Grant Agreement: 814966

CARES

CITY AIR REMOTE EMISSION SENSING

WP2 – Deliverable 2.7:

Final version functions for off-line plume chase & point sampling processing

June 2023

Project website: www.cares-project.eu

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 814966. The sole responsibility for the content of this document lies with the authors. It does not necessarily reflect the opinion of the funding authorities. The funding authorities are not responsible for any use that may be made of the information contained herein.

Document history and validation

When	Who	Comments
5th January 2023	Dr Christina SCHMIDT (UHEI), Dr Denis Pöhler (UHEI & Airyx)	Compiled python programme and code with supporting README.pdf
14th April 2023	Dr. Denis Pöhler (UHEI & Airyx)	Review Report and update license agreement
19th April 2023	Dr Markus KNOLL (TUG)	Point sampling functions
21st April 2023	Dr Christopher Rushton (Uni Leeds)	Description of the CARES Science App data processing and software.
24th April 2023	Dr James TATE (Uni Leeds)	Prepared formatted deliverable document based on above.
25th April 2023	Ake SJODIN (IVL, CARES project coordinator)	Review
26th April 2023	Dr James TATE (Uni Leeds)	Final version
27th April 2023	Ake SJODIN (IVL, CARES project coordinator)	Submission
7th June 2023	Ake SJODIN (IVL, CARES project coordinator)	Revised version submitted after Project Officer's comments

Authors: Dr Christina SCHMIDT (UHEI & Airyyz), Dr Denis PÖHLER (UHEI & Airyx), Dr Markus KNOLL (TUG), Dr Christopher Rushton (UniLeeds). Document collated and formatted by Dr James TATE, ITS, University of Leeds, UK

Contact: Email: James TATE (j.e.tate@its.leeds.ac.uk)

Table of Contents

Executive Summary

Final version functions for off-line plume chase & point sampling processing.

PLUME CHASING:

This software/program evaluates the emission ratios of vehicles from a high-resolution time series measured by instruments in a Plume Chasing (PC) vehicle. The software is written in Python and has a 'Airyx Free Software License v2023.1' (attached) for free use. People are free to share, copy and redistribute the provided software (as it is) in any medium or format as long as they follow the license terms. All commercial use is excluded.

The software is designed so that any pollutant time series measured with the plume chasing method can be analysed with standard and custom settings. It relies on the simultaneous measurement of CO_2 as emission plume signal.

Before the emission ratios are calculated, the background values of the CO_2 and pollutant time series are evaluated by the software. This is the most sensitive part in the data analysis and can be done by different methods:

- **The default and recommended method for all gaseous pollutants is 01MinCO2. It assigns the minimum CO_2 value within a specified time interval from the time series and define the measured concentration of CO2 and of the pollutant as the background value.**
- **For particles it is recommended to use the method 02MinIndividual. It works like the method 01MinCO2 but assigns the minimum for CO_2 and the pollutant individually according to individual minimum in the time series.**
- **If the background is measured with an additional inlet during the Plume Chasing measurements, the method 03BgInlet can be chosen. For this method the carefully time-aligned and offset-corrected time series of the background data needs to be provided in the same data file as the plume data.**

The software then integrates the measures signals of CO_2 and pollutants over the measurements and derives the emission values for the individual vehicles.

The analysis software package has a detailed description and example data. The analysis is based on well-established plume chasing algorithms averaging the pollutant in the identified plume. The derived emission data are saved to data files which can be used for further analysis.

POINT SAMPLING:

The Point Sampling (PS) software framework is a comprehensive data analysis tool which provides functionality from pre-processing raw time series files from various instruments up to analyzing emission statistics of whole vehicle fleets. The core of the software is the developed peak detection algorithm (TUG-PDA), which determines and separates the emissions of the individual vehicles from instrument time series data. The software framework is developed with focus on modularity and extendibility such that new instruments used for PS can be easily integrated or such that new campaign data can be processed without much re-programming. The software framework is developed to be capable of processing emission data on a large-scale for 1,000s of vehicle passes. This was proved during the CARES project, where about 100,000 vehicle passes were processed delivering emission factors for more than 30,000 individual vehicle measurements.

In addition, the CARES *Science App*[1] has interactive modules which add additional functionality to the Point Sampling (PS) software. By analysing the PS CO_2 time-series measured by the instrument, the module generates a 'trigger' that a vehicle has driven past the instrument and a measurement can be processed. These 'triggers' can be used if PS is running without a light-gate or as a replacement/alternative to the light-gate input. The function demonstrates and show-cases the CARES data architecture is more than a secure and accessible database, but also how web-applications can be quickly developed and rapidly deployed to interact with the data, both in terms of data processing and analysis, then visualisation.

