TempestExtremes

Contents

1 Setting up TempestExtremes

All of the necessary source files for compiling TempestExtremes can be found at the GitHub repository `https://github.com/ClimateGlobalChange/tempestextremes`. Users can either download a zip file and extract it into the directory of choice (here denoted as `$USER_DIR`) or

```
git clone https://github.com/ClimateGlobalChange/tempestextremes
```

into `$USER_DIR`.

Once the directory has been set up, simply use the `make` command in the base directory (`$USER_DIR/tempestextremes/`) and it will compile all of the necessary binaries.

2 Objective blocking detection methods

There are two main methods for block detection: the Z500 gradient method of Tibaldi and Molteni 1990 (hereafter referred to as TM90), and anomaly-based methods such as that of Dole and Gordon 1983 (Z500 anomaly, DG83) or Schwierz et al 2004 (potential vorticity anomaly, S04).

Geopotential height gradient (TM90) Two slopes are calculated around a single gridpoint:

$$GHGN = \frac{Z(\phi_n) - Z(\phi_0)}{\phi_n - \phi_0} \tag{1}$$

$$GHGS = \frac{Z(\phi_0) - Z(\phi_s)}{\phi_0 - \phi_s} \tag{2}$$

where ϕ_0, ϕ_n, and ϕ_s are the reference latitude and the latitudes 15° poleward and equatorward of ϕ_0, respectively, and $GHGN$ and $GHGS$ are the height gradients. For $GHGN < -10$ m/deg lat and $GHGS > 0$ m/deg lat, the point is considered instantaneously blocked.

This is a modification of the original TM90 algorithm, which was calculated for all longitude values about a single predetermined latitude. Scherrer et al 2006 and others expanded the definition to encompass all latitudes 35-75 degrees in both hemispheres.

Anomaly methods (S04 and DG83) Each anomaly method has a general workflow that can be described as follows:

1. If doing S04: Generate the input files using `BlockingPV`.
2. Calculate the long term daily average, using `BlockingDFT` (Section 2.5)
3. Calculate the anomalies from the long term daily average, using `BlockingDevs` (Section 2.6)
4. (If using a non-constant threshold) Calculate the long term daily average of the anomalies using `BlockingDFT` for use in the threshold calculations, then calculate a spatiotemporally varying threshold variable using `BlockingThresh`
5. Calculate the normalized anomalies using `BlockingNormDevs`

Potential Vorticity anomaly: S04 uses vertically averaged (150-500 hPa) potential vorticity as the variable of choice for block detection. The presence of negative (positive) anomalies in the Northern (Southern) Hemisphere denotes either a slowing or reversal of the flow with respect to the climatological mean.

Z500 anomaly: DG83 uses 500 hPa geopotential height (Z500) anomaly as the variable of choice for block detection. The anomaly signifies heights that exceed a threshold above a climatological mean.

All of the initial blocking code can be found in the `src/blocking/` directory.

2.1 Blocking Event Detection: The StitchBlobs Framework

StitchBlobs is a standardized framework for identifying persistent events (in this case, blocking). Clusters of grid points that meet minimum spatial constraints (if specified) and overlap over time are defined as individual events and tagged with unique identifiers. The corresponding BlobStats program reads in the StitchBlobs output and provides a text file with per-timestep information on size(in terms of maximum latitude/longitude extent or area) and location (in terms of the cluster centroid's latitude/longitude position). This allows the user to track events from start to finish and and examine trends in the event's spatial characteristics.

There are options to provide constraints in terms of the study region (specifying minimum and maximum latitude and longitude values), cluster size, and minimum duration (in terms of number of timesteps).

2.2 General workflow

The following sections explain the various command line parameters that control the program inputs and outputs. Parameters with an asterisk (`*`) are mandatory and will raise an exception if not included. Parameters with multiple asterisks (`**`), such as `--in` and `inlist`, are also mandatory, but only one of the two should be used; if both are used, the program will raise an exception. If there is a default value, it is included in brackets (`["string"]`); otherwise, the quotation marks indicate a default empty string (`[""]`).

