DOI: 10.5281/zenodo.3923537

Nikola Vasiljevic

e-WindLidar toolbox

(online) NREL talk
30th Jun 2020

Table of contents

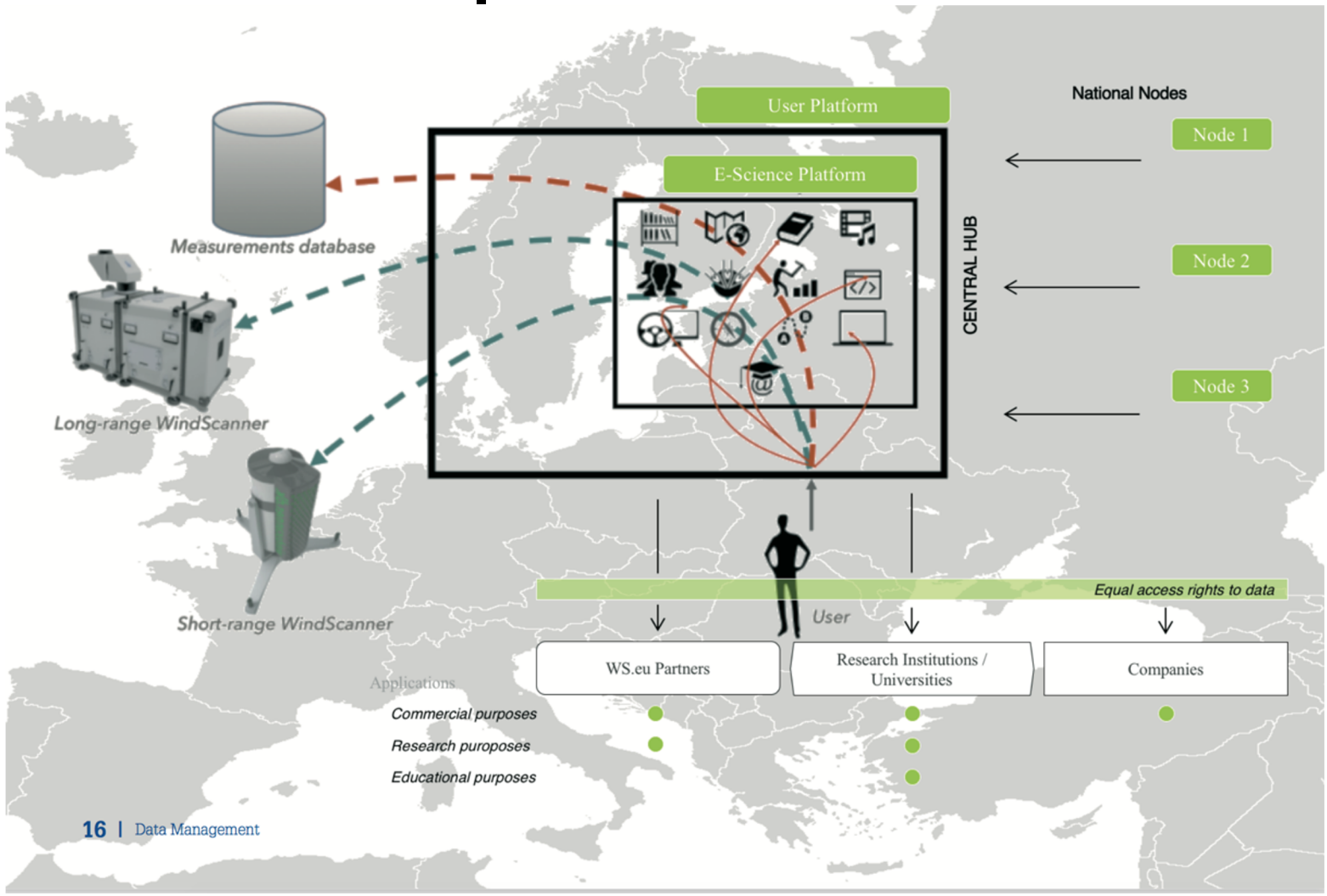
e-WindLidar initiative

- Develop community-based tools that will simplify:
  - Planning and configuration of lidar-based field campaigns
  - Operation of lidars in field campaigns
  - Usage of lidar data

- e-WindLidar idea was conceived during the WindScanner.EU project under the name WindScanner Information System [1]

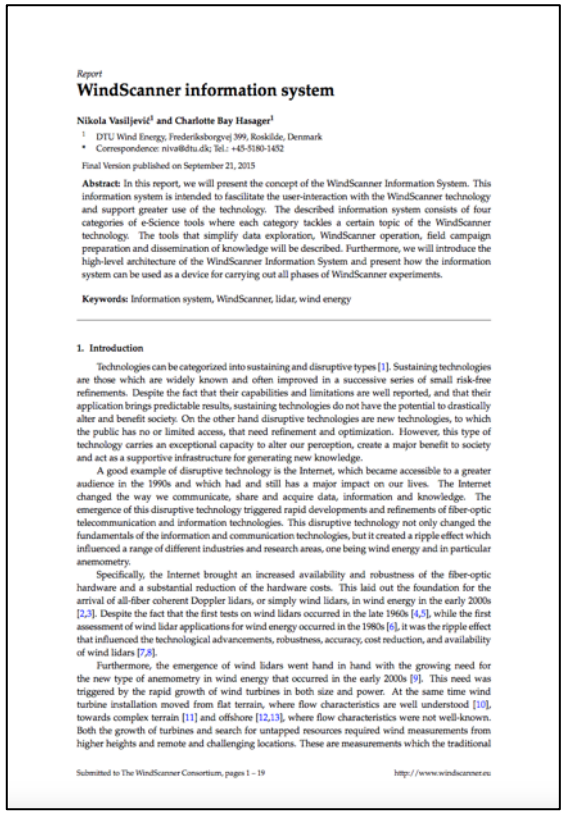
[1] https://zenodo.org/record/1175211

e-Science and user platform



Description

Link



Report

WindScanner information system

Nikola Vasiljević[1] and Charlotte Bay Hasager[1]

[1] DTU Wind Energy, Frederiksborgvej 399, Roskilde, Denmark

* Correspondence: niva@dtu.dk; Tel.: +45-5180-1452

Final Version published on September 21, 2015

Abstract: In this report, we will present the concept of the WindScanner Information System. This information system is intended to fascilitate the user-interaction with the WindScanner technology and support greater use of the technology. The described information system consists of four categories of e-Science tools where each category tackles a certain topic of the WindScanner technology. The tools that simplify data exploration, WindScanner operation, field campaign preparation and dissemination of knowledge will be described. Furthermore, we will introduce the high-level architecture of the WindScanner Information System and present how the information system can be used as a device for carrying out all phases of WindScanner experiments.

Keywords: Information system, WindScanner, lidar, wind energy

1. Introduction

Technologies can be categorized into sustaining and disruptive types [1]. Sustaining technologies are those which are widely known and often improved in a successive series of small risk-free refinements. Despite the fact that their capabilities and limitations are well reported, and that their application brings predictable results, sustaining technologies do not have the potential to drastically alter and benefit society. On the other hand disruptive technologies are new technologies, to which the public has no or limited access, that need refinement and optimization. However, this type of technology carries an exceptional capacity to alter our perception, create a major benefit to society and act as a supportive infrastructure for generating new knowledge.

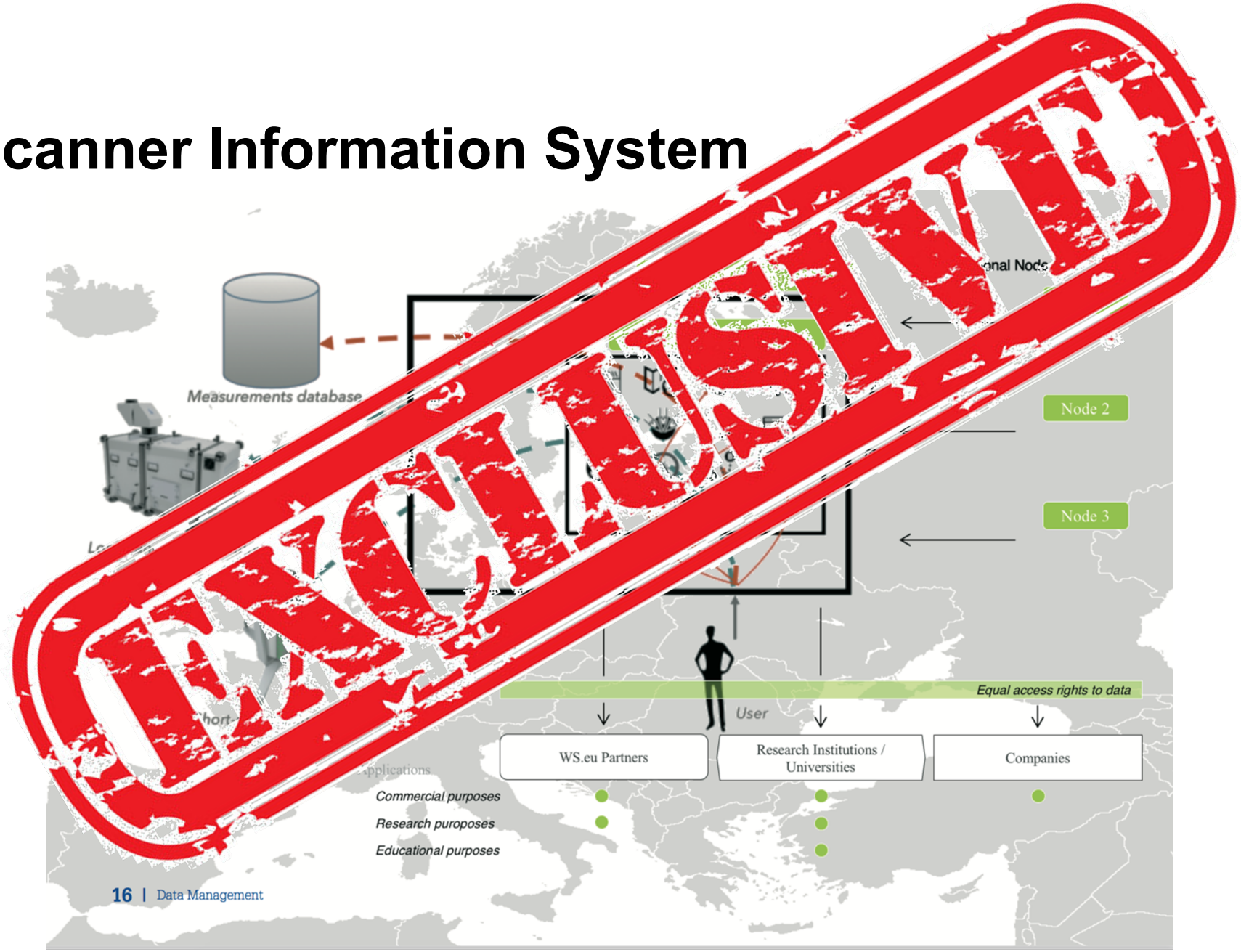
A good example of disruptive technology is the Internet, which became accessible to a greater audience in the 1990s and which had and still has a major impact on our lives. The Internet changed the way we communicate, share and acquire data, information and knowledge. The emergence of this disruptive technology triggered rapid developments and refinements of fiber-optic telecommunication and information technologies. This disruptive technology not only changed the fundamentals of the information and communication technologies, but it created a ripple effect which influenced a range of different industries and research areas, one being wind energy and in particular anemometry.

Specifically, the Internet brought an increased availability and robustness of the fiber-optic hardware and a substantial reduction of the hardware costs. This laid out the foundation for the arrival of all-fiber coherent Doppler lidars, or simply wind lidars, in wind energy in the early 2000s [2,3]. Despite the fact that the first tests on wind lidars occurred in the late 1960s [4,5], while the first assessment of wind lidar applications for wind energy occurred in the 1980s [6], it was the ripple effect that influenced the technological advancements, robustness, accuracy, cost reduction, and availability of wind lidars [7,8].

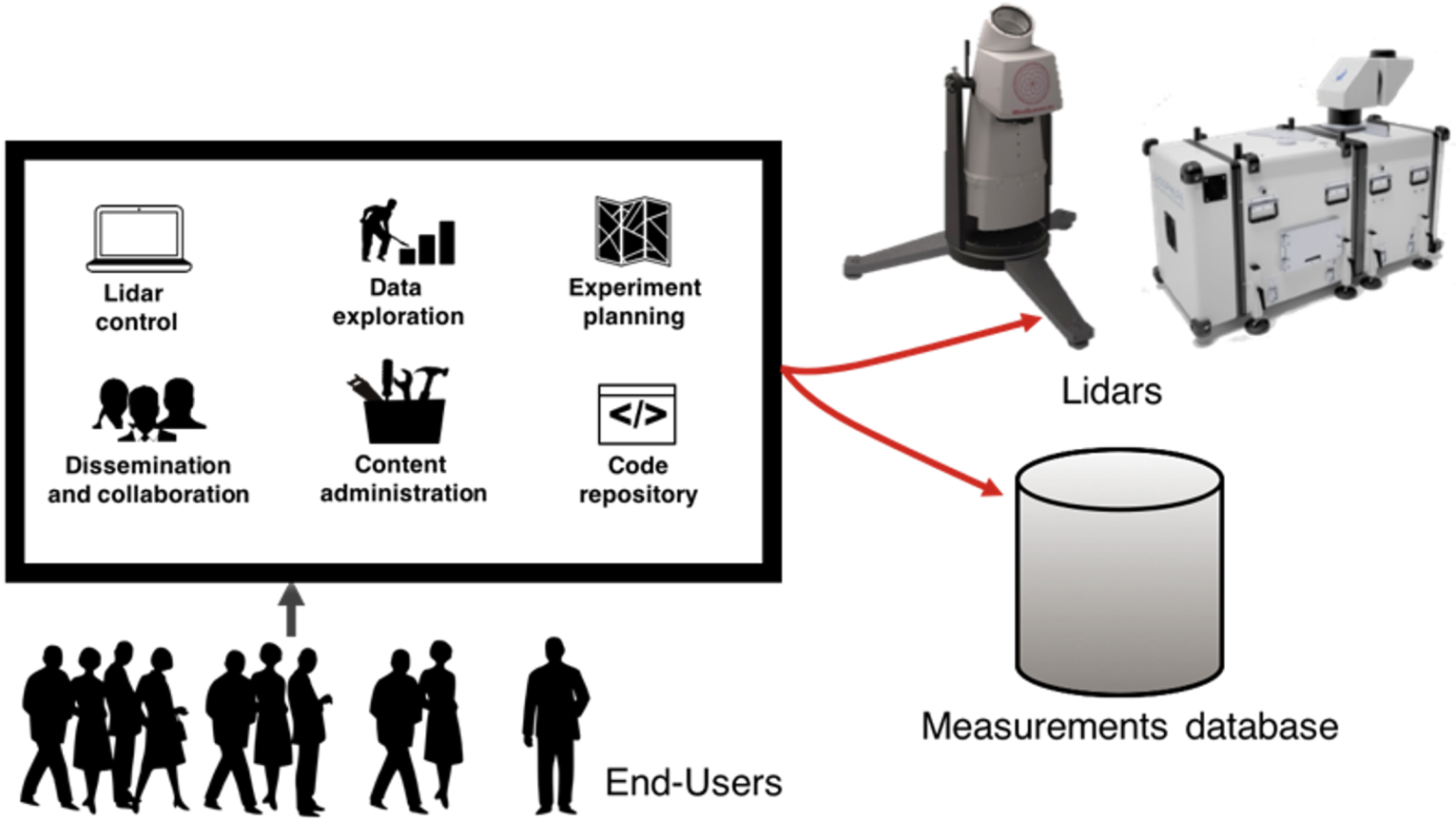
Furthermore, the emergence of wind lidars went hand in hand with the growing need for the new type of anemometry in wind energy that occurred in the early 2000s [9]. This need was triggered by the rapid growth of wind turbines in both size and power. At the same time wind turbine installation moved from flat terrain, where flow characteristics are well understood [10], towards complex terrain [11] and offshore [12,13], where flow characteristics were not well-known. Both the growth of turbines and search for untapped resources required wind measurements from higher heights and remote and challenging locations. These are measurements which the traditional

Link

WindScanner Information System



e-WindLidar toolbox



How a tool should look like?