SOFTWARE ACCESS:

Copies of the off-line research software and supporting documentation and files can be requested by email from the developers:

- **Plume Chasing (PC) | Christina Schmidt (christina.schmidt@airyx.de) & Denis Pöhler (denis.poehler@airyx.de);**
- **Point Sampling (PS) | Markus Knoll (markus.knoll@tugraz.at)**

Else contact the work-package leader **James Tate** (**j.e.tate@its.leeds.ac.uk**) or Project coordinator **Åke Sjödin** (**ake.sjodin@ivl.se**).

[1] https://cares-science-app.azurewebsites.net

Attainment of the Objectives and Explanation of Deviations

Description of work related to deliverables as given in DoW

For the innovative Remote Emission Sensing (RES) technique of Plume Chasing (PC) and Point Sampling (PS), there was not prior to the CARES project standardized analysis software and data infrastructure. In addition to the 'Final version standalone near real-time plume chase analysis software' (Deliverable 2.5) there is the objective to:

- Develop automated off-line processing scripts for the plume chasing and point sampling data into a standardized format, so they can be readily ingressed in to the CARES project database.
- Develop different approaches to off-line correct plume chasing data for varying background concentrations and correct other interfering emission plumes.
- Develop approaches to off-line attribute the point sampling roadside emission measurements to an individual vehicle passing by. This includes developing automated processing scripts for cleaning raw data from point sampling and storing it in the CARES project database.

Time deviation from original DoW

Delivered six months behind schedule according to the last amendment to the Grant Agreement. The delay was communicated with and approved by the project officer and had no impact on the project implementation project.

Content deviation from original DoW

None.

1. Introduction

The aim of this deliverable is to describe the off-line data processing and functions for Remote Emission Sensing (RES) techniques of Plume Chasing (PC) and Point Sampling (PS). These developments are in addition to the 'Final version standalone near real-time plume chase analysis software' (Deliverable 2.5).

The methodology behind Plume Chasing, more details about the algorithm, and comparison to on-board emission measurements, and the capability of the Plume Chasing method to detect tampered vehicles are described in detail in the following reports: Farren, N. et al., 2022a; Farren, N. et al., 2022b; Schmidt, C. et al., 2021.

2. Plume Chasing

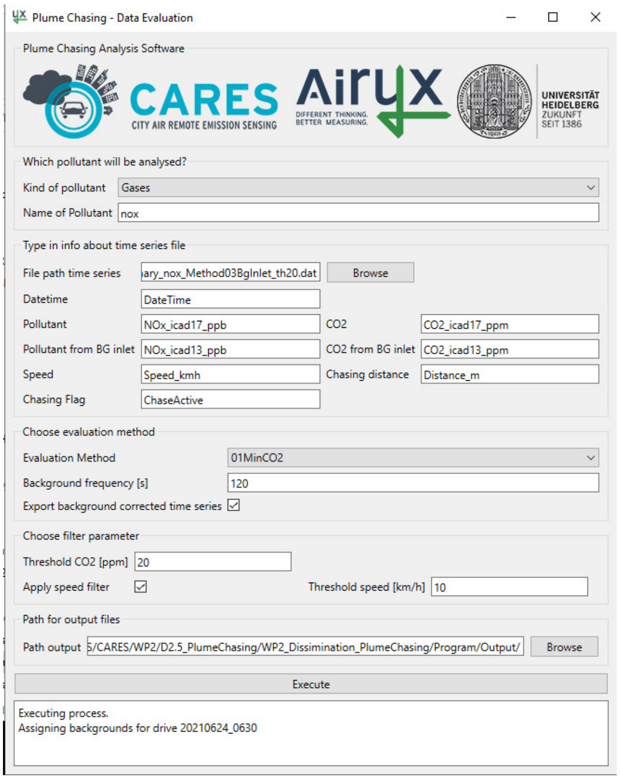
2.1. Software

This software/program evaluates the emission ratios of vehicles from a high-resolution time series measured by instruments in a Plume Chasing vehicle. The software is written in Python and has a Airyx Free Software License v2023.1.

