Hello! 👋

Pieter De Vis

Sam Hunt

Maddie Stedman

Samantha Malone

Joe Riordan

Today’s Exercises

comet-toolkit.org/user-guide/training/vhroda

📋 Today’s Agenda

1. **Presentation: CoMet Toolkit Introduction**
   - □ *What is CoMet?*
   - □ *Uncertainty 101*
   - □ *Tools Intro*
2. **Exercises**
   - □ **punpy** basics with in-situ type data
   - □ **obsarray** basics with EO type data

Today’s Exercises

comet-toolkit.org/user-guide/training/vhroda

IDEAS-QA4EO

The CoMet Toolkit – Uncertainties made easy

Pieter De Vis, Sam Hunt, Maddie Stedman, Samantha Malone, Joe Riordan

National Physical Laboratory

VHRODA hands-on tutorial - 19/11/2025

CoMet Toolkit

Python Tools

Uncertainty Handling

Open Source

Tested

HYPERNETS

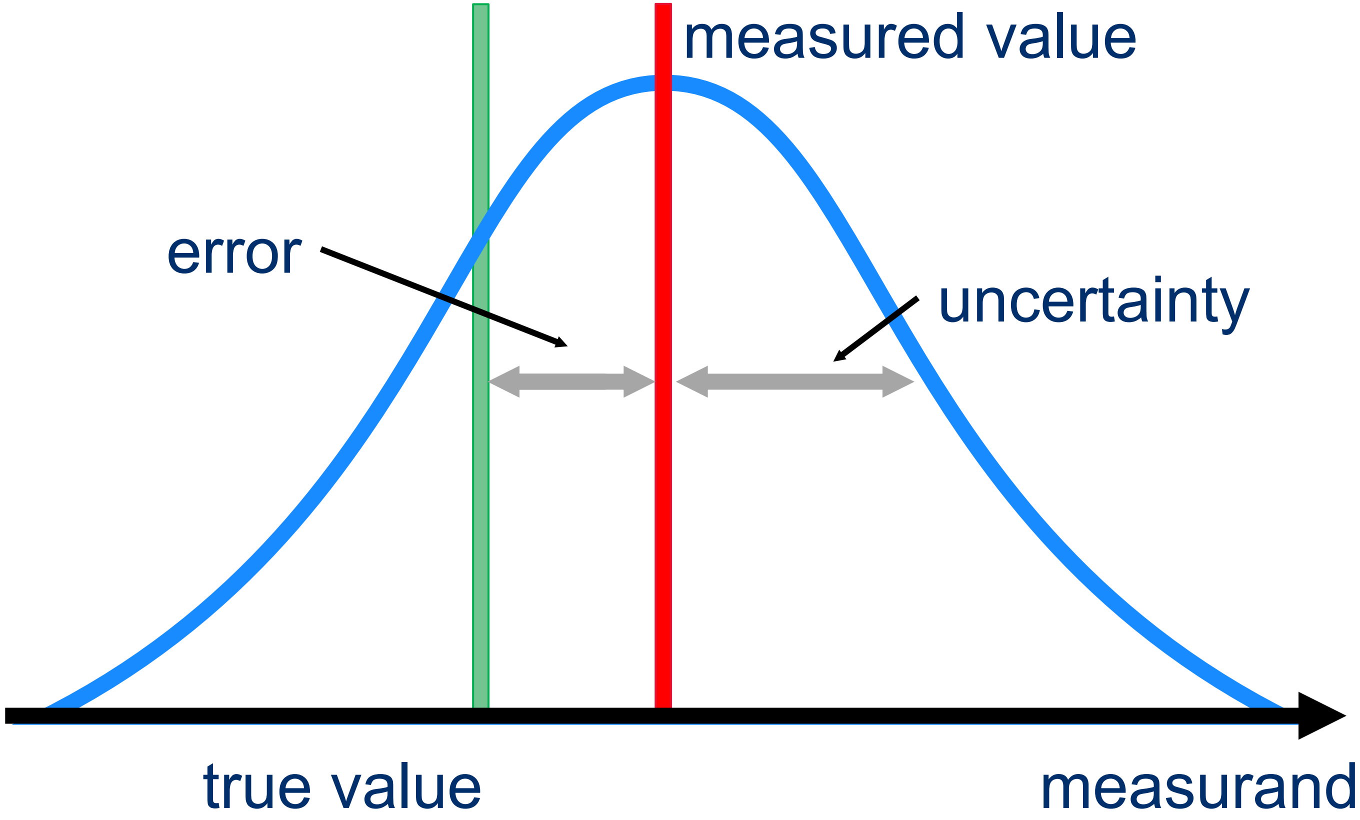
esa truths

Applied

IDEAS-QA4EO

NPL National Physical Laboratory

Uncertainties 101



IDEAS-QA4EO

NPL National Physical Laboratory

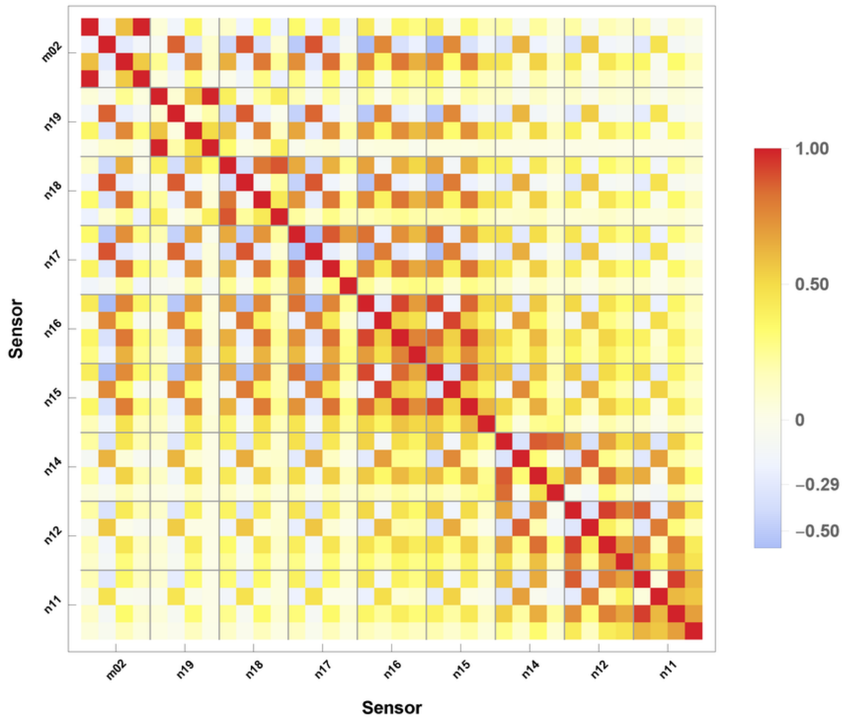
🔍 Error Correlation

🔎 What is Error Correlation?

