USERS’ GUIDE for the Community release of the GFDL Vortex Tracker

September 2013
Version 3.5b

The Developmental Testbed Center

Shaowu Bao  NOAA/ESRL/GSD and CIRES/CU and currently affiliated with S. Carolina Coastal University
Donald Stark  NCAR/RAL/JNT
Ligia Bernardet  NOAA/ESRL/GSD and CIRES/CU

Please send questions to:  wrfhelp@ucar.edu
## TABLE OF CONTENTS

### Chapter 1. Introduction

### Chapter 2. Installation

- 2.1 Downloading the GFDL Vortex Tracker Code ........................................... 4
- 2.2 Basic System Requirements, Libraries and Tools ........................................ 4
- 2.3 Building the GFDL Vortex Tracker .................................................................. 5

### Chapter 3. Executables

- 3.1 `gribindex.exe`
- 3.2 `tracker.exe` ................................................................................................. 9
- 3.3 `vint.exe` ...................................................................................................... 19
- 3.4 `tave.exe` ...................................................................................................... 19

### Chapter 4. GFDL Vortex Tracker’s Additional Capabilities

- 4.1 How to Generate Phase Space Diagnostics ................................................. 20
- 4.2 How to Run the Tracker in Cyclogenesis Mode ............................................. 20

### Chapter 5. Test Case

### Chapter 6. How to Plot the Tracker Output Using ATCF_PLOT

2
Chapter 1. Introduction

The Geophysical Fluid Dynamics Laboratory (GFDL) vortex tracker is a program that ingests model forecasts in GRIB1 format, objectively analyzes the data to provide an estimate of the vortex center position (latitude and longitude), and tracks the storm for the duration of the forecast. Additionally, it reports metrics of the forecast storm, such as intensity (maximum 10-m winds and minimum mean sea level pressure - MSLP) and structure (wind radii for 34, 50 and 64 knot thresholds in each quadrant) at each output time. The GFDL vortex tracker requires the forecast grids to be on a cylindrical equidistant, latitude-longitude (lat/lon) grid. The output files contain the vortex position, intensity and structure information in a modified Automated Tropical Cyclone Forecast (ATCF) format.

The hurricane center is tracked by searching for the average of the maximum or minimum of several parameters in the vicinity of an input first guess position of the targeted vortex. The primary fields used for tracking are relative vorticity at 850 hPa and 700 hPa, MSLP, and geopotential height at 850 and 700 hPa. Secondly, wind speed at 10 m, 850 hPa and 700 hPa are used. Winds at 500 hPa are used, together with other parameters, for advecting the storm and creating a first guess position for all times beyond initialization. Many parameters are used in order to provide more accurate position estimates for weaker storms, which often have poorly defined structures/centers. The GFDL tracker code is able to function when certain input fields are missing or when certain alternate fields are provided instead of the primary fields. However, the Developmental Testbed Center (DTC) will only provide user support for running the tracker with all the fields listed above. Advanced users may see subroutine getdata in the source code to understand how the tracker can be used with alternate fields.

Besides the forecast file in GRIB1 format, the vortex tracker also ingests a GRIB index file, which is generated by running the program grbindex. The program grbindex was developed by the National Centers for Environmental Prediction (NCEP) and is distributed by the DTC as part of this GFDL vortex tracker release.

This software also finds and tracks cyclogenesis and identifies cyclone thermodynamic phases. The identification of cyclone thermodynamic phases requires that the input data contains temperature every 50 hPa from 300 to 500 hPa (for the vtt scheme) or the geopotential height every 50 hPa from 300 to 900 hPa (for the cps scheme) (see 4.1 and 4.2).

More information about the GFDL vortex tracker can be found in the Hurricane Weather Research and Forecasting (HWRF) model scientific documentation at http://www.dtcenter.org/HurrWRF/users/docs/scientific_documents/HWRFv3.5a_ScientificDoc.pdf. Please send all questions to wrfhelp@ucar.edu.
Chapter 2. Installation

2.1 Downloading the GFDL Vortex Tracker Code

GFDL vortex tracker can be obtained through the HWRF website


New users must first register before downloading the source code. Returning users need only provide their registration email address. A successful download produces three compressed tar files,

gfdl-vortextracker_v3.5b.tar.gz, tracker_util_v3.5b.tar.gz and test_data.tar.gz

The tar files can be unpacked by use of the GNU command, gunzip

```
gunzip *.tar.gz
```

and the tar files extracted by running `tar –xf` individually on each of the tar files:

```
tar -xvf gfdl-vortextracker_v3.5a.tar
    tar -xvf tracker_util_v3.5a.tar
    tar -xvf test_data.tar
```

Once unpacked, there should be three directories:

gfdl-vortextracker, tracker_util and test_data

Directory gfdl-vortextracker contains the GFDL vortex tracker code, while tracker_util contains the grbindex program and the libraries needed to build the GFDL vortex tracker. The directory test_data contains sample input data and namelist files for users to test the installed GFDL vortex tracker.

2.2 Basic System Requirements, Libraries and Tools

The source code of the GFDL vortex tracker is in the form of programs written in FORTRAN and C. The build system relies on use of the Perl scripting language and the `make` command.

The basic requirements for building and running the vortex tracker are the following:
- FORTRAN compiler
- C compiler
- Perl
- Make

Because these tools are typically the purview of system administrators to install and maintain, they are lumped together here as part of the basic system requirements.