Before the emission ratios are calculated, the background values of the CO_2 and pollutant time series are evaluated by the. This can be done by different methods:

- **The default and recommended method for all gaseous pollutants is 01MinCO2. It assigns the minimum CO_2 value within a specified time interval before each data point as its background value. The background for the pollutant is defined as the pollutant value at the timestamp of the assigned CO_2 background. Note that the minimum of CO_2 does not always coincide with the minimum of the pollutant, which can result in negative emissions values, as the background is determined too high in this case. In particular, this was observed for particle data due to a shift in arising CO2 and particle emissions. Therefore, this method is not recommended to be used for particles.**
- **For particles it is recommended to use the method 02MinIndividual. It works like the method 01MinCO2 but assigns the minimum for CO_2 and the pollutant individually accounting ins individual minimum concentrations between CO2 and pollutants in the time series.**
- **If the measurement system observes the background with an additional inlet during the Plume Chasing measurements, the method 03BgInlet can be chosen. For this method the carefully time-aligned and offset-corrected time series of the background data needs to be provided in the same data file as the plume data.**

The software has a GUI for simple selection of data to be processed:



2.2. Input data

The directory **Data** contains an example of the data file (2021_06_24_PC_timeseries_data.dat) needed to apply the software:

High-resolution Plume Chasing time series of the pollutant and CO_2 (format: .csv or .dat with tab separation).

This data file can be replaced by users own data. The file path and column headers should then be adapted accordingly in the user interface (see Running the software for more details). The time series data of the pollutant and CO_2 have to be time-aligned and offset-corrected. In addition, a smoothing of the time series may be required to compensate for potentially different response functions of the instruments (the response functions can be inferred by measuring a short emission peak). More details can be found in the following report: Schmidt, C. et al., 2021.

The units of the time series input data have to be:

- *timestamps* always with date and time
- *particle number* in #/ccm
- *particle mass* in g/ccm
- *CO_2* in ppm
- other gases like *NO* in ppb

All data values have to be plausible/valid (e.g. CO_2 > 350 ppm) and in the measurement range of the instrument.

Information about chasing times need to be provided in the time series data via a flag in an additional column. This flag should be set to *1* during active chasing times and to *0* otherwise (between two subsequent vehicle chases at least one entry with flag *0* is required).

2.3. Output data

Emission ratios for each chase are calculated in ppm/% for gases and #/kgfuel or mg/kg fuel for particles.

2.4. Running the software

To open the Plume Chasing software simply execute:

PlumeChasingAnalysis.exe

Press the button Execute to run the analysis. By default, the analysis settings match the example input data file and all output is written to the directory **Output**.

To change the predefined settings, modify these in the user interface.

2.5. License

The software is provided with a Airyx Free Software License v2023.1 (see Annex 6.1). People are free to share, copy and redistribute the provided software (as it is) in any medium or format as long as they follow the license terms. All commercial use is excluded.

3. Point Sampling

3.1. Software

A dedicated software analysis procedure was developed for the PS application. The software framework encompasses functionality starting from handling and harmonization of raw time series data from various instruments up to the calculation of resulting emission factors for the measured vehicles and generating statistics of the result. The software has been developed to be capable of processing emission data on a large scale for 1,000s of vehicles. A focus has been laid into the modularity and extendibility of the software. For each new measurement campaign, a new class is created derived from a general campaign class, where various settings can be defined (e.g. used instruments or relevant settings for the measurement spot such as sampling delays or real time offsets). The input files can be defined (e.g. instrument data, vehicle pass data, vehicle technical information). The software framework is programmed in Python consisting of about 40 classes and files with the usage of common libraries such as Pandas, NumPy or SciPy. In **Error! Reference source not found.**, an overview of the developed PS data analysis framework is shown. The data analysis is separated into three major processing steps:

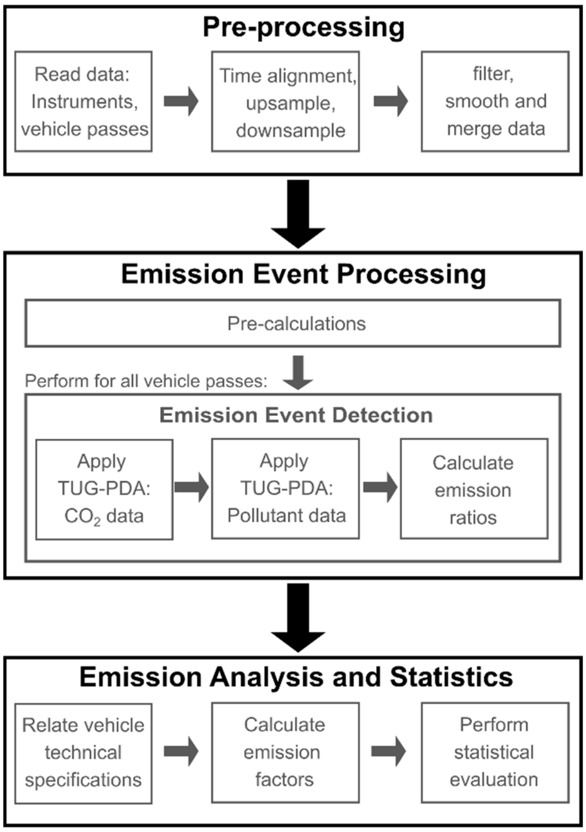


Figure 1. Overview of the PS data analysis software framework. (Illustration from Knoll et al. (2023))