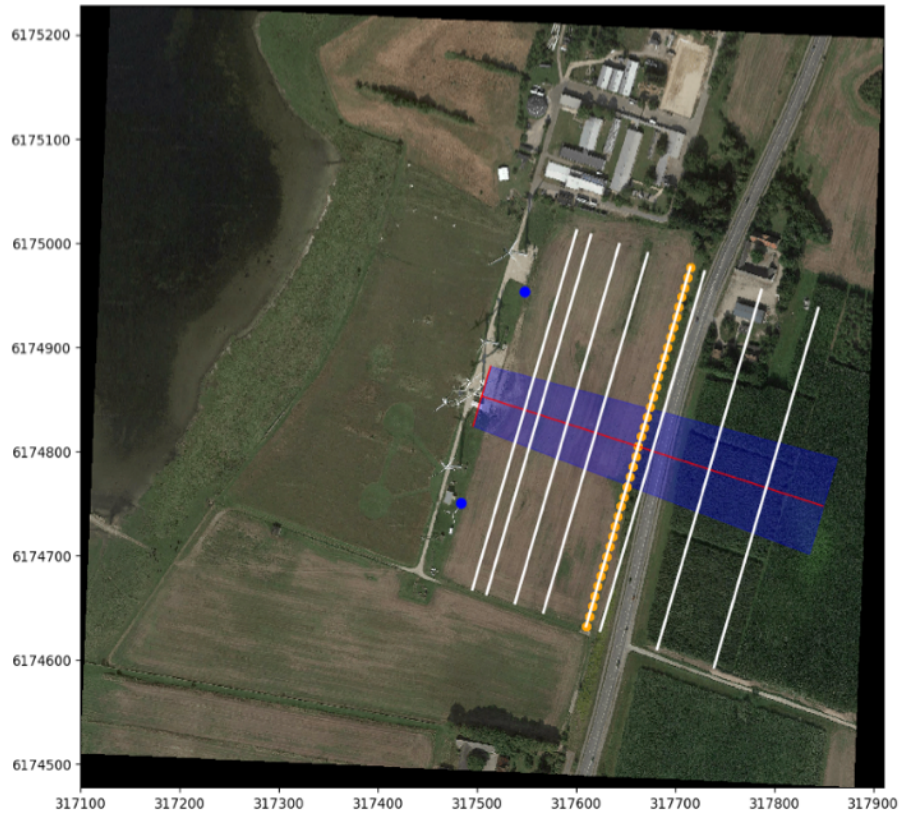
jupyter Experiment planning Last Checkpoint: 08/20/2018 (autosaved)

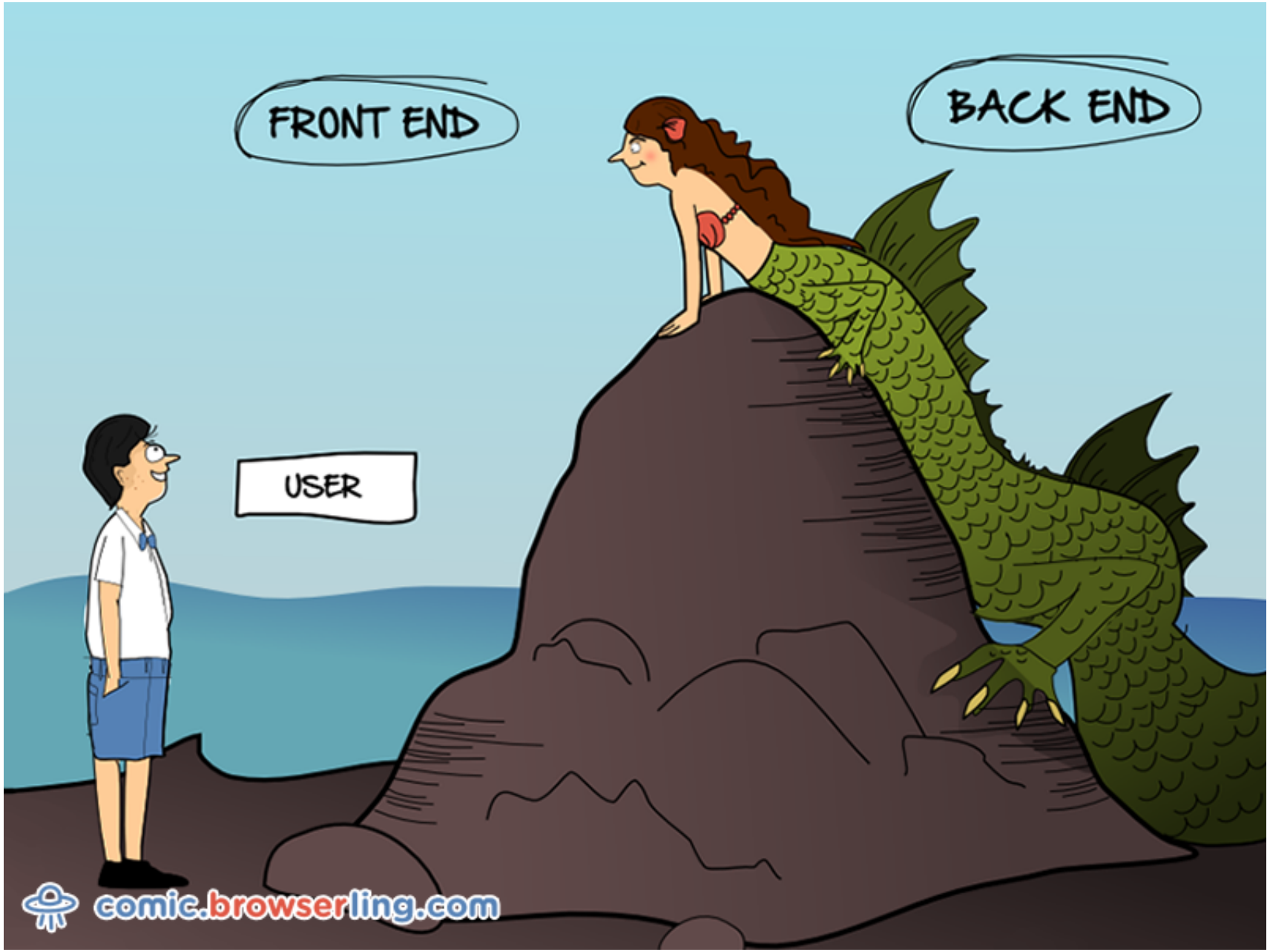
File Edit View Insert Cell Kernel Help

```
plain_of_focus = np.array([0,0])

plain_of_focus = scan_line_coordinates[4,0]
plain_points_XYZ = np.empty((0,3), int)
for height in range(20 + terrain_height, 90 + terrain_height + 1, 10):
    for i in range(0, half_length * 2 + 1, 10): #300 was previously 200
        temp = np.array(spher2cart(wind_dir_2 - 90,0,i)[:2])
        temp = plain_of_focus - temp
        plain_points_XYZ = np.append(plain_points_XYZ,
                                     np.array([[temp[0],temp[1], height]])
                                     ,axis = 0)
plt.scatter( plain_points_XYZ[:,0],plain_points_XYZ[:,1], marker='o', color='orange', s=30)

plt.axis('tight')
plt.show()
```




FRONT END
BACK END
USER
comic.browserling.com

campaign-planning-tool

Following slides are taken (and modified) from presentation :

Vasiljevic, Nikola, Vignaroli, Andrea, Bechmann, Andreas, & Wagner, Rozenn. (2019, June). ***Where to place WindScanners and where to measure with them?****. Zenodo.* *http://doi.org/10.5281/zenodo.3247797*

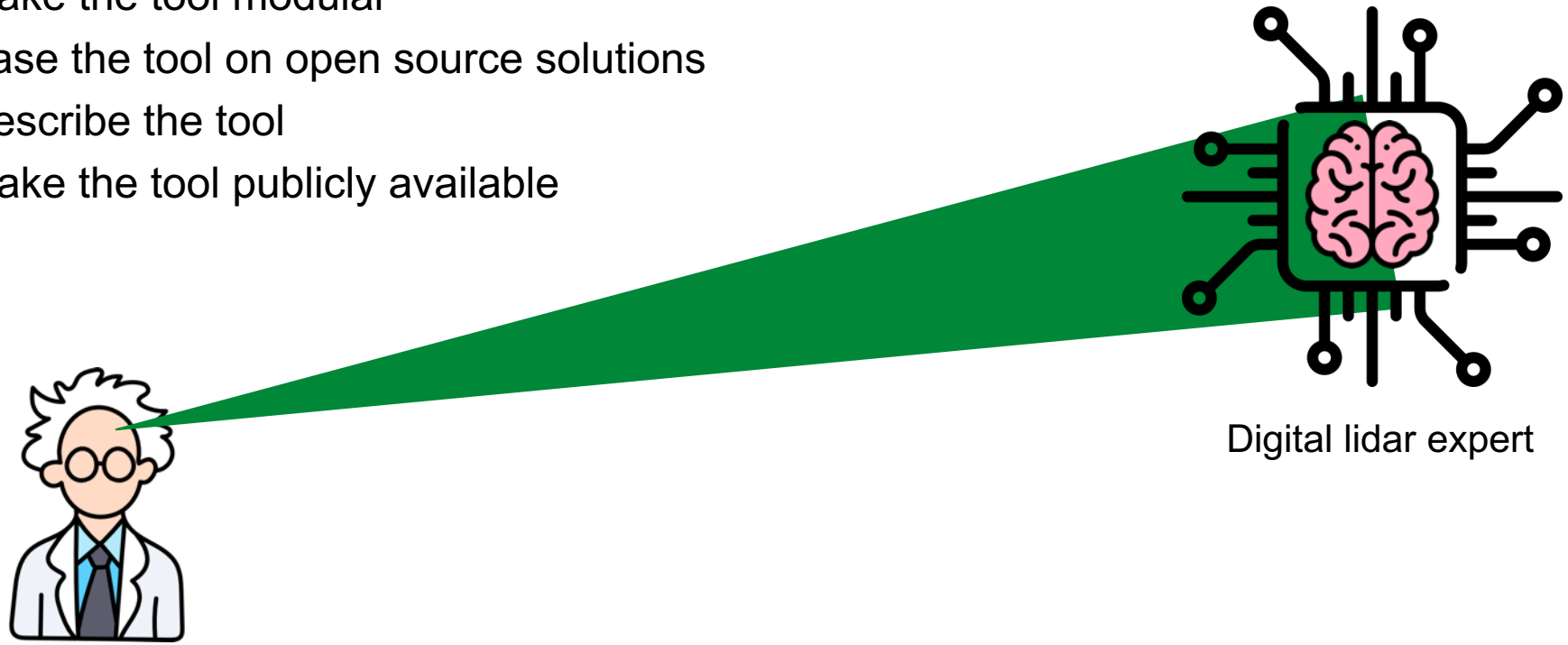
Full paper available:

Nikola Vasiljević, Andrea Vignaroli, Andreas Bechmann, and Rozenn Wagner. (2020 January). ***Digitalization of scanning lidar measurement campaign planning****. WES.*

https://doi.org/10.5194/wes-5-73-2020

campaign-planning-tool background

- Establish the campaign planning workflow
- **Digitalize** the workflow, thus create a tool
- Make the tool modular
- Base the tool on open source solutions
- Describe the tool
- Make the tool publicly available



Digital lidar expert

Physical lidar expert

Workflow Phases

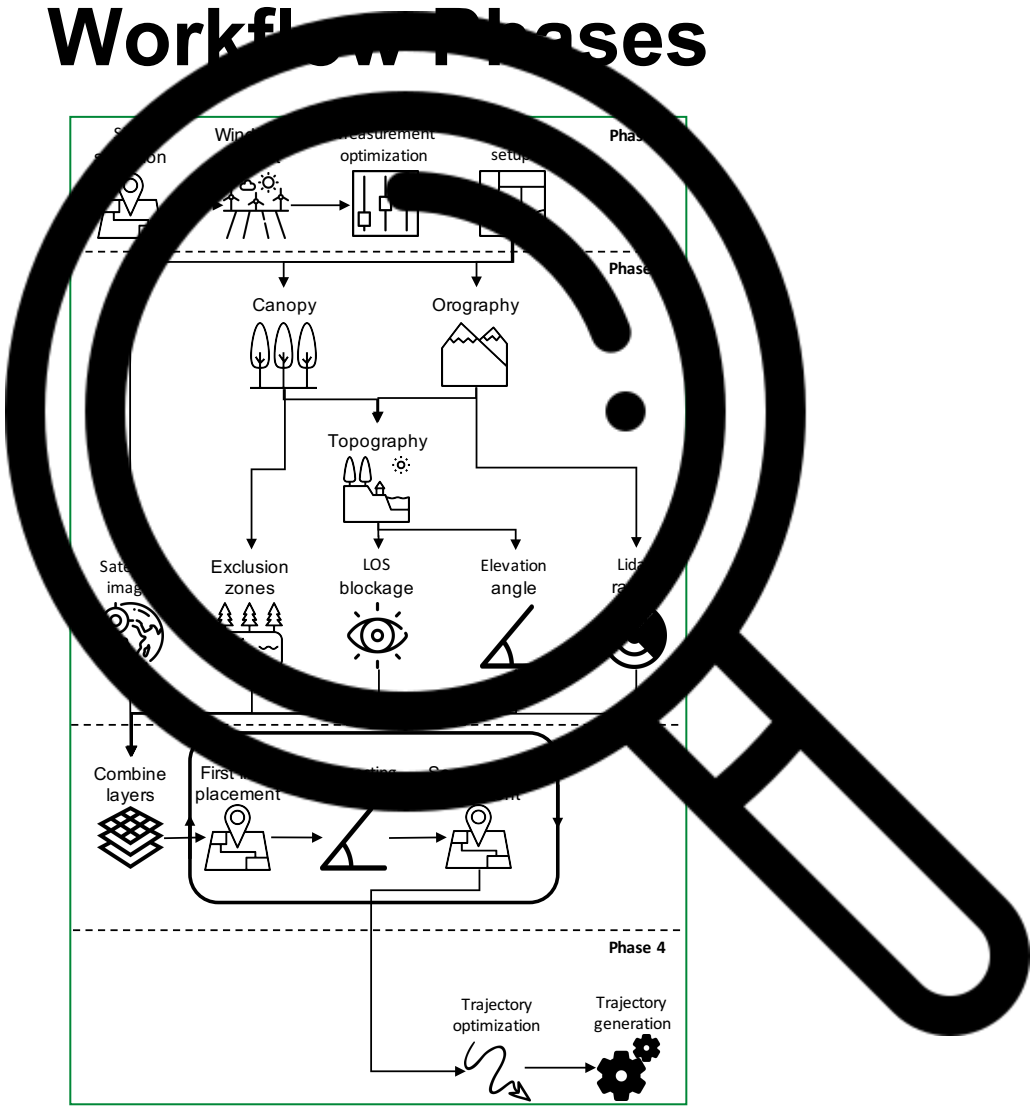
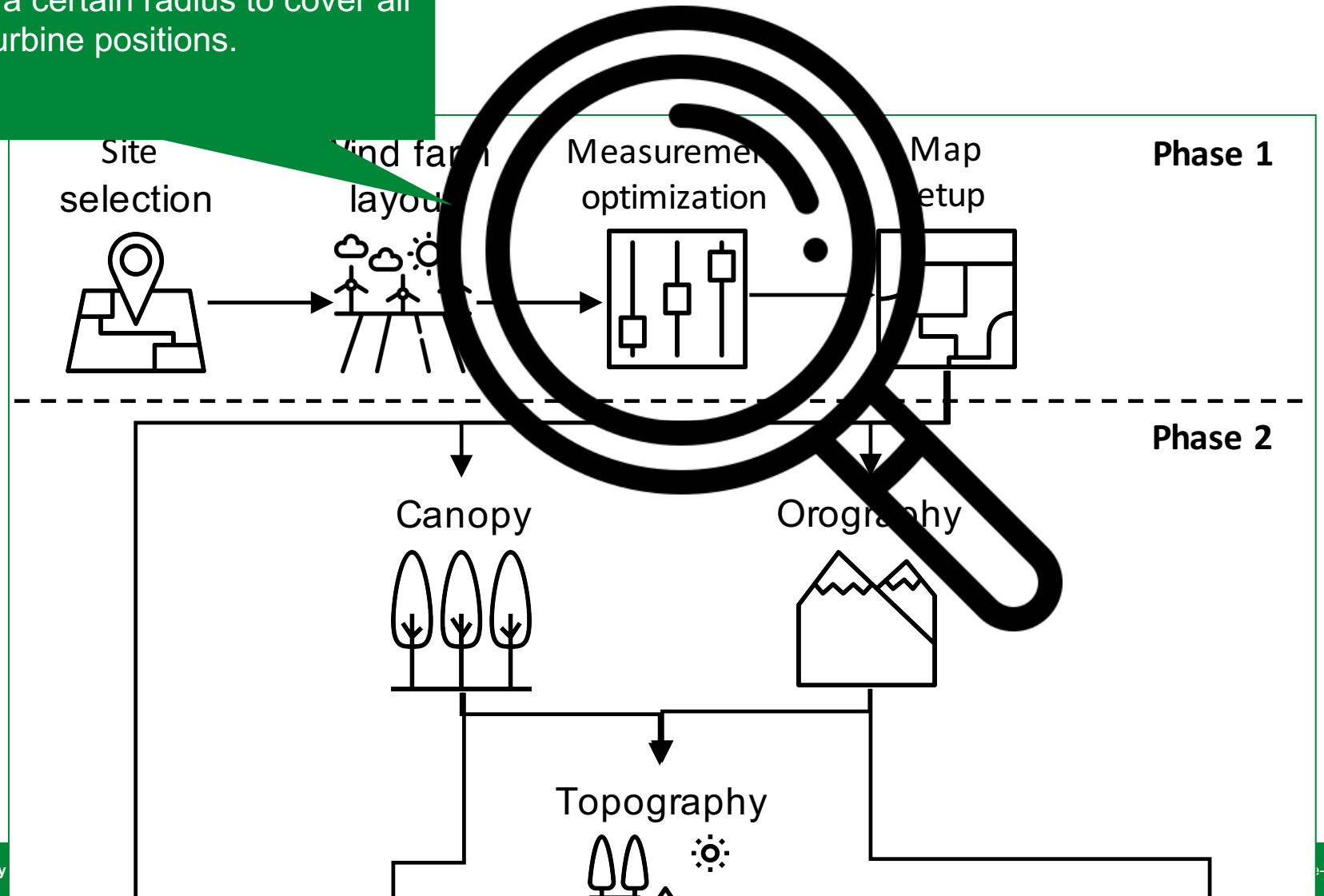
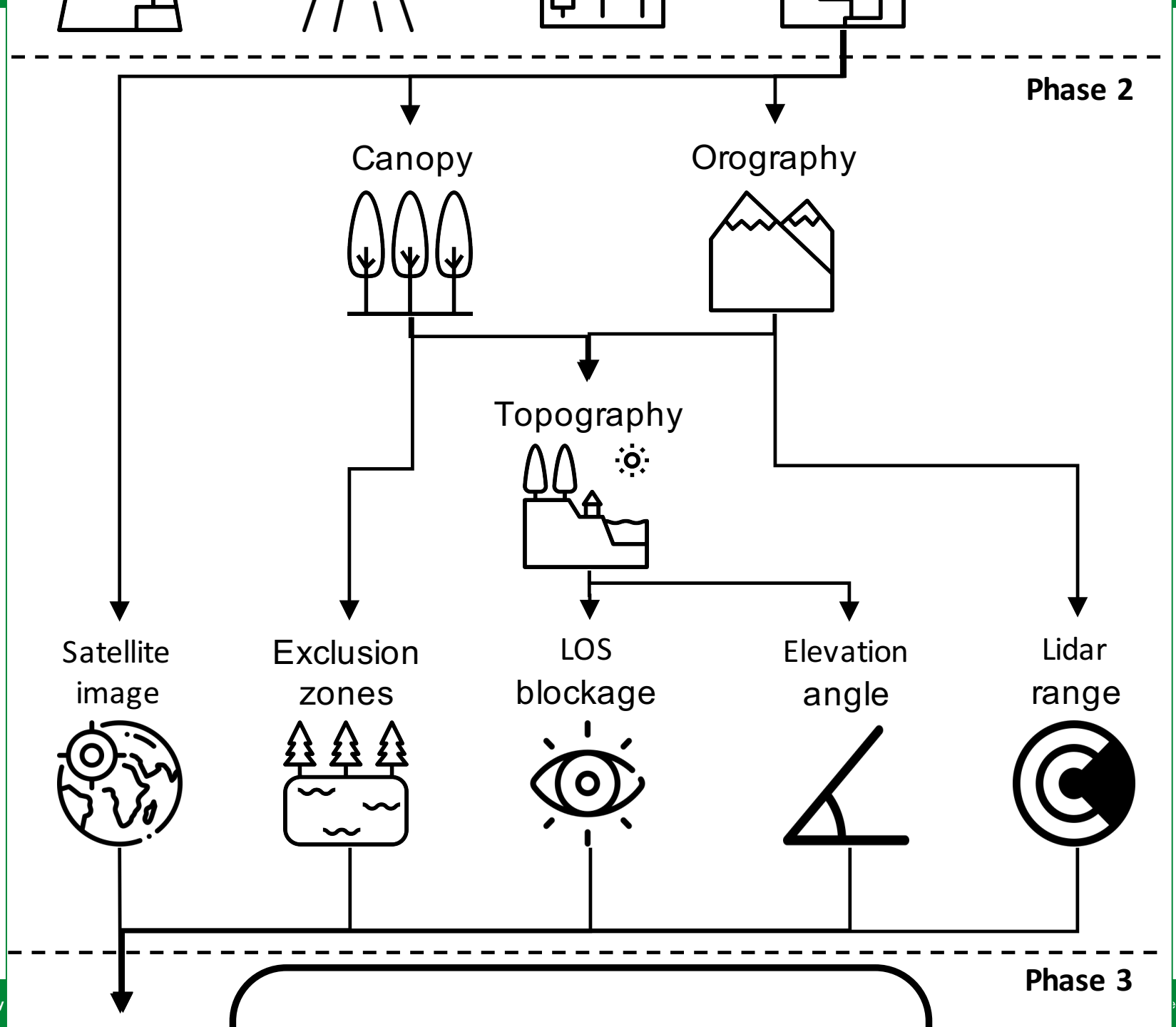


