https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

Open-source tools for processing opportunistic rainfall sensor data: An overview of existing tools and the new OpenSense software packages *poligrain*, *pypwsqc* and *mergeplg*

Christian Chwala[1], Aart Overeem[2], Erlend Øydvin[3], Louise Petersson Wårdh[4,5], Jochen Seidel[6], Maximilian Graf[7], Bas Walraven[8], Elia Covi[9], Hai Victor Habi[10], Martin Fencl[11], Lotte de Vos[2], Filippo Giannetti[12], Amy Green[13], Tess O'Hara[13], Nico Blettner[1], Tom Keel[14], Georges Schutz[15], Abbas El Hachem[16], Nicholas Illich[6], Julius Polz[1], Taoufiq Shit[11], Lukáš Kaleta[17], Damaris Zulkarnaen[6], and Vojtěch Bareš[11]

[1]Institute of Meteorology and Climate Research (IMK-IFU), Karlsruhe Institute of Technology, Garmisch-Partenkirchen, Germany
[2]R&D Observations and Data Technology, Royal Netherlands Meteorological Institute, Utrechtseweg 297, 3731 GA De Bilt, the Netherlands
[3]Faculty of Science and Technology, Norwegian University of Life Sciences, Ås, Norway
[4]Division of Water Resources Engineering, Faculty of Engineering, Lund University, P.O. Box 118, 22100 Lund, Sweden
[5]Swedish Meteorological and Hydrological Institute (SMHI), Folkborgsvägen 17, Norrköping SE-601 76, Sweden
[6]Institute for Modelling Hydraulic and Environmental Systems, University of Stuttgart, 70569 Stuttgart, Germany
[7]Department of Hydrometeorology, Deutscher Wetterdienst, Offenbach, Germany
[8]Department of Water Management, Faculty of Civil Engineering and Geosciences, Delft University of Technology, Delft, Netherlands
[9]Hydro-Meteo-Climate Structure, Arpae Emilia-Romagna, Bologna, Italy
[10]School of Electrical and Computer Engineering,Tel Aviv University, Tel Aviv, Israel
[11]Department of Hydraulics and Hydrology, Czech Technical University in Prague, Prague, Czech Republic
[12]Department of Information Engineering, University of Pisa, Via G. Caruso 16, 56122 Pisa, Italy
[13]School of Engineering, Cassie Building, Newcastle University, Newcastle upon Tyne, NE1 7RU United Kingdom
[14]UK Centre for Ecology & Hydrology (UKCEH), Wallingford, UK
[15]RTC4Water s. à r. l., Roeser, Luxembourg
[16]Federal Waterways Engineering and Research Institute, Karlsruhe, Germany
[17]Department of Telecommunications, Brno University of Technology, Brno, Czech Republic

Correspondence: Christian Chwala (christian.chwala@kit.edu)

Abstract. Opportunistic sensors (OS), such as personal weather stations (PWSs), commercial microwave links (CMLs), and satellite microwave links (SMLs) can provide detailed rainfall information near the Earth's surface, complementary to information from ground-based weather radars and rain gauges from official networks. Due to their opportunistic nature, their data requires dedicated processing and quality control. A variety of open-source tools for these tasks exist. Their interoperability is low, however, and implementation details are too heterogeneous to allow joint community-driven development. Within the COST Action OpenSense (Opportunistic Precipitation Sensing Network) we have set out to improve this situation. Here we first summarize the state of the preexisting open-source tools and then describe the new OpenSense software ecosystem that we built. Specific attention is given to the three new software packages developed within OpenSense: *poligrain*, which simplifies common tasks for working with point, line and gridded sensor data; *pypwsqc*, which provides several quality control algorithms

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

for PWS data, that were previously available as separate packages; *mergeplg*, which contains merging methods for point, line and grid data. These packages, with *poligrain* as the foundation, form the OpenSense software ecosystem for which we show an example use case. This software ecosystem shall serve as a community platform for reproducible research and operational integration of opportunistic sensor data for rainfall estimation.

1 Introduction

1.1 Background and motivation

Accurate rainfall information with high spatial and temporal resolution is vital for many applications. Timely rainfall information and communication through rainfall nowcasting, flash flood forecasting, and/or landslide risk prediction are needed to avoid loss of life and property. Historical rainfall information is needed for climate monitoring, evaluating weather and climate model output, and evaluating extreme rainfall events and their hydrological impact, to name a few. Hence, it helps to improve hydrological, weather and climate models.

The density of official rain gauge networks is frequently too low to capture the high spatial variability of rainfall, especially at the (sub-)hourly timescale and in low-income countries. Satellite precipitation products provide almost global coverage, and are virtually the only source of information above oceans, but have serious limitations concerning accuracy and spatiotemporal resolution. Ground-based weather radar precipitation products have high spatiotemporal resolution, and low latency, but typically need gauge adjustment to achieve a reasonable accuracy and coverage is especially poor over Africa and large parts of Asia (Overeem et al., 2024b). To this respect, both amateur weather monitoring devices and existing operational wireless communication systems can be used to help fill this gap and complement the official observations from conventional instrumentation such as rain gauges, radars and satellites. These gap-filling systems are called opportunistic sensors (OS) since they have been designed for either amateur use or for another purpose, such as wireless communication services, but all of them could be used for weather monitoring and associated applications. Obviously, due to their opportunistic nature, quality control and data processing are crucial steps.

1.2 Achievements of CML, SML and PWS rainfall observation

The usage of commercial microwave links (CMLs), which form a large portion of the cellular network backhaul, for rainfall estimation was demonstrated for the first time almost 20 years ago (Messer et al., 2006; Leijnse et al., 2007). Since then, the maturity of this technique has significantly increased so that country-wide rainfall estimation in real-time is possible and meteorological services, at least in Europe, have been prototyping the integration of CML data into their operational radar-adjustment procedure (Wenzel et al., 2023). Feasibility of CML rainfall estimation in low-income countries, where traditional

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

observation networks are sparse, has also been shown on a region-wide basis (Walraven et al., 2024) and with high-resolution rainfall maps for cities (Djibo et al., 2023). With an estimated 5 million CMLs deployed globally in telecommunications networks (Ericsson, 2018), they represent one of the most promising gap-filling technologies for rainfall estimation, especially in low-income countries.

Similar to CMLs, satellite microwave links (SMLs) are part of microwave-based communication infrastructure. SMLs provide a broadband or broadcast connection between a satellite and a receiving terminal on the ground. Rainfall retrieval with SMLs is similar to CMLs, but they provide rainfall estimates along a slanted path between the Earth's surface up to a suitably-estimated height in the troposphere where the frozen particles start to melt (Giannetti and Reggiannini, 2021). Currently, commercial SML-based weather services are available across southern France (https://www.hd-rain.com/) and the Liguria region in Italy (https://www.artys.it).

In contrast to CMLs and SML, personal weather stations (PWS) have been designed to measure meteorological quantities, albeit not with the accuracy of professional devices and professional maintenance. PWSs provide crowd-sourced rain gauge data, while connected through the internet. Coverage with PWS is typically good in high-income countries, reaching up to ten times higher density than official gauge networks. They remain substantially sparser in low- and mid-income countries (Olsson et al., 2025). PWS data has proven to provide valuable high-resolution rainfall information (de Vos et al., 2019a; Graf et al., 2021; O'Hara et al., 2023) and has been shown to improve weather radar data on the European scale (Overeem et al., 2024a), with data quality control still being a challenge.

1.3 Current limitations of CML, SML and PWS rainfall estimation

Undoubtedly, rainfall estimation with OS data has improved significantly, however, operational (real-time) applications are still limited (Olsson et al., 2025) due to methodological and technological reasons. From a methodological point of view, the use of OS data poses several challenges primarily due to the lack of control over the sensor installation and maintenance, which often lead to substantial measurement errors. In the case of CMLs and SMLs, further complication arises from the indirect nature of the observation, i.e. attenuation rather than rainfall is measured, as well as from the use of hardware not originally designed for meteorological purposes. An additional challenge is the path-integrated nature of the measurements. From a technological perspective, the absence of standardized data formats and a lack of common open-source tools limit the operational use of OS data and make algorithm comparison studies labor-intensive (Garcia-Marti et al., 2023). The final major barrier to broader utilization of OS data is organizational and commercial: raw OS data are typically owned by private entities.

Overcoming the methodological and technological challenges can significantly enhance the quality of OS rainfall observations, reduce the costs associated with design and operation of data processing systems, and ultimately help address the organizational and commercial barriers. FAIR open-source software, maintained by a diverse community of stakeholders, is one promising pathway to transition OS observations from research into operational use and scale them from regional case studies to global applications. Already a lot of software produced in academia is open-source and findable (F) as well as accessible (A) e.g. on Github. But since software development frequently happens in parallel in different groups, interoperability

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

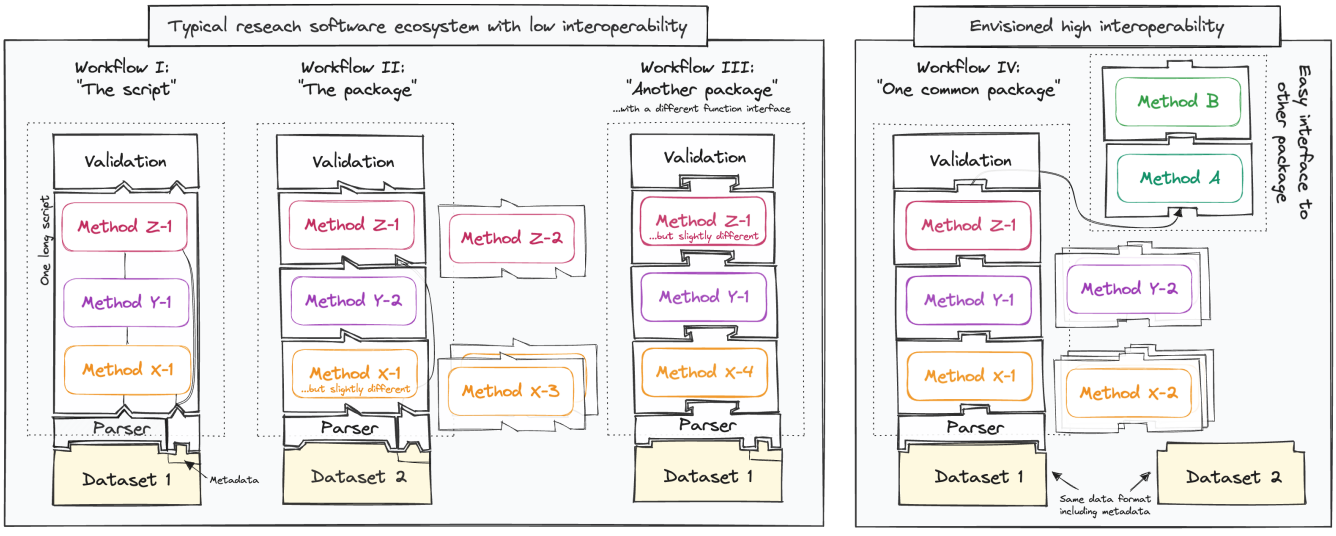

Figure 1. Sketch of a typical ecosystem of research software, on the left. For the shown case, three different workflows have been implemented to do the same task, requiring processing steps X, Y and Z, for which different methods exist, e.g. method X-1 and X-2. Three problems are visible here: 1) Loading and passing data (and metadata) to the methods is handled differently. 2) Methods can often not be exchanged in modular way. 3) Multiple, potentially slightly different, implementations of the same method exist. With the envisioned software ecosystem, shown on the right, these problems can be eliminated, by having a standard data format (including metadata), a simple common function interface (automatically carrying forward the metadata) that also fits with other associated software packages and which is used for implementing all available methods.

(I) is often severely lacking, see Figure 1. This hinders comparison and benchmarking of methods, which in turn hinders the choice of methods for operationalization and making informed decision about future research directions.