The GFDL vortex tracker has been tested on a variety of computing platforms. Currently it is actively supported on Linux computing platforms using both the Intel and PGI compilers. While the build system provides legacy support for IBM AIX platforms, the unavailability of AIX test platforms means all AIX support is cursory at best.

This release of the GFDL vortex tracker has been tested on Linux PGI v13 and Linux Intel v12. Unforeseen build issues may occur when using an older compiler version. As always, the best results come from using the most recent version of compilers.

The `tracker_util` directory includes two libraries in the `tracker_util/libs/src/` and a utility tool in the `tracker_util/tools/grbindex` directories. These libraries and utility tool are built automatically when the `tracker_util` is built. These including libraries and utility tool are:

- BACIO library
- W3 library
- grbindex.exe

### 2.3 Building the GFDL Vortex Tracker

You must build `tracker_util` before building the GFDL vortex tracker. The GFDL vortex tracker requires two external libraries: W3 and BACIO. These libraries are located in the `tracker_util/libs/` directory and should be available if the `tracker_util` was successfully built.

#### 2.3.1 Building the libraries needed by the GFDL vortex tracker

To configure `tracker_util` for compilation, from within the `tracker_util` directory, type:

```
./configure
```

The configure script checks the system hardware, and asks the user to choose a configuration supported by the current machine architecture.

For Linux three options are available:
1. Linux x86_64, PGI compiler w/LAPACK (serial)
2. Linux x86_64, Intel compiler w/MKL (serial)
3. Linux x86_64, Intel compiler w/LAPACK (serial)

Pick option 2 if your Intel compiler includes the Math Kernel Libraries (MKL) and option 3 if it does not.

For the IBM, only one choice is available:

1. AIX (serial)

If successful, the configure script creates a file called configure.tracker_util in the tracker_util directory. This file contains compilation options, rules, and paths specific to the current machine architecture, and can be edited to change compilation options, if desired.

In csh/tcsh, to compile the tracker libraries and save the build output to a log file, type:

```
./compile |& tee compile.log
```

In bash/ksh, type:

```
./compile 2>&1 | tee compile.log
```

To remove all object files, type:

```
./clean
```

To conduct a complete clean which removes all built files, including the executables, libraries, and the configure.tracker_util, type:

```
./clean -a
```

A complete clean is strongly recommended if the compilation failed or if the configuration file is changed.

If the compilation is successful, it will create three libraries in the directory tracker_util/libs/

- libbacio.a: BACIO library
- libw3_i4r4.a: W3 library built with 32 bit integers and reals
- libw3_i4r8.a: W3 library built with 32 bit integers and 64 bit reals

These libraries will be used by the GFDL vortex tracker. You must set the environment variables LIB_W3_PATH and LIB_BACIO_PATH to the absolute path of this tracker_util/libs/ directory.
For csh/tcsh, the library paths to be set are:

```
setenv LIB_W3_PATH /path/to/tracker_util/libs/
setenv LIB_BACIO_PATH /path/to/tracker_util/libs/
```

In bash/ksh, the library paths to be set are:

```
export LIB_W3_PATH=/path/to/tracker_util/libs/
export LIB_BACIO_PATH=/path/to/tracker_util/libs/
```

The program `grbindex.exe` will be created in the directory `tracker_util/exec`

### 2.3.2 Building the GFDL vortex tracker

To configure the vortex tracker for compilation, from within the `gfdl-vortextacker` directory, type:

```
./configure
```

The configure script checks the system hardware, and if the path variables are not set, asks for software paths to the W3 and BACIO libraries. It concludes by asking the user to choose a configuration supported by current machine architecture.

For Linux, the options are:

1. Linux x86_64, PGI compiler (serial)
2. Linux x86_64, Intel compiler (serial)
3. Linux x86_64, Intel compiler super debug (serial)
4. Linux x86_64, PGI compiler, SGI MPT (serial)
5. Linux x86_64, Intel compiler, SGI MPT (serial)
6. Linux x86_64, Intel compiler, IBM POE (serial)

For the IBM, only one choice is available:

1. AIX (serial)

The configure script creates a file called `configure.trk`. This file contains compilation options, rules, and paths specific to the current machine architecture.

The configure file can be edited to change compilation options, if desired.

In csh/tcsh, to compile the vortex tracker and save the build output to a log file, type:

```
./compile |& tee tracker.log
```
In bash/ksh, type:

```
./compile 2>&1 | tee tracker.log
```

To remove all object files, type:

```
./clean
```

To completely clean all built files, object, executable, and `configure.trk`, type:

```
./clean -a
```

A complete clean is strongly recommended if the compilation failed, or if the configuration file is changed.

If the compilation was successful, three executables, `tracker.exe`, `tave.exe` and `vint.exe`, will be created in the directory `gfdd-vortextracker/trk_exec/`. 
Chapter 3. Executables

3.1 *gribindex.exe*

This executable is used to generate a GRIB index file for input in the GFDL vortex tracker.

**INPUT:**
Model forecast in GRIB1 format.

**USAGE:**

```
grindex.exe GRIB_FILE INDEX_FILE
```

**OUTPUT:**
GRIB index file.

3.2 *tracker.exe*

This is the main vortex tracking program.