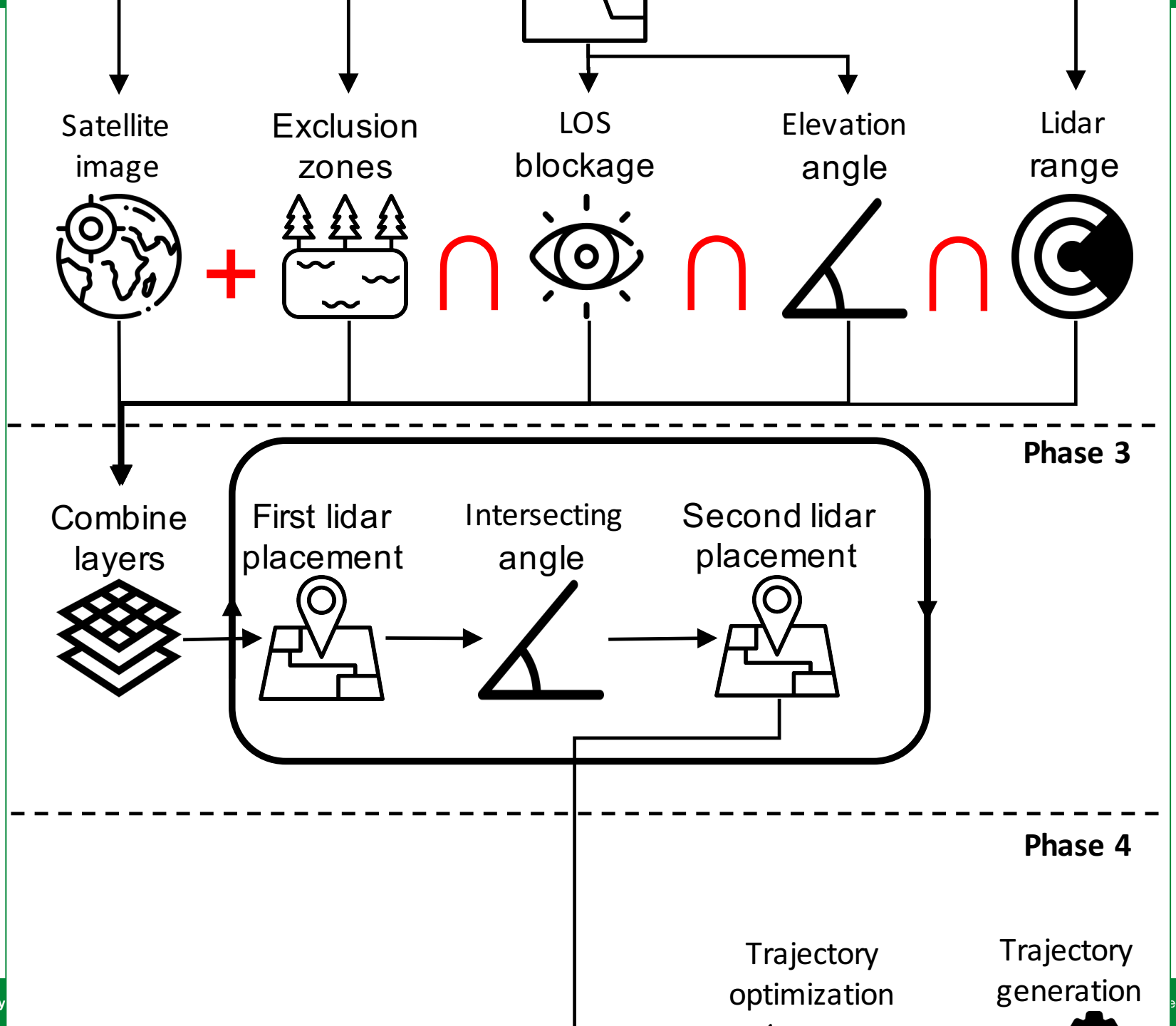
Image source: flaticon.com

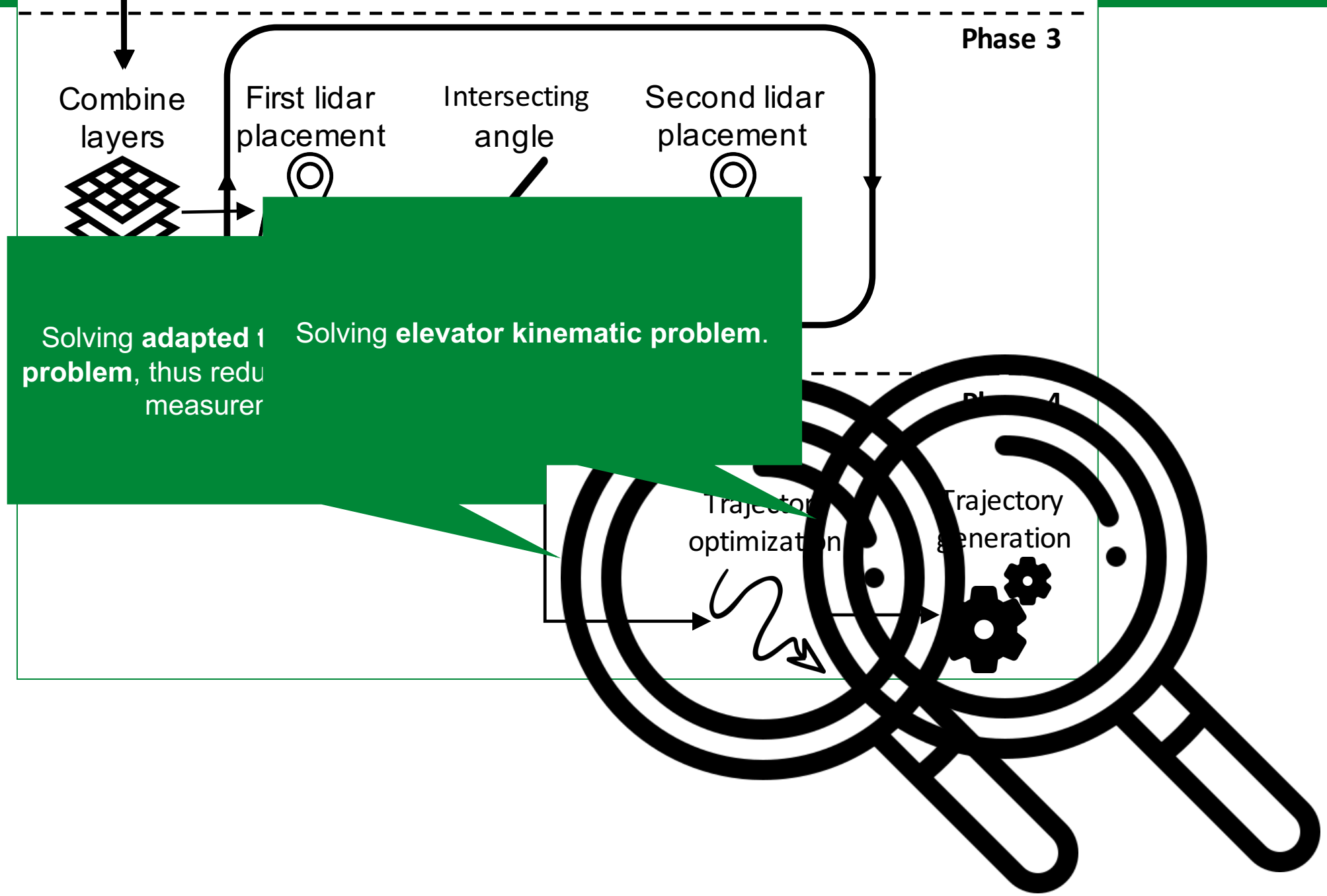
Solving **disc covering problem** in which we are searching for a minimum number of disks with a certain radius to cover all turbine positions.

Workflow Phases

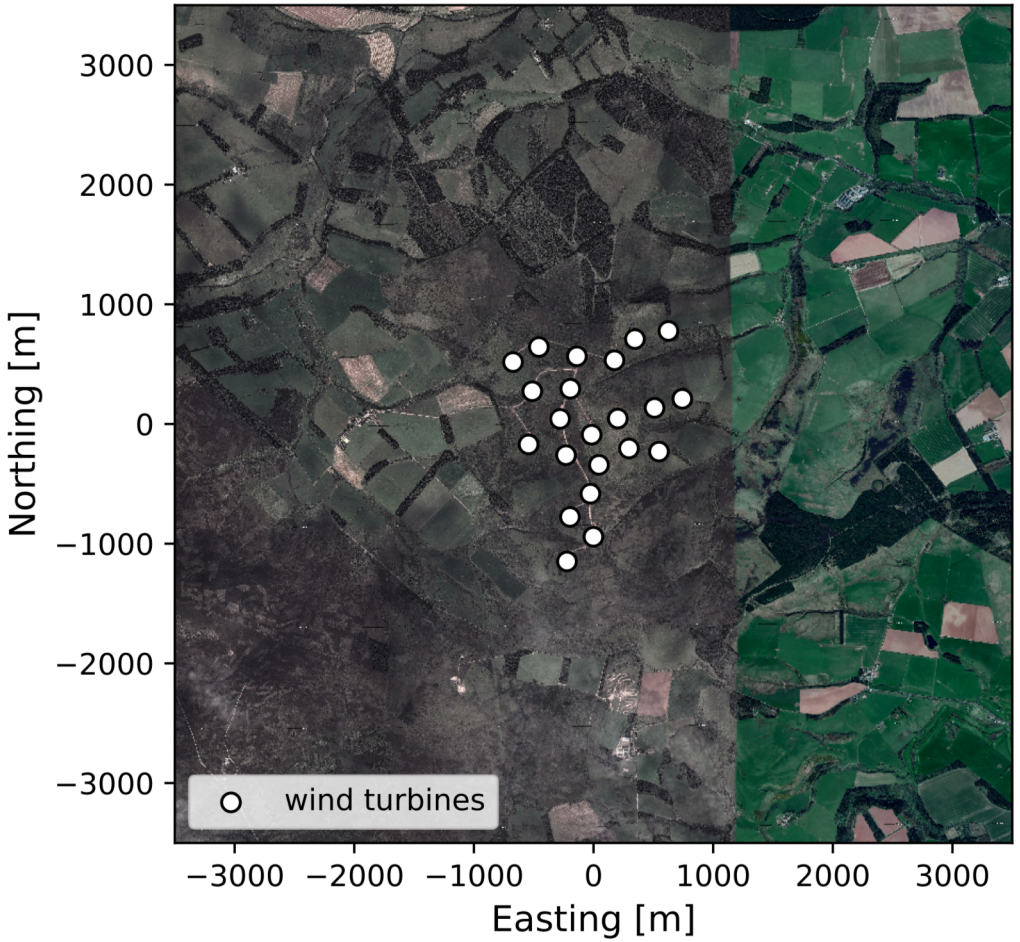



Phase 2
Canopy
Orography
Topography
Satellite image
Exclusion zones
LOS blockage
Elevation angle
Lidar range
Phase 3


Satellite image
Exclusion zones
LOS blockage
Elevation angle
Lidar range
Phase 3
Combine layers
First lidar placement
Intersecting angle
Second lidar placement
Phase 4
Trajectory optimization
Trajectory generation


Phase 3
Combine layers
First lidar placement
Intersecting angle
Second lidar placement
Solving **adapted** 1
problem, thus redu
measurer
Solving **elevator kinematic problem**.
Trajectory optimization
Trajectory generation

Example 1: Scottish site [2]

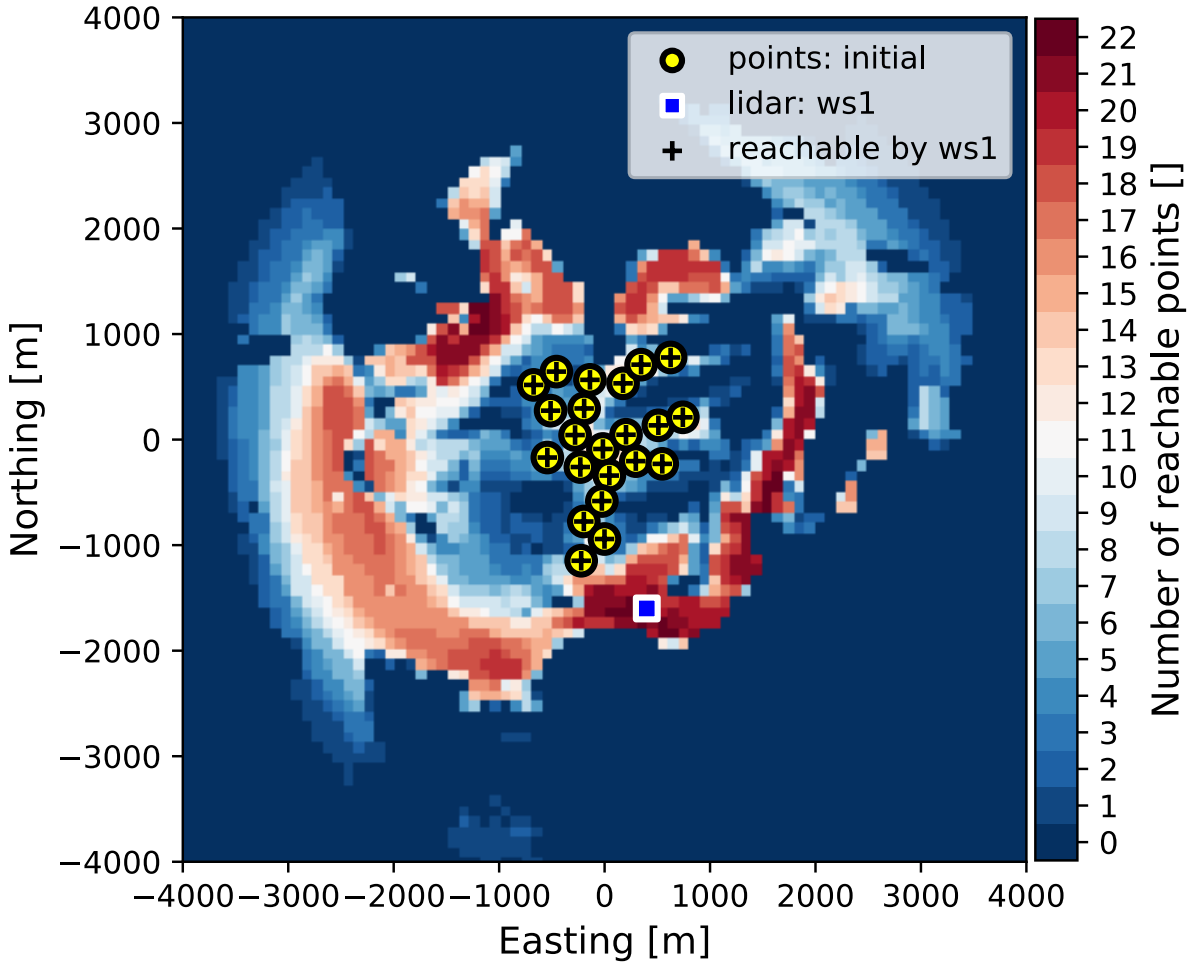


Info:

- 22 wind turbines with 47-m hub-heights
- Hilly terrain
- Aim to measure at each turbine position

[2] https://doi.org/10.11583/DTU.8343989.v3

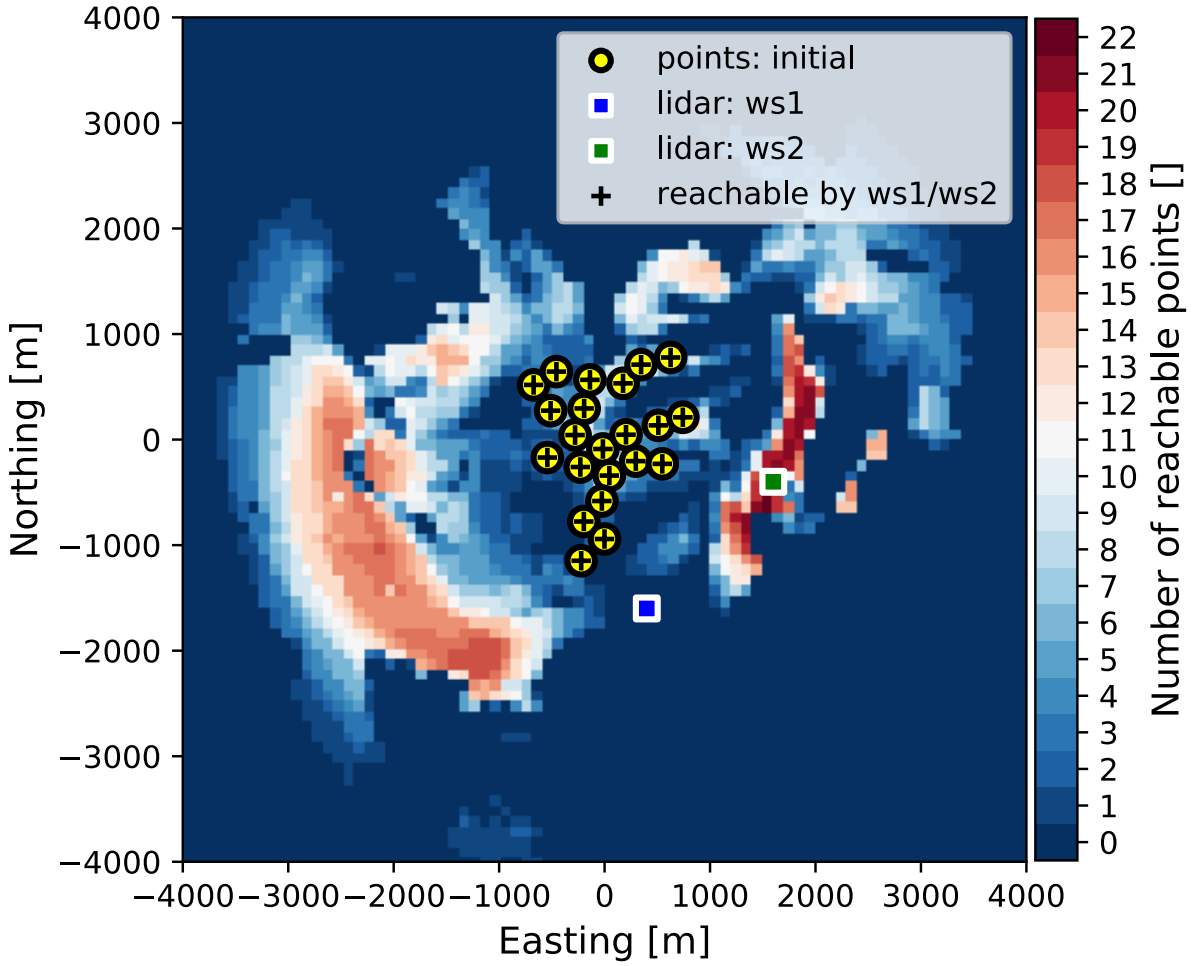
Example 1: Lidar placement GIS maps



Parameter	Value
Average range	3000 m
Max elevation angle	5˚
Min intersecting angle	30˚