1.4 The mission of OpenSense

The COST (European Cooperation in Science and Technology) Action OpenSense (November 2021 - October 2025) has set out with the goal to increase the uptake of OS in rainfall observation and to set the basis for operational usage of OS data. One main component of OpenSense was to standardize, homogenize and inter-compare the methods and software tools used in the field of opportunistic sensing of rainfall. Previous efforts included the development of a joint data format and the open-sourcing of individual implementations of published methods. However, it is only through the coordinated work plan and incentives provided by OpenSense that a solid foundation for continued progress has been established. OpenSense works on the usage of personal weather stations (PWSs), commercial microwave links (CMLs) and satellite microwave links (SMLs), hence, we focus on the open-source tools for these sensors. One of our main achievements has been streamlining a fragmented collection of datasets, methods and software into a new tidy ecosystem (which we describe in this paper) to increase the interoperability of method implementations, as shown in Figure 1.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

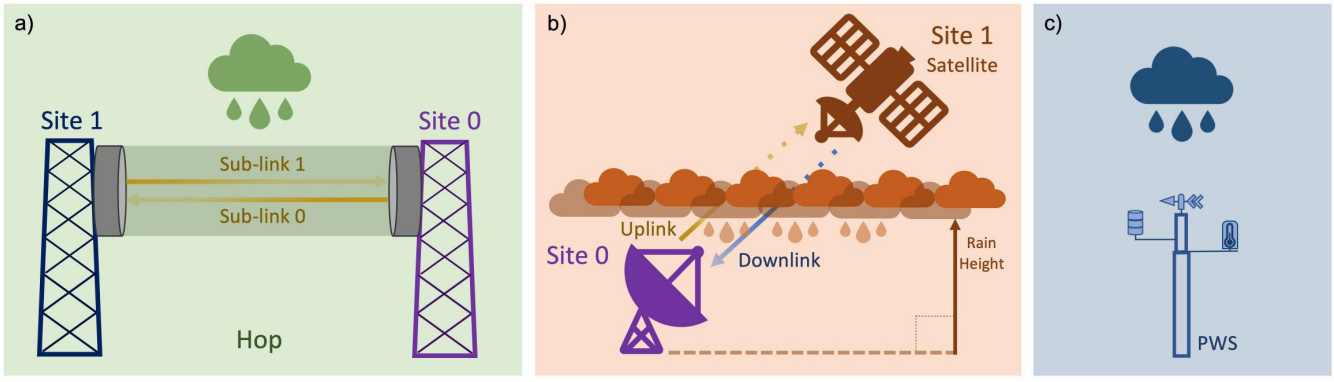

Figure 2. A conceptual overview of rainfall observation with CML, SML and PWS systems. Source: Fencl et al. (2023)

In this paper, we first give a brief overview of the concept and processing steps of CMLs, PWSs and SMLs data for OS rainfall observation in section 2 to provide the required background for understanding the functionalities of the presented software packages. In section 3 we review the state of the open-source software packages for CML, PWS and SML processing that were developed before OpenSense. Section 4 explains the reasoning for building a new software ecosystem and describes the three new OpenSense software packages *poligrain*, *pypwsqc* and *mergeplg*. In addition, a use case of these packages is shown.

2 A brief overview of CML, SML and PWS and the required data processing

This section briefly introduces the concept of OS rainfall observation with CML, SML and PWS, which is also depicted in Figure 2. To better understand the similarities and differences of the existing software packages and their processing methods, we give an overview of the typical processing steps for each of these sensors.

2.1 Processing of CML data

Rainfall estimation from CMLs typically starts with raw data of received signal level (RSL) and transmitted signal level (TSL). Two common types of these data are used: 1) The 15-minute records of minimum and maximum RSL and TSL (but also mean and instantaneous occur) provided by the network management systems of CML network operators (van der Valk et al., 2024). 2) 1-minute or 10-second instantaneously recorded RSL and TSL values either provided by dedicated data collection software stemming from joint projects of CML operators and researchers (Chwala et al., 2016) or by a new generation of network monitoring tools developed by CML vendors. It shall be noted that TSL data might not always be available. If CMLs are configured to have constant TSL it is also not required since the rain rate can be estimated from the RSL alone in that case.

Before the actual processing of the data can be done, typically the time series of RSL and TSL need to be combined with the correct metadata (location of CML towers, path length, frequency and polarization) because most commonly the metadata is

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

provided separately. This combination is not always straightforward since there are no standards for file structure and naming conventions. Once data and metadata are combined, the rainfall retrieval process can be started.

The commonly applied processing steps to go from raw CML data to rainfall estimates are listed in the following. Note that table 2 in section 3.1 lists the available methods of each described software package grouped into these processing steps.

1. Filtering of outlier values, too noisy time series or of CMLs with specific metadata: In this step, clear outliers, e.g. signal spikes, are removed by simple filters. Additionally, the noisiness of the time series can be assessed, e.g. via rolling-window statistics to neglect time series that show too high or erratic signal fluctuations, see e.g. (Overeem et al., 2016a; Blettner et al., 2023). Also, metadata should be checked, for example for too long CML lengths or for frequency-length combinations that are not suitable for rainfall retrieval. Note that the detection of outliers can also be done after step 3 or step 5.

2. Detection of rain events, so called wet-dry classification : This step is required because CML attenuation data shows fluctuations also during dry periods. Thus, the periods with rain-induced attenuation have to be separated from the dry periods, see e.g. (Overeem et al., 2011; Chwala et al., 2012). This is in particular challenging for low rain rates. Depending on the sensitivity of the individual CML to rainfall, there is a lower threshold for rain event detection which is determined by path length, frequency and the quantization of the recorded signal levels.

3. Setting the baseline: In this step the zero-level of the signal, called the baseline, during a rain event is set. From this, the rain-induced attenuation, which is the difference between the baseline and the recorded signal level, is calculated.

4. Estimation of and correction for wet antenna attenuation: The presence of water drops on the antenna during and after rain events (and also stemming from dew during nighttime) lead to additional attenuation, termed wet antenna attenuation (WAA), see e.g. (Schleiss et al., 2013). It has to be estimated and subtracted from the rain-induced attenuation.

5. Calculating path-averaged rainfall estimates from attenuation : From the time series of the rain-induced path-integrated attenuation the path-averaged rain rate can be derived via a simple power law, with the parameters mainly depending on CML frequency and polarization (Olsen et al., 1978). It shall be noted, that the dependency of the power law parameters on the rain drop size distribution is only neglectable for CML frequencies in the range of approximately 15 GHz to 40 GHz. With going to lower or higher frequencies the DSD-dependence increases and thus optimized power law parameters for different rainfall regimes should be considered.

Once rainfall estimates are available for each CML, the data can be used for spatial reconstruction of rainfall fields (Zinevich et al., 2009; Overeem et al., 2013; Graf et al., 2020). Often this is done with the simplification of representing the CML path-averaged rainfall estimates as point in the middle of the path so that common geostatistical spatial interpolation or reconstruction method, developed for point data, can be used. But more sophisticated methods are available for either explicitly considering the path-averaged character of the CML rainfall estimates (Goldshtein et al., 2009; Blettner et al., 2022) or employing a disaggregation procedure before the reconstruction (Fencl and Schleiss, 2025).

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

2.2 Processing of SML data

Similar to CMLs, the rainfall estimation from satellite microwave links (SML) data is also based on path attenuation (Barthès and Mallet, 2013) and the applied steps are similar to the ones shown above for CMLs. However, in the SML case, a constant (though unknown) value is assumed for the TSL. Furthermore, the raw RSL data can be obtained either from diagnostic measurements in commercial satellite broadcast receivers, or by purposely developed devices. In either case, the values of the RSL time series are taken at a rate of 1 or 2 samples per minute and, as in CMLs, typically consist of the values of the received power in dBm; however, commercial satellite receivers also provide the estimates of signal-to-noise-ratio (SNR), i.e. the ratio in dB between the power of the received signal and the power of the channel noise affecting the signal (Giannetti and Reggiannini, 2021). It is worthwhile to remark that satellite broadcast signals can be received anywhere within the satellite footprint and, unlike with CMLs, where data belong to the wireless network operator, in this case data belong to the operator/owner of the ground receiver. In particular, the RSL can be collected without any authorization from the broadcaster or the satellite operator, or any fee to be paid.

Also for SML, before the actual processing of the data can be done, typically the time series of RSL need to be complemented with the associated metadata (location of SML receivers, satellite longitude, signal format, frequency and polarization) since also in this case there are no standards for file structure and naming conventions. Furthermore, in the SML case, the link geometry consists of a slanted path crossing the troposphere and therefore some ancillary information about weather are necessary for the rainfall retrieval process, such as the 0 °C isotherm height and/or the rain height, and the melting layer (ML) thickness.

The commonly applied steps to go from raw SML data to rainfall estimates are the following:

1. Data quality control: It is a fundamental step prior to the processing of raw RSL measurements collected by SMLs (which, like any other OS system, are not optimized for weather monitoring) and includes: i) a search for gaps in the time series to be filled by interpolation; ii) the identification and filtering of outliers, e.g., glitches or step-like variations in signal power due to satellite operator maneuvers or receiver blinding during sun transits behind the satellite; iii) a cross-check of raw RSL data collected by different links (when available), consisting of either different frequency channels of the same satellite downlink or from different satellite downlinks.

2. Detection of rain events, so called wet-dry classification: In SML, the RSL from GEO satellites typically exhibits 24h fluctuations due to the residual inclination of the satellite orbit w.r.t. the equatorial plane and hence, wet/dry classification methods for SML shall take into account this impairment. Kalman filtering can be effectively used as a general method to monitor the RSL and track rapid fluctuations due to rain events (Giannetti et al., 2019). Alternatively, Machine Learning (ML) and other AI-based methods can be used for wet/dry classification.

3. Setting the baseline: TSL is not available in the case of SMLs. Hence, rain attenuation is usually evaluated from the current value of the wet RSL and a baseline value for dry conditions, e.g., based on the latest dry RSL readings before the rain or from previously-stored dry RSL readings.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

4. Estimation of and correction for wet antenna attenuation: Differently from CML, there is no solid experimental evidence of appreciable WAA effect in SML setups for Ku-band satellite TV reception featuring offset parabolic reflectors at mid- and low-latitudes. This is possibly due to the fact that offset reflectors exhibit a more vertical surface when compared to center-fed designs, which are instead rarely used for satellite TV reception.

5. Calculating path-averaged rainfall estimates from attenuation: The rain-induced attenuation is derived from RSL and the baseline level. The attenuation calculation procedure is different depending on whether the RSL represents the received power or the SNR. Then, given the satellite elevation angle, the length of the slanted wet path is calculated, from the ground receiver up to the location-dependent value of the rain height, which can be derived from the 0 °C isotherm height. The latter can be a fixed statistical value, such as a daily or yearly mean, or a short-term forecast by models, or a gradient-based extrapolation of low-altitude temperature measurements (Angeloni et al., 2023). Then, rainfall intensity can be retrieved from specific rain attenuation via the k-R power-law, under the assumption that the rainfall intensity is uniform over the whole slant path. The accuracy of the rainfall estimates can be improved via accounting for enhanced attenuation in the melting layer, which requires additional information about the layer thickness, e.g. from fixed statistical values or from radar bright band measurement. In addition dual-frequency information can potentially improve rainfall estimates (Gelbart et al., 2025).

2.3 Processing of PWS data

PWS data are very different from the data that is provided by CMLs and SMLs. Instead of active remote sensing, they directly provide rainfall sums or rainfall rates per time step derived from low-cost rain gauges. The challenge thus is not the transformation from raw data to rainfall information, but rather quality controlling the provided data. This might sound simple, but due to the limited mechanical precision of the hardware, the potential severe lack of maintenance, issues with the setup, such as problematic installation close to or below obstacles, connectivity or power supply issues, the raw rainfall data from PWS typically has very heterogeneous data quality (de Vos et al., 2017; Bárdossy et al., 2021). Hence, quality control (QC) algorithms are necessary to discard erroneous observations and for bias correction.

