CURSAT v2.1: A simple, resampling-based program to generate pseudoreplicates of data and calculate rarefaction curves.

Email: gabriele.gentile@uniroma2.it

README

Description:

CURSAT ver. 2.1 is an open-source code in QB64 basic, compilable into an executable file, that produces n pseudoreplicates of an empirical data set. Both resampling with and without replacement are allowed by the software. The number (n) of pseudoreplicates is set by the user. Pseudoreplicates can be exported in a file that can be opened by a spreadsheet. Thus, pseudoreplicates are permanently stored and available for calculation of statistics of interest and associated variance. The software also uses the n pseudoreplicate data to reconstruct n accumulation matrices, appended in an output file. Accumulation has applicability in cases in which repeated sample-based data must be evaluated for exhaustiveness. Many situations involve repeated sampling from the same set of observations. For example, if data consist in species occurrence, the software can be used by a wide spectrum of specialists such as ecologists, zoologists, botanists, biogeographers, conservationists for biodiversity estimation. The software allows to perform accumulation irrespectively whether the input data set contains abundance (quantitative) or incidence (binary) data. Accumulation matrices can be imported in statistical packages to estimate distributions of successive pooling of samples and depict accumulation and rarefaction curves with associated variance.

CURSAT ver. 2.1 is released in two editions. Edition #1 is recommended for analysis, whereas Edition #2 generates a log file in which the flow of internal steps of resampling and accumulation routines is reported. Edition #2 is primarily designed for educational purposes and quality check.

Download and installation of QB64:

CURSAT ver. 2.1 is written in basic (QB64). Consequently, the source code needs to be loaded in QB64 and then compiled. QB64 is not a cross-platform software *per se*. However, the QB64 programming language as released at https://www.qb64.org/ is provided with stable builds in Microsoft Windows 32bit and 64bit, Linux, and MacOS. Instructions about how to install QB64 on Windows, Linux, and MacOS can be found at the website:
https://www.qb64.org/wiki/QB64_FAQ#Q:_How_do_I_install_QB64_on_Windows.2C_Linux.2C_macOS.3F
of which I report here a synthetic, re-elaborated, extract :

WINDOWS NT(XP), WINDOWS VISTA, WINDOWS 7, 8 OR 10:

1) Click the following link to download QB64: https://www.qb64.org/

2) Unzip to any folder path you wish. The ZIP file will create a QB64 folder for the program files.

You may need Administrator rights to install or use QB64.

MOST VERSIONS OF LINUX 32 AND 64 BIT

1) Click on the following link to download QB64 for Linux: https://www.qb64.org/

2) Make sure you have installed the following: OpenGL development libraries, ALSA development libraries, GNU C++ Compiler (g++)

3) After extracting the downloaded package, run the installation batch/script called setup_lnx.sh in the main qb64 folder to setup QB64.

4) Do not install QB64 as root.

If you have problems running the install scripts under Linux (./setup_lnx.sh), run the following line in terminal, from your QB64 folder: find . -name '*.sh' -exec sed -i "s/\r//g" {} \;

MacOS

1) You must install Xcode command line tools for C++ compilation from their website. In a terminal window, type the following command: xcode-select --install (more info here: Xcode download)

(you won't be using the Xcode interface, QB64 just needs to have access to the C++ compiler it installs)

2) Click on the following link and MAC OSX box to download QB64 for macOS: QB64 MAC OSX Downloads

Extract the downloaded package and run setup_osx.command, found within the QB64 folder to install the QB64 compiler.