Flags such as `--latname` are only necessary if the variable names differ from the default. These optional flags are marked with a plus (`+`) and are explained in Section 2.10.

Example: Calculating vertically averaged PV anomalies with a non-constant threshold: This example assumes a dataset with default axes `[time, lev, lat, lon]` with the `lev` axis initially in units of hPa.

1. Calculate vertically averaged PV from a list of input files, `inPV.txt`:
   `$USER_DIR/tempestextremes/bin/BlockingPV --hpa --inlist inPV.txt`

2. Calculate the LTDM from a list of vertically averaged PV files, `VPV.txt`:
   `$USER_DIR/tempestextremes/bin/BlockingDFT --inlist VPV.txt --out avgPV.nc --varname VPV --avgname AVPV`

3. Calculate PV anomalies from the LTDM using list `VPV.txt`:
   `$USER_DIR/tempestextremes/bin/BlockingDevs --inlist VAPV.txt --avg avgPV.nc --avgname AVPV --varname VPV --pv`

4. Calculate the LTDM of anomalies from a list of anomaly files, `ADVPV.txt` and using the smoothed anomaly, `ADVPV`:
   `$USER_DIR/tempestextremes/bin/BlockingDFT --inlist ADVPV.txt --out avgADVPV.nc --varname ADVPV --avgname AVG_ADVPV`

5. Calculate a spatiotemporally varying threshold for the normalized anomalies calculation using `ADVPV.txt` and the averaged anomalies file:
   `$USER_DIR/tempestextremes/bin/BlockingThresh --inlist ADVPV.txt --outfile PVthresh.nc --davgfile avgADVPV.nc --davgname AVG_ADVPV --dvarname ADVPV`

6. Calculate normalized anomalies using the threshold file `PVthresh.nc` and list `ADVPV.txt`:
   `$USER_DIR/tempestextremes/bin/BlockingNormDevs --inlist ADVPV.txt --varname ADVPV --thresh PVthresh.nc --threshname THRESHOLD_DFT --pv`

Example: Calculating 500 hPa geopotential height anomalies with a constant threshold: This example assumes a dataset with default axes `[time, lev, lat, lon]` with the `lev` axis initially in units of hPa and geopotential (`GEOPOTENTIAL`), rather than geopotential height, as the input variable.

1. Calculate the LTDM from a list of input files, `fList.txt`, noting that the input has 4 axes rather than the default 3 and the variable is geopotential:
   `$USER_DIR/tempestextremes/bin/BlockingDFT --inlist fLIst.txt --out avgZ500.nc --varname GEOPOTENTIAL--avgname AZ500 --hpa --is4D --gh`

2. Calculate Z500 anomalies from the LTDM using list `fList.txt`:
   `$USER_DIR/tempestextremes/bin/BlockingDevs --inlist fList.txt --avg avgZ500.nc --avgname AZ500 --varname GEOPOTENTIAL --z500 --hpa --gh --is4D`

3. Calculate normalized anomalies

2.3 BlockingGHG

```
Usage: BlockingGHG <parameter list>
Parameters:
  --in <string> [""] **
  --inlist <string>  [""] **
  --varname <string> ["Z"]
  --tname <string> ["time"] +
  --latname <string> ["lat"] +
  --lonname <string> ["lon"] +
  --insuff <string> [".nc"] +
  --outsuff <string> ["_GHG.nc"] +
  --gh +
  --hpa +
  --is4D +
```

`--in <string>`
input file in NetCDF format, with axes `[time, latitude, longitude]`.

`--inlist <string>`
list of input datafiles (one file per line) in NetCDF format, with axes `[time, latitude, longitude]`.

`--varname <string>`
name of Z500 variable (default `Z`).

** Do not use both `--in` and `--inlist` flags simultaneously.

2.3.1 File input and output

This program takes each input file containing the Z500 variable (specified by the `--varname` flag if name is not default `Z`) and outputs a file (original file name plus output suffix), containing the variable `ZG`, which has a value of either 1 (gridpoint is instantaneously blocked) or 0 (not blocked). Use the additional boolean flags (explained in Section 2.10) as necessary.