A first step that should be done before applying any of the open-source quality control (QC) packages, is to check for duplicate coordinates of PWS stations, indicating incorrect location information. Incorrect coordinates may lead to deviations when applying algorithms that compare observations from different locations, but obviously also when the resulting rainfall estimates are employed. Other typical preprocessing concerns combining the metadata and data files from individual stations into a single file for a chosen period and area. In addition, data may have somewhat irregular time intervals and are typically interpolated to 5-min time series before QC.

The common methods that are applied to PWS data can be grouped into the following three categories. Table 3 in section 3.2 lists the available methods of each described software package grouped into these categories:

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

1. Intra-station (sanity) checks on single time series. Typically identifies outliers by employing climate indices or thresholds or by a gross error check that discards observations below 0 mm and above a physically not acceptable upper limit (Ośródka et al., 2022) or above the measurement range (Overeem et al., 2024a).

2. Inter-station (buddy or neighbour) checks to detect and remove suspicious observations. The basic principle of inter-station checks is spatial rainfall correlation in space (van de Beek et al., 2012). Inter-station checks can be done with other PWS observations within, for instance, a 10-km radius (de Vos et al., 2019a) or by comparing PWS rain gauge data with those from other sensors, for instance when deviations in distributions are found with respect to those from professional rain gauges (Bárdossy et al., 2021). Others involve radar data from the same location in algorithms that are not open source: the RainGaugeQC algorithm developed for government gauges from Ośródka et al. (2022) and the radar QC by Overeem et al. (2024a).

3. Bias correction for systematic deviations, such as undercatch due to turbulence. Bárdossy et al. (2021) apply quantile mapping with trustworthy reference rain gauge data to correct for intensity dependent biases. Others obtain a default bias correction by comparing PWS rain gauge accumulations with gauge-adjusted radar accumulations from a previous period and apply it to PWS data from the current period, followed by a dynamic bias correction per PWS location and time interval employing neighbouring PWSs (de Vos et al., 2019a).

Table 1. Overview of the software packages covered in section 3 and 4.

Name	Source	Language	Packaging	Unit tests	License
CML processing packages					
RAINLINK	https://github.com/overeem11/RAINLINK	R	none	no	GPL v3
pycomlink	https://github.com/pycomlink/pycomlink	Python	pypi, conda-forge	yes	BSD-3-clause
PyNNcml	https://github.com/haihabi/PyNNcml	Python	pypi	yes	MIT
Rcmlrain	https://github.com/fenclmar/Rcmlrain	R	none	yes	MIT
PWS processing packages					
pws-pyqc	https://github.com/AbbasElhachem/pws-pyqc	Python	none	no	GPL v3
PWSQC	https://github.com/LotteEdeVos/PWSQC	R	none	no	cc-by-sa 4.0
GSDR-QC	https://github.com/cntdeathter/Intense-qc	Python	pypi	yes	None
SML processing packages					
No open-source software packages yet					
New OPENSENSE software packages					
poligrain	https://github.com/OpenSenseAction/poligrain	Python	pypi, conda-forge	yes	BSD-3-clause
pywspqc	https://github.com/OpenSenseAction/pywspqc	Python	pypi	yes	BSD-3-clause
mergepclg	https://github.com/OpenSenseAction/mergepclg	Python	pypi	yes	BSD-3-clause

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

3 Overview of open-source tools for processing of OS rainfall sensor data

Several software packages or scripts for processing of CML, SML and PWS data have been developed over time. Some of these tools are a modular collection of methods while others are focused on the implementation of one specific workflow only. Some tools contain an overlapping subset of methods and some provide unique implementations. All packages covered here have in common that they are open-source and findable on the internet.

Here we present the software packages that were dedicatedly developed for processing of CML and PWS data. Since there is no open-source software package for SML data processing yet, we refer to small existing open-source code snippets from that domain. Table 1 gives an overview of the software packages that are covered in this section as well as the new OpenSense software packages that are introduced later in section 4.

At the end of this section we also present a collection of software tools that can be executed online to explore the available methods and we briefly summarize software packages that are related to OS rainfall data processing.

3.1 CML data processing software packages

In the following we give a brief introduction of the four CML processing software packages. A detailed list of the processing methods that they provide is given in Table 2.

3.1.1 RAINLINK

The R package RAINLINK has been developed at Wageningen University & Research and the Royal Netherlands Meteorological Institute during the period from 2010-2016, and has been published by Overeem et al. (2016a). It has been tested on 2.5 years of 15-minute minimum and maximum RSL data from the Netherlands (Overeem et al., 2016b), but has also been applied to data from, e.g., a tropical climate (Overeem et al., 2021), or to Dutch instantaneous data (de Vos et al., 2019b) (De Vos et al., 2019). It was originally meant as a package focusing on one method to go from CML data obtained from the network management system to interpolated rainfall maps, which could be visualized in the earliest versions. In the current version the visualization has been removed and taken over by the Python package MapRAINLINK. The main methodological contribution of RAINLINK is the wet-dry classification, called the nearby link approach, and the outlier filter. This makes use of the fact that rainfall is correlated in space. For a chosen sub-link, an interval is classified as wet if at least 50% of the neighboring sub-links have a joint decrease in received signal level. In addition, an interval is discarded if the cumulative difference between the sub-link's specific attenuation and that of the surrounding sub-links becomes too large.

3.1.2 pycomlink

This software package was initially developed at KIT in the year 2014 based on MATLAB code which was used for the first CML data processing in Germany (Chwala et al., 2012). From the beginning it was meant to be a collection of methods for the individual processing steps, including at least two different methods for wet-dry classification, setting the baseline, WAA correction and spatial interpolation. For the first years it has mainly been used at KIT and served as the basis for the country-

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

Table 2. Overview of available methods for different CML processing steps in the four software packages presented in section 3.1.

CML data processing method	RAINLINK	pycomlink	PyNNcml	Rcmlrain
Raw data filter				
Outlier filter	x			x
Wet-dry classification				
Nearby link approach (Overeem et al., 2016a)	x	x		
Rolling standard deviation (Schleiss and Berne, 2010)		x	x	x
Short-term Fourier transform - SFTF (Chwala et al., 2012)		x		
Convolutional neural network - 1D CNN (Polz et al., 2020)		x		
Multilayer perceptron - MLP (Øydvin et al., 2024)		x		
Long short-term memory neural network - LSTM (Habi and Messer, 2018)			x	
Gated recurrent unit neural network - GRU (Habi and Messer, 2019)			x	
Baseline estimation				
Linear		x		x
Constant		x	x	
Rolling minimum			x	
Rolling quantile				x
Rolling median (where min & max RSL are first averaged for min/max sampling)	x			
WAA				
Constant	x		x	x
Time-dependent model with exponential rise and decay (Schleiss et al., 2013)		x		x
Depending on rain rate, CML frequency and antenna properties (Leijnse et al., 2008)		x		x
Depending on rain rate but no frequency dependence (Pastorek et al., 2022)		x		x
Single- and dual-frequency model (Kharadly and Ross, 2001)				x
$k - R$ relation				
Leijnse (2007) based on Dutch DSD data	x			
ITU (2003)		x	x	x
RNN rain estimation (Habi and Messer, 2020)			x	
Min-max power law (Ostrometzky et al., 2016)			x	
Spatial interpolation				
Inverse distance weighting - IDW	x	x	x	
Ordinary Kriging (using R function krige from gstat or using PyKrige package)	x	x		
The "GMZ" method (Goldshtein et al., 2009)			x	x

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

wide processing and analysis of 4000 CMLs in Germany (Graf et al., 2020). With the inclusion of a growing number of methods for wet-dry classification, WAA estimation and spatial interpolation, it has seen a broader adaption in recent years and was used e.g. for generating high-resolution rainfall maps for the city of Ouagadougou in West Africa (Djibo et al., 2023) and for processing the CML data which were used for snow event detection based on radar and CMLs in Norway (Øydvin et al., 2025). Within OpenSense, it has evolved into the central CML processing package because it now includes implementations of the most widely used methods, in particular an implementation of RAINLINK's nearby link approach and outlier filter in Python. Recently, as part of work within OpenSense, a simple interface for using Deep Learning methods from pytorch and tensorflow was added.

Over the years pycomlink has seen three major revisions. The main reason for these revisions was the usage of too rigid data structures (based on own classes) in the early versions which made extension and maintenance cumbersome. With the current version 0.3 (since 2021) the `xarray.Dataset` was adopted as data structure and all processing functions use them as input and output. This approach was then also taken as a blueprint for data handling and processing in OpenSense and is a key concept of our new foundational software package *poligrain* (see section 4.2).

3.1.3 Rcmlrain

Rcmlrain is a set of R language functions that was gradually developed at Czech Technical University in Prague (CTU) since 2013 based on CML data processing in the Czech Republic. A cleaned version of the original code has been accessible through GitHub since 2017 when used during a CML processing workshop organized within the 17th International Conference on Urban Drainage in Prague (10th September 2017). The methods implemented in the code were designed primarily for processing and manipulating CML data from the SNMP-based DAQ systems providing instantaneous attenuation data with high sampling rate (1-minute or sub-minute) and it was meant for intercomparison of different state-of-the-art methods with new methods developed at CTU. It provides a set of functions for processing raw CML data including basic quality control procedures and time series manipulation, functions for rainfall retrieval (baseline separation, WAA, attenuation-rainfall conversion), CML-gauge-adjustment, and rainfall spatial reconstruction. The software was used e.g. for CML adjustment (Fencl et al., 2017), E-band CML data processing (Fencl et al., 2020), or WAA correction (Pastorek et al., 2022) and it is continuously being updated.

3.1.4 PyNNCML

PyNNCML is a software package that was initially developed at Tel Aviv University in the year 2020 using Python and based on the PyTorch framework because the initial goal was to demonstrate the ability of CML processing using deep neural networks. A major benefit of building on PyTorch is to accelerate the processing of data using the graphics processing unit (GPU) as well as easy integration of deep learning algorithms. Initially, PyNNcml included rain detection algorithms using Long Short-Term Memory (LSTM) (Habi and Messer, 2018) and Gated recurrent unit (GRU) (Habi and Messer, 2019) as well as direct rain estimation using recurrent neural network (RNN) (Habi and Messer, 2020). The following releases include several existing non-deep-learning algorithms as well as the processing of min-max data (Ostrometzky et al., 2016) processing algorithms. Moreover, two interpolation algorithms are integrated inside PyNNCML including IDW and GMZ (Goldshtein et al., 2009)

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

which enables the creation of rain maps. For research, a rain field simulator is included inside PyNNcml which is based on RainGAN (Habi and Messer, 2021). PyNNCML goes beyond CML data processing and includes two SML algorithms that translate their attention into rain-rate. This grew PyNNcml to a software package that contained a collection of methods for research and efficient processing of CMLs and SMLs.

3.2 PWS rainfall data processing packages

In the following we give a brief introduction of the three PWS rainfall data processing software packages. A detailed list of the processing methods that they provide is given in Table 3.

Table 3. Overview of available methods in the four PWS processing software packages presented in section 3.2.

