

# Electron Transport in Gaseous Detectors with a Python-based Monte Carlo Simulation Code

B. Al Atoum<sup>a</sup>, S. F. Biagi<sup>b</sup>, D. González-Díaz<sup>c</sup>, B.J.P. Jones<sup>a</sup>, A.D. McDonald<sup>a</sup>

<sup>a</sup> *Department of Physics, University of Texas at Arlington, Arlington, TX 76019, USA*

<sup>b</sup> *University of Liverpool, Physics Department, Liverpool L69 7ZE, United Kingdom*

<sup>c</sup> *Instituto Galego de Física de Altas Enerxías, Univ. de Santiago de Compostela, Campus sur, Rúa Xosé María Suárez Núñez, s/n, Santiago de Compostela, E-15782, Spain*

---

## Abstract

Understanding electron drift and diffusion in gases and gas mixtures is a topic of central importance for the development of modern particle detection instrumentation. The industry-standard **MagBoltz** code has become an invaluable tool during its 20 years of development, providing capability to solve for electron transport (‘swarm’) properties based on a growing encyclopedia of built-in collision cross sections. We have made a refactorization of this code from **FORTRAN** into **Cython**, and studied a range of gas mixtures of interest in high energy and nuclear physics. The results from the new open source **PyBoltz** package match the outputs from the original **MagBoltz** code, with comparable simulation speed. An extension to the capabilities of the original code is demonstrated, in implementation of a new Modified Effective Range Theory interface. We hope that the versatility afforded by the new **Python** code-base will encourage continued use and development of the **MagBoltz** tools by the particle physics community.

---

## 1. Introduction

The development of software that can accurately describe the transport properties of electrons in gas has been invaluable in the development and design of modern gaseous detectors. Experiments based on devices such as time projection chambers, drift chambers, and multiwire or micropattern proportional chambers rely critically on the realization of gas mixtures that optimize various figures of merit including charge multiplication and scintillation, attachment, diffusion or mobility [? ? ]. These properties can, under suitable assumptions, be calculated based on measured or swarm-parameter-based collision cross sections via Monte Carlo codes. Several software packages are presently available [? ] each with somewhat different applications and approaches. Among the more prominent are **MagBoltz** [? ], its sister-code **Degrad**, and **Garfield++** [? ] (which also uses **MagBoltz** cross sections), as well as others with more localized user bases. Many of the codes track the properties of an electron swarm that is evolving in time in step-wise manner, sampling from collision cross sections to evolve the ensemble in phase space. Given accurately described cross sections, these packages can provide critical information on electron drift in gas mixtures.

---

\*Corresponding author.

E-mail address: bashar.atoum@mavs.uta.edu

**MagBoltz**, used either directly or with its cross sections interfaced by **Degrad** or **Garfield++**, is one of the most widely used electron swarm simulation codes (a handful of applications include, for example, Refs [? ? ? ? ? ]). It is written in **FORTRAN**, with a built-in library of collision cross sections that is evolving continuously as the necessity for more accurate transport parameters or the availability of new gases dictates. This package is world-leading in terms of comprehensiveness of the cross section library and performance. Implementation within **FORTRAN**, however, implies some practical limitations that can represent a barrier against inclusion of new functionalities, complicate interfaces to other codes, and discourage some developers from working with the code-base. Students and Postdocs in High Energy and Nuclear Physics today are typically fluent in C++ and **Python**, for example, but infrequently expert at **FORTRAN**.

Motivated by an interest in studying the properties of diffusion-reducing gas mixtures for neutrinoless double beta decay [? ? ? ], we have undertaken a re-factorization of the original **MagBoltz** code into a more modern language. Our past use cases of the original **MagBoltz** code have included making systematic explorations of Xenon-based gas mixtures for reduced transverse diffusion [? ]. Helium appears to be an especially promising additive, and was studied using **MagBoltz** simulations in Ref. [? ]. The mixture has now been tested experimentally both in terms of its electron-cooling properties [? ], and electroluminescence light yield [? ]. A continuing experimental program with Xenon/Helium mixtures is under way to establish the effect on the topological signature of  $0\nu\beta\beta$  within the NEXT-DEMO++ program [? ].

