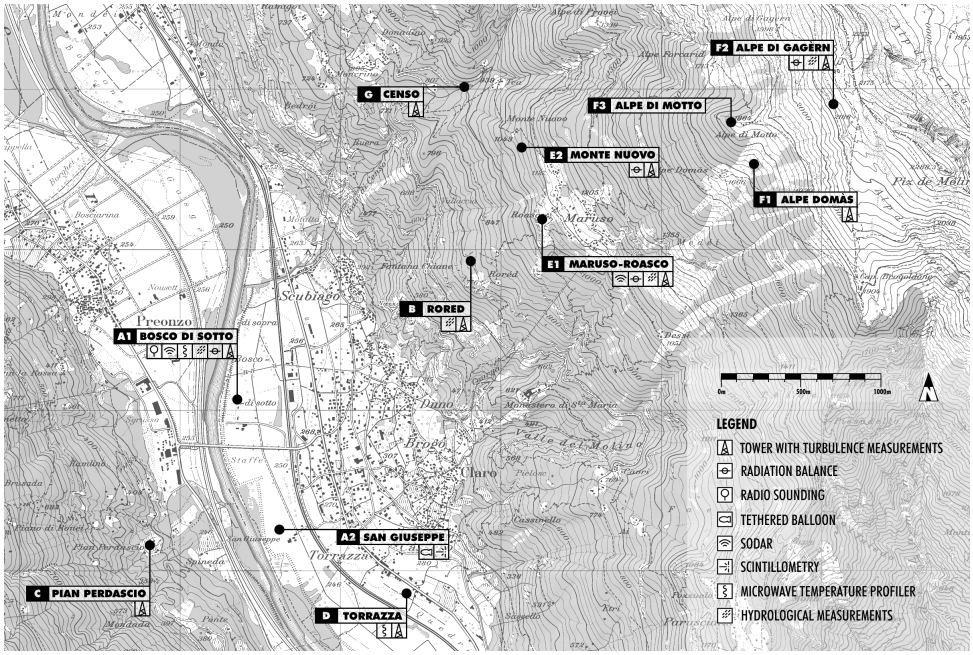
MAP-Riviera: Preparation of data for further analysis

In the Riviera project, there were 9 towers with one or more sonic anemometers each; 4 towers were operated by ETH and 5 by the MCR-Lab (University of Basel). These two groups stored and analyzed their own raw data separately. Each site has an abbreviation which was used project-wide (see the figure below - e.g. A1, B, etc.), however, each group also had their own names for the locations they were responsible for. All the various names, together with the details about the sonics at each location (serial number, frequency, height above the ground, etc.) can be found in the accompanying spreadsheet, as well as in the metadata of all the the produced .nc files.



1 Naming conventions

The ETH group was responsible for 4 towers (A1, B, C, D), and the Basel lab ("MCR") for 5 towers with sonics (E1, E2, F1, F2, G).

Unfortunately, the ETH group also decided to assign their own letter to each of their respective towers, *and* to each of their sonics. The letters they assigned to towers are relevant only when reading the documentation written about the project (i.e. various "README" and "doku" text files), the names are as follows (Project site abbreviation, project site full name; ETH-assigned letter):

A1, Bosco di Sotto; A

B, Rored; B

C, Pian Perdascio; D

D, Torrazza; E

The eight ETH sonics were referred to as A, B, ..., H as well.

The ETH raw data is all stored together in one folder (`data/eth_sonics/Year_DayOfYear`), and the letter in the beginning of each input file denotes the corresponding *sonic* (see the spreadsheet for details).

E.g. a file called `A2681400` corresponds to ETH-sonic A (*not* tower A), day 268, time 14:00.

The Basel lab had a different naming convention; they came up with their own abbreviations for each site (Project site abbreviation, project site full name; Basel-assigned abbreviation):

E1, Maruso-Roasco; ro

E2, Monte Nuovo; mn

F1, Alpe Domas; ad

F2, Alpe di Gagern; ag

G, Ceuso/Kerbtal; kt

However, only at three of these locations (ro, mn, ag) high-resolution sonic data was saved; at ad and kt, there's only 1-minute data available.

The files are stored in separate folders corresponding to these various sites, e.g. `data/mn/rohdaten`. The names of the input files contain the tower abbreviation and the *tower level* of the sonic. Some towers had more levels with instruments than sonics; e.g. the Monte Nuovo tower had 8 levels with instruments, and the sonics were installed at levels 1, 3, 4, 5, 7, 8. These are the levels indicated in the filenames.

2 Input files

The original (input) files (.SLT, .flt, .raw) contain either hourly or half-hourly data. The beginning time of each recording period is indicated in the name of the input file, e.g. `A2131200.SLT`, corresponding to 12:00 on the 213th day of the year 1999, or `MN_N1_1999_219_183000.raw`, corresponding to the 219th day of 1999, 18:30. The input files don't contain any timestamps for the high-frequency measurements; these have to be inferred based on the frequency of the sonics and the length of the time periods.

