Development of software tools for optimizing the operation of gaseous detectors and gas systems

A. Navarro[1, 2], G. Rigoletti[2], and P. Vanslambrouck[2]

[1]Dept. of Telematics Engineering, U. Carlos III of Madrid, Spain
[2]EP-DT-FS, CERN, Switzerland

August 30, 2024

Abstract

This report presents the implementation of signal processing algorithms to study eco-friendly gas mixtures. Key contributions include the implementation of peak detection algorithms, with preliminary insights indicating a negative correlation between the charge of peaks produced and HFO concentrations. The Pandas library remains the preferred data analysis tool for studies of gas mixtures. Finally, the mean over the baseline region was identified as the most effective method for baseline estimation under background radiation conditions, balancing computational efficiency and performance.

Keywords: RPCs, ECO-Friendly Gas Mixtures, Signal Processing, Peak Detection, Python, Pandas, Polars, Arrow, Baseline Estimation, Background Radiation

1 Introduction

Gaseous detectors, especially Resistive Plate Chambers (RPCs) [1], play a crucial role in the detection systems used in LHC experiments [2] such as ATLAS, CMS, ALICE, and LHCb [3]. RPCs rely on gaseous mixtures of gases with high Global Warming Potential (GWP) (e.g., R-134a or SF_6 [4]). With growing environmental concerns and increasing regulatory pressure [5], there is a need for eco-friendly alternatives.

The CERN gas group is actively looking for alternatives that aim to maintain the performance of these detectors while reducing their environmental impact. Recent studies have identified promising candidates, such as CO_2, He, and HFO-1234ze [6], but the characteristics and behavior of these gas mixtures are still unknown.

This project aims to develop and upgrade the software used by the gas group to improve the understanding of these eco-friendly gas mixtures. The primary objective is to design and implement new algorithms to analyze the signals generated by the detectors.

Furthermore, investigations are conducted on the impact of various baseline estimation methods in high-background radiation environments to ensure the accurate performance of the current software.

2 Upgrade of data analysis tools

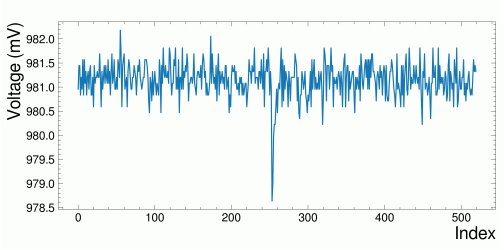
Evaluating eco-friendly gas mixtures demands a thorough understanding of their properties and behavior. The signals generated by these alternative mixtures exhibit greater complexity than those from traditional mixtures. As illustrated in Figure 1, this complexity arises because the gas mixture currently in use, 95.2% R-134a, 4.5% i-C_4H_{10}, and 0.3% SF_6, referred to as the standard gas mixture hereafter, generally produces clear, well-defined signals with a single prominent peak, as shown in Figure 1a. In contrast, alternative gas mixtures (e.g., 30% CO_2, 5% i-C_4H_{10}, 64% R-134a, and 1% SF_6, referred to as the 30% CO_2 gas mixture) often result in higher number of peaks and elevated levels of background noise, as depicted in Figure 1b.

This section focuses on enhancing Olefin [7], the current tool used to evaluate eco-friendly gas mixtures in RPCs. Additionally, alternative backend solutions are explored to ensure the tool's sustainability and long-term viability.

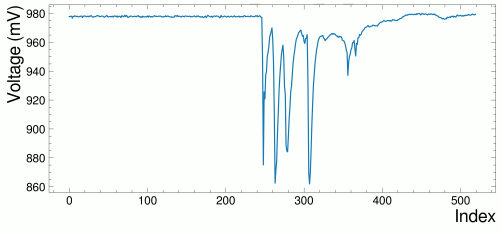
2.1 Implementation of the new features for peak analysis

The primary enhancement aimed for the Olefin tool is incorporating additional features, specifically the detection of after pulses and peaks, which is crucial to understanding the characteristics of alternative eco-friendly gas mixtures.

