

NOAA Technical Memorandum NOS OR&R 41



# General NOAA Operational Modeling Environment (GNOME) Technical Documentation

---

## Data Formats

---

Seattle, Washington  
October 2012

## NOAA's Office of Response and Restoration

NOAA's Office of Response and Restoration (OR&R) is a center of expertise in preparing for, evaluating, and responding to threats to coastal environments, including oil and chemical spills, releases from hazardous waste sites, and marine debris.

To fulfill its mission of protecting and restoring NOAA trust resources, the Office of Response and Restoration:

- Provides scientific and technical support to prepare for and respond to oil and chemical releases.
- Determines damage to natural resources from these releases.
- Protects and restores marine and coastal ecosystems, including coral reefs.
- Works with communities to address critical local and regional coastal challenges.

OR&R is comprised of three divisions: Emergency Response, Assessment and Restoration, and Marine Debris. Collectively, the Office of Response and Restoration provides comprehensive solutions to environmental hazards caused by oil, chemicals, and marine debris.

---

Cite as:

Zelenke, B., C. O'Connor, C. Barker, and C.J. Beegle-Krause (Eds.). (2012). General NOAA Operational Modeling Environment (GNOME) Technical Documentation: Data Formats. *U.S. Dept. of Commerce, NOAA Technical Memorandum NOS OR&R 41*. Seattle, WA: Emergency Response Division, NOAA. 49 pp. [http://response.restoration.noaa.gov/gnome\\_manual](http://response.restoration.noaa.gov/gnome_manual)

NOAA Technical Memorandum NOS OR&R 41

# General NOAA Operational Modeling Environment (GNOME) Technical Documentation

NOAA Office of Response and Restoration  
7600 Sand Point Way NE  
Seattle, WA 98115



October 2012  
Seattle, Washington

---

**U.S. DEPARTMENT OF COMMERCE**

Rebecca M. Blank,  
Acting Secretary

**National Oceanic and  
Atmospheric Administration**

Dr. Jane Lubchenco,  
Under Secretary of Commerce for Oceans  
and Atmosphere and NOAA Administrator

**National Ocean Service**

David Kennedy,  
Assistant Administrator for Ocean  
Services and Coastal Zone  
Management

---

DEPARTMENT OF COMMERCE  
National Oceanic and Atmospheric Administration  
National Ocean Service  
Office of Response and Restoration  
Emergency Response Division  
Technical and Scientific Services Branch  
Seattle, Washington, United States of America

#### NOTICE

This report has been reviewed by the National Ocean Service of the National Oceanic and Atmospheric Administration (NOAA) and approved for publication. Such approval does not signify that the contents of this report necessarily represent the official position of NOAA of the government of the United States, nor does mention of trade names or commercial products constitute endorsement or recommendation for their use.

# Table of Contents

- 1 GNOME Input File Formats ..... 6
  - 1.1 Maps..... 6
    - 1.1.1 BNA Format..... 6
      - 1.1.1.1 Map Bounds ..... 7
      - 1.1.1.2 Spillable Area ..... 8
  - 1.2 Currents..... 8
    - 1.2.1 ASCII Formats ..... 9
      - 1.2.1.1 Currents: Finite Element – Velocities on Triangles, Steady State [CATS] ..... 9
        - 1.2.1.1.1 Example – File Name: *TinyWillapa SAC.CUR*..... 9
          - 1.2.1.1.1.1 Annotated Version of the File ..... 10
      - 1.2.1.2 Currents: Finite Element – Velocities on Nodes [ptCur] ..... 10
        - 1.2.1.2.1 The Header Block ..... 11
        - 1.2.1.2.2 The Point Definition Block ..... 13
        - 1.2.1.2.3 The Topology Block – Optional ..... 15
        - 1.2.1.2.4 The Time-Specific Data Blocks ..... 15
        - 1.2.1.2.5 Example 1 – Filename: *skipptcur.cur* ..... 16
        - 1.2.1.2.6 Example 2 – Filename: *ptCurMap.cur*..... 17
        - 1.2.1.2.7 Example 3 – Filename: *ptCurNoMap.cur*..... 20
      - 1.2.1.3 Currents: Rectangular Grid – Steady State [GridCur]..... 21
        - 1.2.1.3.1 Example 1 – Filename: *GridCurExA.cur* ..... 22
        - 1.2.1.3.2 Example 2 – Filename: *GridCurExB.cur* ..... 22
        - 1.2.1.3.3 Explanation of File Components ..... 22
      - 1.2.1.4 Currents: Rectangular Grid – Time Dependent [GridCurTime]..... 23
        - 1.2.1.4.1 Data in a Single File ..... 23
          - 1.2.1.4.1.1 Example – Filename: *gridcurTime.cur* ..... 23
        - 1.2.1.4.2 Data in Multiple Files ..... 24
          - 1.2.1.4.2.1 Example 1 – Filename: *gridcurtime\_hdr.cur*..... 25
          - 1.2.1.4.2.2 Example 2 – Filenames: *gridcurtime\_hdrA.cur*, *gridcurtime\_hdrB.cur*, and *gridcurtime\_hdrC.cur*..... 25
    - 1.2.2 NetCDF Formats ..... 26

1.2.2.1	NetCDF Rectangular Grid .....	26
1.2.2.2	NetCDF Curvilinear Grid .....	27
1.2.2.3	NetCDF Triangular Grid .....	28
1.2.2.3.1	Example – Triangular Grid Format with Velocities on the Nodes.....	28
1.2.2.3.2	Example – Triangular Grid Format with Velocities on the Triangles .....	30
1.2.2.4	Data in Multiple NetCDF Files: When Your NetCDF Files Start To Get Too Big.....	30
1.2.2.4.1	Example 1 – Filename: <i>MyMasterFileEx.txt</i> .....	31
1.2.3	Scaling Current Patterns .....	31
1.2.3.1	Time-Series File Formats.....	31
1.2.3.1.1	Example – Filename: <i>SouthBend.txt</i> .....	32
1.2.3.1.2	Time Series of Current Magnitude.....	32
1.2.3.1.2.1	Example – Filename: <i>SouthBend.ossm</i> .....	33
1.2.3.1.2.1.1	Annotated Version of the File.....	33
1.2.3.1.3	SHIO Movers: Using Tidal Constituents .....	33
1.2.3.1.3.1	Tidal Heights Constituent Record .....	33
1.2.3.1.3.1.1	Example – Filename: <i>HornIslandPass.shio.txt</i> .....	34
1.2.3.1.3.2	Tidal Currents Constituent Record .....	34
1.2.3.1.3.2.1	Example – Filename: <i>StJohnsRiver.shio.txt</i> .....	35
1.2.3.1.3.2.2	Example – Filename: <i>Edmonds.shio.txt</i> .....	36
1.2.3.1.4	Hydrology Time-Series.....	36
1.2.3.1.4.1	Example – Filename: <i>Hillsborough.HYD</i> .....	37
1.2.3.1.4.1.1	Annotated Version of the File.....	37
1.3	Winds .....	37
1.3.1	Winds: Single Point, Time Series [OSSM].....	38
1.3.1.1	Example – Filename: <i>OSSM Format.WND</i> .....	38
1.3.1.1.1	Annotated Version of the File.....	38
1.3.2	Winds: Rectangular Grid, Time Series [GridWindTime].....	39
1.3.2.1	Example – Filename: <i>GridWindTime.wnd</i> .....	39
1.3.3	Winds: NetCDF Rectangular Grid, Time Series.....	39
1.3.4	Winds: NetCDF Curvilinear Grid .....	40
1.3.5	Data in Multiple Files: When your NetCDF files start to get too big.....	41
1.3.5.1	Example – Filename: <i>MyMasterFileEx.txt</i> .....	41

2	GNOME Output File Formats .....	42
2.1	MOSS Files for GIS Systems.....	42
2.2	NetCDF LE Output File Format .....	42
2.2.1	Example – Contents of NetCDF LE File from Whole 2-D Model Run .....	42
2.2.2	Example – Contents of NetCDF LE File from Whole (pseudo)3-D Model Run .....	44
3	GNOME and GNOME Analyst.....	45
3.1	Custom Logo on Output.....	45



# General NOAA Operational Modeling Environment (GNOME) – Data Formats

---

*National Oceanic & Atmospheric Administration (NOAA)*

In this document, we describe a number of files that can be used or generated by GNOME, version 1.3.5. GNOME uses the following files:

- Maps:** GNOME uses maps to determine the shoreline where the spilled material (typically oil) beaches. Any area of the model domain not defined as “land” is viewed as “water” by GNOME.
- Currents:** Currents move the oil around, and are defined on a particular grid. GNOME recognizes finite element, rectangular, and curvilinear grid circulation models in both American Standard Code for Information Interchange (ASCII) and Network Common Data Form (NetCDF) file formats. Finite element models may have either the velocities on the triangles or on the nodes. Both steady state  $\vec{U}(x, y)$  and time-dependent  $\vec{U}(x, y, t)$  circulation models are supported. A steady state  $\vec{U}(x, y)$  model “current pattern” may be adjusted (“scaled”) and given time dependence through a time-series  $T(t)$  (please see §1.2.3 Scaling Current Patterns). If there is a gap between the circulation grid and the map, GNOME will assume the gap is water – and other movers, such as wind and diffusion, will be able to move the oil around.
- Winds:** Winds may be entered as a constant time series  $T(t)$  or as a regular grid time-dependent model  $\vec{U}_{wind}(x, y, t)$ . Both NetCDF and ASCII formats are supported for wind model data.

Table 1 lists the circulation and wind models that we have converted into GNOME format, and our recommended format.

Table 1. Circulation and wind models that have been converted into GNOME format.

Model	Grid	GNOME Format
CATS	Triangle, velocities on triangles	CATS
POM	Curvilinear	NetCDF
POM	Regular grid	GridCur; GridCurTime; NetCDF
ROMS	Curvilinear	NetCDF
ADCIRC	Triangle, velocities on nodes	PtCur; NetCDF
FVCOM	Triangle, velocities on triangles	NetCDF
RMA2	Polygons – break down into triangles	PtCur
SWAFS	Regular grid	GridCur; GridCurTime; NetCDF
ARPS	Regular grid	GridCur; GridCurTime; NetCDF
HirLAM	Regular grid	GridCur; GridCurTime; NetCDF
HF Radar	Rotated regular grid – use curvilinear	NetCDF

GNOME also outputs files. Maps, as well as standard Map Overlay Statistical System (MOSS) files for geographic information system (GIS) programs, can be exported from GNOME.

The output from both GNOME and GNOME Analyst (an in-house companion program to GNOME that displays oil concentration contours) may be customized by adding your own logo. Instructions for adding your logo are provided in §3.1.

## 1 GNOME Input File Formats

### 1.1 Maps

Currently, GNOME uses only the boundary file atlas (BNA) map file format.

#### 1.1.1 BNA Format

The BNA format consists of a list of lines and polygons that are to be drawn on the screen. Each feature is preceded by a description line, such as the line shown below, from the example file *simple.bna*.

```
"2","1",18
```

The first number in quotes represents an identifier for the feature, and is usually unique.

The second number in quotes identifies the type of feature: “1” is a land feature; “2” is a water feature, or a polygon within another larger polygon.

The third number is the number of points in the feature, in order for drawing. A positive number indicates a polygon. Points are defined in a clockwise direction as you trace the land boundary (as though you are walking on an imaginary beach with your left foot on land and your right foot in the water). A negative number defines a line where the start and end points don’t connect.

**File Name:** *simple.bna*

```
"2","1",18
-82.521416,27.278500
-82.552109,27.353674
-82.564636,27.383394
-82.600746,27.500633
-82.576721,27.581442
-82.541473,27.665442
-82.478104,27.725504
-82.443367,27.755222
-82.250000,27.730673
-82.250000,27.685675
-82.250000,27.640678
-82.250000,27.595680
-82.250000,27.505688
-82.250000,27.460690
-82.250000,27.415693
-82.250000,27.370695
-82.351616,27.278500
-82.453232,27.278500
"2","1",10
-82.250000,27.865969
-82.333580,27.864744
-82.383003,27.879385
-82.479012,27.888107
-82.543144,27.952902
-82.456032,28.066999
-82.405220,28.066999
-82.354408,28.066999
-82.250000,27.977007
-82.250000,27.898989
```

Two special types of polygons are defined for GNOME maps:

1. a map boundary for non-rectangular maps and
2. a spillable area.