**INPUT:**
Input GRIB1 model forecast and index files.

When all lead times of the postprocessed model forecast are in a single GRIB1 file (when `inp%file_seq` in the namelist is set to `onebig`), the input GRIB1 model forecast and index files include:

- **fort.11**: a single GRIB1 file containing the postprocessed model forecast
- **fort.31**: a GRIB index file generated by using the program `gribindex.exe` on the GRIB1 file fort.11

Conversely, when there are multiple input GRIB1 files, each with a forecast lead time (when `inp%file_seq` in the namelist is set to `multi`), the input GRIB1 model forecast and index files include:

```
gmodname.rundescr.atcfdescr.CCYYMMDDHH.fmin
```

Where `gmodname`, `rundescr` and `atcfdescr` are the name of the model, description of the model run and the description of the storm, respectively. These three options are specified in the namelist by `fnameinfo%gmodname`, `fnameinfo%rundescr` and
fnameinfo%atcfdescr, respectively.

CCYMMMDHHH specifies the first two digits in the 4-digit year description, year, month, day and hour of the starting time of the forecast. The first two digits of the year, the second two digits of the year, the month, the day and the hour are specified in the namelist by inp%bcc, inp%byy, inp%bmm, inp%bdd and inp%bhh, respectively.

Variable min specifies the forecast lead time (in minutes). The forecast lead time is specified in the format of 5-digit integer.

As is in the example of one big grib file, for each GRIB1 file, there is a GRIB index file with the suffix .ix. The index file is generated by using the program grbindex.exe on the GRIB1 files.

For example, the following are the input file names where the model is hwrf, the runs are coupled, and the storm is earl, the forecast starting time is 2011082312, and the forecast lead time interval is 360 minutes:

```
hwrf.coupled.earl.2011082312.f00000
hwrf.coupled.earl.2011082312.f00000.ix
hwrf.coupled.earl.2011082312.f00360
hwrf.coupled.earl.2011082312.f00360.ix
hwrf.coupled.earl.2011082312.f07560
hwrf.coupled.earl.2011082312.f07560.ix
```

Note when inp%file_seq in the namelist is set to multi, fort.11 and fort.31 are not needed.

- **fort.12**: TCVitals file containing the first guess location of the forecast vortex.

For example, the following TCVitals file (this should be a 1-line file without line break) provides a first guess location for Hurricane Earl of 19.6 N and 65.2 W

```
NHC 09L IRENE 20110823 1200 206N 0706W 295 051 0978 1008 0556 0222 0222 0241 D 0167 0111 0111 0130 72 280N 780W 0083 0056 0037 0065
```

- **fort.14**: TCVitals file used for tropical cyclonegenesis tracking. The file, can be blank, and must exist in the directory where the tracker is run even when the cyclogenesis capability is not used; otherwise the tracker will stop.

- **fort.15**: Forecast lead times (in minutes) the tracker will process.

For example, the following file specifies that the tracker will process the GRIB1 files for lead times 0, 180, 360 and 540 minutes.

```
1 0
2 180
3 360
4 540
```
Note the format of the records in this file is a 4-digit integer showing the number of the forecast lead time, followed by 1 blank space, followed by a 5-digit integer showing the forecast lead time in minutes.

- **namelist**: Namelist file with fields described on the next page.
### NAMELIST