Errors in a dataset are **not always independent** — e.g., when errors in pixels, bands instruments, or time steps are systematically related.

🤷 Why does it Matter?

- □ Bias uncertainties persist – averaging doesn't help!
- □ Band ratios might be off – affecting retrieval uncertainty
- □ Misleading confidence in trends



Error correlation matrix from Giering et al. 2019

🔍 Error Covariance

🔎 What is Error Covariance?

Combines error correlation and uncertainty

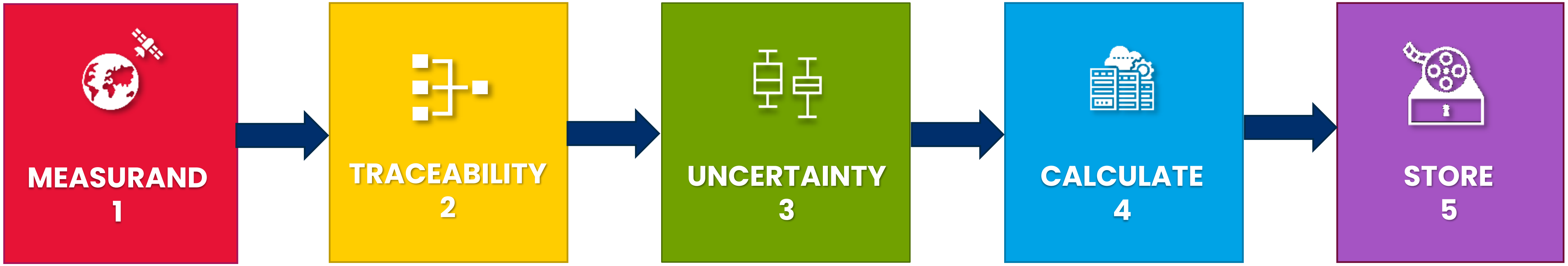
$$\boldsymbol{S} = \boldsymbol{U}\ \boldsymbol{R}\ \boldsymbol{U}^T$$

🤷 Random, systematic and structured uncertainty

- ❑It is the errors that are correlated, not the uncertainty values
- ❑Random means completely uncorrelated, i.e. error correlation is identity matrix
- ❑Systematic means fully correlated, i.e. error correlation filled with 1's

A Metrological Approach

Uncertainties are evaluated and expressed following *QA4EO Five Steps*, a framework which employs the principles of metrology.



