DATA SILSO HISTO quality control Report

Stephen Fay

June 27, 2019

1 Introduction

1.1 Github repository and project

https://github.com/dcxSt/DATA_SILSO_HISTO_search
https://github.com/users/dcxSt/projects/2?fullscreen=true

1.2 Brief History et Mise en Contexte

For centuries we have observed the sun and it's ever mysterious sunspots. The 11 year sunspot cycle has long been a subject of debate. Today we wish to have precise quantification of solar activity throughout the previous centuries. This is made possible by the sunspot series. For the past 3 to 4 hundred years people all over the Eurasian continent have been recording the number of sunspots that appear on the sun's earth facing half.

The aim of this project is to do a quality control of the data in DATA_SILSO_HISTO. Once the data is fixed and cleaned up, it will be stored on a new database - temporarily named GOOD_DATA_SILSO in a more user-friendly format to what currently exists. I will also get rid of any useless or redundant columns (such as the observers comment column - there are no comments)':). A third, temporary database will be mad to keep a closer eye on the data that still needs to be examined with more scrutiny : BAD_DATA_SILSO. This database will act as intermediary between DATA_SILSO_HISTO and GOOD_DATA_SILSO. We will effectively be storing 2 databases-worth of information in 3 databases. The original DATA_SILSO_HISTO will have the old data and will be corrected in due course. The intermediary BAD_DATA_SILSO will start as a copy of DATA_SILSO_HISTO and end up empty as the corrected data is removed from it and placed, in the new format, into GOOD_DATA_SILSO.

2 Processus de filtration / corigee du data (log) / quality control

2.1 Everything wrong with the data

First, it's important to note that note that though I am doing a quality control I do not wish to die of boardom. I will not be verifying each of the 205003 data-points by hand in the Mittheilungen journals, in any case this if I went about it this way I would probably miss most of the errors.

2.2 Annotation keys

2.2.1 What do the flags mean?

0	1	2	3	4
same as Null	suspicious	Comment in journal = ?	move to bin	slightly suspicious
5	6	7	8	9
definitely erroneous	na	probs ok, to be investigated	na	na

Table 1: Flags key

0. The default for the flag is NULL, when is estimate that the datapoint is perfect and there is nothing wrong with it, I can put it to zero 0.

1. If the data looks fishy but I'm not quite sure either what is wrong with it or how wrong it is this is flagged with a 1 - the default.

2. If in the Mitteilungen journals there is written a ? next to one of the data points, I will mark it with a 2, this means that the observer is not quite confident in his/her result.

3. A flag that signifies that this data point is definitely going into the bin

4. For data that is very dodgy but it is ambiguous as to weather or not it is correct, to determine its validity closer examination is required

5. For data that is definitely wrong, the difference between 5 and 4 is illustrated by example: if i find that a datapoint has a groups number of 30 I will mark it with a 4 and comment it, because this is suspicious, if a datapoint has a groups number over 60 or above, it will be marked with a 5 (trust me there are some in the hundreds).

2.3 Search and correct.

2.3.1 Outline

For the first week and a half or so, I spent the bulk of the time acquainting myself with the Mittheilungen journals, and with the software that is used to store and access the database. I also developed the tools in python to facilitate my access to them and to perform the tasks that I need to perform for the filtration process.

2.3.2 Log

I started this (today) on 2019.06.21 (yes, the solstice!)

- Friday June 21
  - Today no-one was in the office in the morning so I didn't have access to the Mittheilungen journals and decided to start writing this instead
  - at 10:20 I was let into my bit with all my notes and the journals and began 'searching the manuals' part of the project documented in the Github project linked
  - been spending time writing in all the pink corrections, including typos
  - started writing 'searching_the_manuals.py'
  - wrote and executed methods : def_correct_typos_for_pink() ; pink()
  - searching the manuals for all comments labeled 'uncertain' so as to figure out what is this word's range of meaning (wishing I had paid attention in German class)

