Exponential and logarithm functions

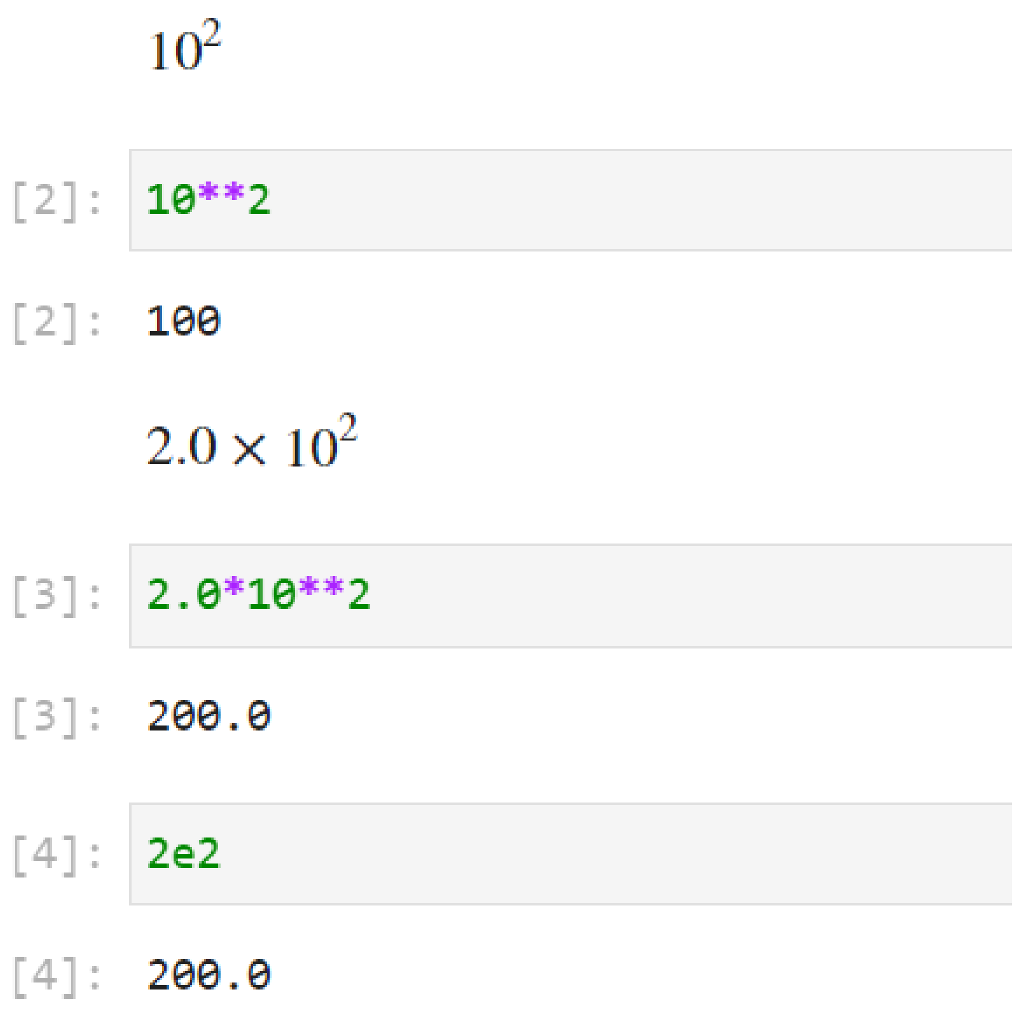
Software 2 – Mathematics Python Labs

Sakari Lukkarinen

Metropolia University of Applied Sciences

Contents

Power function and floating point numbers



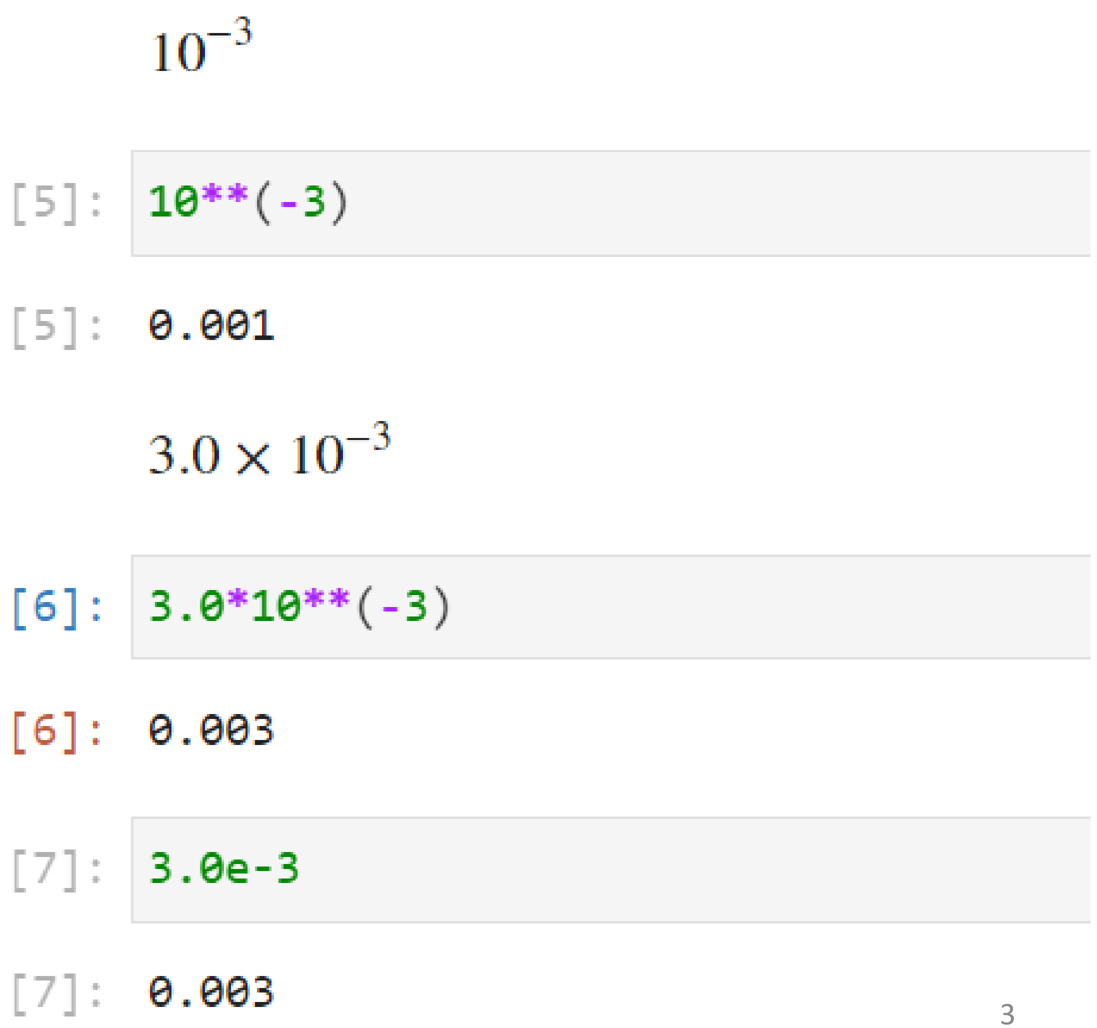
10^2

```
[2]: 10**2
[2]: 100
```

2.0×10^2

```
[3]: 2.0*10**2
[3]: 200.0
```

```
[4]: 2e2
[4]: 200.0
```



10^{-3}

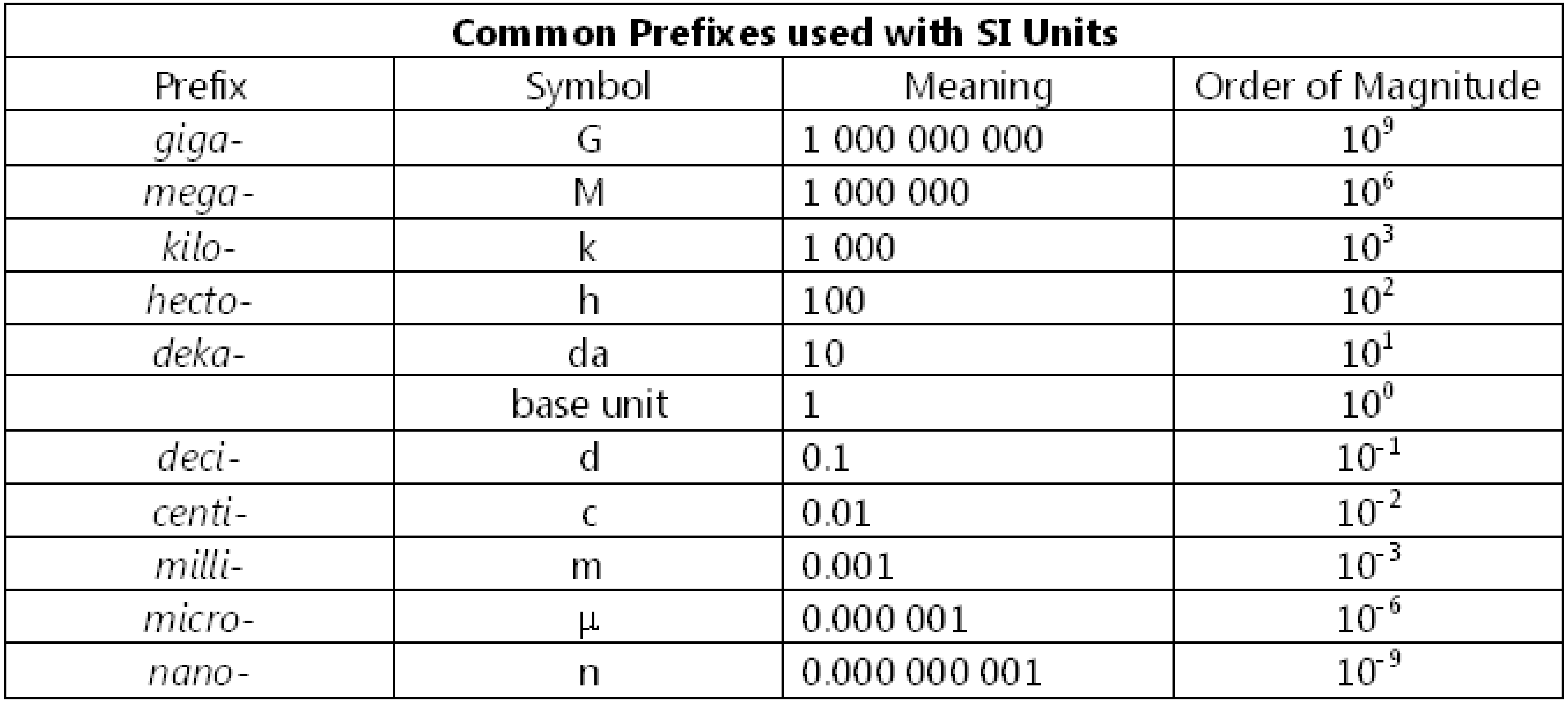
```
[5]: 10**(-3)
[5]: 0.001
```

3.0×10^{-3}

```
[6]: 3.0*10**(-3)
[6]: 0.003
```

```
[7]: 3.0e-3
[7]: 0.003
```