Maps export to GeoTIFF / KML

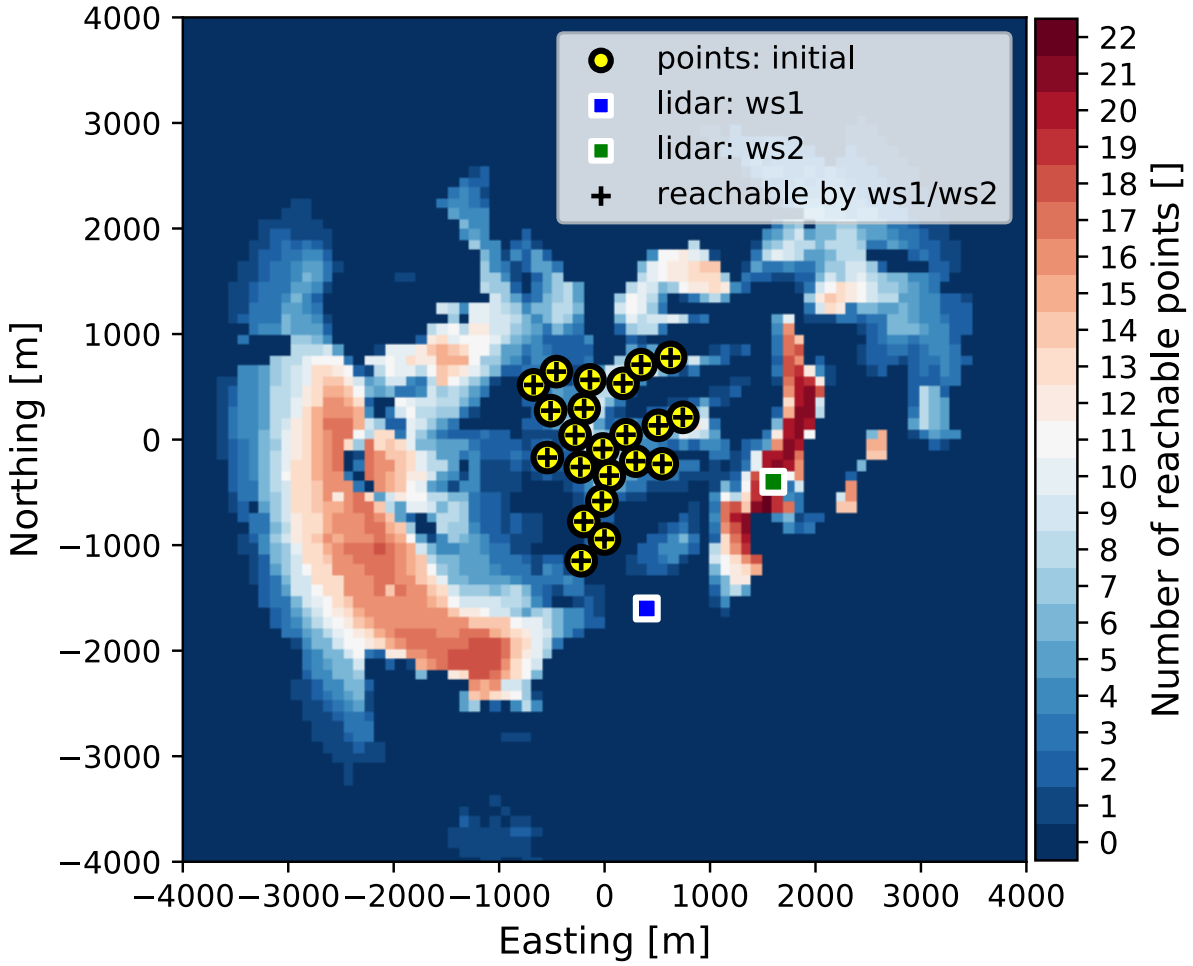
Example 1: Lidar placement GIS maps



Parameter	Value
Average range	3000 m
Max elevation angle	5˚
Min intersecting angle	30˚

Maps export to GeoTIFF / KML

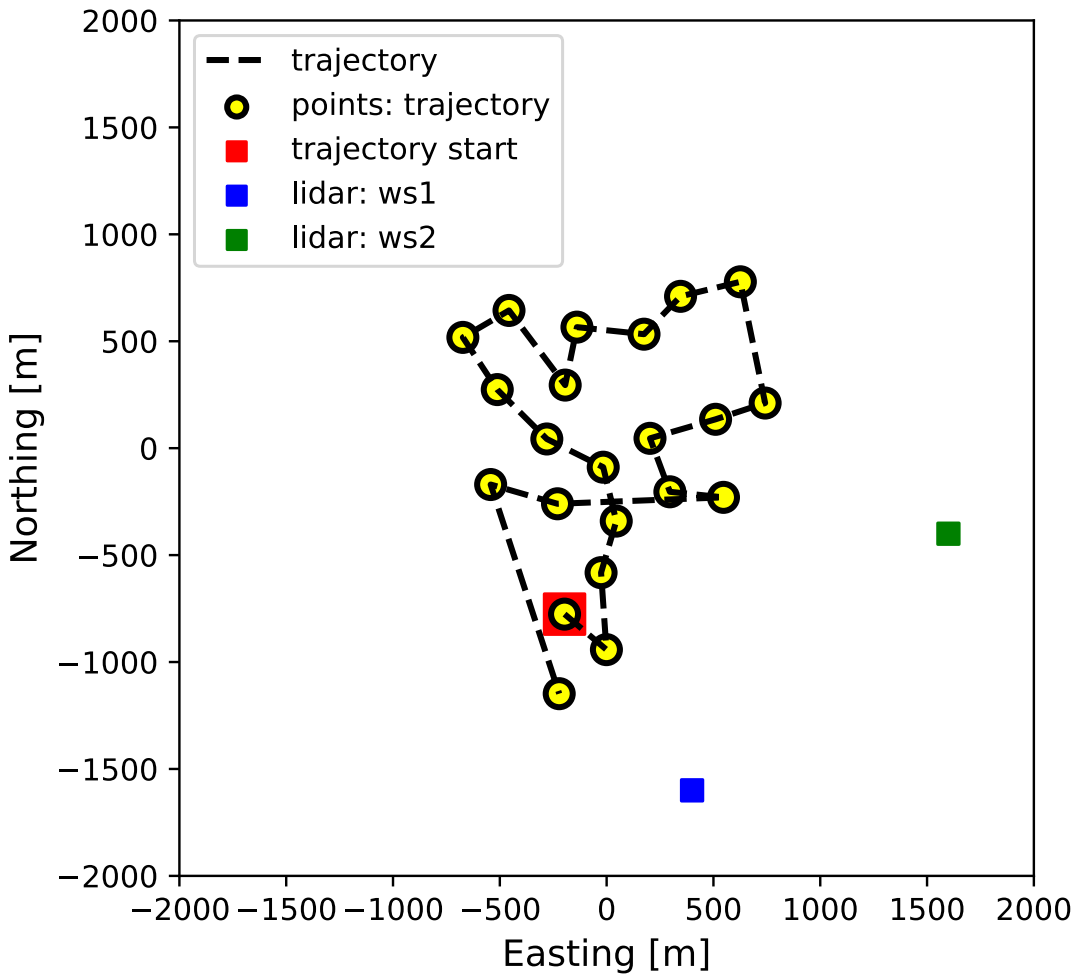
Example 1: Lidar placement GIS maps



Parameter	Value
Average range	3000 m
Max elevation angle	5˚
Min intersecting angle	30˚

Maps export to GeoTIFF / KML

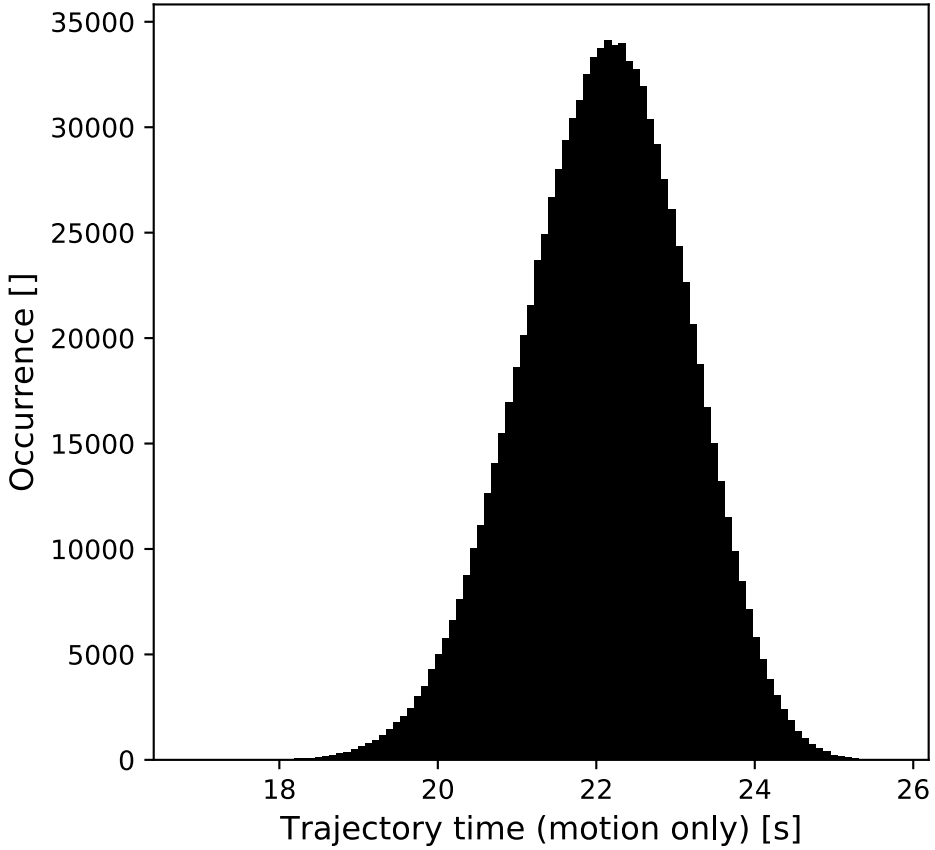
Example 1: Trajectory optimization and generation



Parameter	Value
Max speed	50˚/s
Max acceleration	100˚/s^2
Measurement time	22 s
Motion time	14 s
Trajectory time	36 s

Results export in multiple files

Example 1: How well the trajectory is optimized?

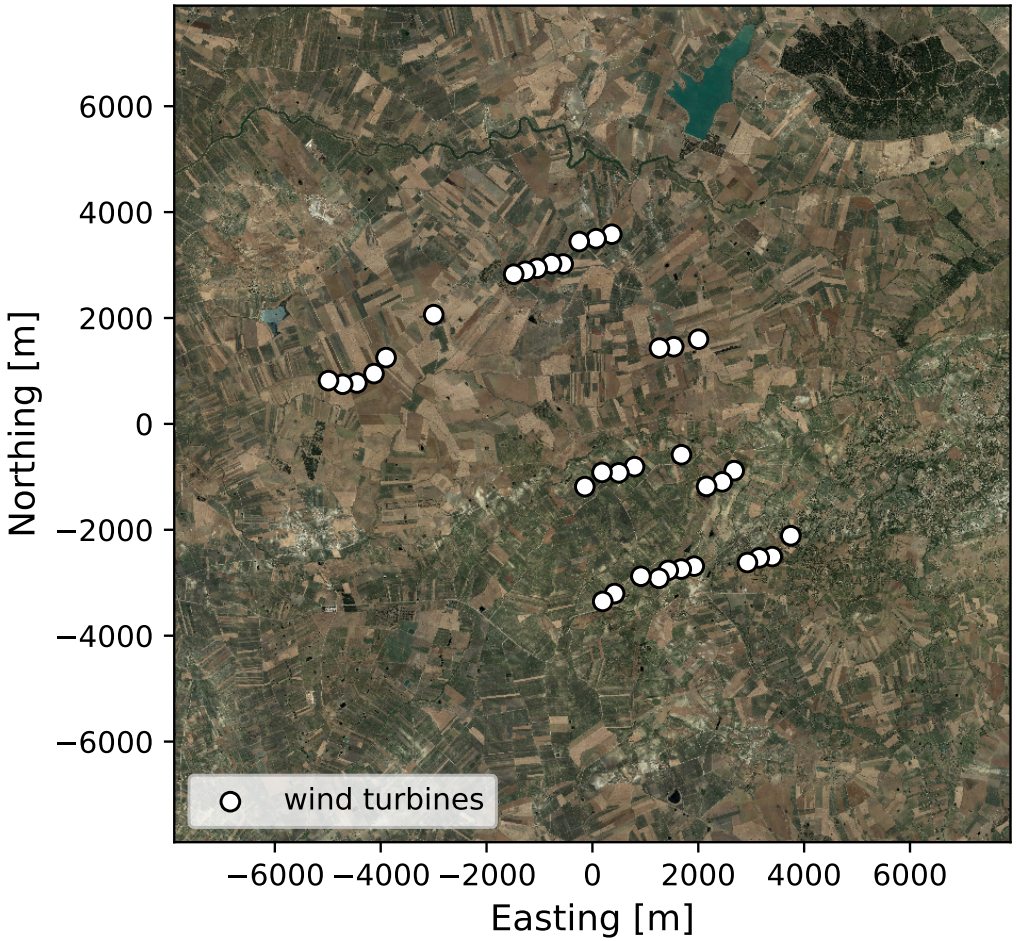


Parameter	Value
Mean motion time	22.09 s
Max motion time	25.76 s
Min motion time	17.15 s
Std motion time	1.03 s

A histogram of motion time for **10^6** randomly generated trajectory configurations for the Scottish site.

On average the optimized trajectory is 8 s shorter in duration!

Example 2: Italian site [3]

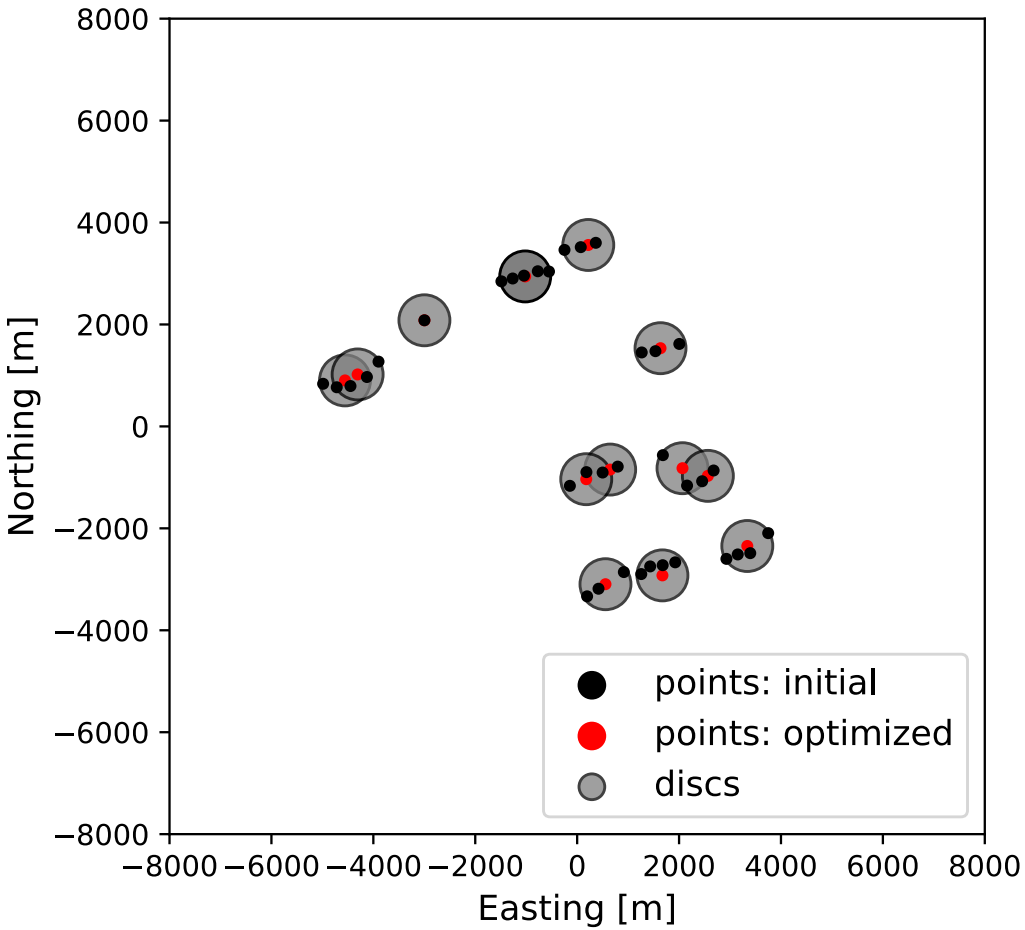


Info:

- 36 wind turbines with 78-m hub-heights
- Hilly terrain
- We will use measurement point optimization

