BlobMetrics: an analysis framework for StitchBlobs outputs

Contents

1 Minimum software and data requirements

- R software (`https://www.r-project.org/`) and the following libraries:
  - `abind`
  - `argparse`
  - `RNetCDF` (`https://journal.r-project.org/archive/2013/RJ-2013-023/RJ-2013-023.pdf`)
- StitchBlobs output (in the form of NetCDF files)
- BlobStats output (in the form of text files)

If the required libraries have not yet been installed in R, then an internet connection will be necessary, as R will attempt to download the missing libraries from online repositories before running the code.

2 Usage

2.1 Command line usage

The main control framework is run from the command line, with the following syntax:

```
Rscript --vanilla stitch_metric_framework.R [flags]
```

In order to view all possible options, run the above command with the `-h` or `--help` flag:

```
Rscript --vanilla stitch_metric_framework.R -h
```

which will print out the list of available flags and exit. These flags will be explained further in subsequent sections.

2.1.1 Namelists

Each utility must be used in conjunction with a namelist file, which will provide all of the necessary variables that are not specified in the command line. Example namelists are included in the directory with the R function source files.

The `-nl` or `--namelist` flag specifies the master namelist (example `namelist_master.R`), which contains the necessary variables for all of the framework utilities. However, each utility can be provided with a separate namelist file if desired (flags specified in subsequent sections), which will override inputs from the master namelist.

For example,

```
Rscript --vanilla stitch_metric_framework.R  --namelist namelist_master.R --readfiles --mergetable
--summarize --namelistst namelist_summarize.R
```

will use `namelist_master.R` for the `readfiles` and `mergetable` utilities but `namelist_summarize.R` for `--summarize`

Each utility has certain mandatory variables. The `nrun_*` variable specifies how many times the utility is to be run; if more than 1, then the variables must be specified as vectors, which have the syntax `c(VAR1,VAR2,VAR3...)`. For example, to read in data from 4 different datasets using `--readfiles`:

```
#Location of working directory
work_dir<-"~/tempestextremes/test/STITCH_METRICS"
#Location where the input data files are stored
input_dir<-"~/tempestextremes/test/STITCH_METRICS"
#Location where the output data files are stored
output_dir<-"~/tempestextremes/test/STITCH_METRICS"

#This will be run 4 times
nrun_rf<-4
###################################
###USER-DEFINED###
#Input file names
list_files<-c("stitch_list","nostitch_list","merra_list","merra_nolist")
#output file names
list_rnames<-c("table_estitch.RData","table_enostitch.RData",
"table_mstitch.RData","table_mnostitch.RData")
#Data frame variable names
list_dfnames<-c("df_estitch","df_enostitch","df_mstitch","df_mnostitch")
###################################

nhrs<-6
varname<-c("ERA","ERA","MERRA","MERRA")
filename_stitchblobs<-""
filelist_stitchblobs<-paste(input_dir,list_files,sep="/")
rfn_stitch<-paste(output_dir,list_rnames,sep="/")
df_stitchname<-list_dfnames
txt_stitch<-""
csv_stitch<-""
```

Note that some variables, such as `nhrs` and `txt_stitch`, have only one input rather than a vector of 4 values. That value will be used for all runs.

The `paste` function appends the appropriate directory to the vectors of input and output file names.

2.2 Interactive R Usage

These various tools can also be utilized in an interactive R session by opening R and loading the various functions from the source R scripts. For example, to read a list of BlobStats files into a single `data.frame`, type the following commands into R:

```
source("read_stitch.R")
file_list<-readLines("list_of_blobfiles.txt")
table_stats<-read_stats_to_table(file_list,6,var="TM90")
```

The `source` command reads in all of the commands from the R script `read_stitch.R`, which loads the function `read_stats_to_table`. This function takes an existing text file (`list_of_blobfiles.txt`) containing a list of BlobStats file names, loads the file names into a vector of strings, and reads all of those files into a data frame that is named `table_stats`. `6` refers to the number of hours per time step in the original data and `TM90` refers to the block detection algorithm (Tibaldi and Molteni 1990) that was used to produce input files for StitchBlobs.