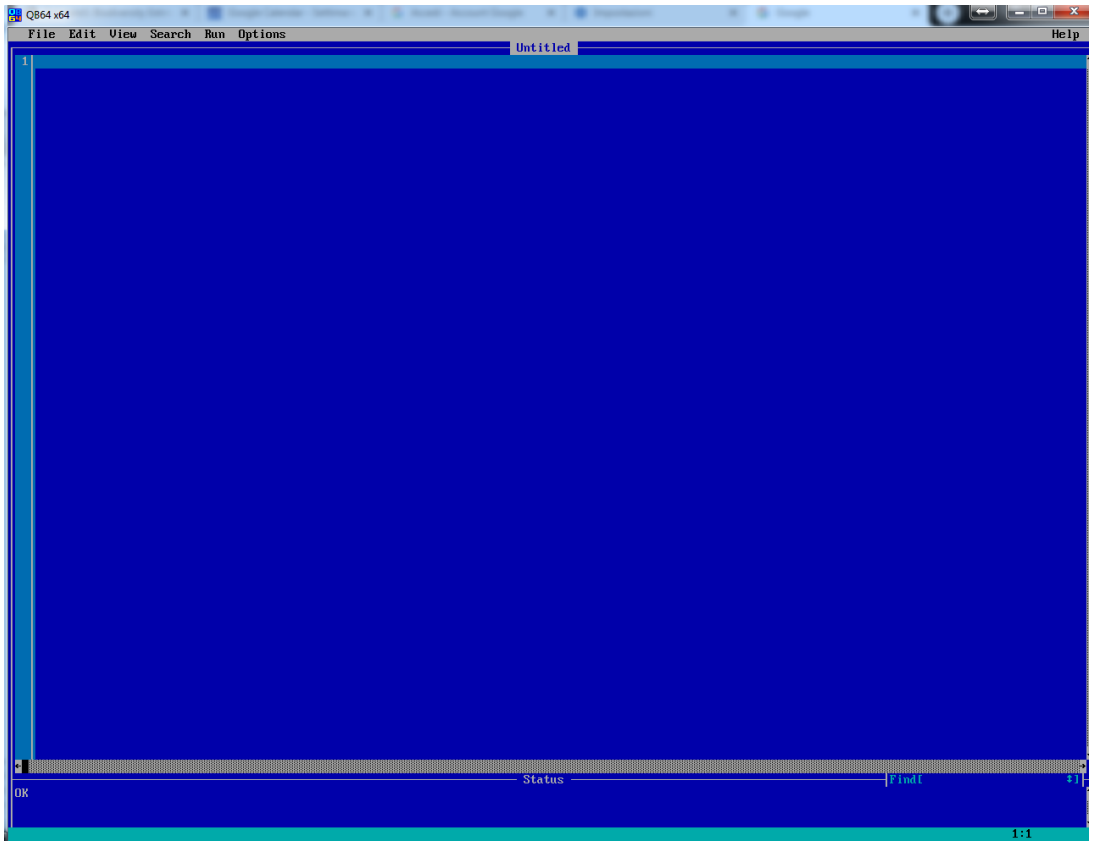
3) After installation you should run qb64_start_osx.command to run qb64.

To help launch executables without a console, a file called *programname_start.command* is created along with the program.

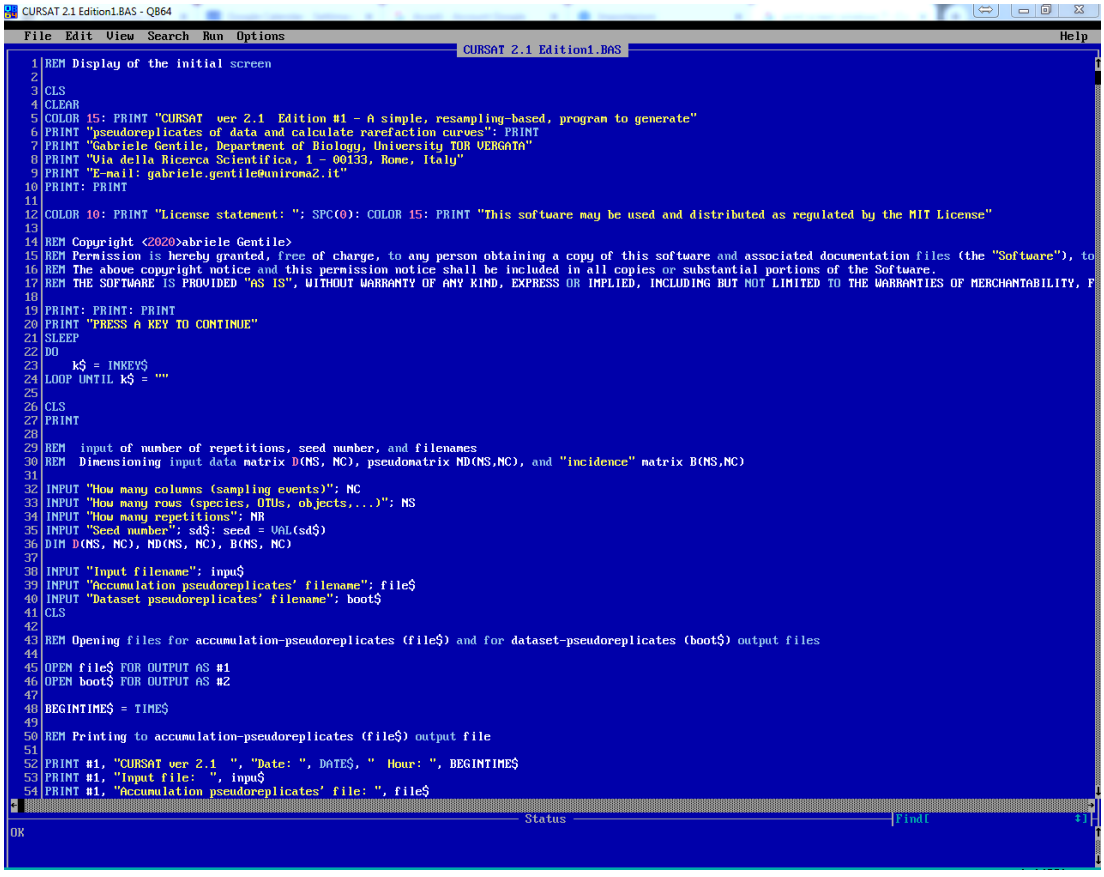
If you have problems running the install scripts under macOS (./setup_osx.command), run the following line in terminal, from your QB64 folder: find . -name '*.command' -exec perl -pi -e 's/\r\n|\n|\r/\n/g' {} \;