The new feature necessitated modifications to the Olefin tool at the signal level, Olefin's architectural



(a) *Signal output from an RPC with the standard gas mixture.*



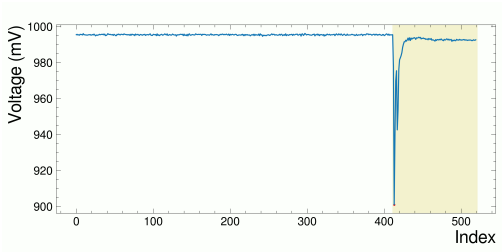
(b) *Signal output from an RPC with the 30% CO_2 gas mixture.*

Figure 1: *Comparison of signals from an RPC using a standard gas mixture versus an eco-friendly alternative. For details on the gas compositions refer to the main text.*

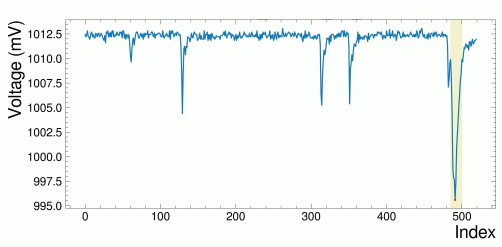
design is detailed in Reference [8]. In summary, the Olefin tool operates through multiple abstraction levels to facilitate signal analysis:

- **Signal Level:** This represents a waveform from a single strip of an RPC. Features, such as charge, time over threshold, and minimum height, are calculated.
- **Event Level:** This level aggregates the signal level, formed by one trigger from multiple strips, to study the characteristics of a trigger over the whole detector.
- **Acquisition Level:** Events are grouped into acquisitions at this level, where RPC performance is estimated at specified voltage points.
- **Run Level:** Finally, runs are formed by multiple acquisitions at different voltages, allowing the comparison of RPC's performances across various conditions.

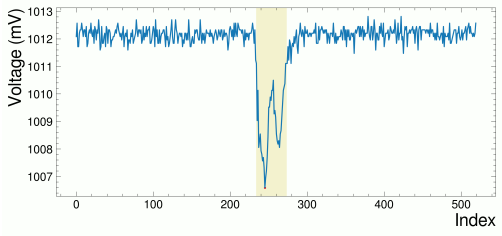
An initial approach to detect peaks is the threshold-based detection method. It involves establishing a threshold to filter out noise and subsequently grouping values exceeding this threshold into peaks. This method faces several challenges due to the diverse shapes of signals and the varying characteristics of peaks, as illustrated in Figure 2. It struggles to accurately detect peaks when signals do not return to baseline levels (see Figure 2a), when peaks are too small (see Figure 2b), or too close to each other (see Figure 2c).



(a) *Signal where certain peaks are not fully recovered after a preceding peak, leading to estimate a longer peak duration than the actual peak.*



(b) *Peaks with a width of only one index length, causing to fail to detect those peaks.*



(c) *Multiple closely spaced peaks, leading to the inability to assign them as separate peaks.*

Figure 2: *Challenges of the threshold-based detection method. Peaks in yellow region. 30% CO_2 gas mixture used.*

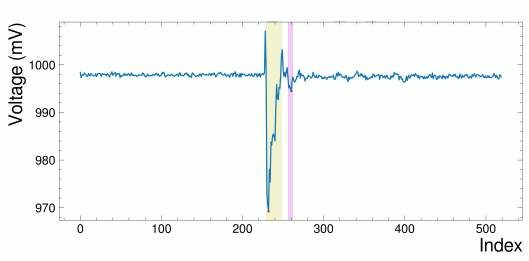


Figure 3: *Implementation of find-peak-based detection method. Peaks in yellow and pink regions. 30% CO_2 gas mixture used.*

To address these challenges, a second algorithm, the find-peak-based detection method, is proposed to more effectively analyze signals and accurately detect peaks, as shown in Figure 3. This new algorithm comprises the following steps:

1. Establish a baseline and threshold to filter signal noise.
2. Group values above the threshold into peaks.
3. Discard peaks that are either too small or too proximate to one another.
4. For each peak, the algorithm identifies the start, end, and maximum value. It then employs the `find_peaks` method from the SciPy library to separate peaks too close to each other. Each detected sub-peak is analyzed using the peak detection algorithm, and the features of the peaks (e.g., charge, time over threshold, and height) are calculated.
5. Filter the detected peaks.
6. Integrate the calculated peaks into the signal features.