- Monday June 24
  - 9.15 picking up from where I left off, I am currently scouring the manuals for any 'uncertain' data
  - 10.40 came across some duplicate data, and mysterious comments... there are some stars '*' that signify a change of instrument but nothing is written. The annoying thing about the duplicate data is that it is coming from
  - spent the morning making that duplicate finding and sorting algorithm, now I need to analyse the nature of the problem further. For each of the duplicates identify what kind it is, weather it's the same observer with the same instrument; if the duplicated data has for example the same rubrics numbers as each other ; if they record the same information (sunspot groups, sunspots, wolf number) ; if check to see if any clues are hidden in the comments of these duplicated data
  - in searching_the_manuals i wrote : find_duplicate_observers() ; find_obs_id_by_date() ; find_observer_alias ; find_duplicates_data() ; write_greater_duplicates_data_text()
  - i'm gonna go and delete some of the data so i will log everything in order to be very careful
- Tuesday June 25
  - 9.45 I have decided to start making modifications to the database, this is risky business - I don't want to have the blood of Galileo's data on my hands, in a few seconds I can destroy hours upon hours of one of my predecessors' work. Which would be a shame. This is why I am creating a new table in both the old and the new database that will serve as a rubbish bin, so that I simultaneously copy and delete some data. The data will be copied and destroyed in the same script but the coping will come *before* so if there are bugs nothing will be lost. First I will back up the databases as they are.
  - While making the rubbish bids for DATA_SILSO_HISTO if found that DATA_DEV was non-empty, it contains data which claims to be observations made by the grandfather of this series - Rudolf Wolf. Only the observations are dated January 1600 - Galileo's time, 216 years before Wolf's birth! And so I renamed the DATA_DEV to RUBBISH_DATA and added the flag column, leaving those four observations inside where they probably belong...
  - Wrote a new script to deal uniquely with deleting the duplicates
    - finished writing move_data_to_bin and delete_entered_twice_duplicates
    - executing delete_entered_twice_duplicates()... done
    - finished commenting these data points in rubbish data in both databases
- Wednesday June 26
  - wrote a new method in db_edit for appending comments rather than replacing them
  - wrote *unreasonable_sn_flag()* a method that takes a look at the groups number and the sunspots number of each entry and determines if it's realistic or not. I decided somewhat arbitrarily that if the groups number was higher than 30 it would be flagged with flag 4, if the groups number was higher than 60 it would be marked with a 5 this is beyond unreasonable. I did something similar for the sunspots number $sunspots > 100 \Rightarrow flag := 4$ and $sunspots > 250 \Rightarrow flag := 5$ (see the flags section 2.2.1). The method was executed and ran without a hitch (after a bit of debugging)
  - just set another 212 flags for putting things in the bin. There are still 4000 pairs of duplicates that need attending to but considering i started with 14000 that's not bad... Some of the duplicates may be left as they are. Also i figured out that i had flagged some which just had a 0 sunspot number and so i went and unflagged them.

- i spent alot of time scrutinising what i had flagged, rereading my scripts, seeing that I've been using really inefficient algorithms, checking things are in the right place. And wondering how on earth many things ended up with the flags they ended up with.
- something that has been annoying me in this search is I can't seem to be able to determine what is an unreasonable number of sunspots that can appear on the sun, because many many observer record having over 250. This is why I will start using graphical tools to help me figure all of this out. I will make the graphs in a jupyter notebook.