Don't forget you need to have Xcode command line utilites installed to use QB64.

Once QB64 has be installed, you can run it and a screen like this will appear:



Then you select “Open” from the “File” menu. Subsequently you select the source code you want to load (in the following example it is the file “CURSAT 2.1 Edition1.bas”).



From the menu “Run” you can select “Start” or “Make EXE Only”. In the first case QB64 will compile the source code and start the executable. In the second case, the executable will be generated but it will not start. An executable can be run as a “stand alone” software and QB64 is no longer needed.

Download and installation of CURSAT v.2.1:

CURSAT v.2.1 software can be downloaded from the following URL:

https://github.com/gabrio62/CURSAT-ver.2.1

Once downloaded, unzip the package in a folder at your convenience. The source code (file with extension .bas) of CURSAT ver.2.1 can be loaded in QB64 environment and be compiled into an executable file that runs under the specific Operating Systems. Make sure you indicate the correct path when loading the source code in QB64 for compiling. Alternatively, and much easier, you can copy the source code in the folder where QB64 has been previously installed, load it in QB64 without indicating any path, and compile it. Once the executable file has been generated, you can copy it in any folder you wish. For your convenience, I suggest to copy the executable in your data folder, so that you will not have to write long paths when digitizing filenames. Two edition of CURSAT v.2.1 are provided. The package provides the source of both editions #1 and #2, and executable file of both editions running on Windows. Double click over the executable (*.exe) file will start the program in Windows. Executable files running on MacOS and Linux have to be generated by first loading the source code in QB64 and subsequent compilation.

Input format:

The structure of a data file is very simple, as it consists in a matrix in which species (OTUs, or objects) are rows and sampling events are columns. The input file is simply structured, with numbers being delimited by either spaces (blanks) or commas. **The use of other separators (tabs, dash, etc.) may cause error messages or generate inconsistent results**.

Examples:

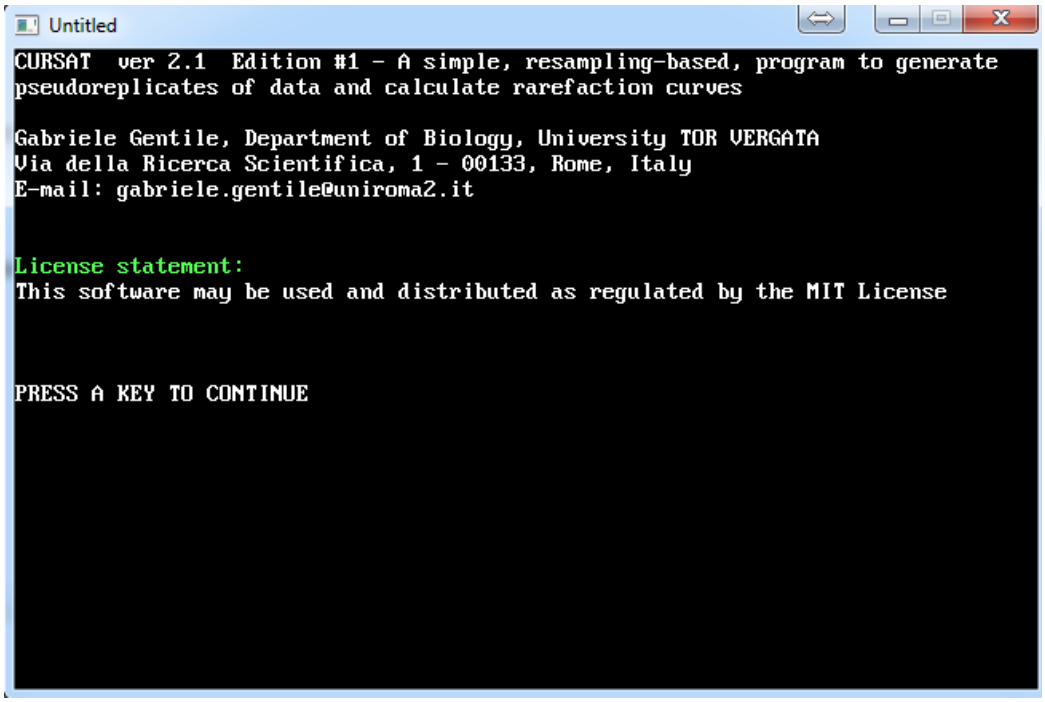
The software is provided along with example data and output files. The input file *incidence.txt* is a 12 x 10 rectangular binary matrix of incidence, derived from the abundance matrix in the file *abundance.txt*. The file *seedbank.txt* is a 34 x 121 abundance data matrix, as provided in the package EstimateS v.9.1.0 (Colwell 2014).