[3] https://doi.org/10.11583/DTU.8343989.v3

Example 2: Measurement point optimization

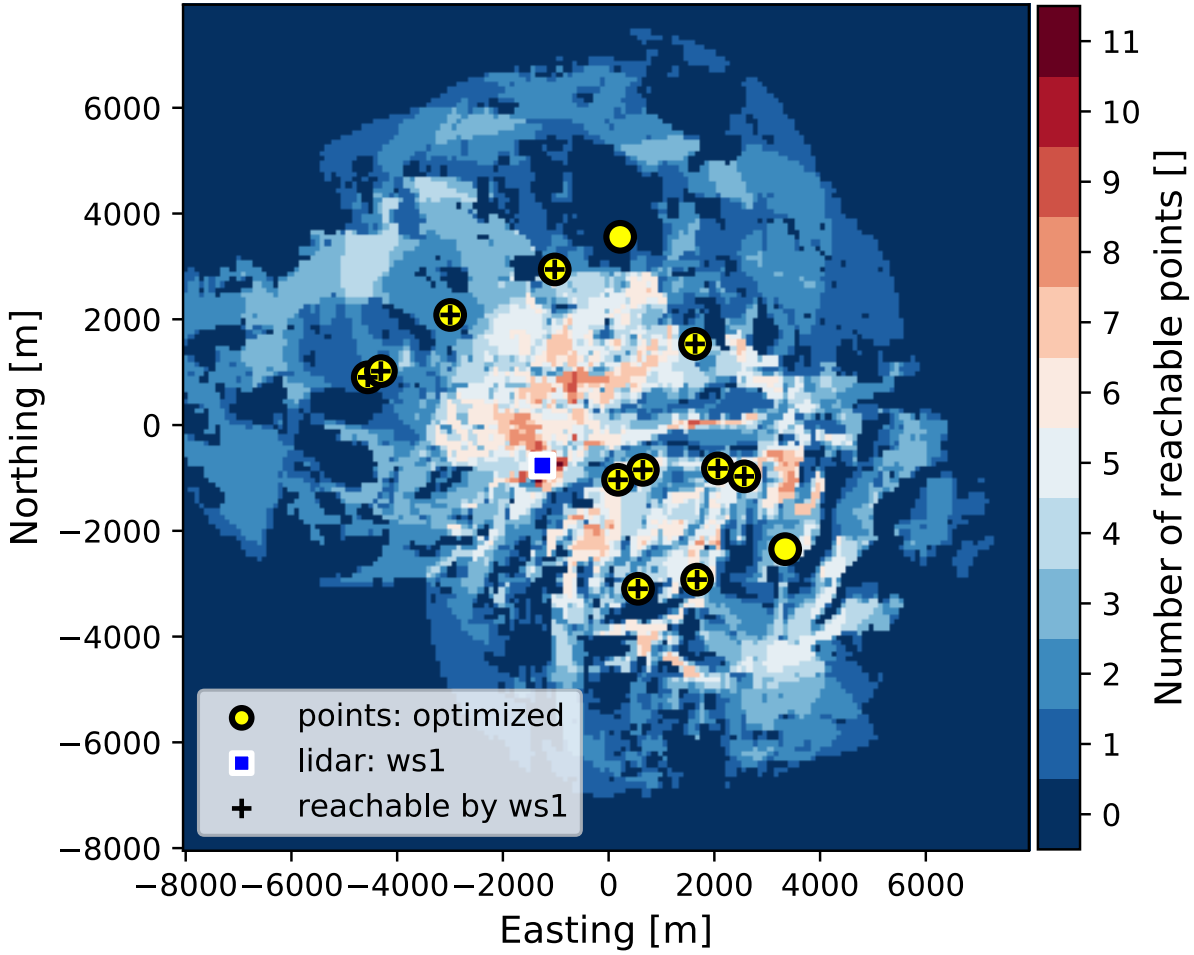


Parameter	Value
Representativeness radius [12]	500 m
No initial points	36
No optimized points	**13**

Reduction by 64%

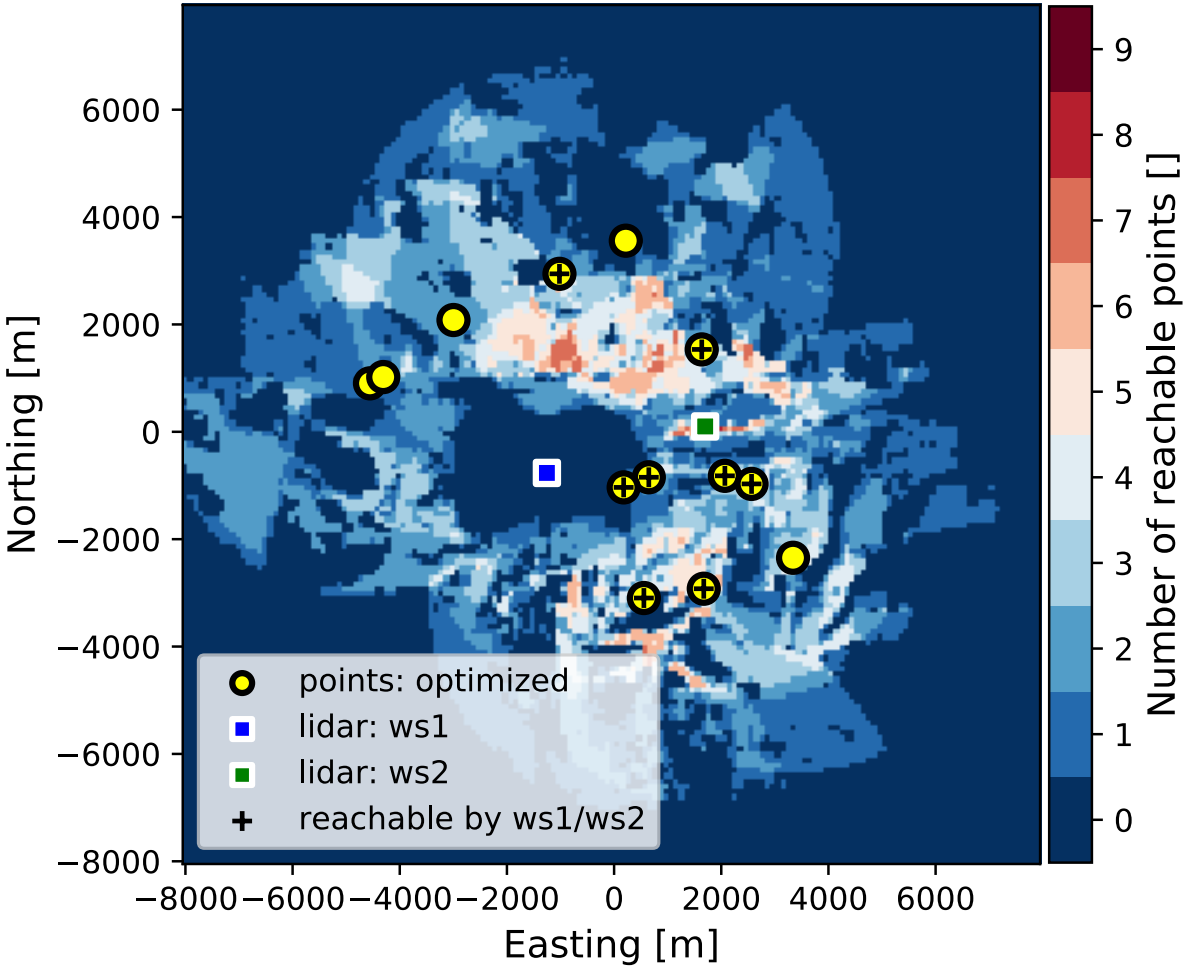
[4] MEASNET Procedure: Evaluation of Site-Specific Wind Conditions

Example 2: Lidar placement GIS maps



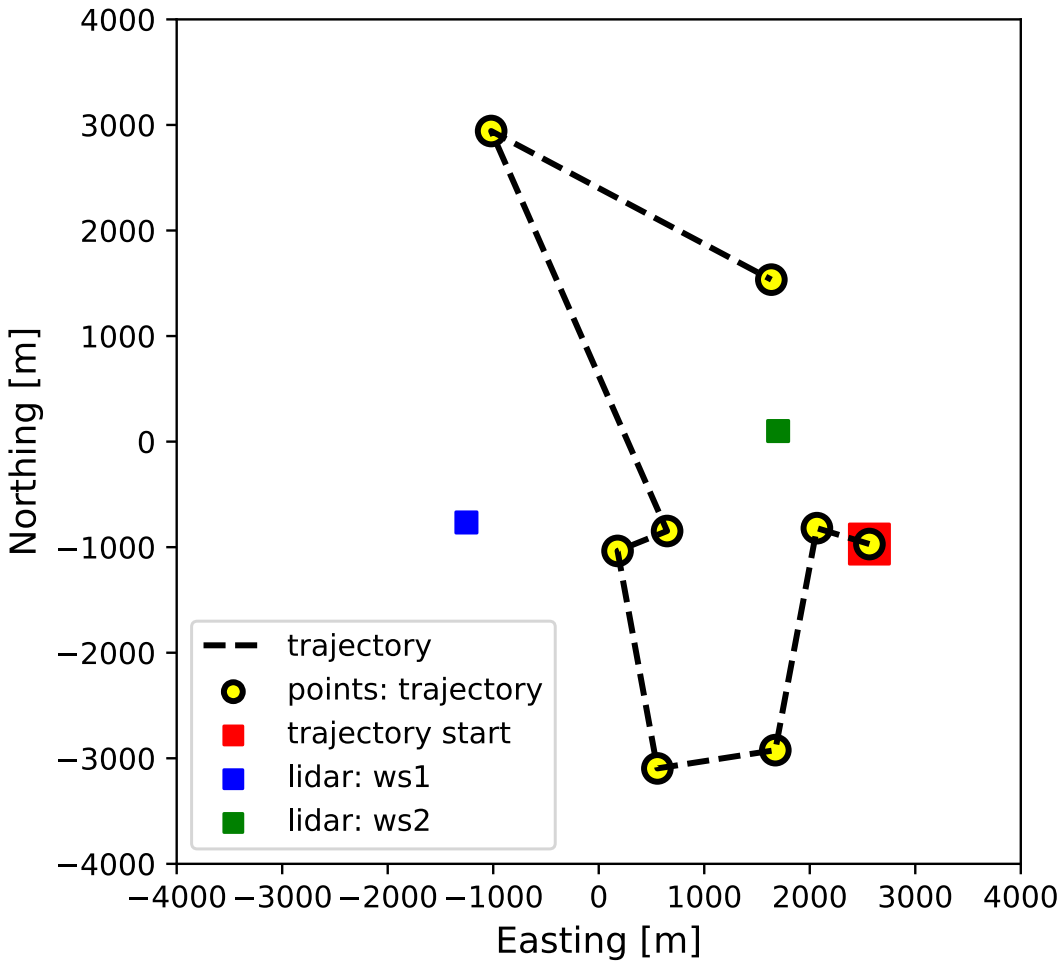
Parameter	Value
Average range	4000 m
Max elevation angle	5˚
Min intersecting angle	30˚

Example 2: Lidar placement GIS maps



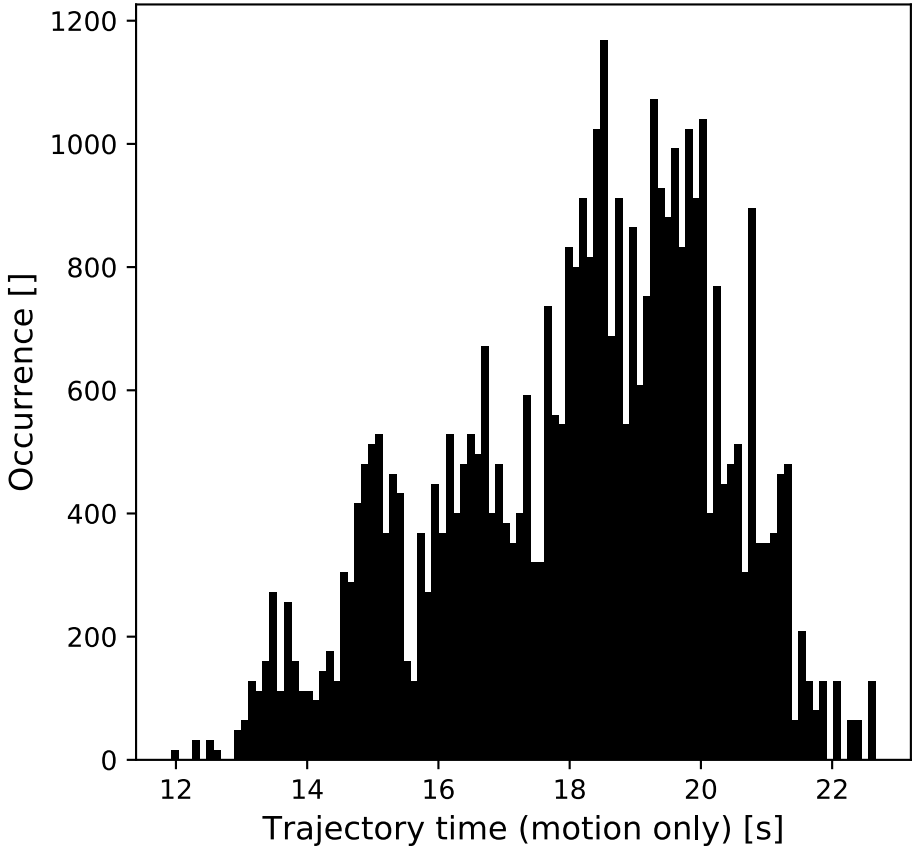
Parameter	Value
Average range	4000 m
Max elevation angle	5˚
Min intersecting angle	30˚

Example 2: Trajectory optimization and generation



Parameter	Value
Max speed	50˚/s
Max acceleration	100˚/s^2
Measurement time	8 s
Motion time	13 s
Trajectory time	21 s

Example 1: How well the trajectory is optimized?



Parameter	Value
Mean motion time	18.14 s
Max motion time	22.66 s
Min motion time	11.93 s
Std motion time	2.10 s

A histogram of motion time for 40320 unique trajectory configurations for the Italian site.

On average the optimized trajectory is 5 s shorter in duration!

Summary

- It takes **couple of minutes** to design and configure scanning lidar campaigns using *campaign-planning-tool* even for non-lidar experts, opposite to probably **days** if this is done manually by lidar experts

- The actual computational time to run *campaign-planning-tool* takes about **30 s**

- Optimizing measurement points can (depending on layout) reduce significantly number of measurement points => boost measurement rate

- Trajectory optimization matters since it can shed some seconds per each scan => boost measurement rate

YADDUM

Following slides are taken (and modified) from presentation :

Vasiljevic, Nikola. (2019, November). ***YADDUM - Yet Another Dual-Doppler Uncertainty Model****. Zenodo.* *http://doi.org/10.5281/zenodo.3551577*

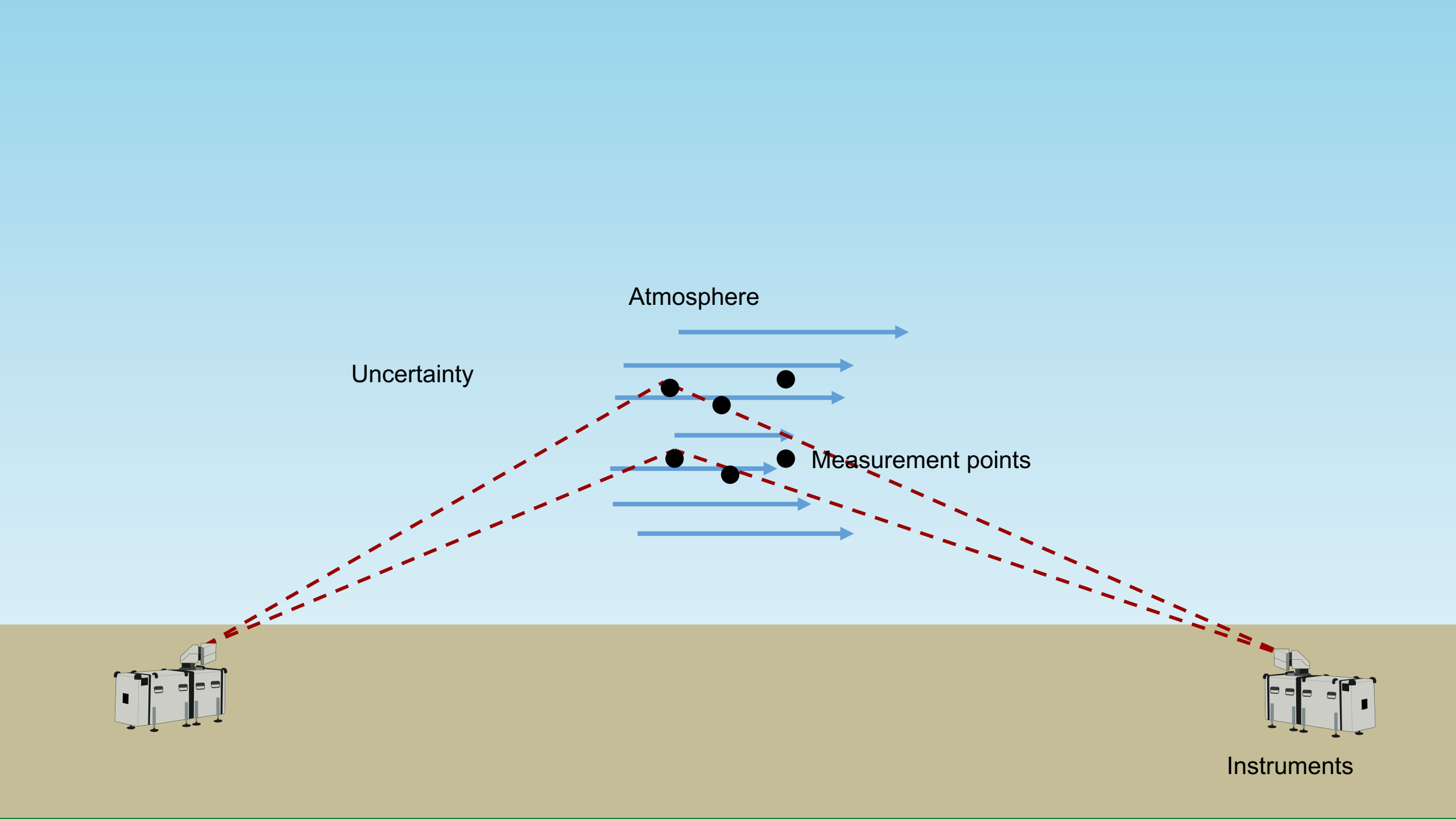
Background

- It is using an analytical model derived in 2017:
  Nikola Vasiljevic, & Michael Courtney. (2017, March). ***Accuracy of dual-Doppler lidar retrievals of near-shore winds***. Zenodo. http://doi.org/10.5281/zenodo.1441178

- The model is based on law of propagation of uncertainties

- Simple and fast uncertainty assessment of dual-Doppler (two lidars or radars) retrievals of wind speed and wind direction

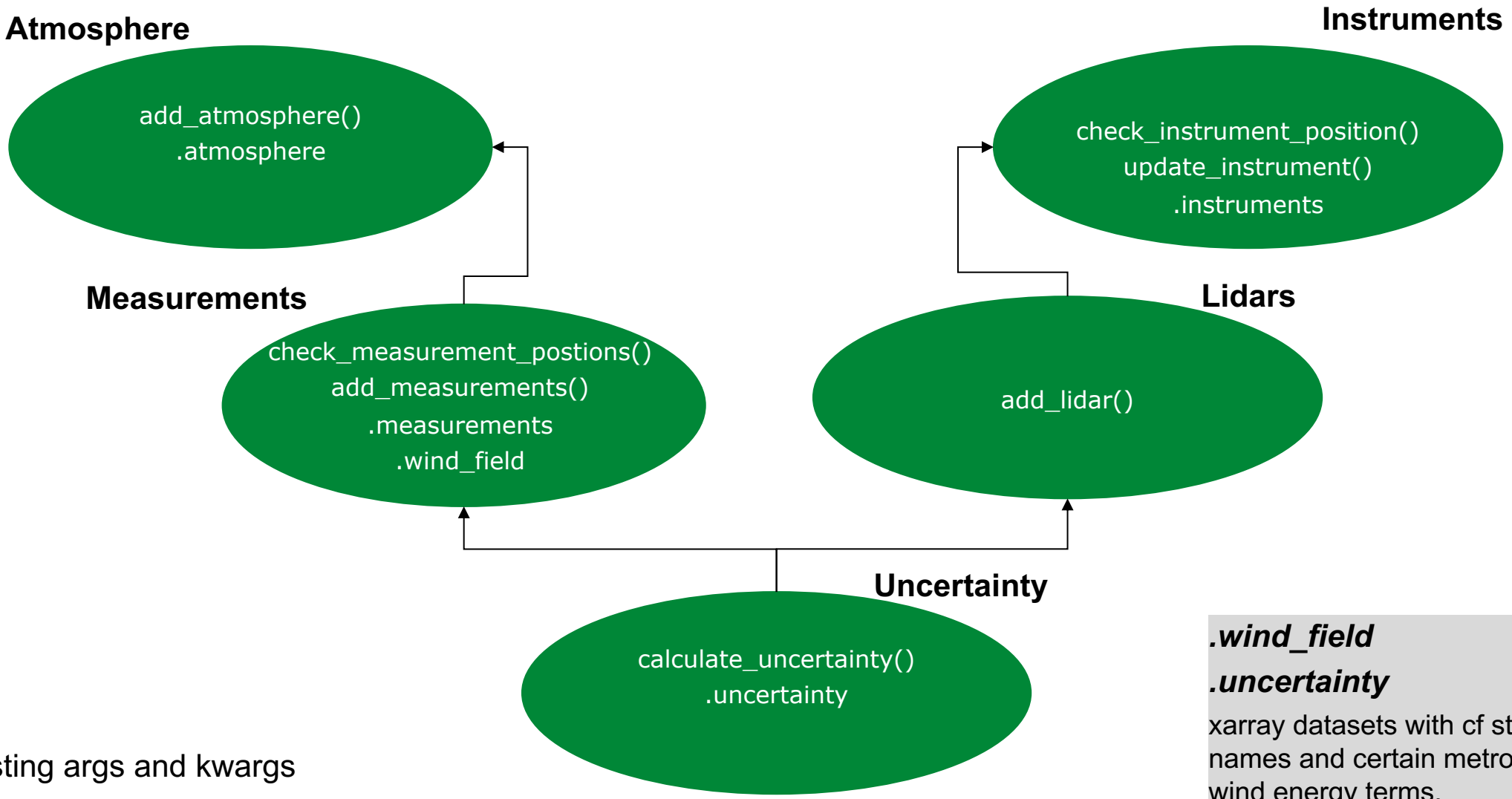
- Application: Dual-Doppler setups operating with low elevation angles and especially offshore


Atmosphere
Uncertainty
Measurement points
Instruments

Basic entities

1. Atmosphere
2. Measurement point(s)
3. Instrument(s)
4. Uncertainty

Classes and their relations



.wind_field

.uncertainty

xarray datasets with cf standard names and certain metrological and wind energy terms.