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inp%bcc</td>
<td>First 2 digits of the year for the initial time of the forecast (e.g., the 20 in 2011)</td>
</tr>
<tr>
<td>inp%byy</td>
<td>Last 2 digits of the year for the initial time of the forecast (e.g., the 11 in 2011)</td>
</tr>
<tr>
<td>inp%bmm</td>
<td>2-digit month (01, 02, etc) for the initial time of the forecast</td>
</tr>
<tr>
<td>inp%bdd</td>
<td>2-digit day for the initial time of the forecast</td>
</tr>
<tr>
<td>inp%bhh</td>
<td>2-digit hour for the initial time of the forecast</td>
</tr>
<tr>
<td>inp%model</td>
<td>Model ID number as defined by the user. This is used in subroutine getdata to define what the GRIB IDs are for surface wind levels. Create a unique number for your model and make sure you have the corresponding IDs set up for it in subroutine getdata. For HWRF use 17. The Model ID numbers for other models are listed below: (1) GFS, (2) MRF, (3) UKMET, (4) ECMWF, (5) NGM, (6) NAM, (7) NCEP/NCAR, (8) GDAS, (10) NCEP Ensemble, (11) ECMWF Ensemble, (13) SREF Ensemble, (14) NCEP Ensemble, (15) CMC, (16) CMC Ensemble, (18) HWRF Ensemble, (19) HWRF-DAS (HDAS), (20) Ensemble RELOCATION (21) UKMET hi-res (NHC)</td>
</tr>
<tr>
<td>inp%lt_units</td>
<td>hours or minutes, this defines the lead time units used by the PDS in your GRIB header</td>
</tr>
<tr>
<td>inp%file_seq</td>
<td>onebig or multi, this specifies if the tracker will process one big input file or multiple files for each individual lead times.</td>
</tr>
<tr>
<td>inp%modtyp</td>
<td>Type of the model. Either global or regional.</td>
</tr>
<tr>
<td>inp%nesttyp</td>
<td>Type of the nest grid. Either moveable or fixed.</td>
</tr>
<tr>
<td>fnameinfo%gmodname</td>
<td>Defines the model name in the input files, e.g., hwrf. Only when inp%file_seq=multi</td>
</tr>
<tr>
<td>fnameinfo%rundescr</td>
<td>Describe the model runs in the input files, e.g., coupled. Only when inp%file_seq=multi</td>
</tr>
<tr>
<td>fnameinfo%atcfdescr</td>
<td>Describe the storm information in the input files, e.g., irene09l. Only when inp%file_seq=multi</td>
</tr>
<tr>
<td>atcfnum</td>
<td>Obsolete; can be set to any integer</td>
</tr>
<tr>
<td>atcfname</td>
<td>Character model ID that will appear in the ATCF output (e.g., GFSO, HWRF, NAHW, HCOM etc)</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>atcfymdh</td>
<td>10-digit yyyymmddhh date that will be used in output text track files</td>
</tr>
<tr>
<td>atcfreq</td>
<td>Frequency (in centahours) of output for atcfunix. Default value is 600 (six hourly).</td>
</tr>
<tr>
<td>trkrinfo%westbd</td>
<td>For genesis runs, the western boundary for searching for new storms. Does not need to match the boundaries of your grid, it can be smaller than your grid.</td>
</tr>
<tr>
<td>trkrinfo%eastbd</td>
<td>For genesis runs, the eastern boundary for searching for new storms. Does not need to match the boundaries of your grid, it can be smaller than your grid.</td>
</tr>
<tr>
<td>trkrinfo%northbd</td>
<td>For genesis runs, the northern boundary for searching for new storms. Does not need to match the boundaries of your grid, it can be smaller than your grid.</td>
</tr>
<tr>
<td>trkrinfo%southbd</td>
<td>For genesis runs, the southern boundary for searching for new storms. Does not need to match the boundaries of your grid, it can be smaller than your grid.</td>
</tr>
<tr>
<td>trkrinfo%type</td>
<td>trkrinfo%type defines the type of tracking to do. A tracker run functions as the standard TC tracker and tracks only storms from the TCVitals. Options tgen and midlat define a genesis runin which the tracker will look for new storms in addition to tracking from TCVitals. Option tgen means all parameters at the various vertical levels wil be used, while midlat means only MSLP is used and no checks are performed to differentiate tropical from non-tropical cyclones.</td>
</tr>
<tr>
<td>trkrinfo%mslpthresh</td>
<td>Threshold for the minimum MSLP gradient (units hPa km⁻¹) that must be met in order to continue tracking.</td>
</tr>
<tr>
<td>trkrinfo%v850thresh</td>
<td>Threshold for the minimum azimuthally-average 850 hPa cyclonic tangential wind speed (ms⁻¹) that must be exceeded in order to keep tracking.</td>
</tr>
<tr>
<td>trkrinfo%gridtype</td>
<td>global or regional, this defines the type of domain grid. For limited area models, choose regional.</td>
</tr>
<tr>
<td>trkrinfo%contint</td>
<td>This specifies the interval (in Pa) used by subroutine check_closed_contour to check for a closed contour in the MSLP field when running in genesis mode. Note that check_closed_contour is also called from the routine that checks for a warm core, but the contour interval is hard-wired in the executable as 1.0 degree K for that usage.</td>
</tr>
</tbody>
</table>
### Table: Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trkrinfo%out_vit</td>
<td>This is only set to y if the tracker is running in genesis mode, and it tells the tracker to write out a TCVitals record for any storms that it finds at the initial time in a forecast.</td>
</tr>
<tr>
<td>phaseflag</td>
<td>y or n, tells the program whether or not to determine the cyclone thermodynamic phase.</td>
</tr>
<tr>
<td>phasescheme</td>
<td>cps, vtt, both, tells the program which scheme to use for checking the cyclone phase. Option cps is Hart's cyclone phase space, vtt is a simple 300-500 hPa warm core check based on Vitart, and both tells the program to use both schemes. Not used if phaseflag=n.</td>
</tr>
<tr>
<td>wcore_depth</td>
<td>The contour interval (in deg K) used in determining if a closed contour exists in the 300-500 hPa temperature data, for use with the vtt scheme.</td>
</tr>
<tr>
<td>structflag</td>
<td>y or n, tells the program whether or not to compute additional diagnostics of the cyclone wind structure.</td>
</tr>
<tr>
<td>ikeflag</td>
<td>y or n, tells the program whether or not to calculate the Integrated Kinetic Energy (IKE) and Storm Surge Damage Potential (SDP).</td>
</tr>
<tr>
<td>use_waitfor</td>
<td>y or n, for waiting for input files. Use n unless for real-time operational runs.</td>
</tr>
<tr>
<td>verb</td>
<td>Level of detail printed to terminal. Choose from 0 (no output), 1 (error messages only), 2 (more messages), 3 (all messages).</td>
</tr>
</tbody>
</table>

### Usage:
When the input files and the namelist are prepared and exist in the work directory, copy or link the executable `tracker.exe` to this work directory and type

```
tracker.exe < namelist
```

### Output:
Three files are output, all in a modified ATCF format: `fort.64`, `fort.68`, and `fort.69`. When the tracker runs in cyclogenesis mode, it produces another ATCF format file: `fort.66`. And if the `ikeflag` is set to y in the namelist, still another output file will be created: `fort.74`. Most of these files are described below, and contents of file `fort.68` can be found in subroutine `outout_atcf_sink` in file `gfdd-vortextracker/trk_src/hwrf_gettrk/gettrk_main.f`. 
A sample of the vortex tracker output *fort.69* is below:

<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
<th>Column 5</th>
<th>Column 6</th>
<th>Column 7</th>
<th>Column 8</th>
<th>Column 9</th>
<th>Column 10</th>
<th>Column 11</th>
<th>Column 12</th>
<th>Column 13</th>
<th>Column 14</th>
<th>Column 15</th>
<th>Column 16</th>
<th>Column 17</th>
<th>Column 18</th>
<th>Column 19</th>
<th>Column 20</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL, 09, 2011082312, 03, HCOM, 00000, 204N, 706W, 87, 978, XX, 34, NEQ, 0103, 0077, 0058, 0095, 0, 0, 24</td>
<td>AL, 09, 2011082312, 03, HCOM, 00000, 204N, 706W, 87, 978, XX, 50, NEQ, 0058, 0040, 0031, 0055, 0, 0, 24</td>
<td>AL, 09, 2011082312, 03, HCOM, 00000, 204N, 706W, 87, 978, XX, 64, NEQ, 0043, 0025, 0016, 0042, 0, 0, 24</td>
<td>AL, 09, 2011082312, 03, HCOM, 00600, 208N, 714W, 95, 964, XX, 34, NEQ, 0155, 0100, 0058, 0145, 0, 0, 21</td>
<td>AL, 09, 2011082312, 03, HCOM, 00600, 208N, 714W, 95, 964, XX, 50, NEQ, 0066, 0057, 0037, 0060, 0, 0, 21</td>
<td>AL, 09, 2011082312, 03, HCOM, 00600, 208N, 714W, 95, 964, XX, 64, NEQ, 0046, 0033, 0028, 0042, 0, 0, 21</td>
<td>AL, 09, 2011082312, 03, HCOM, 01200, 208N, 722W, 94, 963, XX, 34, NEQ, 0123, 0096, 0060, 0109, 0, 0, 23</td>
<td>AL, 09, 2011082312, 03, HCOM, 01200, 208N, 722W, 94, 963, XX, 50, NEQ, 0069, 0048, 0049, 0062, 0, 0, 23</td>
<td>AL, 09, 2011082312, 03, HCOM, 01200, 208N, 722W, 94, 963, XX, 64, NEQ, 0045, 0032, 0036, 0046, 0, 0, 23</td>
<td>AL, 09, 2011082312, 03, HCOM, 01800, 214N, 728W, 100, 960, XX, 34, NEQ, 0093, 0082, 0060, 0084, 0, 0, 19</td>
<td>AL, 09, 2011082312, 03, HCOM, 01800, 214N, 728W, 100, 960, XX, 50, NEQ, 0054, 0051, 0042, 0050, 0, 0, 19</td>
<td>AL, 09, 2011082312, 03, HCOM, 01800, 214N, 728W, 100, 960, XX, 64, NEQ, 0040, 0037, 0034, 0037, 0, 0, 19</td>
<td>AL, 09, 2011082312, 03, HCOM, 01800, 214N, 728W, 100, 960, XX, 64, NEQ, 0040, 0035, 0035, 0039, 0, 0, 18</td>
<td>AL, 09, 2011082312, 03, HCOM, 02400, 218N, 734W, 94, 959, XX, 34, NEQ, 0131, 0097, 0063, 0116, 0, 0, 18</td>
<td>AL, 09, 2011082312, 03, HCOM, 02400, 218N, 734W, 94, 959, XX, 50, NEQ, 0061, 0054, 0043, 0056, 0, 0, 18</td>
<td>AL, 09, 2011082312, 03, HCOM, 02400, 218N, 734W, 94, 959, XX, 64, NEQ, 0040, 0035, 0035, 0039, 0, 0, 18</td>
<td>AL, 09, 2011082312, 03, HCOM, 03000, 225N, 741W, 99, 955, XX, 34, NEQ, 0141, 0106, 0065, 0105, 0, 0, 24</td>
<td>AL, 09, 2011082312, 03, HCOM, 03000, 225N, 741W, 99, 955, XX, 50, NEQ, 0069, 0055, 0045, 0057, 0, 0, 24</td>
<td>AL, 09, 2011082312, 03, HCOM, 03000, 225N, 741W, 99, 955, XX, 64, NEQ, 0044, 0044, 0036, 0041, 0, 0, 24</td>
<td>AL, 09, 2011082312, 03, HCOM, 03600, 232N, 749W, 109, 952, XX, 34, NEQ, 0135, 0104, 0075, 0103, 0, 0, 18</td>
</tr>
</tbody>
</table>

Column 1: basin name. *AL* represents Atlantic and *EP* northeast Pacific.

Column 2: ATCF storm ID number. Irene was the 9th storm in the Atlantic Basin in 2011.

Column 3: model starting time.

Column 4: constant and 03 simply indicates that this record contains model forecast data.

Column 5: model ATCF name.

Column 6: forecast lead time in hours multiplied by 100 (e.g, 00900 represents 9.00 h).

Column 7-8: vortex center position (latitude and longitude multiplied by 10).

Column 9: vortex maximum 10-m wind (in kt).

Column 10: vortex minimum MSLP (in hpa).

Column 11: placeholder for character strings that indicate whether the storm is a depression, tropical storm, hurricane, subtropical storm etc. Currently, that storm type character string is only used for the observed storm data in the NHC Best Track data set.

Column 12: thresholds wind speed in knots, an identifier that indicates whether this record contains radii for the 34-, 50- or 64-knot wind thresholds.

Column 13: *NEQ* indicates that the four radii values that follow will begin in the northeast quadrant and progress clockwise.

Column 14-17: wind radii (in nm) for the threshold winds in each quadrant.

Column 18-19: not used.