3 Read BlobStats files into a single table

This utility takes each BlobStats file and reads the information into a single combined data frame. The columns of the data frame depend upon the included variables in the BlobStats output. Possible variables include `minlat, minlon, maxlat, maxlon, centlat, centlon,` and `area` and are specified when running BlobStats. There is optional functionality to save this output to one of three file types (RData, text, or CSV).

3.1 Requirements

Text files containing output from BlobStats. The format of these files is explained in more detail in Appendix A.

3.2 Command line syntax

```
Rscript --vanilla stitch_metric_framework.R [-rf] [-nl or -nlrf FILE]
```

The following flags are required:

`-rf` (`--readfiles`)
Tell program to read in BlobStats data

`-nl` (`--namelist`) or `-nlrf` (`--namelistrf`) `FILE`
Name of namelist file

3.2.1 Required namelist variables

`nrun_rf`
Number of times to run `--readfiles`

`nhrs`
Number of hours per time step in data files (i.e. 6 is 6 hourly)

`filename_stitchblobs` or `filelist_stitchblobs`
Single file containing BlobStats data or a text file containing a list of file names. Please use one or the other— set the unused variable to a blank string (`""`)

`rfn_stitch, txt_stitch, csv_stitch`
Optional output file names in RData, text, or CSV format (store the respective variable as a blank string to suppress output for that particular file format)

`df_stitchname`
Optional variable name for the output data frame in the RData file. Defaults to `df_tot` if string is left blank.

3.3 Function syntax

To load and use this function in an interactive R session, do

```
source("readfiles.R")
desired_name<-read_stats_to_table(flist,nhrs,..)
```

which will produce a `data.frame` object with the variable name `desired_name`.

The following arguments are required:

`flist`
a vector object containing the BlobStats file names.

`nhrs`
Time resolution in terms of hours (i.e. 6 for 6 hourly)

The following arguments are optional:

`var`
Name of the objective blocking detection algorithm used to produce input files for StitchBlobs. If left blank, a column `var` will be filled with the string `VAR`

`rfn, textfn, csvfn`
Strings specifying output file names for RData, text, and CSV file formats. If left blank, the function will merely return the `data.frame` object to the console.

3.4 Output

An example output `data.frame` looks like this in the R console:

```
             datehour minlat maxlat minlon maxlon centlat centlon    area  area_km bnum var
1 1980-12-01 00:00:00     50     72    187    218    61.0   202.5 0.08299 42330251    1 TM90
2 1980-12-01 06:00:00     50     74    187    222    62.0   204.5 0.08980 45803790    1 TM90

                           file
1   ERA_1980_DJF_NP_Z_stats.txt
2   ERA_1980_DJF_NP_Z_stats.txt
```

`datehour`
The date string in the format `YYYY-MM-DD HH:MM:SS`

`minlat, maxlat, minlon, maxlon, centlat, centlon`
Latitude and longitude coordinates for the block's extent and centroid

`area`
Fractional area of the block

`area_km`
area of the block in km^2

`var`
Algorithm name specified either by the `--algname` flag in the console or `var` in the function (default `VAR`)

`bnum`
The blob ID number as specified in the BlobStats file

`file`
name of the BlobStats file which contains the specified blob information

4 Read previously created data table(s) into an R session

This utility reads output generated in Section 3 into the R session and produces a single output `data.frame`. This function is particularly useful if attempting to examine data from multiple detection algorithms. There is optional functionality to save the output to one of three file types (RData, text, or CSV).

4.1 Requirements

Data tables produced using the procedure outlined in Section 3. The tables can be in RData, text, or CSV file format.

