Newman

Philip Du Toit

February 21, 2009

1 What is Newman?

Newman is software to compute the Finite Time Liapunov Exponent for time-dependent velocity fields. The velocity field is provided either as an analytical expression, or as a numerical data set. The computations can be performed on parallel processors or on a single processor.

Specifically, `Newman` software includes the following features:

1. **N-Dimensional Code:** The underlying code in `Newman` is dimension independent. When the code is compiled, a user-supplied flag indicates the desired dimensionality. In this way, the code is more easily maintained than maintaining a separate code for each dimension, and no sacrifice in efficiency is required.

2. **Parallel Implementation:** The code is written using the Message Passing Interface for use on either a single processor or a parallel cluster. Currently, parallel computations are performed routinely on several hundreds of processors in the Geophysical and Planetary Sciences cluster at Caltech.

3. **Fast and Modular C++ Structures:** `Newman` is written in C++ to ensure computational speed and modularity of code. Users can easily implement their own additions to the code by editing or replacing individual modules.

4. **Light-Weight Data Structures:** `Newman` can handle large data sets and high-resolution computations through efficient memory usage. All input and output is performed in binary format for fastest retrieval over the internet.

5. **Geophysical Applications:** `Newman` has many features specifically intended for geophysical applications. These include the ability to compute and plot FTLE on the sphere in 2D and 3D, or on a Mercator longitude-latitude-altitude projection; track the center of a tropical storm so that FTLE may be computed in storm-centered coordinates; use velocity data computed on multiple moving nested grids as is common for hurricane simulations; and interpolate data provided on non-uniform cartesian grids.

6. **Fast Treatment of Boundaries:** `Newman` can quickly integrate trajectories for flows with boundaries and multiple islands and appropriately compute the FTLE.

7. **Library of Integration Algorithms:** `Newman` uses the GSL library for integration and linear algebra routines. The user can choose from several different integration schemes available.

8. **Miscellaneous Features:** `Newman` allows the user to produce plots of the FTLE, drifter trajectories, and velocity fields both forward and backward in time at a temporal resolution that is independent of the temporal spacing between frames in the data set. Data sets can be specified as periodic along any dimension and in time.

2 Installation

Newman source code can be compiled using the make command. To install Newman you will need to complete the following three steps:

1. Install the MPI library,
2. Install the GSL library,
3. Make Newman,

The MPI and GSL libraries must be installed before Newman can be compiled.

2.0.1 MPI

Newman is designed for industrial strength computations in parallel. Computations can be run on a single processor, however, MPI must still be installed. Visit `http://www.open-mpi.org` to find instructions for downloading and installing the MPI library.

2.0.2 GSL

Newman uses the integration and eigensystem routines in the GNU Scientific Library. Visit `http://www.gnu.org/software/gsl/` to find instructions for downloading and installing the GSL library. Newman requires version GSL 1.10 or greater. To determine which version of GSL you currently have and where it is installed, simply execute the gsl-config command.

2.1 Make Newman

The Newman source code resides in the ./src directory. A shell script is provided that will attempt to compile Newman in two, three, and four dimensions. Make sure that the permissions on the script ./makenewman.sh are set to executable, and then run the script:

```
>./makenewman.sh
```

This will attempt to generate three executable files: `newman2D`, `newman3D`, and `newman4D` in the ./src directory.
If there are errors in compilation, or if you would like to customize the installation, you may need to edit the makefiles. The makefiles can be found in the ./src directory. The most common error encountered when compiling is ensuring that the linker can find the needed libraries.

3 Running Newman

To run Newman from the commandline, type

```
>newman2D saturn.in
```

where saturn.in is a text input file containing parameter values for your computation. Be sure that the executable is in a directory in your path.

See the Examples directory for an example of an input file. A description of each of the parameters and their default values is provided in section 6.

4 Viewing the Results

Newman outputs all results in a lightweight binary format. Separate code modules are then used to convert the raw binary format into the final desired format. This allows for:

- fast retrieval of results from a cluster over the internet,
- easier writing of modules for data conversion to the final desired format required by the user.

The format of the raw binary output files is described in section 5.