Pre-processing

Raw emission data files from various instruments are processed in this first step. Various instruments are already implemented and derived from a general instrument class such as the newly developed Black Carbon Tracker (Knoll et al. (2021)) for BC and CO_2 measurements, a custom developed Diffusion Charger (Schriefl et al., (2020)) for PN detection, an ICAD (Airyx GmbH, Horbanski et al., (2019)) for NO_2, NO_x and CO_2 measurements or common particle measurement instruments such as a Condensation Particle Counter (CPC, TSI Incorporated) or an Aethalometer AE33 (Magee Scientific). The raw time series files of the different instruments are harmonized, time aligned, filtered and merged into one composite data set in this processing step.

Emission Event Processing

The emission event processing uses as input the composite time series data from the pre-processing step and the vehicle pass data (timestamp, speed, acceleration) to resolve the emissions of the individual vehicles. For this purpose, a peak detection algorithm (TUG-PDA) was developed to resolve and separate the emissions of the individual passing vehicles. The TUG-PDA is fully configurable with several settings (e.g. emission thresholds for the different analytes, minimum resolvable time between vehicles, maximum emission delay). The

TUG-PDA processes separately the CO_2 and pollutant emissions (e.g. BC, PN, NO_x). After emission concentrations for the individual vehicle passes are determined, emission ratios are calculated.

Emission Analysis and Statistics

Emission ratios are combined with technical vehicle data such as Euro standards, vehicle or fuel types and type approval CO_2 data. With this combined information various emission factors (e.g., fuel-based, distance-based, energy-based) are calculated. With the resulting information, functionalities are implemented which provide statistical information of the gathered emission data.

A more detailed description of the PS software analysis framework can be found in Knoll et al. (2023), with detailed descriptions of the framework and the TUG-PDA including insights into the performance.

3.2. Input data and settings

For the processing of the data of a conducted measurement campaign, a new *RESCampaign* class is created. The following settings can be configured:

- **Used instruments** (e.g. Black Carbon Tracker, ICAD, …)
- **Relevant information of the hardware measurement setup** (e.g. sampling delay of the individual instruments, real time offsets)
- Default values for the matching between automated number plate recognition (ANPR) camera and the vehicle pass data (from light barriers)

The following input data is required and can be handled by the PS software framework:

- **Raw instrument time series files** of the used instruments (mainly .csv or similar file types)
- **Vehicle pass data** (timestamp, speed, acceleration e.g. from light barriers)
- **ANPR data files** of the vehicle passes (pseudo anonymized IDs of the vehicle passes, time information)
- **Vehicle technical data**. Commonly, the derived vehicle technical data differs from country to country. Therefore, if a new dataset is added, a new vehicle characteristics class must be created for the gathered technical information. The classes share all the same structure and fields (e.g. Euro standard definition, fuel type definition) and can be easily adapted.

3.3. Output data

The following output files are generated by the PS software framework.

- **Pre-processing: One file with the harmonized, time-aligned, filtered and merged emission time series of all used instruments**
- **Emission Event Processing: One file with the calculated emission ratios of the vehicle passes including integrated concentrations and determined emission ratios for all captured vehicle passes**

- **Emission Analysis and Statistics:**
    - **One file with the calculated emission factors (fuel-based, distance-based, energy-based) including technical information of the vehicles.**
    - **Various, configurable plots for the processed vehicle statistics**

3.4. Software usage

The PS software framework is programmed in Python[2]. It is fully configurable and can be easily extended due to the modular architecture based on inheritance of base classes for instruments, campaigns or technical vehicle information. If a new campaign is conducted with already integrated instruments, the usage is straightforward, because only the relevant campaign information and settings must be configured (see section **Error! Reference source not found.**). If new instruments or vehicle technical information must be added, existing classes can be copied and the relevant information can be adapted.

In the "main"-File of the software framework, the different measurement campaigns can be executed.

