TTK17 Model data exercise

alver

June 2021

1 Introduction

SINMOD (Slagstad and McClimans, 2005) is a coupled physical-biological ocean model developed at SINTEF Ocean, which is used for a wide variety of research areas.

In this exercise we will look at an example dataset produced by the SINMOD model in a setup with 800 m horizontal resolution covering the coast outside middle Norway (Fig. 1). The model setup, input values and initialization provides a fairly realistic simulation. The model has been run for about 9 days (April 1-10, 2021), storing elevation, current speed components, temperature and salinity each hours. The data is in NetCDF format, which can be accessed using Matlab. You will be provided a basic Matlab script and a couple of functions to get started with accessing the dataset.

The objectives of this exercise are to:

- Give an opportunity to look at the typical format and structure of ocean model output
- Practice simple analyses of ocean model data

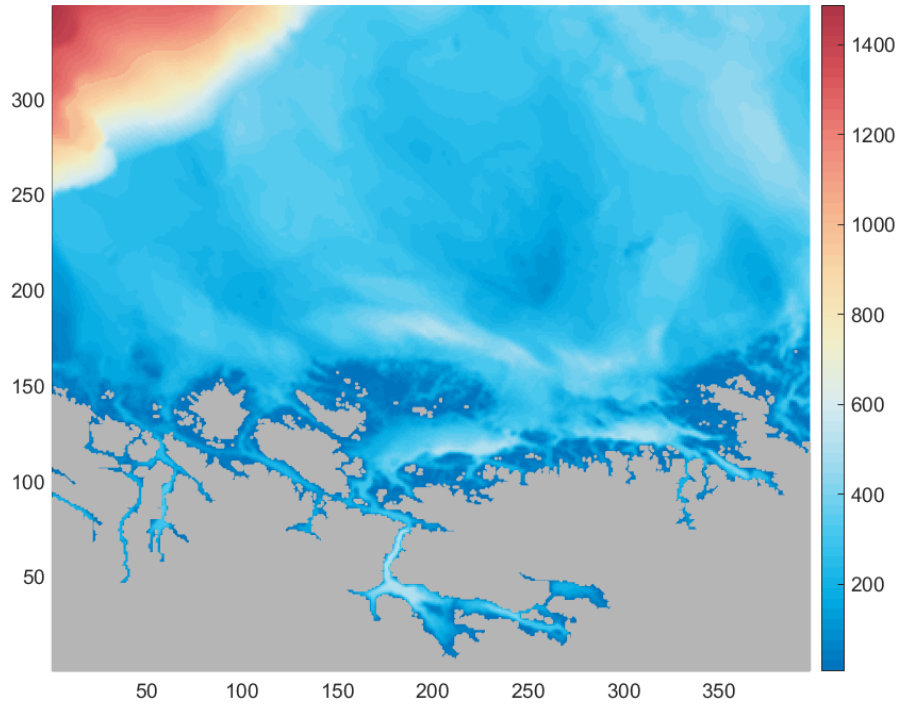

Figure 1: Bathymetry field (depth in m) for the model area. The numbers on the axes indicate grid cells.

2 The NetCDF data format

NetCDF is a set of software libraries and platform independent data formats designed for scientific data, and the most commonly used format for ocean model data. On Windows, NetCDF can be installed by downloading a binary (.exe) package from `https://www.unidata.ucar.edu/downloads/netcdf/` (choose NetCDF-4). On Linux, you can typically install it using your distribution's package manager.