- Thursday June 27
  - first thing i did today is to go though and look at lots of the flagged data from yesterday on the mysql databases
  - Panic! While searching I came across a big problem. Many of the datapoints are labelled '*' in the comments, this corresponds to when there is an asterix in the Mitteilungen journals, but here's the twist: the star is usually a reference to the fact that there is a change in the observer's telescope to his/her secondary lunette. This is written nowhere is many cases in the digital database! This is a new task I must take on am I to accomplish my mission here:
    1. Correct all the comments so that they display useful information i.e. '*' — '* = 8 cm Oeffnung mit 64-facher Vergrosserung und Polarisationshelioscop'
    2. When you tackle the 'Creating New Aliases' part of the project (see my Github - username = dcxSt - project sun)
  - I found there I had flagged all the mysterious '*' comment 1 and that none of them have found their way into my pristine database GOOD_DATA_SILSO and so I went through each of them individually and wrote the changes that I implemented in the python method 'correct_asterix_comments()' in script 'db_homogenise_comments.py'. This took quite some time and included translating German with my good friend google-translate. This corrected about 250 data-points' comments (i didn't bother to count)
  - after a long search of the data in GOOD_DATA_SILSO with FLAG=3 which have no superior duplicates I found that these were infact correctly flag and that their double had not yet made it into my new database because there were commented (usually with an asterisk *) so i moved them into the bin
  - Found a bug in move_flag3_to_bin() which may have been causing some of the perplexing problems I had earlier
  - I ran the method 'flag_many_duplicates()' many times using the duplicates text files I'd made earlier for inspiration to change it subtly so as to catch those sneaky no good duplicates!
  - IMPORTANT: I just found some data which has been written in in the wrong year. In rubrics 820 students have mistakenly typed in the data for Wolfer in the year 1900 and written it in the year 1899.
  - In the folder duplicates/3 2019.06.27 I am writing the file corrections_needed_handwritten.txt which outlines the corrections which are to be made to the data if we want to solve some of these duplicates.

2.3.3 Python scripts - what they contain

3 Comparaison du data avant et apres + visualisations

3.0.1 The original sql data tables format

Table 2: DESCRIBE DATA

Field	Type	Null	Key	Default	Extra
ID	int(11)	No	PRI	NULL	auto_increment
DATE	date	YES		NULL	
FK_RUBRICS	int(11)	YES	MUL	NULL	
FK_OBSERVERS	int(11)	YES	MUL	NULL	
GROUPS	int(11)	YES		NULL	
SUNSPOTS	int(11)	YES		NULL	
WOLF	int(11)	YES		NULL	
QUALITY	int(11)	YES		NULL	
COMMENT	text	YES		NULL	
DATE_INSERT	datetime	YES		NULL	
FLAG (i added this)	tinyint(1)	YES		NULL	

Table 3: DESCRIBE OBSERVERS

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	NULL	auto_increment
ALIAS	varchar(50)	YES		NULL	
FIRST_NAME	varchar(50)	YES		NULL	
LAST_NAME	varchar(50)	YES		NULL	
COUNTRY	varchar(50)	YES		NULL	
INSTRUMENT	varchar(50)	YES		NULL	
COMMENT	text	YES		NULL	
DATE_INSERT	datetime	YES		NULL	

Table 4: DESCRIBE RUBRICS

Field	Type	Null	Key	Default	Extra
RUBRICS_ID	int(11)	NO	PRI	NULL	auto_increment
RUBRICS_NUMBER	int(11) unsigned	NO		NULL	
MITT_NUMBER	int(11) unsigned	NO		0	
PAGE_NUMBER	int(11) unsigned	YES		NULL	
SOURCE	text	NO		NULL	
SOURCE_DATE	date	YES		NULL	
COMMENTS	text	YES		NULL	
DATE_INSERT	datetime	YES		NULL	
NB_OBS	int(11)	YES		NULL	

3.0.2 My new sql data table format

Table 5: DESCRIBE DATA (the only table)

Field	Type	Null	Key	Default	Extra
ID	int(11) unsigned	No	PRI	NULL	auto_increment
DATE	date	YES		NULL	
GROUPS	int(11)	YES		NULL	
SUNSPOTS	int(11)	YES		NULL	
WOLF	int(11)	YES		NULL	
COMMENT	text	YES		NULL	
DATE_INSERT	datetime	YES		NULL	
OBS_ALIAS	varchar(50)	YES		NULL	
FIRST_NAME	varchar(50)	YES		NULL	
LAST_NAME	varchar(50)	YES		NULL	
COUNTRY	varchar(50)	YES		NULL	
INSTRUMENT_NAME	varchar(50)	YES		NULL	
RUBRICS_NUMBER	int(11)	YES		NULL	
MITT_NUMBER	int(11)	YES		NULL	
PAGE_NUMBER	int(11)	YES		NULL	
FLAG	tinyint(1) unsigned	YES		NULL	
RUBRICS_SOURCE	text	YES		NULL	
RUBRICS_SOURCE_DATE	date	YES		NULL	

3.0.3 Graphs and visual representations

As you can see, there are some famous legends in solar science such as Carrington and Kew who were able to see over 4000 sunspots at once on a single face of the sun with their crappy telescopes that they had in the 19th century!