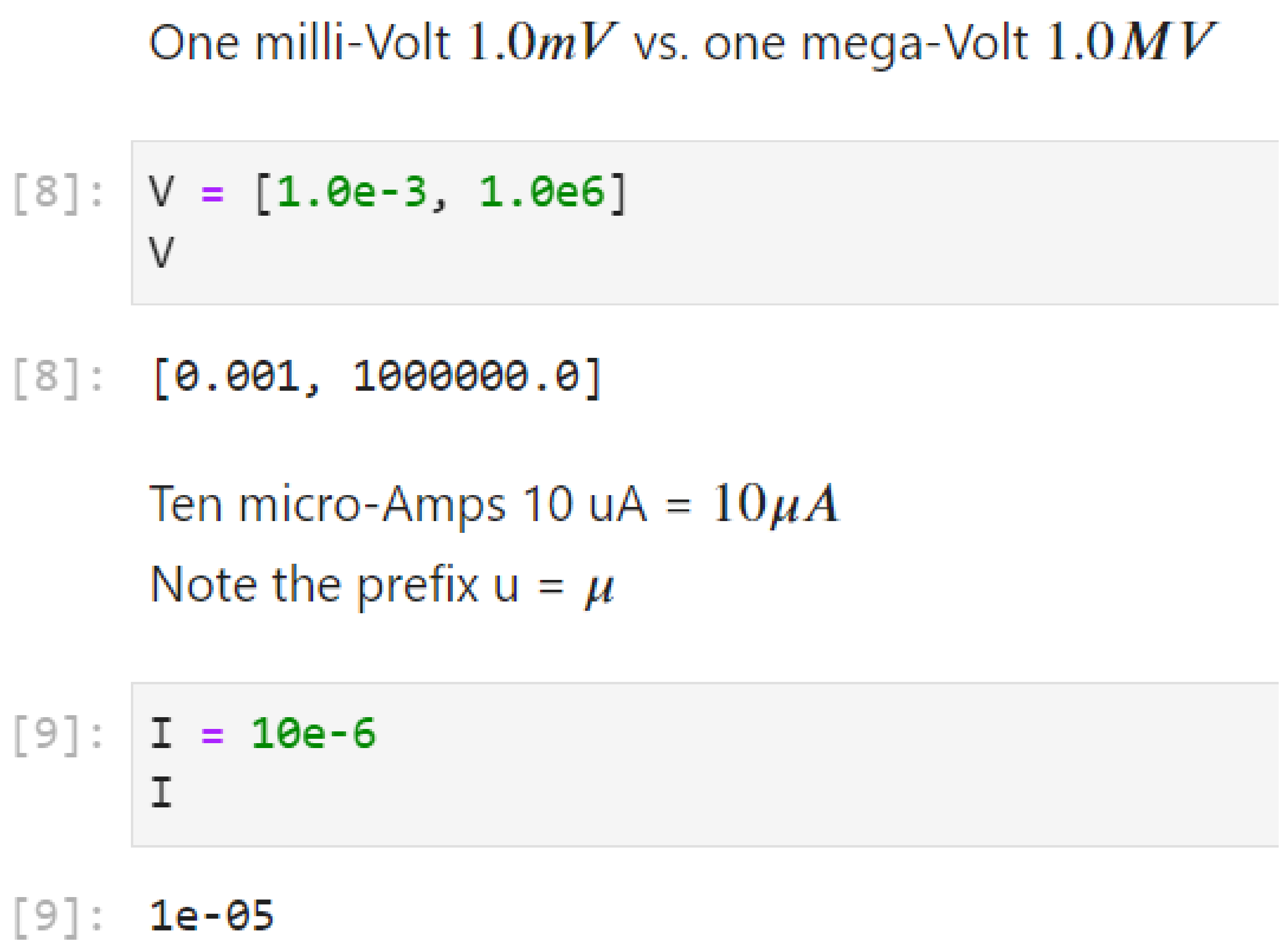
Common prefixes



Common Prefixes used with SI Units

Prefix	Symbol	Meaning	Order of Magnitude
giga-	G	1 000 000 000	10^{9}
mega-	M	1 000 000	10^{6}
kilo-	k	1 000	10^{3}
hecto-	h	100	10^{2}
deka-	da	10	10^{1}
	base unit	1	10^{0}
deci-	d	0.1	10^{-1}
centi-	c	0.01	10^{-2}
milli-	m	0.001	10^{-3}
micro-	μ	0.000 001	10^{-6}
nano-	n	0.000 000 001	10^{-9}

Common pitfalls with prefixes



One milli-Volt $1.0mV$ vs. one mega-Volt $1.0MV$

```
[8]: V = [1.0e-3, 1.0e6]
     V
```

```
[8]: [0.001, 1000000.0]
```