Column 20: radius of maximum winds, in nautical miles.
A sample of the vortex tracker output *fort.64* is listed below:

| Column 1-20: | same as *fort.69* except that column 6, the forecast lead time, instead of being a 5-digit integer as in *fort.69*, is a 3-digit integer. |
| Column 21-35: | space fillers |
| Column 36: | *THERMO PARAMS*, indicating that thermodynamics parameters will follow. |
| Column 37-39: | The three cyclone phase space parameters, and all values shown have been multiplied by a factor of 10. The values are listed in the following order: (1) Parameter B (left-right thickness asymmetry); (2) Thermal wind (warm/cold core) value for lower troposphere (900-600 hPa); and (3) Thermal wind value for upper troposphere (600-300 hPa). |
| Column 40: | Presence of a warm core. In this sample it is *U*, which stands for *undetermined*, meaning the warm core check was not performed. When the warm core check is performed, this field will be either *Y* or *N*, indicating whether the warm core is identified or not. |
| Column 41: | warm core strength x 10 (in degrees). It indicates the value of the contour interval that was used in performing the check for the warm core in the 300-500 hPa layer. |
| Column 42-43: | constant strings. |
A sample of the vortex tracker output `fort.66` is listed below:

| TG, 0001, 2011082312_F000_204N_0706W_FOF, 2011082312, 03, HCOM, 000, 204N, 706W, 87, 978, XX, 34, NEQ, 0103, 0077, 0058, 0095, 1105, 56, 24, -999, -9999, -9999, U, 288, 39, 1191, 6649, 1186, 5714 |
| TG, 0001, 2011082312_F000_204N_0706W_FOF, 2011082312, 03, HCOM, 000, 204N, 706W, 87, 978, XX, 64, NEQ, 0043, 0027, 0019, 0041, 1105, 56, 24, -999, -9999, -9999, U, 288, 39, 1191, 6649, 1186, 5714 |
| TG, 0001, 2011082312_F000_204N_0706W_FOF, 2011082312, 03, HCOM, 006, 208N, 714W, 94, 965, XX, 34, NEQ, 0156, 0096, 0059, 0145, 17, 21, -999, -9999, -9999, U, 292, 45, 1164, 3306, 1116, 3302 |
| TG, 0001, 2011082312_F000_204N_0706W_FOF, 2011082312, 03, HCOM, 006, 208N, 714W, 94, 965, XX, 64, NEQ, 0047, 0031, 0030, 0044, 17, 21, -999, -9999, -9999, U, 282, 42, 1187, 3668, 1122, 2694 |
| Column 1: TG, the basin id for cyclogenesis (when `trkrinfo%type` is set to `midlat`, this id is named `ML`) |
| Column 2: the number of cyclogenesis the tracker identified. |
| Column 3: the ID for the cyclogenesis, `${YYYYMMDDHH}_F${FFF}_$Lat_$Lon_FOF` where `YYYYMMDDHH`, `FFF`, `Lat` and `Lon` are the model initialization time, the forecast lead time, the latitude and the longitude, respectively, in which the cyclogenesis was first identified. |
| Column 4-18: same as Columns 3-17 in `fort.64` |
| Column 19: pressure of last closed isobar (in hPa) |
| Column 20: radius of last closed isobar (nm) |
| Column 21: radius of maximum wind (nm) |
| Column 22-24: The cyclone phase space parameters, and all values shown have been multiplied by a factor of 10. The values are listed in the following order: (1) Parameter B (left-right thickness asymmetry); (2) Thermal wind (warm/cold core) value for lower troposphere (900-600 hPa); (3) Thermal wind value for upper troposphere (600-300 hPa). |
| Column 25: Presence of a warm core. In this sample it is `U`, which stands for `undetermined`, meaning the warm core check is not performed. When the warm core check is performed, this field will be either `Y` or `N`, indicating whether the warm core is identified or not. |
| Column 26: storm moving direction (in degrees) |
| Column 27: storm moving speed (in m s\(^{-1}\)) |
| Column 28: mean 850-hpa vorticity (s\(^{-1}\)x10\(^5\)) |
| Column 29: max (gridpoint) 850-hpa vorticity (s\(^{-1}\)x10\(^5\)) |
| Column 28: mean 700-hpa vorticity (s\(^{-1}\)x10e5) |
| Column 29: max (gridpoint) 700-hpa vorticity (s\(^{-1}\)x10\(^5\)) |
A sample of the vortex tracker output `fort.74` is listed below:

<table>
<thead>
<tr>
<th>Column 1-11</th>
<th>Same as <code>fort.64</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Column 12-13</td>
<td>Fixed fields</td>
</tr>
<tr>
<td>Column 14</td>
<td>Wind damage potential (not computed in this version, therefore is always zero)</td>
</tr>
<tr>
<td>Column 15</td>
<td>Storm surge damage potential (SDP) (multiplied by 10)</td>
</tr>
<tr>
<td>Column 16-18</td>
<td>IKE, in terajoule, for 10 ms⁻¹, 18 ms⁻¹ and 33 ms⁻¹ winds, respectively</td>
</tr>
<tr>
<td>Column 19-21</td>
<td>IKE for 25-40 ms⁻¹, 41-54 ms⁻¹ and 55 ms⁻¹ winds, currently not computed, therefore are always zero</td>
</tr>
<tr>
<td>Column 22-23</td>
<td>Vortex center position (latitude and longitude multiplied by 100)</td>
</tr>
</tbody>
</table>
3.3 vint.exe

This is a program to interpolate from various pressure levels onto a regularly spaced grid, with 50-hpa vertical level intervals. Each run only processes one lead time. Therefore it is necessary to use this executable separately for each lead time.