4.1 Tecplot Format

The `raw2tec` modules are provided to convert the raw binary output from Newman to Tecplot binary format. Tecplot is commercial plotting software. To compile the raw2tec modules, run the makeraw2tec.sh shell script.

```
>./makeraw2tec.sh
```

This script will attempt to compile raw2tec2D, raw2tec3D, and raw2tec4D in the ./raw2tec directory. The first step executed by the script will be to compile the tecio.a library in the ./lib/tecio/ directory. The tecio.a library is written by the Tecplot company. As the compilation proceeds, you will be requested to select your platform. I have found that selecting linux.22 as the platform works well on Unix and Mac platforms. Next, when directed, select the option: “1. Make tecio.a only”. The script should then proceed uninterrupted to the finish. The raw2tec executables will be in the ./raw2tec directory.

To learn more about these modules, type

```
>raw2tec2D -h
```

at the commandline.

The binary data can be formatted for plotting in Cartesian coordinates or on the sphere for geophysical applications. Use the `-s` flag to convert from spherical coordinates to cartesian for plotting on the sphere.

For 3D data on the sphere, use the `-a` option to provide the scale factor by which to scale the atmosphere with respect to its true size. For example, `-a 20.0` will scale the atmosphere by a factor of 20.0 relative to the sphere radius.

4.2 Matlab Format

In the ./scripts directory, you will find the raw2mat.m Matlab script. This script allows for conversion of the raw binary output to Matlab form. The Matlab conversion script raw2mat is not as well supported as the Tecplot conversion module raw2tec. Many of the features available in raw2tec still need to be included in the raw2mat.m script.

5 Output format

Details to follow soon.

6 Parameters

6.1 Atmos_Radius

Values: double
Default: 6378.1
Description: Atmos_Radius is the radius of the sphere to be used for integration on the sphere.

6.2 Atmos_Set

Values: [0 | 1]
Default: 0
Description: Atmos_Set is a switch to turn on/off integration on the sphere using Longitude and Latitude coordinates.

6.3 Boundary_InFile

Values: char*
Default: boundary.dat
Description: Boundary_InFile is the name of the file containing the list of polygon segments to be used for the boundary.

6.4 Boundary_Method

Values: [0 | 1 | 2]
Default: 0
Description: Boundary_Method determines the type of boundary checking that will be used according to the following table:
0 No enforcement of boundaries.
1 Regions outside the boundary are defined as those for which the velocity data cell contains a value that is NaN.

2 The boundary is explicitly provided as a list of polygon segments. 3 An analytical expression hard-wired into the code is evaluated to determine the boundary.

6.5 Data_Format

Values: [0 | 1 | 2]
Default: 2
Description: Data_Format determines the format of the numerical velocity data according to the following table:
0 Raw binary data
1 Tecplot ASCII data
2 ASCII lean velocity data

6.6 Data_InFile

Values: char*
Default: velocitydata
Description: Base name of the files containing the numerical velocity data.

6.7 Data_Max[ND]

Values: double
Default: 1.0
Description: Data_Max[ND] is the maximum value of grid locations in the numerical velocity data.

6.8 Data_Min[ND]

Values: double
Default: 0.0
Description: Data_Min[ND] is the minimum value of grid locations in the numerical velocity data.

6.9 Data_NonUniformGridND]

Values: [0 | 1]
Default: 0
Description: Data_NonUniformGrid[ND] is a switch to turn on/off the use of a non-uniform grid for the locations of numerical velocity data along the given dimension.

6.10 Data_NumInputFiles

Values: int
Default: 2
Description: Number of input files containing the numerical velocity data.

6.11 Data_Periodic[ND]

Values: [0 | 1]
Default: 0
Description: Data_Periodic[ND] is a switch to turn on/off periodicity of the data along the given spatial dimension.

6.12 Data_Res[ND]

Values: int
Default: 2
Description: Data_Res[ND] is the number of grid locations in the numerical velocity data.

6.13 Data_TDelta

Values: double
Default: 1.0
Description: Data_TDelta[ND] is the time interval between time slices in the numerical velocity data.

6.14 Data_TMin

Values: double
Default: 0.0
Description:

6.15 Data_TPeriodic

Values: [0 | 1]
Default: 0
Description: Data_TPeriodic is a switch to turn on/off time periodicity in the numerical velocity data.

6.16 Data_TRes

Values: int
Default: 2
Description: Data_TRes is the number of time slices in the numerical velocity data.

6.17 Filter_Width

Values: double
Default: 0.0
Description: FTLE_Width is the width of the gaussian filter to be used in post process smoothing of the FTLE.