2.4 Anomalies, Step 1 (S04): Input files: BlockingPV

```
Usage: BlockingPV <parameter list>
Parameters:
  --in <string> [""] **
  --in2d <string> [""]
  --inlist <string>  [""] **
  --inlist2d <string> [""]
  --hasPV
  --ipl +
  --hpa +
  --PVname <string> ["PV"] +
  --VPVname <string> ["VPV"] +
  --uname <string> ["U"] +
  --vname <string> ["V"] +
  --tempname <string> ["T"] +
  --tname <string> ["time"] +
  --levname <string> ["lev"] +
  --latname <string> ["lat"] +
  --lonname <string> ["lon"] +
  --insuff <string> [".nc"] +
  --outsuff <string> ["_integ.nc"] +
  --insuff2d <string> [".nc"] +
  --outsuff2d <string> ["_ipl.nc"] +
```

`--in <string>`
Input file in NetCDF format containing temperature and wind variables, with axes `[time, lev, lat, lon]`

`--in2d <string>`
Optional input file in NetCDF format containing surface pressure, with axes `[time, lat, lon]`. Only necessary if input file's vertical axis is on model levels.

`--inlist <string>`
A list of input datafiles in NetCDF format containing temperature and wind variables, with axes `[time, lev, lat, lon]`

`--inlist2d <string>`
Optional list of input datafiles in NetCDF format containing surface pressure, with axes `[time, lat, lon]`. Only necessary if input files' vertical axis is on model levels.

`--hasPV`
Optional bool that tells program that PV variable is already present, and only vertically averaged PV needs to be calculated.

** Do not use both `--in` and `--inlist` flags simultaneously.

2.4.1 File input and output

This program produces vertically averaged (150-500 hPa) potential vorticity for use in S04. Without the `--hasPV flag`, this program takes input files containing temperature (`T`) and zonal (`U`) and meridional (`V`) winds and outputs files (original file name plus suffix `_integ.nc`) with both potential vorticity (`PV`) and vertically averaged potential vorticity (`VPV`).

If potential vorticity variable is already present in the input file, use the `--hasPV` flag (with optional `--PVname` flag if name of variable differs from the default) and the program will output new files (original file name plus suffix `_integ.nc`) containing the vertically averaged `IPV` variable.

Input files should have a vertical axis on pressure levels. If the vertical axis is on model levels, use the `--ipl` flag, as well as input surface pressure files (`--in2d` or `--inlist2d`) to first interpolate variables to pressure levels. If the pressure levels are in units of hPa (or mb), use the `--hpa` flag to inform the program that the units need to be converted.

2.5 Anomalies, Step 2: Long Term Daily Mean: BlockingDFT

```
Usage: BlockingDFT <parameter list>
Parameters:
  --inlist <string>  [""] *
  --out <string>  [""] *
  --varname<string>  [""] *
  --avgname <string>  [""] *
  --ncoef <integer>  [12]
  --is4D +
  --gh +
  --hpa +
  --tname <string> ["time"] +
  --levname <string> ["lev"] +
  --latname <string> ["lat"] +
  --lonname <string> ["lon"] +
```

`--inlist <string>`
A list of input datafiles in NetCDF format containing variables that will be averaged, with axes `[time, lat, lon]` (if axes are `[time, lev, lat, lon]` then use boolean flag `--is4D`)

`--out <string>`
Name of output file

`--varname <string>`
Name of input variable

`--avgname <string>`
Name of output average variable

`--ncoef <integer>`
Number of coefficients used in Fourier transform (default is 12)

2.5.1 File input and output

This program takes a list of files and computes the mean field for each of the 365 days in a year (excluding leap days). The values for these days are Fourier transformed and then back transformed using the specified number of coefficients, producing a long term daily mean which has been smoothed.