MEASURAND 1

TRACEABILITY 2

UNCERTAINTY 3

CALCULATE 4

STORE 5

Define the measurand and measurement function

Establish the traceability with a diagram

Evaluate each source of uncertainty and fill in an effects table

Calculate the data product and uncertainties

Record information for users

QA4EO

IDEAS-QA4EO

FIduceo

CoMet Toolkit

punpy

Propagation UNcertainties in Python

obsarray

Handling uncertainty and error-covariance in datasets

comet_maths

CoMet mathematical algorithms and interpolation tools

UNC Specification

Uncertainty metadata naming conventions

IDEAS-QA4EO

NPL National Physical Laboratory

CoMet Packages:

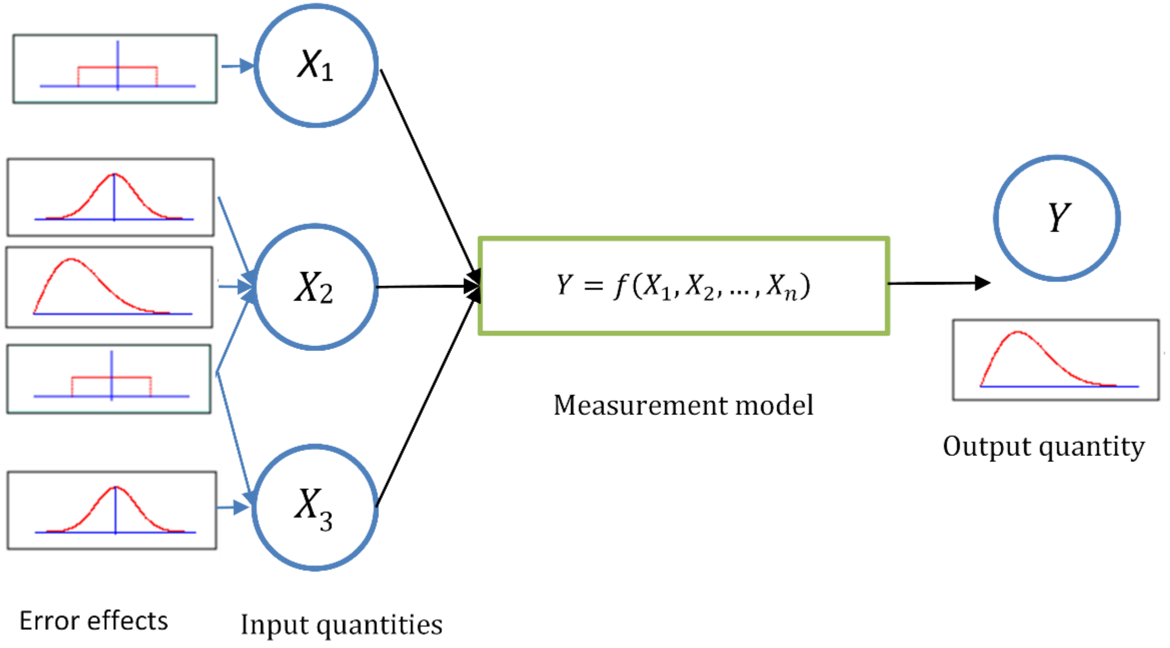
punpy

Propagating Uncertainties with punpy

- ❑ Python module for propagating **random**, **systematic** and **structured uncertainties** through any Python measurement function

- ❑ Flexible in terms of the **specified correlations** along given dimensions or between input quantities

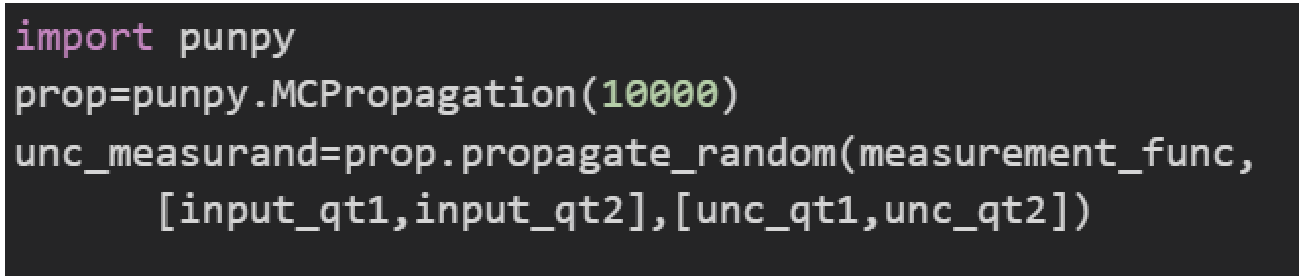
- ❑ ***Monte Carlo*** and ***Law of Propagation of uncertainties*** methods available