The algorithm is integrated into the Olefin tool and evaluated with various signal datasets. It demonstrates consistent performance in detecting and analyzing peaks. Tests reveal that the find-peak-based detection method is approximately 1.77 times slower than the implementation without the peak detection feature on average (from 40 seconds to 70 seconds) when analyzing 100.000 signals. Despite this increase in processing time, the algorithm provided valuable additional information about the signals, making the performance trade-off acceptable for this use case.

Subsequent analyses with the new peak detection algorithm focused on the ECO-friendly gas mixtures to explore potential correlations between peak characteristics and RPC performance.

2.1.1 Study of gas mixtures with peak analysis algorithm

The behavior of various gas mixtures is investigated, focusing specifically on the characteristics of the detected peaks. The study aims to explain the relationship between the number of peaks produced, their respective charges, and the types of gases used.

The charge distribution of individual peaks is assessed for different CO_2 concentrations and the inclusion of HFO as an alternative gas. As shown in Figure 4a, the charge distributions for single peaks in the standard gas mixture and the 30% CO_2 mixture are comparable. In contrast, the HFO-containing mixture demonstrates a notably narrower distribution, suggesting that adding HFO increases the number of peaks with reduced charges.

Similarly, Figure 4b illustrates the relationship between the charge of a single peak and the voltage applied to the RPC. The charge characteristics of peaks in the standard gas mixture and the 30% CO_2 mixture are similar. However, the peaks in the HFO gas mixture are smaller than those in the standard and 30% CO_2 mixtures, indicating that HFO tends to produce smaller peaks compared to the other mixtures.

These findings, while preliminary, highlight the potential of the peak analysis algorithm for identifying and characterizing complex signal behaviors in eco-friendly gas mixtures. The results indicate that the algorithm effectively distinguishes subtle variations in peak characteristics, which can aid in optimizing gas mixtures for RPCs. Further investigation is required to confirm these observations and fully validate the findings. Nonetheless, the ability of the algorithm to provide detailed insights into peak behavior is valuable for advancing the development of more eco-friendly and efficient detector operations.

2.2 Assessment of alternative backend frameworks

The Olefin tool utilizes the Pandas library [9] for data analysis tasks. Pandas is a widely adopted Python library known for its comprehensive data manipulation capabilities. Despite its broad use, Pandas has notable performance limitations when handling large datasets [10].

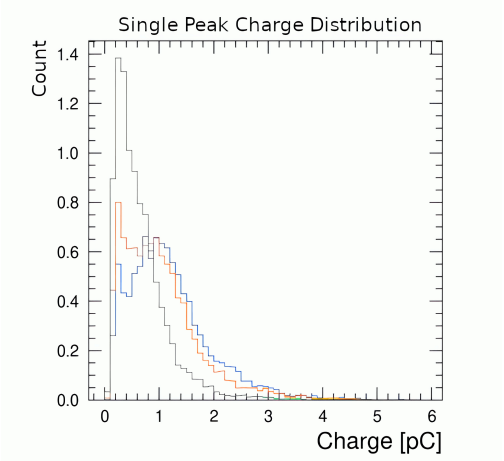
As the Olefin library evolves to include additional features, as described in Section 2.1, the use of more complex data structures (e.g., storing lists within columns) becomes crucial.