These special polygons are most commonly used in Location Files to help users avoid setting spills in areas where the Location File has not been set-up or well calibrated.

### **1.1.1.1 Map Bounds**

The map bounds define a polygon with a format similar to that shown above. This polygon should be the first polygon in the map file.

```
"Map Bounds","1",7
-121.319176,35.199476
-121.319176,34.197944
-121.218496,34.0
-119.378944,34.0
-119.221448,34.152428
-119.221448,35.199476
-121.319176,35.199476
```

### 1.1.1.2 Spillable Area

The spillable area defines a polygon so that the user may not start spills outside the polygon, or over land areas within the polygon. Again, the format is similar to other polygons in the BNA format. This polygon should be the last one defined in the map file.

```
"SpillableArea", "1", 15
-121.319176,35.199476
-121.319176,34.197944
-121.218496,34.0
-120.633640,34.0
-120.445584,34.088112
-120.381776,34.085196
-120.204512,34.026884
-120.066248,34.053124
-119.931528,34.061872
-119.729456,34.015220
-119.534464,34.047292
-119.378944,34.0
-119.221448,34.152428
-119.221448,35.199476
-121.319176,35.199476
```

## 1.2 Currents

GNOME supports steady-state circulation models that produce “current patterns,” as well as time-dependent circulation models. With time-dependent models, the data can be contained in a single file or broken into smaller files. GNOME uses both ASCII and NetCDF file formats, though not all grid types are supported in both formats. Following are the types of circulation models that GNOME supports:

*Finite Element* – Velocities on Triangles, Steady State [CATS] or Time Dependent (see §1.2.1.1 and §1.2.2.3).

*Finite Element* – Velocities on Nodes, Steady State or Time Dependent [ptCur and NetCDF] (see §1.2.1.2 and §1.2.2.3).

*Rectangular Grid* – Steady State [GridCur and NetCDF] (see §1.2.1.3 and §1.2.2.1)

*Rectangular Grid* – Time Dependent [GridCurTime and NetCDF] (see §1.2.1.4 and §1.2.2.4).

*Curvilinear Grid* – Time Dependent [NetCDF only] (see §1.2.2.2).

Current patterns can be adjusted, or “scaled,” and time dependence can be added, connecting the patterns to a time-series. Time-series used for scaling currents can be of the following types:

- tidal currents
- tidal height (GNOME takes the first derivative)
- wind and hydrological flow volume

Tidal current and tidal height time-series can also be represented by the tidal harmonic series. In this case, GNOME calculates the necessary tidal information from the harmonic constants for the start time

of the model run. For long simulations or Location Files, this results in using a smaller file size than the full time-series.

## 1.2.1 ASCII Formats

### 1.2.1.1 Currents: Finite Element – Velocities on Triangles, Steady State [CATS]

For more information about file formats from the NOAA Current Analysis for Trajectory Simulation (CATS) hydrodynamic model, please see the specific documentation for that application.

Note: Beginning with GNOME version 1.2.2, we added the capability to generate a Directional Acyclic Graph (DAG) Tree within GNOME, so that a portion of the current file (viz. the final section of the file, marked DAGTree) is now optional.

#### 1.2.1.1.1 Example – File Name: *TinyWillapa SAC.CUR*

```
DAG 1.0
Vertices 8
8      8
-124.018048    46.694592    1.000000
-124.044816    46.668488    1.000000
-124.017968    46.650984    1.000000
-123.992400    46.664772    1.000000
-123.964264    46.646212    1.000000
-123.929744    46.673788    1.000000
-123.956592    46.696068    1.000000
-123.991760    46.683868    1.000000
Topology 6
0      1      7      5      -1      -1      0.502367      -0.298270
1      2      3      -1     5      -1      0.000000      -0.000000
3      4      5      -1     4      -1      0.000000      -0.000000
5      6      7      -1     4      -1      0.588724      0.297317
7      3      5      2      3      5      0.978753      0.205045
7      1      3      1      4      0      0.971727      -0.100222
DAGTree 13
32     1      7
31     2      5
30     -8     3
2      4      -8
0      -8     -8
7      6      -8
6      -8     -8
26     8      11
25     -8     9
12     10     -8
13     -8     -8
18     12     -8
19     -8     -8
```

### 1.2.1.1.1.1 Annotated Version of the File

```

DAG 1.0
Vertices 8      Number of Vertices
8      8      Number of Vertices, Repeated
Longitude Latitude Depth
-124.018048 46.694592 1.000000
-124.044816 46.668488 1.000000
-124.017968 46.650984 1.000000
-123.992400 46.664772 1.000000
-123.964264 46.646212 1.000000
-123.929744 46.673788 1.000000
-123.956592 46.696068 1.000000
-123.991760 46.683868 1.000000
Topology 6      Number of Triangles
      Points in Tri Adjacent Tri to Seg Velocity (u,v) Tri #
0      1      7      5      -1      -1      0.502367      -0.298270      0
1      2      3      -1      5      -1      0.000000      -0.000000      1
3      4      5      -1      4      -1      0.000000      -0.000000      2
5      6      7      -1      4      -1      0.588724      0.297317      3
7      3      5      2      3      5      0.978753      0.205045      4
7      1      3      1      4      0      0.971727      -0.100222      5
DAGTree 13      Number of Elements in DAGTree
      Seg# Branch Left Branch Right DAGTree Branches
32      1      7      0
31      2      5      1
30      -8      3      2
2      4      -8      3
0      -8      -8      4
7      6      -8      5
6      -8      -8      6
26      8      11      7
25      -8      9      8
12      10      -8      9
13      -8      -8      10
18      12      -8      11
19      -8      -8      12

```

### 1.2.1.2 Currents: Finite Element – Velocities on Nodes [ptCur]

Most finite element circulation models calculate velocities on the triangular grid nodes. The *ptCur* format can be used to make a single time-step “current pattern” or include the full model run time-series. The format can be divided into several portions: header block, point definition block, topology, and time-specific data blocks. The header block provides basic information about the file, and much of the information is optional. The point definition block includes all the points, organized with the boundary points first. The topology block defines the triangular topology and segment list, and the DAGtree defines how to search through the triangles quickly. (These blocks are optional, as GNOME can calculate all this information; although loading the file will take longer without it.) The time-specific data blocks make up the velocity data.

**Note:** There are two different forms of the *ptCur* data format. The first has velocity values at all of the points, including the boundaries. In the second case, the original circulation model does not specifically define the boundary points, and defining these points may be too time-consuming for the user. In this second case, fake boundary points may be defined that have zero velocity at these nodes. A keyword in the *VERTICES* line notifies GNOME that the first

*NumLandPts* have zero velocities, and these points do not show up in the velocity data (i.e., the velocity data start with point *NumLandPts+1*).

### 1.2.1.2.1 The Header Block

The header block is made up of lines that are initiated with a reserve word, which is enclosed in square brackets and is all caps, followed by a tab and the corresponding value. Each of these lines provides some global information about the file, and all but the first two are optional. The other lines have default values that GNOME provides. Except for the first line, the order of header lines is not important; however, if the keyword is in the file, a value must follow it, even if it matches the default value. Table 2 lists the supported header lines.

Table 2. Lines supported in the header block.

Reserve Word	Definition	Data Type	Necessity
[FILETYPE]	"PTCUR"	text	required
[NAME]	"user name for file"	text	optional
[CURSCALE]	<i>multiplicative_scale</i>	float	optional
[UNCERTALONG]	<i>along_axis</i>	float	optional
[UNCERTCROSS]	<i>cross_axis</i>	float	optional
[UNCERTMIN]	<i>min_current_resolution</i>	float	optional
[GRIDTYPE]	<i>vertical model used, bottom bc</i>	text	optional
[MAXNUMDEPTH]	<i>max depth values</i>	int	optional
[USERDATA]	"non GNOME user info"	text	optional

[FILETYPE] is an identifier for GNOME that identifies the following data as a *PTCUR* file. This is a mandatory first line for all *PTCUR* files.

[NAME] is used to identify the type of file for GNOME and allows the user to supply a name for the resulting current mover that will appear in the GNOME Summary List, in the left section of the window.

[CURSCALE] is used to set a global multiplicative scale for all of the current data in the file. In general, GNOME assumes that all of the current data it receives are in units of  $m s^{-1}$ , but the *PTCUR* mover will allow for a change of units through this global scaling term. If this term is not provided in the file, a value of 1.0 will be assumed. In GNOME's Diagnostic Mode, the associated dialog box allows the user to set or override this value.

[UNCERTALONG] and [UNCERTCROSS] are terms whereby the user can specify a pair of coefficients that GNOME will use to set the uncertainty associated with the *PTCUR* mover. The first coefficient will set the bound on the Monte Carol uncertainty components added/subtracted to the along-axis component of the current vector, and the second coefficient will be used to Monte Carol the cross-axis uncertainty of the current vectors. If this term is not provided in the file, values of 0.5 and 0.25 will be assumed. In GNOME's Diagnostic Mode, the associated dialog box allows the user to override these values.

[UNCERTMIN] is currently not implemented, and a value of 0.0 is assumed. When implemented, the Uncertainty Minimum is intended to be used to set a minimum speed resolution that is expected from

the model, and is used to Monte Carol an uncertainty for cases where the predicted currents are very small. If this term is not provided in the file, a value of 0.05 will be assumed. In GNOME’s Diagnostic Mode, the associated dialog box allows the user to override these values.

[GRIDTYPE] is an identifier of how the vertical depth data were modeled. If there are no depth data, the keyword “2D” is used. If there is information about the depth of the area being modeled, but the currents are the same at every depth, the keyword “BAROTROPIC” is used (see Figure 1, below). If the depth is modeled using a sigma coordinate model, the keyword “SIGMA” is used (see Figure 2). If the depth is modeled using a layered system, the keyword “MULTILAYER” is used (see Figure 3). These last two options also require a boundary keyword, either “NOSLIP” or “FREESLIP”, where “NOSLIP” also requires a distance in meters to define the boundary layer. This distance is constant throughout the domain. The default is “2D”, in which case any depth information will be ignored.

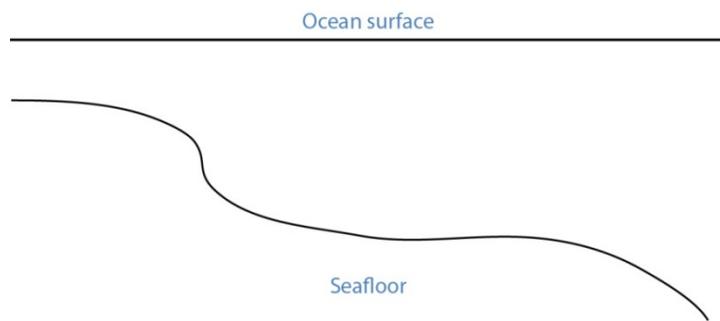


Figure 1. Barotropic model – single velocity top to bottom.

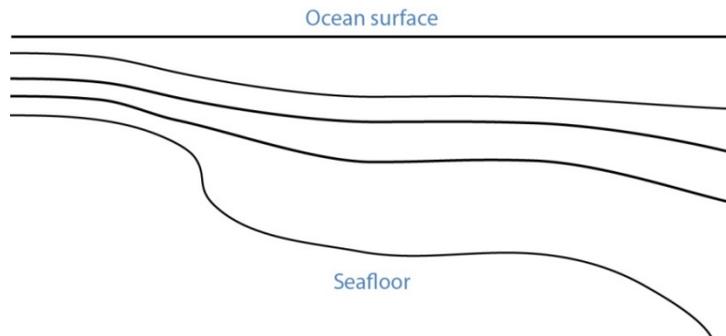


Figure 2. Sigma model – uniform number of layers, thickness scales to total depth.

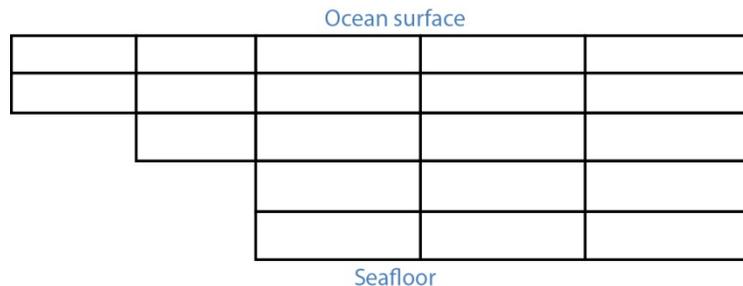


Figure 3. Gridded model – number of layers and layer thickness may vary from place to place.