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
a	1	0	1	1	0	0	0	1	1	0
b	1	1	0	0	0	0	1	1	0	0
c	0	0	0	1	0	1	1	0	1	1
d	0	0	0	0	1	0	1	0	1	0
e	0	0	1	1	0	0	0	1	0	0
f	1	1	0	0	1	0	0	1	0	1
g	0	0	1	1	1	0	0	0	1	0
h	1	1	0	1	0	0	1	1	0	1
i	1	0	0	0	1	1	0	1	0	1
j	0	0	1	1	0	0	1	0	1	0
k	0	0	1	1	0	1	0	1	0	1
l	1	1	1	0	0	0	1	1	0	0

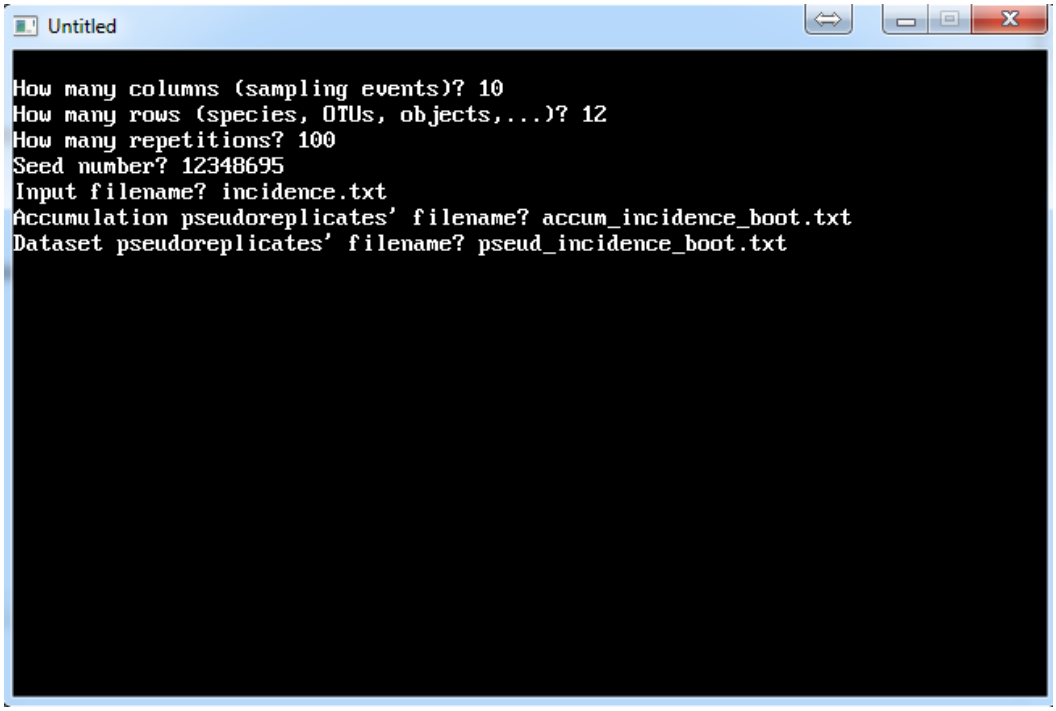
Above: Incidence matrix in file *incidence.txt*

