Indecis
Sectorial Climate Services

Integrated approach for the **d**evelopment across **E**urope of user oriented **c**limate **i**ndicators for GFCS high-priority **s**ectors: Agriculture, disaster risk reduction, energy, health, water and tourism

Work Package 3

Deliverable 3.1.a

INDECIS Quality Control Software and Manual: INQC, beta version

E. Aguilar[1]

[1] *Centre for Climate Change (C3), Rovira i Virgili University (URV), Vila-seca - Spain*

ERA4CS European Research Area for Climate Services

JPI Climate

This report arises from the Project INDECIS which is part of ERA4CS, an ERA-NET initiated by JPI Climate, and funded by FORMAS (SE), DLR (DE), BMWFW (AT), IFD (DK), MINECO (ES), ANR (FR), with co-funding by the European Union’s Horizon 2020 research and innovation programme

TABLE OF CONTENTS

1. Authorship and licensing

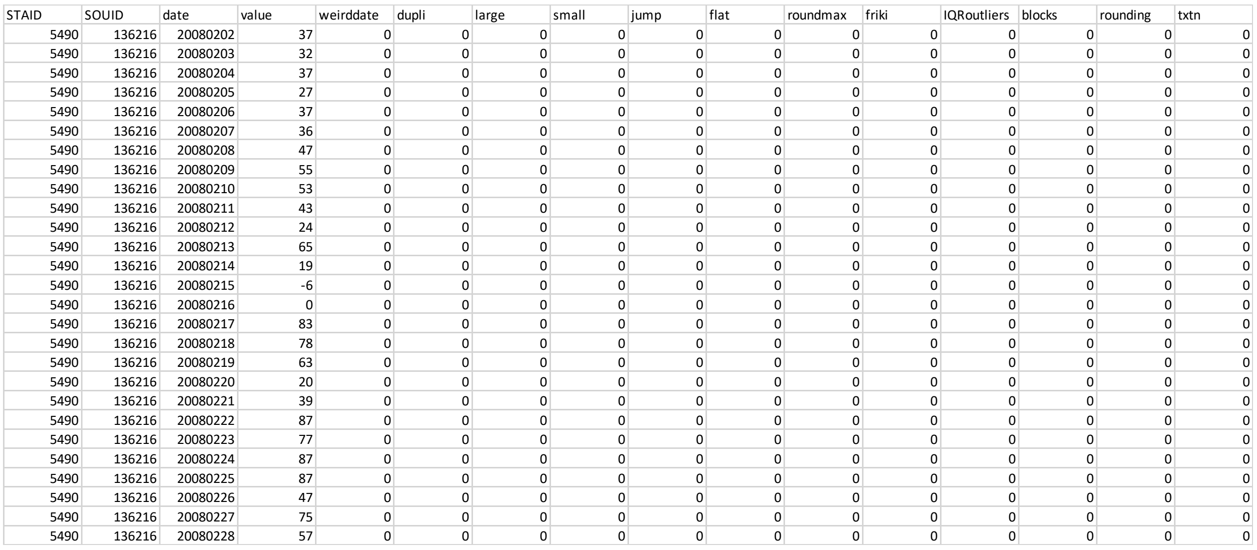
2. An overview

This document complements the documents D3.1a and D3.1c for the completion of the INDECIS' deliverable 3.1 INDECIS Quality Control Software and Manual. Here we present the software INDECIS QC, beta version (inqc_beta.R), created by Enric Aguilar, Center for Climate Change, C3, Universitat Rovira i Virgili, Tarragona (Spain), and licenced under the terms expressed in Section 1.
The software will be made available at : https://github.com/INDECIS-Project/INQC

Contact person: enric.aguilar@urv.cat

INDECIS QC (INQC, from now onwards) is designed to quality control European Climate Assessment and Dataset (ECA&D) daily data of maximum, minimum and average temperature, precipitation, sea level pressure, relative humidity, wind speed, snow depth, cloud coverage and sunshine duration.

