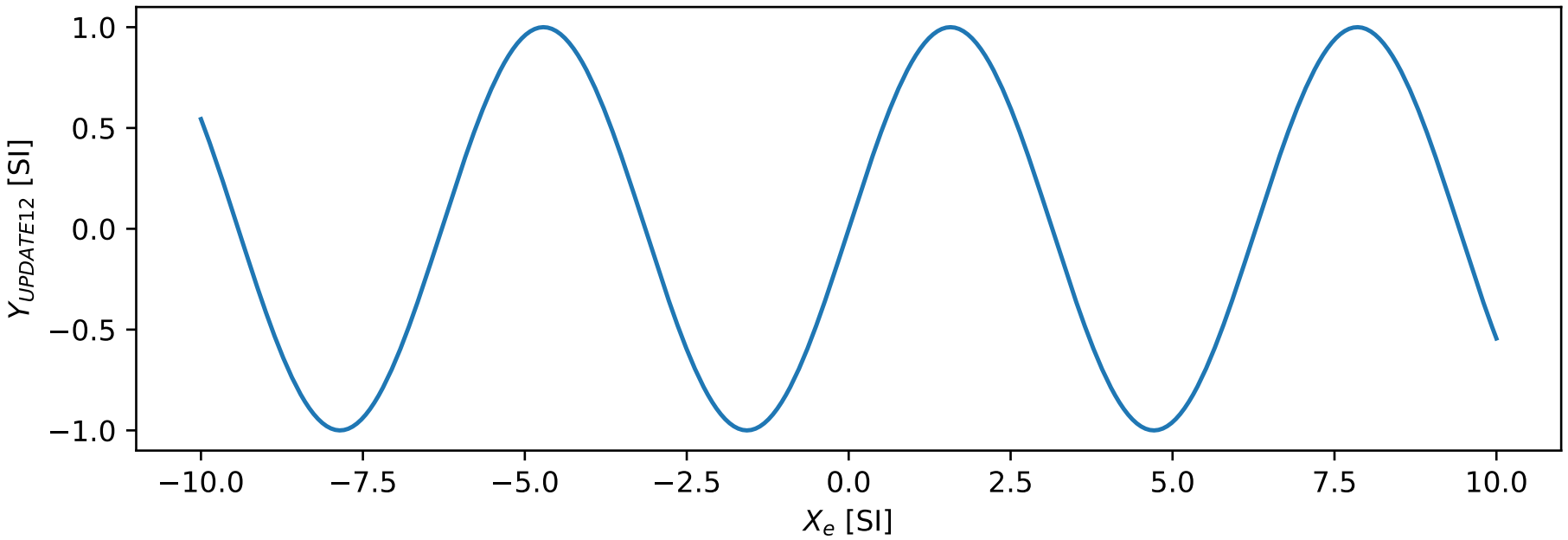


Figure 3: png

Plots produced by the code III

```
plt.figure(figsize=(8,3))
plt.plot(x,y)
plt.xlabel('$X_{e}$ [SI]')
plt.ylabel('$Y_{UPDATE12}$ [SI]')
plt.tight_layout()
jpu.plt2md('myplot','This is a test of how to get proper'+\
       'plot from Jupyter notebook in MD,'+\
       'to be processed using PANDOC','100%','myplot_LABEL')
```



Plots produced by the code IV

Figure 4: This is a test of how to get properplot from Jupyter notebook in MD,to be processed using PANDOC

We can then refer to a given figure using cross-references like this, obtained with:

```
[like this](#myplot)
```

```
plt.figure(figsize=(3,3))
plt.plot(np.sqrt(np.abs(x)),y**2)
plt.xlabel('$X_{e}$ [SI]')
plt.ylabel('$Y_{OTHER}$ [SI]')
plt.tight_layout()
jpu.plt2md('myplot2','This is another test of how to get proper'+\
       'plot from Jupyter notebook in MD,'+\
       'to be processed using PANDOC','50%','myplot2_LABEL')
```