Punpy as a Standalone Tool

❑ Simple user **interface**:

1. Import punpy
2. Define measurement function
3. Create MC or LPU object
4. Propagate uncertainties



```
import punpy
prop=punpy.MCPropagation(10000)
unc_measurand=prop.propagate_random(measurement_func,
      [input_qt1,input_qt2],[unc_qt1,unc_qt2])
```

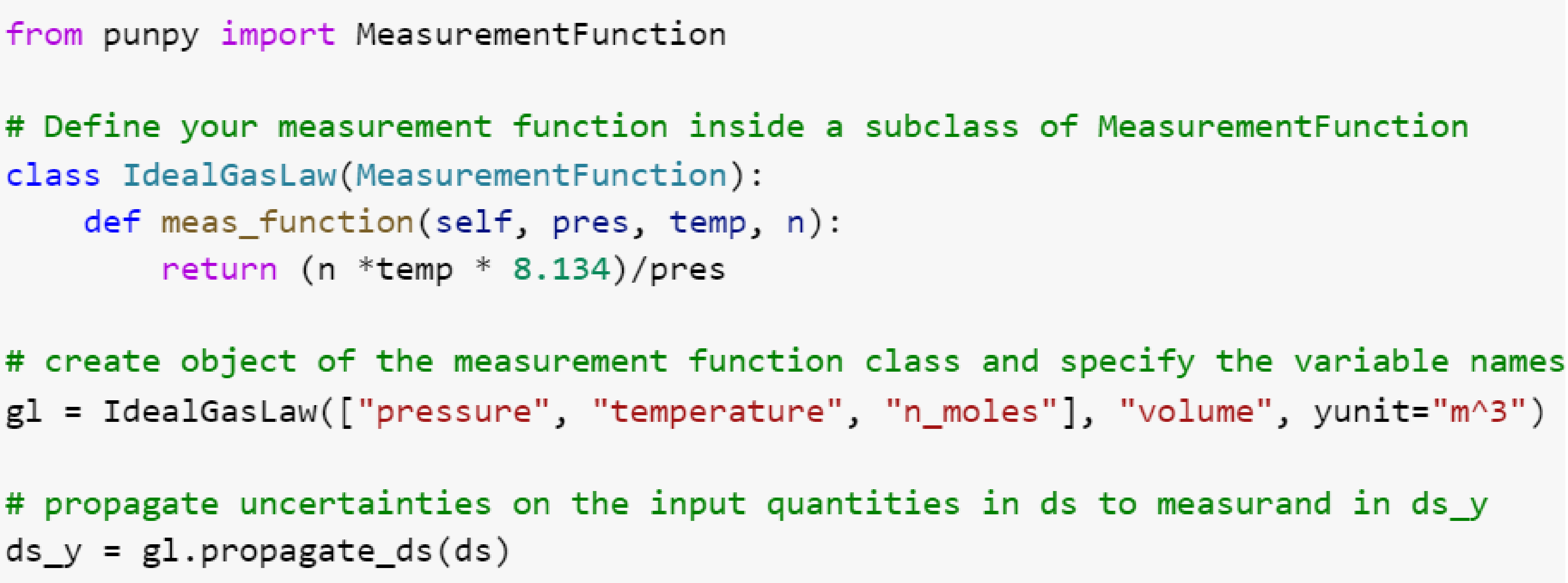
❑ **Measurement function** are defined as python functions that take arrays as input quantities and return an array as measurand

❑ Many optional **keywords** for flexible functionality

- *return_corr*
- *Corr_between*
- *Repeat_dims*
- *Parallel_cores*
- *Output_vars*
- …

Punpy with digital effects tables

- ❑ punpy interfaces with obsarray to make uncertainty propagation as efficient and easy to use as possible
- ❑ ***propagate_ds()*** function returns an obsarray dataset with combined random, systematic and structured uncertainties on measurand