INQC works applying a series of tests to the data. The result of each test (see Figure 1) is either *0 (pass)* or *1 (does not pass)*. At this point (beta version) no decision tool is provided, so users need to filter out those values which, according to the tests failed and their particular purpose, should not be considered for further climatological analyses.



STAID	SOUID	date	value	weirddate	dupli	large	small	jump	flat	roundmax	friki	IQRoutliers	blocks	rounding	txtn
5490	136216	20080202	37	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080203	32	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080204	37	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080205	27	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080206	37	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080207	36	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080208	47	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080209	55	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080210	53	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080211	43	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080212	24	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080213	65	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080214	19	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080215	-6	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080216	0	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080217	83	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080218	78	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080219	63	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080220	20	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080221	39	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080222	87	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080223	77	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080224	87	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080225	87	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080226	47	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080227	75	0	0	0	0	0	0	0	0	0	0	0	0
5490	136216	20080228	57	0	0	0	0	0	0	0	0	0	0	0	0

Figure 1: INQC output example. Daily Maximum Temperature.Each column presents the result of one test applied to the data. The qc value is either 0 (pass) 1 (does not pass).

3. Preparing your computer to run INQC

INQC is designed to quality control ECA&D series. The requirements to run it are the following:

- R (developed and tested under RStudio Version 1.2.1069 and R version 3.3.2)
- An INQC folder :, e.g. *~/INQC*, This folder will store:
    - The INQC code, stored as : *~/INQC/inqc_beta.R*
    - ECA&D stations files (blended version), for each variable to be used, e.g. *~/INQC/ECA_blend_source_tx.txt* (these files can be downloaded from ECA&D)
- A quality control folder, named to your preference (e.g. Sweden for Swedish data): *~/Sweden*
- A raw data folder, created into your data: *~/Sweden/raw* [this folder name MUST be "raw"]. Raw data series must be non-blended ECA&D series (other formats are not supported and will not be supported, see https://www.ecad.eu//dailydata/index.php for information) and stored in this folder, with no sub-folders
- A qc'd data folder, where INQC will store the results: *~/Sweden/QC* [this folder name MUST be "QC", capital letters]

NOTE: "~" stands for "any path before"

4. Wrapper Functions and Jump-Start option

After successfully completing the steps described in Section 2, INQC can be ran using the pre-set up defaults (see Table 1 in section 4 for full description) :

- Open R; set working directory to ~/INQC
- ***inqc(homfolder = .~/INQC)***: Quality controlling all variables (maximum, minimum and average temperature, precipitation, sea level pressure, relative humidity, wind speed, snow depth, cloud coverage and sunshine duration)
- Quality controlling ONE variable:
    - *temperature(home=**.~/INQC/**,element='TX')*: daily maximum temperature
    - *temperature(home=**.~/INQC/**,element='TN'):* daily minimum temperature
    - *temperature(home=**.~/INQC/**,element='TG')*: daily average temperature
    - *precip(home=**.~/INQC/**)*: daily accumulated precipitation
    - *relhum(home=**.~/INQC/**)*: relative humidity
    - *selepe(home=**.~/INQC/**)*: sea level pressure
    - *snowdepth(home=**.~/INQC/**)*: snow depth
    - *sundur(home=**.~/INQC/**)*: sunshine duration
    - *windspeed(home=**.~/INQC**):* windspeed

5. Quality Control Tests

Table 1: INQC Tests Description

Test	Description and objective	Parameters
badfriki	isolates extreme values which are not continuous in the distribution. If the gap is larger than a pre-set big margin, the value is flagged.	***date***: a vector of dates, in ECA&D YYYYMMDD format ***value***: the corresponding vector of values ***margina***: the maximum allowed difference between contiguous values in the empirical distribution call example: ***badfriki(date,value,margina=80)***, this call would flagg values for which the difference with the preceeding value in the empirical distribution is larger than 8°C (expressed as 80 1/10ths of degree). For example, if the second largest value is 28°C (280) and the largest is 37°C (370), the later would be flagged as an outlying value
computecal	produces a calendar with 3 variables: year, month, day between two given years.	***fy***: first year ***ly***: last year call example: ***computecal(fy=1900,ly=2018)***, would return a year,month,day dataframe with dates between 1900 and 2018
drywetlong	detects episodes of too many consecutive wet or dry days. Uses a peak over threshold approach and a pareto distribution fit over	***x***: values ***ret***: pseudo return period for the POT-pareto, computed using the ***parteogadget*** auxiliary functions call example:

	the observed sequences	***drywetlong(x,ret=300)***, this would flag those sequences with longer length that to the 300 y pseudo-return period of with a pot-pareto approach, i.e. will flag "too long" dry or wet sequences.
duplas	detects duplicated dates	***x***: a vector of dates in ECA&D format call example: ***duplas(x),*** would flag any date appearing more than once
flat	detects consecutive equal values. Can be adapted to detect consecutive equal decimal part of the values	***y***: a data vector ***maxseq***: the maximum number of contiguous repetitions of a value (e.g., if 3, sequences of 4 will be flagged) call example: ***flat(y,maxseq=3)***, this would flag any streak of 4 or more consecutive values.
IQRoutliers	computes outliers centralized around a day, using a number of days around it and based on the Inter Quartile Range. Creates a tolerance interval centred around each day of the year, using all the present values in the empirical distribution for the designed window. Values outside the interval, are flagged as outliers	***date***: a vector of dates, in ECA&D YYYYMMDD format ***value***: the corresponding vector of values ***level***: number of IQR to be added to percentile 75 and substracted to percentile 25 to determinate the tolerance interval. Values outside this interval, will be declared as outliers, ***window***: an odd number representing the length of the window for which the outliers will be computed. Note: uses auxiliary function ***julian***. call example:

		IQRoutliers(date,value,level=3,window=11), would flag outliers in value using a window of 11 days (e.g. for July 6th: July, 1st to July 11th)
jumps	to label interdiurnal differences considered to large	***x***: vector of values ***maxjump***: maximum difference allowed call example: ***jumps(x,maxjump=150)*** would flag all consecutive days for which the difference is 150 (e.g. 15°C for temperature, expressed as 150 1/10th s of degree)
paretogadget	Returns the positions exceeding the value corresponding to a return period based on pareto distro and peak over threshold approach	***x***: values ***ret***: pseudo-return period for the pot-pareto distribution approach. Uses ***potpareto*** and ***returnpareto*** call example: ***paretogadget(x,ret=300)***, this would flag all values exceeding the value corresponding to the pot-pareto pseudo return period of 300 years
physics	given a data vector, will compare the values to a specified threshold, considered to be the limit of physically possible values. In some cases. In some cases, the limitation is a consideration (e.g. 60°C), in others, it comes from the nature of the variable (e.g. 0 mm)	***x***: vector of values ***nyu***: comparison threshold, expressed in the same units of the ECA&D variable (e.g. in 1/10 of degree for temperature) ***compare*: logical operation for the comparison of the vector of values to the threshold: 1 larger; 2 larger equal; 3 smaller; 4 smaller equal; 5 equal call example: ***physiscs(x,nyu=0,compare=3)*** would flag all values smaller than 0
potpareto	Fits a pareto distribution to a series	***y***: values

	of values using as "threshold" the value representing a given quantile of the empirical distribution	***thres***: quantile to compute the threshold call example: ***potpareto(y,thres=0.99)***, would fit a pareto distro using the quantile 0.99 of the y vector
putjulian	Adds julian calendar numbers, from 1 to 366	***x***: a dataframe with year, month, day, value call example: ***putjulian(x)***, will return a data frame with year, month, day, julian, value
repeatedvalue	This function tracks values which repeat too many times and, given the typical decaying distribution of the variable (designed for precipitation) are considered too large to repeat that many times	***x***: vector of values ***margin***: the difference in frequency the nearest value ***friki***: the minimum value to be considered call example: ***repeatedvalue(x,margin=20,friki=150)*** would flag any value larger than 15 mm (expressed as 150 1/10th of mm) which repeats 20 times more than the previous value in the empirical distribution. For example, if 40 mm appears 25 times and the nearest value in the distribution is 38 and appears 5 times, all "40s" will be labelled.
returnpotpareto	For a given pareto distribution, returns the value representing a requested return period	***pato***: a pareto distribution fitted with ***potpareto*** ***ret***: pseudo return period ***w***: parameter to equate to return period to a temporal interval (recall the approach is not block maxima but peak over threshold. Typicall value of w to equate the return period to years is 1.65 (See Wilks (2011), Statistical Analysis for the Atmospheric Sciences) call example:

		returnpotpareto(pato,ret=300,w=1.65), would return the value associated to the return period of 300 years.
rounding	splits data by month and looks if a decimal value is repeated too many times	***y***: the vectors of values ***blocksize***: the maximum number of equal decimal values allowed in a block call example: ***rounding(y,blocksize=20)***, would flag all occurences of 20 or more values with the same decimal part in a month NOTE: monthly blocks are far from perfect, but they speed up the process, in comparison to sequential blocks. A fast way to do sequential blocks will be sought in future versions.
roundprecip	splits data by month and looks if a decimal value is repeated too many times. A requested value can be excluded	***y***: the vectors of values ***blocksize***: the maximum number of equal decimal values allowed in a block exclude: the value to be excluded, for example in precipitation 0 should not be considered call example: ***rounding(y,blocksize=20, exclude=0)***, would flag all occurences of 20 or more values with the same decimal part in a month, except for 0. NOTE: monthly blocks are far from perfect, but they speed up the process, in comparison to sequential blocks. A fast way to do sequential blocks will be sought in future
suspectacumprec	Detects values above a threshold preceded by a given number of "no precipitation days"	***datos***: a two columns vector, date and data, in ECA&D format ***limit***: the value above which the function will search

		tolerance: the number of "non precip days" before the value checked that will result in flagging that value call example: ***suspectacumprec(datos=x[,3:4],limit=2000,tolerance=10)***, will flag all the values avbove 2000 (200 mm expressed in 1/10th of mm) which are preceeded by days with no precip, either NA or 0.
toomany	Splits data by month or year and looks if a value is repeated too many times	***y***: two columns with date (in ECA&D format, YYYYMMDD) and vector of values ***blockmany***: maximum number of values tolerated in a block ***scope***: this variable controls whether the "block" are the months (1) or the years (2) ***exclude***: defaulted to NULL, if specified will exclude the value or values specified. Takes a single value (e.g. 0, which should repeat many times in precipitation series) or could take a vector, expressed in the R vectorial form, e.g. exclude = c(0,0.1). Note: As an evolution, it is intended to add the possibility of excluding a range of values (e.g., smaller than 3) call example: ***toomany(y=x[,3:4], blockmany=15,scope = 1, exclude=0)*** , this call would label any value expect for 0, repeating more than 15 times in particular month.
txtn	Compares daily maximum and daily minimum temperature and flags those values	***y***: a vector of values ***id***: the file name, which is passed on to ***closestation*** auxiliary function to identify the "equivalent" tx or tn station. This is not trivial,

	where TX is larger or equal than TN	as ECA&D does not provide "direct relations". See the auxiliary function for details ***home***: home folder (this is used to locate and open the "equivalent" tx or tn station call example: ***txtn(y,id= TX_SOUID135829,home='./Sweden')*** This call would flag all the tx values in this series which are smaller or equal to the values of the series determined to be the corresponding TN series
weirddate	Finds impossible dates (e.g. 19881420 or 19881131) or years out of the range of the range set by the first and the last records in the file	**x: vector of dates** call example: ***weirddate(x),*** would return any existing "impossible" or out of range date.

6. Parametrization of wrapper functions and personalized runs of INQC

In this section we list the tests and preset values included for each wrapper function. A call to ***inqc()*** would ran all variables with exactly these settings. For personalized settings, individual wrappers for each variable should be prepared. We provide one table for each variable.