[MAXNUMDEPTHS] gives the maximum number of depth points where horizontal currents are available. In some cases, points within the model may have fewer defined depth points than this number. The sigma model, however, must have data for *MAXNUMDEPTHS* in the water column at every horizontal point. The layered model has data at a maximum of *MAXNUMDEPTHS* in the water column for any horizontal point. The default is 1, which corresponds to surface data only and is the case for both the 2-D and barotropic grid types. (Though the latter has depth, it only has one unique current value per horizontal point.)

#### 1.2.1.2.2 The Point Definition Block

The “POINT DEFINITION BLOCK” describes the area covered by the model, including all of the horizontal points where data are available and at which depths the information is specified. This part of the model description also completely defines the topological characteristics of the model domain by specifying the boundary segments that divide the region into “inside” and “outside” portions.

The first line in the POINT DEFINITION BLOCK is made up of the keyword “Vertices”, followed by the total number of horizontal points and the number of land points, separated by white space.

[USERDATA] is a reserved word that can be used (repeatedly, if necessary) by the developer of the *PTCUR* data to record any type of text documentation or metadata that they want to keep associated with the file. This is optional and can be thought of as a comment area in the file format.

**Vertices          NPTs      NumLandPts**

The fields are defined as follows:

NPTs -              Gives the total number of horizontal data points (boundary and inside vertices).

NumLandPts -    If data are available at all the horizontal points, this is zero. If there is a separate outer boundary from a land map where current data are not available (assumed to be zero there), the number of these boundary points is given.

The next *NPTs* lines of the POINT DEFINITION BLOCK define the individual horizontal points where data are available. Each line describes a single data point with the following fields, separated by white space.

**Pt#      Longitude      Latitude depth d1 d2 ... dNdepths**

Each of the fields is defined as follows:

Pt# -                      A sequence number, or line number, assigned to each point 1...*NPTs*.

Longitude -              The latitude of the point, given in decimal degrees ranging from -180 to 180. The Western hemisphere is by convention negative latitude.

Latitude -                The longitude of the point, given in decimal degrees ranging from -90 to 90. The Northern hemisphere is by convention positive longitude.

- depth - The depth in meters of the modeled ocean at the point. If the grid-type is 2-D, this field and the rest of the line will not be read.
- d1 d2 ... dNdepths - Each of the *d1* through *dNdepths* values will be interpreted as a depth within the water column where current information will be defined. If the grid-type is barotropic, these points will not be read and the currents that are given will be assumed to represent the entire water column. For any point where data are available at fewer than the maximum number of depths, the user should enter, in order, all the valid depths and end the line with -1 to mark the end of data.

The lines that represent data points have two restrictions on the order in which they are entered into the file:

1. All boundary segments must be at the beginning of the file
2. All boundary segments must have their points entered sequentially in “counter-clockwise” (CC) order.
  - a. CC order is defined as follows: If an observer were to travel along the boundary of the model in a direction such that his/her left hand always pointed to the inside of the model, then they would encounter the boundary points in CC order.
  - b. To build a *PTCUR* file, the user would first enter all of the points in CC order around the outer boundary of the model and follow those by the points in CC order around all the islands (in this case, only one). After the boundary segments are entered in the point list, all other points (the interior ones) can be entered in any order that is convenient.

After a line is entered for each of the model’s horizontal data points, the next line contains a single integer value:

**Number\_of\_Boundary\_Segments**

This is the total number of boundary segments that are needed to define the inside/outside of the model. The first boundary listed is the main outer edge of the model; then each of the islands represented by the model is added. For example, a domain with no islands will have a value of “1”, while a domain with two islands will have a value of “3”.

Following the line that tells GNOME how many boundary segments there are in the model domain will be one line for each of the boundary segments, with the number of the last boundary point on the corresponding segment.

**Last\_point\_in\_segment1**  
**Last\_point\_in\_segment2**  
 ...

You may want to define the land/water map from the vertices of your domain. This may be preferable to using a high resolution shoreline if your model and the shoreline have significant mismatch. In order to define the map, you need to specify if any of the segments are open water.

```
WaterBoundaries    2    5
3
4
```

The numbers in the header line are the number of water boundaries and the total number of boundaries. The listed points are the indices of the end-points of the water boundary segments.

#### 1.2.1.2.3 The Topology Block - Optional

From the POINT DEFINITION BLOCK, GNOME will be able to completely reconstruct the topology and geometry of the model domain and develop an interpolation procedure to estimate data between the specified data points. GNOME will also be able to calculate when a pollutant particle has encountered a boundary of the model domain.

Alternatively, the CATS program can be used to determine the topology. The POINT DEFINITION BLOCK is in similar form to a vertex data (VerDat) file and can easily be transformed to one. To do this:

1. create a separate file with a header line "DOGS",
2. then all the points, comma delimited, followed by a line of zeros, and
3. finally the boundary information.

Any depth information should be removed and a single z value included for each point ( $\geq 1.0$ ). The order of the points in the *PTCUR* file must be the same as those in the VerDat file used in CATS. You can then create a fake current pattern, and export the .CUR file. Select the Topology and DAGtree blocks from a CATS .CUR file and paste them into your *PTCUR* file. (The DAGtree is optional. If GNOME doesn't find a DAGtree, it will create it from the topology.) Then GNOME won't have to perform triangularization, which saves time if the same topology will be used repeatedly with different data sets. GNOME will ignore the velocity information given at the end of each topology line from the CATS .CUR file. For more information on using CATS to transform a POINT DEFINITION BLOCK to a VerDat file, see the CATS-specific documentation.

#### 1.2.1.2.4 The Time-Specific Data Blocks

The TIME SPECIFIC DATA BLOCK contains the actual current velocity data for a fixed current pattern. If the input data represent a time-stepping pattern, then the block will be repeated as many times as necessary to step through the input information.

The first line in the TIME SPECIFIC DATA BLOCK is the keyword [TIME], followed by the time at which the block of current data was taken.

```
[TIME] StartDay StartMo StartYr StartHr StartMin
```

The last five fields on this line define the time when the data in the following data block were taken. If these fields have the default value "-1" in them, it will indicate that the model data represent a steady state and that only one TIME SPECIFIC DATA BLOCK will be present.

The next NPTs lines of data in the POINT DEFINITION BLOCK give the current data for each of the points described in the POINT DEFINITION BLOCK. The line of data contains:

**Pt#      Ud1      Vd1      Ud2      Vd2      ... UdNdepths VdNdepths**

The number of *U-V* pairs that are given on each line will need to correspond to the data given in the POINT DEFINITION BLOCK. For example, if four different depth data points are specified for a particular point, then four *U-V* pairs will be expected. This means that different lines of data may be of different lengths, but they will all end with a return sequence, and the actual number of fields for a particular point will be given by the line defining that point in the POINT DEFINITION BLOCK.

If the TIME SPECIFIC DATA BLOCK does not start with a constant time flag, then it may be followed by another TIME SPECIFIC DATA BLOCK, which is in the same format, but will have a different time. Each succeeding time block must have a time value that is larger than the one from the previous block. The offsets can vary in size, though.

For very large data sets, where having all the currents in one file would be unwieldy (for example, small time-steps or extended time runs, as in Trajectory Analysis Planner [TAP]), there is an alternative format. The [TIME] blocks can be put in separate files, with any number of blocks in each file. In place of these blocks in the header file, the full file path names (or partial paths, relative to the GNOME folder), and the start-time and end-time contained in each file should be listed. The keywords for this are [FILE], [STARTTIME], and [ENDTIME]. If there is a single time in a file, the start-time and end-time are the same. A constant current can also be done this way; start-time and end-time are both a series of negative ones ("-1").

Three annotated example files follow, each starting at the top of a page for easier reading. At this time, only 2-D time-dependent (*x,y,t*) data are shown in the examples.

**1.2.1.2.5 Example 1 – Filename: *skipptcur.cur***

```
[FILETYPE]      PTCUR
[NAME]    skip_ps_ptcur2D
[CURSCALE]      2.0
[UNCERTALONG]      .3052
[UNCERTCROSS]      .127
[UNCERTMIN]      .01
[MAXNUMDEPTHS]      1
[GRIDTYPE]      2-D
[USERDATA]      hi fred
[USERDATA]how are you ?
VERTICES      5056      3150
1      -122.548000      47.588500      1.000
2      -122.547000      47.585500      1.000
3      -122.548000      47.581300      1.000
4      -122.547000      47.578700      1.000
5      -122.543000      47.578200      1.000
6      -122.544000      47.576000      1.000
7      -122.546000      47.574000      1.000
8      -122.549000      47.572400      1.000
9      -122.550000      47.570600      1.000
10      -122.545000      47.568500      1.000
...
...
```

```

5054 -122.447000 47.582600 1.000
5055 -122.437000 47.583300 1.000
5056 -122.427000 47.583600 1.000
Topology 6986
4 5045 5046 4162 2612 2613 -2.536220 3.269671
2 3 4 -1 2 -1 0.662334 0.724430
4 1 2 -1 1 2612 1.187206 0.244548
5 10 5045 8 2613 7 -0.668295 1.037525
6 9 10 -1 7 6 -0.174680 -0.246778
7 8 9 -1 6 -1 -0.130291 -0.090753
...
...
4958 4942 4959 6981 6975 6960 -0.899112 5.741174
4442 4441 4419 6922 6966 6985 0.818613 0.580789
4420 4421 4442 6969 6966 6967 0.626956 0.418461
4442 4461 4441 6980 6983 6970 0.641757 0.720018
DAGTree 15270
34236 1 7758
10448 2 3922
32803 3 1906
23762 4 1005
13772 5 492
34118 6 260
...
...
2448 -8 -8
2455 15268 15269
2454 -8 -8
2466 -8 -8
[TIME] 14 2 00 10 00 Day Month Year Hour Minute
0.889853 0.737729
2.14009 1.90379
2.84519 2.40390
2.84138 2.89028
2.33662 3.00912
0.0381742 1.07280
1.23144 2.63017
1.02648 2.13573
...
-1.961549.09004
1.30358 -7.58093
0.697695 -6.05114
[TIME] 14 2 00 11 00 Day Month Year Hour Minute
0.605738 0.502185
1.38961 1.23618
0.982804 0.830371
-0.529060 -0.538164
-1.72646-2.22335
0.403527 -0.554565
-1.38999-2.69251
...
2.66105 -11.2783
-0.714619 3.31164
2.13874 -12.4378
-0.351095 3.04506 Velocity information ends the file

```

**1.2.1.2.6 Example 2 – Filename: ptCurMap.cur**

```

[FILETYPE] PTCUR
[NAME] PtCur : Negative currents
[CURSCALE] 2.0
[UNCERTALONG] .3052
[UNCERTCROSS] .127

```

```

[UNCERTMIN] .01
[MAXNUMDEPTH] 1
[GRIDTYPE] 2-D
[USERDATA] comments here
[USERDATA]
Vertices 9 0
1 -124.360000 48.574744 1.000000
2 -124.959368 48.563896 1.000000
3 -125.104952 48.182896 1.000000
4 -124.534720 48.210148 1.000000
5 -124.360000 48.288996 1.000000
6 -124.702840 48.452732 97.000000
7 -124.863320 48.383372 60.000000
8 -124.739872 48.299656 102.000000
9 -124.545448 48.400108 75.000000
BoundarySegments 1
5
WaterBoundaries 2 5 (optional section to generate land water map)
3
4
[TIME] 14 2 00 10 00
0.041327 0.001107
0.079485 -0.004495
0.036132 0.002556
0.053070 0.035451
0.086580 0.005730
0.045369 0.012076
0.031629 -0.002985
0.039163 0.009258
0.023545 -0.000079
[TIME] 14 2 00 11 00
0.041327 0.001107
0.079485 -0.004495
0.036132 0.002556
0.053070 0.035451
0.086580 0.005730
0.045369 0.012076
0.031629 -0.002985
0.039163 0.009258
0.023545 -0.000079
[TIME] 14 2 00 12 00
0.041327 0.001107
0.079485 -0.004495
0.036132 0.002556
0.053070 0.035451
0.086580 0.005730
0.045369 0.012076
0.031629 -0.002985
0.039163 0.009258
0.023545 -0.000079
[TIME] 14 2 00 13 00
0.041327 0.001107
0.079485 -0.004495
0.036132 0.002556
0.053070 0.035451
0.086580 0.005730
0.045369 0.012076
0.031629 -0.002985
0.039163 0.009258
0.023545 -0.000079
[TIME] 14 2 00 14 00
0.041327 0.001107