PWS data processing method	pws-pyqc	PWSQC	GSDR-QC	pypwsqc
Intra-station check				
t-test for "Days of the week" (Wilby et al., 2017) and "Hours of the day" (Blenkinsop et al., 2017)			x	
Changepoint analysis, consecutive streak and percentile checks (Lewis et al., 2021)			x	
ETCCI climate indices exceedance and world record checks (Lewis et al., 2021; Frich et al., 2002; Zhang et al., 2011)			x	
Inter-station checks				
Indicator correlation filter using a reference network (Bárdossy et al., 2021)	x			x
Event based filter using a reference network (Bárdossy et al., 2021)	x			
Faulty zeroes and high influx filter using neighbor observations (de Vos et al., 2019a)		x		x
Station outlier detection by low correlations in past timeseries with majority of nearby PWSs (de Vos et al., 2019a)		x		x
Suspicious daily and hourly wet/dry neighbor checks (Upton and Rahimi, 2003; Blenkinsop et al., 2017; Lewis et al., 2018, 2021)			x	
Susicpious monthly totals with neighboring gauges (Lewis et al., 2021)			x	
Timing offset and matching statistic test using nearest daily gauge (Lewis et al., 2021)			x	
Mean (daily) factor difference between gauge and nearest neighbor (Lewis et al., 2021)			x	
Time series of monthly factor difference between gauge and nearest monthly neighbor (Lewis et al., 2021)			x	
Bias correction				
Quantile mapping using trustworthy reference network (Bárdossy et al., 2021)	x			x
Overall correction of PWS network with a trustworthy reference network (de Vos et al., 2019a)		x		
Bias correction per PWS and per timestep based on correlation with the majority of nearby PWSs (de Vos et al., 2019a)		x		x

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

3.2.1 PWSQC

The PWSQC method has been developed at Wageningen University & Research and the Royal Netherlands Meteorological Institute to apply automatic QC on crowdsourced rainfall observations from PWS. The R-code that was published in 2019 is applicable on static PWS datasets, but follows principles such that the method could be applied in near real-time, i.e. at no time step does the QC rely on information from future data points. The method has been developed and tested using a substantial PWS dataset of rainfall observations by Netatmo devices in Amsterdam, the Netherlands (de Vos et al., 2019a), but is not limited to PWSs of this brand. PWSQC uses various comparisons between neighboring PWSs in order to identify three types of errors: faulty zeroes (continuous zero rainfall measurements when rainfall occurred), high influx (measurements far higher than those measured by nearby PWSs) and station outliers (rainfall dynamics over a period of time that has low correlation with the dynamics measured by its neighbors). Additionally, every time step the bias of the PWS compared to the neighbors is determined. If this changes, a bias correction factor is constructed to adjust future values to be more in agreement with the median of the neighbors. The overall assumption of the method is that, while individual PWSs may be incorrect, by evaluating the median of nearby PWSs, one can reliably determine rainfall amounts and dynamics. For that reason, the chosen parameters for neighbor selection should be optimized for the PWS input data's spatial and temporal resolution, and the typical spatial and temporal variability of rainfall in that local climate.

3.2.2 pws-pyqc

The pws-pyqc software package for filtering and quality-controlling data from PWS is written in Python and is the implementation of the methods from (Bárdossy et al., 2021). In contrast to other quality control algorithms which e.g. check for outlier values in time series, this approach uses concepts based on indicator correlations and rank statistics with a focus on high precipitation values. The package pws-pyqc currently includes three QC modules (indicator correlation filter, bias correction and an event-based filter) and requires a reference data set with trustworthy measurements (e.g. rain gauge network from a national met service). The algorithm is designed to work with larger PWS datasets where complete PWS time series are discarded if they do not match the spatial structure of the reference data set. The bias correction is based on a quantile mapping using the information from the reference data set and the event based filter checks individual time steps for outliers.

3.2.3 GSDR-QC

The Global Sub-Daily Rainfall Quality Control (GSDR-QC) provides a rule-based quality control procedure for hourly rain gauge observations (Lewis et al., 2021). The software developed at Newcastle University extends on existing U.K. quality control methods based on an understanding of rainfall patterns and instrumentation (Blenkinsop et al., 2017) comprising 25 quality checks and 11 rules on hourly rainfall amounts and dry periods to flag suspicious values. This is presented as an open-source Python package, consisting of 3 stages: (1) station metadata checks for accurate location data and duplicated stations, (2) flagging of suspicious data using a set of time series checks, (3) removal of suspicious periods in time series based on a set of user-defined rules. This approach enables users to test different rules and develop a customised rule base for their

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

own applications as necessary depending on individual user requirements (Lewis et al., 2018). Additional data is required, as methods rely on the Expert Team on Climate Change Detection and Indices (ETCDII) climate indices for monthly and daily thresholds, to assess the quality of observations. Recently an new implementation of the filters contained in GSDR-QC has been made available via the package RainfallQC (Keel, 2025) and work has been started to establish an interface with the OpenSense QC package *pypwsqc* (described in section 4.3).

3.3 SML data processing code

For the processing of SML data for rainfall estimation there is currently no publicly available software package. Research groups have developed individual functions in MATLAB and Python, but no central open-source collection of methods exists. OpenSense made a first step by providing a set of Python functions and example SML data with the material of the "OpenSense training school on software and methods for data processing from opportunistic rainfall sensors" (https://github.com/OpenSenseAction/training_school_opensene_2023).

Due to the overlap of processing steps for CML and SML data, we are currently considering integrating the existing SML methods into pycomlink. Alternatively, we may start a new software package for SML data processing from scratch.

3.4 The OpenSense software sandbox, a tool to run selected OS processing packages online

Because the existing software packages described above are quite heterogeneous with regards to how easy they are to install and applicable to new datasets, the first step in OpenSense was to make these packages more accessible. To achieve this, we have set up the OpenSense software sandbox (https://github.com/OpenSenseAction/OPENSENSE_sandbox). It is based on a GitHub repository that integrates the software packages as git submodules and allows to run them online via the service mybinder.org, using a containerized environment, based on the conda package manager and the conda-forge package database. Both Python and R software can be run. Currently the sandbox includes the following packages: RAINLINK, pycomlink, PyNNcml, PWSQC and pws-pyqc. For most packages an executable example with an open dataset is provided. For pycomlink and the OpenMRG CML dataset (Andersson et al., 2022) an extended example is provided. A function to transform different open datasets into the common data format that was defined by OpenSense (Fencl et al., 2023) is also provided. The sandbox allowed us to assess the available methods and the individual advantages and disadvantages of the packages. Based on this assessment we decided to build three new software packages, which are described in section 4. While the active development now has shifted to these new software packages and away from the sandbox, it still is valuable to showcase the packages and methods that exist, and to get familiar with the involved processing steps

3.5 Further related software packages for processing hydrometeorological data

The open-source packages presented above have been developed by members of OpenSense, or in associated projects, focusing on processing of OS data. Here we want to list related packages, not limited to OS data processing, that are being used within OpenSense for working with hydrometeorological data.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

Visualization of sensor locations and their rainfall estimates is commonly needed to reveal characteristics of sensor networks and to study the performance of their rainfall estimates. To this end, the open-source Python code MapRAINLINK has been developed by OpenSense member KNMI. It can visualize rain gauge, radar, and CML locations and their rainfall estimates on a map. Maps are highly customizable and MapRAINLINK is potentially applicable for visualizing other variables (https://doi.org/10.5281/zenodo.17279362). Some functionalities, such as background maps, could be added to *poligrain*.

Beyond the simple IDW- and Kriging-based methods for spatial interpolation provided by the packages described in section 3.1, other methods for the generation of rainfall maps from rain gauge and CML data exist. The package RMWSpy (Hörning and Haese, 2021) allows conditional spatial random field simulation which can be conditioned by point and line data (Blettner et al., 2022). The CLEAR method, available as R code, uses a random cascade model for disaggregating the path-averaged rainfall information from CMLs (Fencl and Schleiss, 2025).

The R package Titan and its associated C++ version Titanlib, has been developed by Met Norway for QC weather observation data. It mainly applies inter-station checks and it can employ auxiliary data for QC (Båserud et al., 2020; Lussana et al., 2020). It is commonly applied to temperature data but it was also tested with rain gauge data.

Moving forward to applications, radar-based or OS-based rainfall maps can be used as input by the open-source Python software pySTEPS to generate short-term weather forecasts, called nowcasts. pySTEPS (https://pysteps.github.io) is used operationally for radar-based nowcasting, but has also been tested successfully with CML rainfall fields in research mode (Imhoff et al., 2020)).

A successful example of open-source software for precipitation estimation is the wradlib (https://wradlib.org) open-source Python library for weather radar data processing (Heistermann et al., 2012), which is even used for specific processing steps of operational radar precipitation products. Although it has not been designed for OS data, its operational application and technical performance, as well as its high level of documentation, can serve as further inspiration for improving the OpenSense open-source software for OS data processing. Similarly, the open-source Python ARM Radar Toolkit (Py-ART, https://arm-doe.github.io/pyart/index.html) provides tools for radar data processing.

For rainfall retrieval from CML and SML data, similar to working with weather radar data, one needs to know the relation between rainfall rate and the actual microwave measurements, attenuation in case of CML and SML data. These relations depend on the interaction between microwave signals with hydrometeors. For investigating these, open-source code of the T-Matrix method (Mishchenko et al., 1996) exists, also in the form of the easily applicable Python package pytmatrix (Leinonen, 2014). Similarly, in particular for CML data, atmospheric water vapor can contribute a significant change of the microwave signals. This interaction can be described via the MPM model (Liebe, 1989) which is available as a Python implementation in the package pyMPM (https://github.com/cchwala/pyMPM).

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

4 The new OpenSense software packages

In the following we will present the concept and content of the three new Python software packages *poligrain*, *pypwsqc* and *mergeplg* that have been developed jointly within OpenSense. But first we will explain the software ecosystem, i.e. the interplay of the different new and existing software packages is envisioned.

4.1 Envisioned structure of the ecosystem of software packages

Our envisioned structure of software packages is shown in Figure 3. As foundation, at the bottom of this structure, we decided to create the new software package *poligrain* with core functionalities that are common when processing and analyzing CML, SML and PWS data. This way we avoid interdependencies of the processing packages which would occur if one package imports another one to use a specific implementation e.g. for validation. The packages for data processing, which live one layer above *poligrain* should remain lean and focused on specific sensor data or specific methods. In applications, e.g. where data processing of several sensors and merging of these data is required, the selected processing packages are imported together with other specific required packages, e.g. *wradlib* in case of radar data processing.

With *poligrain* as the foundation of the structure, we can also leverage the fact that we, in OpenSense, have defined a common standard for the structure and the naming conventions of our data and metadata in NetCDF files (Fencl et al., 2023). Hence, we can assume a certain structure of data and metadata and simplify the function calls, in particular those requiring geographic location information. We combine this with using the *xarray* Python package which allows working with labeled multi-dimensional arrays. Our defined structure and naming conventions together with the *xarray* data containers for handling of data and metadata provides a simple and powerful, yet future-proof, API (in the sense of function arguments) for users.

4.2 *poligrain*: The foundational software package

The package *poligrain*, available at https://github.com/OpenSenseAction/poligrain, has the aim to simplify common tasks for working with point, line and gridded sensor data, focusing on rainfall observations. It was built from scratch in early 2024 as a result of the assessment of the current processing packages to centralize common functionalities that individual processing software packages might share. From the start we followed modern best-practices of software development. We based our repository on GitHub on a cookie-cutter template for Python packages that enables strict style check and static code quality analysis using ruff, continuous integration with running unittest and testing the Jupyter notebooks, automated generation of documentation and continuous delivery of package releases on pypi. In addition, a conda-forge build pipeline was set up.

In *poligrain* all high-level functions, i.e. those that a user would typically interact with, use a `xarray.Dataset` or a `xarray.DataArray` as input data type with a fixed naming convention for metadata. The example data that can be accessed directly from *poligrain* follows these standards. This allows concise code and avoids bugs due to common typos when providing coordinates of points, lines or grids.