The program outputs a file that contains the averaged variable with axes `[time, lat, lon]`. The time axis has 365 time steps, each representing a day in the year. The time axis has units of "days since 0001-01-01".

2.6 Anomalies, Step 3: Unsmoothed and Smoothed Anomalies: BlockingDevs

```
Usage: BlockingDevs <parameter list>
Parameters:
  --in <string> [""] **
  --inlist <string>  [""] **
  --out <string> [""] ***
  --avg <string> [""] *
  --avgname <string> [""] *
  --varname <string> [""] *
  --pv
  --z500
  --is4D +
  --gh +
  --hpa +
  --tname <string> ["time"] +
  --levname <string> ["lev"] +
  --latname <string> ["lat"] +
  --lonname <string> ["lon"] +
```

`--in <string>`
Name of input datafile in NetCDF format.

`--inlist <string>`
A list of input datafiles in NetCDF format.

`--avg <string>`
The name of the long term daily average file

`--avgname <string>`
Name of input long term daily average variable.

`--varname <string>`
Name of input variable for which anomaly will be calculated.

`--pv` or `--z500`
Boolean which specifies whether the vertically averaged potential anomaly (`--pv`) or 500 mb geopotential height (`--z500`) anomaly will be calculated. If neither is specified, the names for the unsmoothed (`--devname`) and smoothed (`--adevname`) anomalies are required.

** Do not use both `--in` and `--inlist` flags simultaneously.

*** Only use `--out` in conjunction with the `--in` flag.

2.6.1 File input and output

This program takes a list of files and calculates anomalies from the long term daily mean (the file which was calculated in Section 2.5). If `--z500` is not included, then the anomaly is simply $V^* = V - \overline{V}$. The boolean flag `--z500` tells the program to multiply the Z500 anomalies by a weight that is proportional to the sine of latitude (this eliminates bias in energy propagation due to the convergence of meridians at the higher latitudes).

For each input file, the program produces a corresponding output file containing the calculated anomalies with the suffix (default `_devs`) appended to the original file name. There are 2 output variables: the unsmoothed anomaly (`DVPV/DZ/other`) and the anomaly with 2-day smoothing (`ADVPV/ADZ/other`).

2.7 Anomalies, Step 4a: Long Term Daily Mean of Anomalies: BlockingDFT

The procedure is identical to that of Section 2.5, except the list of input files will be a list of the anomaly files that were calculated in the previous step (Section 2.6). It is recommended to use the smoothed anomalies.

Note that this step is optional if you wish to use a constant threshold value.

2.8 Anomalies, Step 4b: Spatiotemporally Varying Threshold: BlockingThresh

```
Usage: BlockingThresh <parameter list>
Parameters:
  --inlist <string>  [""] *
  --outfile <string>  [""] *
  --davgfile <string>  [""] *
  --davgname <string>  [""] *
  --dvarname <string> [""] *
  --tname <string> ["time"] +
  --levname <string> ["lev"] +
  --latname <string> ["lat"] +
  --lonname <string> ["lon"] +
```

`--inlist <string>`
A list of input datafiles in NetCDF format containing variables that will be averaged, with axes `[time, lat, lon]` (if axes are `[time, lev, lat, lon]` then use boolean flag `--is4D`)

`--outfile <string>`
Name of output file.

`--davgfile <string>`
Name of time-average anomalies file (as calculated in Section 2.7)

`--davgname <string>`
Name of time-average anomaly variable

`--dvarname <string>`
Name of the anomaly variable (preferably the smoothed anomaly variable)

2.8.1 File input and output

This program calculates a threshold field (with variable name `THRESHOLD_DFT`) that varies in both time and space. The threshold is defined as 1.5 times the standard deviation of the anomaly values for a given day over the course of a range of years, and is calculated at each grid point. The threshold is smoothed across the latitude, longitude, and time axes using the same Fourier transform method that was utilized in Section 2.5.

The program outputs a file that contains the threshold variable with axes `[time, lat, lon]`. The time axis has 365 time steps, each representing a day in the year. The time axis has units of "days since 0001-01-01".