The various sonics operated at different frequencies, which leads to a different number of data points in various input files:

Gill R2A (ETH): 20.83 Hz, 60 min: 75000 points

Gill R2A (Basel): 20.83 Hz, 30 min: 37500 points

Gill R3A (ETH): 10 Hz, 30 min: 18000 points

Metek: 10 Hz, 30 min: 18000 points

CSAT3: 20 Hz, 30 min: 36000 points

Gill HS: 20 Hz, 30 min: 36000 points

However, it frequently happens that the input files contain a bit fewer/more points than these round numbers (e.g. 35995 or 36005 for the CSAT3 sonics). Usually, this difference is just a few points; if there are points missing, the files are padded with NaNs at the end of the measurement period up to the round number. If there are too many points, the points from the end of the file are simply cut away. This is done in order to keep the frequency of measurements for each sonic constant among the various files.

In some cases, there are more than just a few data points missing. There are two possible causes for this. First, the sonics did not records a certain number of data points at various times during the measurement period. If this is the case, it is impossible to determine *when* during the measurement period this happened since there is no corresponding timestamp. These files, although missing more than just a few points, are also padded with NaNs at the end of the measurement period. The second possibility is that the measurement period started "late" - not at the whole (half-)hour. This is indicated in the name of the input file (e.g. `MN_N1_1999_206_162243.raw`) and is dealt with differently; the files are padded in the beginning (this applies only to the Basel sonics; see the next section).

3 Output files

The filenames of all the resulting .nc files are formatted as
`TowerName_SonicLevel_Timestamp.nc`

1. **Tower name** (e.g. A1, B, ...)

2. **Level of the sonic**, 1 corresponding to the lowest level. E.g. the files for the sonics at Monte Nuovo would start with E2_1, E2_2, ... E2_6. Note that although the input files from Monte Nuovo and Alpe di Gagern indicate levels 1/3/4/5/7/8 and 2/4 respectively, the output files show these as levels 1/2/3/4/5/6 and 1/2.

3. **Timestamp** of the form YYYY_MM_DD_hhmm.
   However, some of the input files don't start at the full (half-)hour, but e.g. at 16:42. If that's the case, the resulting .nc file has a 16:00/16:30 timestamp (for hourly/half-hourly input files, respectively) and is padded with NaNs between the times 16:00/16:30 - 16:42. This applies only to the Basel files - all ETH files start at the full hour and are thus only padded at the end if some points are missing.

All files contain a minimum of 4 variables: u, v, w wind components and sonic (virtual) temperature T. Only three datasets (ETH sonic "H" at the Torrazza location, i.e. D_1, the Metek sonic at Maruso-Roasco, i.e. E2_1, and the Gill HS sonic also at Maruso-Roasco, E2_6) contain the sonic temperature, for all the other datasets the sonic temperature T has to be calculated from the speed of sound c. For this purpose, the following expression is always used:

$$T_{sonic} = \frac{c^2}{402.7}$$

In the .nc files, the sonic temperature is simply referred to as "temperature".

For some of the ETH sonics, there's also "reference data" available, i.e. half-hourly measurements of average pressure, temperature and humidity. If this data is available, it is used to calculate *absolute* temperature:

$$T_{abs} = \frac{T_{sonic}}{1 + 0.32e/p},$$

where e is the water vapor pressure and p is absolute pressure. The calculated values of absolute temperature are then included in the .nc files, together with the half-hourly reference values. Note that the calculation procedure for the absolute temperature follows the original script from the ETH group (written in Fortran, available), and is discussed in more detail below.

Some of the .nc files contain also variables related to humidity; these are discussed below in the sections specific to the location/sonic/krypton. These are only at locations and levels A1_1, A1_3, B_2, E1_2, E2_6.

4 ETH data (Towers A1, B, C, D)

Data from 7 of the sonics (called A-G by ETH) was saved as hourly files and used the same binary data format, .SLT (encoded as int16). Output from all these Gill R2/R2A sonics is already in *calibrated* uvw mode - no need for further calibrations when analyzing the data.

One fo the sonics that was supposed to be used for the Riviera campaign was found to be defective, and so the sonic at Torrazza (H) was replaced by a different one (Gill R3A). This sonic has a different frequency (10 Hz as opposed to 20.83 for the 7 other sonics), and the data was stored in a .flt (float32) format in 30 min intervals. The data from the H sonic also needed no further processing; it already contained u, v, w, T values.

The krypton data comes as instrument voltage, the calibration values are available and recorded.