- Not listing args and kwargs
- Not listing private methods and attributes

Workflow

1. `add_atmosphere(atmosphere_id, model, model_parameters)`
   – *Currently the method support only wind power law profile as the 'atmospheric model'*

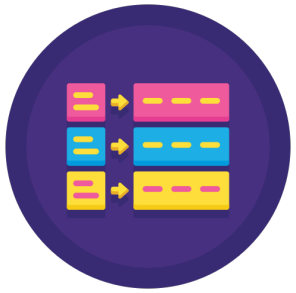
2. `add_measurements(measurements_id, category, utm_zone, **kwargs)`
   – *Currently the method support adding either arbitrary set of points or 2D mesh of points*

3. `add_lidar(instrument_id, position, category, **kwargs)`
   – *Currently the method allow addition of single lidar per time*

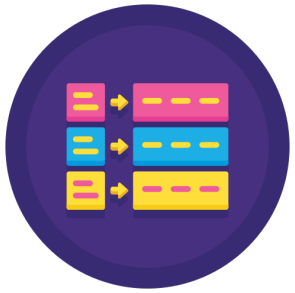
4. `calculate_uncertainty(instrument_ids, measurements_id, atmosphere_id, model)`
   – *Currently radial speed uncertainty and dual-Doppler uncertainty models are supported*

Dictionaries and xarray objects

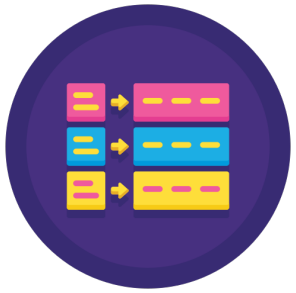
1. `add_atmosphere()`, `add_measurements()` and `add_lidar()` create three dictionaries:



atmosphere



measurements



instruments

2. `calculate_uncertainty()` creates two xarray datasets:

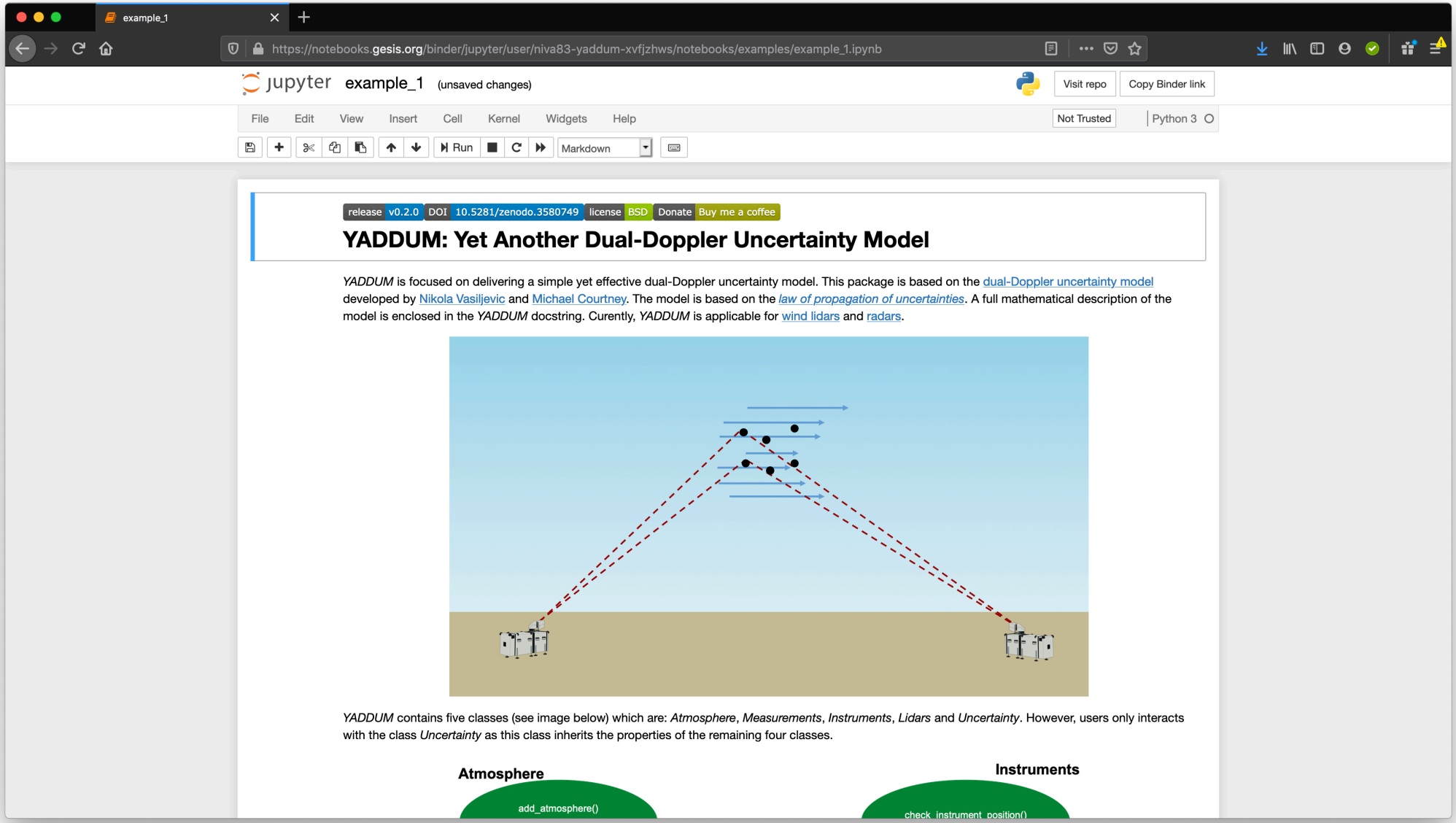
wind_field

uncertainty

icons source: flaticon.com

YADDUM is …

- Using only `xarray` and `numpy` packages
- Fully documented (numpy docstring)
- Versioned on Github
- Persisted on Zenodo
- Available as a conda package through Anaconda cloud
- Or on binder for direct manipulation without any installation



Try it your self < this is clickable

W k beyond 2019…

1. `add_atmo  ()`
   – *Extend catalogu ospheric models*

2. `add_measurements()`
   – *Extend capabilities to LOS, RHI, P DBS structures of points*

3. `Instruments`
   – *Support other instrum class (e.g., radar)*

4. `calculat  ertainty()`
   – *Su er uncertainty models (single- and triple- Doppler, Monte-Ca ulations, etc.)*

mocalum

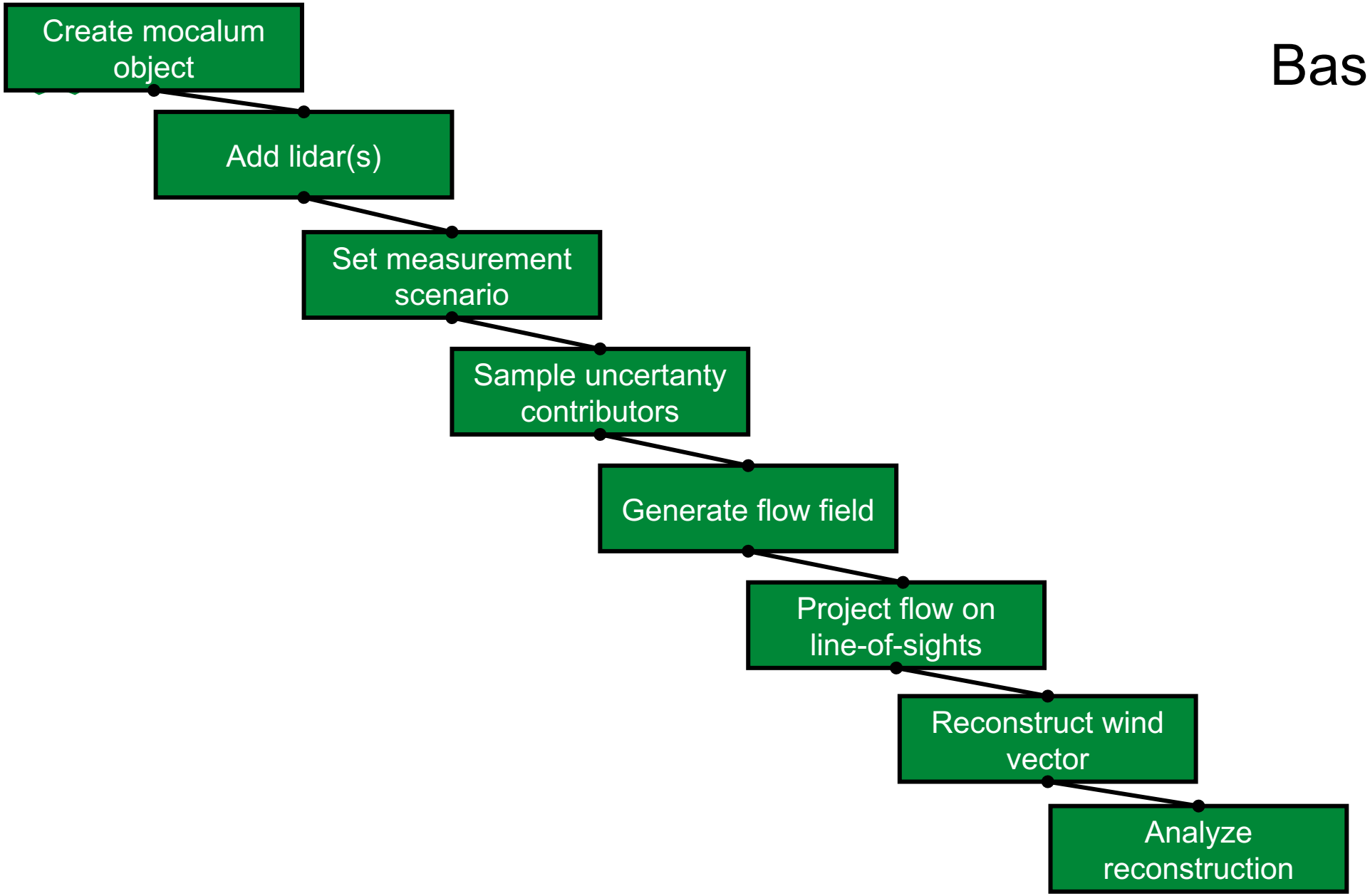
Following slides are taken (and modified) from presentation :

Vasiljevic, Nikola. (2020, May). ***mocalum: python package for Monte-Carlo lidar uncertainty modeling****. Zenodo.* http://doi.org/10.5281/zenodo.3823878

Features

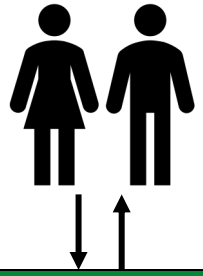
- Fast Monte Carlo uncertainty modeling
- Simulation of single or multi lidar configuration
- Configuration of arbitrary trajectories for single and multi lidars
- Configuration of `IVAP` (sector-scan) trajectory for single lidar
- 3D or 4D / uniform or turbulent flow field generation
- Sampling of correlated or uncorrelated uncertainty terms
- Built-in 2nd order kinematic model for calculation trajectory timing
- 3D or 4D interpolation/projection of flow on lidar(s) LOS(s)
- xarray datasets enriched with metadata

Basic workflow

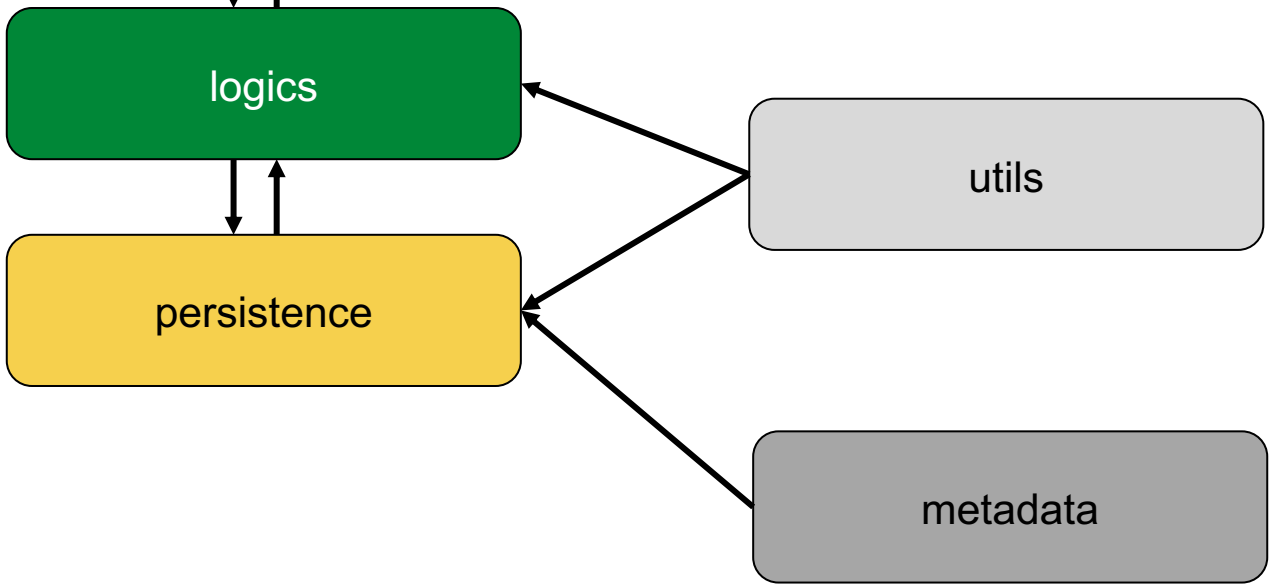


`mocalum` architecture

4 submodules



Methods that peform calculations



Generic functions

Data and data related methods

Single location for metadata

mocalum object – public methods (logics submodule)

1. `add_lidar()`
2. `generate_complex_trajectory()`
3. `generate_PPI_scan()`
4. `generate_uncertainties()`
5. `generate_flow_field()`
6. `project_to_los()`
7. `reconstruct_wind()`
8. `generate_virtual_sonic()`

mocalum object – public data (persistance submodule)

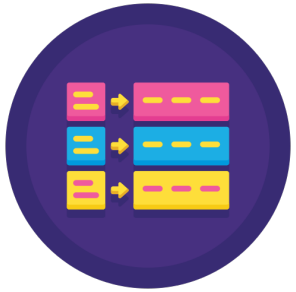
Xarray DataSets (for users and mocalum methods)**:**

- Probing dataset
- Flow field dataset
- LOS dataset
- Reconstructed wind dataset
- Virtual sonic anemometer dataset

datasets

Python dictionaries (for mocalum methods)**:**

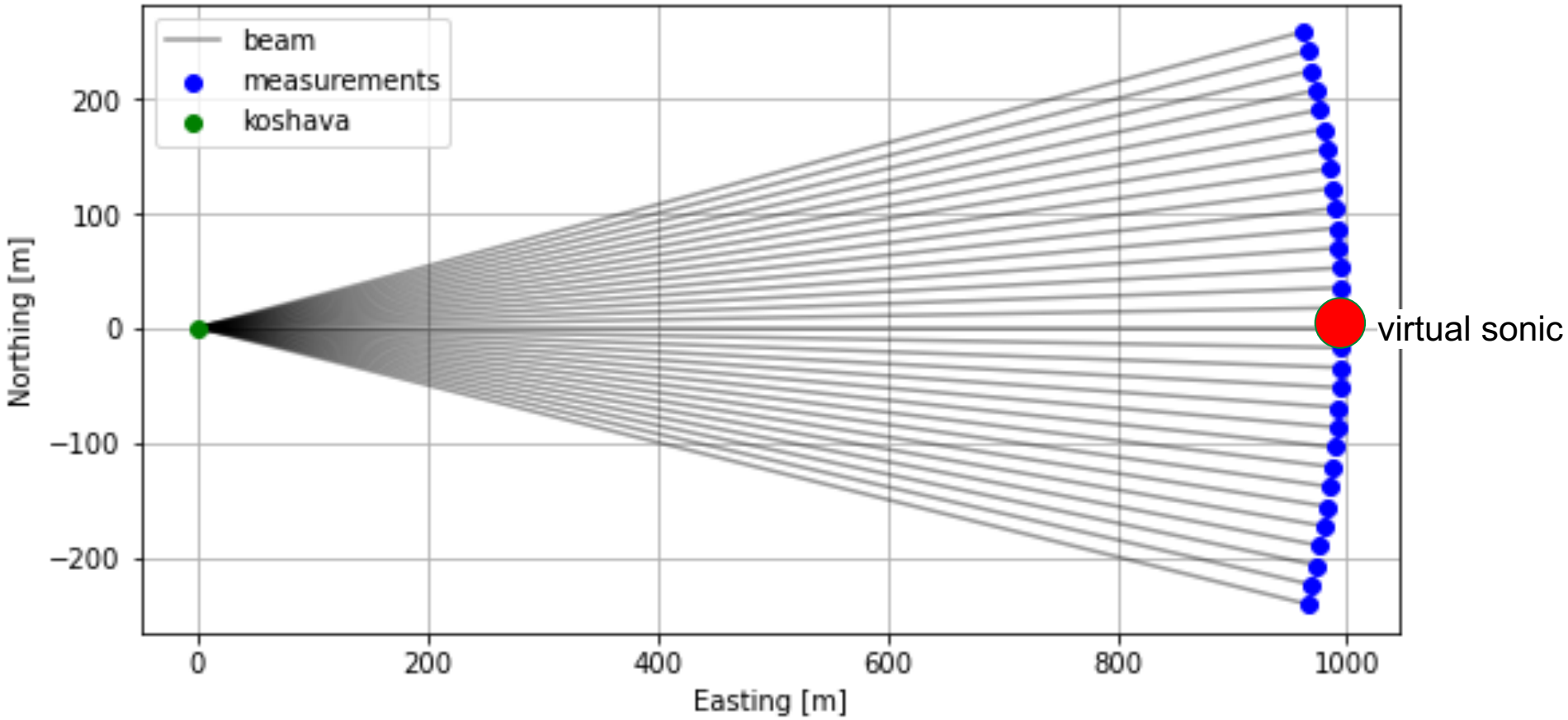
- Flow model config
- Measurement config
- Bounding box for measurement points
- Bounding box for flow field generation