Plots produced by the code V

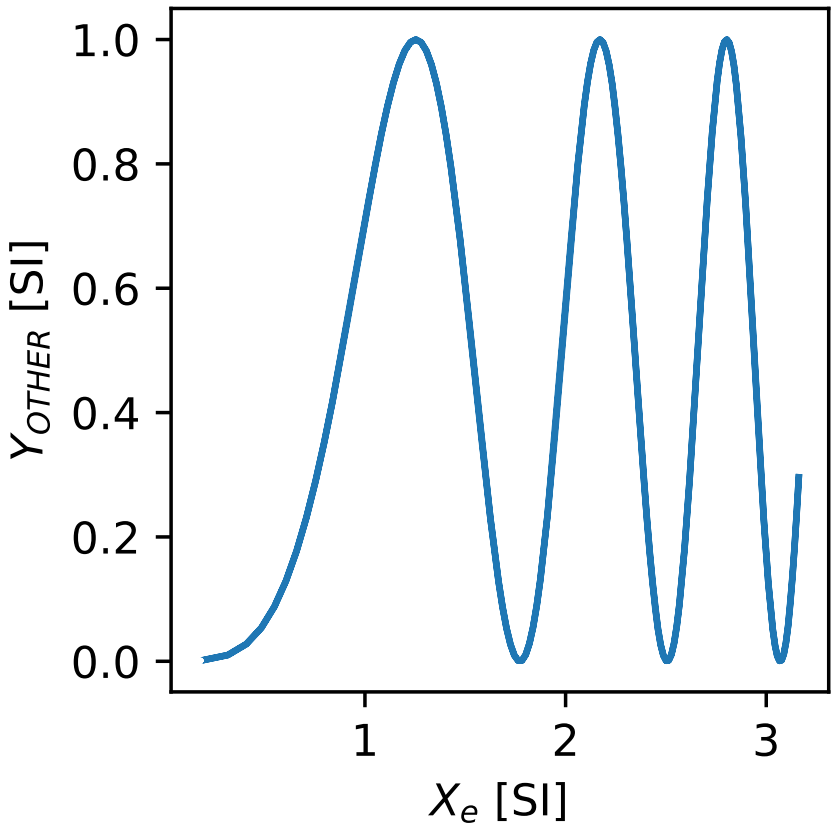
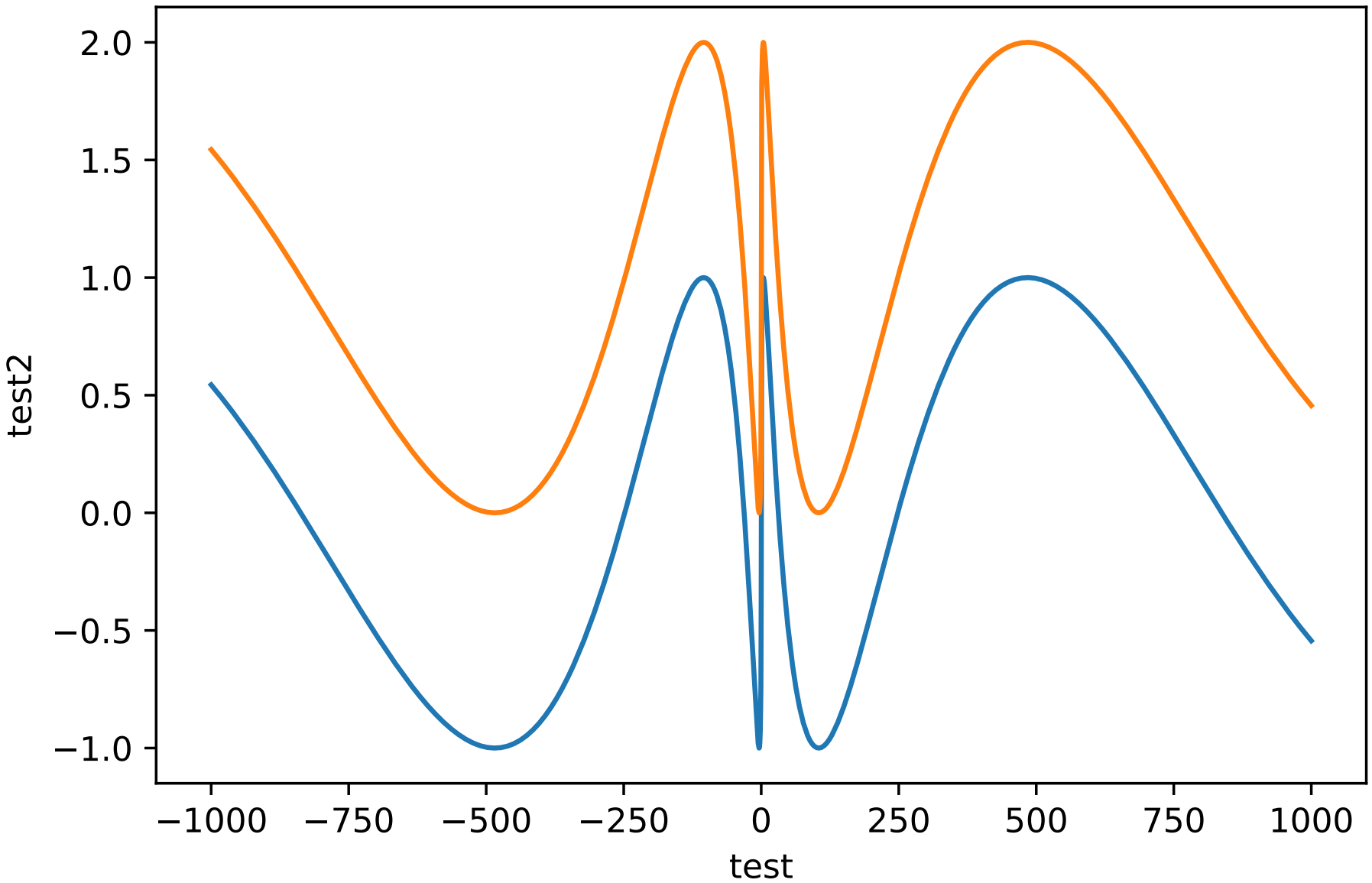