4.2 Command line syntax

```
Rscript --vanilla stitch_metric_framework.R [-rt] [-nl or -nlrt FILE]
```

The following flags are required:

`-rt` (`--readtable`)
Tell program to read in BlobStats data

`-nl` (`--namelist`) or `-nlrt` (`--namelistrt`) `FILE`
Name of namelist file

4.2.1 Required namelist variables

`nrun_rt`
Number of times to run `--readtable`

`ftype_rt`
Input file type (`"R"`, `"text"`, or `"CSV"`)

`filename_read` or `filelist_read`
Single file or list of files containing `--readfiles` output

`rfn_combine, txt_combine, csv_combine`
Optional output file names in RData, text, or CSV format (store the respective variable as a blank string to suppress output for that particular file format)

`df_combinename`
Optional variable name for the output data frame in the RData file. Defaults to `df_data` if string is left blank.

4.3 Function syntax

To load this function in an interactive R session, do

```
source("readtable.R")
desired_name<-combine_dfs(flist,ftype,...)
```

which will combine the loaded `data.frame` objects from each file into a single `data.frame` object with the variable name `desired_name`.

The following arguments are required:

`flist`
a vector object containing the BlobStats file names.

`ftype`
File format of input files (specify one of three strings: `"R"`, `"text"`, or `"CSV"`)

The following arguments are optional:

`rfn, textfn, csvfn`
Strings specifying output file names for RData, text, and CSV file formats. If left blank, the function will merely return the `data.frame` object to the console.

4.4 Output

The output is identical to that seen in Section 3.4, but there might be multiple values for the `var` column if combining tables with data from different algorithms.

5 Handling instances of merging/splitting blobs

There are some instances where multiple blobs will merge into a single blob at a later date, or a single blob will split off into multiple blobs (this was noted in Sinclair 1995). This can cause BlobStats to produce latitude/longitude blob extents which are much larger than those of each individual blob, and the centroid coordinate will subsequently "jump" a noticeable distance from one time step to the next.

The DetectBlobs binary in TempestExtremes will produce output that is very similar to StitchBlobs output, but provides latitude/longitude extents for each unique feature; blobs which split off from larger features have their own separate identifier.

For example, here is output from StitchBlobs for one blob over 24 hours:

```
           datehour minlat maxlat minlon maxlon centlat centlon    area  area_km var bnum
1980-12-01 00:00:00     50     72    187    218    61.0   202.5 0.08299 42330251   Z    1
1980-12-01 06:00:00     50     74    187    222    62.0   204.5 0.08980 45803790   Z    1
1980-12-01 12:00:00     45     75    139    226    60.0   182.5 0.12303 62753232   Z    1
1980-12-01 18:00:00     43     75    138    231    59.0   184.5 0.13999 71403925   Z    1
```

Here is corresponding output from DetectBlobs:

```
           datehour minlat maxlat minlon maxlon centlat centlon    area  area_km var bnum
1980-12-01 00:00:00     50     72    187    218    61.0   202.5 0.08299 42330251   Z    1
1980-12-01 06:00:00     50     74    187    222    62.0   204.5 0.08980 45803790   Z    2
1980-12-01 12:00:00     50     75    187    226    62.5   206.5 0.09634 49139611   Z    3
1980-12-01 12:00:00     45     55    139    155    50.0   147.0 0.02670 13618721   Z    4
1980-12-01 18:00:00     49     75    186    231    62.0   208.5 0.10133 51684833   Z    5
1980-12-01 18:00:00     43     57    138    157    50.0   147.5 0.03866 19719092   Z    6
```

Note that at times 12Z and 18Z, the detected blob in the StitchBlobs dataset (`bnum 1`) is actually comprised of two blobs, because the smaller blob (`bnum 4` and `6` in the DetectBlobs dataset) is separate from the larger blob (`3` and `5` in the DetectBlobs output) at these time steps, but the smaller blob merges into the larger blob at a later time.

While these merged blobs only made up a small subset of instances in our own dataset, we recognize that this data might skew results with respect to distribution of block size or centroid coordinate; therefore, we provide this extra functionality to distinguish between the individual blobs within the larger detected region. The summarization utility (Section 6), which provides information on each unique block's size, speed, etc will note any instances in which there is blob merging. The user can then choose to keep or omit blobs in which there was merging.

