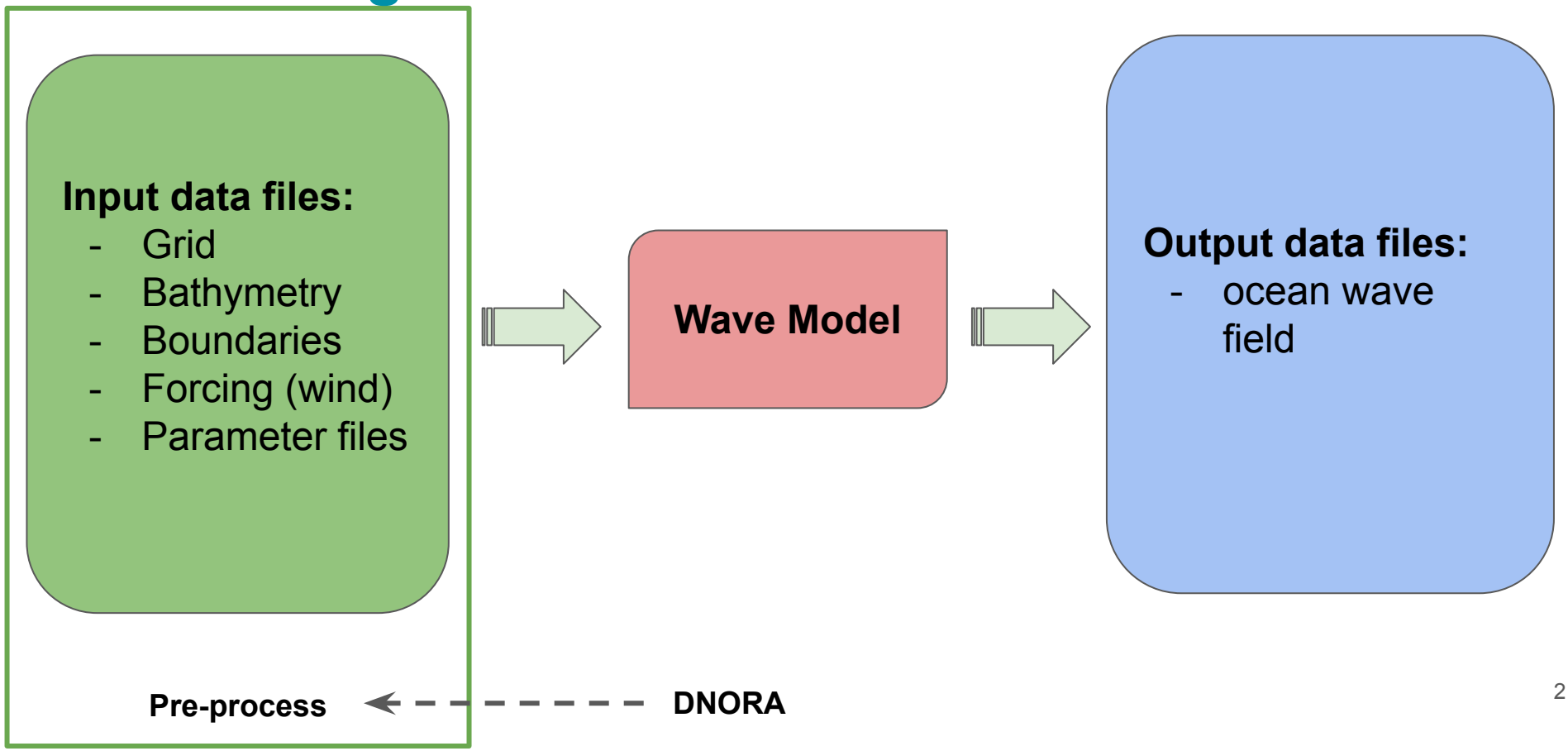
DNORA: Open-access dynamical downscaling of open-ocean wave hindcast/forecast for coastal areas

Konstantinos Christakos (IMT/NTNU & MET Norway) &
Jan-Victor Björkqvist (MET Norway & FMI)

5th Coastal Engineering Day, NTNU/PIANC, 31.03.2022

Running a wave model



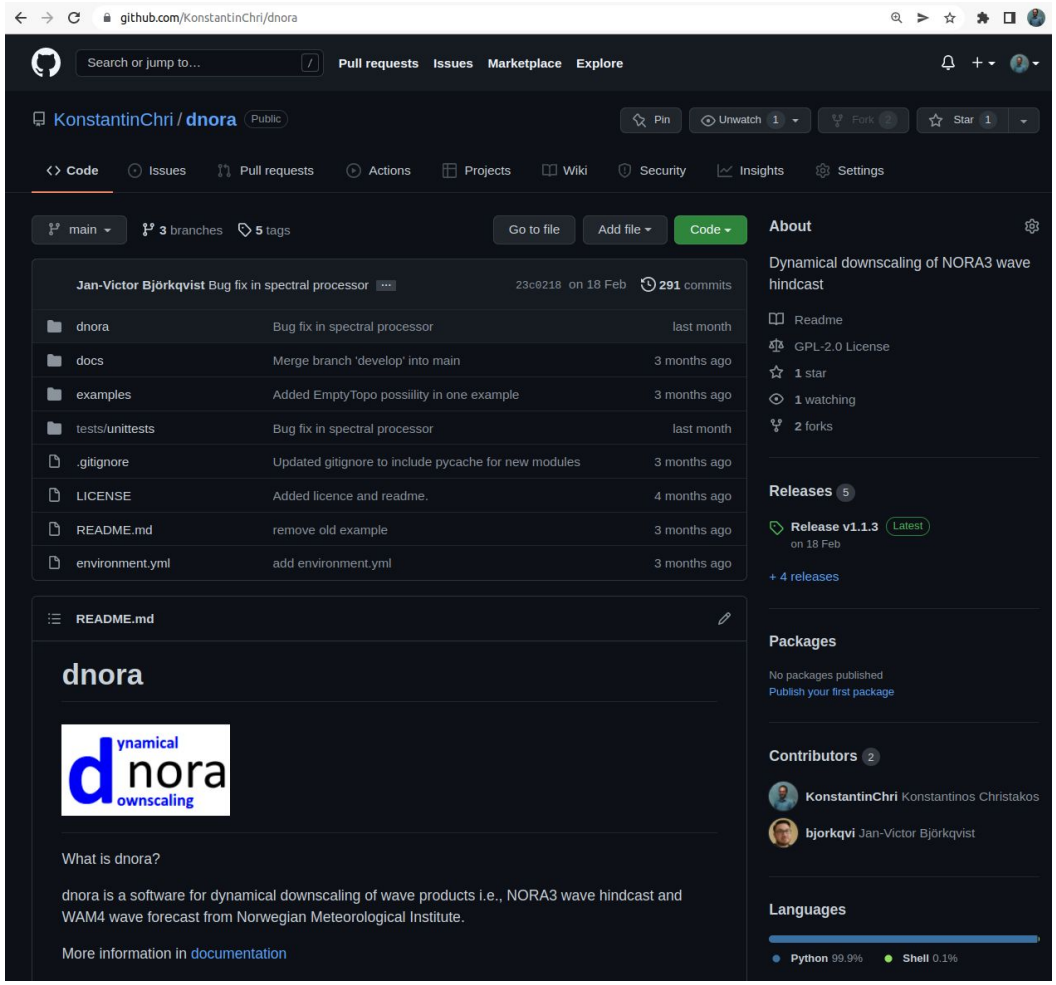
What is DNORA?

