Documentation

ASRB RADIATION
Code porting from IGOR into R

Prisco Frei (prisco.f@hotmail.com)

SLF, Zivi, May – July 2016

Documentation

Inhaltsverzeichnis

In case the hyperlinks within this document don't work properly, open the Word version of this file, save it, re-open the new saved Version and generate a pdf!

1. Introduction

1.1. General purpose of this work

Radiation measurements in the Alps have a long history. As the surface radiation is one of the most important components of our climate system, an accurate estimation of these parameters are preferred. However, due to the sensitivity of the measurment devices, data post-processing is very essential. In their PhD Theses Christoph Marty and Bruno Dürr (among others) developed some methods to correct for faulty radiation measurements and to adjust them according to site specific conditions.

Originally, the code for this post-processing procedure was written in IGOR wavemetrics (www.www.wavemetrics.com/). With the intention of providing these data online via the OSPER platform (www.osper.ch) in near real time, running the IGOR code operationally doesn't seem to be appropriate. Thus, it has been decided to port the entire code into the R language (www.r-project.org) and to fix possible (still existing) bugs in IGOR.

With help of the R code, it is now possible to post-process raw radiation measurements for the ASRB stations Davos (dav) , Weissfluhjoch Versuchsfeld (wfj_vf) and Weissfluhjoch Gipfel (wfj_gipf) in near real time which allows for a faster data distribution. The general setup is built quite dynamically such that additional stations (having the same setup device as the already existing stations and measuring frequency of 2 minutes) can be added at any time.

The following chapters are not meant to lead you trough every radiation step in detail (here I'd like to refer to the already existing literature of Christoph Marty and Bruno Dürr, see here). This documentation should be used as a guide if one aims to do any changes or improvements on the code or likes to do some case studies.

2. Folder structure

There are three main folders: EVALUATION, TEST and OPERATIONAL. The OPERATIONAL directory contains all files to run the code in operational mode (see Section 5), while in the TEST folder, case studies, re-runs etc. can be carried out without affecting operational results at all (see Section 6). The EVALUATION folder contains scripts and data which were used to check whether the code porting from IGOR to R worked properly (see Section4). Each of these three main folders contains the same sub-directories: **ASRB_files**, **preprocessed_data**, **final_data, r_files**, and **r_plots**.

- **ASRB_files**: location for input files which are needed to postprocess data (more about in Section 3).preproccessed_data: Contains data which were partly proccesed (derivation of radiation in W/m^2 based on device measurements taken in V, checked for missing and and disturbed values (flagging) etc.). In the main R code one can choose whether this intermediate step should be stored off or not (more information in Section 3).
- **final_data**: contains the final post processed radiation data
- **r_files**: contains all R-files and R-functions which are used to process data
- **r_plots**: Figures (not in TEST)
- **r_stats:** Daily and monthly (OPERATIONAL only) statistics about measurement errors (flags)

In addition, the EVALUATION and OPERATIONAL folder contains the subdirectories **raw_data** and **varia** (see Section 4) and **products** (see Section 5).

3. R code

In the following, the main R code and its functions will quickly be introduced by using the operational version (ASRB_data_processing_main_file_OPERATIONAL.r) as a reference. The corresponding main files in EVALUATION and TEST look very similar and are just slightly modified for their corresponding needs (see Sections 4 and 6).

3.1. R packages to install and source R functions

The code is tested under the R version 3.0.2.

If you run it for the first time make sure that the following R-packages are installed:

- insol (link)
- EMD (link)
- pracma (link)
- zoo (link)
- fBasics (link) (The fBasics depends on the stabledist package! As its most current version (stabledist 7.0) isn't compatible with R version 3.0.2, one first needs to install version 6.5 manually (link)

At the beginning the individual R functions are loaded from the r_functions directory.

```
21
22 # ***** load R packages *****
23 library(insol) # package to derive solar parameters
24 library(EMD) # package to derive local maximas
25 library(pracma) # package to derive local maximas
26 library(zoo) # package to linearly interpolate missing values
27 library(fBasics) # Generator for Portable Random Innovations
28
29 # ***** load R functions*****
30 source("r_functions/solar_param_calc.r")
31 source("r_functions/check_missing_data_and_outliers.r")
32 source("r_functions/preprocess_data.r")
33 source("r_functions/shadow_width_solar_noon_calc.r")
34 source("r_functions/sw_diffuse_calc.r")
35 source("r_functions/sw_corr.r")
36 source("r_functions/lw_corr.r")
37 source("r_functions/save_data.r")
38
```

3.2. Special parameter setup

```
38
39 astro_data<-1 # Choose how the solar parameters should be calculated
40              # astro=1 --> Load from astro files
41              # astro=2 --> Compute parameters with insol package
42              # For more information see r_functions/solar_param_calc.r
43
44
45 save_preprocessed_data<-1 # Choose whether the preprocessed data should be saved (=1) or
46                           # not (=0). For more information see r_functions/preprocess_data.r
47
48
49 add_noise<-1 # choose wheter noise should be added to the parametrised noon values (=1)
50              # or not (=0). For more information see  r_functions/sw_corr.r and
51              # r_functions/lw_corr.r. Omitting the noise term (add_noise=0) is sometimes
52              # useful for accurate comparisons!
53
54         # !!! add_noise should always be turned on (=1) in OPERATIONAL MODE !!!
55 ##############################################################################
56
```

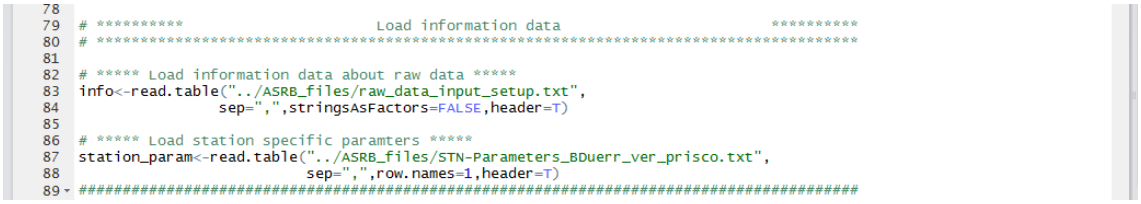
The following setup parameters can be defined:

- **astro_data**: Defines whether the solar parameters should be read from the file (=1) or computed with R's insol package (=2). More information in Subsection 3.4.
- **save_preprocessed_data**: In a first step raw radiation data is only pre-processed (identification of missing or perturbed data, conversion from Voltage measurements to W/m^2 etc. If this parameter is set to 1, these intermediate data is stored (under preproccesed_data), if the parameter is set to 0, data won't be stored.
- **add_noise**: Due to the shadowband influence at noon (see Section 3), the radiation needs to be interpolated for this time. This is done by a polynominal fit of 2nd order or a linear fit with

an additional noise term (**add_noise**=1). However for evaluation purposes it might be useful to turn this noise term off (**add_noise**=0). In the operational mode, the noise should always be turned on!