4.1 Sonics A-G (Gill R2/R2A)

Code: `read_slt.py, utils.py, sonic_metadata.py`

The frequency of these sonics was 20.83 Hz, which corresponds to 75000 data points per hour. The files with fewer than 75000 points were padded *at the end of the file* with NaNs; otherwise, there are no NaN values in the data. As stated previously, this doesn't mean that the measurements were indeed missing at the end; in fact, the measurements could have been missing anywhere during the measurement period.

The high frequency data spans the time period between 30.07.1999 and 28.10.1999. On many days, only measurements for some towers/sonics are available, e.g. on 30.07.1999 it's only the sonics A, B, C - measurements at other sites start later. There were also interruptions at multiple sites lasting from hours to days.

For a *subset* of days, "reference data" is available at low frequencies (30 min intervals), containing mean 30 min temperature and relative humidity at the given tower, and mean 30 min temperature and pressure at Bosco di Sotto. This reference data was originally stored in the `eth_sonics/MAP_alt/MAP Ref Data/ref` folder in a .zip file, sorted in subfolders by sonics. (For analysis, the unzipped folder was moved to the `eth_sonics` folder - this is important only for the path setting in the Python scripts).

To convert the speed of sound to temperature, two methods were used:

1. If no reference data was available, only sonic (virtual) temperature was calculated
2. Whenever reference data was available for a given time period, also absolute temperature was calculated (taking into account humidity, logged in 30-min intervals by other instruments).

Some of the sonics were accompanied by krypton hygrometers, the raw krypton data is an array of voltages. This is discussed in detail in a separate section, however, whenever krypton voltage was recorded, it was also included in the resulting dataset, and water vapor density was calculated from it.

The metadata of the .nc files (datasets) contains info about the sonics (frequency, location, altitude of the tower, sonic height above the ground, serial number, lat/lon coordinates). All this information is also available in the `sonic_metadata.py` module. Furthermore, all variables have their units in their respective variable attributes.

4.1.1 Reference data

The procedure for using reference data for calculating absolute temperature and specific humidity follows the original analysis script (`edi2000v2.f`). Interestingly, for the calculations, the half-hourly values were averaged into hourly values first. Thus, there are also variables with *hourly* values in the .nc files, as well as the half-hourly values. If needed, the procedure could be changed to use only half-hourly data; however, this would require rewriting the code.

30-minute variables:

- Temperature: half-hourly average *not* from the sonics, but from Vaisala temperature and RH probes present at the same level on the tower.
- Relative humidity: measured by the same instrument as temperature.
- Pressure: a pressure sensor was only present at the Bosco di Sotto tower. The pressure at all other towers was calculated taking into account the altitude difference between the reference sensor and the desired instrument (see `utils.py` or the original script `edi2000v2.f` for reference). This is also stated in the variable attributes.

All the 30-min variables have units recorded in their attributes and information saying that they were loaded/calculated from the "reference files".

1-hour variables: Hourly average values vapor pressure, water vapor density and air density (calculated based on the 1-hour averages of temperature, humidity and pressure). These are used to calculate absolute temperature and high-frequency specific humidity (if a krypton is present).

Absolute temperature is calculated whenever reference data is available. Specific humidity is calculated only if both reference data and krypton voltages are available. If there is a krypton and no reference data, the krypton voltage is still saved in the .nc files. If there is no reference data corresponding to the high-resolution file, the 30 min and 1 hour variables are simply not present in the dataset at all.

4.1.2 Kryptons A, C, F

Krypton hygrometers were present at three of the ETH sonics (A1_1, A1_3, B_2). The krypton data was logged as voltage in the range 0-5000 mV. There was no documentation about the calibration except for the constants listed in the `edi2000v2.f` script for each krypton; the values recorded in each file containing voltage are taken from there.

The `edi2000v2.f` script also had subroutines for converting this voltage to *specific humidity*, making use of the 30-min reference data (water vapor density). However, the method was based on Taylor-expanding around a reference voltage value, which is very different from the method suggested in the krypton manual (link to the manual can be found in the Python script `utils.py`, which also contains a bit more detailed documentation than the original analysis script). Specific humidity calculated with this method is called q_ETH in the .nc files, and has units [kg/kg].

Because of the weird analysis method of the ETH group, I decided to use a method of calculating *water vapor density* ρ similar to that used by the MCR lab. The method used for the ETH kryptons differs in one significant fact: there are THREE types of calibration coefficients available for the MCR kryptons, "full range" coefficients for all conditions (see the KH2O manual), and then "dry/wet" ranges. In the MCR code, the full range coefficients are first used to calculate ρ, and based on the results, dry *or* wet coefficients are chosen and ρ is re-calculated (instead of dry/wet, the MCR code uses terms low/high range). In this case we don't have the "full range" coefficients available, and so two arrays of water vapor density are added to the .nc files, one which uses the "dry range" and the second one the "wet range" coefficients (rho_dry, rho_wet). When comparing the data to the 30-min averages from the database, it appears that rho_wet values were used in the database (which is used as a reference for "correctness" of the data processing here); nevertheless, for completeness, also rho_dry values are included in the final files.