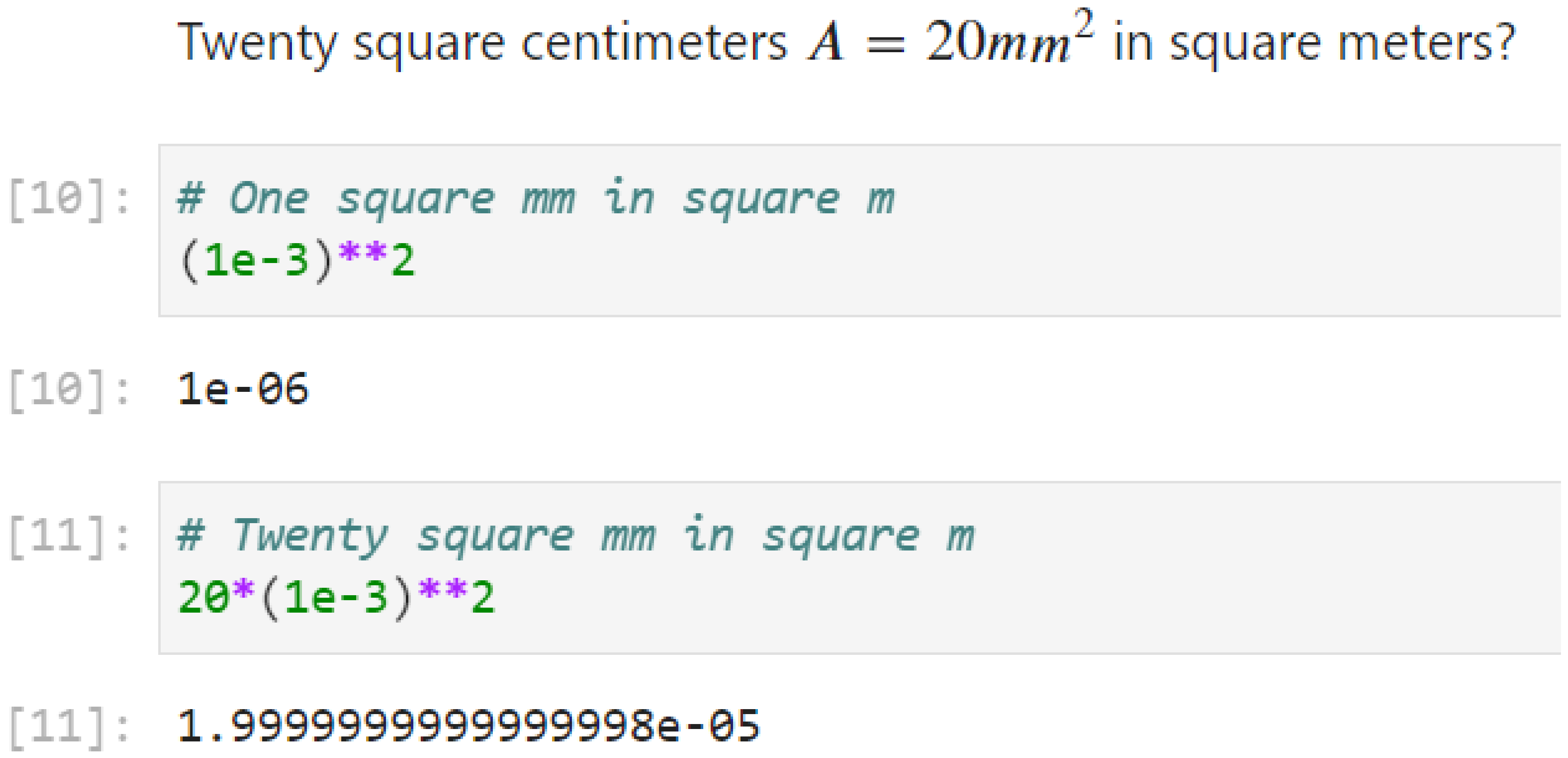
Ten micro-Amps 10 uA = $10\mu A$

Note the prefix u = μ

```
[9]: I = 10e-6
     I
```

```
[9]: 1e-05
```

Common pitfalls – square mm

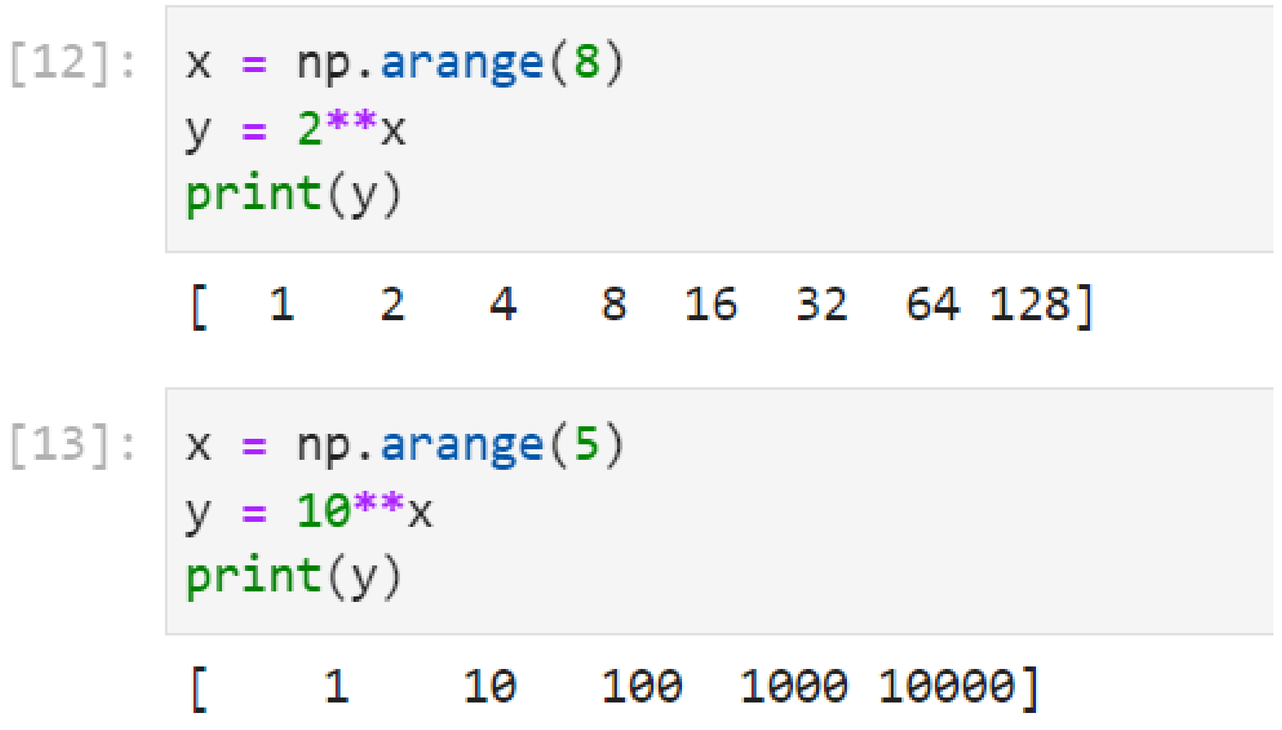


Twenty square centimeters $A = 20mm^2$ in square meters?

```
[10]: # One square mm in square m
      (1e-3)**2

[10]: 1e-06

[11]: # Twenty square mm in square m
      20*(1e-3)**2

[11]: 1.9999999999999998e-05
```

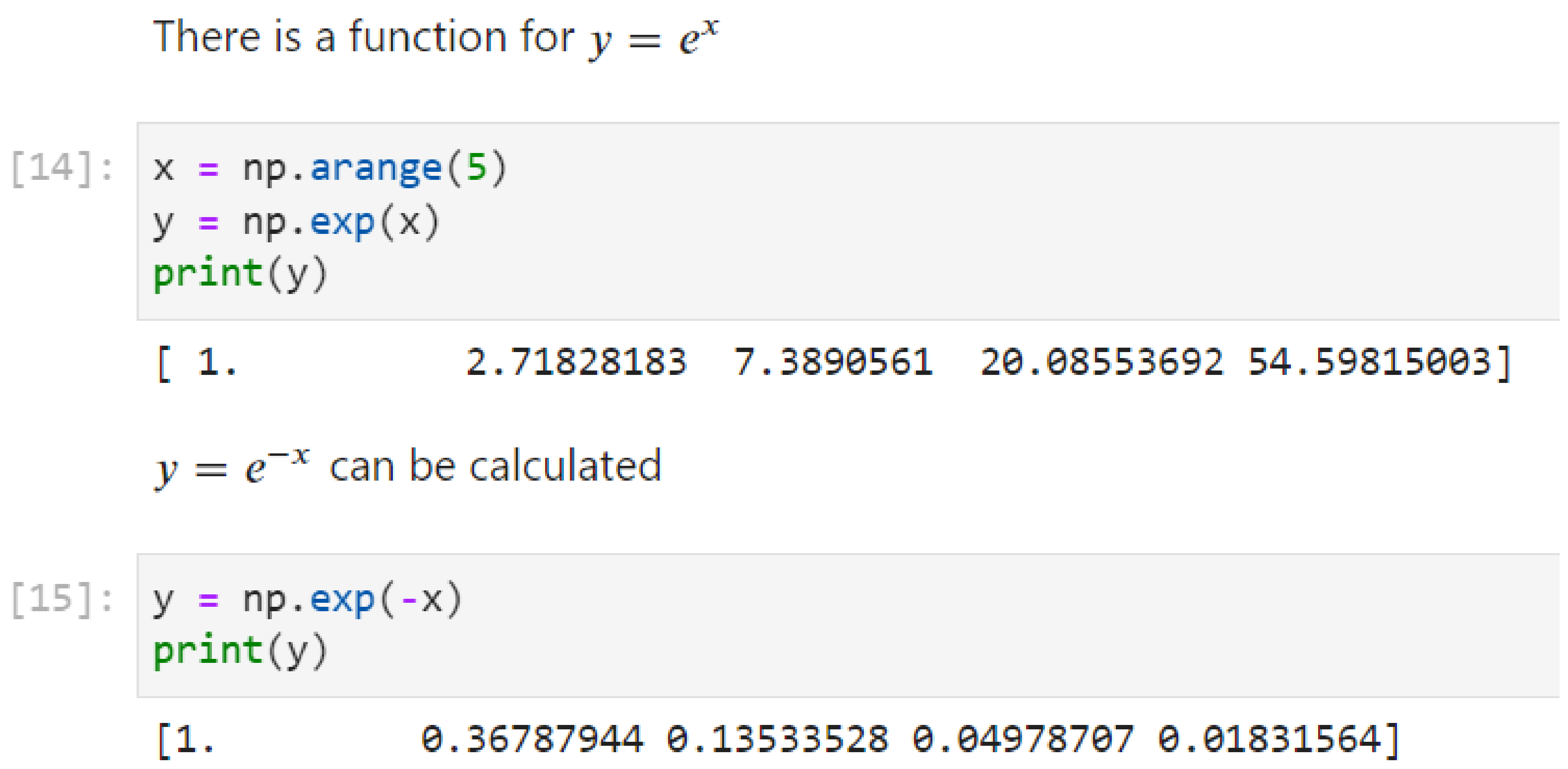
Exponential functions

Power functions



```
[12]: x = np.arange(8)
      y = 2**x
      print(y)

[  1   2   4   8  16  32  64 128]

[13]: x = np.arange(5)
      y = 10**x
      print(y)

[    1    10   100  1000 10000]
```