Currently *poligrain* provides function for the following tasks:

- Plot point, line and grid data on a map

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

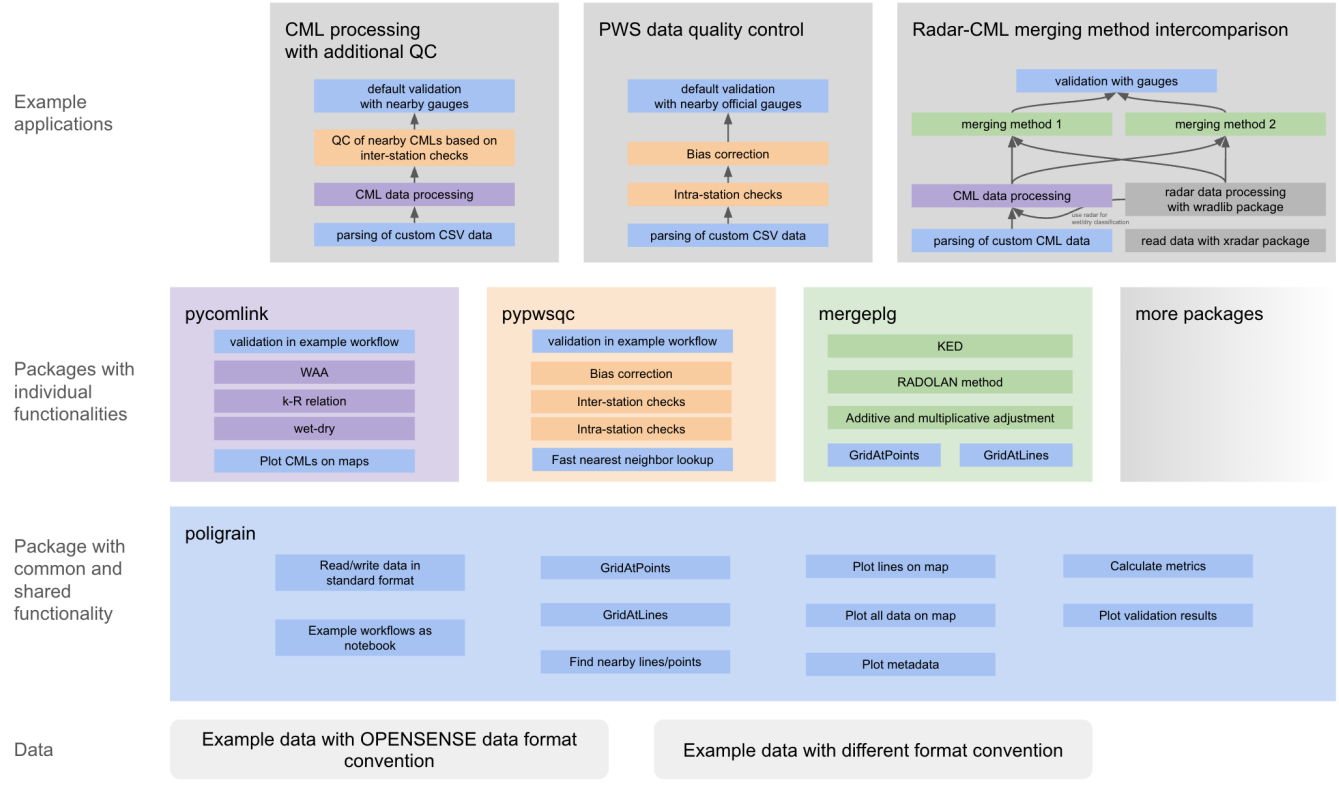

Figure 3. The structure of the envisioned OpenSense ecosystem of software packages described in section 4.1.

- Plot CML metadata
- Find neighbors of points and lines within specified range or create a distance matrix
- Calculate metrics for validation and generate validation plots
- Get data from intersection of lines and gridded data
- Load example datasets from OpenMRG (Andersson et al., 2022), OpenRainER (Covi and Roversi, 2025) and the Amsterdam PWS dataset (as used by de Vos et al. (2019a)

All this functionality is explained and applied in example Jupyter notebooks which are also automatically rendered as part of the online documentation of *poligrain*.

4.3 *pypwsqc*: A compilation of PWS QC methods

The package *pypwsqc*, available at https://github.com/OpenSenseAction/pypwsqc, was started within OpenSense to allow better interoperability of the two QC methods provided by de Vos et al. (2019a) and Bárdossy et al. (2021), described above in

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

section 3.2. It is based on *poligrain*, from which it uses e.g. the fast neighbour-lookup, and follows the same concept of using `xarray.DataArray` and `xarray.Dataset` as data containers with the naming conventions defined in OpenSense. The QC methods which are implemented have not just been copied from their original codebase but refactored to provide a modular set of functions. In the case of the station-outlier-filter fromde Vos et al. (2019a), a significant speed-up (approx. by a factor of 50) was achieved by using the rolling-window functionality from *pandas*. This limits its application to using a fixed window length, but it now allows to apply this filter for large PWS datasets without excessive computation time. The indicator correlation filter from Bárdossy et al. (2021) was extended and now provides a skill score, which allows the user to assess quality of the indicator correlation with neighbouring references. Bias correction via quantile mapping and neighboring stations is also available. Furthermore, a new filter was implemented which detects and corrects short data gaps and rainfall peaks specifically in Netatmo PWS data which are e.g. caused by signal processing errors due to connection interruptions. This may reduce the number of NaN values in PWS time significantly. As a next step we plan to either integrate the most effective filters from GSDR-QC or to interface with the new implementation of GSDR-QC in the RainfallQC (https://github.com/NERC-CEH/RainfallQC) package with the goal to extend our previous work (El Hachem et al., 2024) into a fully reproducible inter-comparison of rainfall QC methods.

4.4 *mergeplg*: Merging methods for point, line and grid data

Merging of radar and rain gauge data is a common concept and there are a variety of methods available (Goudenhoofdt and Delobbe, 2009). Many NMHS also do radar-adjustment with rain gauges on an operational basis. But when adding line-based information, e.g. from CMLs, instead of the point-based information, substantial changes to the adjustment procedure are required. The intersection of individual lines, typically the CML paths, and the grid on which the radar data is provided have to be calculated. Based on this, the path-average of the gridded data along each line can be derived. Since we typically deal with hundreds or even thousands of lines and, in case of testing new methods, several months of data, these calculations have to be computationally efficient. We created *mergeplg*, available at https://github.com/OpenSenseAction/mergeplg, to have one consistent codebase for radar adjustment with both point and line data. Different adjustment methods are available: Additive, multiplicative, the RADOLAN-approach (a weighted average of multiplicative and additive) and Kriging with external drift. The underlying interpolation methods, IDW and Kriging, can also be used for spatial interpolation of rainfall fields from point and line data. For Kriging, a variation using Block-Kriging is available, which takes into account the line-geometry during the Kriging procedure. Based on these implementations in *mergeplg* we are currently carrying out a reproducible method inter-comparison using the two large open datasets OpenMRG and OpenRainER which contain radar, CML and rain gauge data.

4.5 Example use case: CML processing, QC and merging with radar

To showcase the interaction of the packages in the OpenSense software ecosystem we provide a small example use case here. We use *poligrain* to load the example data, process the CML data with *pycomlink* (which is not a new OpenSense package but integrates into the ecosystem) leveraging *poligrain* functionality, apply an outlier filter from *pypwsqc*, and then adjust weather

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

radar data with the CML rain rates using *mergeplg*. The full workflow is available at https://github.com/OpenSenseAction/opensense_software_ecosystem_example and can be executed in a fully-reproducible way online via the service of https://mybinder.org/.

Below we describe the steps of the full workflow and highlight the functionalities of the used software packages.

Step 1: Load OpenMRG example data and plot overview map with *poligrain* (shown in Figure 4)

We use *poligrain* to load an 8-day subset from the OpenMRG dataset (Andersson et al., 2022). To get an overview of the data we plot rainfall accumulation from the radar and the gauges together with the CML paths. We do this using the `poligrain.plot_map.plot_plg()` function can take point, line and grid data as input. Data and metadata are stored and passed as `xarray.DataArray` or `xarray.Dataset`, following the naming conventions defined by OpenSense (Fencl et al., 2023). This in particular handy for line-based data for which no convenient plotting options exist.

```
# Imports also for the subsequent code examples
import poligrain as plg
import pycomlink as pycml
import pypwsqc.indicator_correlation as ic
import mergeplg as mrg

# Load an 8-day example dataset from OpenMRG
(
    ds_rad,
    ds_cmls,
    ds_gauges_municp,
    ds_gauge_smhi,
) = plg.example_data.load_openmrg(subset="8d")

# Plot rainfall sum for radar and gauge and add CML paths
plg.plot_map.plot_plg(
    da_grid=ds_rad.R.resample(time='1h').mean().sum(dim="time"),
    da_gauges=ds_gauges_municp.rainfall_amount.sum(dim='time'),
    da_cmls=ds_cmls,
    vmin=0,
    vmax=100,
    cmap='YlGnBu',
    kwargs_cmls_plot={"line_color": "k", "line_width": 1},
    kwargs_gauges_plot={"edge_color": "r", "s": 20},
)
```

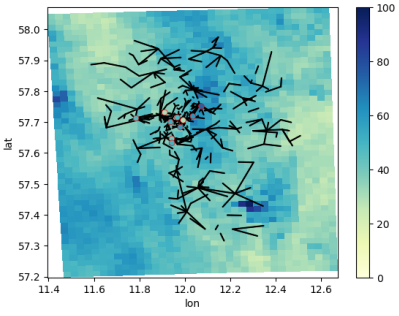

Figure 4. Code listing and example plot for step 1 of the example use case.

Step 2: CML processing with pycomlink using *poligrain* functionality (shown in Figure 5)

For the CML processing we first use *poligrain*'s functionality for extracting path-averages for each CML from the gridded radar rainfall data using `poligrain.spatial.GridAtLines`. We do this here as a demonstration for how radar rainfall data can be used for wet-dry classification. It shall be noted that this assumes that the weather radar data is a very reliable wet-dry indicator, which might not be the case for scenarios with ground clutter, overshooting or other radar-related issues. Ideally the radar-based wet-dry classification is combined with a CML-based one, but this is still an open research topic.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

After deriving the wet indicator from the path-averaged radar rainfall data we extend the wet data points one minute forward and backward in time to smooth out some intermittency in the radar data. We then follow the standard CML data processing, similar to what the basic example provided with *pycomlink* does.

Finally we again use *plot_plg* from *poligrain* to plot the CML rainfall sums and the radar rainfall sums on a map to get an idea of the resulting differences. In addition, we plot the difference between CML and radar rainfall sums for each CML on a map.

```
# get radar along CMLs
grid_at_lines = plg.spatial.GridAtLines(
    da_gridded_data=ds_rad.R,
    ds_line_data=ds_cmls,
)
da_radar_along_cmls = grid_at_lines(ds_rad.R)

# wet-dry classification based on smoothed
# time series of radar rain rates along each CML
ds_cmls['wet'] = (
    (da_radar_along_cmls > 0.1)
    .rolling(time=6*3, center=True)
    .max()
)
ds_cmls['wet'] = ds_cmls.wet.ffill(dim='time')

# Do other processing steps with pycomlink (details omited here)
ds_cmls["baseline"] = pycml.processing.baseline...
ds_cmls["waa"] = pycml.processing.wet_antenna...
ds_cmls["R"] = pycml.processing.k_R_relation...

# Plot CML rain rates vs radar map and plot diff at CML
plg.plot_map.plot_plg...
```

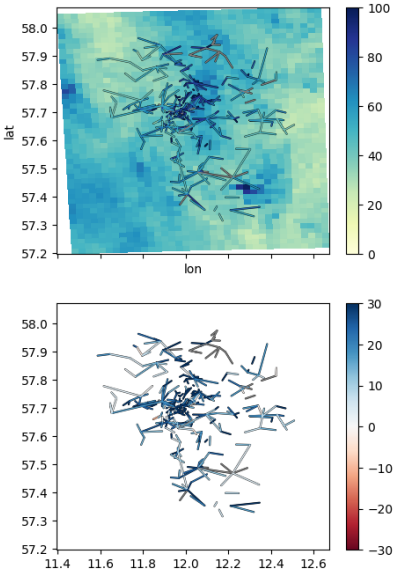

Figure 5. Code listing and plots for step 2 of the example use case. Note that the code has been shortened for presentation here. See the full code online which produces the plots shown here.

Step 3: Quality controlling CML rainfall data with *pypwsqc* (shown in Figure 6)

CML rainfall data, even after sophisticated processing, might still contain low quality data. Here we show, as an example, how a QC method developed for PWS rainfall data, can be applied to identify problematic CMLs. We use the functions from `pypwsqc.indicator_correlation`, imported as `ic` in the example code, to calculate the indicator correlation (Bárdossy et al., 2021) between the reference rain gauges in the CML rainfall data.

Note that the 8-day example data is too short to have robust rainfall statistics for the indicator correlation and there would also be other methods to detect problematic CML time series. This specific application of a PWS QC method for CML data in this example is for demonstration purposes only. If time series are long enough, applying PWS QC methods to CML rainfall data is a valuable approach, though (Nielsen et al., 2024).