Table 2 Default parametrization of the ***temperature()*** *function, as ran by* ***inqc()***

FUNCTION CALL	***temperature (home='../Sweden/',large=500,small=-500, maxjump=150,maxseq=3,margina=80, level=3,window=11,roundmax=10,blocksize=10,step=30, blockmanymonth=15,blockmanyyear=180, blocksizeround=20,element='TX')***	- **The *home* parameter is superseded when this is called from *inqc()*;** - **The *element* parameter is altered for with "TN" and "TG" for daily maximum and daily minimum temperature respectively**

Test	Parametrization	Variable in the qc'd file	Notes
badfriki	***margina*** = 80 (sets the ***margina*** parameter)	friki	
duplas	-	duplas	
flat	***maxseq*** = 3 (sets the ***maxseq*** parameter)	flat	
flat	***roundmax*** = 15 (sets the ***maxseq*** parameter)	roundmax	This function is ran twice. The second call studies the decimal part (e.g. 15.0, 12.0, 10.0, 8.0 …) are part of the same "flat" sequence. For this reason the parameter is

			set to a larger value, 15 as default
IQRoutliers	***level***: 3 (sets the level parameter) ***window***: 11 (sets the ***window*** parameter)	IQRoutliers	
jumps	***maxjump*** = 150 (sets the ***maxjump*** parameter)	jump	
physics	***large*** = 500 (sets the ***nyu*** parameter of the function)	large	Values above the parameter are flagged
physics	***small*** = - 500 (sets the ***nyu*** parameter of the function)	small	Values below the parameter are flagged
rounding	***blocksizeround*** = 20 (sets the ***blocksize*** parameter)	rounding	
toomany	***blockmanymonth*** = 15 (sets the ***blockmany*** parameter)	toomanymonth	Ran with ***scope***=1 (not parametrized in the temperature() function), splitting the series by month
toomany	***blockmanyyear*** = 180 (sets the ***blockmany*** parameter)	toomanyyear	Ran with ***scope***=2 (not parametrized in the temperature() function), splitting the series by month
txtn	-	txtn	Not ran for TG

weirddate	-	weirddate	

*Table 3 Default parametrization of the **precip()** function as ran by **inqc()***

FUNCTION CALL	precip(home='**~/INQC/**',large=5000,small=0,ret=500, retornoracha=1000,margin=20,friki=150,blocksizeround=20, excluido=0**,** blockmanymonth=15,blockmanyyear=180, exclude=0,limit=2000,tolerance=10**,** ,element='RR')	- **The *home* parameter is superseded when this is called from *inqc()*;**	
Test	**Parametrization**	**Variable in the qc'd file**	**Notes**
paretogadget	***ret*** = 300 (sets the ***ret*** parameter)	paretogadget	
duplas	-	duplas	
suspectacumprec	***limit*** = 2000 (sets the ***limit*** parameter) ***tolerance*** = 2000 (sets the ***tolerance*** parameter)	suspectacumprec	
repeatedvalue	***margin*** = 20 (sets the ***margin*** parameter) ***friki*** = 150 (sets the ***friki*** parameter)	repeatedvalue	
roundprecip	***blocksizeround*** = 20 (sets the bloscksize parameter) ***excluido*** = 0 (sets the excluded parameter)		
physics	***large*** = 5000 (sets the ***nyu*** parameter of the function)	large	Values above the parameter are flagged
physics	***small*** = 0 (sets the ***nyu*** parameter of the function)	small	Values below the parameter are flagged
toomany	***blockmanymonth*** = 15 (sets the ***blockmany*** parameter)	toomanymonth	Ran with ***scope***=1 (not parametrized

			in the temperature() function), splitting the series by month
toomany	***blockmanyyear*** = 180 (sets the ***blockmany*** parameter)	toomanyyear	Ran with ***scope***=2 (not parametrized in the temperature() function), splitting the series by month
weirddate	-	weirddate	

TBD: tables for the other variables.

7. Testing INQC and

INQC will be tested using Baboon Benchmark

8. Expected evolution

- Additional functions
- Results interpreter and default decisions
- More comfortable parametrization of the jump-start functions
- Addition INDECIS' website and to GitHub repository, with sample data