

cheat-sheet

February 21, 2026

1 SCUQ Cheat Sheet

First, we have to import from scuq

```
[63]: from scuq.quantities import Quantity
      from scuq.uncertainities import UncertainInput, Context
      from scuq.si import VOLT, WATT
```

We construct some quantities

```
[64]: voltage1_ui = UncertainInput(1, 0.2)
      voltage1 = Quantity(VOLT, voltage1_ui)
      print(f'U1 = {voltage1}')

      voltage2_ui = UncertainInput(2, 0.1)
      voltage2 = Quantity(VOLT, voltage2_ui)
      print(f'U2 = {voltage2}')

      print(f'U1 + U2 = {voltage1 + voltage2}')
```

U1 = 1 +/- 0.2 V

U2 = 2 +/- 0.1 V

U1 + U2 = 3 +/- 0.223606797749979 [NC] V

Note that the uncertainty of the sum is automatically correct:

$$\sigma_{a+b} = \sqrt{\sigma_a^2 + \sigma_b^2 + 2\sigma_{ab}}$$

with $\sigma_{ab} = \rho_{ab}\sigma_a\sigma_b$.

And $\rho_{ab} = 0$ here.

What happen if we sum up the same voltage ($\rho = 1$)?

```
[65]: print(f'U1 + U1 = {voltage1 + voltage1}')
```

U1 + U1 = 2 +/- 0.4 V

1.1 Context

A context is needed to track correlations between quantities.

```
[66]: ctx = Context()
print(f'Correlation U1,U2: {ctx.get_correlation(voltage1, voltage2)}')
print(f'Correlation U1,U1: {ctx.get_correlation(voltage1, voltage1)}')

for cor in [0, 0.5, -0.5, 1]:
    ctx.set_correlation(voltage1, voltage2, cor)
    print(f'Correlation U1,U2: {ctx.get_correlation(voltage1, voltage2)}')
    print(f'Uncertainty of U1+U2: {ctx.uncertainty(voltage1+voltage2)}')

# reset correlation
ctx.set_correlation(voltage1, voltage2, 0)
print(voltage1+voltage2)
voltage_sum = voltage1 + voltage2
print(f'Sum of U1+U2: {voltage_sum}')
# access value, uncertainty, unit
val_sum = voltage_sum.get_value(VOLT).get_value()
uc_sum = ctx.uncertainty(voltage_sum)
unit_sum = voltage_sum._unit
print(f'Value: {val_sum}; Uncertainty: {uc_sum}; Unit: {unit_sum}')
```

```
Correlation U1,U2: 0.0
Correlation U1,U1: 1.0
Correlation U1,U2: 0
Uncertainty of U1+U2: 0.223606797749979 V
Correlation U1,U2: 0.5
Uncertainty of U1+U2: 0.2645751311064591 V
Correlation U1,U2: -0.5
Uncertainty of U1+U2: 0.17320508075688776 V
Correlation U1,U2: 1
Uncertainty of U1+U2: 0.30000000000000004 V
3 +/- 0.223606797749979 [NC] V
Sum of U1+U2: 3 +/- 0.223606797749979 [NC] V
Value: 3; Uncertainty: 0.223606797749979 V; Unit: V
```

1.2 How to get rid of [NC]?

```
[67]: voltage1_ui = UncertainInput(1, 0.2)
voltage1 = Quantity(VOLT, voltage1_ui)

voltage2_ui = UncertainInput(2, 0.1)
voltage2 = Quantity(VOLT, voltage2_ui)

voltage_sum = voltage1 + voltage2
print(f'Sum of U1+U2 before assign to context: {voltage_sum}')

ctx = Context()
voltage_sum = ctx.value_of(voltage_sum)
```

```
print(f'Sum of U1+U2 after assign to context: {voltage_sum}')
```

Sum of U1+U2 before assign to context: 3 +/- 0.223606797749979 [NC] V

Sum of U1+U2 after assign to context: 3 +/- 0.223606797749979 V

Now, it is easy to set a correlation between U1 and U2. Different correlations result in different uncertainty:

```
[68]: ctx.set_correlation(voltage1, voltage2, 1)
print(f'U1+U2 = {voltage_sum}')
```

```
ctx.set_correlation(voltage1, voltage2, 0)
print(f'U1+U2 = {voltage_sum}')
```

U1+U2 = 3 +/- 0.30000000000000004 V

U1+U2 = 3 +/- 0.223606797749979 V

Often we want to extract value, uncertainty and unit from a Quantity. This is done via the context too:

```
[69]: val, uncert, unit = ctx.value_uncertainty_unit(voltage_sum)
print(val, uncert, unit)
```

3 0.223606797749979 V

1.3 Example code snipped

```
[72]: power = Quantity(WATT, UncertainInput(1, 0.1))
cbl_s21 = Quantity(WATT/WATT, UncertainInput(0.9, 0.05))

power_corrected = power * cbl_s21
print(power_corrected)

ctx = Context()
power_corrected = ctx.value_of(power_corrected)
print(power_corrected)

p_val, p_unc, p_unit = ctx.value_uncertainty_unit(power_corrected)
print(p_val, p_unc, p_unit)
```

0.9 +/- 0.10295630140987001 [NC] W

0.9 +/- 0.10295630140987001 W

0.9 0.10295630140987001 W

1.4 Compatibility with pandas

```
[76]: import pandas as pd

data = [Quantity(WATT, UncertainInput(1, 0.1)), Quantity(WATT, UncertainInput(0.
↪9, 0.05))]
df = pd.DataFrame(data)
```

```
#print(df)

csv_string = df.to_csv(index=True) # index=False, um den Index nicht
    einzuschließen
print(csv_string)
```

```
,0
0,1 +/- 0.1 W
1,0.9 +/- 0.05 W
```

How to write value, uncertainty and unit to different columns?

```
[80]: ctx = Context()

data2 = [ctx.value_uncertainty_unit(d) for d in data]
df2 = pd.DataFrame(data2, columns=['value', 'uncertainty', 'unit'])
# print(df2)

csv_string2 = df2.to_csv(index=False)
print(csv_string2)
```

```
value,uncertainty,unit
1.0,0.1,W
0.9,0.05,W
```