Exponential function



There is a function for $y = e^x$

```
[14]: x = np.arange(5)
      y = np.exp(x)
      print(y)
```

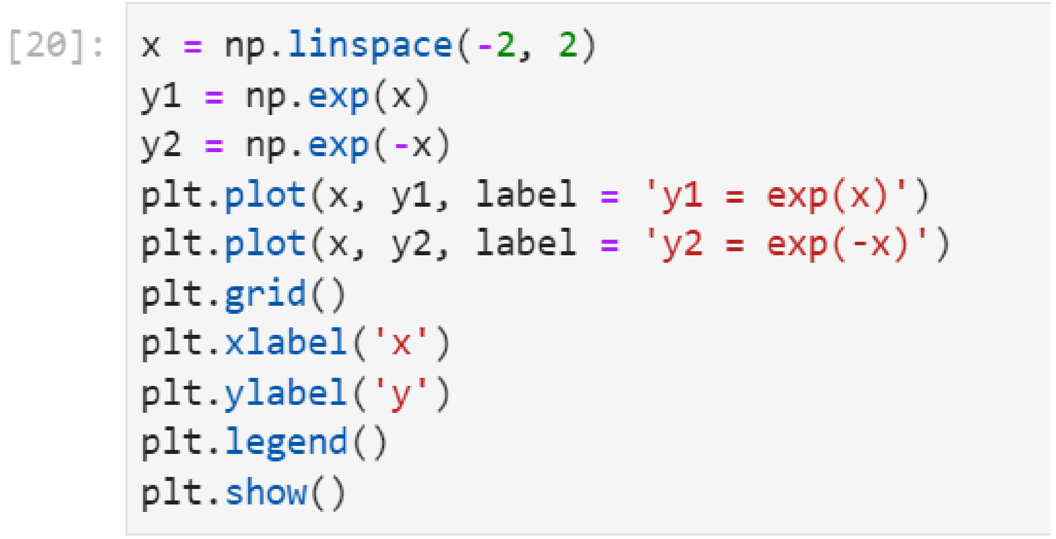
```
[ 1.          2.71828183  7.3890561  20.08553692 54.59815003]
```

$y = e^{-x}$ can be calculated

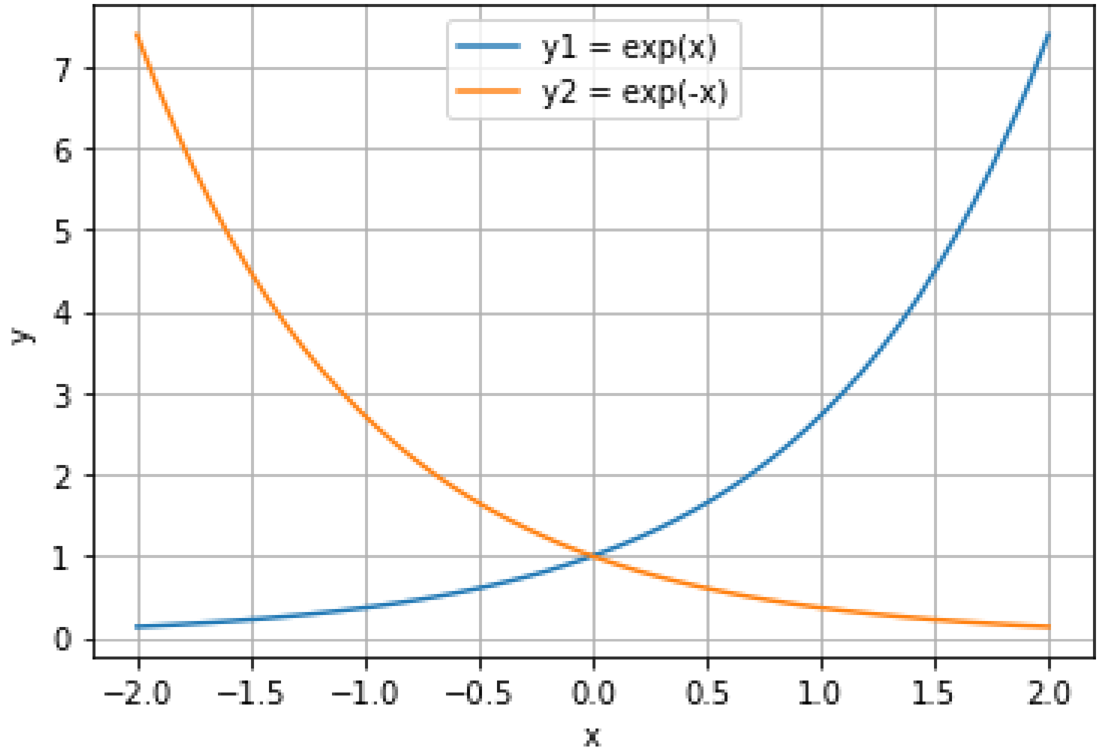
```
[15]: y = np.exp(-x)
      print(y)
```

```
[1.         0.36787944 0.13533528 0.04978707 0.01831564]
```

exp(x) vs. exp(-x)



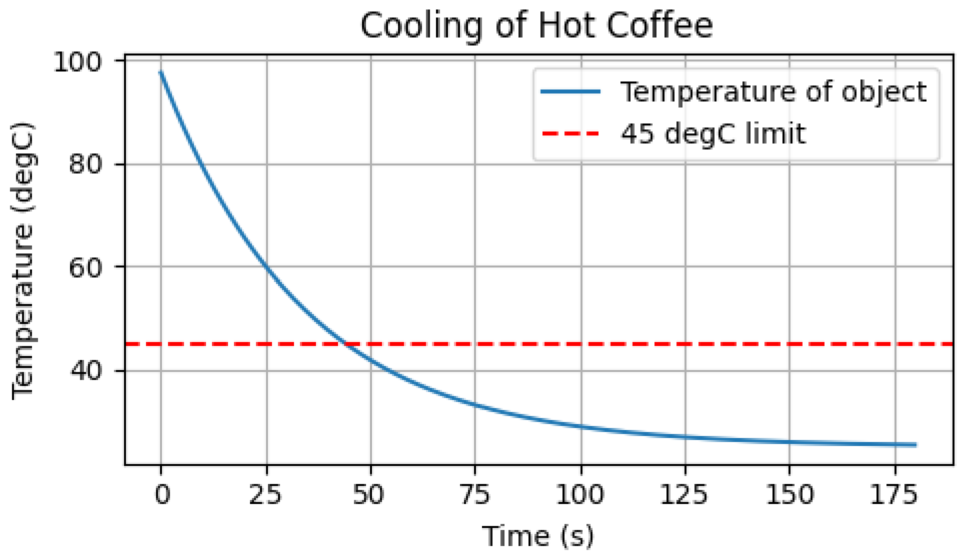
```
[20]: x = np.linspace(-2, 2)
      y1 = np.exp(x)
      y2 = np.exp(-x)
      plt.plot(x, y1, label = 'y1 = exp(x)')
      plt.plot(x, y2, label = 'y2 = exp(-x)')
      plt.grid()
      plt.xlabel('x')
      plt.ylabel('y')
      plt.legend()
      plt.show()
```



Example – Heating or cooling of a body

Heat transfer is a classical example of exponential behaviour. **Isaac Newton** (1701) was first to formulate the rate of heat loss of a body is directly proportional to the difference in the temperatures between the body and its environment. When this is solved, the temperature of the body follows an exponential expression.