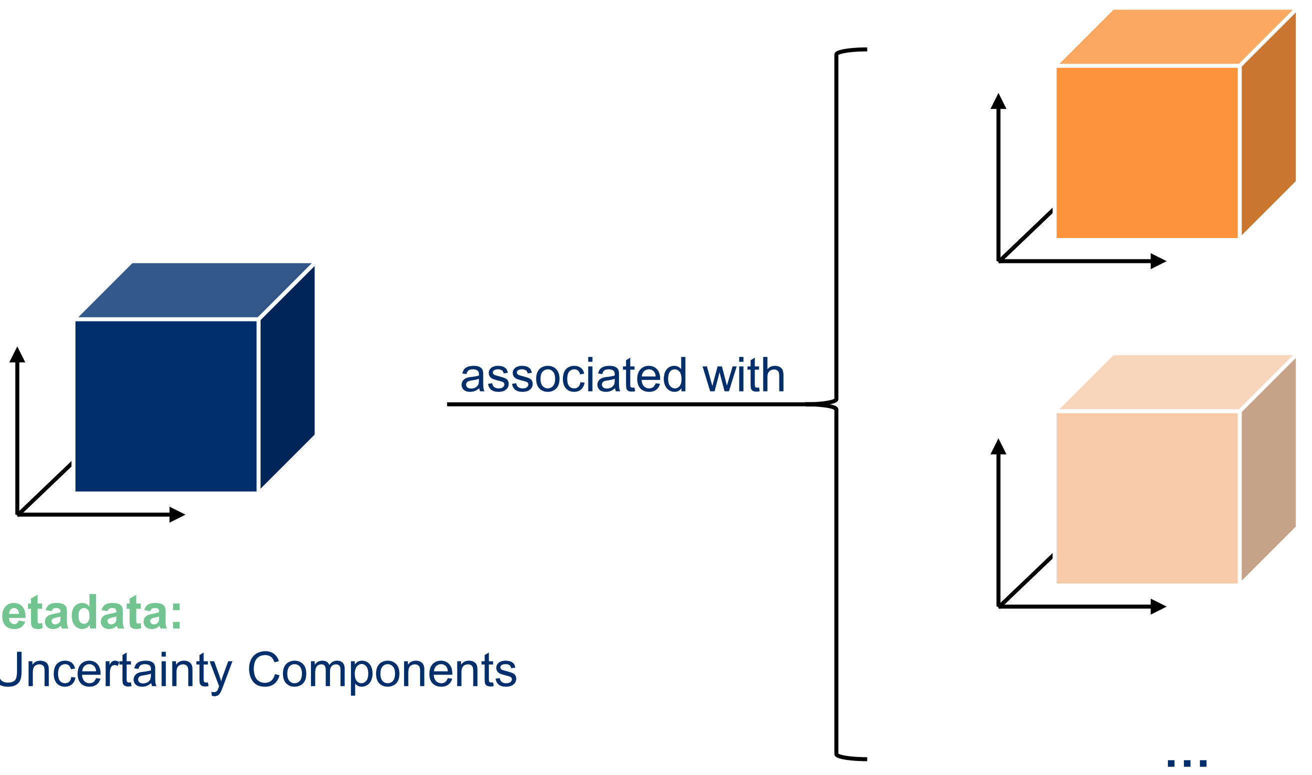
```
from punpy import MeasurementFunction

# Define your measurement function inside a subclass of MeasurementFunction
class IdealGasLaw(MeasurementFunction):
    def meas_function(self, pres, temp, n):
        return (n *temp * 8.134)/pres

# create object of the measurement function class and specify the variable names
gl = IdealGasLaw(["pressure", "temperature", "n_moles"], "volume", yunit="m^3")

# propagate uncertainties on the input quantities in ds to measurand in ds_y
ds_y = gl.propagate_ds(ds)
```

CoMet Packages:

obsarray

UNC Specification

Uncertainty Variable Metadata



Metadata:
- **PDF Shape** (gaussian, …)
- **Units** (abs. or rel.)
- **Error-Correlation**…