```

0.079485	-0.004495
0.036132	0.002556
0.053070	0.035451
0.086580	0.005730
0.045369	0.012076
0.031629	-0.002985
0.039163	0.009258
0.023545	-0.000079
[TIME]	14 2 00 15 00
0.041327	0.001107
.079485	-0.004495
0.036132	0.002556
0.053070	0.035451
0.086580	0.005730
0.045369	0.012076
0.031629	-0.002985
0.039163	0.009258
0.023545	-0.000079
[TIME]	14 2 00 16 00
0.041327	0.001107
0.079485	-0.004495
0.036132	0.002556
0.053070	0.035451
0.086580	0.005730
0.045369	0.012076
0.031629	-0.002985
0.039163	0.009258
0.023545	-0.000079
[TIME]	14 2 00 17 00
0.041327	0.001107
0.079485	-0.004495
0.036132	0.002556
0.053070	0.035451
0.086580	0.005730
0.045369	0.012076
0.031629	-0.002985
0.023545	-0.000079
0.027216	0.003247

### 1.2.1.2.7 Example 3 – Filename: *ptCurNoMap.cur*

```
[FILETYPE]      PTCUR
[NAME] PtCur : Negative currents
[CURSCALE]      2.0
[UNCERTALONG]   .3052
[UNCERTCROSS]   .127
[UNCERTMIN]     .01
[MAXNUMDEPTH]   1
[GRIDTYPE]      2-D
[USERDATA]      comments here
Vertices 9      0
1      -124.360000   48.574744   1.000000
2      -124.959368   48.563896   1.000000
3      -125.104952   48.182896   1.000000
4      -124.534720   48.210148   1.000000
5      -124.360000   48.288996   1.000000
6      -124.702840   48.452732   97.000000
7      -124.863320   48.383372   60.000000
8      -124.739872   48.299656   102.000000
9      -124.545448   48.400108   75.000000
BoundarySegments 1
```

5 *(Note that the Water Boundaries section is missing)*

```
[TIME] 14      2 00 10 00
0.041327      0.001107
0.079485      -0.004495
0.036132      0.002556
0.053070      0.035451
0.086580      0.005730
0.045369      0.012076
0.031629      -0.002985
0.039163      0.009258
0.023545      -0.000079
[TIME] 14 2 00 11 00
0.041327      0.001107
0.079485      -0.004495
0.036132      0.002556
0.053070      0.035451
0.086580      0.005730
0.045369      0.012076
0.031629      -0.002985
0.039163      0.009258
0.023545      -0.000079
[TIME] 14 2 00 12 00
0.041327      0.001107
0.079485      -0.004495
0.036132      0.002556
0.053070      0.035451
0.086580      0.005730
0.045369      0.012076
0.031629      -0.002985
0.039163      0.009258
0.023545      -0.000079
[TIME] 14 2 00 13 00
0.041327      0.001107
0.079485      -0.004495
0.036132      0.002556
0.053070      0.035451
0.086580      0.005730
0.045369      0.012076
0.031629      -0.002985
0.039163      0.009258
0.023545      -0.000079
```

```

[TIME] 14 2 00 14 00
0.041327      0.001107
0.079485     -0.004495
0.036132      0.002556
0.053070      0.035451
0.086580      0.005730
0.045369      0.012076
0.031629     -0.002985
0.039163      0.009258
0.023545     -0.000079
[TIME] 14 2 00 15 00
0.041327      0.001107
0.079485     -0.004495
0.036132      0.002556
0.053070      0.035451
0.086580      0.005730
0.045369      0.012076
0.031629     -0.002985
0.039163      0.009258
0.023545     -0.000079
[TIME] 14 2 00 16 00
0.041327      0.001107
0.079485     -0.004495
0.036132      0.002556
0.053070      0.035451
0.086580      0.005730
0.045369      0.012076
0.031629     -0.002985
0.039163      0.009258
0.023545     -0.000079
[TIME] 14 2 00 17 00
0.041327      0.001107
0.079485     -0.004495
0.036132      0.002556
0.053070      0.035451
0.086580      0.005730
0.045369      0.012076
0.031629     -0.002985
0.023545     -0.000079
0.027216      0.003247

```

### 1.2.1.3 Currents: Rectangular Grid – Steady State [GridCur]

The GridCur file should contain velocity information in the x and y directions on a rectangular grid. The first eight lines contain header information that defines the file type, grid size, and grid location. The remaining lines contain the current data. The keywords are the words shown in capital letters below. They must appear exactly as shown. This documentation consists of two example files followed by an explanation of each of the file components. You can set the range of the data by providing:

- a) the upper left corner position and the increment size of  $\Delta_x$  and  $\Delta_y$ , or
- b) the bounding latitude and longitude box.

If you would like to try either of these current patterns, you will also need the *GridCur.bna* file.

**Note:** If you have missing values, you may simply skip those grid points in the data file.

### 1.2.1.3.1 Example 1 – Filename: *GridCurExA.cur*

In this first example, *GridCurExA.cur*, position information is given from a starting latitude and longitude and an increment.

```
[GRIDCUR]
NUMROWS 100
NUMCOLS 100
STARTLAT 33.8
STARTLONG -120.4
DLAT .008
DLONG .01
row col u v
1 1 .10 .10
1 2 .10 .10
1 3 .10 .10
1 4 .10 .10
1 5 .10 .10
1 6 .10 .10
...
```

### 1.2.1.3.2 Example 2 – Filename: *GridCurExB.cur*

In this second example, *GridCurExB.cur*, the grid location is given by bounding latitudes and longitudes.

```
[GRIDCUR]
NUMROWS 100
NUMCOLS 100
LOLAT 33.4
HILAT 35
LOLONG -120.4
HILONG -119
row col u v
1 1 .10 .10
1 2 .10 .10
1 3 .10 .10
1 4 .10 .10
1 5 .10 .10
...
```

### 1.2.1.3.3 Explanation of File Components

The first line of the file is a flag identifying the file as an outside current file:

**NUMROWS** nrows  
**NUMCOLS** ncols

Lines 4 through 7 give the grid bounds and can be specified in either of two ways:

- (1) By the latitude and longitude of the grid origin (assumed to be the upper-left corner) and the increment size:

**STARTLAT** lat  
**STARTLON** long  
**DLAT** dlat  
**DLONG** dlong

(2) By low and high latitude and longitude ranges:

```
LOLAT lolat
HILAT hilat
LOLONG lolong
HILON hilong
```

In the former case, the velocities are assumed to be given at the grid points, and in the latter case, the velocities are assumed to be in the center of grid rectangles.

Line 8 is designed to be a header, identifying the columns of data. It is read, but not used.

```
row col u v
```

This header information is followed by NROWS × NCOLS lines of current data. Each line consists of 4 elements, corresponding to the items in Line 8. These are the point's location in the grid, given by a row and column, and its velocity components in the x and y directions, assumed to be in  $m s^{-1}$ . The file must contain a line for each of the NROWS × NCOLS grid points.

#### **1.2.1.4 Currents: Rectangular Grid – Time Dependent [GridCurTime]**

If you have a rectangular grid time-dependent model, you can use this data format to create the time-series of currents for GNOME. Large models and/or long time-series can produce large files of output fields. You have the option to store all your data in one file, or in a series of files. We have been successful in obtaining daily forecasts in separate files, archiving them, and then “stringing” them together to create a time-series for a single incident.

##### **1.2.1.4.1 Data in a Single File**

GNOME accepts rectangular grid models in a simple file format, similar to the single current pattern described above. The header now indicates with [GRIDCURTIME] that time has been added, and the time of the first time-step has been given in the [TIME] line.

Note: As in the rectangular GridCur data format, if you have missing values, you may simply skip those grid points in the data file. You may also create a constant current pattern by setting all the time references to -1.

##### **1.2.1.4.1.1 Example – Filename: gridcurTime.cur**

```
[GRIDCURTIME]
NUMROWS 100
NUMCOLS 100
LOLAT 34.0
HILAT 34.4
LOLONG -120.8
HILONG -119.2
[TIME] 14 2 00 10 00 day month year hour minute
1 1 .10 .10
1 2 .10 .10
1 3 .10 .10
1 4 .10 .10
...
```

Each succeeding time-step is simply appended onto the bottom:

```
...
100 97 .10 .10
100 98 .10 .10
100 99 .10 .10
100 100 .10 .10
[TIME] 14 2 00 11 00  next timestep information
1 1 .20 .20
1 2 .20 .20
1 3 .20 .20
1 4 .20 .20
1 5 .20 .20
1 6 .20 .20
```

#### 1.2.1.4.2 Data in Multiple Files

With larger time-series of current data, it may be useful to break the current time-series into separate files that make up a long time-series all together. In that case, GNOME supports multi-file data with a header file that indicates data and hard drive location information, and the subsequent files. The format is similar to the header on the regular *GridCurTime* format; however, rather than including the time information and the data, this header includes the file name and location, and the start and end times for each of the files. GNOME uses linear interpolation between time-steps within and across files. The references to the locations of the different current files on the computer's hard drive can be given two ways: a full path description of the directory or a relative description of the directory.

**Note:** A constant current can be created using a single record with all the time indicators set to -1. A single time-step is acceptable in a file with the start and end times listed as the same time in the header file.

The following four files are provided as examples with full path descriptions:

```
gridcurtime_hdr.cur
gridcurtime_hdrA.cur
gridcurtime_hdrB.cur
gridcurtime_hdrC.cur
```

#### 1.2.1.4.2.1 Example 1 – Filename: *gridcurtime\_hdr.cur*

The first file, *gridcurtime\_hdr.cur*, contains the header information, and the three subsequent files comprise the data.

```
[GRIDCURTIME]
NUMROWS 78
NUMCOLS 92
LOLAT 34.
HILAT 34.4
LOLONG -120.8
HILONG -119.2
[FILE] C:\GridCurTime\gridcurtime_hdrA.cur
[STARTTIME] 30 1 2002 1 0
[ENDTIME] 30 1 2002 2 0
[FILE] C:\GridCurTime\gridcurtime_hdrB.cur
[STARTTIME] 30 1 2002 3 0
[ENDTIME] 30 1 2002 5 0
[FILE] C:\GridCurTime\gridcurtime_hdrC.cur
[STARTTIME] 30 1 2002 6 0
[ENDTIME] 30 1 2002 8 0
```

#### 1.2.1.4.2.2 Example 2 – Filenames: *gridcurtime\_hdrA.cur*, *gridcurtime\_hdrB.cur*, and *gridcurtime\_hdrC.cur*

In the next example, the paths start with a colon (:), to indicate that they are relative paths.

```
[GRIDCURTIME]
NUMROWS 78
NUMCOLS 92
LOLAT 34
HILAT 34.4
LOLONG -120.8
HILONG -119.2
[FILE] :gridcurtime_hdrA.cur
[STARTTIME] 30 1 2002 1 0
[ENDTIME] 30 1 2002 2 0
[FILE] :gridcurtime_hdrB.cur
[STARTTIME] 30 1 2002 3 0
[ENDTIME] 30 1 2002 5 0
[FILE] :gridcurtime_hdrC.cur
[STARTTIME] 30 1 2002 6 0
[ENDTIME] 30 1 2002 8 0
```

Each subsequent file contains only the data, with no header information:

```
[TIME] 30 1 2002 1 0
1 1 0.00000 0.00000
1 2 0.00000 0.00000
1 3 0.00000 0.00000
1 4 0.00000 0.00000
1 5 0.00000 0.00000
1 6 0.00000 0.00000
1 7 0.00000 0.00000
1 8 0.00000 0.00000
1 9 0.00000 0.00000
1 10 0.00000 0.00000
1 11 0.00000 0.00000
```

## 1.2.2 NetCDF Formats

Currently, GNOME can read in NetCDF files for rectangular, curvilinear, and triangular grids. This section includes examples of the three formats currently in use and some descriptions of the required information. Please note that the NetCDF formats described here are presently undergoing revision to conform to the newly forming Climate & Forecast unstructured grid data model, to be adopted in future releases of GNOME.