Ongoing efforts to understand the detailed microscopic behaviour of electrons in various gas mixtures, including but not limited to diffusion suppression in Xenon+Helium, has required studying and modifying the **MagBoltz** calculation in some detail. This prompted us to re-factorize the original **FORTRAN** code into a more flexible format. Our refactorization involved a near-complete rewrite, redesigning to incorporate a modular and object-based structure, and re-optimizing the program flow. Algorithmically, the calculations are equivalent to the modern version of **MagBoltz**, and we take this opportunity to unambiguously assign all scientific credit for algorithmic development, tuning and evolution to original author, Steve Biagi [? ].

The framework chosen to support this project is the **Cython** [? ] extension of **Python**. **Cython** maintains the flexibility and code syntax of **Python** while inheriting some functionality from C++ to allow compilation, for improved speed of numerical calculations. The choice of **Cython** reflects the combined goals of implementing a **Python**-style interface for ease of use while maintaining the computational performance of the lower-level **FORTRAN** language (Sec. 3). The new **PyBoltz** code and documentation is publicly available at [? ], and is provided as open source, with further development and extension encouraged.

## 2. Electron Transport Implementation

The original **MagBoltz** code obtained its name on the basis of being a solver of the Boltzmann equation in a Magnetic field [? ? ]. However, since 1999, calculations within **MagBoltz** have been based instead upon Monte Carlo integration, following approximately the methods of Frasier and Mathieson [? ]. The **PyBoltz** code utilizes the same Monte-Carlo integration technique as **Magboltz**, which was outlined by Biagi in [? ]. Here we describe this method.

For the purposes of optimal computation speed, independent integrators are implemented for transport with and without thermal motion, and with no magnetic field, magnetic field parallel to the electric field, and with magnetic field at a generic angle to electric field.

The simulation proceeds electron-by-electron, and collision-by-collision. A specified number of real collisions are simulated, divided into a small number of “samples”. The samples each provide an independent measure of all drift parameters, and their standard deviation is used to assess statistical uncertainty on the simulation. Presently, all transport parameters are extracted from a single electron, propagated over a sufficiently large number of collisions that it is assumed to ergodically explore the available configuration space.

As pointed out by Skullerud [? ], sampling of time-to-next-collision given a general velocity-dependent cross section in principle involves a costly numerical integration between each scattering event. This computational problem can be overcome via Skullerud’s Null Collision method. Here, collisions are forced at a frequency much higher than the true collision frequency. However, the majority of these collisions are “Null” collisions, in that they transfer no energy or momentum. The benefit of this method is that the time between collisions is forced to be sufficiently short that the collision cross section can be assumed to be locally velocity independent. In such cases, the kinematic equations for electron transport can be solved analytically to yield the probability distribution for time-to-next-null-collision. This distribution is independent of applied magnetic field since it does not affect the energy of the electron in flight. After an appropriate, randomly sampled number of null collisions, a real collision is forced at a frequency determined by the various scattering cross sections of the gases. This method offers a substantial performance improvement over calculation of the time-to-next-real-collision directly.