The calibration coefficients were taken from the `edi2000v2.f` script - they are not recorded anywhere else in the original data. There are "clean window coefficients" for kryptons A and C, and "scaled window coefficients" for all A, C and F kryptons. Both sets of coefficients can be found in the `sonic_metadata.py` script - the "scaled window" coefficients are commented out. With the "clean window" coefficients, the values for kryptons A and C are correct - again, the "scaled window" coefficients are recorded only for completeness. Since there are no "clean window" coefficients available for the F-krypton, the "scaled" ones were copied into the "clean window" python dictionary. Note that with the use of these coefficients, one does NOT retrieve the values recorded in the 30-min database for this krypton. The values seem to be offset by a constant: the high frequency values are lower than the reference values by 2.3704 g/m^3.

The resulting .nc files contain the original voltage measurements [mV], calibration coefficients (recorded in the attributes of the voltage variable, i.e. `dataset.voltage.attrs`), rho_dry, rho_wet [g/m^3] and if reference data is available, also q_ETH [kg/kg]. For the ETH locations, water vapor density is referred to as "rho" for clarity, in order to not be confused with "q", which stands for specific humidity. For the MCR locations, "q" denotes water vapor density - this can be seen from the units and information, recorded in the attributes of the files.

4.2 Sonic H (Gill R3A)

Code: `read_flt.py, utils.py, sonic_metadata.py`

The data was also stored in the same day-of-the-year folders as for the A-G sonics, but in 30 min intervals, using the binary data format .flt (with float32 encoding).

There was no reference data for this sonic; all data was originally already in m/s or °C (temperature was converted to Kelvin in the resulting .nc files), not in speed of sound as for the other sonics. No other conversions were necessary, but metadata was added to the attributes of the resulting .nc files. As previously, the metadata includes frequency, location (tower), altitude of the tower, sonic height above the ground, serial number, and lat/lon coordinates.

4.2.1 Krypton H

Although in the data there were krypton measurements at this site starting on 20.09.1999, there is no krypton/humidity metadata available. The "humidity" array has values around ∼0.9–1.5, but no units are indicated. When referring to the database of 30-min ("slow") data from the Riviera project, there is no indication of humidity measurements at the Torrazza site. The humidity data was nevertheless included in the final .nc files, just in case that it turns out to be useful, and the variable attributes remind the users about the unknown units etc.

5 MCR data (Towers E1, E2, F1, F2, G)

The MCR data was stored in multiple subfolders, named after the tower abbreviations (ro, mn, ad, ag and kt, corresponding to sites E1, E2, F1, F2 and G); only at E1, E2 and F2 fast data is available.

At location Maruso-Roasco (ro, E1), two sonics of different types were used - Gill R2 and Metek. At location Monte Nuovo (mn, E2), six sonics were used: three Gill R2 sonics, one Gill HS, and two CSAT3 sonics. At Alpe di Gagern (ag, F2), two Gill R2 sonics were used.

Although all high-frequency data was saved in .raw format, data from each sonic type was logged differently and required a different analysis script.

There was no actual documentation provided in the folder with all the Riviera project data, only poorly commented code (`prog` folder) for data processing and analysis. This code barely ever indicated what it was used for, and if it did, the comments were frequently in German and not very helpful.

Upon request, Dr. Roland Vogt from the University of Basel kindly provided some explanations, some of his analysis scripts, and *manufacturer* calibration files for all the Gill R2 sonics. Dr. Vogt also said that Metek and CSAT3 sonics require no calibration, however, he was not sure if the Gill HS sonic (uppermost sonic on the Monte Nuovo tower) was operated in the calibrated or the uncalibrated mode. This information together with the code provided by him was a good starting point to finding out what all the code in the Riviera folder does, and which parts of it are actually useful. The information provided by Dr. Vogt is available in the MCR documentation.

The original code comes mostly from Eva van Gorsel and Andreas Christen ("EVG" and "AC"). Certain analysis scripts come in two versions - one written by each of them, and some scripts were used by both. A combination of their scripts was used to back-trace how the Basel data was analyzed. A selection of their most useful scripts is also saved in the folder holding the Python analysis scripts, simply for reference. Since their scripts contained multiple flags for different analysis options, the processed data was also compared to the available 30-minute averages. This is discussed later in this document (especially for the Gill R2 sonics), together with the data processing options.

As it turns out, all the Gill sonics were operated in "transit count" mode and require calibration. However, only three of them were calibrated using the manufacturer calibration - to the other three "matrix calibration" was applied, based on wind tunnel experiments. The MCR lab applied their own matrix calibration to all the other sonics (Metek, CSAT3, Gill HS) as well. This is apparent when one compares the processes the high-frequency data, takes 30-minute

averages and compares those to the database of 30-min values. When this is done, the "calibrated" values of each sonic fit those with the database way better than the raw, uncalibrated ones. All files based on Basel raw data contain calibrated values. However, if one wishes to produce files *without* applying calibration, all data processing scripts contain a "calibrate" flag which can be set to False. Note that because of the different coordinate systems of the sonics, some of the wind values have to be re-mapped before calibration is applied (e.g. $u \rightarrow -u$ for the Gill HS sonic).