2.1 File structure

After installing NetCDF, you should have access to the **ncdump** command, which outputs selected parts or all the contents of a NetCDF file. Called with the **-h** flag, it displays only the headers which show the structure of the file. Calling **ncdump -h** on our dataset gives the following output:

```
netcdf samples {
dimensions:
        xc = 400 ;
        yc = 350 ;
        zc = 36 ;
        time = UNLIMITED ; // (217 currently)
variables:
        double time(time) ;
                time:units = "days since 2021-04-01 00:00:00" ;
                time:calendar = "standard" ;
                time:standard_name = "time" ;
                time:_CoordinateAxisType = "Time" ;
        int grid_mapping ;
                grid_mapping:straight_vertical_longitude_from_pole = 58. ;
                grid_mapping:horizontal_resolution = 800. ;
                grid_mapping:latitude_of_projection_origin = 90. ;
                grid_mapping:longitude_of_projection_origin = 58. ;
                grid_mapping:standard_parallel = 60. ;
                grid_mapping:origoRef = 0., 0. ;
                grid_mapping:false_easting = 2226000. ;
                grid_mapping:false_northing = 1916000. ;
                grid_mapping:scale_factor_at_projection_origin = 1. ;
        float LayerDepths(zc) ;
                LayerDepths:units = "m" ;
                LayerDepths:standard_name = "cell_thickness" ;
        float xc(xc) ;
                xc:units = "meter" ;
                xc:standard_name = "projection_x_coordinate" ;
                xc:_CoordinateAxisType = "GeoX" ;
        float yc(yc) ;
```

```
        yc:units = "meter" ;
        yc:standard_name = "projection_y_coordinate" ;
        yc:_CoordinateAxisType = "GeoY" ;
float zc(zc) ;
        zc:units = "m" ;
        zc:positive = "down" ;
        zc:standard_name = "depth" ;
        zc:_CoordinateAxisType = "Height" ;
float depth(yc, xc) ;
        depth:standard_name = "depth" ;
        depth:units = "m" ;
        depth:SINMODVarID = 351 ;
        depth:grid_mapping = "grid_mapping" ;
        depth:_FillValue = 9.96921e+36f ;
float DXxDYy(yc, xc) ;
        DXxDYy:standard_name = "gridSize" ;
        DXxDYy:units = "m2" ;
        DXxDYy:SINMODVarID = 354 ;
        DXxDYy:grid_mapping = "grid_mapping" ;
        DXxDYy:_FillValue = 9.96921e+36f ;
float gridLats(yc, xc) ;
        gridLats:SINMODVarID = 21 ;
        gridLats:units = "degrees_north" ;
        gridLats:standard_name = "latitude" ;
        gridLats:axis = "Y" ;
        gridLats:_CoordinateAxisType = "Lat" ;
float gridLons(yc, xc) ;
        gridLons:SINMODVarID = 22 ;
        gridLons:units = "degrees_east" ;
        gridLons:standard_name = "longitude" ;
        gridLons:axis = "X" ;
        gridLons:_CoordinateAxisType = "Lon" ;
short u-velocity(time, zc, yc, xc) ;
        u-velocity:units = "m/s" ;
        u-velocity:scale_factor = 0.0001831111f ;
        u-velocity:add_offset = 0.f ;
        u-velocity:_FillValue = -32768s ;
        u-velocity:missing_value = -32768s ;
short v-velocity(time, zc, yc, xc) ;
        v-velocity:units = "m/s" ;
        v-velocity:scale_factor = 0.0001831111f ;
        v-velocity:add_offset = 0.f ;
        v-velocity:_FillValue = -32768s ;
        v-velocity:missing_value = -32768s ;
short u-wind(time, yc, xc) ;
        u-wind:units = "m/s" ;
```

```
        u-wind:scale_factor = 0.006103702f ;
        u-wind:add_offset = 0.f ;
        u-wind:_FillValue = -32768s ;
        u-wind:missing_value = -32768s ;
short v-Wind(time, yc, xc) ;
        v-Wind:units = "m/s" ;
        v-Wind:scale_factor = 0.006103702f ;
        v-Wind:add_offset = 0.f ;
        v-Wind:_FillValue = -32768s ;
        v-Wind:missing_value = -32768s ;
short w_velocity(time, zc, yc, xc) ;
        w_velocity:units = "m/s" ;
        w_velocity:scale_factor = 0.000122074f ;
        w_velocity:add_offset = 0.f ;
        w_velocity:coordinates = "time zc yc xc" ;
        w_velocity:_CoordinateAxes = "time zc yc xc" ;
        w_velocity:grid_mapping = "grid_mapping" ;
        w_velocity:_FillValue = -32768s ;
        w_velocity:missing_value = -32768s ;
short temperature(time, zc, yc, xc) ;
        temperature:standard_name = "sea_water_temperature" ;
        temperature:units = "K" ;
        temperature:scale_factor = 0.004882962f ;
        temperature:add_offset = 160.f ;
        temperature:coordinates = "time zc yc xc" ;
        temperature:_CoordinateAxes = "time zc yc xc" ;
        temperature:grid_mapping = "grid_mapping" ;
        temperature:_FillValue = -32768s ;
        temperature:missing_value = -32768s ;
short salinity(time, zc, yc, xc) ;
        salinity:standard_name = "sea_water_salinity" ;
        salinity:units = "ppt" ;
        salinity:scale_factor = 0.001525925f ;
        salinity:add_offset = 50.f ;
        salinity:coordinates = "time zc yc xc" ;
        salinity:_CoordinateAxes = "time zc yc xc" ;
        salinity:grid_mapping = "grid_mapping" ;
        salinity:_FillValue = -32768s ;
        salinity:missing_value = -32768s ;
short elevation(time, yc, xc) ;
        elevation:standard_name = "sea_surface_height_above_sea_level" ;
        elevation:units = "m" ;
        elevation:scale_factor = 0.0003051851f ;
        elevation:add_offset = 0.f ;
        elevation:_FillValue = -32768s ;
        elevation:missing_value = -32768s ;
```

```
// global attributes:
                :grid_mapping_name = "polar_stereographic" ;
                :Conventions = "CF-1.5" ;
                :title = "Current data from SINMOD" ;
                :institution = "SINTEF Ocean" ;
                :grid_mapping = "grid_mapping" ;
                :straight_vertical_longitude_from_pole = 58. ;
                :horizontal_resolution = 800. ;
                :coordinate_north_pole = 2781., 2393.5 ;
                :latitude_of_projection_origin = 90. ;
                :standard_parallel = 60. ;
                :barotropic_timestep = 4.13793103448276 ;
                :baroclinic_timestep = 120. ;
                :_FillValue = -32768s ;
                :setup = "mids" ;
                :relax_e = "T" ;
                :nested = "T" ;
                :tidal_input = "F" ;
                :DHA = 1.f ;
                :smagorin = "T" ;
                :biharmonic = "F" ;
                :KBi = 6000000.f ;
                :COLDSTART = "F" ;
                :ATMODATA = 5 ;
                :CM = 1.f ;
                :CM2D = 1.f ;
                :CH = 0.3f ;
                :CI = 5.f ;
                :icedyn = "F" ;
                :tidal_components = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ;
```