3.3. Input information files

In order to run the code, two information files need to be loaded.



```
78
79  # **********                    Load information data                    **********
80  # ********************************************************************************
81
82  # ***** Load information data about raw data *****
83  info<-read.table("../ASRB_files/raw_data_input_setup.txt",
84                   sep=",",stringsAsFactors=FALSE,header=T)
85
86  # ***** Load station specific paramters *****
87  station_param<-read.table("../ASRB_files/STN-Parameters_BDuerr_ver_prisco.txt",
88                            sep=",",row.names=1,header=T)
89  ################################################################################
```

raw_data_input.txt (located in ASRF_files folder) contains general information about the station data which should be processed. It is also the file where an additional station can be added (or removed). Here a list about the parameters which need to be set:

- **stn_name_long:** Name of the station (e.g. Davos)
- **stn_short:** Short name of the station which will later on be used to create the data output paths (e.g. dav)
- **path_to_raw_file:** Path to the raw radiation measurements of the corresponding station. The paths can be absolute or relative to the location of the main file ASRB_data_processing_main_file_OPERATIONAL.r (e.g. ../raw_files/)
- **raw_file_name:** Name of the raw radiation data file (e.g. 16Flst_ASRB_Radiation.dat)
- **lines_to_skip:** How many header lines need to be skipped until the first measurement entry line starts.
- **rad_in:** Is incoming radiation measured at this station? yes=1, no=0. The code doesn't work for stations at which no incoming radiation is measured!
- **rad_out:** Is outgoing radiation measured at this station? yes=1, no=0. The code works for stations where incoming radiation is available but not outgoing, e.g. wfj_gipf
- **col_nr_date:** Column of the raw data file in which the date is located.
- **col_nr_U_pyr_in:** Column of the raw data file in which the voltage data of the upward pointing pyranometer (incoming radiation) are located.
- **col_nr_U_pir_in:** Column of the raw data file in which the voltage data of the upward pointing pyrgeometer (incoming radiation) are located.
- **col_nr_T_pyr_in:** Column of the raw data file in which the upward pointing pyranometer body temperature data is located.
- **col_nr_T_pir_in:** Column of the raw data file in which the upward pointing pyrgeometer body temperature data is located.
- **col_nr_T_n_in:** Column of the raw data file in which the upward pointing northward dome temperature sensor data is located.
- **col_nr_T_se_in:** Column of the raw data file in which the upward pointing south-eastward dome temperature sensor data is located.
- **col_nr_T_sw_in:** Column of the raw data file in which the upward pointing south-westward dome temperature sensor data is located.
- **col_nr_STD_pyr_in:** Column of the raw data file in which the voltage standard deviation of the upward pointing pyranometer is located.
- **col_nr_STD_pir_in:** Column of the raw data file in which the voltage standard deviation of the upward pointing pyrgeometer is located.

- **col_nr_U_pyr_out:** Column of the raw data file in which the voltage data of the downward pointing pyranometer (outgoing radiation) are located.
- **col_nr_U_pir_out:** Column of the raw data file in which the voltage data of the downward pointing pyrgeometer (outgoing radiation) are located.
- **col_nr_T_pyr_out:** Column of the raw data file in which the downward pointing pyranometer body temperature data is located.
- **col_nr_T_pir_out:** Column of the raw data file in which the downward pointing pyrgeometer body temperature data is located.
- **col_nr_T_n_out:** Column of the raw data file in which the downward pointing northward dome temperature sensor data is located.
- **col_nr_T_se_out:** Column of the raw data file in which the downward pointing south-eastward dome temperature sensor data is located.
- **col_nr_T_sw_out:** Column of the raw data file in which the downward pointing south-westward dome temperature sensor data is located.
- **col_nr_STD_pyr_out:** Column of the raw data file in which the voltage standard deviation of the downward pointing pyranometer is located.
- **col_nr_STD_pir_out:** Column of the raw data file in which the voltage standard deviation of the upward pointing pyrgeometer is located.

If for a specific station, no outgoing radiation measurements are available, the corresponding **col_nr_XXX_out** is set to **NA**. See for example wfj_gipf.

STN-Parameters_BDuerr_ver_prisco.txt (located in the ASRB_files folder) contains station specific parameter values. The line order shouldn't be changed at all, i.e., if in the future additional parameters are added, append them at the end of the file. In case of new stations, add the corresponding station parameters in this file. The column name should be set up in the following way:

Column for station parameters related to incoming ("in") radiation:

- **<stn_short>_in** : short station name as defined in raw_data_input.txt (e.g. **dav_in**)

Column for station parameters related to outgoing ("out") radiation:

- **<stn_short>_out** : short station name as defined in raw_data_input.txt (e.g. **dav_out**)

After defining the date which should be processed, i.e., the day before the actual date in the operational mode (defined automatically!), and loading of the raw data file of the corresponding station, the first R function (solar_param_calc.r) is called.

3.4. R function solar_param_calc.r

This R function loads various solar parameters for the corresponding day. As already mentioned in Subsection 0, with the parameter **astro_data,** one can choose based on which source the solar parameters should be derived. If **astro_data** is set to 1, values of the specific day are loaded from a pre-calculated astro file located under ASRB_files/astro/. These files were created at the PMOD in Davos (http://www.pmodwrc.ch/) with help of a Fortran code (sun_list.for). Under ASRB_files/astro/ solar parameter data is available up to the year 2033. In addition there is the possibility to compute the solar parameters with help of the R package insol. To do so, simply set **astro_data=2.** However as long as the astro files are available one should use these ones. This allows for homogeneity in the time series.

3.5. R function preprocess_data.r

After solar parameters are extracted, the raw data is pre-processed with help of the R function preprocess_data.r. Within this function it is ensured data each day gets 720 lines of measurements (2minute data measurements from 00:00-23:58 o'clock, i.e. missing timestamps are filled in). Timestamps are converted from UTC+1 to UTC and radiation measurements are converted from V to W/m^2.

Note: If the time stamp of the raw data file, which is currently %d.%m.%Y %H:%M:%S (i.e. 01.12.2015 09:58:00) should change in the future, the code needs to be adjusted in this routine.

```
6
7   # get dates of input file , the are measured in CET=UTC+1 = GMT+1 = Etc/GMT-1
8   raw_date_cet<-as.POSIXct(raw_dat[,stn_info$col_nr_date],"%d.%m.%Y %H:%M:%S",tz="Etc/GMT-1")
9
10  # convert dates of input files to UTC
11  raw_dat[,stn_info$col_nr_date]<-as.character(format(raw_date_cet,tz="UTC", "%d.%m.%Y %H:%M:%S"))
12  raw_date<-format(raw_date_cet,tz="UTC", "%Y-%m-%d")
13
14  # extract indices of yesterday's raw date
15  raw_date_ind<-which(raw_date==yesterday)
16
17  ################################################################################
```

3.6. R function check_missing_data_and_outliers.r

Before (possibly) storing the preprocessed data, the array is checked for missing values and outliers with the R function check_missing_data_and_outliers.r.
Here a quick info about the routine:

- **Missing value**: If data is missing, a flag for the corresponding parameter (Sw_{in},Lw_{in} or Lw_{out}) is set to 128.
- **Outlier**: If the a measurement value of Sw or Lw radiation is not in a predefined range, the corresponding values gets flagged with 64 (but the outlier is not deleted!)
- **Wildpoint**: If a particular Sw or Lw measurement is significantly larger than the running median value (window width = 3 measurements), a flag = 16 is set and the corresponding measurement value is set to median value.
- **I_heiz:** In case that the electric current, used to ventilate the devices, is not within a specific range (station dependent, see STN-Parameters_BDuerr_ver_prisco.txt), a flag=32 will be **added**. The addition is only applied if the data is not missing. E.g.:
  --> Flag_Sw_in=32 means that only the electric current is disturbed but that the measurement is within the expected range.
  --> Flag_Sw_in=64 means that the measurement is not within the expected range while the electric current is ok!
  --> Flag_Sw_in=96 means that both the electric current as well as the measurement are not in the expected range
  (The I_heiz flag is only added to Flag_Sw_in and Flag_Lw_in but not to Flag_Sw_out and Flag_Lw_out since the same electric current is used)

3.7. Store of preprocessed data

Finally if desired, the preprocessed files can be stored.

```
190
191  # **********                Store of first part of processed data            **********
192  # ********************************************************************************************
193  if (save_preprocessed_data==1){
194
195    file<- paste(format(yesterday,"%Y%m%d"),"_",stn,"_002_new_test.csv",sep="")
196    path<- paste(data_002_out_path,stn,"/",sep="")
197
198    # Create directory if it doesn't already exists
199    if (file.exists(path)==0){
200      dir.create(path,recursive=T)
201    }
202
203    # save data
204    write.table(dat_002,paste0(path,file), sep=",", row.names=F,quote=F)
205  }
206  # ##############################################################################
```

3.8. R function shadow_width_solar_noon_calc.r

At some radiation measurement sites a fix installed shadowband leads to noon dips in Sw_{in} and Lw_{in} radiation. The dip in Lw_{in} allows for a correction of the Sw influence on the pyrgeometer measurements (which are supposed to measure the atmospheric (Lw) radiation). However, the shadowband also blocks the direct Sw radiation during noon, which is later in filled up by using information from the R function shadow_width_solar_noon_calc.r checks for possible dips at noon which will later on be "filled up". The dip is located by finding local peaks in the standard deviation measurement of the pyranometer shortly before and after solar noon (see Subsection 7.1).

3.9. R function sw_diffuse_calc.r

Sw_{in}, i.e. the global incoming radiation, can be divided into a direct component ($Sw_{dir,in}$) and a diffuse component ($Sw_{dif,in}$). By parametrising $Sw_{dif,in}$ (see Bruno_Duerr_PhD_thesis.pdf) one can estimate $Sw_{dir,in}$ which is needed to adjust Lw_{in}, over the entire day. Thus one doesn't rely on the noon information of stations with a shadowband which only give the Sw_{dir}/Sw_{in} relationship for a particular time of the day. In his PhD thesis, Christoph Marty estimated $Sw_{dif,in}$ with this noon information and interpolated it over the entire day by appyling a cubic spline. This method is of course only possible for shaded stations. Bruno Dürr came up with a new (more appropriate) approach in which the shadowband is not needed. All past data have been processed with this new approach. The latter procedure is implemented in sw_diffuse_calc.r.

3.10. R function sw_corr.r

After extracting the solar parameters, pre-processing the data and estimating the diffuse radiation $Sw_{dif,in}$ we can finally step to the correction of Sw_{in} (Sw_{out} won't be corrected at all).This is done by calling the sw_corr.r function .

In a first step Sw_{in} at solar noon is estimated by applying 2nd order polynominal fit and measured minimal Sw_{in} around noon is calculated. These2 steps allow for a quantification of the percental direct Sw radiation at noon, which is needed to correct for the noon dip later on. Thus, it only needs to be derived for shaded stations.

In the second part of the function two Sw_{in} adjustments are applied:

- **Shadowband influence at noon:** If a noon dip in Sw_{in} is defined, the values of the corresponding time are interpolated by applying a polynominal (2nd order) or linear fit (according to different preconditions).
- **Shadowband influence over the entire day:** As the shadowband constantly covers a fractional part of the sky (estimated: 2.5%), the measured $Sw_{dif,in}$ contributing to Sw_{in} is too low and needs to be adjusted

```
343
344     # Adjust Sw_in_corr and  because of shadowband influence
345     Sw_in_corr<-Sw_in_corr+shadowband*1000*Sw_dif_in_for_f
346
```

3.11. R function lw_corr.r

In the lw_corr.r function all adjustments regarding Lw radiation are made. First of all the so called g-correction is applied (to both Lw_{in} and Lw_{out}). It corrects for Lw measurement influences by the station mast. The f-correction accounts for pyrgeometer measurements in the Sw spectra which can't be averted. This adjustmend is only applied to Lw_{in} and uses information derived in the sw_diffuse_calc.r function because it depends on Sw_{dir} (see Subsection 3.9). Last but not least, the possible noon dip in Lw_{in} at shaded stations needs to be filled up. This is done by a linear interpolation.

3.12. R function save_data.r

Once all corrections are made, the post-processed data is stored off by using the save_data.r function. Data is saved in daily file under **final_data/<stn_name>/** with the following filename structure:

<year><month><day>_<stn_name>.csv (e.g. 20160629_dav.csv)
Compared to the old IGOR code, the final data for both radiation types, i.e. incoming and outgoing, at a particular station are stored in the same file. Thus, in such a case the following parameters are available:

- **date:** Timestamp of the measurement (e.g. 09.05.2016 00:22:00)
- **tz:** Time zone (e.g. UTC)
- **doy:** Day of the year
- **Sw_in:** Incoming Sw (global) radiation corrected
- **Lw_in:** Incoming Lw radiation corrected
- **Flag_Sw_in:** Flags for incoming Sw (global) radiation (see check_missing_data_and_outliers.r and sw_corr.r functions for more information)
- **Flag_Lw_in:** Flags for incoming Lw radiation (see check_missing_data_and_outliers.r and lw_corr.r functions for more information)
- **T_pir_in:** Body temperature of upward pointing pyrgeometer (same data as in raw input files)
- **STD_pyr_in:** Standard deviation of Sw_{in}
- **STD_pir_in:** Standard deviation Lw_{in}
- **Sw_dif_in:** Parametrised incoming diffuse radiation (according to method of Bruno Dürr**)**
- **Sw_out:** Outgoing Sw (global) radiation (same data as raw input files)
- **Lw_out:** Outgoing Lw radiation corrected
- **Flag_Lw_out:** Flags for outgoing Lw radiation (see check_missing_data_and_outliers.r and lw_corr.r functions for more information)
- **T_pir_out:** Body temperature of downward pointing pyrgeometer (same data as in raw input files)
- **STD_pyr_out:** Standard deviation of Sw_{out}
- **STD_pir_out:** Standard deviation of Lw_{out}

4. Evaluation

4.1. General Evaluation information

To ensure a proper Code porting from IGOR to R, pre-processed data based on the two codes and for the 12 month period between the 01.04.2015 and the 31.03.2016 were compared with each other. As expected some differences occurred between the two re-processing laguages due to language errors. However unexpectedly these rounding errors rarely caused major differences in the radiation fluxes during solar noon, because different mininma were found if different conditions got true. The IGOR based data can be found under EVALUATION/varia/. There are different IGOR data folders:

- **IGOR_asrb3_no_noise:** Monthly pre processed files for the stations Davos (incoming radiation=da, outgoing radiation = do),Weissfluhjoch Versuchsfeld (incoming radiation=vsf, outgoing radiation=vof) and Weissfluhjoch Gipfel (incoming radiation=wfj, no outgoing radiation) with deactivated noise term for the solar noon dip filling. These evaluation is performed with "no noise", because the IGOR introduced noise (enoise) is random and not normal distributed
- **IGOR_asrb3_noise:** Monthly pre processed files for the stations Davos (incoming radiation=da, outgoing radiation = do),Weissfluhjoch Versuchsfeld (incoming radiation=vsf, outgoing radiation=vof) and Weissfluhjoch Gipfel (incoming radiation=wfj, no outgoing radiation) with activated noise term for the solar noon dip filling

Then, during the Code porting some bugs in the IGOR code were detected. These include:

- Wrong routine checking for a possible leap year
- Wrong computation of doy (day of the year)
- Slightly wrong routine to define possible peaks (which are needed to detect noon dips in Sw_{in} and Lw_{in})

The post-processed data of this not entirely correct IGOR code can be found under EVALUATION/varia/IGOR_asrb3_no_noise_not_corrected_code and EVALUATION/varia/IGOR_asrb3_noise_not_corrected_code, respectively. The impact of these errors can be large (>10W/m^2) for the shadowband correction on some single days. However, generally the impact is very small.

The modifications in the Igor code can be compared by taking a look at the two pdf IGOR_code_bugs.pdf and IGOR_code_modified.pdf which are stored under Documentation/igor_codes/.

! Very important !

The main operational (ASRB_data_processing_main_file_OPERATIONAL.r), the test (ASRB_data_processing_main_file_TEST.r) R codes as well as the corresponding R functions loaded within these two files are based on the modified IGOR code (IGOR_code_modified.pdf).

In the EVALUATION/r_files folder there are two different main files:

- **ASRB_data_processing_main_file_EVALUATION.r**: R code based on modified IGOR code (IGOR_code_modified.pdf). Calls the R functions stored in the R_functions folder.
- **ASRB_data_processing_main_file_EVALUATION_not_corrected_IGOR_code.r:** R code based on the IGOR code still containing the bugs mentioned above (IGOR_code_bugs.pdf). Calls the R functions in the r_functions_not_corrected_IGOR_code folder.

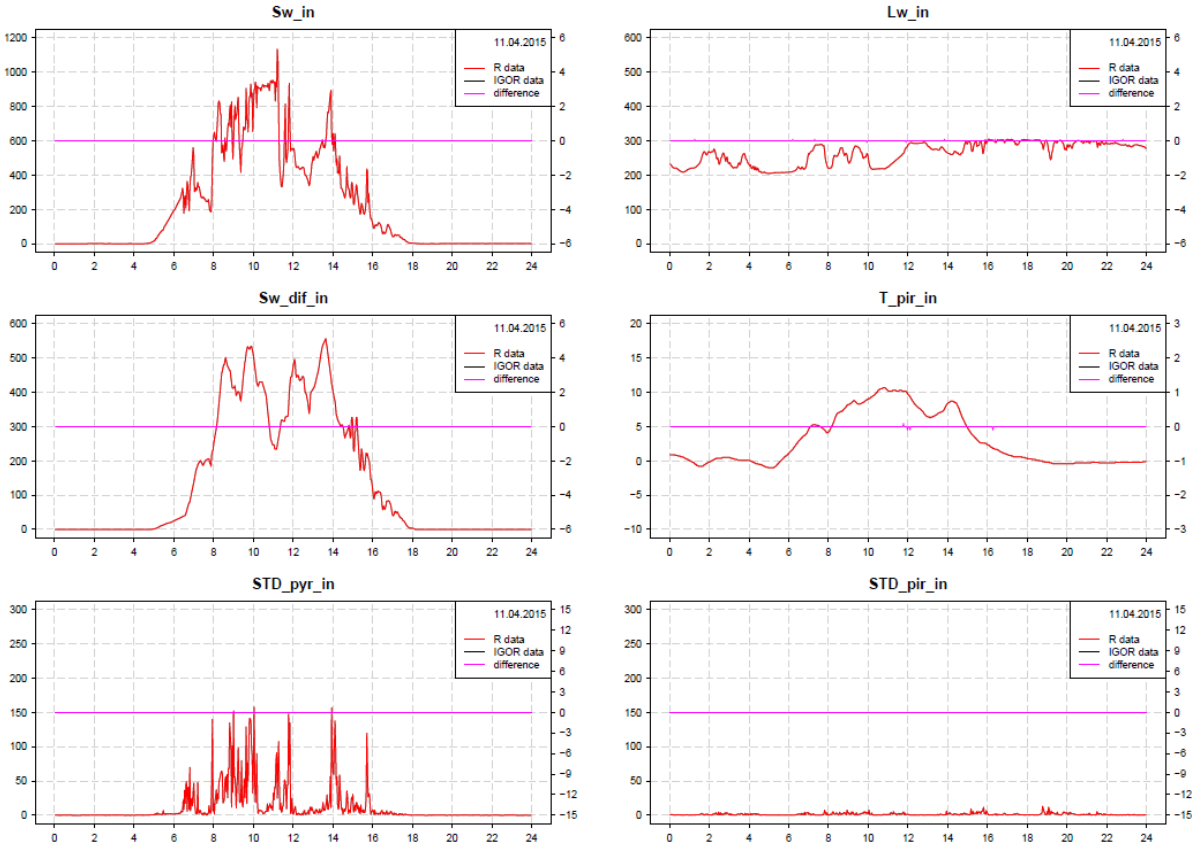
The ASRB_data_processing_main_file_EVALUATION_not_corrected_IGOR_code.r is only kept for backup reasons and shouldn't be used in the future anymore.

Furthermore it has been decided to post-process radiation measurements after March 2015 again with the here presented R code. The re-processing of older data is complex, since many metadata have changed over the years. There, older data than April 2015 has not been reprocessed, i.e. they are based on the old IGOR code (IGOR_code_bugs.pdf). For more information contact Christoph Marty.

4.2. Evaluation Plots(no noise)

In the following some evaluation plots which are based on the post-processed R data without noise stored under EVALUATION/final_data/no_noise/ and the corresponding IGOR data stored under EVALUATION/varia/IGOR_asrb3_no_noise/ will be discussed. The plots were generated with the R code ASRB_plots_IGOR_vs_R.r (EVALUATION/r_files/) and are located under EVALUATION/r_plots/no_noise/.