**INPUT:**

- `fort.11`: GRIB1 file containing the postprocessed model output containing at least two levels of temperature data: 300 and 500 hpa.
- `fort.16`: text file containing the number of input pressure levels.
- `fort.31`: index file of `fort.11`
- `namelist`: text file containing the following
  ```
  &timein ifcsthour=${fhour} iparm=${gparm}/
  ```
  where `${fhour}` is the forecast lead time and `${gparm}` is the variable to be processed. For phase space diagnostics, geopotential height (when `phasescheme=cps`, `${gparm}`=7) or temperature (when `phasescheme=vtt`, `${gparm}`=11) or both (when `phasescheme=both`) need to be processed.

**USAGE:**

```
vint.exe < namelist
```

**OUTPUT:**

- `fort.51`: GRIB1 file containing the temperature data on vertical levels 300, 350, 400, 450 and 500 hpa.

3.4 tave.exe

This is a program to vertically average temperature in the 500-300 hpa layer.

**INPUT:**

- `fort.11`: GRIB1 file containing the temperature at least at levels 300, 350, 400, 450 and 500 hpa. This file can be generated by `vint.exe`
- `fort.16`: text file containing the number of input pressure levels.
- `fort.31`: index file of `fort.11`
- `namelist`: text file containing the following
  ```
  &timein ifcsthour=${fhour}, iparm=11/
  ```

**USAGE:**

```
tave.exe < namelist
```

**OUTPUT:**

- `fort.51`: GRIB1 file containing the mean temperature in the 300-500 hpa layer.
Chapter 4. GFDL Vortex Tracker’s Additional Capabilities

4.1 How to Generate Phase Space Diagnostics

To use this function, the user should run the following procedures.

1) In the GFDL vortex tracker namelist set the following items:
   
   \begin{verbatim}
   phaseflag=y,
   phasescheme=both or cp’ or vtt
   wcore_depth=1.0
   \end{verbatim}

2) If phasescheme is set to \textit{cps}, run \texttt{vint.exe} (see 3.3) to vertically interpolate the geopotential from 300 to 900 \text{hPa} at a 50 \text{hPa} interval. Then append these geopotential variables to the tracker’s GRIB1 format input file.

3) If phasescheme is set to \textit{vtt}, run \texttt{vint.exe} (see 3.3) to vertically interpolate the temperature from 300 to 500 \text{hPa} at a 50 \text{hPa} interval. Then run \texttt{tave.exe} (see 3.4) to obtain the average temperature between 300 and 500 \text{hPa}. This average temperature field is appended to the tracker’s GRIB format input file.

4) If phasescheme is set to \textit{both}, then both steps 2) and 3) are needed.

5) When the phase space diagnostics is performed, the output will be generated in \texttt{fort.64} as fields 37-41 (see 3.2).

4.2 How to Run the Tracker in Cyclogenesis Mode

To use this function, the user should run the following procedures.

1) In the GFDL vortex tracker namelist set the following items:

\begin{verbatim}
trkrinfo\%westbd
trkrinfo\%eastbd
trkrinfo\%southbd
trkrinfo\%northbd
\end{verbatim}

They are the boundaries for searching for new storms in cyclogenesis mode. They do not need to match the boundaries of the forecast grid.

2) In the GFDL vortex tracker namelist, set the item \texttt{trkrinfo\%type=tcgen} or \texttt{trkrinfo\%type=midlat} (for the difference between \textit{tcgen} and \textit{midlat}, see 3.1).

3) The tracker in cyclogenesis mode requires that the files \texttt{fort.12} and \texttt{fort.14} exist in the working directory, but these two files can be blank, as created by the commands \texttt{“touch fort.12”} and \texttt{“touch fort.14”}, respectively.

4) In addition to \texttt{fort.64} and \texttt{fort.69}, another ATCF format output file, \texttt{fort.66}, will be produced by the tracker when it runs in cyclogenesis mode.
Chapter 5. Test Case

Sample input data and namelists are provided. Refer to Section 2.1 for downloading and uncompressing the test case data file. The sample input data are the HWRF forecast of Hurricane Irene in 2011, starting from 2011 August 23rd 12 UTC. We recommend that the users test the installed GFDL vortex tracker program with the included sample input data and namelist before they work on their own model output.

The directory of the sample input data, test_case, contains two sub-directories: onebig and multi. Onebig contains the sample input data that is provided as one single GRIB1 file (when inp%file_seq in the namelist is set to onebig), while multi contains the same input data but provided as multiple GRIB1 files each of which contains the forecast for one forecast lead time (when inp%file_seq in the namelist is set to multi).

To test the installed GFDL vortex tracker using the single GRIB1 input data file provided, enter the directory test_case/onebig, and check the following files:

- forti1: link to 2011082312.grib.HCOM.
- forti3: link to 2011082312.grib.HCOM.ix.
- forti5: link to fcst_minutes.
- forti2: link to TCVitals file.
- tcvital.as: TCVitals file containing the first guess location of the forecast vortex.
- Namelist.onebig : namelist for tracker.exe.
- forti4 : TCVitals file used for tropical cyclonegenesis tracking. This file is currently not used, but must exist. It can be a blank file generated by “touch forti4” command.
- 2011082312.grib.HCOM: GRIB1 model output file.
- 2011082312.grib.HCOM.ix: GRIB index file.
- fcst_minutes: forecast lead times (in minutes).

Now link or copy the executable tracker.exe to test_case/onebig directory and type

```
tracker.exe < namelist.onebig
```

If the program runs successfully, the output files forti64 and forti69 should be generated.