$$T(t) = T_{\text{env}} + (T(0) - T_{\text{env}})e^{-rt}$$

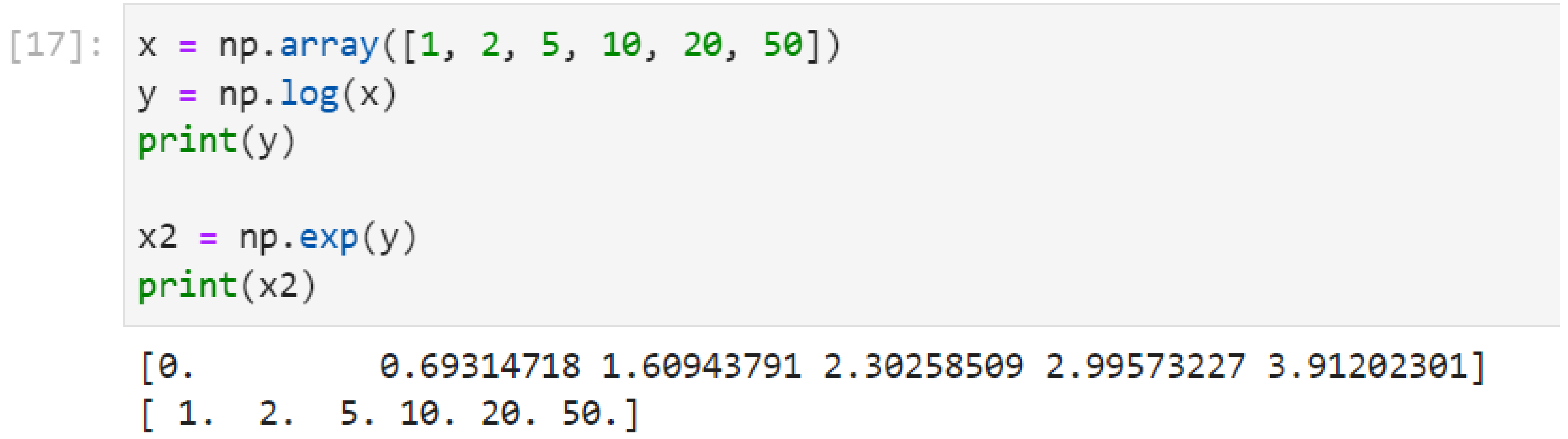


Newton's law of cooling - Wikipedia

Logarithmic functions

Natural logarithm

$$y = ln(x) \leftrightarrow x = e^y$$

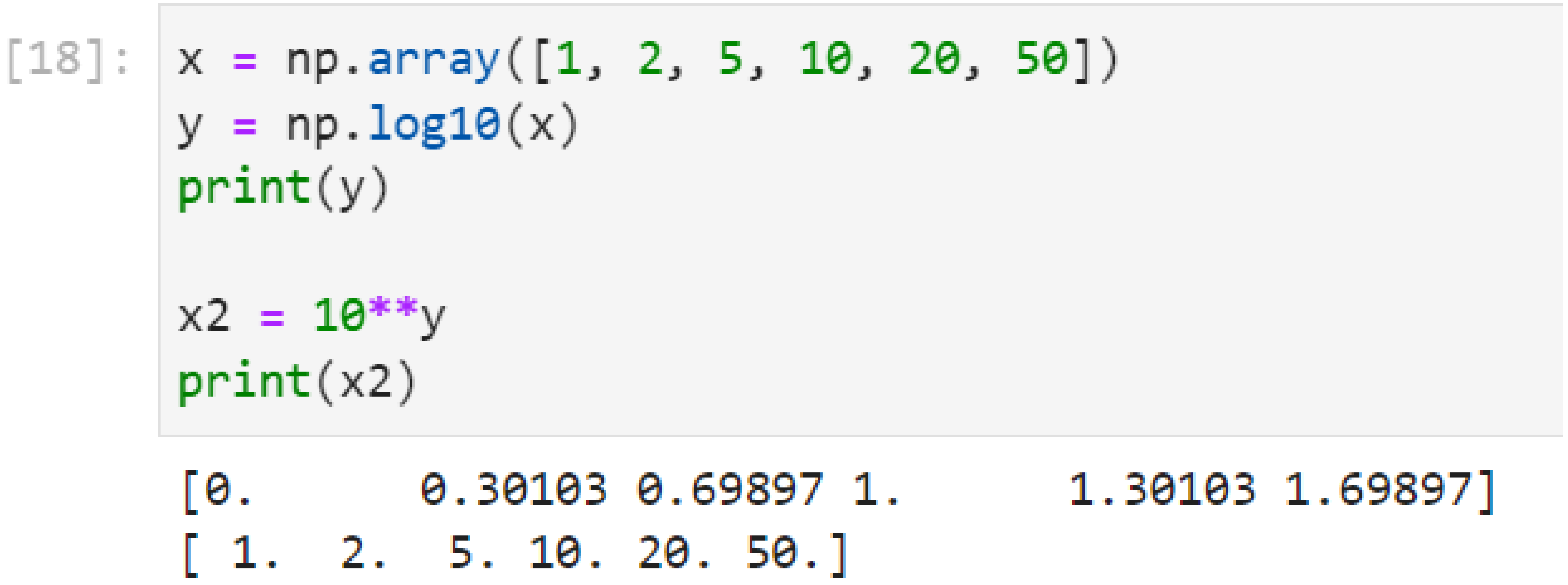


```
[17]: x = np.array([1, 2, 5, 10, 20, 50])
      y = np.log(x)
      print(y)

      x2 = np.exp(y)
      print(x2)
```

```
[0.         0.69314718 1.60943791 2.30258509 2.99573227 3.91202301]
[ 1.  2.  5. 10. 20. 50.]
```

10-base logarithm

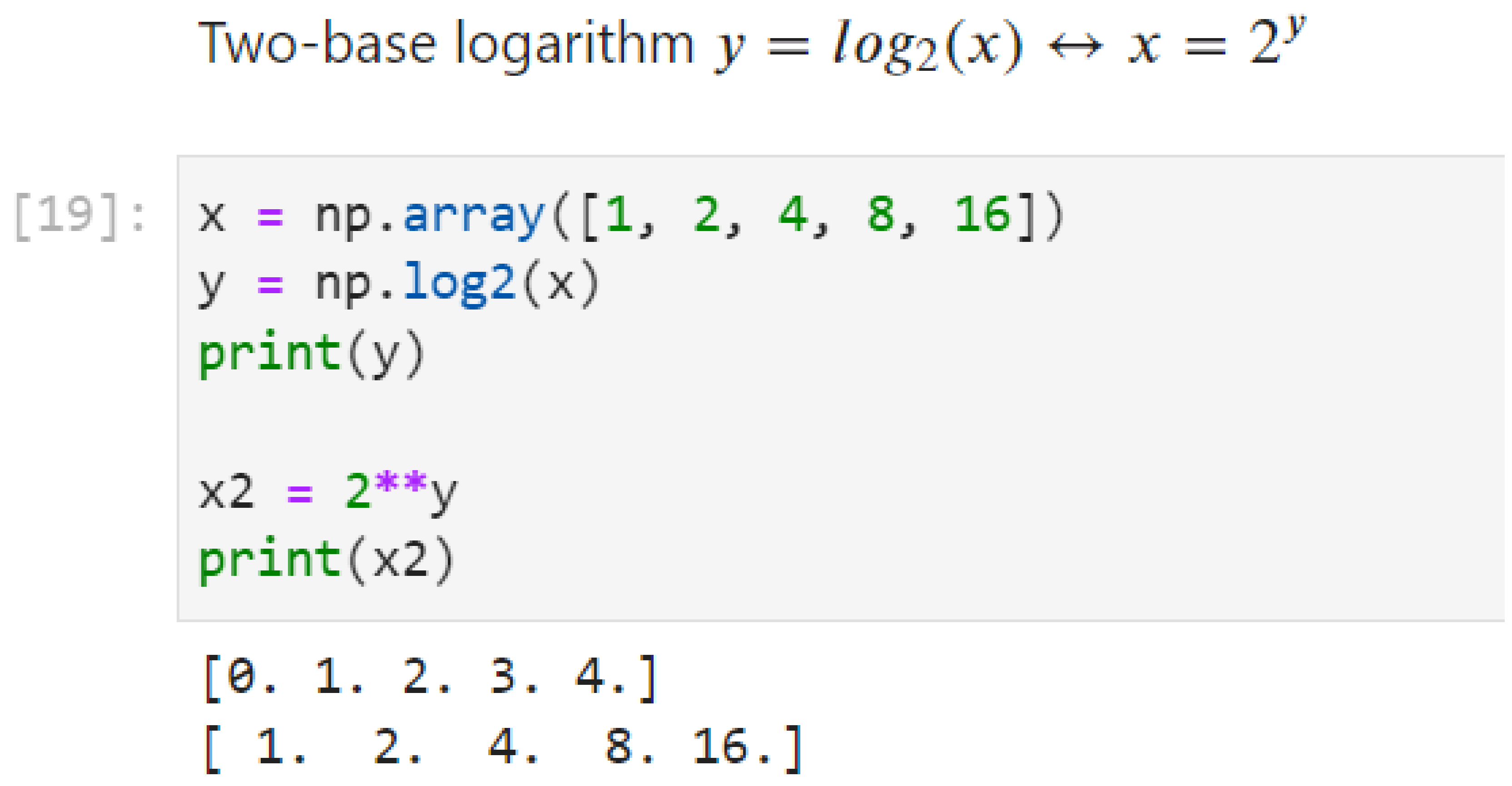
$$y = log(x) \leftrightarrow x = 10^y$$



```
[18]: x = np.array([1, 2, 5, 10, 20, 50])
      y = np.log10(x)
      print(y)

      x2 = 10**y
      print(x2)
```

```
[0.         0.30103 0.69897 1.         1.30103 1.69897]
[ 1.  2.  5. 10. 20. 50.]
```