Note that this step is optional if you wish to use a constant threshold value.

2.9 Anomalies, Step 6: Normalized Anomalies: BlockingNormDevs

```
Usage: BlockingNormDevs <parameter list>
Parameters:
  --in <string> [""] **
  --inlist <string>  [""] **
  --out <string> [""] ***
  --varname <string> [""] *
  --thresh <string> [""] ***
  --threshname <string> [""] ***
  --normname <string> [""]
  --pv
  --z500
  --const ***
  --threshval <double> [0.]
  --minthreshval <double> [0.]
  --tname <string> ["time"] +
  --levname <string> ["lev"] +
  --latname <string> ["lat"] +
  --lonname <string> ["lon"] +
```

`--in <string>`
Name of input datafile in NetCDF format.

`--inlist <string>`
A list of input datafiles in NetCDF format.

`--out <string>`
Name of a single output file (only use with `--in` flag).

`--varname <string>`
Name of input anomaly variable which will be normalized.

`--pv` or `--z500`
Boolean which specifies whether the vertically averaged potential anomaly (`--pv`) or 500 mb geopotential height (`--z500`) anomaly will be used. If neither is specified, a generic anomaly normalization is performed.

`--thresh <string>` Name of threshold file calculated in Section 2.8 (if using spatiotemporally varying threshold)

** Do not use both `--in` and `--inlist` flags simultaneously.

*** Either specify a threshold file with a spatiotemporally varying threshold (`--thresh`) or use the `--const` boolean to specify a constant threshold.

2.9.1 File input and output

This program calculates anomalies that are normalized by the specified threshold value. The `--pv` flag tells the program to search for negative anomalies in the Northern Hemisphere and positive anomalies in the Southern Hemisphere (minimum magnitude 1.1 PVU for varying threshold, or a constant 1.2 PVU threshold). The `--z500` flag tells the program to search for positive anomalies (minimum magnitude 100 m for varying threshold, or a constant 170 m threshold). If neither `--pv` nor `--z500` is specified,

2.10 Optional flags

Some of the flags, such as `--latname`, are optional; if not provided, they will default to the value specified in the code. These flags are not all available for each program, so check the documentation to see which ones can be utilized (you can also run the desired program without any flags, which will print out the command list, raise an exception, and terminate).

2.10.1 Variable and axis names

`--tname <string>`
Name of the time axis (default `time`).

`--levname <string>`
Name of the vertical level axis (default `lev`).

`--latname <string>`
Name of the latitude axis (default `lat`).

`--lonname <string>`
Name of the longitude axis (default `lon`).

`--PVname <string>`
Name of the potential vorticity variable (default `PV`).

`--VPVname <string>`
Name of the output vertically averaged potential vorticity variable (default `VPV`)

`--uname <string>`
Name of the zonal wind variable (default `U`).

`--vname <string>`
Name of the meridional wind variable (default `V`).

`--tempname <string>`
Name of the temperature variable (default `T`).

2.10.2 File naming conventions

`--insuff <string>`
Input file extension string (default `.nc`).

`--insuff2d <string>`
Input surface pressure file extension string (default `.nc`).

`--outsuff <string>`
Output file naming convention (default varies between programs).

`--outsuffipl<string>`
Output interpolated variable file naming convention (default varies between programs).

2.10.3 Booleans

`--gh`
Boolean that tells the program to convert geopotential to geopotential height ($Z = \frac{\Phi}{g}$).

`--hpa`
Boolean that tells the program that the pressure axis is in hPa instead of Pa.

`--ipl`
Optional bool directing program to create new input file with vertical pressure axis (using input file with vertical model level axis and surface pressure file)

`--is4D`
Boolean that tells the program that the input variable has 4 dimensions (time, vertical level, latitude, longitude).