For each station, day and radiation type (ingoing (in) or outgoing (out)) daily panel plots are generated. The contained information is explained by using the example for the incoming radiation on Weissfluhjoch Versuchsfeld at the 11.04.2015 (EVALUATION/r_plots/no_noise/wfj_vf/in/):



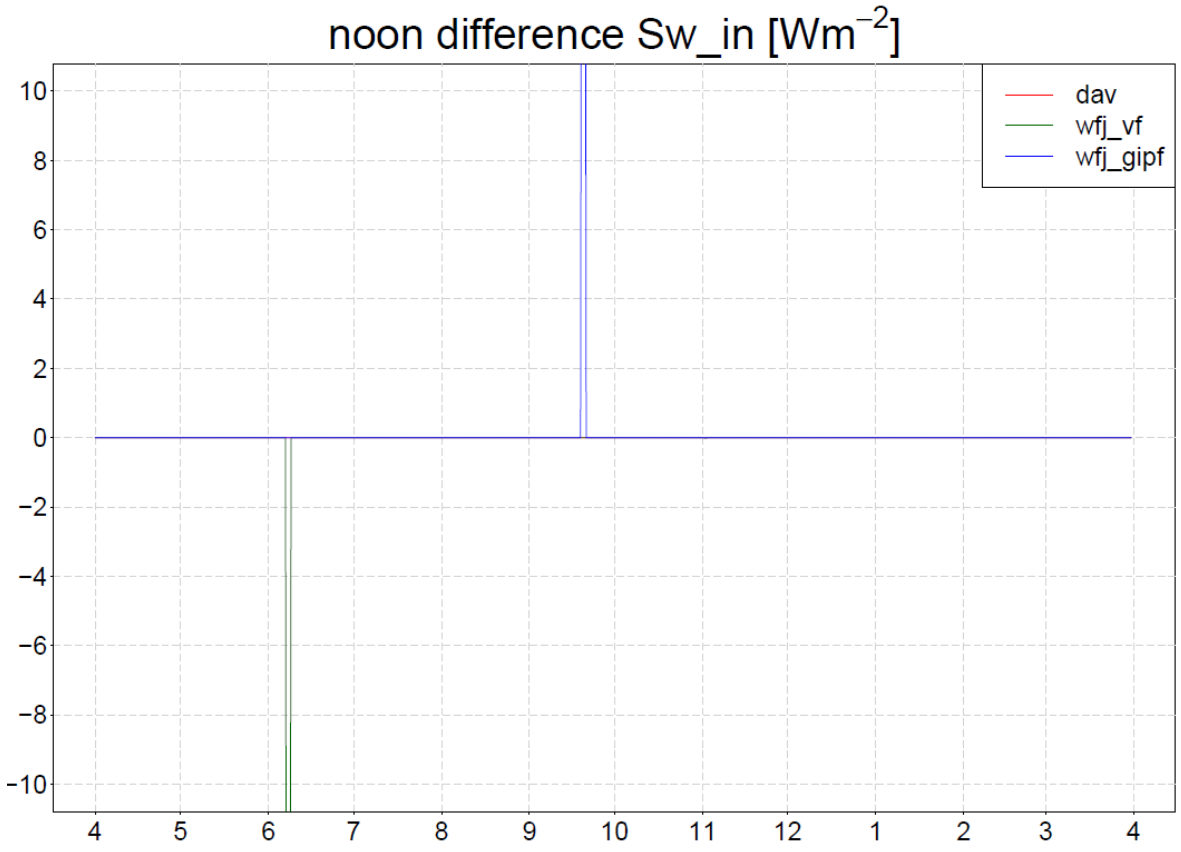
The red (black) lines show the absolute post-processed measurements based on the R (IGOR) code while the difference (R code - IGOR code) is denoted by the magenta line. The deviations can be read from the right y-axis.

In addition, for Swin,Lwin and Swdif,in daily mean difference times series (**day_diff_<radiation_parameter>.pdf**) and noon difference time series (10-12 UTC, **noon_diff_<radiation_parameter>.pdf**) are generated and saved under EVALUATION/r_plots/no_noise/. These plots allow for a quick check of possible deviations between the R and IGOR code over the entire evaluation period between April 2015 and March 2016.

4.3. Major post-processed data differences between the IGOR and R code (no noise)

In the following, the few cases with the largest differences between the post-processed R code and IGOR code are shortly discussed. These dates have been identified by taking a look at the time series showing the noon differences, i.e., noon_diff_Sw_in.pdf, noon_diff_Lw_in.pdf and noon_diff_Sw_dif_in.pdf

For Sw_{in}, two major noon differences between April 2015 and March 2016 are identified:



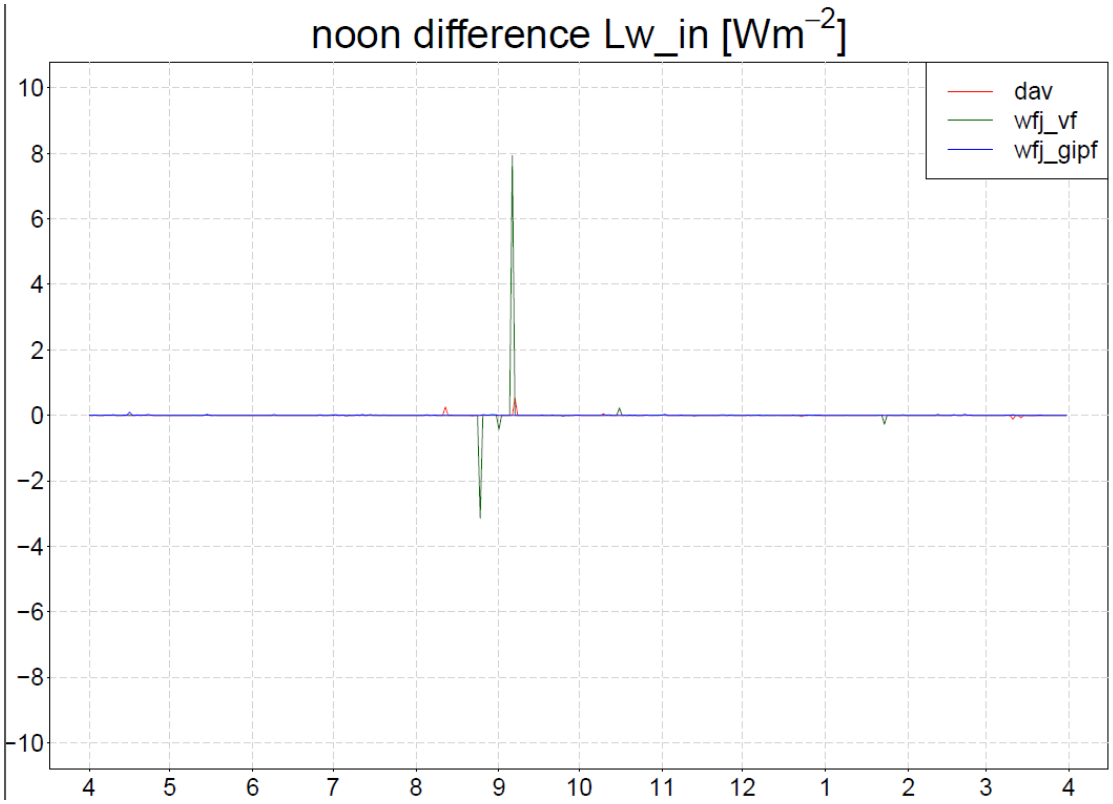
- **wfj_vf, 08.06.2015:** The problem here is that the R code doesn't adjust the initial solar noon index while IGOR does (see shadow_width_solar_noon_calc.r function). This is due to the fact that IGOR rounds **T_sn_in** values which are needed to define the minial Temperature at noon (**T_sn_in_min_noon_ind**) which is then used to decide whether the initial solar noon index (**solar_noon_ind**) should be adjusted or not. In this particular case IGOR finds several **T_sn_in** indices which have the same (minimal!) value and simply takes the first one for **T_sn_in_min_noon_ind** while the **T_sn_in** values in R have a higher accuracy which allows for the detection of just one minimal value. Long story short: Differences are introduced by rounding problems.

```
32
33  indices<-c(solar_noon_ind-7):(solar_noon_ind+7) # indices to analyse
34
35  T_sn_in_min_noon_ind<- indices[which.min(T_sn_in[indices])]
36
37  # derive index difference between theoretical solar noon (solar_noon_ind) and calculated
38  # solar noon (T_sn_in_min_noon_ind)
39  solar_noon_ind_diff<-(T_sn_in_min_noon_ind[1]-solar_noon_ind) # if several minima, take the first one
40
41  # adjust solar noon index (solar_noon_ind) if necessary
42  if (abs(solar_noon_ind_diff)>=5&abs(solar_noon_ind_diff)<=10&T_sn_in[T_sn_in_min_noon_ind]!=0){
43    solar_noon_ind_corr<-round(solar_noon_ind+(solar_noon_ind_diff/2),0)
44  } else{
45    solar_noon_ind_corr<-solar_noon_ind
46  }
47  ################################################################################
```