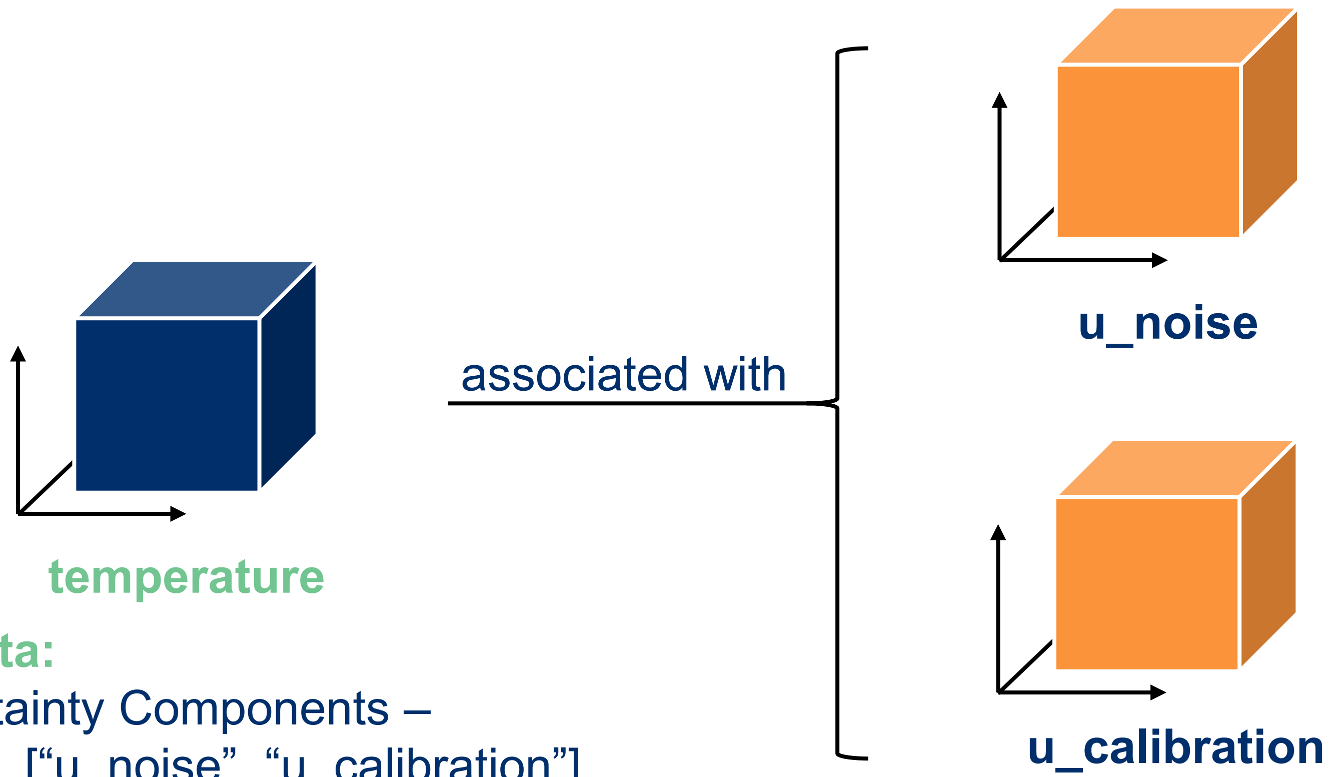
Metadata:
- Uncertainty Components

UNC Specification

🌡️ Example – Temperature Dataset

Observation Variables

Uncertainty Variables



Metadata:
- Uncertainty Components – ["u_noise", "u_calibration"]

Metadata:
- PDF Shape – "gaussian"
- Units – %
- Error-Correlation:
  - All dims - Random

Metadata:
- PDF Shape – "rectangular"
- Units – "K"
- Error-Correlation:
  - **x, y** – systematic
  - **time** – defined by matrix