5.1 Requirements

Separate files or file lists for StitchBlobs and DetectBlobs data. If using the interactive R session, two separate data tables must first be produced using the method outlined in Section 3 or Section 4).

5.2 Command line syntax

```
Rscript --vanilla stitch_metric_framework.R [-mt] [-nl or -nlmt FILE]
```

The following flags are required:

`-mt` (`--mergetable`)
Tell program to merge the data from StitchBlobs and DetectBlobs

`-nl` (`--namelist`) or `-nlmt` (`--namelistmt`) `FILE`
Name of namelist file

5.2.1 Required namelist variables

`nrun_mt`
Number of times to run `--mergetable`

`ftype_mt`
Input file type (`"R"`, `"text"`, or `"CSV"`)

`stitch_file` or `stitch_list`
Single file or list of files containing data from StitchBlobs (generated using either `--readfiles` or `--readtable`)

`detect_file` or `detct_list`
Single file or list of files containing data from DetectBlobs (generated using either `--readfiles` or `--readtable`)

`rfn_merged, txt_merged, csv_merged`
Optional output file names in RData, text, or CSV format (store the respective variable as a blank string to suppress output for that particular file format)

`df_merged`
Optional variable name for the output data frame in the RData file. Defaults to `df_merged` if string is left blank.

5.3 Function syntax

To load this function in an interactive R session, do

```
source("mergetable.R")
desired_name<-merge_dfs(df_stitch,df_nostitch,...)
```

which will produce a `data.frame` object with the variable name `desired_name`. Note that `df_stitch` and `df_nostitch` will need to be created using either `read_table` or `combine_tables`.

The following arguments are required:

`df_stitch`
a data frame (created using `read_table` or `combine_tables`) containing BlobStats output with StitchBlobs data

`df_nostitch`
a data frame (created using `read_table` or `combine_tables`) containing BlobStats output with DetectBlobs data

The following arguments are optional:

`rfn, textfn, csvfn`
Strings specifying output file names for RData, text, and CSV file formats. If left blank, the function will merely return the `data.frame` object to the console.

5.4 Output

The output `data.frame` looks similar to one returned by the first two methods, with the exception of an additional `bnum2` variable. When `bnum=bnum2`, the latitude/longitude extent is encompassing a single blob.

```
           datehour minlat maxlat minlon maxlon centlat centlon    area   area_km var bnum bnum2
1980-03-06 00:00:00     34     42    191    206    38.0   198.5 0.02822  14394019   Z    1     1
```

When the two do not match, there are multiple blobs contained within the latitude/longitude extent of the original StitchBlobs output.

```
           datehour minlat maxlat minlon maxlon centlat centlon    area   area_km var bnum bnum2
1980-03-08 18:00:00     39     45    136    156    42.0   146.0 0.02500  12751612   Z    1    13
1980-03-08 18:00:00     32     51    194    222    41.5   208.0 0.07581  38667988   Z    1    12
```

6 Create a per-blob summary table

This utility reads in a data frame with per-timestep information and creates a table that provides per-blob information on quantities such a the blob's starting and ending centroid coordinates, the blob's duration in days, and others described in more detail below. There is optional functionality to save the output to one of three file types (RData, text, or CSV). If desired, blobs which are comprised of multiple blobs that merge into a single blob are omitted.

6.1 Requirements

6.2 Command line syntax

```
Rscript --vanilla stitch_metric_framework.R [-st] [-nl or -nlst FILE]
```

The following flags are required:

`-st` (`--summarize`)
Tell program to summarize each unique blob's data

`-nl` (`--namelist`) or `-nlst` (`--namelistst`) `FILE`
Name of namelist file

6.2.1 Required namelist variables

`nrun_st`
Number of times to run `--summarize`

`ftype_st`
Input file type (`"R"`, `"text"`, or `"CSV"`)

`filename_summ` or `filelist_summ`
Single file or list of files containing data from StitchBlobs (generated using either `--readfiles` or `--readtable`)

`keep_merge`
Keep or omit blobs which are comprised of multiple blobs? If `TRUE`, all blobs are kept; if `FALSE`, blobs where the `merged` column has a value of `YES` are omitted from the output.