Two-base logarithm



Two-base logarithm $y = log_2(x) \leftrightarrow x = 2^y$

```
[19]: x = np.array([1, 2, 4, 8, 16])
      y = np.log2(x)
      print(y)

      x2 = 2**y
      print(x2)
```

```
[0. 1. 2. 3. 4.]
[ 1.  2.  4.  8. 16.]
```

Example – Sound intensity in decibels

One of the most common examples of logarithmic functions is **sound intensity measured in decibels (dB)**. Sound intensity level (in decibels) is related to the actual intensity (in W/m^2) by:

$$I_{dB} = 10 \cdot \log_{10}\left(\frac{I}{I_0}\right)$$

Where $I_0 = 1 \times 10^{-12}\ W/m^2$ is the reference intensity (threshold of human hearing). Human ears perceive sound intensity logarithmically rather than linearly. Examples:

- A whisper ~ 30 dB ($\mathrm{I} = 1 \times 10^{-9}\ W/m^2$)
- Normal converstation ~ 60 dB ($\mathrm{I} = 1 \times 10^{-6}\ W/m^2$)
- Rock concert at stadium ~ 120 dB ($\mathrm{I} = 1\ W/m^2$)

Sound intensity - Wikipedia

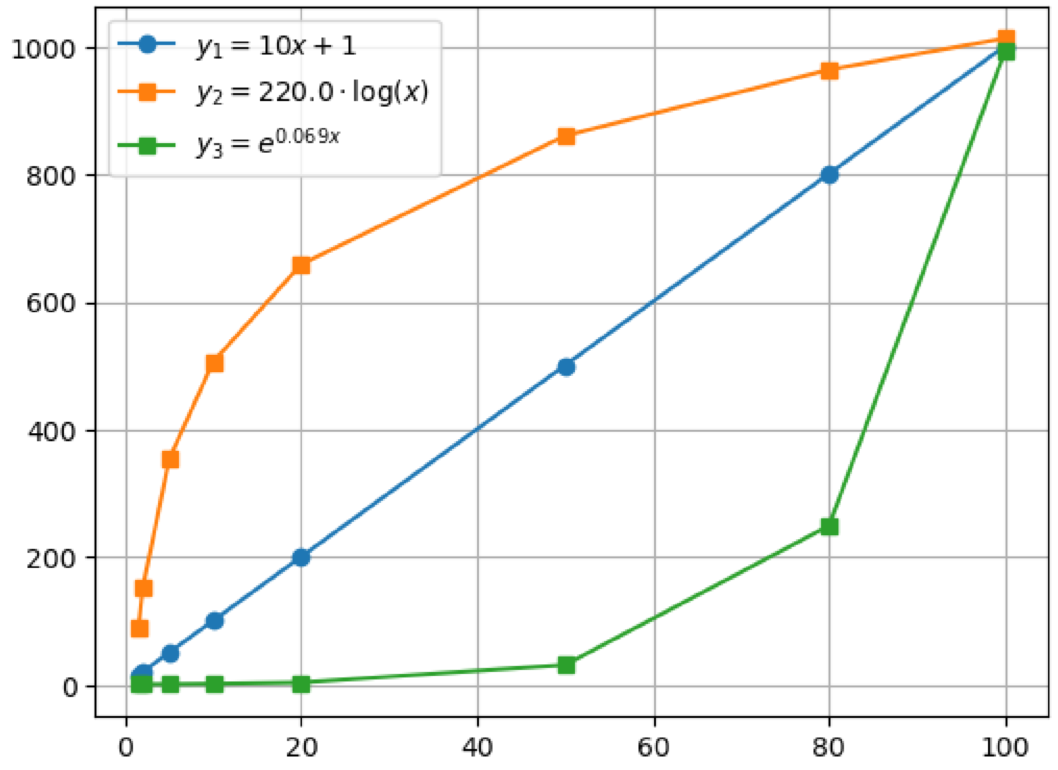
Semilog and log-log scales

Example functions – linear, logarithmic, and exponential

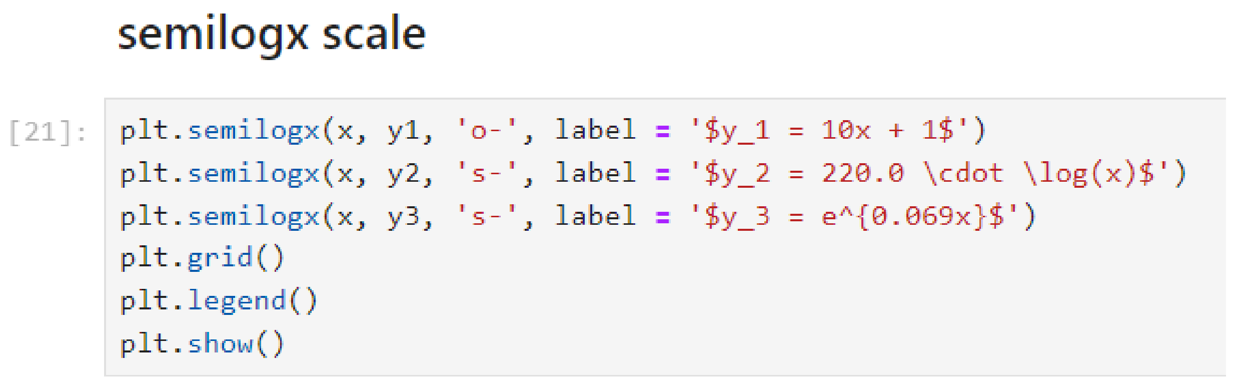
Three functions in linear scales (plt.plot).

```
[20]: x = np.array([1.5, 2.0, 5.0, 10.0, 20.0, 50.0, 80.0, 100.0])
      y1 = 10*x + 1
      y2 = 220.0*np.log(x)
      y3 = np.exp(0.069*x)
      plt.plot(x, y1, 'o-', label = '$y_1 = 10x + 1$')
      plt.plot(x, y2, 's-', label = '$y_2 = 220.0 \cdot \log(x)$')
      plt.plot(x, y3, 's-', label = '$y_3 = e^{0.069x}$')
      plt.grid()
      plt.legend()
      plt.show()
```

In this example, we have used plot function. It is usually tried first. Both x and y axis are linear, so linear function, such as y = kx + b, is shown as a line (blue curve).

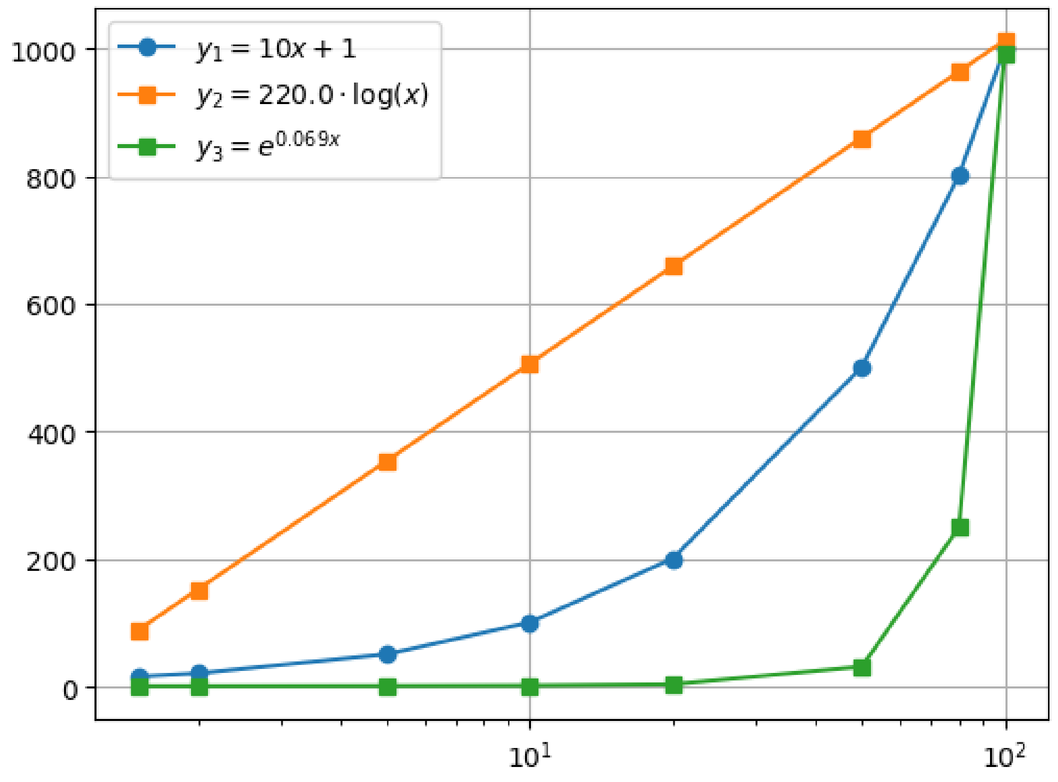


Semilogx scale (x - logarithmic, y - linear)