Some the Basel files did not start at the whole 30-min interval, but e.g. at 16:04 rather than 16:00. These files are padded with NaNs in the beginning (i.e. the 4 minutes between 16:00 and 16:04), and the resulting .nc name would indicate the 16:00 time. Otherwise, if the file starts exactly at the 30 min interval but doesn't contain the required number of points, it is padded with NaNs at the end of the time period. This doesn't mean that the data was indeed missing at the end of the file - in fact, any measurements within the 30 min interval could have been missing. However, since there is no timestamp other than that indicating in the name of the file, it is impossible to tell where exactly the measurements are missing. For easy dealing with the data, all the .nc files based on Basel sonics have attributes indicating how many points were added in the beginning and at the end of each file (padded_front, padded_end).

Only two sonics were accompanied by kryptons; the Gill R2 sonic at Maruso-Roasco, and the Gill HS sonic at Monte Nuovo. Details about these are explained in a separate section.

5.1 CSAT3 sonics

Code: `read_raw_csat.py, matrix_calibration.py, sonic_metadata.py`

The encoding of the CSAT3 data is described in detail in the reference manual of the sonics. Each measurement is saved in 10 bytes, some of the bytes have to be parsed into separate bits in order to get the required values. The parsing method in the Python script is equivalent to the one described by the original IDL analysis script provided by Dr. Vogt (`AML_SON_read_csat3.pro`), although it is written slightly differently for ease of understanding.

As stated previously, Dr. Vogt wrote that these sonics don't need calibration, however, the "slow" data processed by Uni Basel was calibrated. Therefore the Python analysis script includes a flag for whether the data should be calibrated (`calibrate = True`), and the produced .nc files hold calibrated data; this is indicated in the dataset attributes (calibration applied =

matrix/none).

5.2 Metek sonic

Code: `read_raw_metek.py, matrix_calibration.py, sonic_metadata.py`

Some of the lines in the ascii files don't contain all the necessary data - in that case, the lines are replaced with NaN values instead. Therefore, if there are NaNs in the E2_1 files which are *not* at the end of the measurement period, these are caused by faulty lines in the input data. The file with most faulty lines has 44 of them, and in total 42 files have more than 10 faulty lines.

The Metek data already contains temperature, and not speed of sound like most of the other sonics.

As for the CSAT3 sonics, Dr. Vogt wrote that the Metek sonic doesn't need calibration, however, the "slow" data processed by Uni Basel was calibrated. Therefore the Python analysis script includes a flag for whether the data should be calibrated (`calibrate = True`), and the produced .nc files hold calibrated data; this is indicated in the dataset attributes (calibration applied = matrix/none).

5.3 Gill R2

Code: `read_raw_gillR2.py, gill_calibration.py, matrix_calibration.py, krypton_calibration.py, sonic_metadata.py`

The data from the Gill R2 sonics was recorded as *transit counts*. From these transit counts wind speed along each (non-orthogonal) axis of the sonic is first calculated, and the orthogonal u, v, w wind components are obtained from these axis speeds based on the sonic geometry. These raw wind components are then calibrated. Subsequently, the speed of sound along each axis is calculated and corrected for cross-wind.

There are three different settings/options which influence the resulting values: sonic path lengths, type of calibration, and applying the cross-wind correction when calculating the speed of sound. For clarity, these three options are recorded in the .nc attributes as `calibration applied`, `pathlengts` and `temperature correction`.

Each measurement corresponds to 18 bytes (or 20 if there's a krypton as well). 2 bytes for beginning of each measurement, 2 bytes for a count flag, 2 bytes for transit count number on each axis (12 bytes in total for 6 axes), then 2 bytes per analog channel (krypton) and finally 2 bytes denoting the end of each measurement.

Sometimes also one of the transducer axes fails, outputting a value of -10000 counts to indicate an invalid measurement. When this happens, the measurement is replaced with NaNs on all axes in the resulting .nc files. Thus, for these sonics, NaNs can occur in the middle of the files as well.

Occasionally, a faulty measurement missing some of the bytes occurs; it can be easily identified by an incorrect number of values between the beginning and the end bytes. The counter is unfortunately not helpful in identifying how many measurements are missing. All the .nc files from the Gill R2 sonics contain an 'info' attribute indicating whether there were some corrupt bytes or not ("Raw data contained no errors" or "Raw data contained corrupt bytes"). If the file was corrupt, a different read-in method was used, cutting out all the faulty lines/measurements.