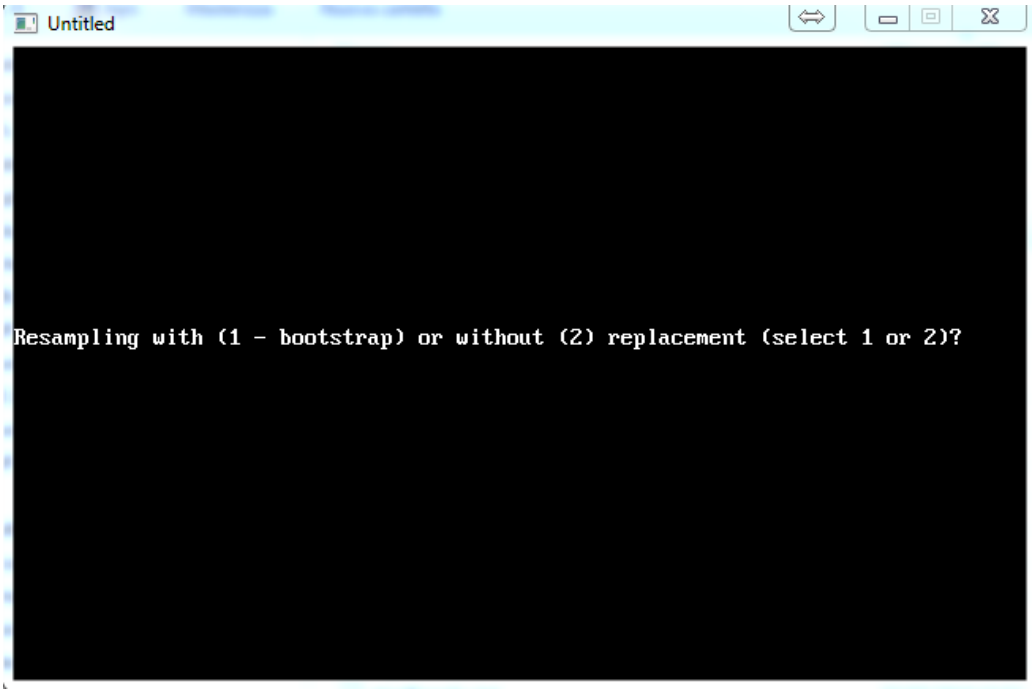
Running CURSAT v.2.1

Once compiled, Edition#1 and Edition#2 of CURSAT v.2.1 executables can be run. Running CURSAT v.2.1 is very simple. In Windows, the initial mask of the software CURSAT v.2.1 Edition#1 is:

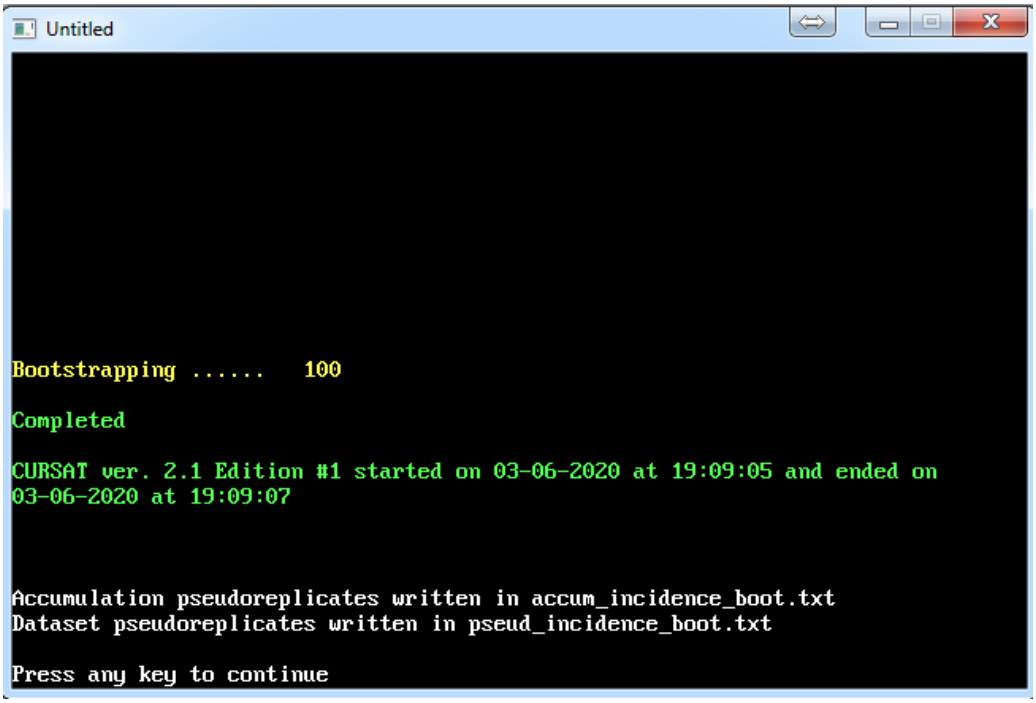


After pressing any key the software will ask for input. Here, the file *incidence.txt* is used and a bootstrap procedure is selected, as follows:

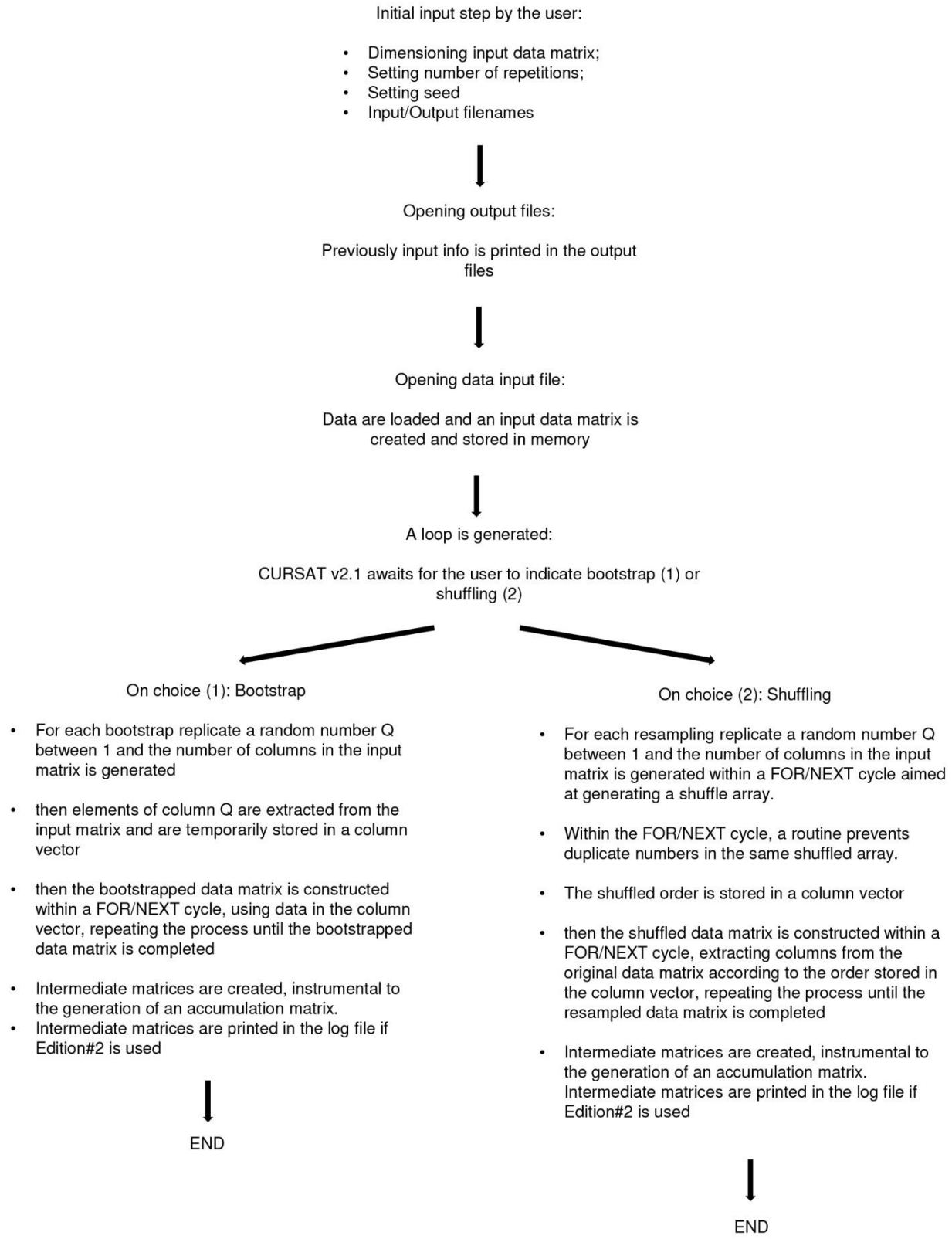




When the run has ended, the screen will be as follows:



Functional structure of CURSAT ver.2.1:



Output:

Data stored in the files *accum_incidence_boot.txt* and *pseud_incidence_boot.txt* are provided in bundle with the source code and Windows executables. In both files, the first column marks the resample replicate during which the accumulation and correspondent data pseudoreplicate were generated. The files *accum_abundance_shuf.txt* and *pseud_abundance_shuf.txt*, are also provided. They are output files obtained when the option “resampling without replacement” (shuffling) was selected. Similarly, analogous output files are also provided as resulting from analyses of data in the file *incidence.txt* and others. Runs were all performed with seed=12348695 and 100 replicates.