The **dimensions** section gives the names and sizes of all variable dimensions used in the file. Here, three dimensions (**xc**, **yc** and **zc**) are used to define the 3D grid of the ocean model, and one (**time**) is used for the time steps in the file. The latter dimension is defined as **UNLIMITED**, meaning it can be expanded as more data is added to the file. Dimensions do not contain data, but are used to define the size of variables.

The **variables** section lists the names and properties of all the variables in the file. A variable can be scalar, or it can be defined on an n-dimensional grid by using one or more of the dimensions defined for the file. For instance, **depth** is a 2D grid of the model's bathymetry, which is only stored a single time (since it isn't time dependent), and is therefore defined using the **xc** and **yc** dimensions (dimensions are listed in reverse order in the **ncdump** output). The **temperature** variable is a 3D field that varies with time, and is therefore defined using the **xc**, **yc**, **zc** and **time** dimensions.

Each variable has a data type (displayed before the name) and any number of named attributes. Attributes can give standard names (which can be used by 3rd party software to understand the meaning of the variable), units, fill values (used to denote cells where a variable is undefined, such as dry cells for an ocean model), offsets and scale factors. The latter two attributes, if defined, are important since the offset and scale factor must be applied to get the appropriate values when reading a variable (correct value = file value × **scale_factor** + **add_offset**). If they are not defined, the stored values can be used as they are.

The **global attributes** section gives additional metadata. For this file, it states that the file was produced by SINMOD, details the map projection used, describes that the CF-1.5 conventions were used (a standard for files to be structured so they are understandable by 3rd party software) in addition to giving some parameter values used in the SINMOD simulation.

2.2 Reading NetCDF files

You can get libraries for reading (and writing) NetCDF files in any programming language. Matlab supports NetCDF through the set of functions with the prefix **netcdf**, e.g. **netcdf.open** and **netcdf.getVar**. Type the command **help netcdf** to get more information about these functions. The process of reading a variable consists of opening a file (using **netcdf.open**), finding a variable's **id** (using **netcdf.inqVarID**) and reading the desired slice of the variable (using **netcdf.getVar**.