5.3.1 Path lengths

To convert transit counts into axis velocities and speed of sound, the path length along each of the sonic axes needs to be known. The choice of path lengths affects both the wind speed and the temperature; the effect is way more pronounced for temperature (in the order of a few Kelvin).

The manufacturer "default" path length is 0.149 m for all R2 sonics. However, the Basel lab measured/calculated the path length of each axis of each of these sonics during the "San Vittore intercomparison experiment", where all the sonics were thoroughly tested. Finally, for some of the sonics, the path lengths were measured also during the wind tunnel "wk99" experiments. All the path lengths (default, San Vittore, wind tunnel) can be found in the `GetSoncal_map.pro` script.

Through trial and error I found out that the "wind tunnel" path lengths were *not* used for any of the sonics. The Python script for data pre-processing thus contains only two options for the path lengths (i.e. the `pathlength_type` option): `default` and `sanvittore`. Again, through trial and error, I found out that the raw data from 4 of the R2 sonics (towers E1 and E2) was processed with the San Vittore path lengths, whereas the data from the F2 tower was processed with the default manufacturer path lengths. The settings are easy to change and the data can be easily re-analyzed if needed - the flag in the python script should be very clear.

5.3.2 Calibration

Unlike the other sonics, the data from Gill R2 sonics necessarily *requires* calibration. (The choice of calibration affects only wind speed and has no influence on temperature if *uncalibrated* wind speed is used for temperature correction, see next subsection.)

For each of the Gill R2 sonics there's manufacturer ("Gill") calibration available: this cali-

bration changes the magnitude and the direction of the horizontal wind and then the magnitude of vertical wind. There is one file per sonic with two arrays of values for the horizontal calibration - the names of these files contain the sonic serial number, e.g. `0160rcal.h`. The vertical wind calibration is the same for all Gill R2 sonics, based on the file `Wcal.h`

However, the MCR lab decided that the manufacturer calibration is not that great, and for *three* of the six R2 sonics they used a so-called matrix calibration, where each wind component influences each, as if the original wind vector was multiplied by a matrix. Whenever available, this calibration method was used. Since the matrix calibration files are not available for three of the sonics, those had to calibrated with the manufacturer values.

Both calibrations depend on the angle of the wind with respect to the sonic since the calibration is supposed to correct the effect of the arms of the sonics being in the path of the wind. The manufacturer calibration is based on 1-degree steps, and the matrix calibration on 4-degree steps (i.e. there are 360 and 90 different calibration values, respectively).

This settings can also be easily changed for each sonic in the Python script. For completeness, the option to not calibrate the data is included as well. The calibration type is and optional argument of the `produce_files` function, with the options `calibration = matrix/gill/False`, and the default option is `matrix`.

Matrix-calibrated R2 sonics: 043, 212, 160

Gill-calibrated R2 sonics: 208, 211, 213

5.3.3 Speed of sound correction

The speed of sound (and thus sonic temperature) can be either calculated from raw transit counts without any corrections, or corrected for crosswind. For this corrections, wind speed perpendicular to each sonic axis ("crosswind") needs to be calculated.

$$\text{wspd} = \sqrt{u^2 + v^2 + w^2} \tag{1}$$

$$\text{crosswind} = \sqrt{(\text{wspd})^2 - (\text{axis speed})^2} \tag{2}$$

When correcting for crosswind, an obvious question arises - should the calibrated or the uncalibrated wind speed be used?

If I correctly understood the scripts by AC and EVG (specifically `r2cal.pro`), it looked like they used the already calibrated wind speeds for the correction, however, the reference data indicates that the wind speed correction was calculated from *uncalibrated* wind components. Thus I decided to apply the crosswind correction *before* the wind speed calibration as well.

Again, if this needs to be changed later for whatever reason, the corresponding setting

in the script is `temperature_correction`, which can be set to either `None` (for no crosswind correction), `before_calibration` (default setting, reproducing the reference data) or `after_calibration` (which would reproduce the analysis method of AC/EVG). However, as long as the crosswind correction is applied, the difference between using the calibrated or the uncalibrated wind speed is minuscule.

This correction influences only sonic temperature and has no effect on wind speed.

5.4 Gill HS

Code: `read_raw_gillHS.py`, `matrix_calibration.py`, `krypton_calibration.py`, `sonic_metadata.py`

The Gill HS sonic data format has 15 bytes per measurement. Some flags in the data are stored as raw bytes, the useful bytes (u, v, w, T, krypton voltage) are all stored as int16 (2-byte integers), and scaled by a factor of 0.01 to obtain decimal values.

As stated previously, Dr. Vogt was unsure whether the Gill HS data is calibrated or not, however, the matrix calibration needs to be applied to the data in order to match the reference values. Again, the Python analysis script includes a flag for whether the data should be calibrated (`calibrate = True`), and the produced .nc files hold calibrated data; this is indicated in the dataset attributes (calibration applied = matrix/none).