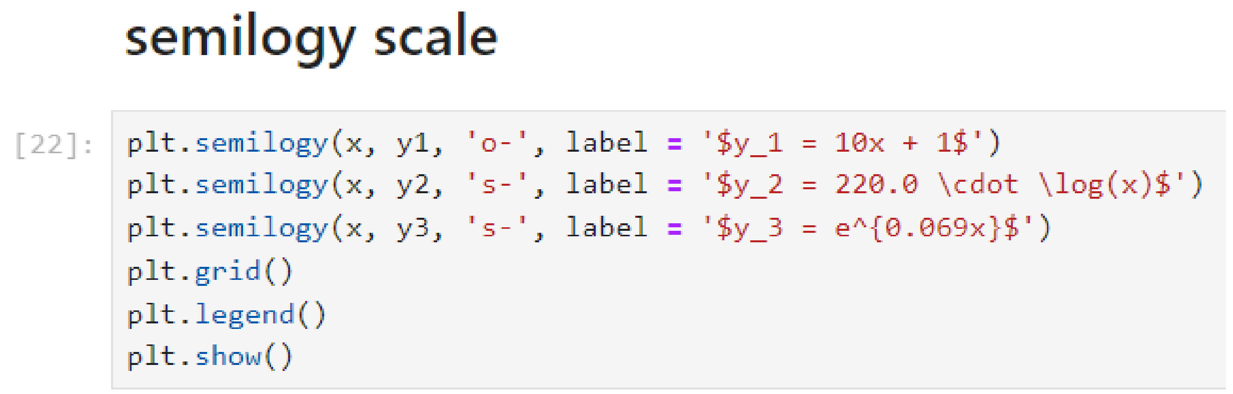

semilogx scale

```
[21]: plt.semilogx(x, y1, 'o-', label = '$y_1 = 10x + 1$')
      plt.semilogx(x, y2, 's-', label = '$y_2 = 220.0 \cdot \log(x)$')
      plt.semilogx(x, y3, 's-', label = '$y_3 = e^{0.069x}$')
      plt.grid()
      plt.legend()
      plt.show()
```

In this second example, we have used **semilogx** function. Now x-axis is logarithmic and y-axis is linear. This scale is usally used, if the x values vary over several decades. The logarithmic function y = b + k*log(x) is shown linear (orange curve).



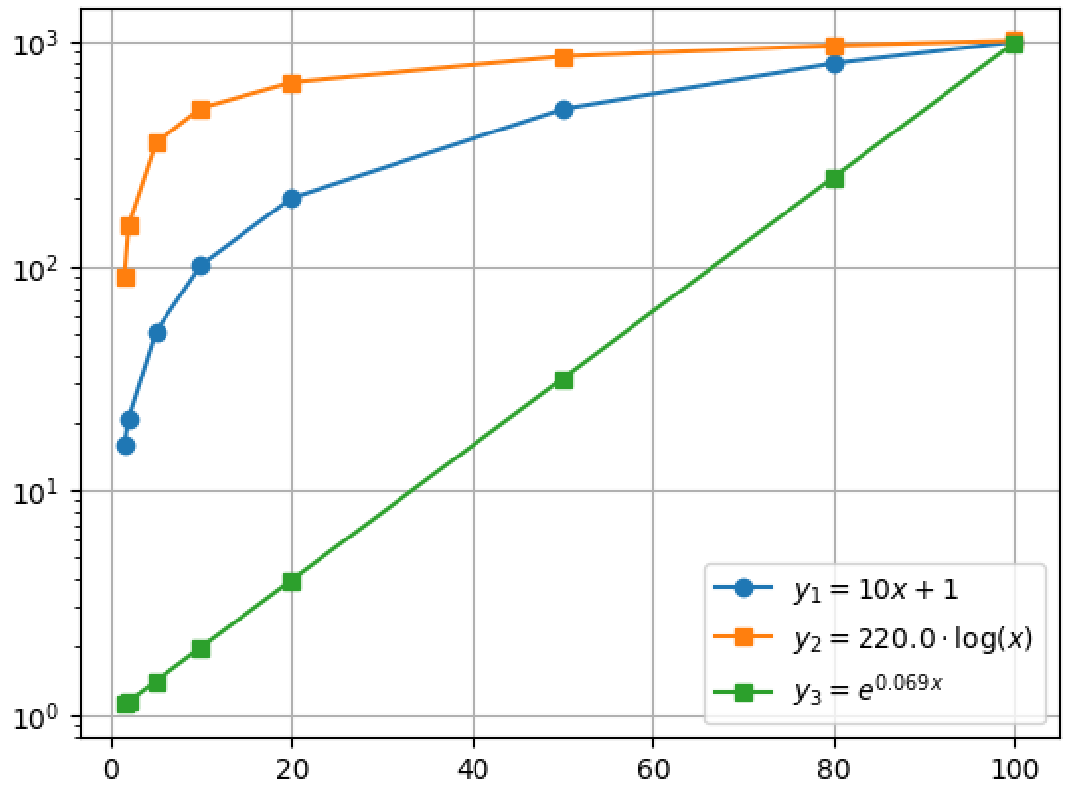
Semilogy scale (x - linear, y - logarithmic)



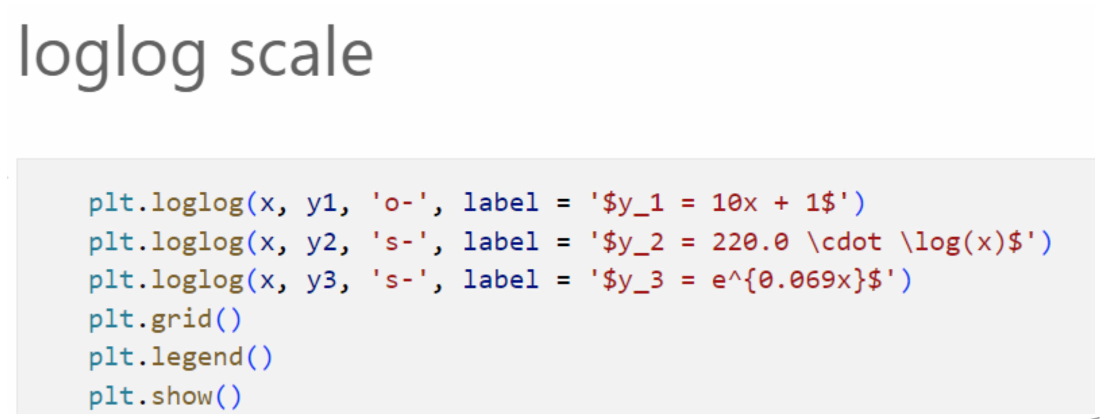
semilogy scale

```
[22]: plt.semilogy(x, y1, 'o-', label = '$y_1 = 10x + 1$')
      plt.semilogy(x, y2, 's-', label = '$y_2 = 220.0 \cdot \log(x)$')
      plt.semilogy(x, y3, 's-', label = '$y_3 = e^{0.069x}$')
      plt.grid()
      plt.legend()
      plt.show()
```

In this third example, we have used **semilogy** scale. In this case x-axis is linear and y-axis is logarithmic. It is typically used, when y values can vary a lot. The exponential function y = exp(k*x) is shown linear (green curve).



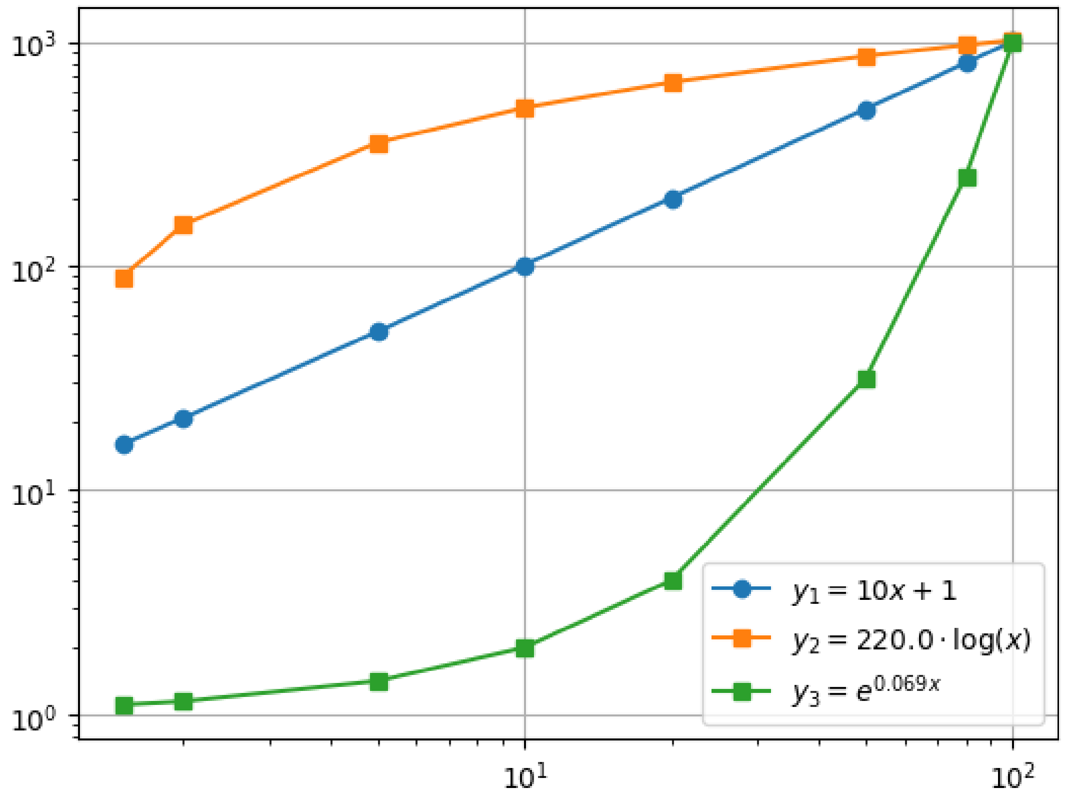
Log-log scale (both x and y are logarithmic)