3 DetectCyclonesUnstructured

```
Usage: DetectCyclonesUnstructured <parameter list>
Parameters:
  --in_data <string> [""]
  --in_data_list <string> [""]
  --in_connect <string> [""]
  --out <string> [""]
  --out_file_list <string> [""]
  --searchbymin <string> [""] (default PSL)
  --searchbymax <string> [""]
  --minlon <double> [0.000000] (degrees)
  --maxlon <double> [0.000000] (degrees)
  --minlat <double> [0.000000] (degrees)
  --maxlat <double> [0.000000] (degrees)
  --topofile <string> [""]
  --maxtopoht <double> [0.000000] (m)
  --mergedist <double> [0.000000] (degrees)
  --closedcontourcmd <string> [""] [var,delta,dist,minmaxdist;...]
  --noclosedcontourcmd <string> [""] [var,delta,dist,minmaxdist;...]
  --thresholdcmd <string> [""] [var,op,value,dist;...]
  --outputcmd <string> [""] [var,op,dist;...]
  --timestride <integer> [1]
  --regional <bool> [false]
```

```
--out_header <bool> [false]
--verbosity <integer> [0]
```

`--in_data <string>`
A list of input datafiles in NetCDF format, separated by semi-colons.

`--in_data_list <string>`
A file containing the `--in_data` argument for a sequence of processing operations (one per line).

`--in_connect <string>`
A connectivity file, which uses a vertex list to describe the graph structure of the input grid. This parameter is not required if the data is on a latitude-longitude grid.

`--out <string>`
The output file containing the filtered list of candidates in plain text format.

`--out_file_list <string>`
A file containing the `--out` argument for a sequence of processing operations (one per line).

`--searchbymin <string>`
The input variable to use for initially selecting candidate points (defined as local minima). By default this is "PSL", representing detection of surface pressure minima. Only one of `searchbymin` and `searchbymax` may be set.

`--searchbymax <string>`
The input variable to use for initially selecting candidate points (defined as local maxima). Only one of `searchbymin` and `searchbymax` may be set.

`--minlon <double>`
The minimum longitude for candidate points.

`--maxlon <double>`
The maximum longitude for candidate points.

`--minlat <double>`
The minimum latitude for candidate points.

`--maxlat <double>`
The maximum latitude for candidate points.

`--mergedist <double>`
Merge candidate points with distance (in degrees) shorter than the specified value. Among two candidates within the merge distance, only the candidate with lowest `searchbymin` or highest `searchbymax` value will be retained.

`--closedcontourcmd <cmd1>;<cmd2>;...` Eliminate candidates if they do not have a closed contour. Closed contour commands are separated by a semi-colon. Each closed contour command takes the form `var,delta,dist,minmaxdist`. These arguments are as follows.

> `var <variable>` The variable used for the contour search.
>
> `dist <double>` The great-circle distance (in degrees) from the pivot within which the closed contour criteria must be satisfied.
>
> `delta <double>` The amount by which the field must change from the pivot value. If positive (negative) the field must increase (decrease) by this value along the contour.
>
> `minmaxdist <double>` The distance away from the candidate to search for the minima/maxima. If `delta` is positive (negative), the pivot is a local minimum (maximum).

`--noclosedcontourcmd <cmd1>;<cmd2>;...`
As `closedcontourcmd`, except eliminates candidates if a closed contour is present.

`--thresholdcmd <cmd1>;<cmd2>;...` Eliminate candidates that do not satisfy a threshold criteria (there must exist a point within a given distance of the candidate that satisfies a given equality or inequality). Threshold commands are separated by a semi-colon. Each threshold command takes the form `var,op,value,dist`. These arguments are as follows.

`var <variable>` The variable used for the contour search.

`op <string>` Operator that must be satisfied for threshold (options include `>`, `>=`, `<`, `<=`, `=`, `!=`).

`value <double>` The value on the RHS of the comparison.

`dist <double>` The great-circle-distance away from the candidate to search for a point that satisfies the threshold (in degrees).

`--outputcmd <cmd1>;<cmd2>;...` Include additional columns in the output file. Output commands take the form `var,op,dist`. These arguments are as follows.

`var <variable>` The variable used for the contour search.