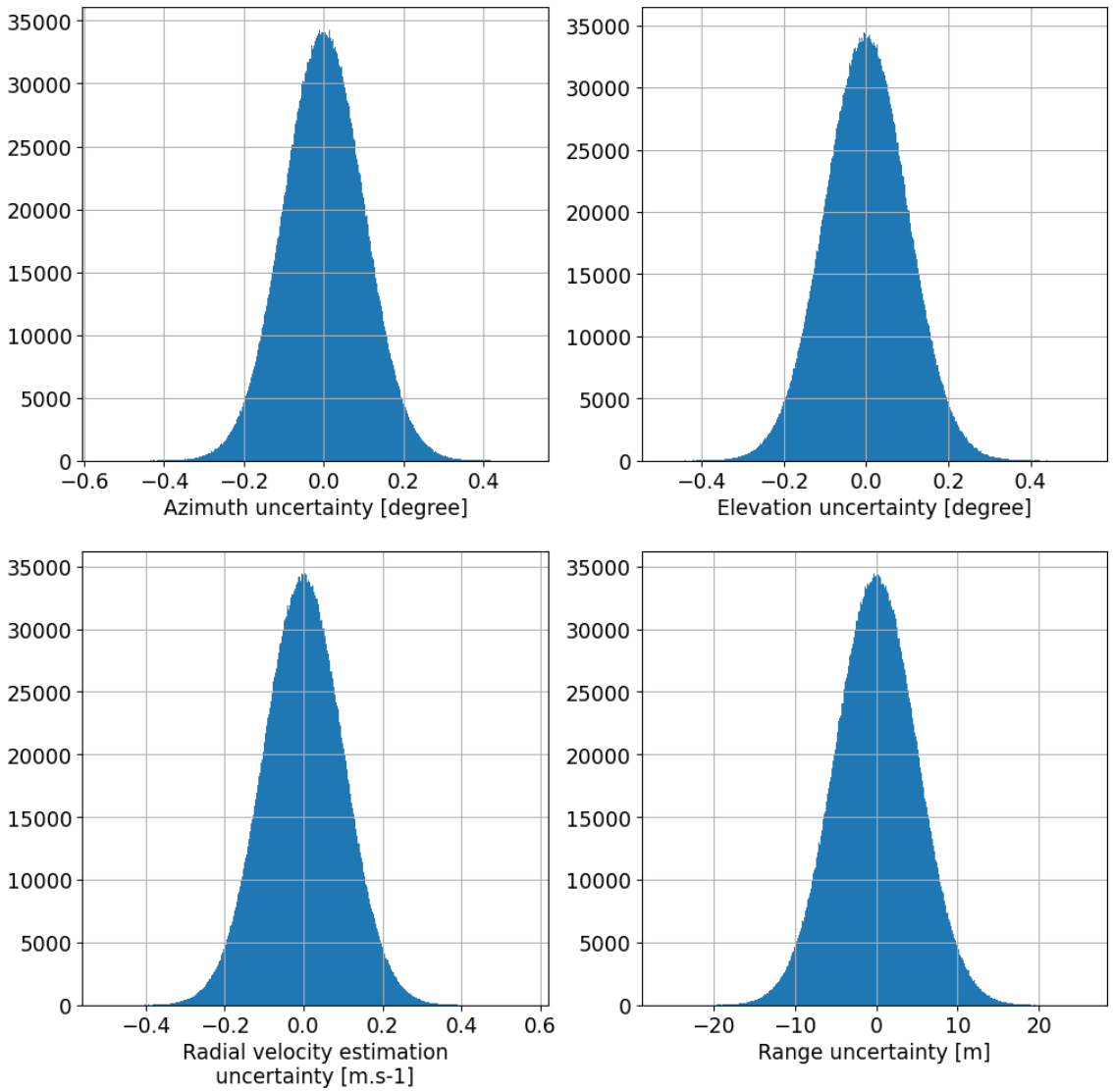
dictionaries

Add lidar(s)	**add_lidar()**: - creates measurement config
Set measurement scenario	**generate_complex_trajectory() or generate_PPI_scan():** - updates measurement config - creates bounding box for measurement points - creates probing dataset
Sample uncertanty contributors	**generate_uncertainties()**: - updates measurement config - updates bounding box for measurement points - updates probing dataset
Generate flow field	**generate_flow_field()**: - creates bounding box for flow field generation - creates flow model config - creates flow field dataset
Project flow on line-of-sights	**project_los()**: - creates LOS dataset
Reconstruct wind vector	**reconstruct_wind()**: - creates reconstructed wind dataset **generate_virtual_sonic ()**: - creates virtual sonic dataset

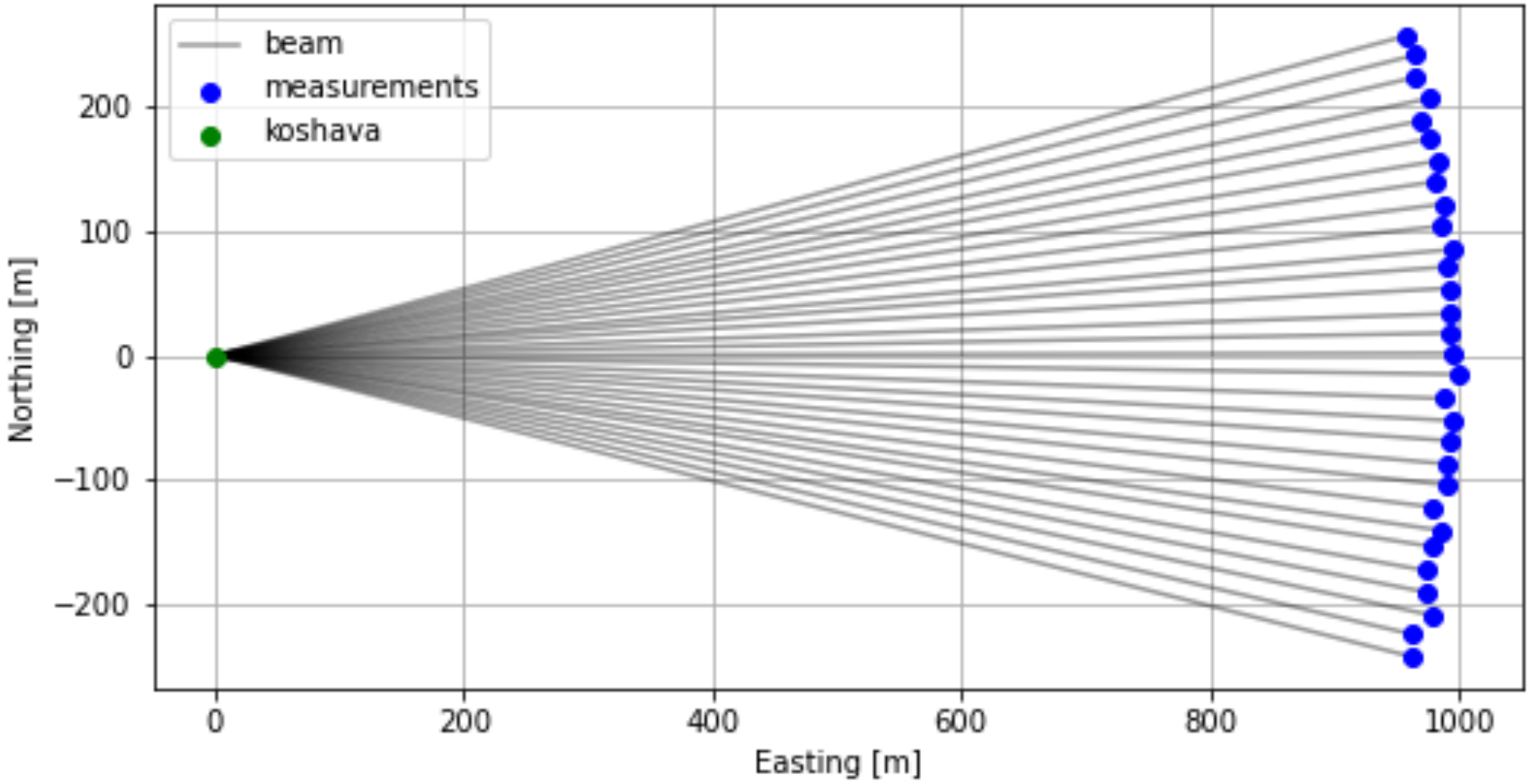
Adding lidar + config of PPI scan



Sampling uncertainties



Sampling uncertainties



Generating flow field

- **uniform flow field**: wind is only changing with height according to the power law
- **turbulent flow field**: generated using a wrapper around pyconturb

PyConTurb implementation

Start of turbulence box

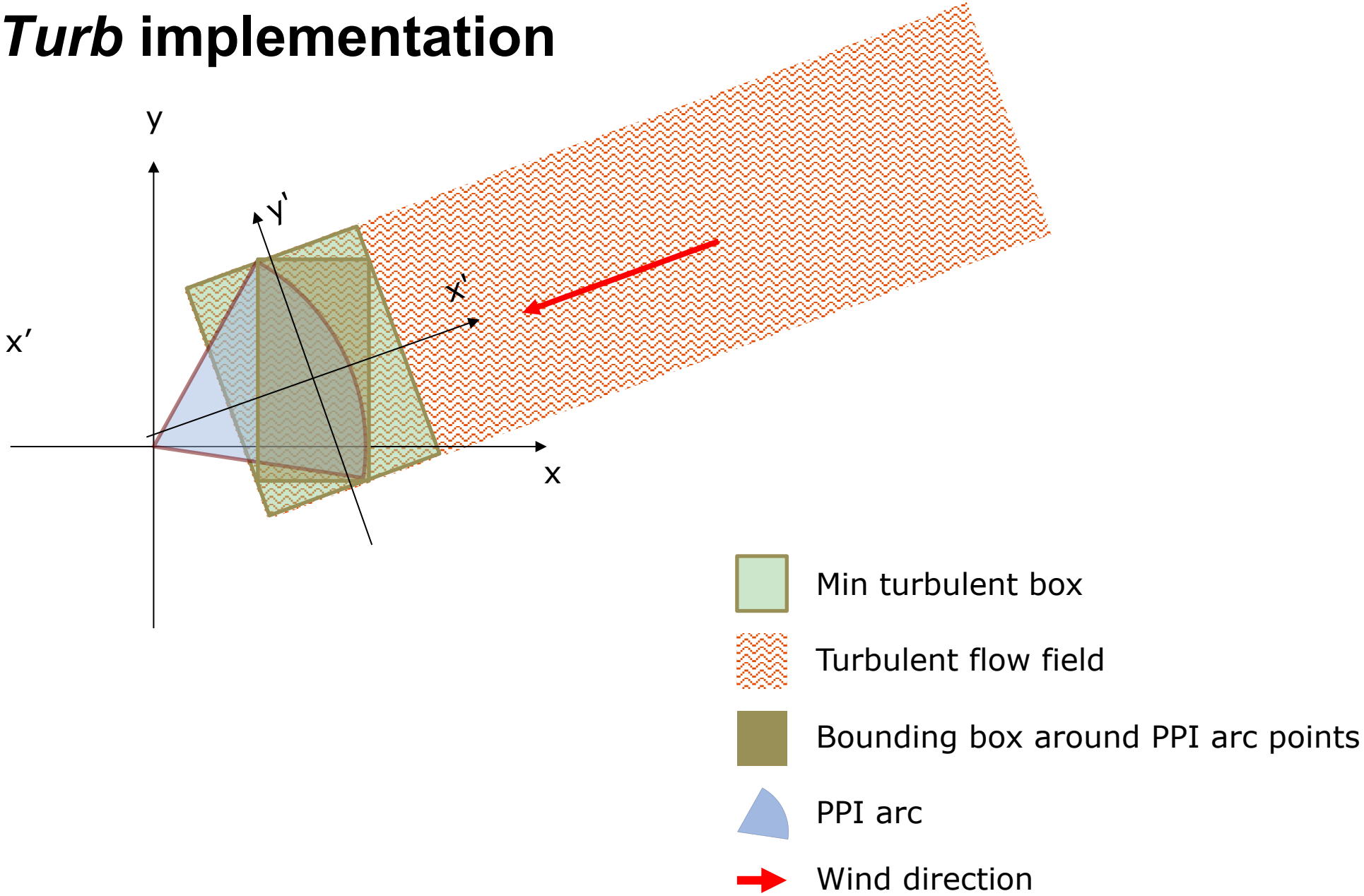
T=0*Δt

Replacing Time coord with x'

