A Brief Introduction to Running Windsurf

What is Windsurf?

Windsurf is a modeling framework to simulate the co-evolution of the coastal profile in response to both oceanographic and meteorological forcings. The Windsurf framework is still in proof of concept phase, with the first test of this program coded in Matlab. This Matlab implementation uses an offline approach which calls standalone, independent model executables to simulate processes for user-defined time spans (e.g., 1 hour), with Windsurf exchanging information between models at that coupling interval. Currently Windsurf combines:

- XBeach (http://xbeach.org/)
- Coastal Dune Model (CDM) (https://github.com/openearth/cdm)
- Aeolis (https://github.com/openearth/aeolis-python)

Together these models are capable of simulating the predominant hydrodynamic, sediment transport, and morphology change processes relevant for outer coast systems (Figures 1 and 2). Note that two subaerial models (CDM and Aeolis), which having some overlapping capabilities, are currently implemented since both has unique simulation capabilities. CDM incorporates spatio-temporal vegetation growth and its effect on the wind field. Aeolis allows for the ability to simulate multi-fraction aeolian sediment transport.

There are a number of potential future improvements to Windsurf that are envisioned, including an online coupler using the Basic Model Interface and an extension of the model to 2D. However, this document reflects the most current version of the (matlab) Windsurf, as found at:

https://github.com/ncohn/Windsurf

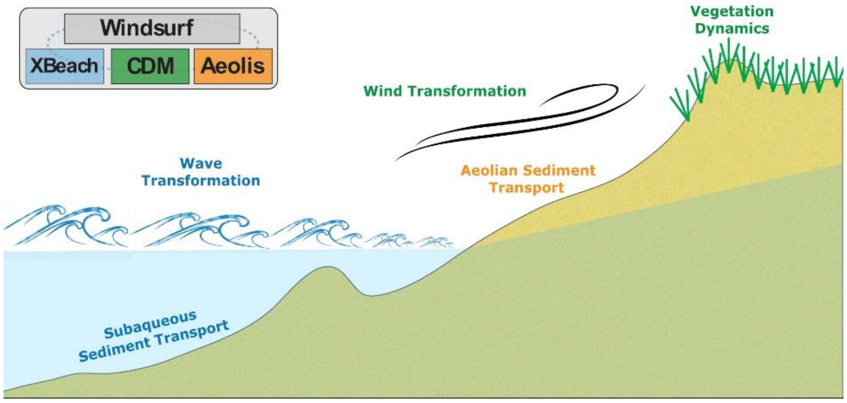

Figure 1. Cross-shore schematic of the Windsurf framework showing the general processes resolved by the model cores.

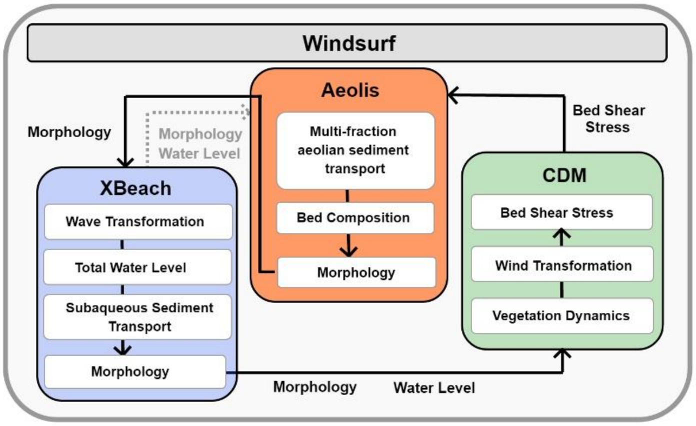

Figure 2. Schematic representing the general Windsurf model framework which includes three standalone numerical models (XBeach, Coastal Dune Model, Aeolis) that are coupled offline through a back-end Matlab interface. Major processes resolved within each model core and the outputs exchanged by the coupler are shown.

Windsurf Computing Requirements

- Matlab – 2016b or more recent needed
- Compiled XBeach, CDM, and Aeolis executables (Windsurf has been tested with the most recent versions of each of these models at the time of writing)
- The program should be operating system agnostic, but has not been tested on Mac OS

Running Windsurf

Input Files

The information needed to run Windsurf is fully contained within a series of matlab structure variables. These structure variables are broken down by model core processes/function and include:

- project
- grids
- sed
- flow
- tides
- waves
- wind
- veg

See the example file for all of the variables included within these structures – though generally they include either project level variables in the outermost structure (e.g., time series of significant wave height is waves.Hs) and model level functions/parameters further down in the hierarchy (e.g., the wave breaking parameter for XBeach would be waves.XB.gamma). Generally only the project level variables need to be changed for a given simulations (with some exceptions). These include:

- project
    - project.Directory
        - where the model will be run
    - project.XB.XBExecutable
        - where the XBeach model executable is located
    - project.CDM.CDMExecutable
        - where the CDM model executable is located
    - project.Aeolis.AeolisExecutable
        - where the Aeolis model executable is located
    - project.timeStep
        - the model coupling time step in seconds. This should be the same as the duration of each environmental input
    - project.flag
        - these variables can be changed to turn on and off indivual models within Windsurf but are not fully tested at the moment. The default configuration is:
            - project.flag.XB = 1 (XBeach turned on)
            - project.flag.CDM = 2 (CDM turned on to provide winds only)
            - project.flag.Aeolis = 1 (Aeolis turned on)
            - project.flag.VegCode = 1 (external vegetation code, not calculated by CDM, is turned on)
            - project.flag.nourishment = 0 (nourishment code is not turned on)
    - project.XB.hydro_spinup
        - this is an added time to every XBeach simulation to allow for waves/currents to spin-up before morphology changes occurs or any outputs are stored. This is related to how far offshore the grid extends, what wave conditions are used, and what morphologic acceleration factor is used.
- grids
    - grids.XGrid
        - cross-shore distances of grid. Offshore most value should be the lowest X value and the first entry. Note all entries are in meters. You can have a variable dx offshore – but should have a constant grid size for all land cells due to CDM requirements.
    - grids.ZGrid
        - elevations following the same convention as grids.XGrid
- sed
    - sed.XB.D10 [note that all grain sizes are in meters]
    - sed.XB.D50
    - sed.XB.D90

    - sed.Aeolis.D10 [note that the Aeolis grain size distribution into Windsurf is still being tested and will improved as the model is not limited to a fixed number of grain size bins as the interface currently has]
    - sed.Aeolis.D50
    - sed.Aeolis.D90
    - sed.CDM.D [this should be the D50]
- tides
    - tides.waterlevel
        - time series of still water level values that have the same time step as project.timeStep
- waves
    - waves.Hs
        - time series of significant wave heights at the offshore boundary that have the same time step as project.timeStep
    - waves.Tp
        - time series of peak wave period at the offshore boundary that have the same time step as project.timeStep
    - waves.D
        - time series of wave direction at the offshore boundary that have the same time step as project.timeStep. Note the XBeach nautical convention is used where 270° is cross-shore (onshore)
- wind
    - winds.windspeed
        - time series of wind speed that have the same time step as project.timeStep
    - winds.winddir
        - time series of wind direction that have the same time step as project.timeStep. Note that the wind direction convention is that 0 degrees is cross-shore (onshore)
    -
    - winds.threshold
        - This is here to save some computational expense if it is known that aeolian transport below some threshold velocity is negligible. If velocities are below this value the aeolian transport model will not run.
- Veg
    - veg.CDM.elevMin
        - The minimum elevation that vegetation can grow
    - veg.CDM.maximumDensity
        - The maximum density (0 to 1) that vegetation can grow

In the example file “windsurf_example_case.mat” has all of these variables set up with (mostly) model default values. This is a very simple case just meant to show that the model inputs look like and that the model framework runs. This can serve as a template for more complex cases, but note that model tuning will likely be necessary as all the models can sensitive to model input choices.

Matlab Code Structure

Windsurf_Run.m initiates and executes the full Windsurf framework, using the pre-loaded structure variables for all model tasks. This code first initializes the model, such as setting up individual model input files, changing formats compatible with legacy code, and creating a netcdf file for all model output. The model then goes through a loop based on the number of time steps and (if all model processes are turned on) goes through a sequence of running XBeach, followed up the vegetation code, CDM, and Aeolis, and then saving relevant model output at the end of each coupling time step (Figure 3). There are a number of additional sub-functions not explained here called by these codes as well. To run Windsurf, you should add all codes to your path (e.g., addpath(genpath(PATH TO CODE))), load the structure variables to the workspace, update project.Directory and local paths to model executables, and simply type in Windsurf_Run. The models should start running if things are set up appropriately.

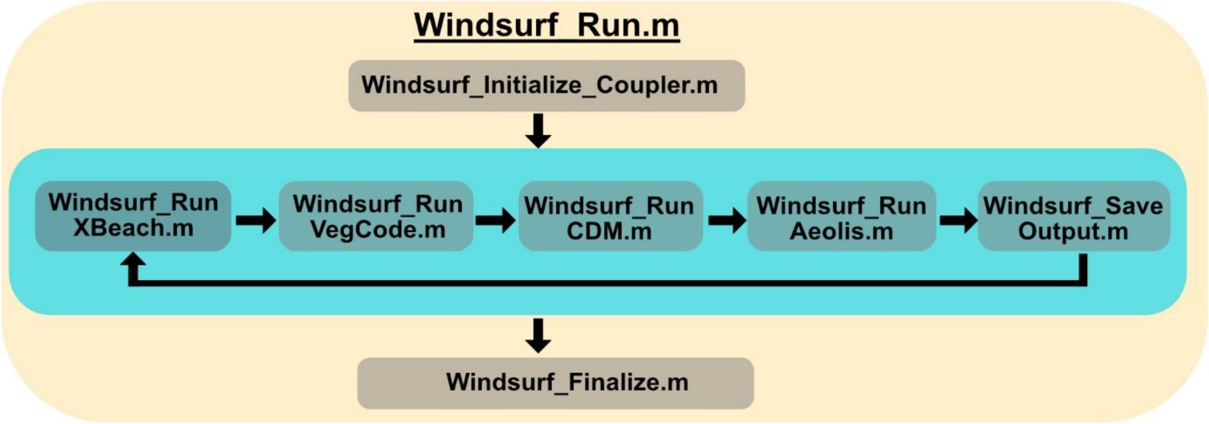

Figure 3. Schematic of the Windsurf command order

Model Output

Windsurf saves model output to windsurf.nc. If you want to view the resulting Windsurf predicted profile change you can read the output via:

x = ncread(‘windsurf.nc’, ‘x’);

zb = ncread(‘windsurf.nc’, ‘zb’);

plot(x,zb)

The model derived bed level changes from waves and winds can be accessed via the netcdf varaibles “dz_wave” and “dz_wind”, respectively.