Matlab also provides simpler commands for quick access to NetCDF data, such as **ncread** that takes a filename, a variable name and optionally additional arguments, and performs the entire process of opening the file, finding and reading the variable, and closing the file. The quick commands are more convenient in many cases, but in other cases it is useful to have the fine grained control provided by the standard NetCDF functions.

As part of this exercise, you have been provided with some Matlab example code. The script **readdata** gives an example of the process of opening and reading data. A couple of helper functions (**getVariable**, **getProfile** and **getTimeSeries**) have been made to simplify the process of reading 2D fields, vertical profiles and time series from the data file. When working on the following exercises, you are advised to use **readdata** as a starting point, and develop your own scripts that use the helper functions to read and plot the necessary data.

3 Exercises

3.1 Visualize dynamics

To get a feeling for how ocean dynamics work, it is very effective to visualize how the current fields or temperature/salinity fields develop over time. This is fairly easy to do in Matlab by plotting successive time steps and creating video files:

1. Create a Matlab figure.
2. Start a `VideoWriter` (e.g. `v = VideoWriter('file.avi'); v.FrameRate = 10; open(v);`).
3. Loop through the time steps of the data file.
4. For each time step, read model fields using the `getVariable` function
5. Plot into the open figure using `pcolor` or `quiver` (or both, e.g. plotting current speed as a color map with quiver arrows on top). If plotting with `pcolor`, make sure to keep the range of the color bar consistent between time steps (using `caxis`).
6. Write frame to the video: `frame = getframe(gcf); writeVideo(v, frame);`
7. After the loop finishes, close the video writer: `close(v);`

3.2 Tidal current

Trondheimsfjorden is fully within the model area, and the narrowest part of the inlet goes straight along the y direction in the model setup. This is a convenient location to take a look at the tidal current into and out of the fjord.

Check whether the current at this location is mainly driven by the pressure gradient caused by a gradient in elevation:

1. Extract the time series for the v current component at a chosen point (e.g. 176, 56).
2. Calculate the acceleration of the v component based on its rate of change.
3. Find the time series for $\frac{\partial E}{\partial y}$ based on the difference in elevation between two nearby points (one on each side of the first point you chose. You will have to scale the result appropriately, taking the model resolution and distance in grid points into account.
4. Calculate the time series for pressure gradient $\frac{\partial p}{\partial y}$ caused by $\frac{\partial E}{\partial y}$, and the acceleration that would result from the pressure gradient (given by the momentum equation if you disregard advective, coriolis and viscosity terms).

5. Compare the calculated acceleration with the one observed in the v time series.

Check whether the local acceleration seems to be mainly driven by the slope in elevation. What other factors may be contributing to the local acceleration rate?

Do the same analysis for a point in the open ocean (e.g. 200, 250). Can you conclude the same way for this point? If not, what are the main differences between the locations. (*Hint: for part of the answer, consider the Coriolis terms*).

3.3 Bathymetry and currents

You can use the function *plotWithContours* to plot a current field (given by the u and v fields) overlaid with bathymetry contours. For this model area, you could for instance plot the 100 m, 250 m and 500 m contours to get an impression of the area's important features.

Take a look at the relation between the bathymetry and the currents in the area:

1. Plot the current field at a couple of selected time steps, e.g. from layer 10 in the model (to avoid the strongest impact of wind forcing). Can you see clear patterns of topographic steering? How consistent are these between different times?

2. Compute the averaged current field over the entire period (by averaging the u and v fields separately). Can you see clearer patterns in the averaged currents? If so, why is this?

3. Fig. 2 shows an overview of characteristic current patterns in a part of the model area. Do you recognize patterns from the figure in the SINMOD dataset?