DNORA is a Python package for **D**ynamical downscaling of **NORA** wave hindcast using the wave models SWAN, WAVEWATCH III, SWASH, and HOS-ocean.

The package contains functions that:

- Create a high-resolution grid using open-access bathymetry/topography datasets
- Prepare the boundary conditions (NORA3, WAM4)
- Prepare the wind (NORA3, WAM4) forcing
- Create input parameter files for the wave models (SWAN, SWASH, HOS-ocean)
- Run the model (SWAN, SWASH, HOS-ocean)

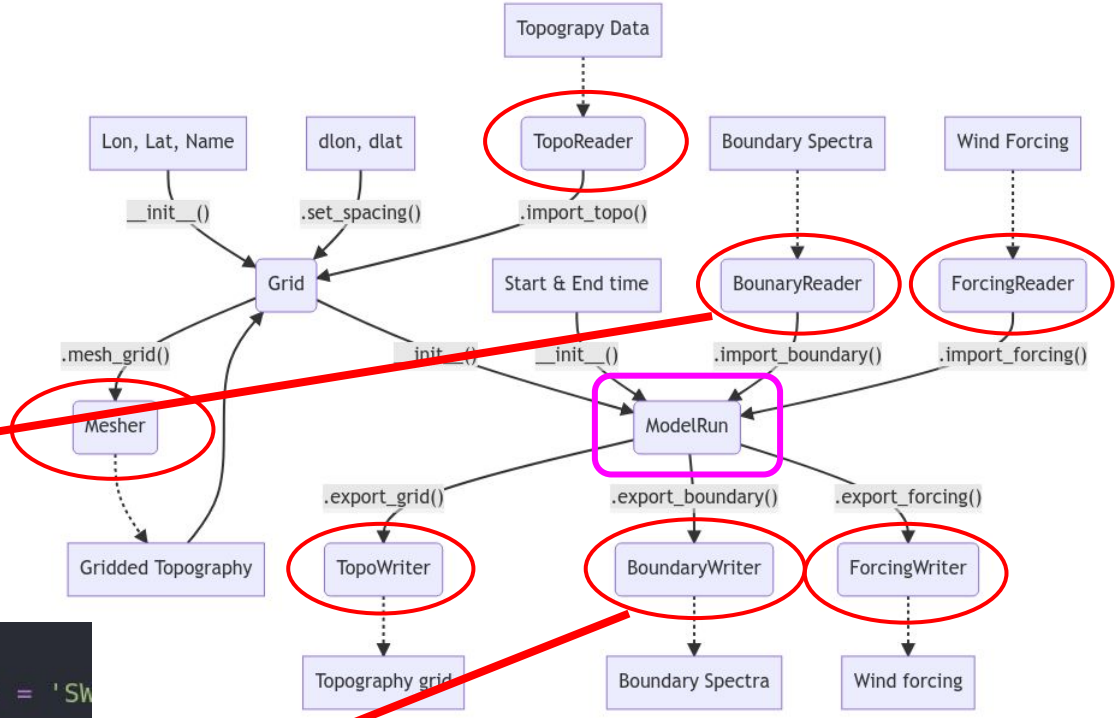
Available at https://github.com/KonstantinChri/dnora



Ideology of DNORA

```python
class NORA3(BoundaryReader):
    def __init__(self, stride: int=24, hours_per_file: int=2
        self.stride = copy(stride)
        self.hours_per_file = copy(hours_per_file)
        self.lead_time = copy(lead_time)
        self.last_file = copy(last_file)

        return

    def convention(self) -> str:
        return 'Ocean'
```

```python
class SWAN(BoundaryWriter):
    def __init__(self, factor = 1E-4, out_format = 'SW
        self.factor = factor
        self.out_format = out_format
        return

    def _convention_in(self) -> str:
        """Convention of spectra"""
        return 'Met'
```