`op <string>` Operator that is applied over all points within the specified distance of the candidate (options include `max`, `min`, `avg`, `maxdist`, `mindist`).

`dist <double>` The great-circle-distance away from the candidate wherein the operator is applied (in degrees).

`--timestride <integer>`
Only examine discrete times at the given stride (by default 1).

`--regional`
When a latitude-longitude grid is employed, do not assume longitudinal boundaries to be periodic.

`--out_header`
Output a header describing the columns of the data file.

`--verbosity <integer>`
Set the verbosity level (default 0).

3.1 Variable Specification

Quantities of type `<variable>` include both NetCDF variables in the input file (for example, "Z850") and simple operations performed on those variables. By default it is assumed that NetCDF variables are specified in the `.nc` file as

`float Z850(time, lat, lon)` or `float Z850(time, ncol)`

for structured latitude-longitude grids and unstructured grids, respectively. If variables have no time variable, they have the related specification

`float Z850(lat, lon)` or `float Z850(ncol)`

If variables include an additional dimension, for instance,

`float Z(time, lev, lat, lon)` or `float Z(time, lev, ncol)`

they may be specified on the command-line as `Z(<lev>)`, where the integer index `<lev>` corresponds to the first dimension (or the dimension after `time`, if present).

Simple operators are also supported, including

_ABS(<variable>) Absolute value of a variable,

_AVG(<variable>, <variable>) Pointwise average of variables,

_DIFF(<variable>, <variable>) Pointwise difference of variables,

_F() Coriolis parameter,

_MEAN(<variable>, <distance>) Spatial mean over a given radius,

_PLUS(<variable>, <variable>) Pointwise sum of variables,

_VECMAG(<variable>, <variable>) 2-component vector magnitude.

For instance, the following are valid examples of <variable> type,

_MEAN(PSL,2.0), _VECMAG(U850, V850) and _DIFF(U(3),U(5)).

3.2 MPI Support

The DetectCyclonesUnstructured executable supports parallelization via MPI when the --in_data_list argument is specified. When enabled, the parallelization procedure simply distributes the processing operations evenly among available MPI threads.

4 StitchNodes

```
Usage: StitchNodes <parameter list>
Parameters:
  --in <string> [""]
  --out <string> [""]
  --format <string> ["no,i,j,lon,lat"]
  --range <double> [5.000000] (degrees)
  --minlength <integer> [3]
  --min_endpoint_dist <double> [0.000000] (degrees)
  --min_path_dist <double> [0.000000] (degrees)
  --maxgap <integer> [0]
  --threshold <string> [""] [col,op,value,count;...]
  --timestride <integer> [1]
  --out_format <string> ["std"] (std|visit)
```

--in <string>
The input file (a list of candidates from DetectCyclonesUnstructured).

--out <string>
The output file containing the filtered list of candidates in plain text format.

--format <string>
The structure of the columns of the input file.

--range <double>
The maximum distance between candidates along a path.

`--minlength <integer>`
The minimum length of a path (in terms of number of discrete times).

`--min_endpoint_dist <double>`
The minimum great-circle distance between the first candidate on a path and the last candidate (in degrees).

`--min_path_dist <double>`
The minimum path length, defined as the sum of all great-circle distances between candidate nodes (in degrees).

`--maxgap <integer>`
The largest gap (missing candidate nodes) along the path (in discrete time points).

`--threshold <cmd1>;<cmd2>;...`
Eliminate paths that do not satisfy a threshold criteria (a specified number of candidates along path must satisfy an equality or inequality). Threshold commands are separated by a semi-colon. Each threshold command takes the form `col,op,value,count`. These arguments are as follows.

> `col <integer>` The column in the input file to use in the threshold criteria.
>
> `op <string>` Operator used for comparison of column value (options include `>`, `>=`, `<`, `<=`, `=`, `!=`).
>
> `value <double>` The value on the right-hand-side of the operator.
>
> `count <integer>` The minimum number of candidates along the path that must satisfy this criteria.

`--timestride <integer>`
Only examine discrete times at the given stride (by default 1).

Index