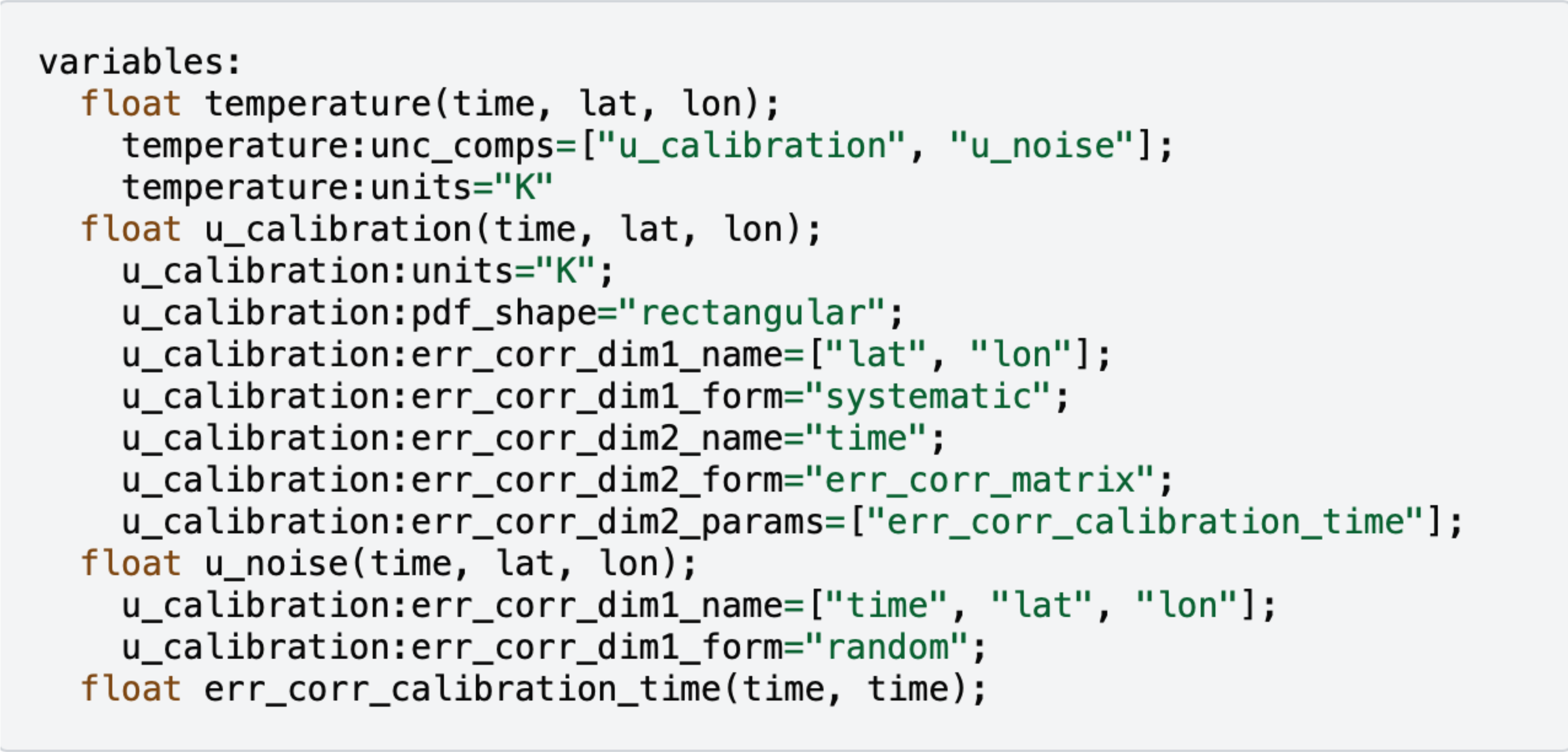
IDEAS-QA4EO

NPL National Physical Laboratory

UNC Specification

obsarray

Example – Temperature Dataset



```
variables:
  float temperature(time, lat, lon);
    temperature:unc_comps=["u_calibration", "u_noise"];
    temperature:units="K"
  float u_calibration(time, lat, lon);
    u_calibration:units="K";
    u_calibration:pdf_shape="rectangular";
    u_calibration:err_corr_dim1_name=["lat", "lon"];
    u_calibration:err_corr_dim1_form="systematic";
    u_calibration:err_corr_dim2_name="time";
    u_calibration:err_corr_dim2_form="err_corr_matrix";
    u_calibration:err_corr_dim2_params=["err_corr_calibration_time"];
  float u_noise(time, lat, lon);
    u_calibration:err_corr_dim1_name=["time", "lat", "lon"];
    u_calibration:err_corr_dim1_form="random";
  float err_corr_calibration_time(time, time);
```

obsarray

Measurement data handling in Python

- □ obsarray is an extension to **xarray** to support defining, storing and interfacing with measurement data – using the UNC specification.
- □ Also has functionality for defining flags following **CF Conventions**.
- □ Working towards including UNC specification within **CF Conventions**.
- □ It is designed to work well with **netCDF** files and for the **Earth Observation** community.

Plugs straight into punpy for propagation through measurement functions!