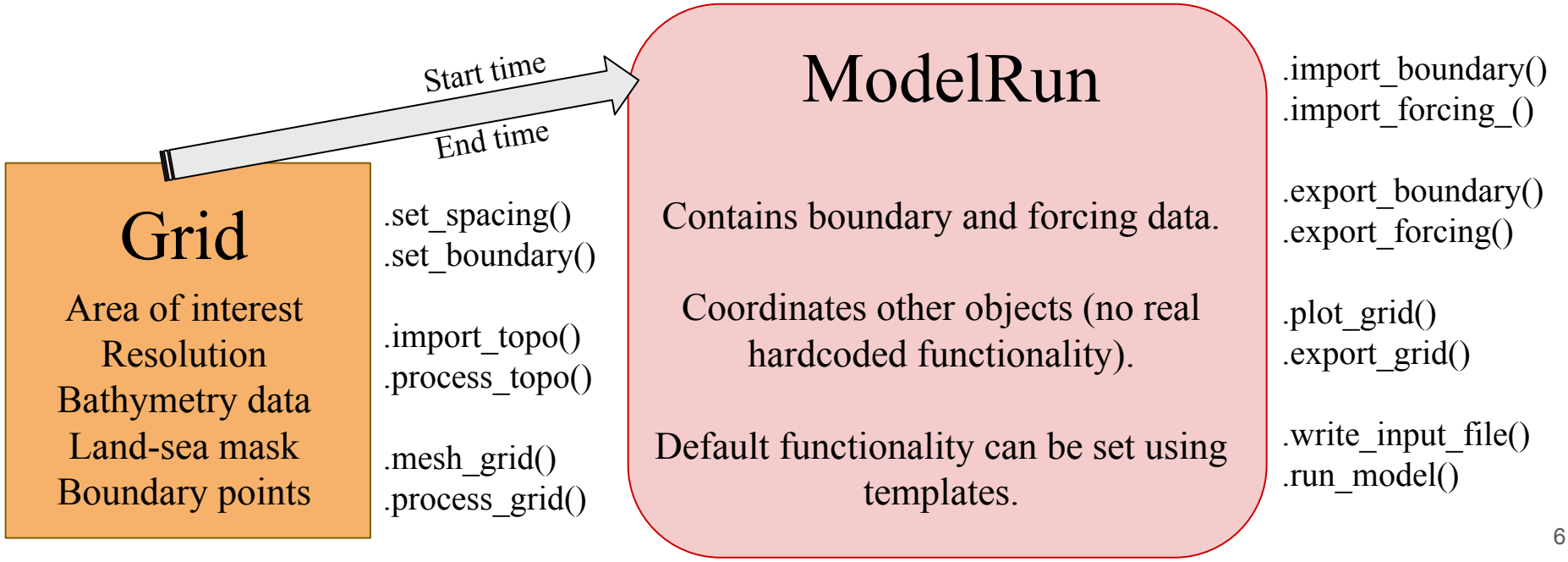


Functionality can be **added incrementally** based on needs.

Typical combinations can also be **packaged as templates** for quick and easy use.

Structure of DNORA

To use DNORA the user needs to work directly with two objects:

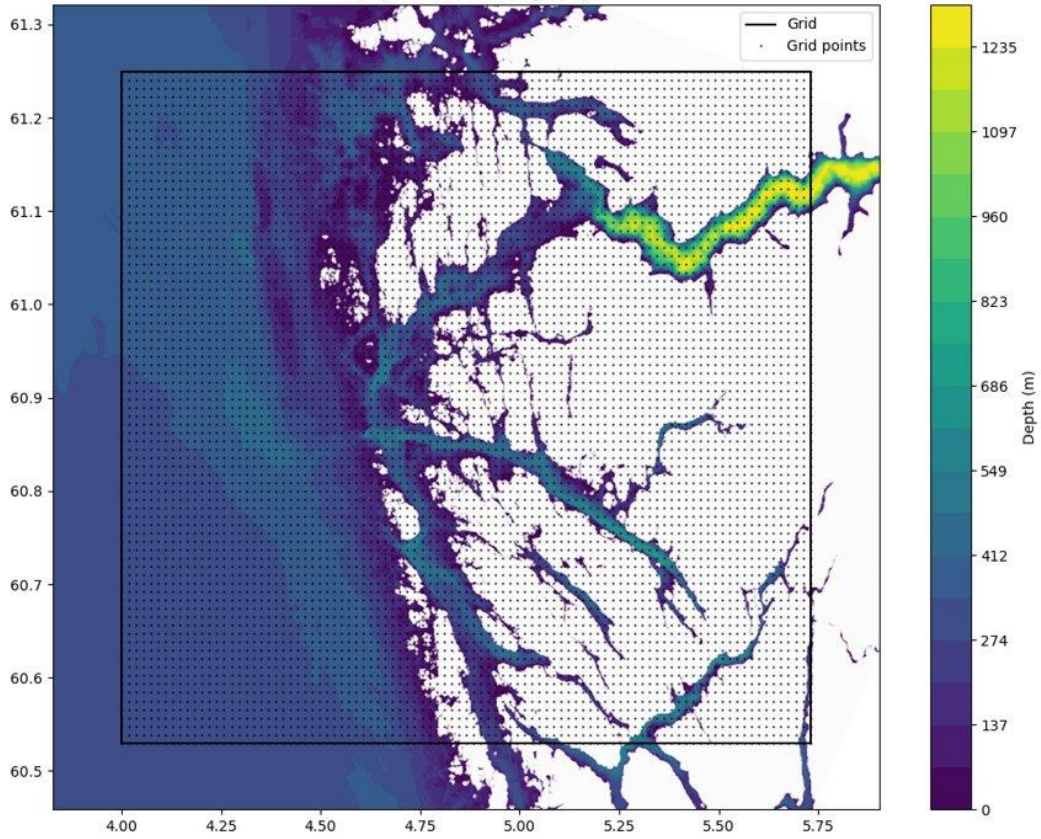


Defining regular grid

```
mdl.plot_grid()
```

Input data files:

- **Grid**
- **Bathymetry**

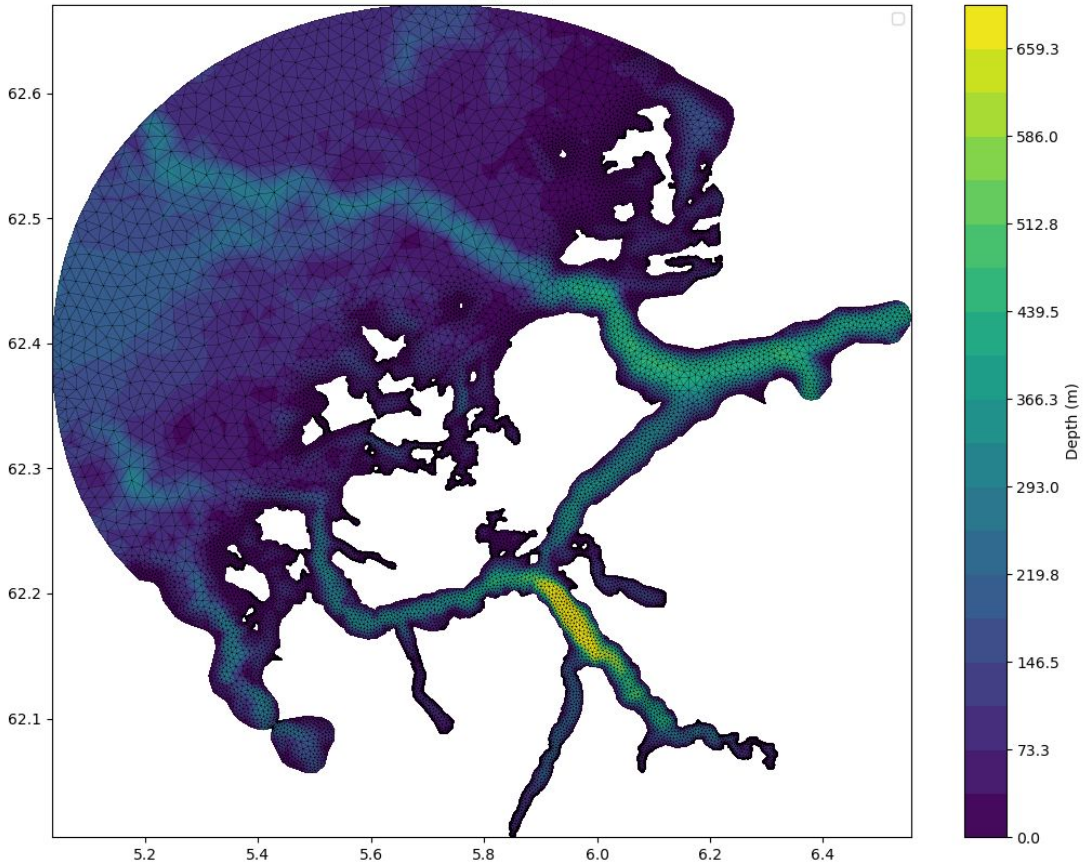


Defining unstructured grid

```
mdl.plot_grid()
```

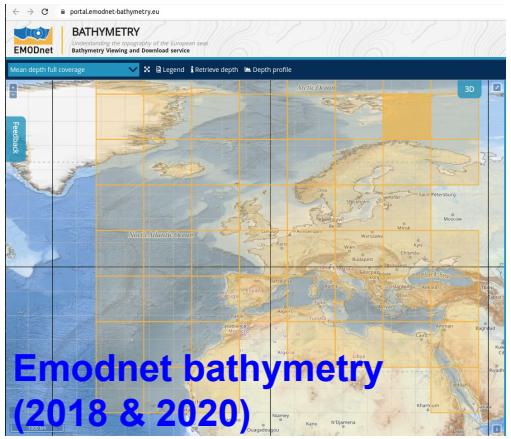
Input data files:

- **Grid**
- **Bathymetry**

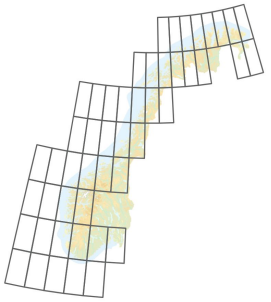


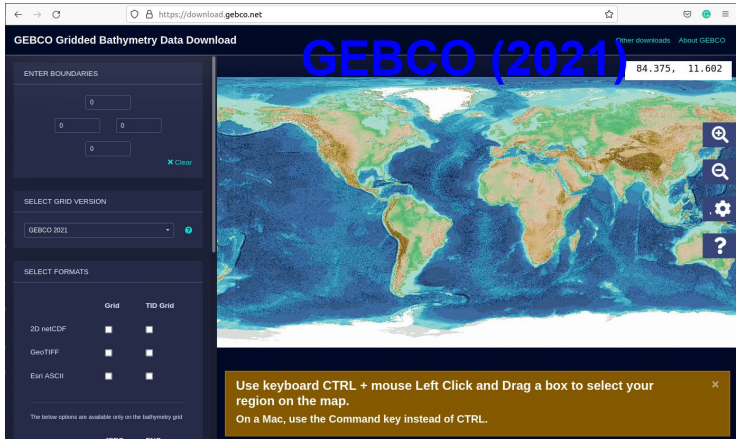
Input data files:

- Grid
- **Bathymetry**



Kartverket50mNo



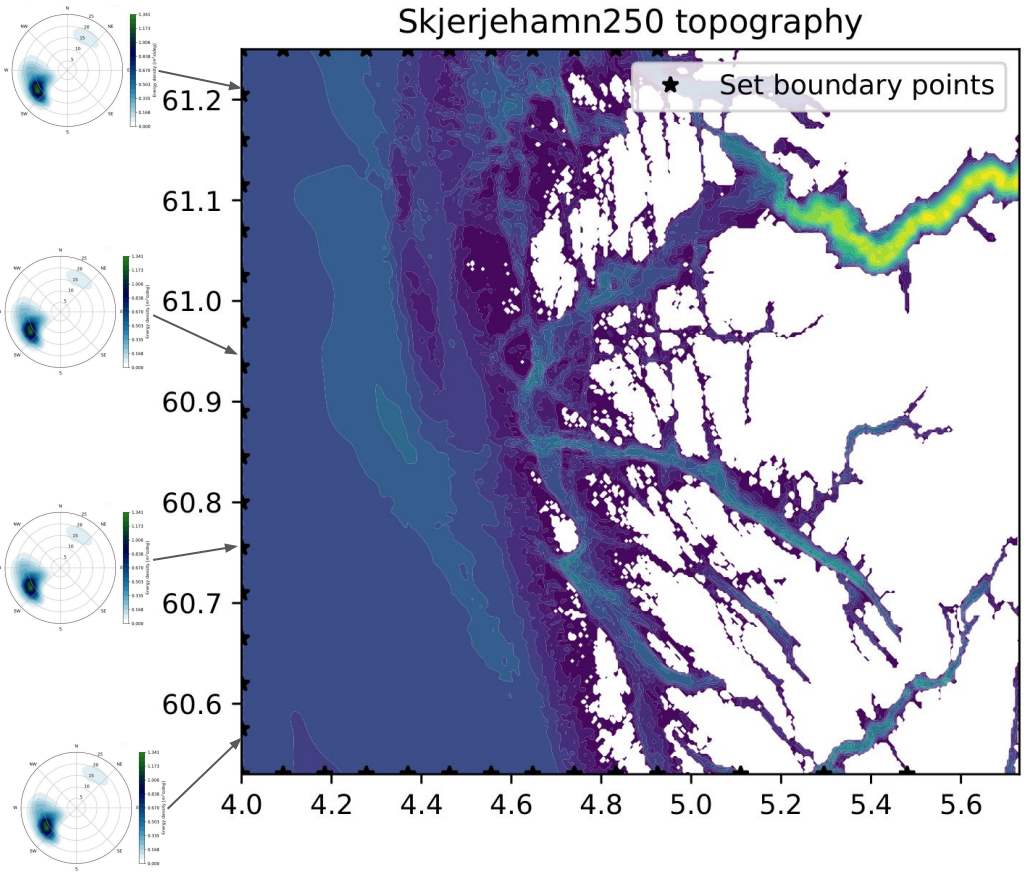


Defining Boundaries

```
mdl.plot_grid()
```

Input data files:

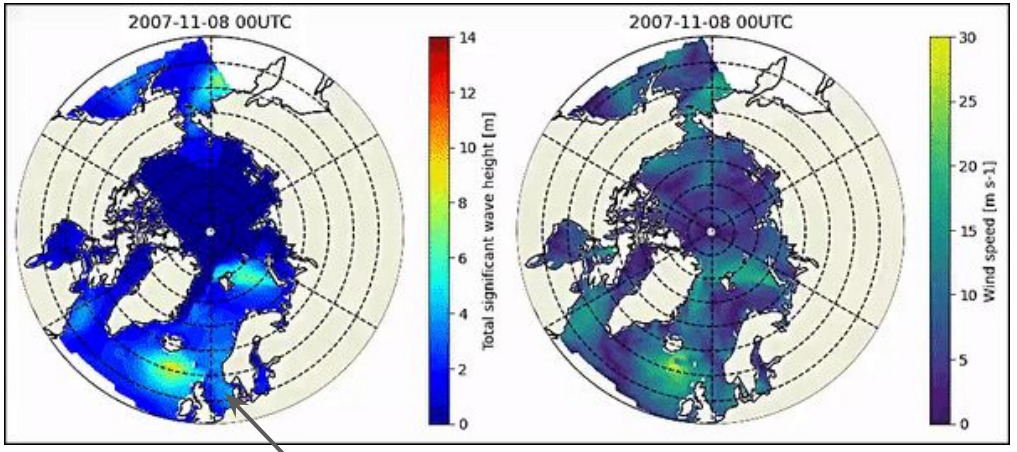
- Grid
- Bathymetry
- **Boundaries**



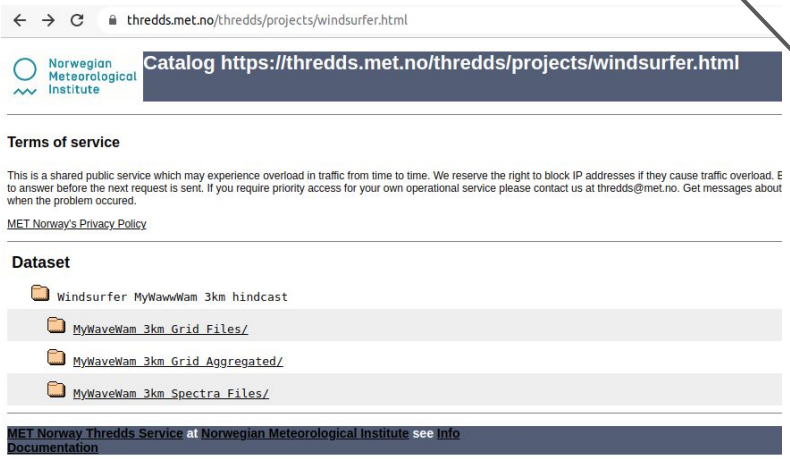
Boundaries & Forcing (NORA3:1992-2020)

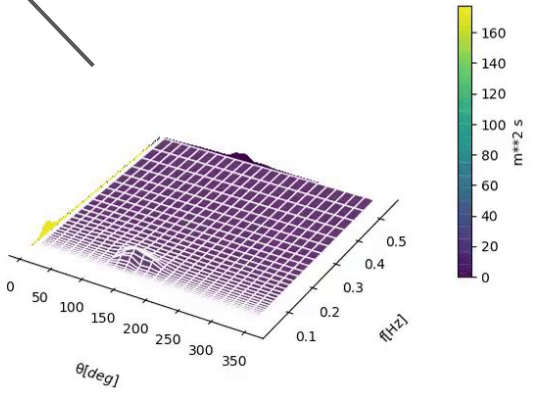
Input data files:

- Grid
- Bathymetry
- **Boundaries**
- **Forcing**



NORA3
lon.=3.0574315,lat=56.56322
2007-11-08T00
θ_p=172.5deg
T_p=12.28s



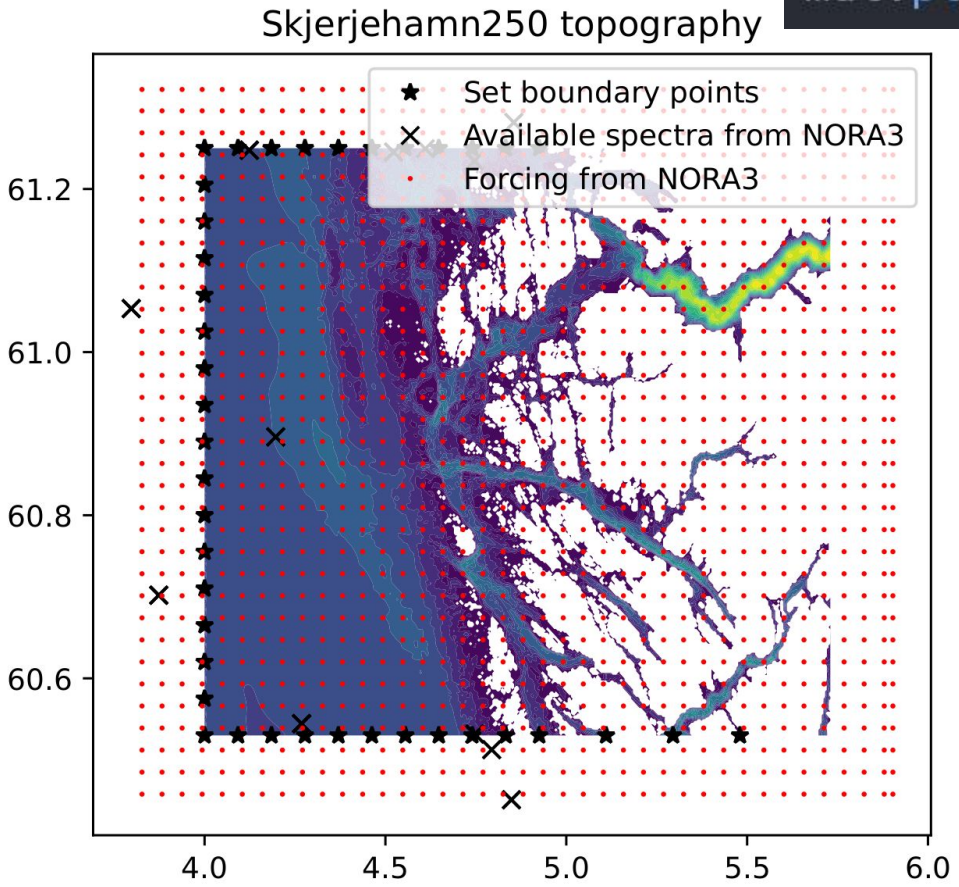


Defining boundaries and forcing

Input data files:

- Grid
- Bathymetry
- **Boundaries**
- **Forcing**

```
mdl.plot_grid()
```



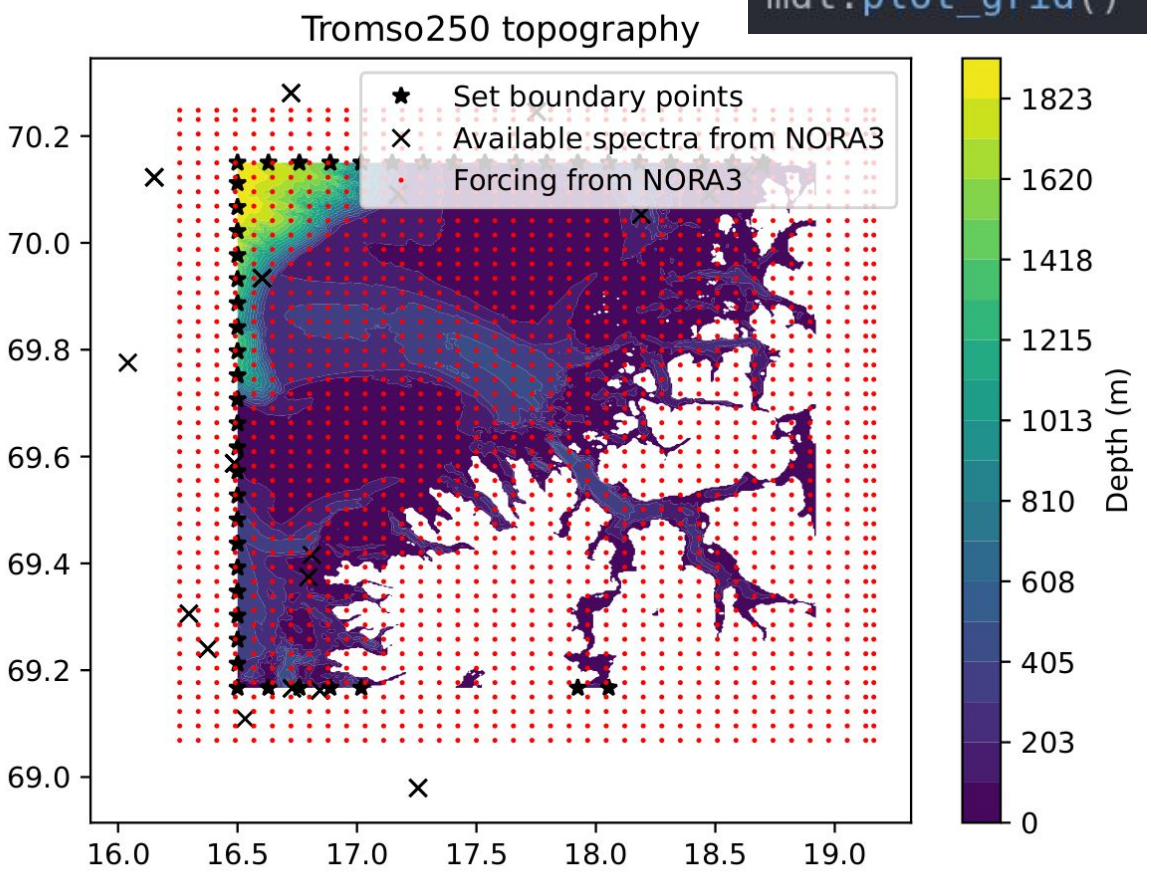
Example: DNORA/SWAN using NORA3

```python
# ================================================================================
# IMPORT dnora
# ================================================================================
import sys
dnora_directory = '../'
sys.path.insert(0, dnora_directory)
from dnora import grd, mdl, inp
# ================================================================================
# DEFINE GRID OBJECT
# ================================================================================
# Set grid definitions
grid = grd.Grid(lon=(16.500, 18.9212), lat=(69.167, 70.150), name='Tromso250')

# Set spacing and boundary points
grid.set_spacing(dm=250)

# Import topography and mesh it down to the grid definitions

grid.import_topo(topo_reader=grd.read.EMODNET2018(tile='C6',
                                  folder='/home/konstac/bathy/'))

grid.mesh_grid()
#
# # Set the boundaries
bnd_set = grd.boundary.EdgesAsBoundary(edges=['N', 'W', 'S'], step=20)
grid.set_boundary(boundary_setter=bnd_set)
#
#
```

```python
# ================================================================================
# DEFINE MODEL OBJECT
# ================================================================================
model = mdl.SWAN_NORA3(grid, start_time='1993-01-31T12:00',
                               end_time='1993-02-04T12:00')
# ================================================================================
# IMPORT BOUNDARIES AND FORCING
# ================================================================================
model.import_boundary()
model.import_forcing()
# ================================================================================
# PLOT GRID, FORCING AND BOUNDARIES
# ================================================================================
model.plot_grid(save_fig=True,show_fig=False)
# ================================================================================
# WRITE OUTPUT FOR SWAN RUN
# ================================================================================
model.export_grid()
model.export_boundary()
model.export_forcing()
model.write_input_file(input_file_writer=
                       inp.SWAN(spec_points=[(18.13, 69.83)]))
# ================================================================================
# SWAN RUN
# ================================================================================
model.run_model()
```