- **wfj_gipf, 20.09.15:** In IGOR the percentage of $Sw_{dir,in}$ on Sw_{in} is **Sw_dir_perc**=60.00066. In R it's **SW_dir_perc**=59.9653. In the sw_corr.r function, **Sw_dir_perc**=60 is crucial threshold parameter determining the type of the noon dip correction. In this particular case. The noon dip in Sw_{in} is filled up differently due to this threshold dependency. The main error source is again the rounding problem.
-

```
145
146  if(shad_time>5&shad_time<25&STD_pyr_in_Sm<5&Sw_dir_perc_in>60){ # Polyfit for clear sky days
147
```

For Lw_{in}, two major noon differences between April 2015 and March 2016 are identified:



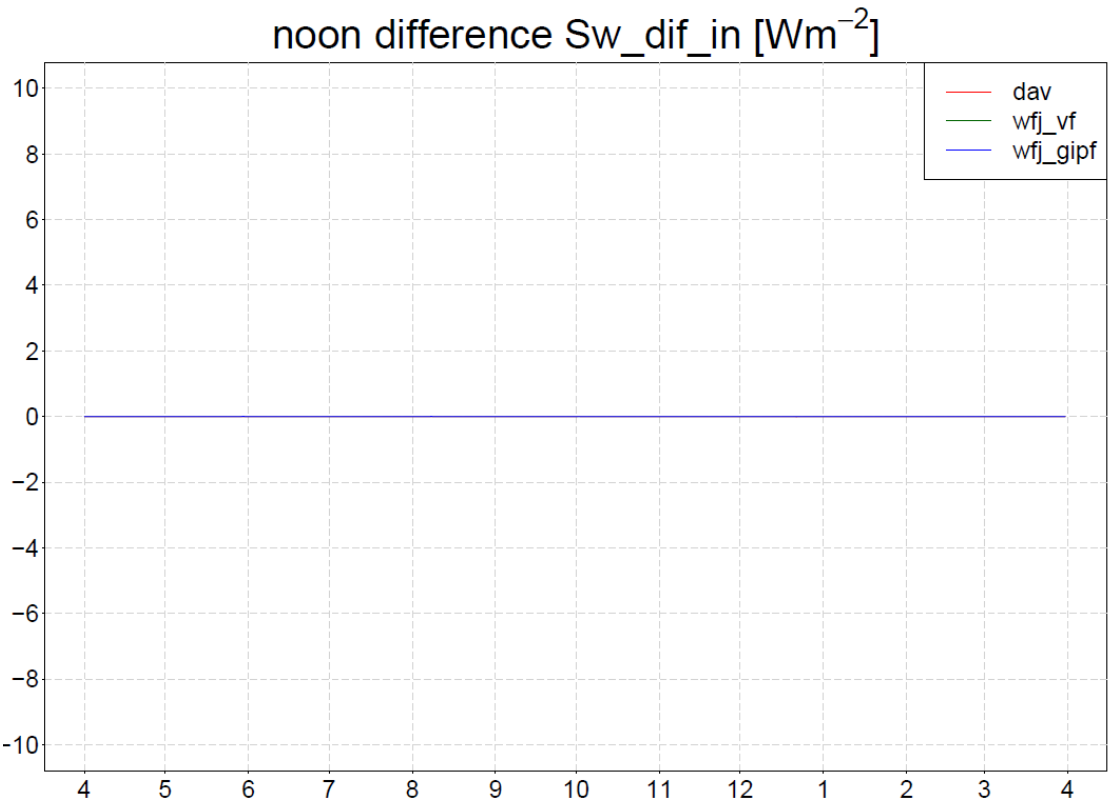
- **wfj_vf, 25.08.2015:** The initial problem is the same as for **wfj_gipf, 20.09.15.** Differen solar noon indices lead to different mean noon pyranometer standard deviations **STD_pyr_in_Sm** (see shadow_width_solar_noon_calc.r function)

```
188
189  # *************      Derive mean standard deviation of incoming shortwave    **********
190  #                  radiaition before/after solar noon (solar_noon_ind_corr)
191  # ********************************************************************************
192
193  # mean standard deviation before noon
194  STD_pyr_in_S1<-mean(dat_002$STD_pyr_in[(solar_noon_ind_corr-20):(solar_noon_ind_corr-10)],na.rm=T)
195
196  # mean standard deviation after noon
197  STD_pyr_in_S2<-mean(dat_002$STD_pyr_in[(solar_noon_ind_corr+10):(solar_noon_ind_corr+20)],na.rm=T)
198
199  # mean standard deviation before and after noon
200  STD_pyr_in_Sm<-mean(c(STD_pyr_in_S1,STD_pyr_in_S2))
201  # ################################################################################
```

  In the lw_corr.r function **STD_pyr_in_Sm** is used as a threshold to define if and how Lw_{in} should be corrected at noon.

- **wfj_vf, 06.09.2015:** Exactly the same problem as for **wfj_vf, 25.08.2015!**

For $Sw_{dif,in}$,no major noon differences between April 2015 and March 2016 are identified:



However this doesn't mean that the R and IGOR code always compute the same values! The R function sw_diffuse_calc.r contains a lot of if statements depending on threshold values. It is always possible that the two codes don't take the same "route" due to rounding errors of specific parameters.

4.4. Summary Evaluation

Our Evaluation (with noise turned off) indicates the proper functionality of the new R code. However, there are rare cases for which the results based on the two codes differ. This is most probably due to rounding problems which might have a large influence on computed values. A comparison with the noise term turned on (see EVALUATION/r_plots/noise/) doesn't make much sense because the noise is random and not normal distributed .

5. Operational setup

There is not much more to say about the operational setup as it has already been used as reference in Section 3.

With the current setup (see ASRB_data_processing_main_file_OPERATIONAL.r), each day, the past 30 days are processed. The advantage: If data of the previous day is not assimilated properly, one has 30 days time to fix possible issues in the raw data and it will still be processed (some days later).