	A	B	C	D	E	F	G	H	I	J	K
1	CURSAT ver 2.1 Edition #1 Date: 03-05-2020 Hour: 14:58:39										
2	Input file: incidence.txt										
3	Accumulation pseudoreplicates' file: accum_incidence_boot.txt										
4	Dataset pseudoreplicates' file: pseud_incidence_boot.txt										
5	Number of columns (sampling events): 10										
6	Number of rows (species, OTUs, objects,...): 12										
7	Number of repetitions: 100										
8	Seed number: 12348695										
9											
10	Bootstrap (resampling with replacement)										
11											
12	Bootstrap replicate										
13											
14	1	0	0	1	1	0	1	1	1	0	1
15	1	1	0	0	1	0	0	0	1	1	1
16	1	1	1	1	0	1	0	1	0	1	0
17	1	1	0	0	0	0	0	1	0	1	0
18	1	0	0	1	0	0	1	0	1	0	1
19	1	0	0	0	1	0	0	0	1	0	1
20	1	0	0	1	0	0	1	1	0	0	0
21	1	1	0	1	1	0	0	0	1	1	1
22	1	0	1	0	1	1	0	0	1	0	1
23	1	1	0	1	0	0	1	1	0	1	0
24	1	0	1	1	0	1	1	0	1	0	1
25	1	1	0	0	1	0	1	0	1	1	1
26											
27	2	1	1	1	1	0	0	0	1	0	1
28	2	0	1	1	1	0	1	1	0	0	1
29	2	1	0	0	0	0	1	0	1	1	0
30	2	0	0	0	0	1	1	0	0	0	0
31	2	1	0	1	1	0	0	0	1	0	0
32	2	0	1	1	1	1	0	1	0	0	1
33	2	1	0	0	0	1	0	0	1	0	0
34	2	1	1	1	1	0	1	1	1	0	1
35	2	0	1	1	1	1	0	0	0	1	1
36	2	1	0	0	0	0	1	0	1	0	0
37	2	1	0	1	1	0	0	0	1	1	0
38	2	0	1	1	1	0	1	1	0	0	1

Above: File *pseud_incidence_boot.txt* opened in Microsoft Excel. It consists in the output file with 100 bootstrapped pseudoreplicates of the original dataset in the file incidence.txt. The first column is a label that marks the resample replicate during which the pseudoreplicate was generated. Only the first two replicates are shown.

	A	B	C	D	E
1	CURSAT ver 2.1 Edition #1 Date: 03-05-2020 Hour: 14:58:39				
2	Input file: incidence.txt				
3	Accumulation pseudoreplicates' file: accum_incidence_boot.txt				
4	Dataset pseudoreplicates' file: pseud_incidence_boot.txt				
5	Number of columns (sampling events): 10				
6	Number of rows (species, OTUs, ojects,...): 12				
7	Number of repetitions: 100				
8	Seed number: 12348695				
9					
10	Bootstrap (resampling with replacement)				
11					
12	Bootstrap replicate	Sampling event	Accumulation per n. of events		
13	1	1	6		
14	1	2	8		
15	1	3	11		
16	1	4	12		
17	1	5	12		
18	1	6	12		
19	1	7	12		
20	1	8	12		
21	1	9	12		
22	1	10	12		
23	2	1	7		
24	2	2	11		
25	2	3	11		
26	2	4	11		
27	2	5	12		
28	2	6	12		
29	2	7	12		
30	2	8	12		
31	2	9	12		
32	2	10	12		

Above: Accumulation data file *accum_incidence_boot.txt* opened in Microsoft Excel. It consists in the output file with 100 accumulation replicates based on the bootstrap of original dataset in the file *incidence.txt*. The first column marks the resample replicate during which the accumulation was generated. In the second and third columns sampling events and associated cumulative number of objects are respectively reported. Only the first two replicates are shown.

Creation of a Log file:

The software is provided in two editions. Edition #1 is recommended for analysis purposes. Edition #2 adds a log file (named by the user) where information used by the software (matrices D, ND, W, B, and F in the source code) generated and used during resampling and accumulation routines is stored and saved. Such information may prove useful for users to identify whether the software is operating as expected.

Data.txt (Input matrix D)

CURSAT ver 2.1 Edition #2 Date: 03-05-2020 Hour: 15:28:29
Input file: data.txt
Accumulation pseudoreplicates' file: accum_data_shuf.txt
Dataset pseudoreplicates' file: pseud_data_shuf.txt
Number of columns: 5
Number of rows: 4
Number of repetitions: 100
Seed number: 12348695
Log filename: data_shuf_log.txt

Resampling without replacement

4 2 1 3 5 -> Shuffling order in this replicate (vector F)