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

Nevertheless, we find that the CMLs with conspicuously low indicator correlation of rainfall data are the ones with erratic fluctuations in their raw TL (total loss) data. We highlight one of the problematic CMLs in the scatter plot of all indicator correlation values, show its rainfall time series compared to the one form the closest rain gauge and plot the TL time series.

```
# Aggregate to 15 minutes
da_gauges = ds_gauges_municp.rainfall_amount.resample(time='15
    min').sum()
da_cmls = (
    ds_cmls.R.isel(sublink_id=0, drop=True)
    .resample(time='15min').mean()
    .rename({'cml_id': 'id'})
)

# Calculate indicator correlation and distance matrix
# for reference gauges
dist_mtx_ref, indcorr_mtx_ref = ic.indicator_distance_matrix(
    da_gauges,
    da_gauges,
    max_distance=30e3,
    prob=0.95,
    min_valid_overlap=6 * 24,
)

# Calculate indicator correlation and distance matrix between
# reference gauges and CML rainfall data
dist_mtx, indcorr_mtx = ic.indicator_distance_matrix(
    da_gauges,
    da_cmls,
    max_distance=30e3,
    prob=0.95,
    min_valid_overlap=6 * 24,
)

# Plot indicator correlation vs distance
# (see full workflow online)

# Plot time series of CML with low indicator correlation
# (see full workflow online)
```

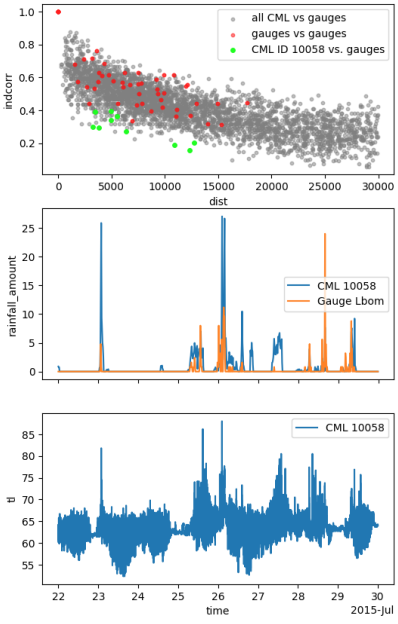

Figure 6. Code listing and plots for step 3 of the example use case. Note that the code has been shortened for presentation here. See the full code online which produces the plots shown here.

Step 4: Adjust radar data with CML rain rates using *mergeplg* (shown in Figure 7)

Finally we use the CML rainfall estimates to adjust the weather radar rainfall data. To do a simple IDW-based additive radar adjustment we use `mergeplg.merge.MergeDifferenceIDW` using the CML path-averages for adjustment instead of the typical point data from rain gauges. The `MergeDifferenceIDW` and all other merging methods in `mergeplg` support both point and line data.

We use hourly rainfall aggregation and do the adjustment for one specific hour. Again, we use `plot_plg` from `poligrain` for plotting either or radar, CML and gauge data. Visually, the adjusted radar rainfall maps already shows much better correspondence with the rain gauges. This is also clearly shown when plotting the differences between radar and gauge data at the gauge locations, which we calculated using `poligrain.spatial.GridAtPoints`.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026


```python
# Set up object to get radar values at rain gauges
grid_at_points = plg.spatial.GridAtPoints(
    da_gridded_data=ds_rad.R,
    da_point_data=ds_gauges_municp.rainfall_amount,
    nnear=1,
)

# Set up the merging object which internally calculates
# and stores the CML-radar intersection weights
merge_diff_idw = mrg.merge.MergeDifferenceIDW(
    ds_rad=ds_rad.R,
    ds_cmls=ds_cmls,
    method='additive',
    nnear=12,
)

# Aggregate to 1h
R_radar = ds_rad.R.resample(time='1h').mean()
R_cmls = (
    ds_cmls.isel(sublink_id=0).R.resample(time='1h').mean()
)
R_gauges = (
    ds_gauges_municp.rainfall_amount.resample(time='1h').sum()
)

# Do adjustment for one selected timestamp
t = '2015-07-26␣03:00:00'
R_grid_idw = merge_diff_idw(
    da_rad=R_radar.sel(time=t),
    da_cmls=R_cmls.sel(time=t),
)

# Get diff of radar and gauges at gauge location
R_diff_rad_gauge = grid_at_points(
    da_gridded_data=R_radar.sel(time=t),
    da_point_data=R_gauges.sel(time=t),
)
R_adjusted_diff_rad_gauge = grid_at_points(
    da_gridded_data=R_grid_idw,
    da_point_data=R_gauges.sel(time=t),

# Plotting the result (details omited)
plg.plot_map.plot_plg(
    da_grid=R_radar.sel(time=t),
    da_gauges=R_gauges.sel(time=t),
)
plg.plot_map.plot_plg(
    da_gauges=R_rad_at_gauge - R_gauges.sel(time=t),
)
# ...full code in online example
```

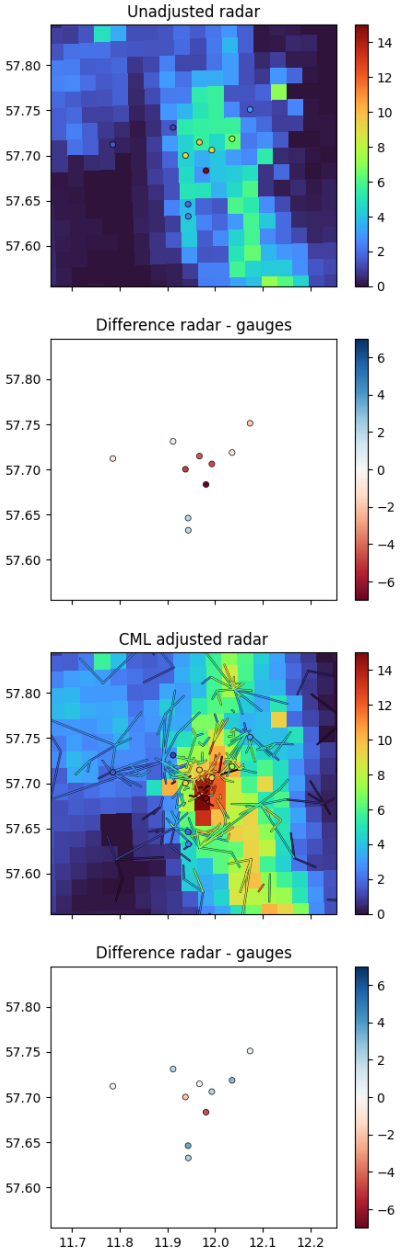

Figure 7. Code listing and plots for step 4 of the example use case. Note that the code has been shortened for presentation here. See the full code online which produces the plots shown here.

5 Conclusions

Open-source tools allow researchers to focus on improving algorithms and to accelerate analyses on new datasets instead of writing similar code that is already available in other research groups. Moreover, more contributors facilitate software

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

maintenance, solve issues - e.g. with new versions of dependencies - much quicker, and help to develop more robust code. Open-source tools not only foster scientific progress, but also lower the barrier to implementation in operational products.

Here, we provided an overview of open-source tools for processing opportunistic rainfall sensor data with a focus on CMLs and PWSs. Three software packages were specifically developed within the framework of the COST action OpenSense: *poligrain*, which simplifies common tasks for working with point, line and gridded sensor data, such as plotting on a map or collocating data from different sensor types; *pypwsqc*, which provides several QC algorithms for PWS data, that were previously available as a separate package; *mergeplg*, which contains merging methods for point, line and grid data. Within OpenSense, the already existing tool pycomlink now incorporates the most widely used methods for CML processing. Almost all remaining functionality from one of the other CML processing packages, RAINLINK, are being transferred to pycomlink and *poligrain*. This implies that a complete suite of open-source tools is now available for handling data from different OS and traditional sensors for rainfall estimation.

Available open OS datasets typically first need to be converted to the OpenSense standard data format to allow running with the open-source OS tools requiring more open-source data converters. Unfortunately, CML data as obtained by researchers from a mobile network operator is typically stored in different data formats meaning available example conversion code, although useful, still needs to be tailored to the specific dataset. Here, we make a plea for standardizing data and metadata at the network management system from the mobile network operators. This is being stimulated by a spin-off of the COST action OpenSense: the Global Microwave Link Data Collection Initiative (GMDI) for which a 12-month incubator phase was funded via the COST Innovators Grant for the project "Setting up the Global Microwave link Data collection Initiative for hydrometeorological applications" (setGMDI). This also addresses the collection and processing of CML data with the open-source OS tools. Improving sampling strategy and standardizing CML data and metadata is pursued by developing an ITU recommendation.

To facilitate the uptake of OS data in applications, such as hydrological or nowcasting models, such as the open-source model pySTEPS, tools should be developed to convert the output of the open-source OS tools to an appropriate format.

Although the developed OS tools already provide a lot of functionality, much more functionality could be added. This is partly connected to remaining research questions on OS rainfall estimation, such as developing better merging methods that use data from different sensors (point, line, gridded), for instance, PWS, automatic weather stations, CML and radar accumulations. This may take into account the support or foot print of the data, but should especially consider how to assign and weigh quality indices for data with different quality and spatial density. For merging and other applications, but also for rainfall estimation as such, more research, and hence source code, is needed to incorporate uncertainty in OS rainfall estimates. We hope that existing and new contributors will help to increase the functionality of the open-source OS tools.

Finally, how could the use of open-source tools be promoted? Dedicated coding meetings and short-term scientific missions help to deeply involve more (junior) researchers while training schools enhance the dissemination of knowledge on processing methods and how to use open-source tools to a broader audience. In our case, funding from the COST Action OpenSense did significantly reduce the barriers to participate by funding travel and accommodation. Without the support of OpenSense, which ended in October 2025, we now need to carry forward our work with the momentum and community established during the

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

last four years. With our ecosystem of open-source tools for OS rainfall data processing, we are confident to have the perfect foundation for this endeavor. First community-driven work based on these new tools - an inter-comparisons and benchmarking of CML, QC and merging methods - are under way, with publications planed to appear in the year 2026.

Code and data availability. All code is open-source and available on Github, with archiving via zenodo.org, as indicated by the URLs and references in the text. In the example use case we use the openly availble OpenMRG dataset (Andersson et al., 2022).

Author contributions. CC: Conceptualization, Visualization, Software, Writing - Original Draft, Writing - Review & Editing; AO: Conceptualization, Software, Writing - Original Draft, Writing - Review & Editing; EØ: Software, Writing - Review & Editing; LPW: Software, Writing - Original Draft, Writing - Review & Editing; JS: Software, Writing - Original Draft, Writing - Review & Editing; MG: Software, Writing - Review & Editing; BW: Software, Writing - Review & Editing; EC: Software; HVH: Software, Writing - Original Draft, Writing - Review & Editing; MF: Software, Writing - Original Draft, Writing - Review & Editing; LdV: Software, Writing - Original Draft; FG: Writing - Original Draft, Writing - Review & Editing; AG: Writing - Original Draft, Writing - Review & Editing; ToH: Writing - Original Draft, Writing - Review & Editing; NB: Software; TK: Software; GS: Software; AeH: Software; NI: Software; JP: Software; TS: Software; LK Software; DZ: Software, Writing - Review & Editing; VB: funding acquisition, project administration

Competing interests. The contact author has declared that none of the authors has any competing interests.

Acknowledgements. This publication is based upon work from the COST Action “Opportunistic Precipitation Sensing Network” (OPENSENSE, ref. CA20136), supported by COST (European Cooperation in Science and Technology).

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

References

Andersson, J. C. M., Olsson, J., van de Beek, R. C. Z. ., and Hansryd, J.: OpenMRG: Open data from Microwave links, Radar, and Gauges for rainfall quantification in Gothenburg, Sweden, Earth System Science Data, 14, 5411–5426, https://doi.org/10.5194/essd-14-5411-2022, publisher: Copernicus GmbH, 2022.