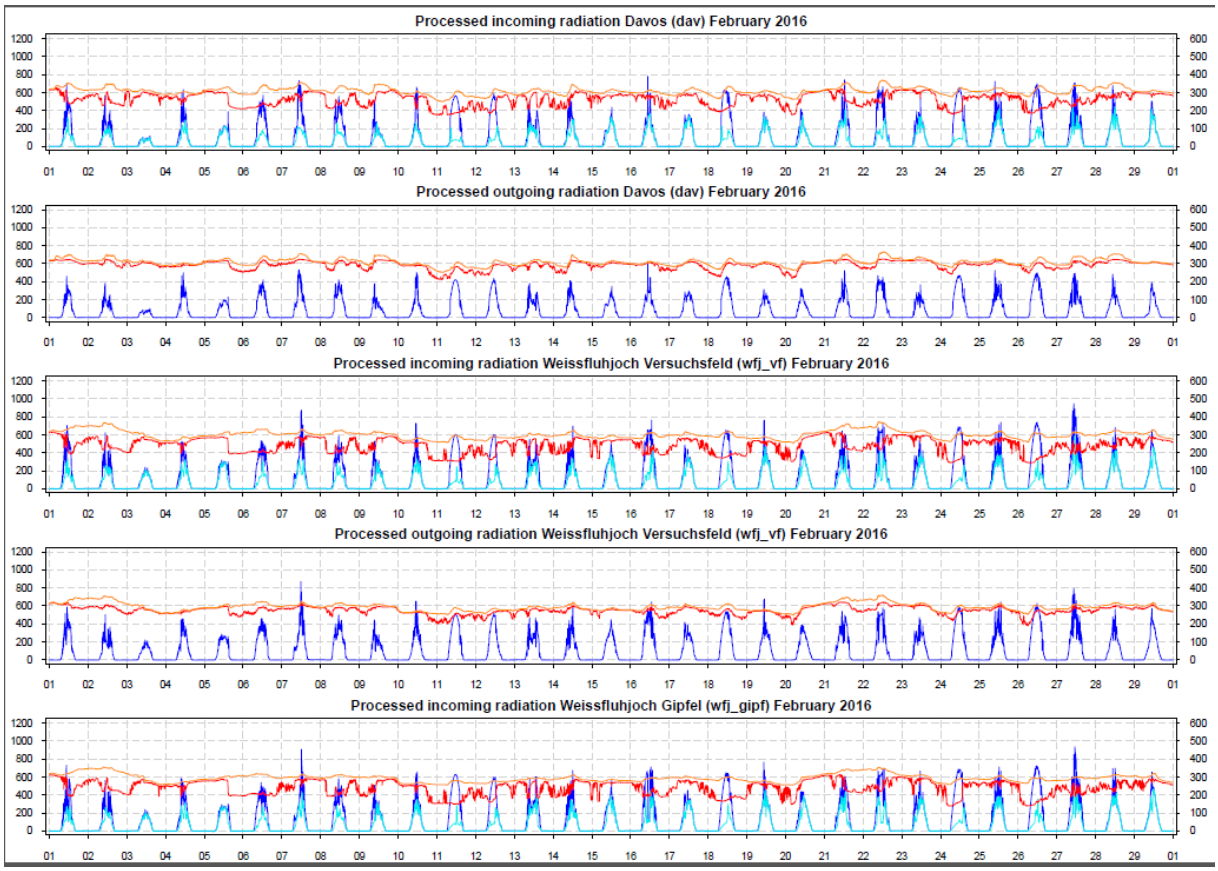
In addition, at the very end of operational code, for each station all final daily files (e.g. OPERATIONAL/final_data/dav/) are concatenated into one large file (see

OPERATIONAL/final_data/merged/). In a second step a copy of this merged file is also saved in the products folder (for more information see Subsection 5.1)

Furthermore, we do the same for the daily flag statistics file. The daily files (only available if flags were observed at any of the three stations for a particular day) are grabbed from OPERATIONAL/r_stats/ and saved under OPERATIONAL/r_stats/merged. A copy of it will also be available in the products folder (for more information see Subsection 5.1).

At the first day of each month, ASRB_data_processing_monthly_plots_and_stats.r will be run as well. This script produces radiation plots for the past two months. First the pdf is saved under OPERATIONAL/r_plots/ and then also copied to the products folder (for more information see Subsection 5.1). A merged Version,i.e. all processed monthly plots in one pdf, are available under OPERATIONAL/r_plots/merged/ as well as in the products folder.

The script is rather "hard coded". In the monthly generated pdf's the relevant data for all stations and types of radiation (incoming and outgoing) is plotted.



The setup is as follows:

- first panel: incoming radiation Davos (dav); second panel: outgoing radiation Davos (dav); third panel: incoming radiation Weissfluhjoch Versuchsfeld (wfj_vf); fourth panel: outgoing radiation Weissfluhjoch Versuchsfeld (wfj_vf); fifth panel: incoing radiation Weissfluhjoch Gipfel (wfj_gipf)

- red lines: Sw radiation; darkblue lines: Lw radiation; lightblue lines: Sw_dif radiation; orange lines: Lw Radiation estimated based on dome temperature of pyrgeometer with Stefan Boltzmann formula
- left y-axis: Sw radiation; right y-axis: Lw radiation, Sw_dif radiation, estimated Lw radiation with pyranometer
- missing data (see Subsection 3.6) will be denoted by darkgreen dots on the y-axis=1200. Other flags are not plotted!

Furhtermore , ASRB_data_processing_monthly_plots_and_stats.r also generates monthly flag statistics files. This is done by merging the daily flag statistics file of the particular month. The output is written to OPERATIONAL/r_stats/monthly/ and the products folder, respectively.

5.1. Products directory

end users find all relevant data in the products directory (they shouldn't access the final_data, r_plots or r_stats folders and certainly not delete anything therein)!

Within the products folder (OPERATIONAL/products/) are three subdirectories:

- **data:** contains the merged final data files (one file for each station)
- **plots:** contains the monthly-based post-processed radiation plots as well as pdf version with all months in one file
- **stats:** contains the monthly-based flag statistics files as well as a merged version

Important: In case you want to re-run the post-processing for particular days and stations or if you like to test some new parametrisations, use the test environment (see Section 6).

6. Test setup

The test environment (TEST) should act as a playground for further code testing or re-runs in case of failures in the operational run(s). It is completely independent of the operational code, i.e., changes in this directory don't impact operational results at all. The main code ASRB_data_processing_main_file_TEST.r is located under TEST/r_files/. It is very similar to ASRB_data_processing_main_file_OPERATIONAL.r presented in Section 3. While the operational code only post-processes data of the previous 30 days for all stations which are defined in TEST/ASRB_files/raw_data_input_setup.txt, one can here chose the time period and station(s) for which data should be processed:

```
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#

# Define time period you want to analyse
dates_to_analyse<-seq(as.Date("2015-04-01"), as.Date("2015-04-02") ,by=1) # dates you want to analyse

# define the stations you want to analyse (available are "dav", "wfj_vf" and/or "wfj_gipf")
# load available station names from info array
stn_vec<-c("dav", "wfj_vf", "wfj_gipf")

```

Simply enter the start and end date (in the format **<year>-<month>-<day>**) and change **stn_vec** according to your stations you want to have analysed. The station names are the ones defined in the **stn_name** column in raw_data_input_setup.txt (eg. "wfj_vf" for Weissfluhjoch Versuchsfeld).