6.18 FTLE_Compute

Values: [0 | 1]
Default: 1
Description: A switch to turn on/off computation of the Finite Time Liapunov Exponent field.

6.19 FTLE_IntTLength

Values: double
Default: 1.0
Description: FTLE_IntTLength is the integration time to be used in the definition of the FTLE.

6.20 FTLE_Max[ND]

Values: double
Default: 1.0
Description: FTLE_Max[ND] is the maximum grid location for computation of the FTLE.

6.21 FTLE_Min[ND]

Values: double
Default: 0.0
Description: FTLE_Min[ND] is the minimum grid location for computation of the FTLE.

6.22 FTLE_OutFile

Values: char*
Default: repatt
Description: FTLE_Outfile is the base name of the output file for FTLE data.

6.23 FTLE_Res[ND]

Values: int
Default: 3
Description: FTLE_ResND] is the number of grid locations for computation of the FTLE.

6.24 FTLE_TrackWidth[ND]

Values: double
Default: 1.0
Description: FTLE_TrackWidth[ND] is the width of the grid for computation of the FTLE centered about the storm center when the Track_Storm option is used.

6.25 Int_AbsTol

Values: double
Default: 1e-5
Description: Int_AbsTol is the tolerance threshold for the absolute error for an adaptive step size integrator.

6.26 Int_MaxTimeStep

Values: double
Default: 1.0
Description: Int_MaxTimeStep is the maximum allowed timestep for the RK45 integrator with adaptive step-sizing and step-size thresholds. The algorithm attempts to choose the largest step size between Int_MinTimeStep and Int_MaxTimeStep such that the error tolerances are satisfied.

6.27 Int_Method

Values: [0 | 1 | 2 | 3]
Default: 0
Description: Int_Method determines the type of integrator used to integrate trajectories according to the following table:

0	Adaptive step size integrator with strict error tolerances
1	Fixed step size integrator
2	Euler integrator
3	RK45 integrator with adaptive step-sizing and step-size thresholds.

6.28 Int_MinTimeStep

Values: double
Default: 0.1
Description: Int_MinTimeStep is the minimum allowed timestep for the RK45 integrator with adaptive step-sizing and step-size thresholds. The algorithm attempts to choose the largest step size between Int_MinTimeStep and Int_MaxTimeStep such that the error tolerances are satisfied.

6.29 Int_RelTol

Values: double
Default: 1e-6
Description: Int_RelTol is the tolerance threshold for the relative error for an adaptive step size integrator.

6.30 Int_TimeStep

Values: double
Default: 1.0
Description: Int_TimeStep is the size of the timestep for a fixed step size integrator.

6.31 MapCoord

Values: [0 | 1]
Default: 0
Description: A switch to turn on/off mapping of points to an integration space before integration of trajectories. This is convenient if the FTLE is to be plotted in a space that is different form the space in which the trajectories are to be integrated.

6.32 Nest_NumNests

Values: int
Default: 0
Description: Nest_NumNests is the number of total nests available for nested velocity grids.

6.33 Nest_List

Values: { int }
Default: 0
Description: Nest_List is an ordered list specifying which of the nested velocity sets should be used for the computation. Elements in the list can be any integer from 0 to Nest_NumNests-1 inclusive.

6.34 Output_TDelta

Values: double
Default: 1.0
Description: Output_TDelta is the desired time interval between frames of output data.

6.35 Output_TRes

Values: int
Default: 2
Description: Output_TRes is the desired number of frames of output data.

6.36 Output_T1

Values double
Default: 0.0
Description: Output_T1 is the desired time of the first frame of output data.

6.37 Parallel_LoadRatio

Values double
Default: 1.0
Description: Parallel_LoadRatio is the factor by which the number of jobs to be assigned is larger than the number of processors available. Choosing a larger ratio can help to improve load balancing among the nodes at the expense of a nominal overhead.

6.38 Parameters_Print

Values: [0 | 1]
Default: 1
Description: A switch to turn on/off printing of all parameter values to the screen.

6.39 Path_Input

Values: char*
Default: Input
Description: Path_Input is the directory in which all input files can be found.

6.40 Path_Output

Values: char*
Default: Output
Description: Path_Output is the directory where all output data generated will be placed.

6.41 Path_Work

Values: char*
Default: Work
Description: Path_Work is a directory where temporary working files used by the Master node during the computation will be placed.