$\Delta x' = WS_{mean} * \Delta t$

y,x absolut coordinates

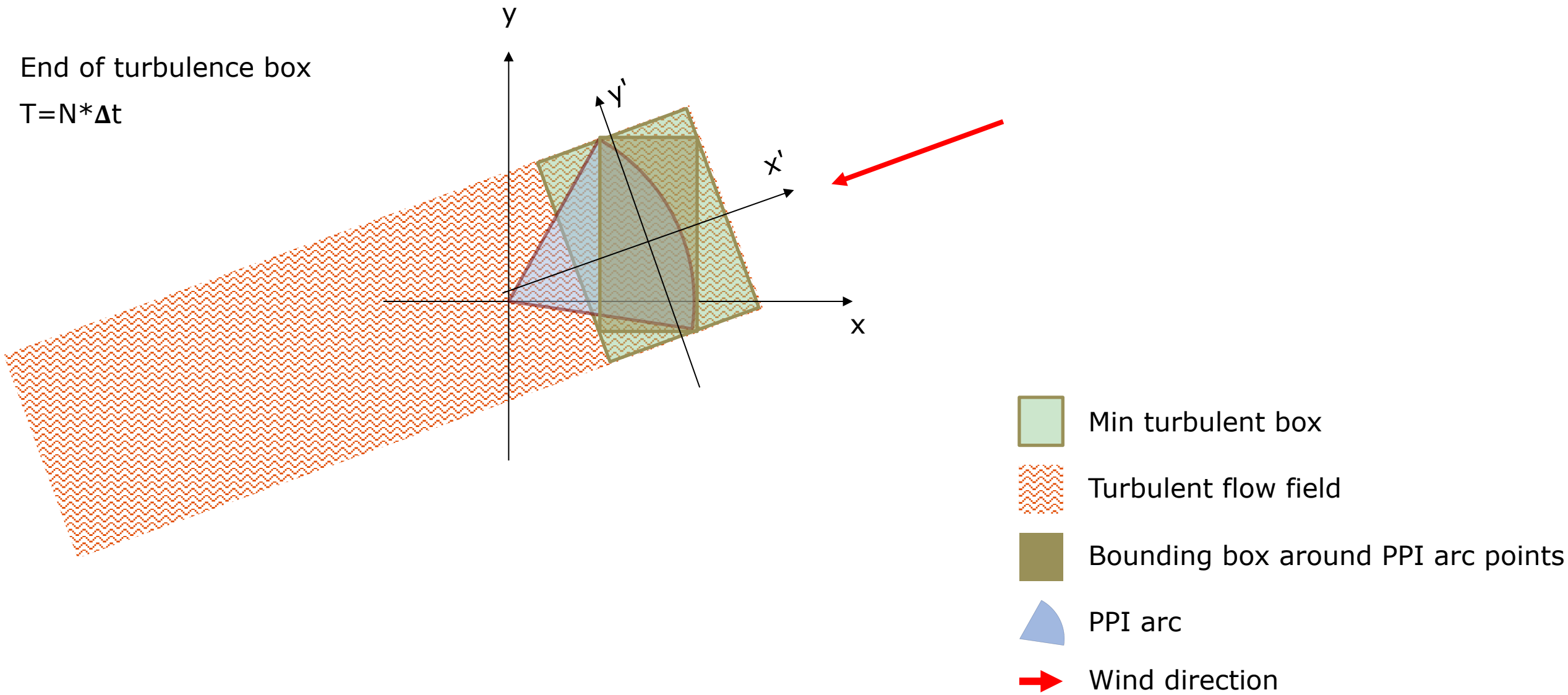
y',x' relative coordinates



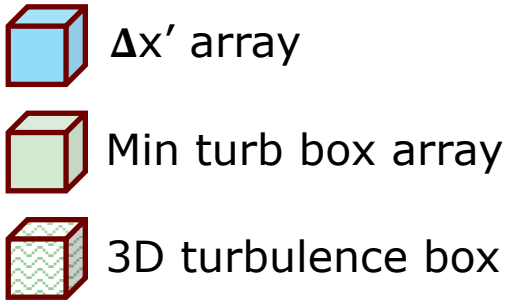
PyConTurb implementation

End of turbulence box

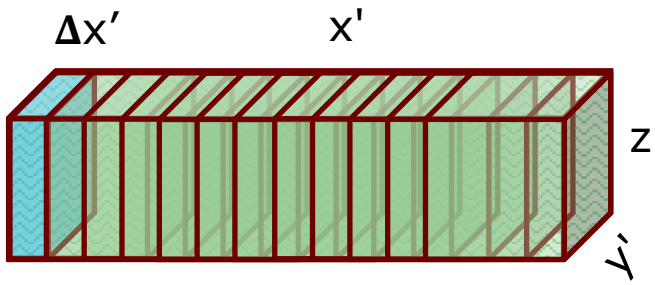
T=N*Δt



From 3D to 4D xarray dataset

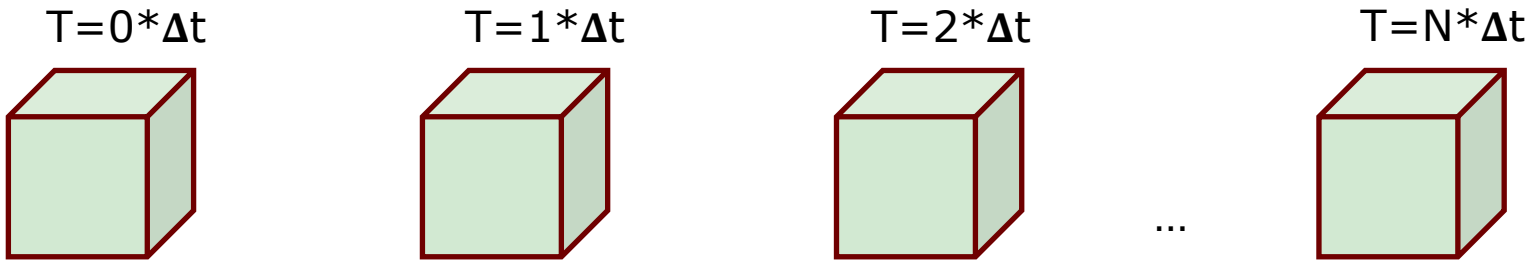


3D dataset

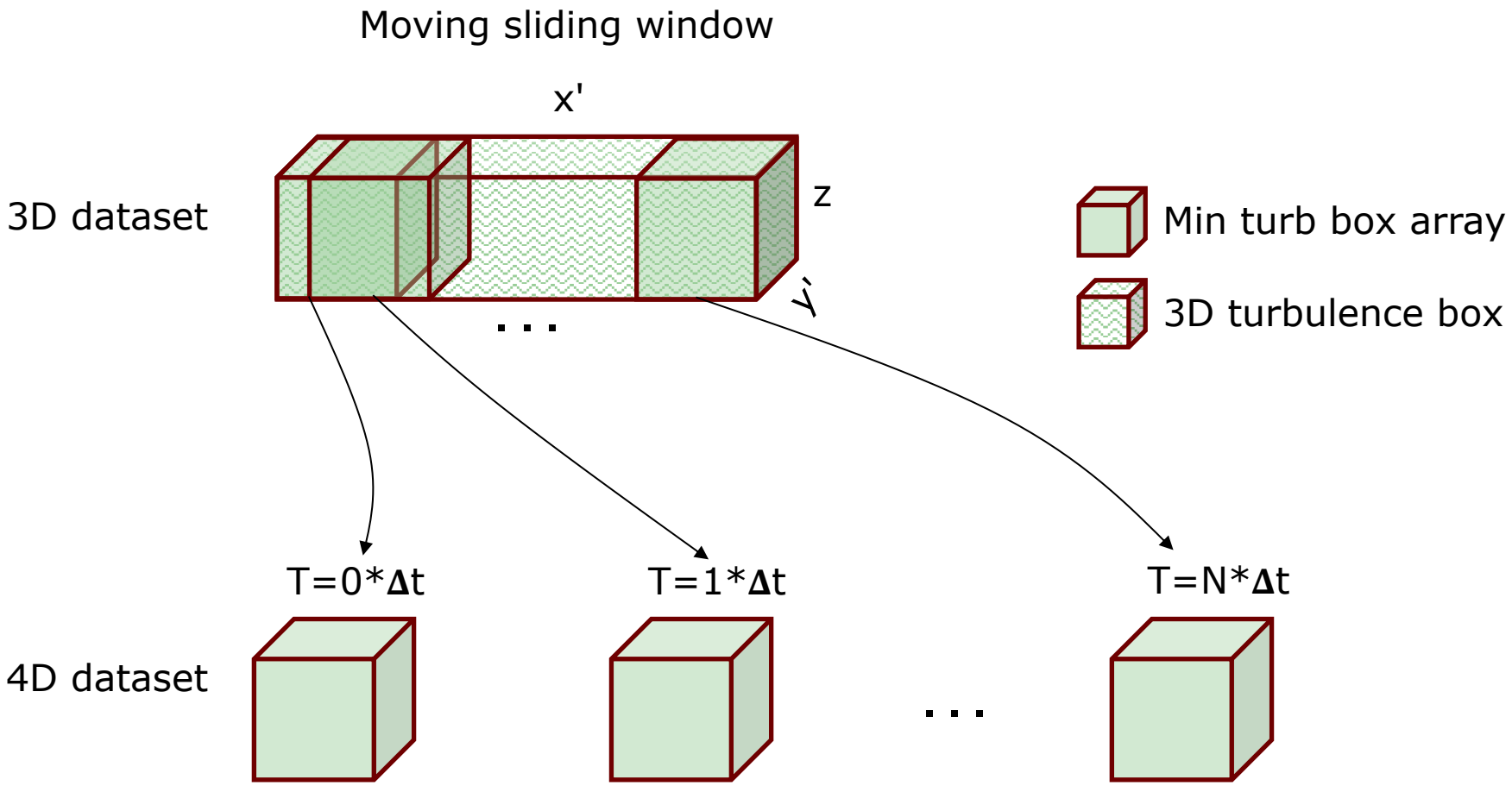


Moving sliding window

4D dataset



From 3D to 4D xarray dataset

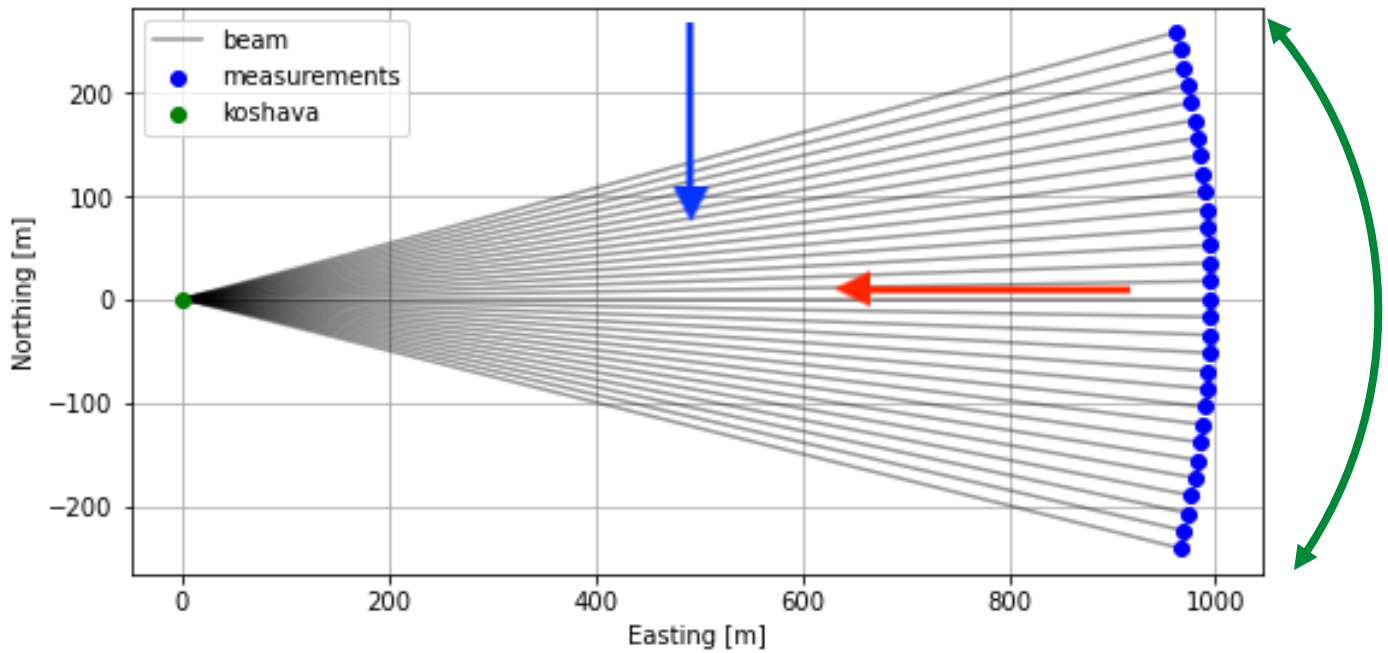


Example: single-Doppler uncertainty (Tutorial 4)

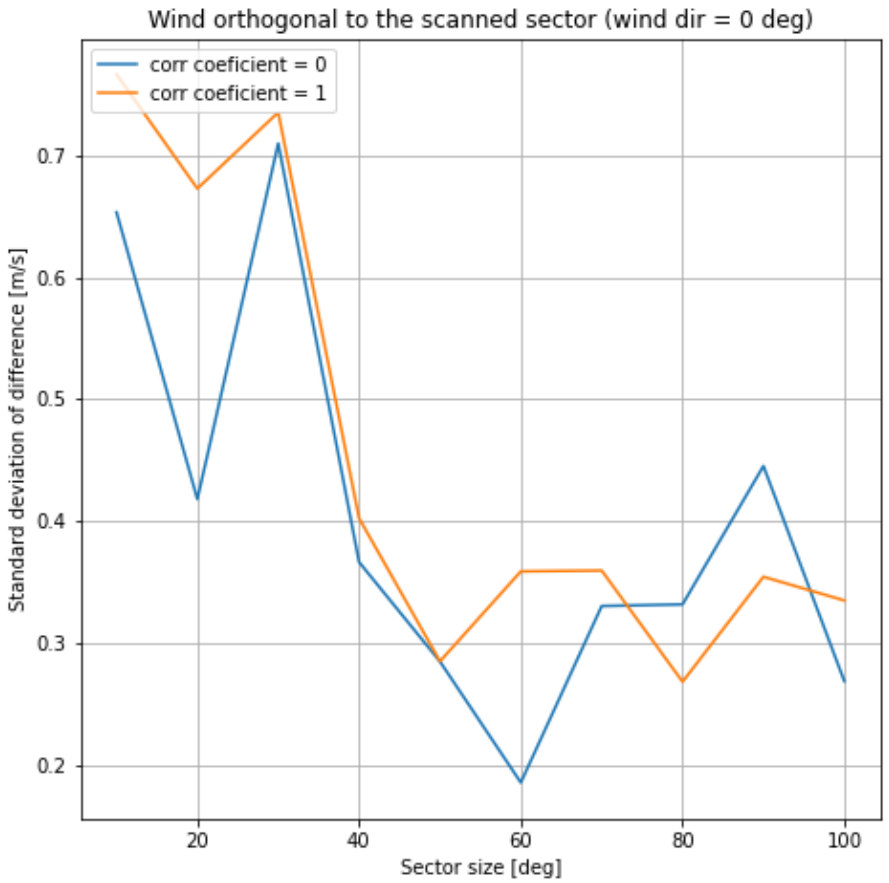
- Scanned sector varied from 10˚ to 100˚ in steps of 10˚
- For two wind directions (0˚ - perpendicular direction, 90˚ - parallel direction)
- Same type of uncertainty contributors correlated/uncorelated for all LOS
- For each simulation configuration 189000 virtual PPI scans performed
- LOS 10-min averaged prior the wind reconstruction

Total: 7.5 million virtual scans

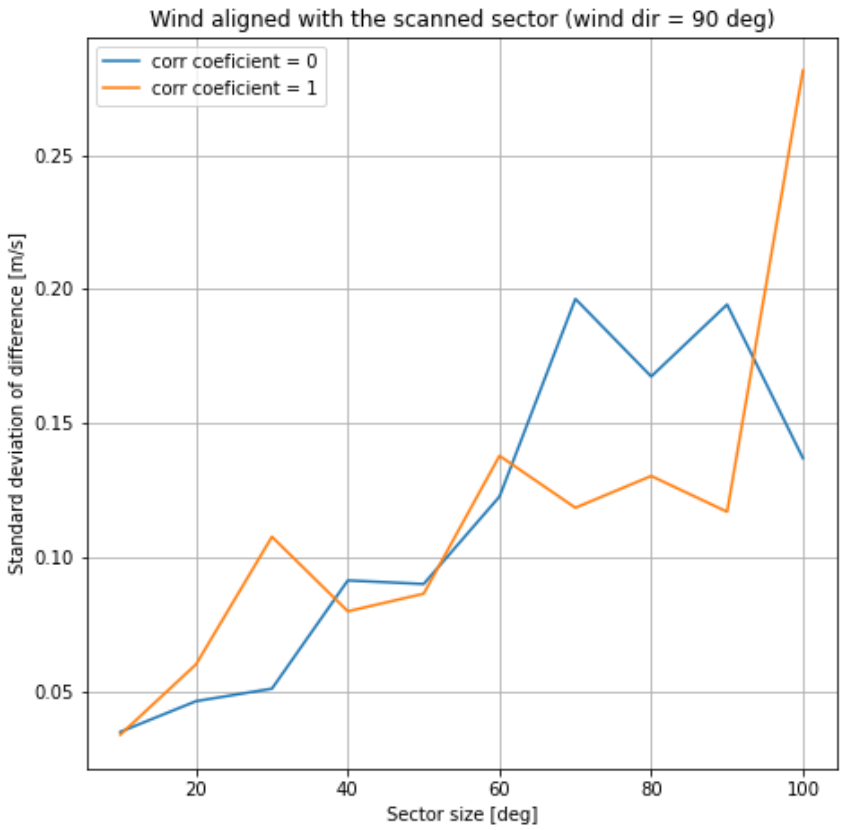
Time: 105 minutes



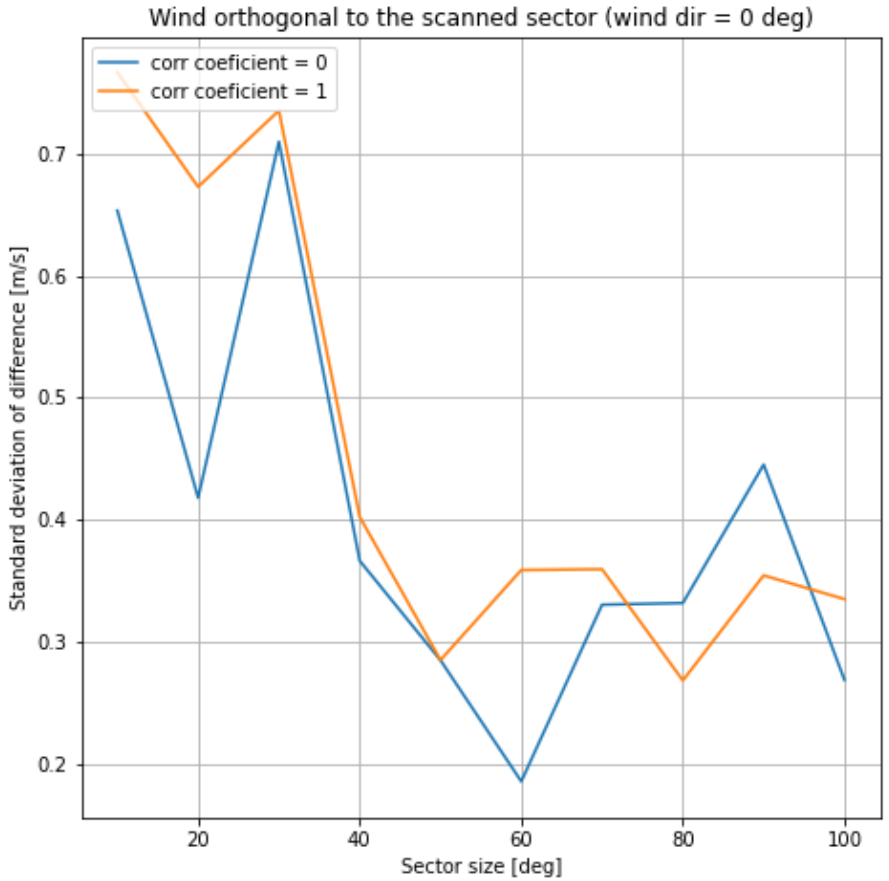
Results wind direction 0˚

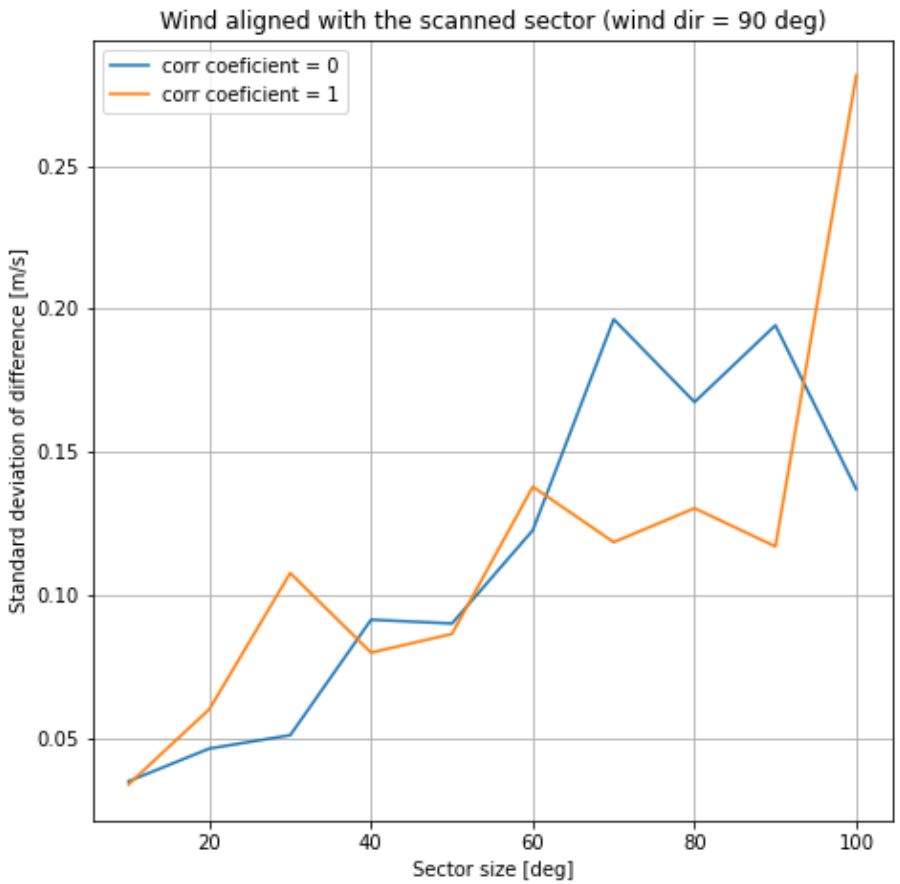


Results wind direction 90˚



Results side by side





mocalum is

- on GitHub for:
  https://github.com/niva83/mocalum

- persisted on Zenodo:
  https://zenodo.org/record/3822069

- explained using jupyter-notebook based tutorials:
  https://github.com/niva83/mocalum/tree/master/examples

- and as a web-based report:
  http://e-windlidar.windenergy.dtu.dk/reports/sd-dd-uncertainty/

Thank you for your attention
Questions?

niva@dtu.dk