Before simulations begin, the gas properties are processed to produce an effective summed cross section for each gas. Data tables of the elastic, elastic momentum transfer, attachment, rotation, vibration, excitation and ionization cross-sections are used to compute the summed energy-dependent cross-section. These energy-dependent cross section on each gas are calculated using a finite energy binning, which can be specified by the user, or calculated quickly on-the-fly via an iterative procedure within `PyBoltz`. This procedure is illustrated in the flowchart of Fig 2. The so-determined energy binning is also used to report electron energy distributions after thermalization. When a physical collision is realized, a gas species from the mixture is selected based on the concentration-weighted, energy-dependent cross sections in the relevant energy bin. Electron final state kinematics following scattering are drawn using one of a small number of scattering formalisms, which can be specified by the user. The presently available methods include the anisotropic scattering formalisms of Okhrimovskyy *et al* [? ] and Capitelli *et al* [? ], as well as simple isotropic scattering. In the case of Capitelli *et al* [? ], the angular distribution is calculated from the provided momentum-transfer and total cross sections at runtime. For Okhrimovskyy *et al* [? ], the angular distribution parameters are provided, pre-calculated from the cross sections, within the gas data tables.

Before and after each collision the energy, velocity, and position are updated and stored. The drift velocity is calculated per sample, given the total drift distance  $Z$  and time  $T$ , via:

$$W_z = \frac{Z}{T}. \quad (1)$$

Diffusion constants are calculated iteratively from the instantaneous electron coordinates per collision  $[x_i, y_i, z_i, t_i]$  via the equations:

$$D_x = \sum_i^{N_{coll}} \frac{(x_i - x_{i-M})^2}{t_i - t_{i-M}} \times \frac{t_i - t_{i-1}}{T} \quad (2)$$

$$D_y = \sum_i^{N_{coll}} \frac{(y_i - y_{i-M})^2}{t_i - t_{i-M}} \times \frac{t_i - t_{i-1}}{T} \quad (3)$$

$$D_z = \sum_i^{N_{coll}} \frac{(z_i - z_{i-M} - \hat{W}_z(t_i - t_{i-M}))^2}{t_i - t_{i-M}} \times \frac{t_i - t_{i-1}}{T} \quad (4)$$

From these constants, the conventionally defined  $D_L$  and  $D_T$  can be extracted, according to:

$$D_L = D_z \quad (5)$$

$$D_T = (D_x + D_y)/2 \quad (6)$$

In the ensemble-averages for  $D_{x,y,z}$ , the right factor converts the sum into a time-average by weighting according to the time between collisions. The left encodes calculates the mean square distance the electron has migrated over a large number of collisions  $M$ . This is divided by the time taken for those  $M$  collisions to occur. The sum over  $i$  then gives a suitable ensemble-average that converges to the diffusion constant. For accurate convergence the “decorrelation number”  $M$  must be suitably long that the positions at time  $t_i$  are uncorrelated with those at  $t_{i-M}$ . The admissible values of  $M$  are larger for pure noble gas mixtures than for mixtures with molecular additives, since the latter cool the electrons and suppress their correlations over time. The decorrelation number as implemented is slightly more complex than the simple picture above, applying several sequential sums at different integer multiples of the decorrelation distance, once electrons have travelled sufficiently far to reach a steady state behaviour. The decorrelation parameters can be set by hand, or set to zero in order to be assigned automatically by **PyBoltz**.

It is notable that the expression for diffusion in the  $z$  direction,  $D_z$ , requires prior knowledge of the drift velocity  $W_z$  in order to be calculated. This parameter is therefore only determined after the first two samples have been processed in order to estimate  $W_z$ . In each sample, the present best estimate of  $W_z$ , which we label  $\hat{W}_z$  is used in the calculation of  $D_z$ . Other quantities, for example, the mean electron energy, and the full diffusion tensor including correlations, are also calculated using a method similar to the one described above. Also accessible within the **PyBoltz** object are the collision-weighted energy spectrum and information about each individual collision.

### 3. **PyBoltz** Code Description

#### 3.1. Program flow and structure

As part of the refactorization from **FORTAN** into more modern languages, the structure of the code has been changed to reflect modular design principles. Program flow is handled by a central **PyBoltz** object, written and compiled in **Cython**. A user friendly wrapper script **PyBoltzRun** can optionally be used, which makes interactive passing and receiving of parameters more straightforward. Example scripts are provided for both wrapped and un-wrapped interface modes. There are independent modules for handling gas data (**Gases**) and Monte Carlo propagators (**Monte**). All variables have names in natural english and the majority of the code is extensively commented.