Angeloni, S., Adirosi, E., Sapienza, F., Giannetti, F., Francini, F., Magherini, L., Valgimigli, A., Vaccaro, A., Melani, S., Antonini, A., and Baldini, L.: Enhanced Estimation of Rainfall from Opportunistic Microwave Satellite Signals, IEEE Transactions on Geoscience and Remote Sensing, pp. 1–1, https://doi.org/10.1109/TGRS.2023.3349100, conference Name: IEEE Transactions on Geoscience and Remote Sensing, 2023.

Barthès, L. and Mallet, C.: Rainfall measurement from the opportunistic use of an Earth–space link in the Ku band, Atmos. Meas. Tech., 6, 2181–2193, https://doi.org/10.5194/amt-6-2181-2013, 2013.

Blenkinsop, S., Lewis, E., Chan, S. C., and Fowler, H. J.: Quality-control of an hourly rainfall dataset and climatology of extremes for the UK, International Journal of Climatology, 37, 722–740, https://doi.org/10.1002/joc.4735, _eprint: https://rmets.onlinelibrary.wiley.com/doi/pdf/10.1002/joc.4735, 2017.

Blettner, N., Chwala, C., Haese, B., Hörning, S., and Kunstmann, H.: Combining Commercial Microwave Link and Rain Gauge Observations to Estimate Countrywide Precipitation: A Stochastic Reconstruction and Pattern Analysis Approach, Water Resources Research, 58, e2022WR032 563, https://doi.org/10.1029/2022WR032563, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2022WR032563, 2022.

Blettner, N., Fencl, M., Bareš, V., Kunstmann, H., and Chwala, C.: Transboundary Rainfall Estimation Using Commercial Microwave Links, Earth and Space Science, 10, e2023EA002 869, https://doi.org/10.1029/2023EA002869, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2023EA002869, 2023.

Bárdossy, A., Seidel, J., and El Hachem, A.: The use of personal weather station observations to improve precipitation estimation and interpolation, Hydrology and Earth System Sciences, 25, 583–601, https://doi.org/10.5194/hess-25-583-2021, publisher: Copernicus GmbH, 2021.

Båserud, L., Lussana, C., Nipen, T. N., Seierstad, I. A., Oram, L., and Aspelien, T.: TITAN automatic spatial quality control of meteorological in-situ observations, in: Advances in Science and Research, vol. 17, pp. 153–163, Copernicus GmbH, https://doi.org/10.5194/asr-17-153-2020, iSSN: 1992-0628, 2020.

Chwala, C., Gmeiner, A., Qiu, W., Hipp, S., Nienaber, D., Siart, U., Eibert, T., Pohl, M., Seltmann, J., Fritz, J., and Kunstmann, H.: Precipitation observation using microwave backhaul links in the alpine and pre-alpine region of Southern Germany, Hydrol. Earth Syst. Sci., 16, 2647–2661, https://doi.org/10.5194/hess-16-2647-2012, 2012.

Chwala, C., Keis, F., and Kunstmann, H.: Real-time data acquisition of commercial microwave link networks for hydrometeorological applications, Atmos. Meas. Tech., 9, 991–999, https://doi.org/10.5194/amt-9-991-2016, 2016.

Covi, E. and Roversi, G.: OpenRainER, https://doi.org/10.5281/zenodo.14731404, 2025.

de Vos, L. W., Leijnse, H., Overeem, A., and Uijlenhoet, R.: The potential of urban rainfall monitoring with crowdsourced automatic weather stations in Amsterdam, Hydrology and Earth System Sciences, 21, 765–777, https://doi.org/https://doi.org/10.5194/hess-21-765-2017, 2017.

de Vos, L. W., Leijnse, H., Overeem, A., and Uijlenhoet, R.: Quality Control for Crowdsourced Personal Weather Stations to Enable Operational Rainfall Monitoring, Geophysical Research Letters, 46, 8820–8829, https://doi.org/10.1029/2019GL083731, 2019a.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

de Vos, L. W., Overeem, A., Leijnse, H., and Uijlenhoet, R.: Rainfall Estimation Accuracy of a Nationwide Instantaneously Sampling Commercial Microwave Link Network: Error Dependency on Known Characteristics, Journal of Atmospheric and Oceanic Technology, 36, 1267–1283, https://doi.org/10.1175/JTECH-D-18-0197.1, 2019b.

Djibo, M., Chwala, C., Graf, M., Polz, J., Kunstmann, H., and Zougmoré, F.: High-Resolution Rainfall Maps from Commercial Microwave Links for a Data-Scarce Region in West Africa, Journal of Hydrometeorology, 24, 1847–1861, https://doi.org/10.1175/JHM-D-23-0015.1, publisher: American Meteorological Society Section: Journal of Hydrometeorology, 2023.

El Hachem, A., Seidel, J., O'Hara, T., Villalobos Herrera, R., Overeem, A., Uijlenhoet, R., Bárdossy, A., and de Vos, L. W.: Technical note: A guide to using three open-source quality control algorithms for rainfall data from personal weather stations, Hydrology and Earth System Sciences, 28, 4715–4731, https://doi.org/10.5194/hess-28-4715-2024, publisher: Copernicus GmbH, 2024.

Ericsson: Ericsson Microwave Outlook, Tech. rep., https://www.ericsson.com/4a312c/assets/local/reports-papers/microwave-outlook/2018/ericsson-microwave-outlook-report-2018.pdf, 2018.

Fencl, M. and Schleiss, M.: CLEAR: a new discrete multiplicative random cascade model for disaggregating path-integrated rainfall estimates from commercial microwave links, Atmospheric Measurement Techniques, 18, 4467–4482, https://doi.org/10.5194/amt-18-4467-2025, publisher: Copernicus GmbH, 2025.

Fencl, M., Dohnal, M., Rieckermann, J., and Bareš, V.: Gauge-adjusted rainfall estimates from commercial microwave links, Hydrol. Earth Syst. Sci., 21, 617–634, https://doi.org/10.5194/hess-21-617-2017, 2017.

Fencl, M., Dohnal, M., Valtr, P., Grabner, M., and Bareš, V.: Atmospheric observations with E-band microwave links – challenges and opportunities, Atmospheric Measurement Techniques, 13, 6559–6578, https://doi.org/10.5194/amt-13-6559-2020, publisher: Copernicus GmbH, 2020.

Fencl, M., Nebuloni, R., C. M. Andersson, J., Bares, V., Blettner, N., Cazzaniga, G., Chwala, C., Colli, M., de Vos, L. W., El Hachem, A., Galdies, C., Giannetti, F., Graf, M., Jacoby, D., Victor Habi, H., Musil, P., Ostrometzky, J., Roversi, G., Sapienza, F., Seidel, J., Spackova, A., Van De Beek, R., Walraven, B., Wilgan, K., and Zheng, X.: Data formats and standards for opportunistic rainfall sensors, Open Research Europe, 3, 169, https://doi.org/10.12688/openreseurope.16068.1, 2023.

Frich, P., Alexander, L. V., Della-Marta, P., Gleason, B., Haylock, M., Tank, A. M. G. K., and Peterson, T.: Observed coherent changes in climatic extremes during the second half of the twentieth century, Climate Research, 19, 193–212, https://doi.org/10.3354/cr019193, 2002.

Garcia-Marti, I., Overeem, A., Noteboom, J. W., de Vos, L. W., de Haij, M., and Whan, K.: From proof-of-concept to proof-of-value: Approaching third-party data to operational workflows of national meteorological services, International Journal of Climatology, 43, 275–292, https://doi.org/10.1002/joc.7757, _eprint: https://rmets.onlinelibrary.wiley.com/doi/pdf/10.1002/joc.7757, 2023.

Gelbart, L., Barthès, L., Mercier-Tigrine, F., Chazottes, A., and Mallet, C.: Enhanced quantitative precipitation estimation through the opportunistic use of Ku TV-SAT links via a dual-channel procedure, Atmospheric Measurement Techniques, 18, 351–370, https://doi.org/10.5194/amt-18-351-2025, publisher: Copernicus GmbH, 2025.

Giannetti, F. and Reggiannini, R.: Opportunistic Rain Rate Estimation from Measurements of Satellite Downlink Attenuation: A Survey, Sensors, 21, 5872, https://doi.org/10.3390/s21175872, number: 17 Publisher: Multidisciplinary Digital Publishing Institute, 2021.

Giannetti, F., Moretti, M., Reggiannini, R., and Vaccaro, A.: The NEFOCAST System for Detection and Estimation of Rainfall Fields by the Opportunistic Use of Broadcast Satellite Signals, IEEE Aerospace and Electronic Systems Magazine, 34, 16–27, https://doi.org/10.1109/MAES.2019.2916292, conference Name: IEEE Aerospace and Electronic Systems Magazine, 2019.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

Goldshtein, O., Messer, H., and Zinevich, A.: Rain Rate Estimation Using Measurements From Commercial Telecommunications Links, Signal Processing, IEEE Transactions on, 57, 1616–1625, https://doi.org/10.1109/TSP.2009.2012554, 2009.

Goudenhoofdt, E. and Delobbe, L.: Evaluation of radar-gauge merging methods for quantitative precipitation estimates, Hydrology and Earth System Sciences, 13, 195–203, https://doi.org/https://doi.org/10.5194/hess-13-195-2009, publisher: Copernicus GmbH, 2009.

Graf, M., Chwala, C., Polz, J., and Kunstmann, H.: Rainfall estimation from a German-wide commercial microwave link network: optimized processing and validation for 1 year of data, Hydrology and Earth System Sciences, 24, 2931–2950, https://doi.org/https://doi.org/10.5194/hess-24-2931-2020, publisher: Copernicus GmbH, 2020.

Graf, M., El Hachem, A., Eisele, M., Seidel, J., Chwala, C., Kunstmann, H., and Bárdossy, A.: Rainfall estimates from opportunistic sensors in Germany across spatio-temporal scales, Journal of Hydrology: Regional Studies, 37, 100 883, https://doi.org/10.1016/j.ejrh.2021.100883, 2021.

Habi, H. V. and Messer, H.: Wet-Dry Classification Using LSTM and Commercial Microwave Links, in: 2018 IEEE 10th Sensor Array and Multichannel Signal Processing Workshop (SAM), pp. 149–153, https://doi.org/10.1109/SAM.2018.8448679, 2018.

Habi, H. V. and Messer, H.: RNN Models for Rain Detection, in: 2019 IEEE International Workshop on Signal Processing Systems (SiPS), pp. 184–188, https://doi.org/10.1109/SiPS47522.2019.9020603, iSSN: 2374-7390, 2019.

Habi, H. V. and Messer, H.: Recurrent Neural Network for Rain Estimation Using Commercial Microwave Links, IEEE Transactions on Geoscience and Remote Sensing, pp. 1–10, https://doi.org/10.1109/TGRS.2020.3010305, conference Name: IEEE Transactions on Geoscience and Remote Sensing, 2020.

Habi, H. V. and Messer, H.: RainGAN: A Conditional Rain Fields Generator, in: 2021 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS), pp. 529–532, https://doi.org/10.1109/COMCAS52219.2021.9629000, 2021.

Heistermann, M., Jacobi, S., and Pfaff, T.: Technical Note: An open source library for processing weather radar data (*wradlib*), Hydrology and Earth System Sciences Discussions, 9, 12 333–12 356, https://doi.org/10.5194/hessd-9-12333-2012, 2012.

Hörning, S. and Haese, B.: RMWSPy (v 1.1): A Python code for spatial simulation and inversion for environmental applications, Environmental Modelling & Software, 138, 104 970, https://doi.org/10.1016/j.envsoft.2021.104970, 2021.

Imhoff, R. O., Overeem, A., Brauer, C. C., Leijnse, H., Weerts, A. H., and Uijlenhoet, R.: Rainfall Nowcasting Using Commercial Microwave Links, Geophysical Research Letters, 47, e2020GL089 365, https://doi.org/https://doi.org/10.1029/2020GL089365, _eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2020GL089365, 2020.

ITU: ITU-R P.838-2: Specific attenuation model for rain for use in prediction methods, International Telecommunication Union, 2003.

Keel, T.: NERC-CEH/RainfallQC: 0.3.1, https://doi.org/10.5281/zenodo.17457184, 2025.