CoMet Application Examples

punpy

obsarray

comet_maths

IDEAS-QA4EO

NPL
National Physical Laboratory

CoMet Toolkit in Action

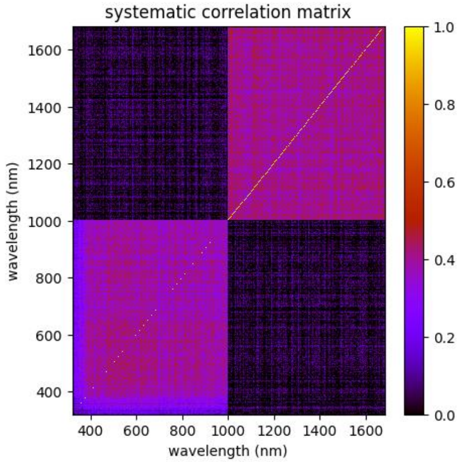
- ❑ Validated against **NIST** uncertainty engine:

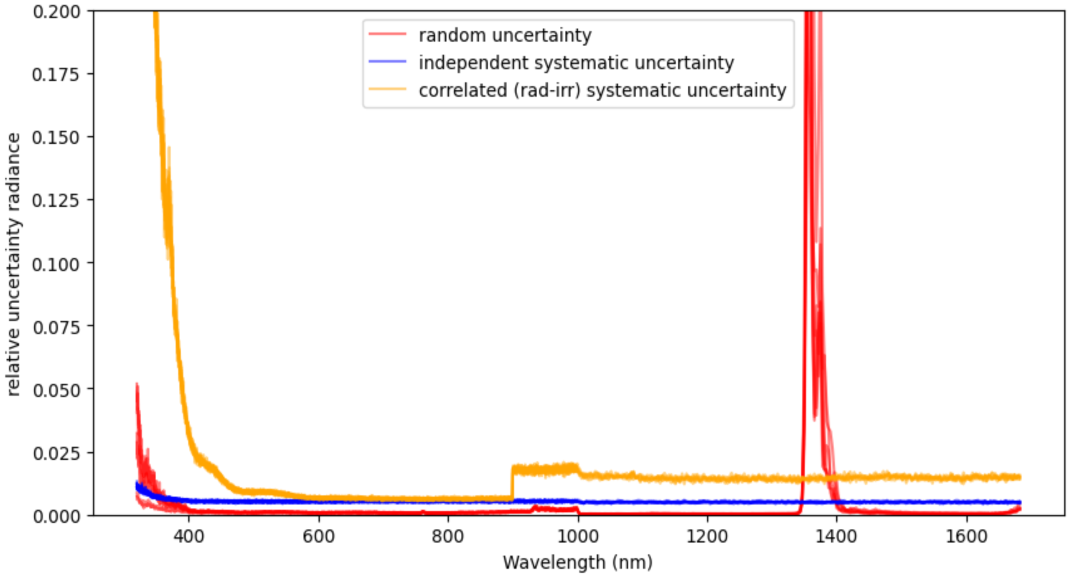
  https://colab.research.google.com/github/comet-toolkit/comet_training/blob/main/NIST_example.ipynb

- ❑ *CoMet* is used in various other projects, such as **QA4EO**, **HYPERNETS**, **CHIME L2**, **FLEX** validation, **TRUTHS** science studies, **LIME**, **FRM4SOC**, **RPV4PICS** see https://www.comet-toolkit.org/projects/

- ❑ **Example** from *hypernets_processor* :

Hypernets is an automated network of in-situ instruments measuring reflectance for L2 satellite validation





Exercises

- ❑ Please go to www.comet-toolkit.org/user-guide/training/

Today’s Exercises

comet-toolkit.org/user-guide/training/vhroda

🎉 CoMet Release

CoMet

□ V1.0 of Comet toolkit has been released as **open source** toolkit:

- www.comet-toolkit.org
- github.com/comet-toolkit

□ Accompanied by training material (**Jupyter** notebooks hosted on google colab):

- www.comet-toolkit.org/examples

□ Documentation & ATBD for individual tools:

- obsarray.readthedocs.io/en/latest/
- punpy.readthedocs.io/en/latest/
- comet-maths.readthedocs.io/en/latest/

📈 Outlook

□ **Current release** will be presented De Vis & Hunt (in prep)

□ Looking to continue to expand the use cases the developed tools
- Aiming to enable uncertainty propagation through **any python measurement function**
- 📧 Please get in touch if you are interested!

□ This has been our first step into this way of working, **many more ideas in a roadmap**
- MetEOR (e.g. CEOS-PVP Cal/Val pipeline, BRDF tool, …)

MetEOR

www.comet-toolkit.org/meteor/

□ **1.5 day training course** tentatively planned for May at NPL

https://qa4eo.org/met4eo/

✅ Summary

□ The **CoMet toolkit** is an open-source software project to develop Python tools for the handling of error-covariance information in the analysis of measurement data

□ This toolkit is based on **robust metrology**, and makes dealing with complexities of uncertainties much easier

□ Includes **obsarray**, **punpy** **&** **comet_maths** as initial offering, to be extended

□ These tools are already being used operationally in various projects (e.g. Hypernets)