A Python Integrated Data Environment (IDE) such as *PyCharm*[3], *Spyder*[4] or *VSCode*[5] is recommended for the usage of the software.

[2] https://www.python.org/
[3] https://www.jetbrains.com/help/pycharm/installation-guide.html
[4] https://www.spyder-ide.org/
[5] https://code.visualstudio.com/

4. CARES database Science App

The CARES project aims to reduce the barrier to entry for remote sensing both from a cost perspective and from a skill perspective. The data-science element of the CARES project aims to improve the access to remote sensing insights by standardising the data format, processing and analysis algorithms. The CARES data architecture, under-pinning database, and the interactive *City App* and *Science App* are documented in Deliverable D2.1 'Comprehensive CARES Remote Sensing Database and Apps' (April 2023). The standardisation of data is achieved by integrating new data sets into a document style *Cosmos DB*[6] database and the standardisation of analysis algorithms is achieved by packaging a set of database queries and graphing algorithms alongside a standard data format in a web application that can be accessed by a range of pre-approved users.

The *Science App* has interactive modules which add additional functionality to the Point Sampling (PS) methods described in Section 3 by using the CO_2 time-series measured by the instruments to generate 'triggers' that a vehicle has driven past the instrument and a measurement can be processed. These 'triggers' can be used if PS is running without a light-gate or as a replacement/alternative to the light-gate input.

4.1. Implementation

The function *generate_peaks_from_timeseries(dataframe)* uses the *streamlit*[7], *altair*[8] and *scipy*[9] packages in Python[10] to create a dialogue that allows a user to visualise the time series of data and fine tune the parameters used to fit the data. The full implementation of the Python code is shown in Annex 6.2. Figure 2 shows the rendered function in a web browser along with help for each parameter. The y-axis is normalised in this plot.

Figure 2: Generate Triggers dialogue from the CARES Science App using CO2_TUG_BCT2 data channel between 10:00 and 11:00 on 27 September 2021

[6] https://azure.microsoft.com
[7] https://streamlit.io/
[8] https://pypi.org/project/altair/
[9] https://scipy.org/
[10] https://www.python.org/

Three parameters are configurable in the *scipy.find_peaks* function. Peak threshold specifies the minimum peak value, horizontal separation in arbitrary units of the time series sets the separation between two peaks and peak width in arbitrary units of the time series sets the minimum width of the peak. The arbitrary units for this time series are 0.5s, so a peak width of 4 relates to a peak width of 2 seconds.

The instrument can be characterized over the three input parameters to determine the best input parameter selection for further analysis. The height was assessed from 0.1 to 1 in steps of 0.05, and the separation and peak widths were assessed between 1 and 20 in steps of 1 for approximately 6500 assessments. Assessments that returned no peaks were removed from the analysis. The correlation between these three variables and the count of detected peaks is shown in Figure 3.

Further analysis is required to determine how best to maximize the yield of measurements from each point sampling campaign but some linear decrease in count appears to occur after 6 units or 3 seconds of separation, indicative of the time prior to that being dominated by noise. The peak width appears to stabilize at around 1.5 – 2 seconds of measurements suggesting that these parameters might be good starting points for analyzing the point sampling data. The normalization of the data means that the peak height should be assessed on a case-by-case basis, but in this example 0.25 of the maximum appears to be where the data begins to stabilize.