### 1.2.2.1 NetCDF Rectangular Grid

Below is an example of the regular grid format for NetCDF files. The global attribute *grid\_type* = *REGULAR* is the default. Time units can be hours, minutes, seconds, or days. A separate map will be needed in order to set a spill.

```
NetCDF MacintoshHD:Desktop Folder:test {
dimensions:
    lat = 16 ;
    lon = 20 ;
    time = UNLIMITED ;          (85 currently)
variables:
    double lat(lat) ;
        lat:long_name = "Latitude" ;
        lat:units = "degrees_north" ;
        lat:point_spacing = "even" ;
    double lon(lon) ;
        lon:long_name = "Longitude" ;
        lon:units = "degrees_east" ;
        lon:point_spacing = "even" ;
    double time(time) ;
        time:long_name = "Valid Time" ;
        time:units = "minutes since 1999-11-25 00:00:00" ;
    float water_u(time, lat, lon) ;
        water_u:long_name = "Eastward Water Velocity" ;
        water_u:units = "m/s" ;
        water_u:_FillValue = -9.9999e+32f ;
        water_u:scale_factor = 1.f ;
        water_u:add_offset = 0.f ;
    float water_v(time, lat, lon) ;
        water_v:long_name = "Northward Water Velocity" ;
        water_v:units = "m/s" ;
        water_v:_FillValue = -9.9999e+32f ;
        water_v:scale_factor = 1.f ;
        water_v:add_offset = 0.f ;

    global attributes:
        :grid_type = "REGULAR" ;

data:

lat = 51.144606, 51.234386, 51.324167, 51.413944, 51.503722, 51.5935,
51.683275, 51.77305, 51.862825, 51.952594, 52.042364, 52.132133, 52.2219,
52.311664, 52.401425, 52.491186 ;

lon = 2.3155722, 2.4583139, 2.6010833, 2.743875, 2.8866917, 3.0295306,
3.1723917, 3.3152694, 3.4581667, 3.6010833, 3.7440139, 3.8869583,
4.0299167, 4.1728861, 4.3158667, 4.4588583, 4.6018583, 4.7448639,
4.887875, 5.0308917 ;

time = 7020, 7080, 7140, 7200, 7260, 7320, 7380, 7440, 7500, 7560, 7620,
7680, 7740, 7800, 7860, 7920, 7980, 8040, 8100, 8160, 8220, 8280, 8340,
```

8400, 8460, 8520, 8580, 8640, 8700, 8760, 8820, 8880, 8940, 9000, 9060,  
 9120, 9180, 9240, 9300, 9360, 9420, 9480, 9540, 9600, 9660, 9720, 9780,  
 9840, 9900, 9960, 10020, 10080, 10140, 10200, 10260, 10320, 10380, 10440,  
 10500, 10560, 10620, 10680, 10740, 10800, 10860, 10920, 10980, 11040,  
 11100, 11160, 11220, 11280, 11340, 11400, 11460, 11520, 11580, 11640,  
 11700, 11760, 11820, 11880, 11940, 12000, 12060 ;

### 1.2.2.2 NetCDF Curvilinear Grid

Below is an example of the curvilinear format for NetCDF files. The global attribute *grid\_type* = *CURVILINEAR* is required (the default is *grid\_type* = *REGULAR*). In addition to *x* and *y*, there are several other dimension name options for latitude and longitude. The dimension names only need to start with *X*, *Y* or *LAT*, *LON* to be recognized. The variable names must appear as shown. The velocities can be short, float, or double precision numbers. Time units can be hours, minutes, seconds, or days. The land-mask is required if you want to use the grid boundary as the shoreline: *0* is land, *1* is water. If no map is available, the mask is used to identify land points (land = *0*, water = *1*) and a boundary map is created. The first sigma value is used, though currently GNOME is being extended to handle 3-D currents. The topology can be saved out the first time and reloaded.

```
netcdf 20040726_11z_HAZMAT {
dimensions:
    x = 73 ;
    y = 163 ;
    sigma = 3 ;          optional
    time = UNLIMITED ;  (12 currently)
variables:
    float time(time) ;
        time:long_name = "Time" ;
        time:base_date = 2004, 1, 1, 0 ;
        time:units = "days since 2004-01-01 0:00:00 00:00" ;
        time:standard_name = "time" ;
    float lon(y, x) ;
        lon:long_name = "Longitude" ;
        lon:units = "degrees_east" ;
        lon:standard_name = "longitude" ;
    float lat(y, x) ;
        lat:long_name = "Latitude" ;
        lat:units = "degrees_north" ;
        lat:standard_name = "latitude" ;
    float mask(y, x) ;
        mask:long_name = "Land Mask" ;
        mask:units = "nondimensional" ;
    float depth(y, x) ; optional
        depth:long_name = "Bathymetry" ;
        depth:units = "meters" ;
        depth:positive = "down" ;
        depth:standard_name = "depth" ;
    float sigma(sigma) ;          optional
        sigma:long_name = "Sigma Stretched Vertical Coordinate at Nodes" ;
        sigma:units = "sigma_level" ;
        sigma:positive = "down" ;
        sigma:standard_name = "ocean_sigma_coordinate" ;
        sigma:formula_terms = "sigma: sigma eta: zeta depth: depth" ;
    float u(time, sigma, y, x) ;
        u:long_name = "Eastward Water Velocity" ;
        u:units = "m/s" ;
        u:missing_value = -99999.f ;
        u:_FillValue = -99999.f ;
```

```

        u:standard_name = "eastward_sea_water_velocity" ;
float v(time, sigma, y, x) ;
        v:long_name = "Northward Water Velocity" ;
        v:units = "m/s" ;
        v:missing_value = -99999.f ;
        v:_FillValue = -99999.f ;
        v:standard_name = "northward_sea_water_velocity" ;

```

*global attributes:*

```

:file_type = "Full_Grid" ;
:Conventions = "COARDS" ;
:grid_type = "curvilinear" ;
:z_type = "sigma" ;
:model = "POM" ;
:title = "Forecast: wind+tide+river" ;

```

data:

```

time = 208.4688, 208.4792, 208.4896, 208.5, 208.5104, 208.5208, 208.5312,
208.5417, 208.5521, 208.5625, 208.5729, 208.5833,,;

```

```

sigma = 0, .5, 1.;
}

```

### 1.2.2.3 NetCDF Triangular Grid

#### 1.2.2.3.1 Example – Triangular Grid Format with Velocities on the Nodes

Below is an example of the triangular grid format for NetCDF files with velocities on the nodes. The global attribute *grid\_type = TRIANGULAR* is required (the default is *grid\_type = REGULAR*). The first depth value is used. Time units can be hours, minutes, seconds, or days. A map will be created using the boundary data. The topology can be saved out the first time and reloaded.

The NetCDF header description for finite element model:

NetCDF MacintoshHD:Desktop Folder:testFile {

dimensions:

```

node = 7258 ;
nele = 13044 ; not currently used
nbnd = 1476 ;
nbi = 4 ;
sigma = 11 ; optional
time = UNLIMITED ; (12 currently)

```

variables:

```

short bnd(nbnd, nbi) ;
        bnd:long_name = "Boundary Segment Node List" ;
        bnd:units = "index_start_1" ;
float time(time) ;
        time:long_name = "Time" ;
        time:units = "days since 2003-01-00 0:00:00 00:00" ;
        time:base_date = 2003, 1, 0, 0 ;
float lon(node) ;
        lon:long_name = "Longitude" ;
        lon:units = "degrees_east" ;
float lat(node) ;
        lat:long_name = "Latitude" ;
        lat:units = "degrees_north" ;
float sigma(sigma) ; optional
        sigma:long_name = "Stretched Vertical Coordinate" ;
        sigma:units = "sigma_level" ;
        sigma:positive = "down" ;
float u(time, sigma, node) ;

```

```

u:long_name = "Eastward Water Velocity" ;
u:units = "m/s" ;
u:missing_value = -99999.f ;
u:_FillValue = -99999.f ;
float v(time, sigma, node) ;
v:long_name = "Northward Water Velocity" ;
v:units = "m/s" ;
v:missing_value = -99999.f ;
v:_FillValue = -99999.f ;

```

*global attributes:*

```

:file_type = "FEM" ;
:Conventions = "COARDS" ;
:grid_type = "Triangular" ;

```

data:

```

time = 26.95833, 27, 27.04167, 27.08333, 27.125, 27.16667, 27.20833, 27.25,
27.29167, 27.33333, 27.375, 27.41667 ;

```

```

sigma = 1, 0.9807215, 0.9306101, 0.83061, 0.6807215, 0.5, 0.3192785,
0.1693899, 0.06938996, 0.01927857, 0 ;
}

```

**Notes:**

1. The boundary list is an array of dimension  $bnd(nbnd, 4)$ . It consists of node numbers of the line segments, with a digit to indicate which land or island the segment is a part of, and a digit to indicate whether a boundary is land or water:

node1	node2	island	land/water (0/1)	
1	2	0	0	<i>1 is usually the continent and outer water BC</i>
2	5	0	0	
5	23	0	1	
...				
3568	1	0	1	<i>The last segment joins up with the first.</i>
551	552	1	0	<i>next island</i>
552	567	1	0	
...				
677	551	1	0	
789	388	2	0	
...				<i>next island, etc.</i>

2. Only the first sigma level is used, although GNOME is currently being extended to handle 3-D currents.

### 1.2.2.3.2 Example – Triangular Grid Format with Velocities on the Triangles

Following is an example of the triangular grid format for NetCDF files with velocities on the triangles. The global attribute *grid\_type = TRIANGULAR* is required (the default is *grid\_type = REGULAR*). The first depth value is used. Time units can be hours, minutes, seconds, or days. A map will be created using the boundary data. The topology must be included in the file.

```
netcdf FVCOM_example {
dimensions:
    node = 32649 ;
    nele = 60213 ;
    nbnd = 5099 ;
    nbi = 4 ;
    time = UNLIMITED ; // (1 currently)
    three = 3 ;
variables:
    int bnd(nbnd, nbi) ;
    float time(time) ;
        time:units = "days since 1978-11-17 00:00:00 0:00" ;
        time:long_name = "time" ;
        time:time_zone = "UTC" ;
        time:format = "modified julian day (MJD)" ;
    float lon(node) ;
    float lat(node) ;
    float u(time, nele) ;
        u:units = "meters s-1" ;
        u:long_name = "Eastward Water Velocity" ;
        u:grid = "fvcom_grid" ;
        u:type = "data" ;
    float v(time, nele) ;
        v:units = "meters s-1" ;
        v:long_name = "Northward Water Velocity" ;
        v:grid = "fvcom_grid" ;
        v:type = "data" ;
    int nbe(three, nele) ;
    int nv(three, nele) ;

// global attributes:
        :grid_type = "Triangular" ;
data:
time = 11452 ;
}
```

#### Notes:

1. The boundary list is an array of dimension *bnd(nbnd, 4)*, same as above.
2. The triangle vertices are contained in *nv* and the neighboring triangles in *nbe*.

### 1.2.2.4 Data in Multiple NetCDF Files: When Your NetCDF Files Start To Get Too Big

Longer simulations require more model data, and that can cause problems with putting the entire time-series into one data file. GNOME allows you to break the time-series into separate files using a master file to identify all the pieces of the time-series in order. This also makes possible using a series of nowcasts and forecasts strung together to make a times-series. This technique worked well during the 2002 T/V *Prestige* incident in Spain.

First create a text master file with the list of file path-names (relative to the GNOME directory) in order. Next supply the full path name if the files are not in the same directory as GNOME, or in a subdirectory. The file will also need a header line, “NetCDF Files”.

When you go to load the currents in GNOME, load your master file (e.g., §1.2.2.4.1 *Example 1 – Filename: MyMasterFileEx.txt*). GNOME will use this as the list of files for the time-series.

#### **1.2.2.4.1 Example 1 – Filename: *MyMasterFileEx.txt***

```
NetCDF Files
[FILE] :day1.nc
[FILE] :day2.nc
[FILE] :day3.nc
[FILE] :day4.nc
[FILE] :day5.nc
[FILE] :day6.nc
```

### **1.2.3 Scaling Current Patterns**

Since the current patterns created in CATS only indicate the direction of the current and the relative speeds, these current patterns need to be scaled in order to be useful with the trajectory model. For example, consider a fictitious current pattern with only two triangles, A and B. The velocity in triangle A is 1.2 to the east and the velocity in triangle B is 1.8 to the north. Observations indicate that the velocity in triangle A should be 3.0 knots to the east, so we must scale the current pattern by the ratio of these velocities in triangle A, or  $(3.0 \text{ knots} \div 1.2 = 2.5 \text{ knots})$ . That is, multiplying the velocity in triangle A in the current pattern (1.2) by the scale factor (2.5 knots) yields the observed velocity (3.0 knots). The direction did not change. To find the velocity in triangle B, we multiply the velocity in triangle B in the current pattern (1.8) by the scale factor (2.5 knots) to get a velocity of 4.5 knots. The velocity in triangle B is still to the north, since the direction does not change in the current pattern.