In response to these concerns, several alternative backend frameworks are evaluated to the current Pandas library. These options are the following:

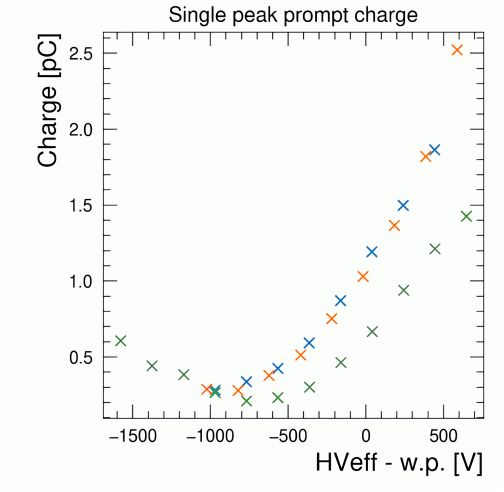
Pandas with PyArrow [11] integrates Apache Arrow [12] as its backend, which offers a columnar memory format optimized for modern hardware, including Compute Processing Units (CPUs) and Graphical Processing Units (GPUs). This framework supports native list storage within columns, addressing one of the critical limitations of Pandas.

Polars [13], developed by the Rust community, aligns with the principles of Apache Arrow and supports native list storage within columns. Engineered for speed and efficiency, it offers a potential improvement over Pandas and its extensions.

ROOT [14], a data analysis framework developed in C++ and extensively used at CERN, provides capa-



(a) *Demonstrates the histogram of the charge distribution of the single peaks for different gas mixtures.*



(b) *Illustrates the relationship between the voltage points, normalized to the working point of each gas mixture, and the average charge of single peaks for different gas mixtures.*

Figure 4: *Comparison of peak characteristics for different gas mixtures: the blue corresponds to the standard mixture, the orange to the 30% CO_2 gas mixture, and the green to a gas mixture with a composition of 60% CO_2, 35% HFO, 4% i-C_4H_{10}, and 1% SF_6.*

Library	Peaks	Read out	Transp.
P. (NumPy)	-	-	-
P. (PyArrow)	1.06x	0.14x	1994x
Polars	1.06x	0.54x	41063x

Table 1: *Comparison of Pandas with PyArrow, P. (PyArrow), and Polars libraries against Pandas with NumPy library, P. (NumPy), which serves as the reference. The table summarizes performance across three different tests normalized to Pandas with NumPy. Higher values indicate decreased performance.*

bilities for analyzing large, heterogeneous datasets. Nevertheless, its limited compatibility with Python libraries like SciPy [15], along with the extensive refactoring of its Application Programming Interface (API), led to its exclusion from further consideration.

Performance and suitability tests are conducted on each framework to assess their impact on the Olefin tool, as detailed in Table 1.

The `Peaks` test involves executing the `find_peaks` method [16] from the SciPy library on data from an RPC to evaluate the performance of each library with external functions. The results indicate that all frameworks perform equivalently in this task.

`Read out` test assesses the performance of reading a 10GiB dataset from an RPC. The results show that the columnar data schema of Polars and PyArrow demonstrate superior performance in this scenario as there is a 2x increase in performance compared to Pandas with Numpy.

Finally, `Transp.` test evaluates the performance of transposing the matrix containing the data of the signals across different backends. Tests reveal that Pandas with NumPy is significantly more efficient than column memory format implementations.

The findings suggest that none of the alternative frameworks outperform the existing Pandas-based implementation. While Pandas with PyArrow and Polars provide benefits such as native list storage and improved performance in read operations, the marginal gains do not justify the extensive modifications required to adapt the Olefin tool to these new frameworks. Therefore, the current implementation of Olefin based on Pandas with NumPy remains the most suitable choice.