The flow of **PyBoltz** follows the flow of **Magboltz**. This starts by setting up the required global constants and values, such as the correlation length, and electron charge and mass, and so on (step 1). **PyBoltz** then estimates the appropriate energy binning for the specified gas mixture (steps 2 and 3). This is done by iteratively choosing an energy binning, calculating cross sections, propagating over a small simulation distances, and testing whether the electron energy spectrum

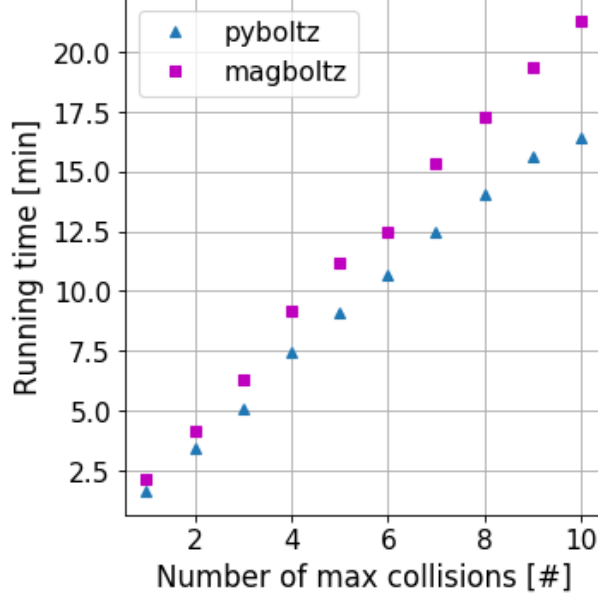


Figure 1: **PyBoltz** speed comparison to **MagBoltz**. The error on the running times was derived from repeating the measurement 5 times and is on the order of 100ms, which is too small to be visible above. The collision numbers here are reported in terms of the **MagBoltz** convention, as multiples of  $\eta_{coll} = 4 \times 10^7$ .

overflows the last bin. Once the energy scale has been determined, the **Mixer** object populates all binned cross section arrays appropriately. From the output of the gas mixing object, collision frequencies are extracted and stored in memory as members of the **PyBoltz** object. Finally, an appropriate Monte-Carlo integrator function is called from the **Monte** module to simulate a large number of collisions and to calculate drift properties from simulated collisions (step 4), using the algorithm outlined in the previous section. Outputs are returned as member variables of the **PyBoltz** object, or if using the **PyBoltzRun** wrapper, as named members in **Python** dictionaries.

### 3.2. Performance Testing

One motivation for choosing **FORTRAN** as a language for scientific computation has been the superior speed of execution afforded by such a low-level language. There are significant advantages to higher level languages like **Python** (when simple interfaces or interpretive execution are preferred), or **C++** (for codes with a complex, modular structure), especially for faster, modern computers, when performance allows. In the case of **MagBoltz**, the computations for gas mixtures of interest remain intensive, sometimes requiring several hours to scan the parameter points of interest on one CPU. This implies that obtaining optimal code performance is an important requirement when considering potential re-factorizations.

The **Cython** language is a hybrid that effectively compiles **Python**-like code into C, offering much of the flexibility and usability of **Python** with the improved performance of a compiled language. Using **Cython**, **PyBoltz** enjoys the combined benefits of a modular structure, **Python**-like code, and fast execution. The speeds of the **FORTRAN** and **Python** implementations were directly compared. For this purpose, a test system consisting of  $\text{CO}_2$  at 1 bar at an electric field of 1000V/cm was chosen. The results of this performance comparison are shown in Fig. 1. As demonstrated there,