GNOME is quite helpful in scaling current patterns. At a given reference point in the current pattern, GNOME tells you what the flow is. You then input into GNOME what you would like the velocity to be at the reference point, and GNOME calculates the scaling coefficient for the pattern for you!

The direction of the flow in the current fields in GNOME can reverse by multiplying the pattern by a negative scaling coefficient. The ebb and flow of tides are simulated this way, through a time-series of positive and negative scaling values. You can scale currents with either a constant value or a time-series. The acceptable file formats for time-series are outlined below.

#### **1.2.3.1 Time-Series File Formats**

Current patterns in GNOME can be scaled to be time dependent with two different file types:

- (1) a time-series of current magnitude or
- (2) a “SHIO mover” that contains data for GNOME to use in calculating tidal current magnitudes.

All data in this section were created by the NOAA SHIO application (“shio” comes from Japanese for “tide”).

The South Bend, Washington, U.S.A. station on the Willapa River was chosen for all the examples in this section. Below is the information found in the *SouthBend.text* file to illustrate the information GNOME needs in order to calculate the tidal currents at this station. This particular file is not a data file for GNOME. Those data are represented in data files presented later in this discussion.

### 1.2.3.1.1 Example – Filename: *SouthBend.text*

Tidal currents at South Bend, Willapa River, WASHINGTON COAST

Station No. CP1009

Meter Depth: n/a

Latitude: 46°0' N

Longitude: 123°47' W

Maximum Flood Direction: 90°

Maximum Ebb Direction: 270°

Time offsets	Hour:Min
Min Before Flood	0:19am
Flood	0:20am
Min Before Ebb	0:24am
Ebb	-0:06am

Flood Speed Ratio: 0.6

Ebb Speed Ratio: 0.5

	Speed(kt)	Direction(deg.)
Min Before Flood	00.0	n/a
Flood	01.2	090
Min Before Ebb	00.0	n/a
Ebb	01.4	270

Based on Grays Harbor Ent.

Local Standard Time

```
-----
Mon, Aug 24, 1998--Sunrise -- 6:09am  Sunset -- 7:55pm
      0:38am  +01.2    Max Flood
      3:31am  +00.0    Min Before Ebb
      6:29am  -01.6    Max Ebb
      9:59am  +00.0    Min Before Flood
      1:08pm  +01.4    Max Flood
      4:12pm  +00.0    Min Before Ebb
      6:56pm  -01.4    Max Ebb
     10:17pm  +00.0    Min Before Flood
```

### 1.2.3.1.2 Time Series of Current Magnitude

Time series file for currents have the format

**dd,mm,yy,hr,min,|U|,0.0**

where *dd* is the day, *mm* is the month, *yy* is the year, *hr* is the hour, *min* is the minute, *|U|* is the magnitude of the velocity, and 0.0 is a number to indicate that the file is in a magnitude format rather than a *U,V* format. The direction is left blank because the current pattern supplies the individual current vectors. There is an optional header –

- The first line lists the station name.

- The second line lists the station position.
- The third line provides the units.

For example, the *SouthBend.ossm* file contains one day of tidal information for South Bend, Washington, U.S.A.

### 1.2.3.1.2.1 Example – Filename: *SouthBend.ossm*

```
South Bend
-123.78,46
knots
24, 8, 98, 0, 37, 1.2, 0.0
24, 8, 98, 3, 30, 0.0, 0.0
24, 8, 98, 6, 28, -1.6, 0.0
24, 8, 98, 9, 58, 0.0, 0.0
24, 8, 98, 13, 7, 1.4, 0.0
24, 8, 98, 16, 11, 0.0, 0.0
24, 8, 98, 18, 55, -1.4, 0.0
24, 8, 98, 22, 16, 0.0, 0.0
```

#### 1.2.3.1.2.1.1 Annotated Version of the File

Day,	Month,	Year,	Hour,	Min.,	Speed,	Direction (Dummy Value)
24,	8,	98,	0,	37,	1.2,	0.0
24,	8,	98,	3,	30,	0.0,	0.0
24,	8,	98,	6,	28,	-1.6,	0.0
24,	8,	98,	9,	58,	0.0,	0.0
24,	8,	98,	13,	7,	1.4,	0.0
24,	8,	98,	16,	11,	0.0,	0.0
24,	8,	98,	18,	55,	-1.4,	0.0
24,	8,	98,	22,	16,	0.0,	0.0

### 1.2.3.1.3 SHIO Movers: Using Tidal Constituents

GNOME can use both **tidal height** and **tidal current constituent data** to scale current patterns. In the case of tidal height station data (§1.2.3.1.3.1, below), GNOME will take the time derivative of the tidal heights, and request the user to scale that derivative to calculate the tidal currents. For the tidal current station data (§1.2.3.1.3.2, below), GNOME will use the calculated currents directly. The constituent record data are rather complex, so we have provided information about the data fields and then provided an example file.

#### 1.2.3.1.3.1 Tidal Heights Constituent Record

Line 1	[StationInfo]	
Line 2	Type=H	station type for heights is "H"
Line 3	staName	station name
Line 4	Latitude= <i>latStr</i>	decimal degrees
Line 5	Longitude= <i>longStr</i>	decimal degrees
Line 6	[Constituents]	
Line 7	DatumControls.datum= <i>datum</i>	mean sea level
Line 8	DatumControls.FDir=0	bug. Type as seen. Will be fixed in 1.2.7
Line 9	DatumControls.EDir=0	bug. Type as seen. Will be fixed in 1.2.7
Line 10	DatumControls.L2Flag=0	bug. Type as seen. Will be fixed in 1.2.7
Line 11	DatumControls.HFlag=0	bug. Type as seen. Will be fixed in 1.2.7
Line 12	DatumControls.RotFlag=0	bug. Type as seen. Will be fixed in 1.2.7

Lines 13-17      *constituent amplitudes in order*<sup>1</sup>      M2, S2, N2, K1, M4, O1, M6, MK3, S4, MN4, NU2, S6, MU2, 2N2, OO1, LAM2, S1, M1, J1, MM, SSA, SA, MSF, MF, RHO, Q1, T2, R2, 2Q1, P1, 2SM2, M3, L2, 2MK3, K2, M8, MS4 [feet]

Lines 18-23      *constituent phases in order*      see above, lines 13-17 [degrees]  
 Line 24      [Offset]

*Note: Lines 25-30 use a second integer to indicate to GNOME whether there is valid data in the field. "0" indicates no data, so GNOME can skip the calculation. "1" indicates valid data exists (that may be zero).*

Line 25      HighTime=*highTime* 1      *high water time adjustment (minutes)*  
 Line 26      LowTime=*lowTime* 1      *low water time adjustment*  
 Line 27      HighHeight\_Mult=*highHeightScalar* 1      *high water height multiplier*  
 Line 28      HighHeight\_Add=*highHeightAdd* 1      *high water height addend*  
 Line 29      LowHeight\_Mult=*lowHeightScalar* 1      *low water height multiplier*  
 Line 30      LowHeight\_Add=*lowHeightAdd* 1      *low water height addend*

1.2.3.1.3.1.1 Example – Filename: HornIslandPass.shio.txt

```
[StationInfo]
Type=H
Name=Horn Island Pass
Latitude=30.2167
Longitude=-88.483333
[Constituents]
DatumControls.datum=0.620000
DatumControls.FDir=0
DatumControls.EDir=0
DatumControls.L2Flag=0
DatumControls.HFlag=0
DatumControls.RotFlag=0
H=0.066000 0.022000 0.013000 0.468000 0.000000 0.460000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.020000 0.000000 0.000000 0.033000 0.036000 0.000000 0.120000 0.299000
0.000000 0.000000 0.018000 0.099000 0.000000 0.000000 0.012000 0.139000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
kPrime=358.500000 355.700012 0.000000 327.000000 0.000000 324.200012 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 329.799988 0.000000 0.000000 325.500000 328.399994 0.000000
32.099998 151.800003 0.000000 0.000000 323.000000 314.100006 0.000000 0.000000 321.299988 331.899994
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000
[Offset]
HighTime=-0.516667 1
LowTime=-0.883333 1
HighHeight_Mult=1.300000 1
HighHeight_Add=0.000000 1
LowHeight_Mult=1.300000 1
LowHeight_Add=0.000000 1
```

1.2.3.1.3.2 Tidal Currents Constituent Record

Line 1      [StationInfo]  
 Line 2      Type=C      *station type for currents is "C"*  
 Line 3      *staName*      *station name*  
 Line 4      Latitude=*latStr*      *decimal degrees*  
 Line 5      Longitude=*longStr*      *decimal degrees*  
 Line 6      [Constituents]  
 Line 7      DatumControls.datum=*datum*      *datum*  
 Line 8      DatumControls.FDir=*floodDirection*      *flood direction*  
 Line 9      DatumControls.EDir=*ebbDirection*      *ebb direction*  
 Line 10      DatumControls.L2Flag=*L2Flag*      *L2Flag*  
 Line 11      DatumControls.HFlag=*hydraulicFlag*      *hydraulic flag*

<sup>1</sup> Exceptions: For the cases of ANCHORAGE, CALETA PERCY, DAEHUKSAN DO, LISBON, MUKHO HANG, and SOKCHO HANG, there are more than 37 constituents.

Line 12 DatumControls.RotFlag=0 *For non-rotary tides, use "0". For rotary tides defined relative to North or East, use "1". For rotary tides defined by major and minor axes, use "2".*

Lines 13-17 constituent amplitudes in order<sup>2</sup> *M2, S2, N2, K1, M4, O1, M6, MK3, S4, MN4, NU2, S6, MU2, 2N2, OO1, LAM2, S1, M1, J1, MM, SSA, SA, MSF, MF, RHO, Q1, T2, R2, 2Q1, P1, 2SM2, M3, L2, 2MK3, K2, M8, MS4 [knots]*

Lines 18-23 constituent phases in order *see above, lines 13-17 [degrees]*

Line 24 [Offset]

Note: Lines 25-38 use a second integer to indicate to GNOME whether there is valid data in the field. "0" indicates no data, so GNOME can skip the calculation. "1" indicates valid data exists (that may be zero).

Line 25 MinBefFloodTime= minBefFloodTime 1 *minimum before flood time adjustment*

Line 26 FloodTime= floodTime 1 *flood time adjustment*

Line 27 MinBefEbbTime= minBefEbbTime 1 *minimum before ebb time adjustment*

Line 28 EbbTime= ebbTime 1 *ebb time adjustment*

Line 29 FloodSpdRatio=floodSpeedRatio 1 *flood speed ratio*

Line 30 EbbSpdRatio=ebbSpeedRatio 1 *ebb speed ratio*

Line 31 MinBFloodSpd=minBefFloodAvgSpeed 0 *average speed - minimum before flood*

Line 32 MinBFloodDir=minBefFloodAvgDir 0 *average direction - minimum before flood*

Line 33 MaxFloodSpd=maxFloodAvgSpeed 0 *average speed - flood*

Line 34 MaxFloodDir=maxFloodAvgDir 0 *average direction - flood*

Line 35 MinBEbbSpd=minBefEbbAvgSpeed 0 *average speed - minimum before ebb*

Line 36 MinBEbbDir=minBefEbbAvgDir 0 *average direction - minimum before ebb*

Line 37 MaxEbbSpd=maxEbbAvgSpeed 0 *average speed - ebb*

Line 38 MaxEbbDir=maxEbbAvgDir 0 *average direction - ebb*

#### 1.2.3.1.3.2.1 Example – Filename: StJohnsRiver.shio.txt

```
[StationInfo]
Type=C
Name=ST. JOHNS RIVER ENT. (between jetties)
Latitude=30.400000
Longitude=-81.383333
[Constituents]
DatumControls.datum=-0.350000
DatumControls.FDir=275
DatumControls.EDir=100
DatumControls.L2Flag=0
DatumControls.HFlag=0
DatumControls.RotFlag=0
H=1.993000 0.333000 0.404000 0.216000 0.293000 0.174000 0.092000 0.000000 0.000000 0.000000 0.078000
0.000000 0.000000 0.054000 0.000000 0.014000 0.000000 0.012000 0.014000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.034000 0.020000 0.000000 0.000000 0.071000 0.000000 0.000000 0.054000
0.000000 0.091000 0.044000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
kPrime=227.199997 244.399994 208.800003 98.800003 131.100006 122.699997 238.699997 0.000000 0.000000
0.000000 211.199997 0.000000 0.000000 190.300003 0.000000 235.199997 0.000000 110.699997 86.800003
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 134.600006 244.600006 0.000000 0.000000 99.199997
0.000000 0.000000 245.699997 0.000000 244.000000 100.800003 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000
[Offset]
MinBefFloodTime=0.000000 1
FloodTime=0.000000 1
MinBefEbbTime=0.000000 1
EbbTime=0.000000 1
FloodSpdRatio=1.000000 1
EbbSpdRatio=1.000000 1
MinBFloodSpd=0.000000 0
MinBFloodDir=0.000000 0
MaxFloodSpd=0.000000 0
MaxFloodDir=0.000000 0
```

<sup>2</sup> Exceptions: For the cases of ANCHORAGE, CALETA PERCY, DAEHUKSAN DO, LISBON, MUKHO HANG, and SOKCHO HANG, there are more than 37 constituents.

