How to run spaghetti on a Linux system

Author: Bjørn Ådlandsvik
Date: 2010-08-03

Introduction

This document describes how to run the spaghetti system for geolocation of fish tags. It focus on how to set up and run the system for geolocation of salmon from northern Norway.

The system (for cod in the Barents Sea) is described scientifically in:

```
B. Ådlandsvik, G. Huse and K. Michalsen, 2007,
Introducing a method for extracting horizontal migration patterns
from data storage tags,
Hydrobiologia, 582, 187-197.
```

Software requirements

- Fortran 90 compiler (free gfortran from GNU works nicely)
- NetCDF library for Fortran
- python
- numpy
- netCDF for python (preferably netcdf4-python)

The first four are typically available as packages for a Linux distribution. The netcdf4-python is a bit more tricky to install. Other netcdf libraries for python can be used instead with only minor modifications.

For postprosessing with python, the matplotlib library is recommended. Other tools such as Matlab can be used instead.

Directory setup

I setup a main directory for running the system. The fortran code is put into a subdirectory called "fortran". The input data: tag, temperature, topography is placed in a subdirectory called "input". The merging program, merge.py, is put into the main directory.

Input set-up

This describes the input needed for running the model for tracking salmon in norther Norway.

Tag data

Text file with 4 columns: date, time, sea surface temperature, max depth Data are hourly, SST is mean of surface obs. during that hour or 9999.00 if not at surface, and max depth is maximum depth recorded during the hour

Ole Petter has a Matlab script that produces this format. Python script "smooth.py" in input directory can be used to smooth the temperature data, produces a new file with the same format.

Format example:

```
2008-05-26 20:00:00     5.02   10.76
2008-05-26 21:00:00     6.04    0.00
2008-05-26 22:00:00     6.04    0.00
2008-05-26 23:00:00     6.15    5.38
2008-05-27 00:00:00  9999.00   10.76
2008-05-27 01:00:00     5.83   16.14
2008-05-27 02:00:00     5.83   16.14
```

SST data

Have a file oisst_2008.nc per year. This file is in netcdf format created by python script sst2nc. Data fra "NOAA Optimum Interpolation (OI) Sea Surface Temperature (SST) V2" http://www.esrl.noaa.gov/psd/data/gridded/d

Topography data

From a single column data file, etopo2_codyssey.dat With etopo2-data covering the Barents Sea

Fortran setup

There are different fortran-modules matching the input,

- sst_module in sst.f90
- topography in topo.f90
- tagdata in tagdata.f90

These modules are meant to meatch the input format, and are likely to be changed if the model is to be used for something different than salmon in northern Norway.

The main module to be changed is the position module found in the source file position.f90

Parameters that may be changed:

nFish Number of trajectories

randspeed Random speed, the "noice" in the trajectories

Limiter works by termination if | sst_oisst - sst_tag | > temp_lim, or depth_tag > dfac*depth_etopo + add_depth

NOTE: Sometimes these test are overruled by the code, which is the ultimate guide to what is being done.

After modification of position.f90, type "make" and copy the executable file "spaghetti" to the mother directory

Set-up file

The model run is governed by a set-up file, spaghetti.sup in the main directory Presently it contains:

- start time and position
- end time and position
- name of tag file
- name of output fules
- depth factor
- additiv depth contibution
- temperature error tolerance
- input time step
- output time step

When performing several model experiments it can be useful to make a separate set-up fil, for instance "caseA.sup" to document the settings. A link can be used to give the information to the spaghetti-program:

```
ln -sf caseA.sup spaghetti.sup
```

Running the spaghetti model

The spaghetti executable runs first the forward and thereafter the backwards trajectories. It is run by typing "spaghetti" in the main directory.

When running it displays the number of surviving trajectories

```
...
daynr, #fish =          28      126201
daynr, #fish =          29      123315
...
```

This way one can keep track of how far it has been running and if the trajectories are becoming extinct. The number of trajectories half time, when they are to be merged with the opposite trajectories, is the most important.

Output format from spaghetti

The spaghetti program makes two netcdf-files, one for the forwards and one for the backwards trajectories. The files can be large, with many dead tracks. The files are temporary, meant to be used by the merge program and can be deleted afterwards. The files can however be used for finer study of why and when tracks are terminated.

The format as reported by ncdump:

```
dimensions:
      time = UNLIMITED ; // (179 currently)
      fish = 200000 ;
variables:
      float time(time) ;
```

```
                time:long_name = "time" ;
                time:units = "days since 2008-05-27" ;
        float X(time, fish) ;
                X:long_name = "Longitude" ;
                X:units = "degrees east" ;
        float Y(time, fish) ;
                Y:long_name = "Latitude" ;
                Y:units = "degrees north" ;
        int life(time, fish) ;
                life:long_name = "Flag for active trajectory" ;

// global attributes:
                :type = "Spaghetti forward  output file" ;
```

Running merge

Merge is run by typing:

```
python merge3.py [setup_file]
```

at the command line in the main directory. It should not be necessary to change anything in the program code it should use a setup-file of the same type as the spaghetti run. If no setup file is given as a command line argument, the default "spaghetti.sup" is used.

Output format

The output from the merge script is a NetCDF file, mergeA.nc in the example above. The structure is quite simple as shown by *ncdump -h*:

```
dimensions:
        time = 179 ;
        tracknr = 1654 ;
variables:
        double time(time) ;
                time:long_name = "time" ;
                time:units = "days since 2008-05-27" ;
        float lon(time, tracknr) ;
                lon:long_name = "longitude" ;
                lon:units = "degrees_east" ;
        float lat(time, tracknr) ;
                lat:long_name = "latitude" ;
                lat:units = "degrees_north" ;
        int p(tracknr) ;
                p:long_name = "Forward index" ;
        int q(tracknr) ;
                q:long_name = "Backward index" ;

// global attributes:
                :Conventions = "CF-1.0" ;
                :institution = "Institute of Marine Research" ;
                :source = "The spaghetti geolocation model by Bjørn Åd-
landsvik, IMR" ;
                :history = "created 2010-08-04 by merge2.py from FWA.nc and BWA.nc" ;
```

It has 179 time steps (days) and 1654 trajectories. The positions are given in the lon/lat arrays while p and q gives the forward and backwards track numbers for the merged track.

Post prosessing

The model results can be examined with any software that can read netCDF. For instance plotting the first tracks (but not more than 100) can be done by

```
import matplotlib as plt
from netCDF4 import Dataset

...

# Read the tracks
f = Dataset('mergeA.nc')
N = min(f.nTrack, 100)   # at most 100 tracks
X = f.variables('lon')[:,:N]
Y = f.variables('lat')[:,:N]

...

# Plot the tracks
for i in xrange(N):
    plt.plot(X[:,i], Y[:,i], color='blue')

plt.show()
```

A more elaborate version plotting coast, bathymetry and mean trajectory is given as plotbane2.py in the main directory