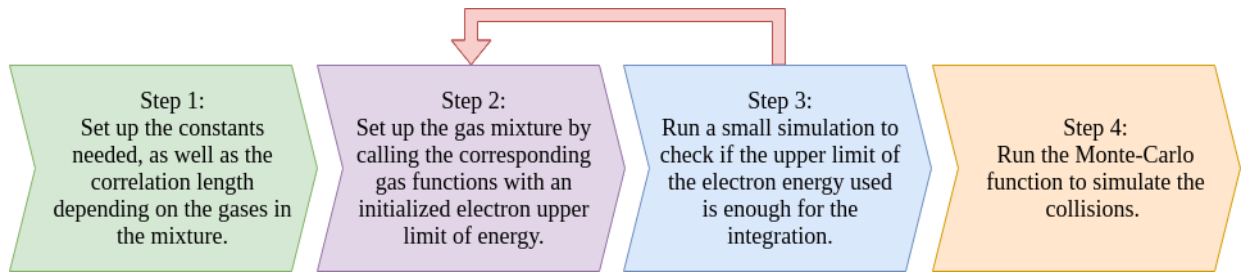


Figure 2: Simplified flow charge showing the PyBoltz / MagBoltz method of estimating the upper energy limit for binned evolution.

the `Cython` implementation outperforms the `FORTRAN` one with speed enhancements at the 20% level, independent from the number of collisions simulated. A similar trend has been observed consistently during various cross-checks of the two codes.

Even more important than speed is accuracy, and the two codes have been cross-compared against each other and against data for several systems of interest. We report results of these validations in Sec. 4.

#### 4. Validation with Transport data

In this section we compare the predictions of `PyBoltz` and `MagBoltz` Monte Carlo implementations with data taken in various gas mixtures and experimental configurations.

In time projection chambers, proportional counters, and other systems employing charge gain using noble gases, molecular additives are commonly used to quench VUV photons that can facilitate electrical breakdowns. A second property of molecular gases added to noble gases is that they cool the electrons during drift, via efficient transfer of excess energy to various rotational and vibrational degrees of freedom of the additive. This results in reduced diffusion constants relative to pure noble gases, and tunable drift velocity by up to two orders of magnitude. A wealth of experimental data exists on these mixtures, and we have picked three model systems with which to compare the accuracy of `PyBoltz` and `MagBoltz`.

A validation data set was chosen from Ref. [?], which contains measurements from two well studied gas mixtures: Ar-CH<sub>4</sub> (also known as P10), the gas mixture used in the STAR TPC, and Ne-CO<sub>2</sub>, the base gas mixture used in development of the ALICE experiment. Choosing these mixtures not only gives a robust data comparison but tests the performance of the code with mixed rather than single gases.

Simulations with both `MagBoltz` and `PyBoltz` were executed at 1 bar pressure over the range of electric fields (20-200 V/cm). This range was chosen because most gaseous TPC's operate in this region of reduced field (V/cm/bar). Irrespective from the operating pressure, scaling of drift-diffusion parameters with reduced field is generally satisfied in this regime [?], and so the results can be scaled to other pressures and drift fields using standard methods. In order to achieve accurate results the number of collisions was set to  $4 \times 10^8$  divided across ten samples, which was found sufficient to achieve convergence without an excessive run time. The gas mixtures were set to 90% Argon with 10% CH<sub>4</sub> and 91% Ne with 9% CO<sub>2</sub>, to match the reported fractions in the Ref. [?]. Both simulations agree to high degree of accuracy and both match the data sets within simulation error bars. The strong agreement in  $D_L$ ,  $D_T$  and  $V_z$  can be seen in Fig. 3.