`rfn_summ, txt_summ, csv_summ`
Optional output file names in RData, text, or CSV format (store the respective variable as a blank string to suppress output for that particular file format)

`df_summ`
Optional variable name for the output data frame in the RData file. Defaults to `df_summ` if string is left blank.

6.3 Function syntax

To load this function in an interactive R session, do

```
source("summarize.R")
desired_name<-gen_summary_table(df_in,...)
```

which will summarize each unique blob in the input `data.frame` object and produce an output `data.frame` object with the variable name `desired_name`. Note that `df_in` will need to be created using either `read_table` or `combine_tables`.

The following arguments are required:

`df_in`
Name of the input data frame (create using methods from Section 3 or 4).

The following arguments are optional:

`keep_merge`
Default is `TRUE`; if set to `FALSE`, the final data table will not contain merged blobs.

`rfn, textfn, csvfn`
Strings specifying output file names for RData, text, and CSV file formats. If left blank, the function will merely return the `data.frame` object to the console.

6.4 Output

An example summary table looks like this in the R console:

```
             startdate             enddate duration_days merged start_centlat
1  1980-03-06 00:00:00 1980-03-14 12:00:00          8.50    YES          38.0
2  1980-03-17 06:00:00 1980-03-26 00:00:00          8.75     NO          39.5
```

```
   start_centlon end_centlat end_centlon   dist_km zonal_dist_km
1          198.5        43.5       194.0  719.2533      379.0267
2          204.0        39.0       208.5  391.4136      387.4485

   zonal_speed_kph min_area  max_area avg_area var bnum
1        1.8579740 12751612  75295717 32349835   Z    1
2        1.8449929 15500859  35505588 32349835   Z    2
```

`startdate, enddate`
The date string in the format `YYYY-MM-DD HH:MM:SS`

`duration_days`
Number of days that block persists

`merged`
Checks whether or not the block extent is the result of multiple blobs merging (see Section 5). If this value is `YES`, then it is recommended to check the per-timestep information in order to see when and where the blobs merge, as well as how it affects the calculation of the block size and centroid.

`start_centlat, start_centlon, end_centlat, end_centlon`
start and end coordinates of block centroid.

`dist_km`
Great circle distance from start to end coordinates.

`zonal_dist_km`
Only the zonal component of the distance between the start and end coordinates (calculated using the start and end longitude coordinates of the centroid and the midpoint of the start and end latitude coordinates).

`zonal_speed_kph`
Average speed, in km/hr, of the block's movement. Calculated as distance over duration.

`min_area, max_area, avg_area`
Various information for the block size, in km^2

`var`
Algorithm name (as provided previously when reading in the BlobStats data)

`bnum`
The block ID number from BlobStats (might differ from the ID number of DetectBlobs output)

7 Reading in NetCDF data

This utility reads data from NetCDF files into arrays in R, combining data from multiple files if desired. There is an optional utility to read in only a geographical subset of the data, as well as the ability to save the specified variables to either a single NetCDF output file or an Rdata file.

7.1 Requirements

NetCDF files with the desired variables. Note that if reading multiple files into a single session, they should all contain the variables that are specified in the command line or function call, otherwise the utility will throw an exception.

If the dataset is very large, it might cause R to crash due to memory constraints (although this is dependent upon the available system memory— this scenario is more likely to happen on personal computers). It is recommended to split the data up by time or regional subsets if this is likely to be a problem.

7.2 Command line syntax

```
Rscript --vanilla stitch_metric_framework.R [-rn] [-nl or -nlrn FILE]
```

The following flags are required:

`-rn (--readnetcdf)`
Tell program to read NetCDF data into R.