MinBEbbSpd=0.000000 0  
MinBEbbDir=0.000000 0  
MaxEbbSpd=0.000000 0  
MaxEbbDir=0.000000 0

#### 1.2.3.1.3.2.2 Example – Filename: Edmonds.shio.txt

```
[StationInfo]
Type=C
Name=Edmonds, 2.7 miles WSW of
Latitude=47.800000
Longitude=-122.450000
[Constituents]
DatumControls.datum=-0.500000
DatumControls.FDir=180
DatumControls.EDir=5
DatumControls.L2Flag=0
DatumControls.HFlag=0
DatumControls.RotFlag=0
H=1.954000 0.460000 0.402000 0.847000 0.000000 0.421000 0.000000 0.000000 0.000000 0.000000 0.078000
0.000000 0.000000 0.054000 0.018000 0.013000 0.000000 0.030000 0.033000 0.000000 0.000000 0.000000
0.000000 0.000000 0.016000 0.081000 0.028000 0.000000 0.000000 0.280000 0.000000 0.000000 0.055000
0.000000 0.125000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
kPrime=66.400002 84.099998 39.400002 72.500000 0.000000 66.199997 0.000000 0.000000 0.000000 0.000000
43.000000 0.000000 0.000000 12.300000 78.800003 74.599998 0.000000 69.300003 75.800003 0.000000 0.000000
0.000000 0.000000 0.000000 63.500000 63.000000 84.400002 0.000000 0.000000 73.199997 0.000000 0.000000
93.400002 0.000000 83.400002 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
[Offset]
MinBefFloodTime=0.733333 1
FloodTime=0.100000 1
MinBefEbbTime=0.216667 1
EbbTime=0.316667 1
FloodSpdRatio=0.100000 1
EbbSpdRatio=0.200000 1
MinBFloodSpd=0.000000 1
MinBFloodDir=0.000000 0
MaxFloodSpd=0.200000 1
MaxFloodDir=170.000000 1
MinBEbbSpd=0.000000 1
MinBEbbDir=0.000000 0
MaxEbbSpd=0.500000 1
MaxEbbDir=0.000000 1
```

#### 1.2.3.1.4 Hydrology Time-Series

Hydrology time-series files for currents have the format per the bulleted list below. An example hydrology time-series file, *Hillsborough.HYD*, is also provided in §1.2.3.1.4.1.

- The first line lists the station name.
- The second line contains the reference point position for scaling the current pattern with the hydrology volume transport time-series.
- The third line provides the units for the volume transport:
  - cubic feet per second (CFS)
  - kilo cubic feet per second (KCFS)
    - Defined as 1,000 cubic feet of water passing a given point for an entire second.
  - cubic meters per second (CMS)
  - kilo cubic meters per second (KCMS)

- Defined as 1,000 cubic meters of water passing a given point for an entire second.

The data are given in the same time-series format as the currents, except that the magnitude of the current is changed to the volume transport.

#### 1.2.3.1.4.1 Example – Filename: Hillsborough.HYD

```
HILLSBOURGH STATION
28.029534,-82.688080
CMS
01,10,2002,0,0,432,0
02,10,2002,0,0,309,0
03,10,2002,0,0,310,0
04,10,2002,0,0,312,0
05,10,2002,0,0,311,0
06,10,2002,0,0,287,0
07,10,2002,0,0,234,0
08,10,2002,0,0,235,0
09,10,2002,0,0,232,0
10,10,2002,0,0,177,0
```

##### 1.2.3.1.4.1.1 Annotated Version of the File

```
HILLSBOURGH STATION      Station Name
28.029534,-82.688080    Position (latitude, longitude)
CMS      Units
Day,     Month,   Year,   Hour,   Min.,   Transport,   Direction (Dummy Value)
01,      10,      2002,   0,      0,      432,        0
02,      10,      2002,   0,      0,      309,        0
03,      10,      2002,   0,      0,      310,        0
04,      10,      2002,   0,      0,      312,        0
05,      10,      2002,   0,      0,      311,        0
```

## 1.3 Winds

GNOME uses winds, in addition to the currents and diffusion, to move the oil. For small-scale uses, a single point forecast, *[OSSM]*, is sufficient for trajectories. In this case, the time-series can be created in GNOME using the variable or constant wind dialog boxes, or loaded as a file. You will find it useful to load winds as a file if you are downloading archived wind observations, or if you are creating a blended time-series, with archived observations combined with a wind forecast.

For large-scale areas, you may use winds generated by atmospheric circulation model data. Be careful in mapping latitude and longitude of the grid points with the proper projection of the model.

GNOME supports both ASCII and NetCDF formats for wind. The rectangular grid wind model in a time-series format is *[GridWindTime]*. The gridded output time series format is the same as for currents, except the starting keyword is different. This is to prevent the user from accidentally loading a wind as a current, and vice versa. The Finite Element ASCII Format (for winds at the model nodes) is *[ptWind]*. GNOME also supports a NetCDF file structure for rectangular grid models. If your particular atmospheric model is not supported, please let the GNOME Wizard know (ORR.GNOME@noaa.gov), as we are always interested in adding new grids to our collection.

### 1.3.1 Winds: Single Point, Time Series [OSSM]

If you input an On-Scene Spill Model (OSSM) format wind file into GNOME, you can specify the units when the file is loaded, or there is an optional header –

- The first line lists the station name.
- The second line lists the station position.
- The third line provides the units.

#### 1.3.1.1 Example – Filename: OSSM Format.WND

```
Inchon
-126.63,37.5
knots
8,4,99,01,00,10,S
8,4,99,05,00,10,S
8,4,99,09,00,10,S
8,4,99,11,00,10,S
8,4,99,15,00,10,SW
8,4,99,21,00,10,SW
9,4,99,01,00,10,SW
9,4,99,05,00,10,SW
9,4,99,09,00,10,SW
9,4,99,11,00,10,SW
9,4,99,15,00,10,SW
9,4,99,21,00,10,SW
10,4,99,01,00,10,SW
10,4,99,05,00,05,S
10,4,99,09,00,05,S
10,4,99,11,00,05,S
10,4,99,15,00,05,S
10,4,99,21,00,05,S
11,4,99,01,00,10,SW
11,4,99,05,00,10,SW
11,4,99,09,00,10,SW
11,4,99,11,00,10,W
11,4,99,15,00,10,W
11,4,99,21,00,10,W
12,4,99,01,00,25,NW
12,4,99,05,00,25,NW
12,4,99,09,00,25,NW
12,4,99,11,00,25,NW
12,4,99,15,00,25,NW
12,4,99,21,00,25,NW
```

#### 1.3.1.1.1 Annotated Version of the File

Day,	Month,	Year,	Hour,	Min.,	Speed,	Direction
8,	4,	99,	01,	00,	0,	S
8,	4,	99,	05,	00,	10,	S
8,	4,	99,	09,	00,	10,	S
8,	4,	99,	11,	00,	10,	S
8,	4,	99,	15,	00,	10,	SW
...						

Note: Direction can also be in degrees.

### 1.3.2 Winds: Rectangular Grid, Time Series [GridWindTime]

The rectangular grid surface wind model formats are very similar to the *GridCurTime* formats for rectangular grid time-dependent currents. The only difference is that the first line of the file changes from *[GRIDCURTIME]* to *[GRIDWINDTIME]*.

#### 1.3.2.1 Example – Filename: GridWindTime.wnd

```
[GRIDWINDTIME]
NUMROWS 19
NUMCOLS 26
LOLAT 36.6
HILAT 47.8
LOLONG -15.4
HILONG -4.3
[TIME] 19 11 02 1 00
1 1 0.15 -.16
1 2 0.16 -.17
1 3 0.17 -.19
1 4 0.19 -.21
...
```

### 1.3.3 Winds: NetCDF Rectangular Grid, Time Series

The NetCDF rectangular grid surface wind model formats are very similar to the NetCDF rectangular grid current. The only difference is that *air\_u* and *air\_v* are used instead of *water\_u* and *water\_v* for the *U* and *V* velocity components.

```
netcdf pwsWind2004080904 {
dimensions:
    lon = 155 ;
    lat = 150 ;
    time = UNLIMITED ;      (49 currently)
variables:
    float time(time) ;
        time:long_name = "Time in AST" ;
        time:units = "hours since 2004-08-09 00:00:00" ;
    float lon(lon) ;
        lon:long_name = "Longitude" ;
        lon:units = "degrees_East" ;
        lon:point_spacing = "even" ;
    float lat(lat) ;
        lat:long_name = "Latitude" ;
        lat:units = "degrees_North" ;
        lat:point_spacing = "even" ;
    float air_u(time, lat, lon) ;
        air_u:valid_range = -30.f, 30.f ;
        air_u:long_name = "Eastward Air Velocity" ;
        air_u:units = "m/s" ;
        air_u:_FillValue = -9.9999e+32f ;
        air_u:scale_factor = 1.f ;
        air_u:add_offset = 0.f ;
    float air_v(time, lat, lon) ;
        air_v:valid_range = -30.f, 30.f ;
        air_v:long_name = "Northward Air Velocity" ;
        air_v:units = "m/s" ;
        air_v:_FillValue = -9.9999e+32f ;
        air_v:scale_factor = 1.f ;
        air_v:add_offset = 0.f ;
```