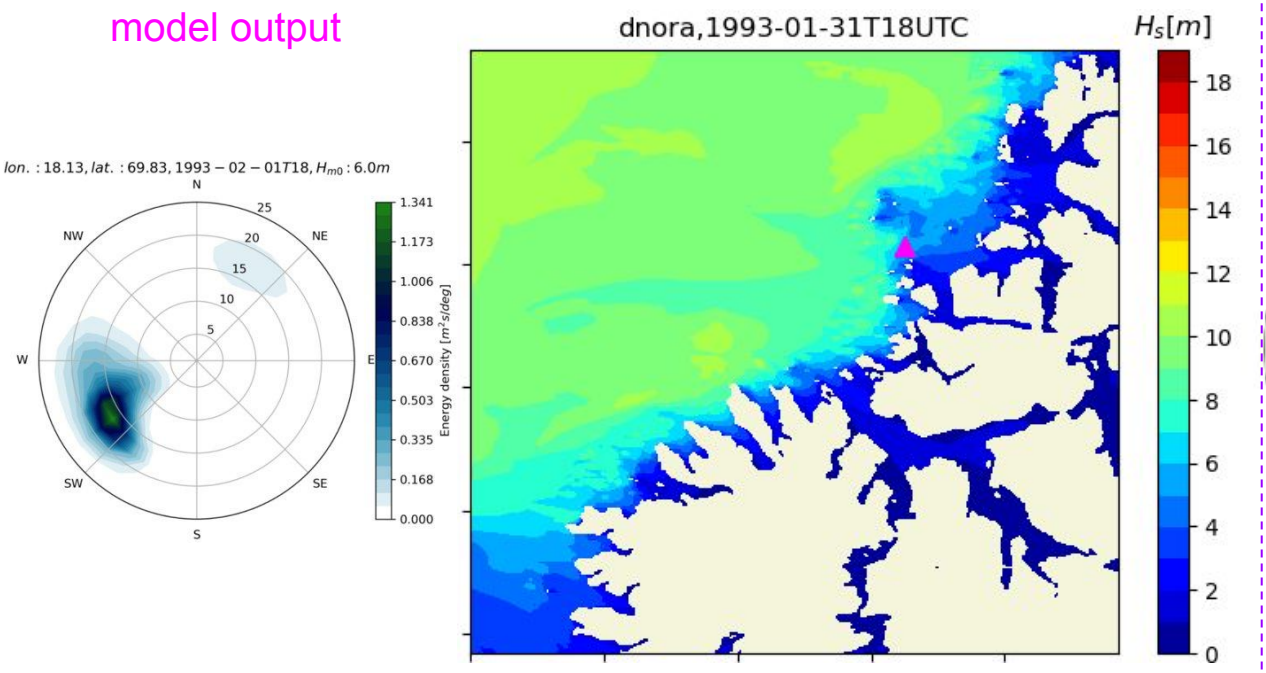
Example: DNORA/SWAN using NORA3

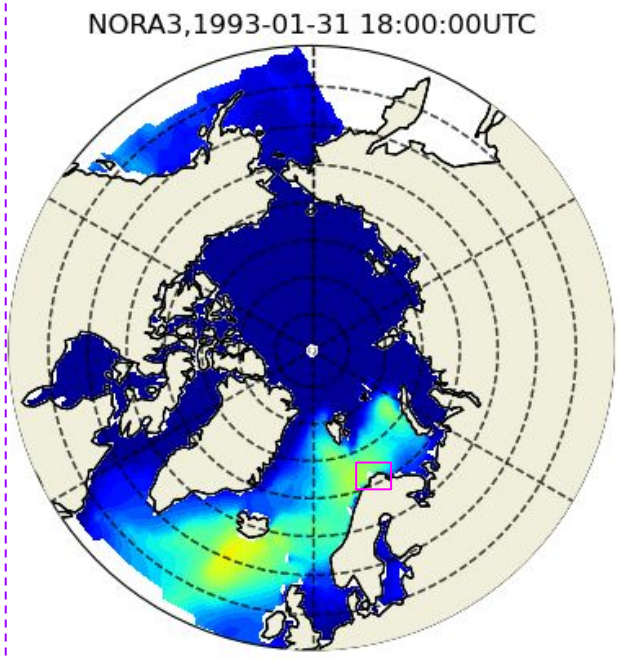
```
mdl.plot_grid()
```



Example: DNORA/SWAN using NORA3

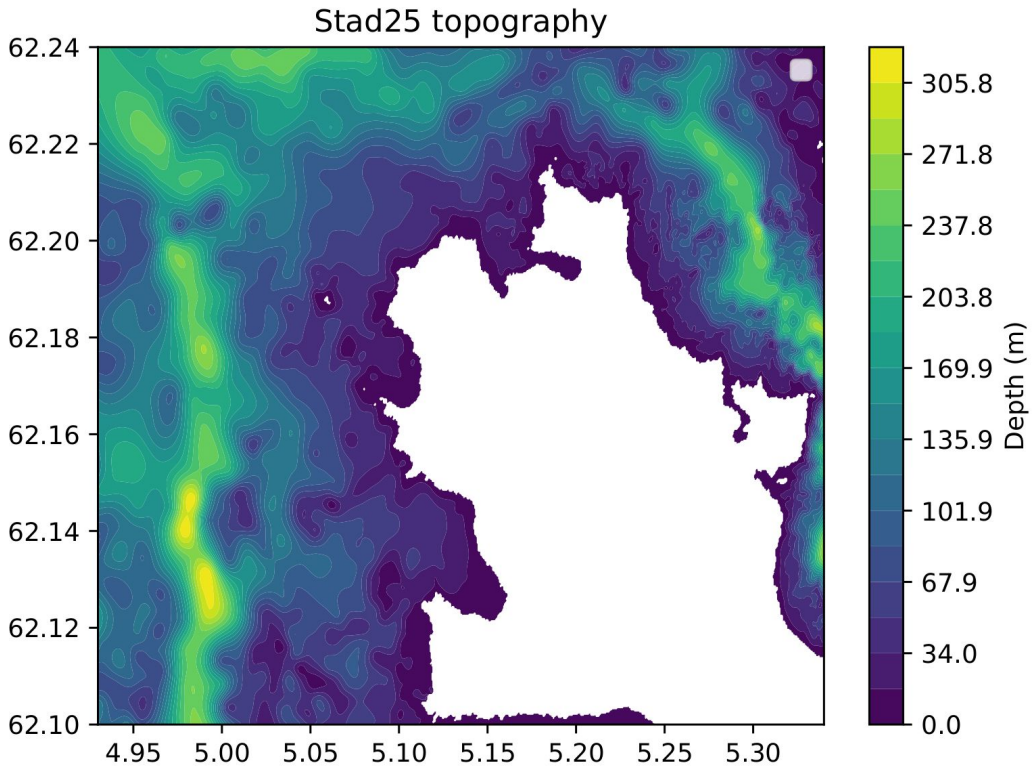
model output





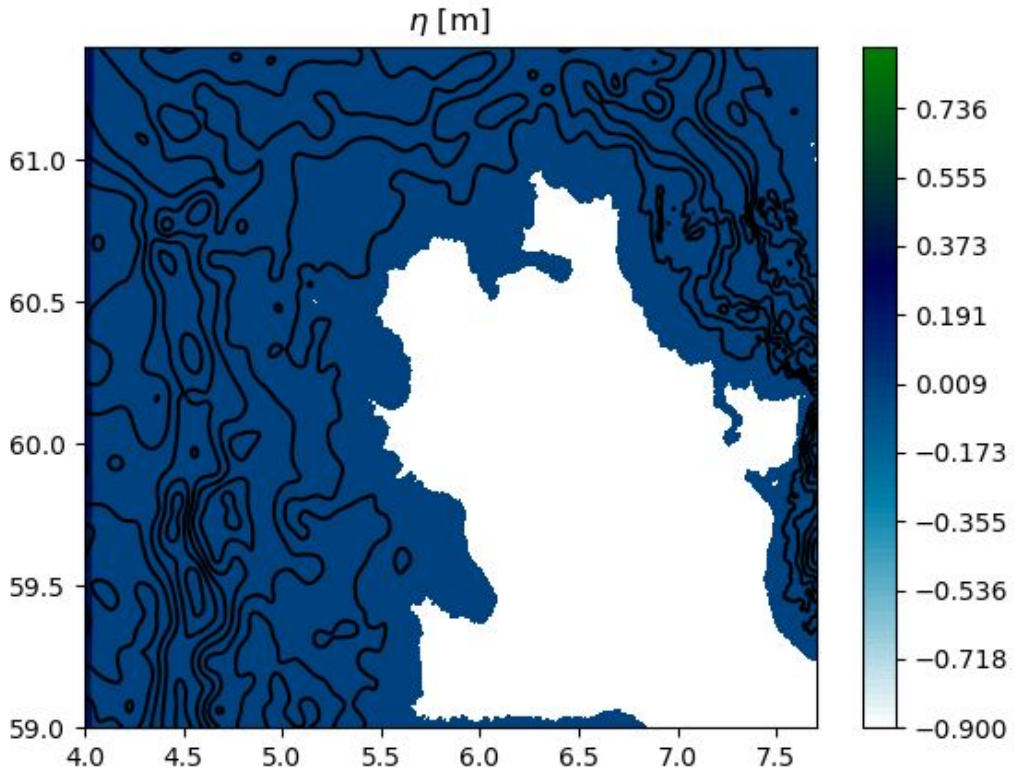
DNORA/SWASH

```python
# ==========================================================================
# IMPORT dnora
# ==========================================================================
import sys
dnora_directory = '../'
sys.path.insert(0, dnora_directory)
from dnora import grd, mdl, inp
# ==========================================================================
# DEFINE GRID OBJECT
# ==========================================================================
# Set grid definitions
grid = grd.Grid(lon=(4.93, 5.34), lat=(62.10, 62.24), name='Stad25')

# Set spacing and boundary points
grid.set_spacing(dm=25)

# Import topography and mesh it down to the grid definitions
grid.import_topo(topo_reader=grd.read.KartverketNo50m(tile='B1408',
                                folder='/home/konstac/bathy/Kartverket'))
grid.mesh_grid()

# ==========================================================================
# DEFINE MODEL OBJECT
# ==========================================================================
model = mdl.SWASH(grid, start_time='2018-08-25T00:00', end_time='2018-08-25T02:00')

# ==========================================================================
# PLOT GRID, FORCING AND BOUNDARIES
# ==========================================================================
model.plot_grid(save_fig=True,show_fig=False)

# ==========================================================================
# WRITE OUTPUT FOR SWASH RUN
# ==========================================================================
model.export_grid()
model.write_input_file(input_file_writer=inp.SWASH(
    bound_side_command='BOU SIDE W CCW CON REG 1.0 20 270 '))