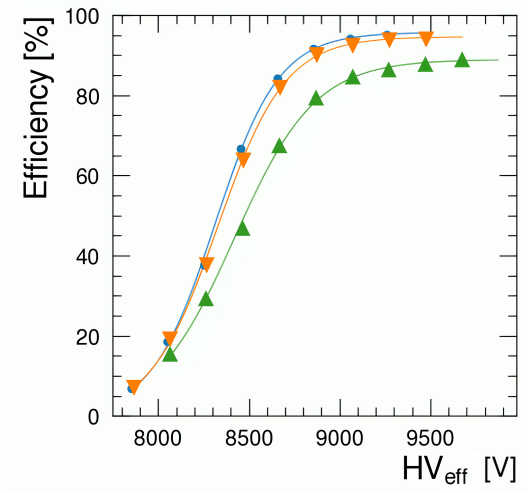


Figure 5: *Shows the efficiency of three tests concerning the corresponding working voltage of the RPC with varying amounts of background radiation. The gas mixture used is the same across the three, in this case, 54.5% R-134a, 40% CO_2, 5% i-C_4H_{10}, and 0.5% SF_6. The radiation levels of the different tests are:* $216\,\mu Sv\,h^{-1}$ *for the blue line with circle symbols,* $394\,\mu Sv\,h^{-1}$ *for the orange line with inverted triangle symbols, and* $1637\,\mu Sv\,h^{-1}$ *for the green line with triangle symbols.*

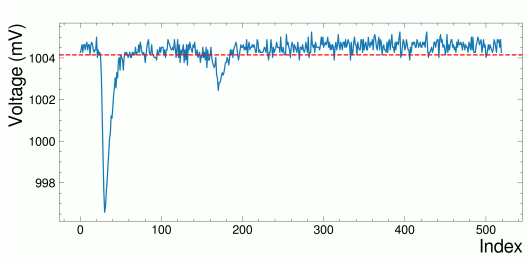


Figure 6: *Highlights a signal taken from an RPC with a 30% CO_2 gas mixture with a radiation level of 1594 uSv/h. The red horizontal line shows the estimation of the baseline level by performing the average of the samples on the baseline region.*

3 Research on radiation effects in the RPCs

Upon examining the RPCs' performance with various gas mixtures, as shown in Figure 5, the efficiency declines with increasing background radiation. This raises concerns, suggesting potential issues with the detectors or the Olefin analysis tool.

A thorough review reveals that the observed efficiency drop could be attributed to peaks developing in the baseline region, as depicted in Figure 6. The baseline, calculated as the average of the first 180 samples, is underestimated due to the presence of these peaks. Some peaks get missed, reducing the tool's efficiency compared to the one where the baseline is correctly calculated.

The existing baseline estimation technique seems inadequate, so alternative approaches to estimate the baseline are investigated such as:

- **Median of the baseline:** The main advantage is its robustness to outliers and extreme values, which means it provides a stable estimate even if some baseline samples are affected by noise or transient signals.
- **Mode of the baseline:** This method can be helpful if the baseline has a consistent and repeating value. Nevertheless, it may not be as reliable if the baseline region is noisy or multiple values are occurring with similar frequency.
- **Mean of the entire signal:** This strategy may be less effective if the signal has significant variations or noise. But it serves as an estimate for the other tests.
- **Savitzky-Golay filter:** It is a technique that applies polynomial smoothing to the baseline data. It works by fitting a polynomial of a specified degree to a moving window of the data points and using this fit to estimate the baseline. This approach is particularly beneficial for maintaining the signal's shape while reducing noise, making it valuable for baseline estimation in fluctuating data.

The performances of these methods are summarized in Table 2 where tests reveal that none of the alternative methods improve the efficiency by more than 1%. Specifically, the Mode method shows a 0.18% increase in efficiency but has a runtime that is 500 times longer. The Median method, while slightly more efficient than the Mode, still requires 23 times more computation time. The Filter method has zero efficiency improvement and demands 54 times the computation time of the mean method. The Mean Full method is the most performant providing a 1% efficiency increase with only a runtime increase of 1.46 times.

Given these results, it is clear that the mean method remains the most balanced choice, offering the best trade-off between efficiency and runtime.

4 Conclusion

New algorithms for analyzing eco-friendly gas mixtures are developed and successfully implemented. These tools incorporate peak detection techniques, providing promising insights into the behavior of various alternative gas mixtures. For instance, a preliminary analysis reveals a negative relationship be-

Method	Efficiency	Runtime
Mean	-	-
Mode	+0,18%	511x
Median	+0,15%	23x
Mean Full	+1%	1.46x
Filter	+0%	54x

Table 2: *Comparison of various baseline estimation methods based on efficiency and time increase normalized to the mean method. The table lists the efficiency of each method as the percentage increase compared to the mean baseline. Runtime is a factor relative to the mean method, where higher values signify longer computation times.*

tween the charge of the peaks and the concentration of HFO.