*global attributes:*

```
:experiment = "PWS-NFS" ;
:grid_type = "REGULAR" ;
:base_date = 2004, 8, 9 ;
```

data:

```
time = 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52 ;
```

```
lon = -148.72, -148.7, -148.68, -148.66, -148.64, -148.62, -148.6, -148.58,
-148.56, -148.54, -148.52, -148.5, -148.48, -148.46, -148.44, -148.42,
-148.4, -148.38, -148.36, -148.34, -148.32, -148.3, -148.28, -148.26,
-148.24, -148.22, -148.2, -148.18, -148.16, -148.14, -148.12, -148.1,
-148.08, -148.06, -148.04, -148.02, -148, -147.98, -147.96, -147.94,
-147.92, -147.9, -147.88, -147.86, -147.84, -147.82, -147.8, -147.78,
-147.76, -147.74, -147.72, -147.7, -147.68, -147.66, -147.64, -147.62,
-147.6, -147.58, -147.56, -147.54, -147.52, -147.5, -147.48, -147.46,
-147.44, -147.42, -147.4, -147.38, -147.36, -147.34, -147.32, -147.3,
-147.28, -147.26, -147.24, -147.22, -147.2, -147.18, -147.16, -147.14,
-147.12, -147.1, -147.08, -147.06, -147.04, -147.02, -147, -146.98,
-146.96, -146.94, -146.92, -146.9, -146.88, -146.86, -146.84, -146.82,
-146.8, -146.78, -146.76, -146.74, -146.72, -146.7, -146.68, -146.66,
-146.64, -146.62, -146.6, -146.58, -146.56, -146.54, -146.52, -146.5,
-146.48, -146.46, -146.44, -146.42, -146.4, -146.38, -146.36, -146.34,
-146.32, -146.3, -146.28, -146.26, -146.24, -146.22, -146.2, -146.18,
-146.16, -146.14, -146.12, -146.1, -146.08, -146.06, -146.04, -146.02,
-146, -145.98, -145.96, -145.94, -145.92, -145.9, -145.88, -145.86,
-145.84, -145.82, -145.8, -145.78, -145.76, -145.74, -145.72, -145.7,
-145.68, -145.66, -145.64 ;
```

```
lat = 59.79, 59.8, 59.81, 59.82, 59.83, 59.84, 59.85, 59.86, 59.87, 59.88,
59.89, 59.9, 59.91, 59.92, 59.93, 59.94, 59.95, 59.96, 59.97, 59.98,
59.99, 60, 60.01, 60.02, 60.03, 60.04, 60.05, 60.06, 60.07, 60.08, 60.09,
60.1, 60.11, 60.12, 60.13, 60.14, 60.15, 60.16, 60.17, 60.18, 60.19,
60.2, 60.21, 60.22, 60.23, 60.24, 60.25, 60.26, 60.27, 60.28, 60.29,
60.3, 60.31, 60.32, 60.33, 60.34, 60.35, 60.36, 60.37, 60.38, 60.39,
60.4, 60.41, 60.42, 60.43, 60.44, 60.45, 60.46, 60.47, 60.48, 60.49,
60.5, 60.51, 60.52, 60.53, 60.54, 60.55, 60.56, 60.57, 60.58, 60.59,
60.6, 60.61, 60.62, 60.63, 60.64, 60.65, 60.66, 60.67, 60.68, 60.69,
60.7, 60.71, 60.72, 60.73, 60.74, 60.75, 60.76, 60.77, 60.78, 60.79,
60.8, 60.81, 60.82, 60.83, 60.84, 60.85, 60.86, 60.87, 60.88, 60.89,
60.9, 60.91, 60.92, 60.93, 60.94, 60.95, 60.96, 60.97, 60.98, 60.99, 61,
61.01, 61.02, 61.03, 61.04, 61.05, 61.06, 61.07, 61.08, 61.09, 61.1,
61.11, 61.12, 61.13, 61.14, 61.15, 61.16, 61.17, 61.18, 61.19, 61.2,
61.21, 61.22, 61.23, 61.24, 61.25, 61.26, 61.27, 61.28 ;
}
```

### 1.3.4 Winds: NetCDF Curvilinear Grid

The NetCDF curvilinear grid surface wind model format is very similar to the NetCDF curvilinear grid current format. The only differences are (1) that *air\_u* and *air\_v* are recommended instead of *u* and *v* for the *U* and *V* velocity components and (2) the land mask is not used. The dimension names only need to start with *X*, *Y* or *LAT*, *LON* to be recognized. The variable names must appear as shown. The topology can be saved out the first time and reloaded.

```
netcdf 20040726_11z_HAZMAT {
dimensions:
    x = 73 ;
    y = 163 ;
    time = UNLIMITED ;      (12 currently)
```

```

variables:
    float time(time) ;
        time:long_name = "Time" ;
        time:base_date = 2004, 1, 1, 0 ;
        time:units = "days since 2004-01-01 0:00:00 00:00" ;
        time:standard_name = "time" ;
    float lon(y, x) ;
        lon:long_name = "Longitude" ;
        lon:units = "degrees_east" ;
        lon:standard_name = "longitude" ;
    float lat(y, x) ;
        lat:long_name = "Latitude" ;
        lat:units = "degrees_north" ;
        lat:standard_name = "latitude" ;
float air_u(time, y, x) ;
    air_u:long_name = "Eastward Air Velocity" ;
    air_u:units = "m/s" ;
    air_u:missing_value = -99999.f ;
    air_u:_FillValue = -99999.f ;
    air_u:standard_name = "eastward_wind" ;
float air_v(time, y, x) ;
    air_v:long_name = "Northward Air Velocity" ;
    air_v:units = "m/s" ;
    air_v:missing_value = -99999.f ;
    air_v:_FillValue = -99999.f ;
    air_v:standard_name = "northward_wind" ;

```

*global attributes:*

```

:file_type = "Full_Grid" ;
:Conventions = "COARDS" ;
:grid_type = "curvilinear" ;
:title = "Forecast: wind+tide+river" ;

```

data:

```

time = 208.4688, 208.4792, 208.4896, 208.5, 208.5104, 208.5208, 208.5312,
208.5417, 208.5521, 208.5625, 208.5729, 208.5833,;

```

```

}

```

### 1.3.5 Data in Multiple Files: When your NetCDF files start to get too big.

Longer simulations require more model data, and that can cause problems with putting the entire time-series into one data file. GNOME allows you to break the time-series into separate files using a master file to identify all the pieces of the time-series in order. This also makes using a series of nowcasts and forecasts strung together to make a time-series. This technique worked well during the 2002 T/V *Prestige* incident in Spain.

Create a text master file with the list of file pathnames (relative to the GNOME directory) in order. The full path name is needed if the files are not in the same directory as GNOME, or in a subdirectory. The file will also need a header line, *NetCDF Files*.

To load the winds in GNOME, load this master file (e.g. §1.3.5.1 *Example – Filename: MyMasterFileEx.txt*). GNOME will use this as the list of files for the time-series.

#### 1.3.5.1 Example – Filename: MyMasterFileEx.txt

```

NetCDF Files
[FILE] :day1.nc

```

[FILE] :day2.nc  
[FILE] :day3.nc  
[FILE] :day4.nc  
[FILE] :day5.nc  
[FILE] :day6.nc

## 2 GNOME Output File Formats

### 2.1 MOSS Files for GIS Systems

GNOME outputs MOSS files 3 through 7:

- File 3: Header information, such as scenario information and any caveats.
- File 4: Positions for Best Guess (Forecast) Lagrangian elements (LEs).
- File 5: Attributes of each of the LEs in File 4.
- File 6: Same as File 4 for the Minimum Regret (Uncertainty) LEs.
- File 7: Same as File 5 for the Minimum Regret (Uncertainty) LEs.

The file formats are documented extensively in HAZMAT Report 96-4, “Digital Distribution Standard for NOAA Trajectory Analysis Information,” January 1996, J. A. Galt, D. L. Payton, H. Norris, and C. Friel<sup>3</sup>. We have not provided example files since you can easily export your own examples from GNOME’s GIS or Diagnostic Modes given a Location File<sup>4</sup>.

### 2.2 NetCDF LE Output File Format

Below is an example of the NetCDF format for outputting the model LEs. GNOME produces a single file either for a particular time or for the whole model run. It is important to note that each of these attributes grows along the data axis, which is distinguished from the time axis. For example, the attribute *particle\_count* – which grows along the time axis – describes the number of particles being tracked at every time-step. If the model has been configured for uncertainty, an additional file will be created for Minimum Regret.

#### 2.2.1 Example – Contents of NetCDF LE File from Whole 2-D Model Run

Format:

```
classic
Global Attributes:
  comment      = 'Particle output from the NOAA GNOME model'
  creation_date = '2012-09-11 09:38:00'
  source       = 'GNOME version 1.3.5'
  references    = 'http://response.restoration.noaa.gov/gnome'
  feature_type = 'particle_trajectories'
  institution   = 'NOAA Emergency Response Division'
  conventions   = 'CF-1.6'
Dimensions:
  time          = 241
  data          = 241000 (UNLIMITED)
```

<sup>3</sup> Appendix G of this report provides samples of five trajectory analysis files; samples 3-5 correspond to the above descriptions of File 3, File 4, and File 5.

<sup>4</sup> Available at <http://response.restoration.noaa.gov/oil-and-chemical-spills/oil-spills/response-tools/gnome-location-files-and-associated-resources.html>

Variables:

```

time
  Size:          241x1
  Dimensions:    time
  Datatype:      double
  Attributes:
    units        = 'seconds since 2012-09-11 09:00:00'
    long_name     = 'time'
    standard_name = 'time'
    calendar      = 'gregorian'

particle_count
  Size:          241x1
  Dimensions:    time
  Datatype:      int32
  Attributes:
    units        = '1'
    long_name     = 'number of particles in a given timestep'
    ragged_row_count = 'particle count at nth timestep'

longitude
  Size:          241000x1
  Dimensions:    data
  Datatype:      single
  Attributes:
    long_name     = 'longitude of the particle'
    units         = 'degrees_east'

latitude
  Size:          241000x1
  Dimensions:    data
  Datatype:      single
  Attributes:
    long_name     = 'latitude of the particle'
    units         = 'degrees_north'

mass
  Size:          241000x1
  Dimensions:    data
  Datatype:      single
  Attributes:
    units         = 'grams'

age
  Size:          241000x1
  Dimensions:    data
  Datatype:      int32
  Attributes:
    description   = 'from age at time of release'
    units         = 'seconds'

flag
  Size:          241000x1
  Dimensions:    data
  Datatype:      int8
  Attributes:
    long_name     = 'particle status flag'
    valid_range   = [0.00e+00 5.00e+00]
    flag_values   = [1.00e+00 2.00e+00 3.00e+00 4.00e+00]
    flag_meanings = 'on_land off_maps evaporated below_surface'

id
  Size:          241000x1
  Dimensions:    data
  Datatype:      int32
  Attributes:
    Description   = 'particle ID'
    units         = '1'

```

## 2.2.2 Example – Contents of NetCDF LE File from Whole (pseudo)3-D Model Run

Format:

classic

Global Attributes:

comment = 'Particle output from the NOAA GNOME model'  
creation\_date = '2012-09-11 10:18:00'  
source = 'GNOME version 1.3.5'  
references = 'http://response.restoration.noaa.gov/gnome'  
feature\_type = 'particle\_trajectories'  
institution = 'NOAA Emergency Response Division'  
conventions = 'CF-1.6'

Dimensions:

time = 289  
data = 229803 (UNLIMITED)

Variables:

time

Size: 289x1  
Dimensions: time  
Datatype: double  
Attributes:  
units = 'seconds since 2010-01-24 00:00:00'  
long\_name = 'time'  
standard\_name = 'time'  
calendar = 'gregorian'

particle\_count

Size: 289x1  
Dimensions: time  
Datatype: int32  
Attributes:  
units = '1'  
long\_name = 'number of particles in a given timestep'  
ragged\_row\_count = 'particle count at nth timestep'

longitude

Size: 229803x1  
Dimensions: data  
Datatype: single  
Attributes:  
long\_name = 'longitude of the particle'  
units = 'degrees\_east'

latitude

Size: 229803x1  
Dimensions: data  
Datatype: single  
Attributes:  
long\_name = 'latitude of the particle'  
units = 'degrees\_north'

depth

Size: 229803x1  
Dimensions: data  
Datatype: single  
Attributes:  
long\_name = 'particle depth below sea surface'  
units = 'meters'  
axis = 'z positive down'

mass

Size: 229803x1  
Dimensions: data  
Datatype: single  
Attributes:  
units = 'grams'

age

Size: 229803x1

```

Dimensions:    data
Datatype:     int32
Attributes:
  description  = 'from age at time of release'
  units        = 'seconds'
flag
Size:         229803x1
Dimensions:   data
Datatype:    int8
Attributes:
  long_name    = 'particle status flag'
  valid_range  = [0.00e+00 5.00e+00]
  flag_values  = [1.00e+00 2.00e+00 3.00e+00 4.00e+00]
  flag_meanings = 'on_land off_maps evaporated below_surface'
id
Size:         229803x1
Dimensions:   data
Datatype:    int32
Attributes:
  Description  = 'particle ID'
  units        = '1'

```

### 3 GNOME and GNOME Analyst

#### 3.1 Custom Logo on Output

Both GNOME and GNOME Analyst can have a custom logo added to their output products. If no custom logo is added, MOSS files will include the GNOME logo, and GNOME Analyst output will include only the GNOME logo in the upper-right corner. (The custom logo is added to the upper-left corner).

1. Create a bitmap called *logo.bmp* that contains the desired graphic.
2. Place *logo.bmp* into the application folder.
  - a. If you are writing MOSS files, then *logo.bmp* is written into the .MS3 file and *logo.bmp* is copied into the directory where the .MS\* files are being saved. Environmental Systems Research Institute, Inc.'s (Esri's) *ArcGIS for Desktop - Basic* (formerly *ArcView*) uses a 150 × 150 pixel file.
  - b. If you are working with GNOME Analyst, the file *logo.bmp* will be used to draw the logo on the upper-left of the output materials (viz. printed page, bitmap). GNOME Analyst uses a 48 × 48 pixel file.