Kharadly, M. and Ross, R.: Effect of wet antenna attenuation on propagation data statistics, IEEE Transactions on Antennas and Propagation, 49, 1183–1191, https://doi.org/10.1109/8.943313, 2001.

Leijnse, H.: Hydrometeorological application of microwave links: measurement of evaporation and precipitation, Ph.D. thesis, Wageningen University, http://webdocs.dow.wur.nl/internet/hwm/hidde/PhD_thesis_Hidde_Leijnse.pdf, 2007.

Leijnse, H., Uijlenhoet, R., and Stricker, J. N. M.: Rainfall measurement using radio links from cellular communication networks, Water Resources Research, 43, W03 201, https://doi.org/200710.1029/2006WR005631, 2007.

Leijnse, H., Uijlenhoet, R., and Stricker, J.: Microwave link rainfall estimation: Effects of link length and frequency, temporal sampling, power resolution, and wet antenna attenuation, Advances in Water Resources, 31, 1481–1493, https://doi.org/10.1016/j.advwatres.2008.03.004, 2008.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

Leinonen, J.: High-level interface to T-matrix scattering calculations: architecture, capabilities and limitations, Optics Express, 22, 1655–1660, https://doi.org/10.1364/OE.22.001655, 2014.

Lewis, E., Quinn, N., Blenkinsop, S., Fowler, H. J., Freer, J., Tanguy, M., Hitt, O., Coxon, G., Bates, P., and Woods, R.: A rule based quality control method for hourly rainfall data and a 1km resolution gridded hourly rainfall dataset for Great Britain: CEH-GEAR1hr, Journal of Hydrology, 564, 930–943, https://doi.org/10.1016/j.jhydrol.2018.07.034, 2018.

Lewis, E., Pritchard, D., Villalobos-Herrera, R., Blenkinsop, S., McClean, F., Guerreiro, S., Schneider, U., Becker, A., Finger, P., Meyer-Christoffer, A., Rustemeier, E., and Fowler, H. J.: Quality control of a global hourly rainfall dataset, Environmental Modelling & Software, 144, 105 169, https://doi.org/10.1016/j.envsoft.2021.105169, 2021.

Liebe, H. J.: MPM—An atmospheric millimeter-wave propagation model, International Journal of Infrared and Millimeter Waves, 10, 631–650, 1989.

Lussana, C., Nipen, T. N., Båserud, L., Seierstad, I. A., Oram, L., and Aspelien, T.: metno/TITAN: version 2.1.1, https://doi.org/10.5281/zenodo.3667625, 2020.

Messer, H., Zinevich, A., and Alpert, P.: Environmental Monitoring by Wireless Communication Networks, Science, 312, 713, https://doi.org/10.1126/science.1120034, 2006.

Mishchenko, M. I., Travis, L. D., and Mackowski, D. W.: T-matrix computations of light scattering by nonspherical particles: A review, Journal of Quantitative Spectroscopy and Radiative Transfer, 55, 535–575, https://doi.org/10.1016/0022-4073(96)00002-7, 1996.

Nielsen, J. M., van de Beek, C. Z. R., Thorndahl, S., Olsson, J., Andersen, C. B., Andersson, J. C. M., Rasmussen, M. R., and Nielsen, J. E.: Merging weather radar data and opportunistic rainfall sensor data to enhance rainfall estimates, Atmospheric Research, 300, 107 228, https://doi.org/10.1016/j.atmosres.2024.107228, 2024.

O'Hara, T., McClean, F., Villalobos Herrera, R., Lewis, E., and Fowler, H. J.: Filling observational gaps with crowdsourced citizen science rainfall data from the Met Office Weather Observation Website, Hydrology Research, 54, 547–556, https://doi.org/10.2166/nh.2023.136, 2023.

Olsen, R., Rogers, D., and Hodge, D.: The aR^brelation in the calculation of rain attenuation, Antennas and Propagation, IEEE Transactions on, 26, 318–329, 1978.

Olsson, J., Horváth-Varga, L., Beek, R. v. d., Graf, M., Overeem, A., Szaton, M., Bareš, V., Bezak, N., Chwala, C., Michele, C. D., Fencl, M., Seidel, J., and Todorović, A.: How close are opportunistic rainfall observations to providing societal benefit?, Journal of Hydrometeorology, -1, https://doi.org/10.1175/JHM-D-25-0043.1, publisher: American Meteorological Society Section: Journal of Hydrometeorology, 2025.

Ostrometzky, J., Raich, R., Eshel, A., and Messer, H.: Calibration of the attenuation-rain rate power-law parameters using measurements from commercial microwave networks, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3736–3740, https://doi.org/10.1109/ICASSP.2016.7472375, iSSN: 2379-190X, 2016.

Overeem, A., Leijnse, H., and Uijlenhoet, R.: Measuring urban rainfall using microwave links from commercial cellular communication networks, Water Resources Research, 47, W12 505, https://doi.org/201110.1029/2010WR010350, 2011.

Overeem, A., Leijnse, H., and Uijlenhoet, R.: Country-wide rainfall maps from cellular communication networks, Proceedings of the National Academy of Sciences, 110, 2741–2745, https://doi.org/10.1073/pnas.1217961110, 2013.

Overeem, A., Leijnse, H., and Uijlenhoet, R.: Retrieval algorithm for rainfall mapping from microwave links in a cellular communication network, Atmos. Meas. Tech., 9, 2425–2444, https://doi.org/10.5194/amt-9-2425-2016, 2016a.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

Overeem, A., Leijnse, H., and Uijlenhoet, R.: Two and a half years of country-wide rainfall maps using radio links from commercial cellular telecommunication networks, Water Resources Research, 52, 8039–8065, https://doi.org/10.1002/2016WR019412, 2016b.

Overeem, A., Leijnse, H., Leth, T. C. v., Bogerd, L., Priebe, J., Tricarico, D., Droste, A., and Uijlenhoet, R.: Tropical rainfall monitoring with commercial microwave links in Sri Lanka, Environmental Research Letters, 16, 074 058, https://doi.org/10.1088/1748-9326/ac0fa6, publisher: IOP Publishing, 2021.

Overeem, A., Leijnse, H., van der Schrier, G., van den Besselaar, E., Garcia-Marti, I., and de Vos, L. W.: Merging with crowdsourced rain gauge data improves pan-European radar precipitation estimates, Hydrology and Earth System Sciences, 28, 649–668, https://doi.org/10.5194/hess-28-649-2024, publisher: Copernicus GmbH, 2024a.

Overeem, A., Uijlenhoet, R., and Leijnse, H.: Quantitative precipitation estimation from weather radars, personal weather stations and commercial microwave links, in: Advances in Weather Radar. Volume 3: Emerging applications, edited by Bringi, V., Mishra, K. V., and Thurai, M., pp. 27–68, Institution of Engineering and Technology, ISBN 978-1-83953-626-7 978-1-83953-627-4, https://doi.org/10.1049/SBRA557H_ch2, 2024b.

Ośródka, K., Otop, I., and Szturc, J.: Automatic quality control of telemetric rain gauge data providing quantitative quality information (RainGaugeQC), Atmospheric Measurement Techniques, 15, 5581–5597, https://doi.org/10.5194/amt-15-5581-2022, publisher: Copernicus GmbH, 2022.

Pastorek, J., Fencl, M., Rieckermann, J., and Bareš, V.: Precipitation Estimates From Commercial Microwave Links: Practical Approaches to Wet-Antenna Correction, IEEE Transactions on Geoscience and Remote Sensing, 60, 1–9, https://doi.org/10.1109/TGRS.2021.3110004, conference Name: IEEE Transactions on Geoscience and Remote Sensing, 2022.

Polz, J., Chwala, C., Graf, M., and Kunstmann, H.: Rain event detection in commercial microwave link attenuation data using convolutional neural networks, Atmospheric Measurement Techniques, 13, 3835–3853, https://doi.org/https://doi.org/10.5194/amt-13-3835-2020, publisher: Copernicus GmbH, 2020.

Schleiss, M. and Berne, A.: Identification of Dry and Rainy Periods Using Telecommunication Microwave Links, Geoscience and Remote Sensing Letters, IEEE, 7, 611–615, https://doi.org/10.1109/LGRS.2010.2043052, 2010.

Schleiss, M., Rieckermann, J., and Berne, A.: Quantification and Modeling of Wet-Antenna Attenuation for Commercial Microwave Links, IEEE Geoscience and Remote Sensing Letters, 10, 1195–1199, https://doi.org/10.1109/LGRS.2012.2236074, 2013.

Upton, G. J. G. and Rahimi, A. R.: On-line detection of errors in tipping-bucket raingauges, Journal of Hydrology, 278, 197–212, https://doi.org/10.1016/S0022-1694(03)00142-2, 2003.

van de Beek, C. Z., Leijnse, H., Torfs, P. J. J. F., and Uijlenhoet, R.: Seasonal semi-variance of Dutch rainfall at hourly to daily scales, Advances in Water Resources, 45, 76–85, https://doi.org/10.1016/j.advwatres.2012.03.023, 2012.

van der Valk, L. D., Coenders-Gerrits, M., Hut, R. W., Overeem, A., Walraven, B., and Uijlenhoet, R.: Measuring rainfall using microwave links: the influence of temporal sampling, Atmospheric Measurement Techniques, 17, 2811–2832, https://doi.org/10.5194/amt-17-2811-2024, publisher: Copernicus GmbH, 2024.

Walraven, B., Overeem, A., Coenders-Gerrits, M., Hut, R., Valk, L. v. d., and Uijlenhoet, R.: Relating Rainfall Retrieval Parameters to Network and Environmental Features to Improve Rainfall Estimates from Commercial Microwave Links in the Tropics, Journal of Hydrometeorology, 25, 1769–1791, https://doi.org/10.1175/JHM-D-24-0023.1, publisher: American Meteorological Society Section: Journal of Hydrometeorology, 2024.

Wenzel, M., Vogel, C., Graf, M., Winterrath, T., and Chwala, C.: Development of a Python Framework (pyRadman) for QPE using radar and CML data at DWD, Copernicus Meetings, https://doi.org/10.5194/egusphere-egu23-13779, 2023.

https://doi.org/10.5194/egusphere-2025-5438
Preprint. Discussion started: 10 February 2026

Wilby, R. L., Clifford, N. J., De Luca, P., Harrigan, S., Hillier, J. K., Hodgkins, R., Johnson, M. F., Matthews, T. K., Murphy, C., Noone, S. J., Parry, S., Prudhomme, C., Rice, S. P., Slater, L. J., Smith, K. A., and Wood, P. J.: The 'dirty dozen' of freshwater science: detecting then reconciling hydrological data biases and errors, WIREs Water, 4, e1209, https://doi.org/10.1002/wat2.1209, _eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wat2.1209, 2017.

Zhang, X., Alexander, L., Hegerl, G. C., Jones, P., Tank, A. K., Peterson, T. C., Trewin, B., and Zwiers, F. W.: Indices for monitoring changes in extremes based on daily temperature and precipitation data, WIREs Climate Change, 2, 851–870, https://doi.org/10.1002/wcc.147, _eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wcc.147, 2011.

Zinevich, A., Messer, H., and Alpert, P.: Frontal Rainfall Observation by a Commercial Microwave Communication Network, Journal of Applied Meteorology and Climatology, 48, 1317–1334, https://doi.org/10.1175/2008JAMC2014.1, 2009.

Øydvin, E., Graf, M., Chwala, C., Wolff, M. A., Kitterød, N.-O., and Nilsen, V.: Technical Note: A simple feedforward artificial neural network for high temporal resolution classification of wet and dry periods using signal attenuation from commercial microwave links, EGUsphere, pp. 1–15, https://doi.org/10.5194/egusphere-2024-647, publisher: Copernicus GmbH, 2024.

Øydvin, E., Gaban, R., Andersson, J., (C. Z.) van de Beek, R., Wolff, M. A., Kitterød, N.-O., Chwala, C., and Nilsen, V.: Combining commercial microwave links and weather radar for classification of dry snow and rainfall, Atmospheric Measurement Techniques, 18, 2279–2293, https://doi.org/10.5194/amt-18-2279-2025, publisher: Copernicus GmbH, 2025.