loglog scale

```
plt.loglog(x, y1, 'o-', label = '$y_1 = 10x + 1$')
plt.loglog(x, y2, 's-', label = '$y_2 = 220.0 \cdot \log(x)$')
plt.loglog(x, y3, 's-', label = '$y_3 = e^{0.069x}$')
plt.grid()
plt.legend()
plt.show()
```

Log-log-scales are used if both x-and y-values vary over several decades. Linear functions are drawn linear in these scales, logarithmic function is almost similar as in linear scales, and exponential function rises up, like with linear scale, as it can be seen in this graph.

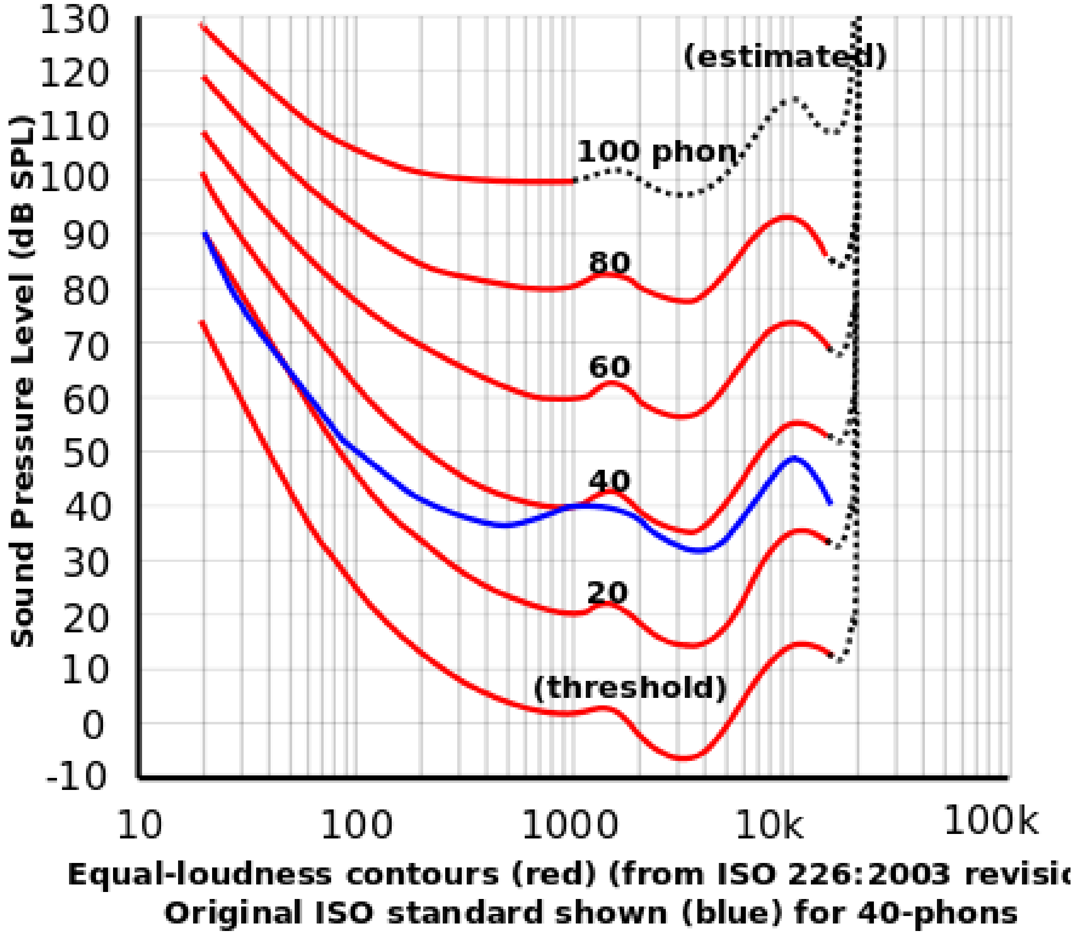


Typical engineering applications

Logarithmic scales are commonly used in engineering applications as the measured values can vary a lot:

- Sound pressure level measurements
- Spectral density calculations
- Decibel scale
- Signal-to-noise ratio
- Dynamic range

On the right is a graph of equal-loudness contours of normal human auditory system. Notice both x and y scales are logarithmic (decibel, dB, is a logarithmic function).

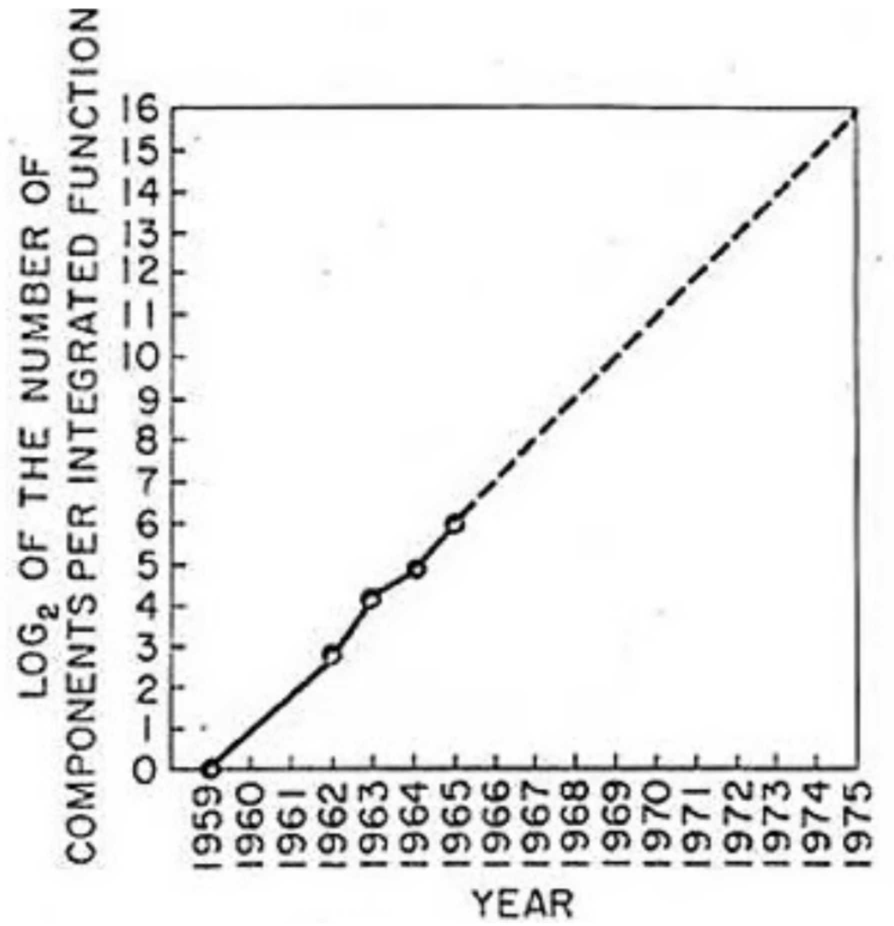


Example – Number of transistors in ICs

A classical example of exponential growth and use of semilog scales is the observation that the number of transistors in integrated electronic circuits (IC) in computers doubles every two years.

This was first observed by Gordon E. Moore in 1965, when he was asked to predict the future of electronics for the next 10 years.

[What is Moore's Law? - Our World in Data]



Next steps

- Practice – Lab 4
  - Moodle contains a practice quiz
  - Notebook can be found from OMA assignments
- Read more
  - Exponents and Logs with Python - Python for Undergraduate Engineers
    - Note: the same naming convention is used both in basic Python math-package and in numpy functions!
    - If you need numerical arrays, remember use the numpy package (np.exp())
  - numpy.exp — NumPy v1.23 Manual
  - numpy.log — NumPy v1.23 Manual