5.5 Basel Kryptons

The calibration constants for the kryptons were found in the IDL script `KH2OSCAL.pro`. There were clean/scaled window values - only the "clean" ones were ever used in the code. Similarly, there were three sets of values - full range, low range, and high range. In the IDL script, at first, negative or zero voltage values are replaced by a very small value of 0.01 mV. Then, water vapor density ρ was calculated with the full range values. Based on the average of the resulting values, ρ was re-calculated from voltage with either low or high range values, and this array of values was returned - the "full range" values were only ever used for decision making.

There are only very short time windows when the measurements at E1_2 are available and not faulty - the database has only 155 values that are not zero or NaN. Interestingly, when the analysis procedure from the IDL script is reproduced, the water vapor density values at E1_2 are reproduced exactly for some of the time periods when using the calibration coefficients from the IDL script; at other times, the values differ. By trial and error, I realized that the krypton values at E1_2 are matching almost exactly whenever the "high range" calibration values are

used. Thus, I decided to include two functions in the krypton calibration script - one where the decision is made based on the "full range" values (`calibration = decide`), and one where the "high range" values are always used (`calibration = high`). The water vapor densities in the .nc files are based on the "high range" values - this is documented is the variable attributes (`dataset.q.attrs['calibration'] = decide/high`).

At F2_2, the values never match, regardless of which calibration function is used. There seems to be a constant offset between the high-frequency and reference database values: the database values are higher by 3.5301 g/m^3 (when using `calibration = high`).

For the ETH locations, water vapor density is referred to as "rho" for clarity, in order to not be confused with "q", which stands for specific humidity. For the MCR locations, "q" denotes water vapor density - this can be seen from the units and information, recorded in the attributes of the files.

5.6 Fixing the time: CEDT/CET at Maruso-Roasco

In the folder containing the raw data at the Maruso-Roasco location, there were text files (`hinweis.txt`) explaining that the files between days 212 and 214/216 (the end date was unclear to the author of the text file) were recorded with CEDT instead of CET timestamps.

When comparing the high-frequency data to the reference "database" data, there is no time shift. I did not figure out any other way to compare whether the timing of the files is correct, and so the time of the .nc files matches the time of the .raw files - I did not apply any changes/shifts.

6 Some file statistics

ETH Sonics A-G: In total, there are 11704 non-empty files from these sonics. Out of those files, only 4 exceed the expected number of points (75000; all for sonic G; in one case by over 800 points) - these 4 files were cut to 75000 points. All files contain at least 73700 data points. Only 2 files had fewer than 74000 points, and 555 files had fewer than 74900 points.

ETH Sonic H: There are 2463 files, all the data contained exactly 18000 measurements and thus no padding/cutting was needed.

Basel Sonics CSAT3: The frequency of these sonics was 20 Hz, corresponding to 36000 data points per 30 minutes. There are 5811 files associated with these sonics. Altogether, 63 files from these sonics contain less than 50% of the data (i.e. less than 18000 data entries),

and 245 files in total contain less than 90% of data (32400 points). Frequently, the number of data points exceeds 36000 - these are mostly files with 36001 or 36002 points. Only 13 files contained 36004 to 36008 points.

Basel Sonic Metek-USA: There are 2665 half-hourly files for this sonic. Its frequency is 10 Hz, but there are as many as 27 files with over 18010 data points (maximum 18025) saved with `ascii` encoding. As previously, if there are too many data points, these are cut away.

In total, 337 files give Unicode Errors, i.e. some lines in the ascii files are wrongly encoded. All of these files have at most 54 kB (whereas files which are not faulty are over 770 kB), and so these files are simply ignored and not processed at all. This leaves 2328 "useful" files; out of these, only 19 contained fewer than 17000 measurements.

Basel Sonics Gill R2: There are still 64 broken files which contain very few or no usable lines, although the files themselves are not completely empty. 58 of the broken files are from the Maruso-Roasco tower, the other 6 from Monte Nuovo.

There are in total 12704 files from all six of these sonics. Only 14 are longer than 37500 values (at most 37507). From the "non-broken" files, 365 are shorter than 37490 values, and 102 are shorter than 30000.

Basel Sonic Gill HS: The frequency of this sonic was 20 Hz, corresponding to 36000 measurements per 30 minutes. There are 2937 Gill HS files in total. Out of those, 42 are faulty and cannot be processed, and 4 more are very small (less than 100 kB, full files are over 500kB). Most files are 36002-36003 points long - only 54 files are longer and the longest one has 36009 measurements. Only 22 files are shorter than 35900, 12 files are shorter than 35500 points.