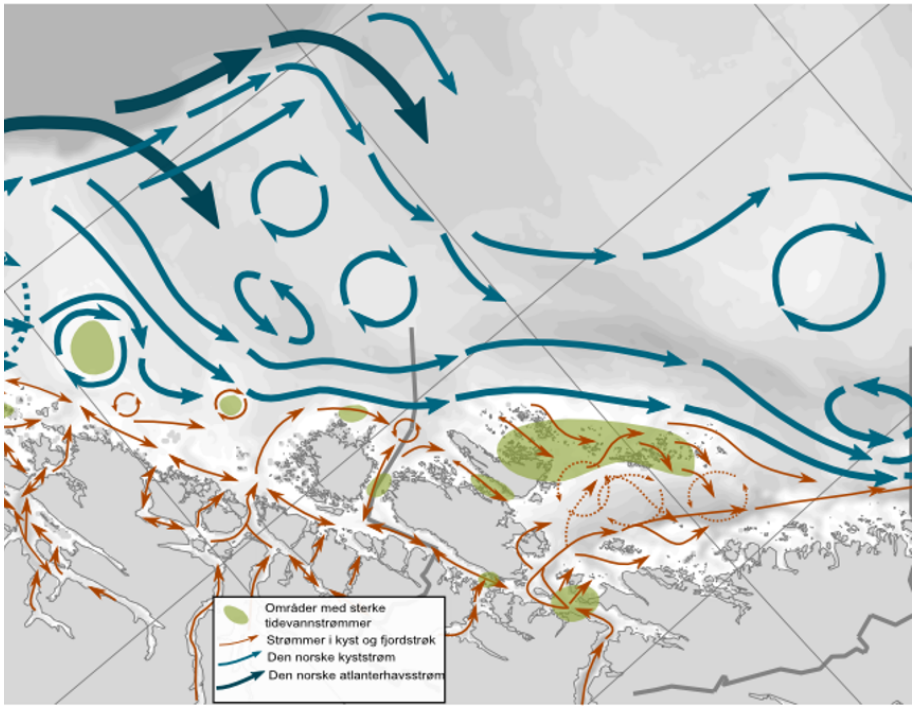

Figure 2: Map of characteristic current patterns outside Middle Norway. Map by Ingrid H. Ellingsen, SINTEF Ocean.

3.4 Wind and vertical profiles

Wind imposes forcing on the ocean surface, which in turn affects current speeds and directions in the upper part of the water column. At high wind speeds the influence can go more than 100 m down. The wind fields are recorded in the SINMOD dataset along with the model output, allowing us to look more closely at the relation between wind and currents.

An example of a strong wind event is in the evening of April 5 (around sample 115), and (200, 300) is a possible location to look at. To compute speeds from components you can use Pythagoras ($\sqrt{u^2 + v^2}$), and to compute directions you can use the Matlab function **atan2**.

1. Plot time series of the current speed at a chosen location in the surface layer and a layer further down. Compare it to the time series of the wind speed in the same location (*hint: you can plot 2.5% of the wind speed in the same figure as current speed to get reasonably similar values*). Do you see signs that the wind is affecting current speed?

2. Plot vertical profiles of wind speed, wind direction, temperature and salinity before, during and after (e.g. 24 hours before and after) the strongest wind event. Does the strong wind event affect the extent of the upper mixed layer (the upper layer where temperature and salinity is almost constant with depth).

3. Take a look at how the current speed is affected close to the surface, and what the profile of the current direction looks like.

3.5 Comparison with observation data

Some ocean variables can be measured remotely by satellites. The file *obs_sst_ttk17.nc* contains sea surface temperature observations from the data product *METOFFICE-GLO-SST-L4-NRT-OBS-SST-V2*, downloaded for the same period SINMOD has been run, and interpolated to the same grid as the model data. There is one observation per day. The variable *sst* contains the estimated sea surface temperature, and *sst_stdev* contains the estimated uncertainty.

The observation data can be plotted similarly to the SINMOD data, and since the dataset has been interpolated to the SINMOD grid, it is possible to plot direct comparisons (e.g. difference between modelled and observed values).

1. Plot comparisons for a few selected times for a selected area or for the whole model grid, and evaluate where the model fits better or worse with the observations.

2. Choose a few locations, and plot time series from the model data and the observations. You can also plot the interval within ± 1 standard deviation of the observations to compare the measurement uncertainty with the deviation of the model.

References

Slagstad, D. and McClimans, T. A. (2005). Modeling the ecosystem dynamics of the Barents Sea including the marginal ice zone: I. Physical and chemical oceanography. *Journal of Marine Systems*, 58(1-2):1–18.