To test the installed GFDL vortex tracker using the multiple GRIB1 input data files provided, enter the directory test_case/multi, and check for the following files.

- forti5: link to fcst_minutes.
- forti2: link to TCVitals file.
- tcvital.as: TCVitals file containing the first guess location of the forecast vortex.
- namelist.multi : namelist for tracker.exe.
- forti4 : TCVitals file used for tropical cyclonegenesis tracking. This file is currently not used, but must exist. It can be a blank file generated by “touch forti4” command.
hwrf.coupled.IRENE09L.2011082312.fMMMMMM: GRIB1 model output files, where MMM is the forecast lead time in minutes.
hwrf.coupled.IRENE90l.2011082312.fMMMMMM.ix: GRIB index files.
fctest_minutes: forecast lead times (in minutes).

Now link or copy the executable tracker.exe to test_case/multi directory and type

```
tracker.exe < namelist.multi
```

If the program runs successfully, the output files fort.64 and fort.69 should be generated.

Note that the files fort.64 and fort.69 generated in onebig and multi directories should be identical.
Chapter 6. How to Plot the Tracker Output Using ATCF_PLOT

*The GFDL vortex tracker release includes atcf_plot*, which is a set of GrADS scripts that can be used to plot hurricane track files in ATCF format. The *atcf_plot* package can be found in directory `gfdl-vortextracker/trk_plot/plottrak`.

To use *atcf_plot* to plot the storm’s track:

- Enter the directory `gfdl-vortextracker/trk_plot/plottrak`.
- Run `gribmap` on the GrADS ctl file `plottrak.ctl`. The GrADS utility `gribmap` maps what is in the ctl file with the binary data that it finds inside the actual GRIB data file. It creates a map (`plottrak.ix`) that points to the locations where the requested binary data starts for the different variables and levels. Note that `gribmap` is not provided in the GFDL vortex tracker or the `track_util` package. Rather, it is available within the GrADS package and is described on [http://www.iges.org/grads/gadoc/gradutilgribmap.html](http://www.iges.org/grads/gadoc/gradutilgribmap.html).

Create the map file by using the command:

```
gribmap -v -i plottrak.ctl
```

You should see one line in the output that has `MATCH` in the string. Both the `plottrack.ctl` and the newly created `plottrak.ix` map file need to be in the directory where `atcf_plot` will be run.

- Edit `atcfplot.sh` to set the following paths:
  1. `gradsv2`: path to the GrADS executable (for example, `/contrib/grads/bin/gradsc`).
  2. `GADDIR`: path to the directory containing the supplemental font and map files in for GrADS (for example, `/contrib/grads/lib`).
  3. `scrdir`: path to the working directory (for example, `gfdl-vortextracker/trk_plot/plottrak`).
  4. `plotdir`: path to the directory where the plot files will be created (for example, `gfdlvortextracker/trk_plot/plottrak/tracks`).

- Edit `atcfplot.gs` to define the following paths:
  1. `rundir`: same as `scrdir` in `atcfplot.sh`. Note rundir must end with a `/`.
  2. `_netdir`: same as `plotdir` in `atcfplot.sh`. Note netdir must end with a `/`.

- Edit `get_mods.sh` to define the following paths:
  1. `rundir`: same as `scrdir` in `atcfplot.sh`.
  2. `netdir`: same as `plotdir` in `atcfplot.sh`.
  3. `ndate`: path to the script `ndate.ksh`.
  4. `nhour`: path to the script `nhour.ksh`.

- Edit `get_verif.sh` to define the following paths:
  1. `rundir`: same as `scrdir` in `atcfplot.sh`. 
2. netdir: same as plotdir in atcfplot.sh.
3. ndate: path to the script ndate.ksh.
4. nhour: path to the script nhour.ksh.

- The users need to insert or append their vortex tracker output, *fort.64*, into the file $a{\{Basin\}}{\{SID\}}{\{YYYY\}}.dat$.
- After setting up the paths to the correct locations in your system, run the script using the command:

```
\texttt{atcfplot.sh \{YYYY\} \{Basin\}}
```

This will start a GUI window and read in ATCF format track files $a{\{Basin\}}{\{SID\}}{\{YYYY\}}.dat$ in $\text{\$rundir\}$ directory. Note that \{SID\} is the ID for storms in year \{YYYY\} in the \{Basin\} basin.

For example, the user can use the command “\texttt{atcfplot.sh 2011 al}” to plot the track files $aal{\{SID\}}2011.dat$ in the \$rundir\$ directory.

When the GUI window appears, from the drop down menu, select a storm, start date, and a model name (atcfname in the GFDL vortex tracker namelist), then click the Plot button to plot the track. The plots can be exported to image files by using the Main and then Print menu options.

The default tracker namelist is set to use the ATCF model name \textit{HCOM}. If the user changes this name in the tracker namelist, the ATCF_PLOT GUI will not recognize the new name. In this case, the user needs to replace an unused atcfname with the new atcfname. The atcfnames in the GUI can be found by searching in function \texttt{modnames} in file atcfplot.gs. Note all three instances of the unused atcfname need to be replaced in atcfplot.gs.

For example, if \textit{USER} was employed as the ATCF model name in the users’ GFDL Vortex Tracker output *fort.64*, file atcfplot.gs needs to be modified to have the ATCF_PLOT program GUI interface show a button for the atcfname \textit{USER}. To do that, open file atcfplot.gs, go to function \texttt{modnames}, find an atcfname that will not be used, for example \textit{HCOM}, and manually replace the string \textit{HCOM} with \textit{USER}.