7 Checking if data processing was correct

Code: `check_ETH_values.py, check_Basel_values.py`

The original folder containing all the Riviera data contains a database (`data/database`) with 30-min averages for all variables. The data is stored as .asc and .xdr binary files - one per instrument per variable (e.g. wind speed at E2_1). These 30-min arrays have ∼4560 values - missing values are read as NaNs. Some variables are available in both .asc and .xdr formats, but some are available only in one format - if both formats are available, they do indeed store the same values, and so it doesn't matter which one is used.

The raw wind components saved in the database are rotated - thus, u/v/w values cannot be compared. However, it's still possible to compare wind speed - this doesn't change regardless of the coordinate system used. Wind speed is called "MSA" in the database.

Furthermore, sonic temperature "ATA" is compared at each sonic, as well as humidity (water vapor density) "AHA" whenever available.

To check if the values of whichever variable are processed correctly, a location (tower + sonic) is first chosen, some files from this location are selected and loaded into one combined dataset (e.g. files 50-75 from tower E2_5). Loading more than 20-25 files at once requires a lot of memory, but this amount of files is sufficient to see if the data matches. Then the variables are averaged over 30-minute intervals and compared to the database, where the already-processed 30-minute averages are available. For comparison, the numerical values are both printed and plotted.

Only one location can be checked at once. Comment and un-comment variables to be plotted as desired.

These scripts were used to figure out which types of path lengths/calibration/processing settings were used to obtain the values that are now stored in the .nc files. When in doubt, I chose the settings that matched this database most closely. Some variables matched the database perfectly (up to 4th decimal place), some were a bit more off.

- **ETH Sonics A-G:** Temperature is off by a bit (around 0.01 K), but no major offsets. Wind speed matches exactly.
- **ETH Kryptons A, C, F:** A and C match the database exactly when "wet" calibration coefficients for a clean window are used. Krypton F does not match the database - there seems to be a constant offset since there are no clean window calibration coefficients.
- **ETH Sonic H:** Both temperature and wind speed match exactly.
- **Basel Sonics CSAT3:** Both temperature and wind speed match exactly.
- **Basel Sonic Metek-USA:** Temperature matches exactly, wind speed is off by a tiny bit (it's barely visible in the plots).
- **Basel Sonics Gill R2:** Gill-calibrated sonics: Temperature matches exactly. Differences in wind speed are usually up to 0.05 m/s - there are no values for the F2_1 location.

  Matrix-calibrated sonics: temperature matches exactly most of the time, some values are a tiny bit off (by fractions of a degree), the same is true for wind speed (although wind speed is not available for the F2_2 location in the database). Humidity values for the one sonic differ by at most tenths of g/m^3.
- **Basel Sonic Gill HS:** Temperature matches exactly, wind speed is off by a tiny bit (it's barely visible in the plots). Humidity is offset by a constant at all times - my guess is that different calibration coefficients were used.

8 Other random information

- For information about how the sonics were mounted, refer to the sketches in the metadata report; some sonics were mounted at an angle (parallel/perpendicular to the slopes). The ETH sonics were all mounted in the standard way, not at an angle (this information is not written in the metadata report, I found it written in the `MAP_ETHZ_SonicsProcess.pro` script).

- For the Alpe di Gagern location, "fast" data exists only up to day 252 (September 9th). Afterwards, only 1 minute averages were stored.

- Based on the metadata report, it is not 100% clear at what heights the upper sonic at Alpe di Gagern was mounted during the time period with high frequency data, although it was most likely at 11.00 meters (and the bottom one at 3.1 m).

- For the Basel towers, the sonic heights written in the tables in the metadata report do not correspond to those shown in the sketches. This is most likely because the "sonic height" written in the tables corresponds to the central plane of the sonics, whereas the sketches show at what height the booms holding the sonics were mounted. Moreover, the heights written in the tables can also be found in some analysis scripts from Eva van Gorsel. The metadata in the .nc files is based on the information taken from the tables, not from the sketches.

  However, the thesis by Eva van Gorsel mentions even another height: 1.5 m above the ground level for the Metek sonic, E1_1). Later in the thesis (Table 3.1) she writes that the sonic height is 2 m again, which is a bit confusing, but at least the 2 m is consistent with some other documentation.

- Other possibly useful reference data could be all the `*_TURB.dat` files in the `data/mdc` folder - however, I have not checked if this is indeed the processed high-frequency data, it's just a suspicion.

  The `mdc.xls` file has some of the processed sonic data, (among the the 30-minute values from other instruments). Since this .xls file only has values for a few of the sonics, I also haven't checked whether the processed high-frequency data agrees with this table.

  The ag, mn and ro folders contain also sonic_30min and tagesfiles folders, which possibly contain data which could be used for checking if the data processing was correct.

- Thus, the only useful folders in the `MAP/data` folder are: ag, !database, eth_sonics, MapCal, (possibly) mdc, mn, ro. The `MAP/prog` folder contains the IDL scripts from the MCR lab, however, most of them are not useful.