# ==========================================================================
# SWASH RUN
# ==========================================================================
model.run_model()

```



DNORA/SWASH

```python
# ========================================================================
# IMPORT dnora
# ========================================================================
import sys
dnora_directory = '../'
sys.path.insert(0, dnora_directory)
from dnora import grd, mdl, inp
# ========================================================================
# DEFINE GRID OBJECT
# ========================================================================
# Set grid definitions
grid = grd.Grid(lon=(4.93, 5.34), lat=(62.10, 62.24), name='Stad25')

# Set spacing and boundary points
grid.set_spacing(dm=25)

# Import topography and mesh it down to the grid definitions
grid.import_topo(topo_reader=grd.read.KartverketNo50m(tile='B1408',
                                folder='/home/konstac/bathy/Kartverket'))
grid.mesh_grid()

# ========================================================================
# DEFINE MODEL OBJECT
# ========================================================================
model = mdl.SWASH(grid, start_time='2018-08-25T00:00', end_time='2018-08-25T02:00')

# ========================================================================
# PLOT GRID, FORCING AND BOUNDARIES
# ========================================================================
model.plot_grid(save_fig=True,show_fig=False)

# ========================================================================
# WRITE OUTPUT FOR SWASH RUN
# ========================================================================
model.export_grid()
model.write_input_file(input_file_writer=inp.SWASH(
    bound_side_command='BOU SIDE W CCW CON REG 1.0 20 270 '))
# ========================================================================
# SWASH RUN
# ========================================================================
model.run_model()

```



Thank you