`-nl` (`--namelist`) or `-nlrn` (`--namelistrn`) `FILE`
Name of namelist file

7.2.1 Required namelist variables

`nrun_rn`
Number of times to run `--readnetcdf`

`filename_netcdf` or `filelist_netcdf` Single NetCDF file name or list of NetCDF file names

`varvec`
Vector of variable names to be read to R in the format `c("VAR1","VAR2","VAR3"...)`

`outvec`
Vector of output variable names (must be same length as `varvec`

`outrdata, outnetcdf`
Optional output file names in RData or NetCDF format

`timename`
Name of time axis

`levname`
Name of vertical axis

`latname`
Name of latitude axis

`lonname`
Name of longitude axis

`minlat, maxlat, minlon, maxlon`
subsetting boundaries for horizontal (lat/lon) direction; if not subsetting, set variable to blank string (`""`)

`minlev,maxlev`
If applicable, subsetting boundaries for the vertical axis. If subsetting to a single level (i.e. 500 mb) set both `minlev` and `maxlev` to that value. If vertical axis doesn't exist, set variable to blank string (`""`)

7.3 Function syntax

To load this function in an interactive R session, do

```
source("read_netCDF_to_R.R")
vlist<-c("VAR1","VAR2","VAR3")
vars_output<-read_netcdf(flist,vlist,...)
```

The folllowing arguments are required:

`flist`
Vector containing input NetCDF file names

`vlist`
Vector containing variable names that will be read into R

which will read all of the NetCDF variables into R matrices and return a `list` object with the name `vars_output` containing all of the resulting output variables, as well as the axes for time, latitude, longitude, etc.

The following arguments are optional:

`olist`
Vector containing variable names of output variables (default is the input variable list)

`timename, levname, latname, lonname`
Names of the axis variables (defaults are `"time", "lev", "lat", "lon"`)

`minlat, maxlat, minlon, maxlon, minlev, maxlev`
Coordinates of lat/lon extent and vertical levels, if subsetting.

`ncout, rdataout`
Strings specifying output file names for NetCDF and RData file formats. If left blank, the function will return a `list` object that contains all of the variables.

7.4 Output

If neither `ncout` nor `rdataout` are specified, the function returns a `list` object (here named `vars_output` as per the example).

Each variable stored within `vars_output` can be accessed via `vars_output[["VAR"]]`. In order to turn the `vars_output` object into distinct R variables that are accessible from the global environment, do the following:

```
list_variables<-names(vars_output)
for (v in list_variables){
    assign(v,vars_output[[v]])
}
remove(vars_output)
```

`list_variables` is a vector of names of all of the variables that are contained within `vars_output`. This sequence of commands will store each variable in `vars_output` in the global environment, then delete the `list` object (for space saving purposes).

7.4.1 NetCDF

The returned NetCDF will contain the variables specified by the user, with axes `time_axis, lat_axis, lon_axis` and, optionally, `lev_axis` (not all variables are on multiple vertical levels). Currently, the output NetCDF does not contain the metadata from the original file, although this might change in the future.

`time_axis` is in units of `hours since 1800-01-01 00:00` regardless of the original time units.

7.4.2 RData

The RData file contains the axis variables (`time_axis, lev_axis, lat_axis, lon_axis`), a vector of string date times (`time_format`), and the variables specified within the variable list.

A BlobStats File Format

Each BlobStats file is formatted as follows:

Line 1: Date of first time step in format `YYYY-MM-DD`

Line 2: Tab-separated column names

Blob information line: `Blob IDNUM (NUM_TIMESTEPS)` where `IDNUM` is the blob's unique identifier number and `NUM_TIMESTEPS` is the number of timesteps in the blob's lifespan.

Per-timestep blob information: Always contains the timestep number in column 1. The other columns depend on the included variables.

For example, a BlobStats file with two Blobs, each with a lifetime of 2 time steps, would look like this:

```
1980-12-01
Time    minlat  maxlat  minlon  maxlon  centlat centlon area
Blob 1 (2)
1       50.00000        74.00000        187.00000       222.00000       62.00000        204.50000
2       45.00000        75.00000        139.00000       226.00000       60.00000        182.50000
Blob 2 (2)
53      39.00000        48.00000        226.00000       253.00000       43.50000        239.50000
54      36.00000        49.00000        221.00000       254.00000       42.50000        237.50000
```