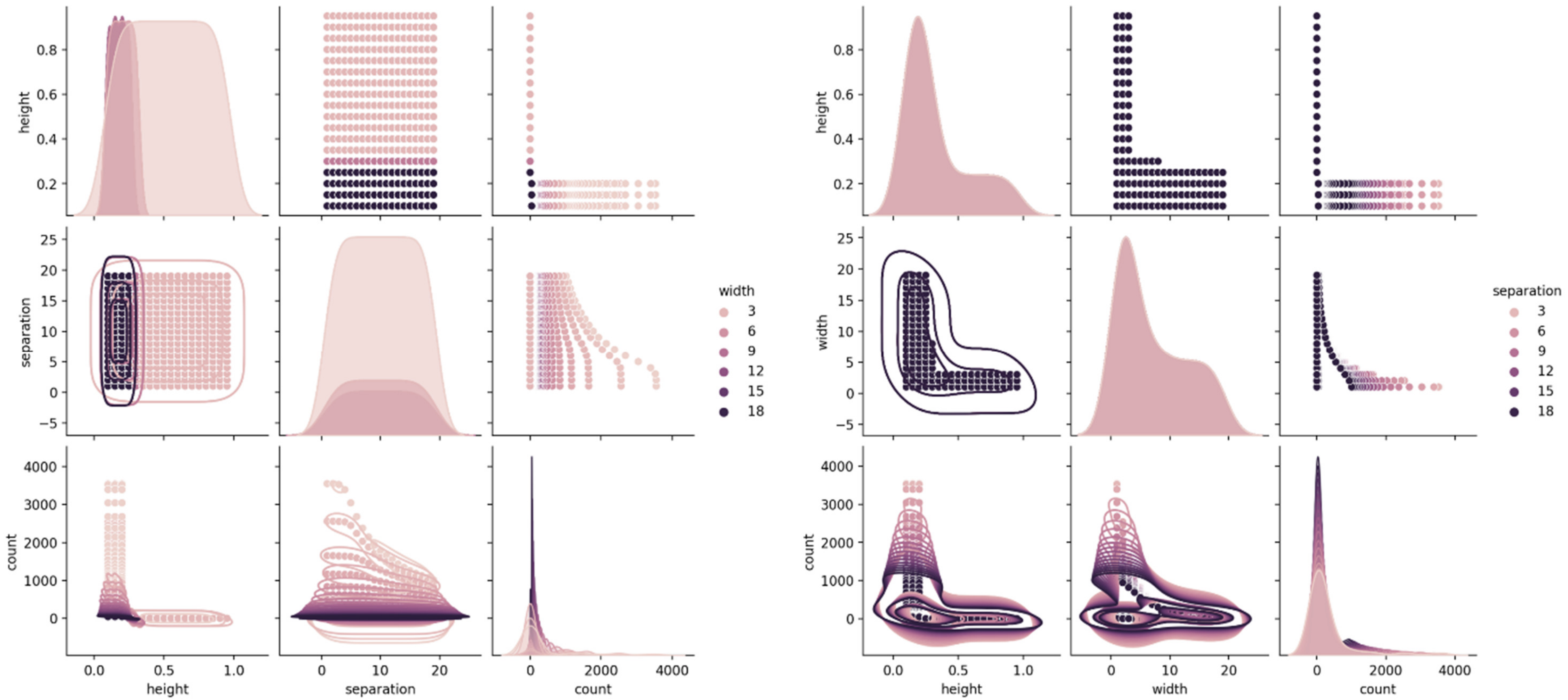


Figure 3: Panel plot showing instrument characterization by width (left) and separation (right)

4.2. Results

Input parameters of 0.25, 6 and 4 for the threshold, separation and width were inputted to the peak finding algorithm for the CARES Milan campaign measurements between 11:00 and 17:00 on 27 September 2021 using the point sampling channel *CO2_TUG_BCT2*. The point sampling channels for *CO2_ICAD_1*, *CO2_ICAD_2* and *NO2_ICAD* were analysed using the method described in section 3. Threshold values for the CO_2 and NO_2 channels were set to 5 and 2 respectively. The limit of detection parameter was set to 2 for all channels. 24 instances of co-observed peaks across all channels were found. An example (peak id = 53) co-peak is shown in Figure 4 and the measurement data is shown in the table following it.

Figure 4: Example co-peak for CO2 and NO2

id	*start_time*	*stop_time*	*bg_concentration*	*concentration*	*description*	*mid_time*
53	27/09/2021 16:19	27/09/2021 16:20	473.5405	5043.788	CO2_ICAD_1	20:09.5
53	27/09/2021 16:19	27/09/2021 16:20	471.5615	4116.481	CO2_ICAD_2	20:11.5
53	27/09/2021 16:19	27/09/2021 16:20	20.381	2521.106	NO2_ICAD	20:08.5

4.3. Discussion

The automatic peak detection method shows some potential for further reducing the instrument and deployment costs associated with point sampling methodologies. The time series analysis still requires some expert in the loop to extract the maximum value from these experiments, but as these methods become more automated, then less additional work will be required by the end user. The function demonstrates and show-cases the CARES data architecture is more than a secure and accessible database, but also how web-applications can be quickly developed and rapidly deployed to interact with the data, both in terms of data processing and analysis, then visualisation.