6.42 Path_Scratch

Values: char*
Default: /tmp
Description: Path_Work is a directory where temporary working files used by the slave nodes during the computation will be placed. This should be a directory local to the node to avoid unnecessary transfer of files.

6.43 Plot_Max[ND]

Values: double
Default: 1.0
Description: Plot_Max[ND] is the maximum grid location for plotting of the velocity.

6.44 Plot_Min[ND]

Values: double
Default: 0.0
Description: Plot_Max[ND] is the minimum grid location for plotting of the velocity.

6.45 Plot_OutFile

Values: char*
Default: velocityplot.dat
Description: Plot_OutFile is the name of the file where the plotted velocity output data will be placed.

6.46 Plot_Res[ND]

Values: int
Default: 2
Description: Plot_ResND] is the number of grid locations for plotting of the velocity.

6.47 Plot_Velocity

Values: [0 | 1]
Default: 0
Description: A switch to turn on/off output of the velocity field for velocity data.

6.48 Query_Velocity

Values: [0 | 1]
Default: 0
Description: A switch to turn on/off querying of the velocity data at a specified point.

6.49 Query_X[ND]

Values: double
Default: 0.0
Description: Query_X[ND] are the coordinates at which the velocity will be queried and printed to the screen.

6.50 Time_Direction

Values: [-1 | 1]
Default: 1
Description: For Plot_Velocity and Trace computations, Time_Direction determines the direction of time in the output data. For FTLE_Compute, Time_Direction determines the direction of itegration of trajectories.

6.51 Time_Origin

Values: 6 int
Default: 0 0 0 0 0 0
Description: Time_Origin contains an array of six integers representing the time and date that correspond to time zero in the data. The integers represent: Year Month Day Hour Minute Second. Specifying the time origin allows the raw2tec utilities to time and datestamp the output for plotting purposes.

6.52 Trace_ColorDimension

Values: int
Default: 0
Description: Trace_ColorDimension is the dimension that will be used to color drifters according to their initial position.

6.53 Trace_Compute

Values: [0 | 1]
Default: 0
Description: A switch to turn on/off computation of passive drifter trajectories.

6.54 Trace_GenerateMesh

Values: [0 | 1]
Default: 0
Description: Trace_GenerateMesh is a switch to turn on/off the generation of a grid of initial locations for drifter trajectories.

6.55 Trace_InFile

Values: char*
Default: drifterinput.dat
Description: Trace_InFile is the name of the input file containing initial locations of drifters.

6.56 Trace_MeshColor

Values: double
Default: 0.0
Description: Trace_MeshColor is the color to be assigned to the grid of generated drifters.

6.57 Trace_MeshMax[ND]

Values: double
Default: 1.0
Description: Trace_MeshMax[ND] is the maximum grid location for initial positions of drifters.

6.58 Trace_MeshMin[ND]

Values: double
Default: 0.0
Description: Trace_MeshMin[ND] is the minimum grid location for initial positions of drifters.

6.59 Trace_MeshReleaseTime

Values: double
Default: 0.0
Description: Trace_MeshReleaseTime is the time at which the grid of drifters will be released in the flow.

6.60 Trace_MeshRes[ND]

Values: int
Default: 2
Description: Trace_MeshRes[ND] is the number of grid locations for initial positions of drifters.

6.61 Trace_OutFile

Values: char*
Default: drifteroutput.dat
Description: Trace_OutFile is the name of the output file containing drifter trajectories.

6.62 Track_InFile

Values: char*
Default: track.dat
Description: Track_File is the name of the file containing the coordinates of the track to be used for computation of FTLE for moving grids.

6.63 Track_Storm

Values: [0 | 1]
Default: 0
Description: Track_Storm is a switch to turn on/off computation of FTLE in storm centered coordinates.

6.64 V[ND]

Values: char*
Default: x
Description: List of analytical equations for the velocity field.

6.65 Velocity_Format

Values: [0 | 1 | 2]
Default: 0
Description: Velocity_Format determines the type of velocity to be used during the computation according to the following table:
0 Equations for velocity hard wired into the code at compile time in the velocity.cpp file. 1 Equations specified by the user at run time using the equation parser. 2 Velocity is to be determined from a numerical data set.

6.66 Velocity_Null[ND]

Values: [0 | 1]
Default: 0
Description: Velocity_Null[ND] is a switch to turn on/off the zeroing of the velocity along the specified dimension.