3.0.4 Thought repository - ideas that may or may not come into fluition depending on how efficiently I work and get things that need to be done done

- make some data visualisations to compare each observer's primary and secondary observing equipment
- for each day / month / year find the highest observation and the lowest observations and add it to the graph so that we have like an upper bound and a lower bound.
- figure out how to smooth graphs with matplotlib and make something nice out off the big mess i currently have
- pie chart of observers with their number of observations
- in the final sunspots number graph cut it into 3 or 4 sections that mark changes in the theory behind sunspots: before wolf ; time where plato's ideas of the sun being a perfect sphere still were around ; 1908 George Ellery Hale discovers the magnetic link (p14 of nature's 3rd cycle) ; 1955 eugene parker's theory (p19 of nature's 3rd cycle) ; Nasa send their probe to near the sun

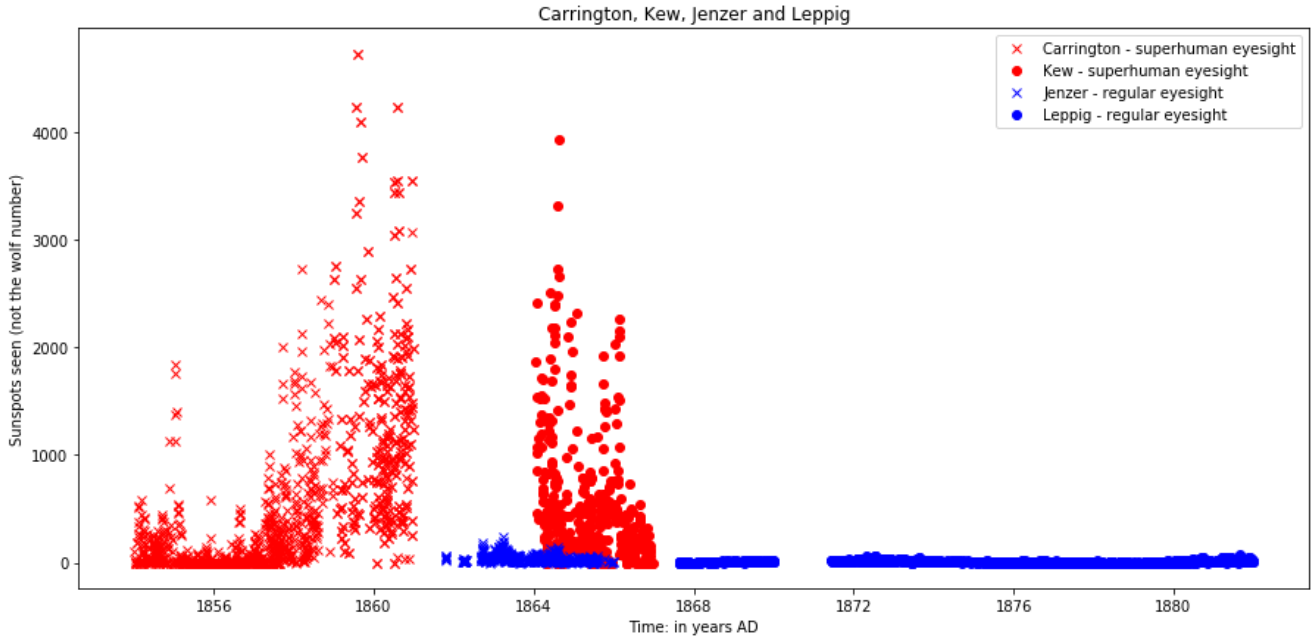

Figure 1: Carrington has great eyesight!