5. REFERENCES

Farren, N. et al., 2022a. CARES deliverable D1.1. Measurement technology intercomparison and evaluation. Report (https://cares-project.eu/measurement-tech-compare-d1-1/)

Farren, N. et al., 2022b. CARES deliverable D1.2. Monitoring of vehicle tampering. Report (https://caresproject.eu/monitoring-vehicle-tampering-d1-2/)

Schmidt, C. et al., 2021. Evaluation measurement campaign Utrecht 24.-28.02.2020. Report Version 2021.01.28 (accessible at CARES sharepoint: Documents/WP1-Technology development/Sniffer car February 2020/Summary_Evaluation)

Knoll, M., Penz, M., Juchem, H., Schmidt, C., Pöhler, D., Bergmann, A. Point Sampling as Roadside Vehicle Emission Screening Technique. Submitted to Atmospheric Measurement Techniques. 2023.

Knoll, M. Lang, B., Bergmann, A. Performance of Black Carbon Instruments for Extractive Remote Emission Sensing. AAAR conference. https://aaarabstracts.com/2021/AbstractBook.pdf. 2021.

Schriefl, M., Nishida, R.T., Knoll, M., Boies, A.M., Bergmann, A. Characterization of particle number counters based on pulsed-mode diffusion charging. Aerosol Science and Technology. 2020.

Horbanski, M., Pöhler, D., Lampel, J. Platt, U. The ICAD (iterative cavity-enhanced DOAS) method. Atmospheric Measurement Techniques. 2019.

6. ANNEXES

6.1. PLUME CHASING SOFTWARE LICENSE

Airyx Free Software License[11]
===========================

Version 2023.1
_Copyright © 2023 Airyx GmbH

This license agreement applies to the free version of the "Plume Chasing – Data Evaluation" Software package.

Terms and Conditions
BY DOWNLOADING, INSTALLING, USING, TRANSMITTING, DISTRIBUTING OR COPYING THIS SOFTWARE ("THE SOFTWARE"), YOU AGREE TO THE TERMS OF THIS AGREEMENT (INCLUDING THE SOFTWARE LICENSE AND DISCLAIMER OF WARRANTY) WITH Airyx GMBH THE OWNER OF ALL RIGHTS IN RESPECT OF THE SOFTWARE.
PLEASE READ THIS DOCUMENT CAREFULLY BEFORE USING THE SOFTWARE.
IF YOU DO NOT AGREE TO ANY OF THE TERMS OF THIS LICENSE THEN DO NOT DOWNLOAD, INSTALL, USE, TRANSMIT, DISTRIBUTE OR COPY THE SOFTWARE.
THIS DOCUMENT CONSTITUTES A LICENSE TO USE THE SOFTWARE ON THE TERMS AND CONDITIONS APPEARING BELOW.
The Software is licensed to you without charge for use only upon the terms of this license, and AIRYX GmbH reserves all rights not expressly granted to you.
AIRYX GmbH retains ownership of all copies of the Software.

1) License
You may use the Software without charge.
You may distribute exact copies of the Software to anyone for non-commercial purposes.

2) Restrictions
AIRYX GmbH reserves the right to revoke the above distribution right at any time, for any or no reason.
YOU MAY NOT MODIFY, ADAPT, TRANSLATE, RENT, LEASE, LOAN, SELL, ONSELL, REQUEST DONATIONS OR CREATE
DERIVATIVE WORKS BASED UPON THE SOFTWARE OR ANY PART THEREOF.
The Software contains trade secrets and to protect them you may not decompile, reverse engineer, disassemble or otherwise reduce the Software to a humanly perceivable form. You agree not to divulge, directly or indirectly, until such trade secrets cease to be confidential, for any reason not your own fault.

3) Termination
This license is effective until terminated. The License will terminate automatically without notice from AIRYX GmbH if you fail to comply with any provision of this License. Upon termination you must destroy the Software and all copies thereof. You may terminate this Licence at any time by destroying the Software and all copies thereof. Upon termination of this license for any reason, you shall continue to be bound by the provisions of Section 2 above. Termination will be without prejudice to any rights Airyx GmbH may have as a result of this agreement.

4) Disclaimer of Warranty, Limitation of Remedies
TO THE FULL EXTENT PERMITTED BY LAW, AIRYX GmbH HEREBY EXCLUDES ALL CONDITIONS AND WARRANTIES, WHETHER IMPOSED BY STATUTE OR BY OPERATION OF

[11] https://creativecommons.org/licenses/by-nc/4.0/ & https://creativecommons.org/licenses/by-nc/4.0/legalcode