Additionally, the current data backend framework of Olefin, Pandas with Numpy, remains the optimal choice for the foreseeable future. This configuration offers the best balance between performance and functionality.

The evaluation of various baseline estimation methods confirmed that the mean method is the most effective for accurately estimating signal baselines.

A promising direction for future research involves refining the peak analysis algorithm by optimizing algorithm parameters for diverse signal types and exploring techniques for adaptive parameter tuning.

Furthermore, exploring the integration of machine learning methods for peak detection and baseline estimation represents an exciting new area of focus. Investigating how these models might be adapted to generalize across different types of gas mixtures and signal patterns could open new avenues for improving the performance and reliability of the Olefin tool in real-world applications.

5 Acknowledgements

In the first place, I want to express my sincere gratitude to my supervisors, Dr. G. Rigoletti and P. Vanslambrouck, for their exceptional mentorship and support throughout this project. I particularly appreciate Dr. G. Rigoletti for his invaluable technical expertise and guidance, which have been crucial to my research and professional growth.

I also wish to thank CERN and the CERN gas group for the invaluable opportunity to participate in the CERN Summer Student Programme, which has been a truly transformative experience. And to Prof. D. Larrabeiti and L. Leutgeb for their reference letters, which made this opportunity possible.

Furthermore, I would like to acknowledge my colleagues at CERN for their camaraderie. Finally, I am deeply grateful to my family for their support and encouragement.

References

[1] Marcello Abbrescia, Vladimir Peskov, and Paulo Fonte. *Resistive gaseous detectors: designs, performance, and perspectives*. John Wiley & Sons, 2018.

[2] Lyn Evans. "Particle accelerators at CERN: From the early days to the LHC and beyond". In: *Technological Forecasting and Social Change* 112 (2016), pp. 4–12.

[3] R Guida, M Capeans, and B Mandelli. "Gas Systems for Particle Detectors at the LHC Experiments: Overview and Perspectives". In: *Springer Proc. Phys.* 212 (2018), pp. 91–96. DOI: 10.1007/978-981-13-1313-4_19. URL: https://cds.cern.ch/record/2649531.

[4] Gianluca Rigoletti, Roberto Guida, and Beatrice Mandelli. "Studies on RPC detectors operated with environmentally friendly gas mixtures in LHC-like conditions". In: *The European Physical Journal Plus* 138.9 (2023), p. 841.

[5] *2020 Update of the European Strategy for Particle Physics (Brochure)*. Tech. rep. Geneva, 2020. DOI: 10.17181/CERN.JSC6.W89E. URL: https://cds.cern.ch/record/2721370.

[6] Mattia Verzeroli. "Studies of eco-friendly gas mixtures for RPC detectors at CERN LHC experiments". Presented 29 Apr 2022. Pavia U., 2022. URL: https://cds.cern.ch/record/2808797.

[7] Gianluca Rigoletti. *Olefin*. https://gitlab.cern.ch/gasteam/olefin.

[8] Gianluca Rigoletti. "Studies to reduce greenhouse gases emissions from particles detectors operation at the CERN LHC experiments". PhD thesis. Université de Lyon, 2022.

[9] *Pandas - Python Data Analysis Library*. https://pandas.pydata.org/.

[10] Felix Nahrstedt et al. "An Empirical Study on the Energy Usage and Performance of Pandas and Polars Data Analysis Python Libraries". In: *Proceedings of the 10th International Conference on Evaluation and Assessment on Software Engineering (EASE)*. 2024.

[11] *Pandas with PyArrow*. https://pandas.pydata.org/docs/user_guide/pyarrow.html.

[12] *Apache Arrow*. https://arrow.apache.org/.

[13] *Polars*. https://pola.rs/.

[14] CERN. *ROOT*. https://root.cern/.

[15] *SciPy*. https://scipy.org.

[16] *Find Peaks - SciPy*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html.