Figure 5: This is another test of how to get properplot from Jupyter notebook in MD,to be processed using PANDOC

Plots produced by the code VI

```
plt.figure(figsize=(6,4))
plt.plot(x**3,y)
plt.plot(x**3,y+1)
plt.xlabel('test')
plt.ylabel('test2')
plt.tight_layout()
jpu.plt2md('myplot3','Adding more plot without re-creating a figure: curves
↪  cumulates')
```

Plots produced by the code VII



Operator Highlighing Check

Tables

Markdown as plain text

First a *markdown* table:

Table 1: my caption

Column 1	Column 2
1	3
~~a~~	b
4	&

Pandas as default and Markdown I

```
import pandas as pd
df=pd.DataFrame(np.random.randn(10,3))
```

Default printing is HTML, so it looks good on the web but it is not well rendered in pdf via ipynb->MB->pdf (using nbconvert and pandoc). A special function `df2md(df)` is included in the `jupy_pandoc_utils` package to write out a markdown format. This allows to set caption and even to refer to the table in the main document like this (using vanilla pandoc) or cited like table Table 2 (using pandoc-crossref filter, but hyperlink doesn't seem to work in HTML though)

```
# Good in HTML, but not pure markdown
# Impossible to put a caption
df.describe()
```

0

1

2

count

Pandas as default and Markdown II

10.000000

10.000000

10.000000

mean

-0.124437

-0.193223

0.086777

std

0.865165

0.856129

0.811450

min

Pandas as default and Markdown III

-1.667288

-1.287436

-1.705184

25%

-0.570475

-0.762087

-0.242516

50%

-0.040271

-0.230659

0.453195

75%

Pandas as default and Markdown IV

0.438126

0.007493

0.646274

max

1.100231

1.791935

0.828771

```
# Good in pure MD and possible to put a caption and a label
jpu.df2md(df.describe(),'Caption table','#tbl:label2')
```

Pandas as default and Markdown V

Table 2: Caption table

labels	0	1	2
count	10	10	10
mean	-0.124437	-0.193223	0.0867771
std	0.865165	0.856129	0.81145
min	-1.66729	-1.28744	-1.70518
25%	-0.570475	-0.762087	-0.242516
50%	-0.0402713	-0.230659	0.453195
75%	0.438126	0.00749318	0.646274
max	1.10023	1.79194	0.828771

Pandas as default and Markdown VI

One might want to display table without showing the code which leads to it, especially in a proper documentation. The following block will display the table in the final document but not the line

```
jpu.df2md(df.describe(),'Caption table with hidden source code this time.
↪  {#tbl:label3}')
```

which produces it, as shown in Table 3

Table 3: Caption table with hidden source code this time

labels	0	1	2
count	10	10	10
mean	-0.124437	-0.193223	0.0867771
std	0.865165	0.856129	0.81145
min	-1.66729	-1.28744	-1.70518
25%	-0.570475	-0.762087	-0.242516
50%	-0.0402713	-0.230659	0.453195

Pandas as default and Markdown VII

75%	0.438126	0.00749318	0.646274
max	1.10023	1.79194	0.828771

Sympy output

Cell tags

No tag at all

```
l='1 3 5 7 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75
↪  78 81 84 87 90 93 96 99 103'
print(l.split(' '))
l
```

[‘1’, ‘3’, ‘5’, ‘7’, ‘9’, ‘12’, ‘15’, ‘18’, ‘21’, ‘24’, ‘27’, ‘30’, ‘33’, ‘36’, ‘39’, ‘42’, ‘45’, ‘48’, ‘51’, ‘54’, ‘57’, ‘60’, ‘63’, ‘66’, ‘69’, ‘72’, ‘75’, ‘78’, ‘81’, ‘84’, ‘87’, ‘90’, ‘93’, ‘96’, ‘99’, ‘103’]

‘1 3 5 7 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75 78 81 84 87 90 93 96 99 103’

Tag `hide`

Tag `hide_input`

['1', '3', '5', '7', '9', '12', '15', '18', '21', '24', '27', '30', '33', '36', '39', '42', '45', '48', '51', '54', '57', '60', '63', '66', '69', '72', '75', '78', '81', '84', '87', '90', '93', '96', '99', '103']

'1 3 5 7 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75 78 81 84 87 90 93 96 99 103'

Tag `hide_ouput`

```
l='1 3 5 7 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75
↪  78 81 84 87 90 93 96 99 103'
print(l.split(' '))
l
```