LAW OR OTHERWISE, NOT EXPRESSLY SET OUT HEREIN. THE SOFTWARE, AND ALL ACCOMPANYING FILES, DATA AND MATERIALS ARE DISTRIBUTED "AS IS" AND WITH NO WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED. AIRYX GmbH DOES NOT WARRANT, GUARANTEE OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE WITH RESPECT TO ITS CORRECTNESS, ACCURACY, RELIABILITY, CURRENTNESS OR OTHERWISE. THE ENTIRE RISK OF USING THE SOFTWARE IS ASSUMED BY YOU. AIRYX GmbH MAKES NO EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS INCLUDING, WITHOUT LIMITATION, THE WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE SOFTWARE. NO ORAL OR WRITTEN INFORMATION OR ADVICE GIVEN BY AIRYX GmbH, IT'S DISTRIBUTORS, AGENTS OR EMPLOYEES SHALL CREATE A WARRANTY, AND YOU MAY NOT RELY ON ANY SUCH INFORMATION OR ADVICE.
IMPORTANT NOTE: Nothing in this Agreement is intended or shall be construed as excluding or modifying any statutory rights, warranties or conditions which by virtue of any national or state Fair Trading, Trade Practices or other such consumer legislation may not be modified or excluded. If permitted by such legislation, however, AIRYX GmbH' liability for any breach of any such warranty or condition shall be and is hereby limited to the supply of the Software licensed hereunder again as AIRYX GmbH at its sole discretion may determine to be necessary to correct the said breach.
IN NO EVENT SHALL AIRYX GmbH BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, AND THE LOSS OF BUSINESS INFORMATION OR COMPUTER PROGRAMS), EVEN IF AIRYX GmbH OR ANY AIRYX GmbH REPRESENTATIVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ADDITION, IN NO EVENT DOES AIRYX GmbH AUTHORIZE YOU TO USE THE SOFTWARE IN SITUATIONS WHERE FAILURE OF THE SOFTWARE TO PERFORM CAN REASONABLY BE EXPECTED TO RESULT IN A PHYSICAL INJURY, OR IN LOSS OF LIFE. ANY SUCH USE BY YOU IS ENTIRELY AT YOUR OWN RISK, AND YOU AGREE TO HOLD AIRYX GmbH HARMLESS FROM ANY CLAIMS OR LOSSES RELATING TO SUCH UNAUTHORIZED USE.

5) General

All rights of any kind in the Software which are not expressly granted in this Agreement are entirely and exclusively reserved to and by AIRYX GmbH.
This Agreement shall be governed by German law. Exclusive jurisdiction and venue for all matters relating to this Agreement shall be in courts and fora located in the State of Baden-Württemberg, Germany, and you consent to such jurisdiction and venue. This agreement contains the entire Agreement between the parties hereto with respect to the subject matter hereof and supersedes all prior agreements and/or understandings (oral or written). Failure or delay by AIRYX GmbH in enforcing any right or provision hereof shall not be deemed a waiver of such provision or right with respect to the instant or any subsequent breach. If any provision of this Agreement shall be held by a court of competent jurisdiction to be contrary to law, that provision will be enforced to the maximum extent permissible, and the remaining provisions of this Agreement will remain in force and effect.

6.2. Function to Generate Triggers from Time Series

```
def generate_peaks_from_timeseries(df):

    max_col_value = df['pollutant'].max()
    df['pol_norm'] = df['pollutant'] / max_col_value

    p = alt.Chart(df).mark_line().encode(
        x = alt.X('timestamp:T'),
        y = alt.Y('pol_norm:Q'),
        color = alt.Color('description', legend = alt.Legend(orient = 'top'))
    ).interactive()

    st.altair_chart(p, use_container_width = True)

    c1, c2, c3 = st.columns(3)
    with c1:
        select_height = st.number_input(label = 'Set peak threshold', min_value = 0.0, max_value = 1.0,
value = 0.9)
        st.info(':information_source: __Info:__ Peak threshold sets how high, between 0 and 1, the peak
in the time series must be to be considered.')
    with c2:
        select_separation = st.number_input(label = 'Select horizontal separation', min_value = 0,
max_value = 30, value = 5)
        st.info(':information_source: __Info:__ Peak horizontal separation sets how much separation is
required between other peaks for a peak in the time series must be to be considered.')
    with c3:
        select_width = st.number_input(label = 'Select peak width', min_value = 0, max_value = 30,
value = 3)
        st.info(':information_source: __Info:__ Peak width sets how wide the peak in the time series
must be to be considered.')


    peaks = find_peaks(df['pol_norm'], height = select_height, distance = select_separation, width =
select_width)
    locations = peaks[0]
    triggers = []
    for l in locations:
        temp_ts = df['timestamp'][l]
        triggers.append(temp_ts)

    triggers = pd.DataFrame(triggers).rename(columns = {0:'Date_in'})

    return triggers
```