Replicate 1

ND(1 1)= 0	0 *	D(1 4)= 0
ND(2 1)= 6		D(2 4)= 6
ND(3 1)= 3		D(3 4)= 3
ND(4 1)= 1		D(4 4)= 1

Replicate 1

ND(1 2)= 0	0+0=0 **	D(1 2)= 0
ND(2 2)= 1		D(2 2)= 1
ND(3 2)= 1		D(3 2)= 1
ND(4 2)= 4		D(4 2)= 4

Replicate 1

ND(1 3)= 2	0+0+2=2 ***	D(1 1)= 2
ND(2 3)= 0		D(2 1)= 0
ND(3 3)= 1		D(3 1)= 1
ND(4 3)= 3		D(4 1)= 3

Replicate 1

ND(1 4)= 5	0+0+2+5=7 ****	D(1 3)= 5
ND(2 4)= 3		D(2 3)= 3
ND(3 4)= 0		D(3 3)= 0
ND(4 4)= 2		D(4 3)= 2

Replicate 1

ND(1 5)= 1	0+0+2+5+1=8 *****	D(1 5)= 1
ND(2 5)= 7		D(2 5)= 7
ND(3 5)= 4		D(3 5)= 4
ND(4 5)= 0		D(4 5)= 0

Replicate 1

W(1 1)= 0 *	if W > 1 then B =1	B(1 1)= 0
W(1 2)= 0 **	... idem ...	B(1 2)= 0
W(1 3)= 2 ***	... idem ...	B(1 3)= 1
W(1 4)= 7 ****	... idem ...	B(1 4)= 1
W(1 5)= 8 *****	... idem ...	B(1 5)= 1

Replicate 1

W(2 1)= 6	B(2 1)= 1
W(2 2)= 7	B(2 2)= 1
W(2 3)= 7	B(2 3)= 1
W(2 4)= 10	B(2 4)= 1
W(2 5)= 17	B(2 5)= 1

Replicate 1

W(3 1)= 3	B(3 1)= 1
W(3 2)= 4	B(3 2)= 1
W(3 3)= 5	B(3 3)= 1
W(3 4)= 5	B(3 4)= 1
W(3 5)= 9	B(3 5)= 1

.....omissis

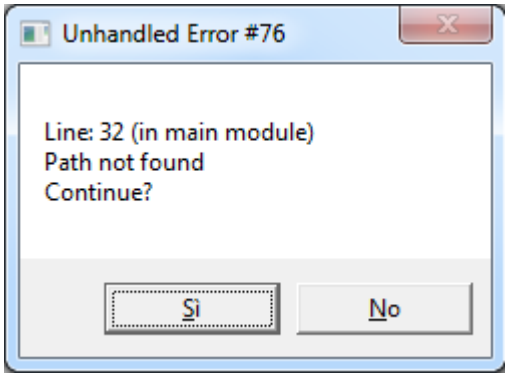
Above: An example of log file generated by running Edition #2, illustrating in detail the algorithm used by CURSAT ver. 2.1. The input data matrix D (top-left) in the file *data.txt* was used in this example. Only outcomes of replicate n.1 from a shuffling procedure are shown because the procedure is very similar for

bootstrap. For each resampling replicate, a pseudoreplicate data matrix (ND) is generated by extracting the elements from a column of the input matrix D, according to the shuffling order stored in vector F. In this example, elements in column #1 of ND are extracted from column #4 of the input matrix D (green); subsequently, elements in column #2 of ND are extracted from column #2 of the input matrix D (orange), and so on. A new matrix (W) is then constructed by cumulatively summing elements of the pseudoreplicate data ND matrix by row (see asterisks). Finally, the matrix B (same dimensions as W) is constructed from matrix W, by replacing elements > 0 with 1. This is instrumental to correctly perform accumulation irrespectively whether the input file contains abundance (as in this case) or incidence data. The accumulation replicates are constructed by cumulatively summing elements of matrix B by column (see files *data_shuf_log.txt* and *accum_data_shuf.txt*). The accumulation replicates are not stored in memory, but they are printed in the output file meanwhile they are created.

Errors and troubleshooting:

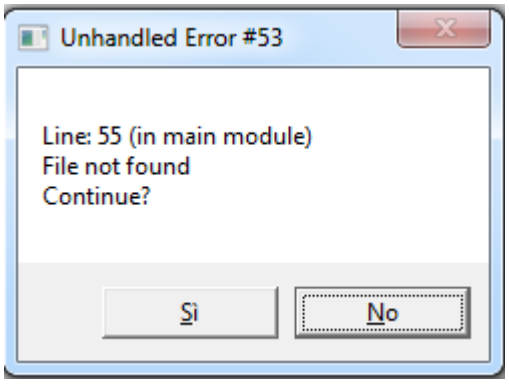
CURSAT v.2.1 is rather easy to use. However, you may still run into some problems and receive error messages.

Usually, this depends on the fact that you made some mistake when digitizing input info or path names. To avoid this kind of errors, it is recommended to copy the executable AND your data files in the same directory. If you don't do it, it's easy to make mistakes and receive the following kind of messages:



If this happens, you have to terminate the program and start it over.

Other messages are rather obvious, such as:



In general, error messages generally provide helpful information about the type of error occurred. However, you must be aware that they refer to a line that does not correspond to the same line in the source code. A more comprehensive list of different kind or error codes can be found at: https://qb64.org/wiki/ERROR_Codes . No official support is provided, but users are welcome to contact the author for troubleshooting.

License and term of use:

This software can be used as per the MIT License here below.