If you add an additional station in raw_data_input_setup.txt make sure that its corresponding station parameters are properly set in STN-Parameters_BDuerr_ver_prisco.txt (see Subsection 3.3).

The test environment is not set up to evaluate R-based results with the former IGOR code. For this, use the EVALUATION environment (see Section 4).

In case you tested new features under TEST and like to make them operational, apply the following procedures

- Replace OPERATIONAL/ASRB_files/raw_data_input_setup.txt by TEST/ASRB_files/raw_data_input_setup.txt (only if changes were made in this file!)
- Replace OPERATIONAL/ASRB_files/ STN-Parameters_BDuerr_ver_prisco.txt by TEST/ASRB_files/ STN-Parameters_BDuerr_ver_prisco.txt (only of changes were made in this file!)
- Replace the R functions under OPERATIONAL/r_files/r_functions/ with the corresponding R functions under TEST/r_files/r_functions/ (only for functions which were changed)
- Change ASRB_data_processing_main_file_OPERATIONAL.r according to the changes you made in ASRB_data_processing_main_file_TEST.r. Replacing ASRB_data_processing_main_file_OPERATIONAL.r by ASRB_data_processing_main_file_TEST.r is not recommended as the files have a slightly different setup!

In case you re-run one particular date (for e.g. because the fixing of the raw data file took more than 30 days (see Section 5), only copy/replace the generated final data files from TEST/final_data/ to OPERATIONAL/final data/ and daily flag statistics files from TEST/r_stats/ to OPERATIONAL/r_stats/ . As soon as the operational file (ASRB_data_processing_main_file_OPERATIONAL.r) is run, the transferred data files will be incorporated into the corresponding merged files and copied in the products folder!

7. Further remarks

7.1. Identification of noon dip

An essential part of the pre-processing is the identification of possible noon dips in Sw and Lw. The method applied here is to identify peaks in the Standard deviation of the pyranometer measurements (**dat_002$STD_pyr_in**) before and after solar noon (**solar_noon_ind_calc_corr**) (see R function shadow_width_solar_noon_calc.r). In IGOR, possible (local) peaks are defined by the internal "FindPeak" function. In R no package, identifying peaks in the same way as "FindPeak" does, is available. Thus, the peak identification Routine in R needed to be coded from scratch. In the following this procedure is introduced with help of the code shadow_width_solar_noon_calc.r.

1. Identify **indices** at which a possible peak should be identified
2. Set **peak_threshold**, i.e. the minimal **dat_002$STD_pyr_in** value which should be exceeded for deriving a possible peak

3. Calculate the first derivative (**deriv1**) of **dat_002$STD_pyr_in** by using centred differences
4. Calculate the second derivative (**deriv2**) of **dat_002$STD_pyr_in** by using centred differences
5. Start looping over the pre-defined **indices**
6. A peak is identified if the following conditions are fulfilled:

   a) **dat_002$STD_pyr_in** of the current and next index is larger than the **peak_threshold**
   b) **deriv1** of the current index has an opposite sign than the **deriv1** of the next index --> zero crossing of first derivative
   c) **deriv2** at the zero crossing point location **zero_point_loc** is negative.

7. If for a given index a peak is found, we exit the loop if not, the next index is checked.

In R function sw_corr.r this peak find procedure is used as well to define the fill up method for the noon dip.

```
# **********   Derive shadow width at noon based on the standard deviation   **********
#                    of the upward pointing pyranometer (dat$002$STD_pyr_in)
# ********************************************************************************************

# *** left side of solar noon index (solar_noon_ind_corr) ***
indices<-c((solar_noon_ind_corr-1):(solar_noon_ind_corr-15)) # indices you want to analyse
search_dir<-sign(indices[2]-indices[1]) # search direction; +1 = increasing indices => left to right
#                   -1 = decreasing indices => right to left
peak_threshold<-15 # mean threshold to identify a peak

# derive 1st derivative (centred difference)
deriv1<-rep(NA,length(dat_002$STD_pyr_in))
for (x in c((indices[1]-search_dir*10):(indices[1]-search_dir*1),indices,(tail(indices,n=1)+search_dir*1):
           (tail(indices,n=1)+search_dir*10))){
  deriv1[x]<-(dat_002$STD_pyr_in[x+1*search_dir]-dat_002$STD_pyr_in[x-1*search_dir])/2
}

# derive second derivative (centred difference)
deriv2<-rep(NA,length(dat_002$STD_pyr_in))
for (x in c((indices[1]-search_dir*10):(indices[1]-search_dir*1),indices,(tail(indices,n=1)+search_dir*1):
           (tail(indices,n=1)+search_dir*10))){
  deriv2[x]<-(deriv1[x+1*search_dir]-deriv1[x-1*search_dir])/2
}

# loop over indices and check for peak
for (x in indices){

  if (dat_002$STD_pyr_in[x]>=peak_threshold&dat_002$STD_pyr_in[x+1*search_dir]>=peak_threshold){ # possible peak loc

    # get 1st derivative of possible peak x and x+1
    centred_diff_x<-deriv1[x] # centered difference at possible peak = x
    centred_diff_x1<-deriv1[x+search_dir*1] # centered difference after possible peak = x+1
    # get 2nd derivative of possible peak x and x+1
    centred_diff2_x<-deriv2[x] # centered difference at possible peak = x
    centred_diff2_x1<-deriv2[x+search_dir*1] # centered difference after possible peak = x+1

    if (sign(centred_diff_x)*sign(centred_diff_x1)<0){ #check if zero crossing

      # get location of zero crossing
      slope_deriv1<-((x+search_dir*1)-x)/(centred_diff_x1-centred_diff_x)
      zero_point_loc<- slope_deriv1*(0-centred_diff_x)+x

      # derive second derivative of at zero pint location
      slope_deriv2<-((x+search_dir*1)-x)/(centred_diff2_x1-centred_diff2_x)
      deriv2_zero_point_loc<-1/slope_deriv2*(zero_point_loc-x)+centred_diff2_x

      # Check if second derivative at zero point <0 --> peak
      if (deriv2_zero_point_loc<0){

        # extract indices left and right from zero crossing point
        peak_ind_vec<-c(floor(zero_point_loc),ceiling(zero_point_loc))
        # find max value of these two indices
        final_peak_ind<-which.max(dat_002$STD_pyr_in[peak_ind_vec])
        # assign index corresponding to maximum value
        shad_1_loc_ind<-peak_ind_vec[final_peak_ind]

        break # peak found, exit for loop
      }else{ # second derivative at zero point >=0
        shad_1_loc_ind<-0
      }
    }else{ # no zero crossing ==> no peak
      shad_1_loc_ind<-0
    }

  }else{ # no possible peak located
    shad_1_loc_ind<-0
  }
} # end foor loop
```

8. References

Marty, Christoph. *Surface radiation, cloud forcing and greenhouse effect in the Alps*. Geographisches Institut Eidgenössische Technische Hochschule, 2001.

Dürr, Bruno. *The greenhouse effect in the Alps–by models and observations*. Diss. SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH, 2004.