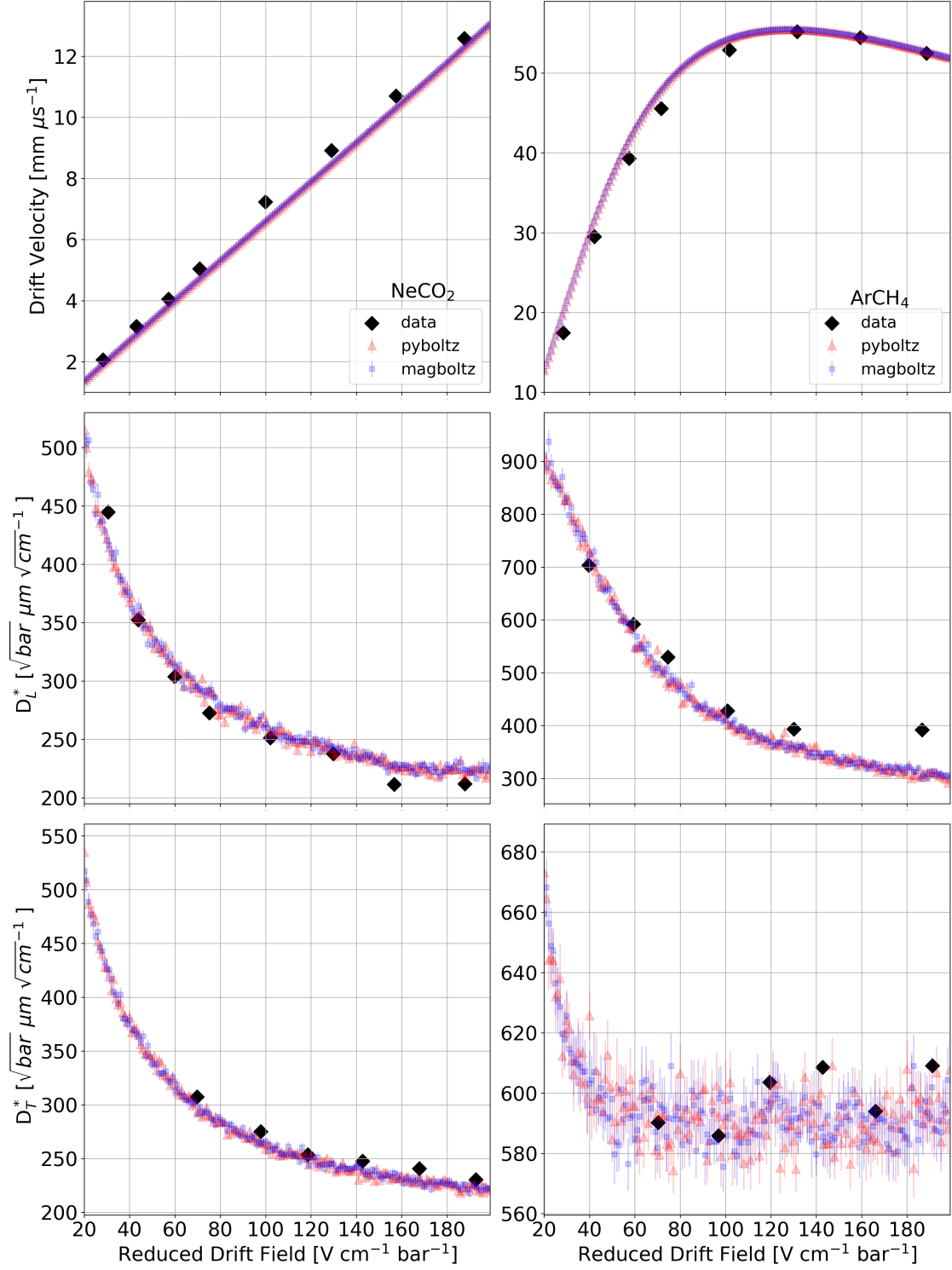


Figure 3: Left: the drift velocity, longitudinal and transverse diffusion of 91%Ne 9%CO<sub>2</sub>. Right: The drift velocity, longitudinal and transverse diffusion of 90%Argon 10%CH<sub>4</sub>. All calculations are at standard temperature and pressure. Data sets were taken from Ref [? ].

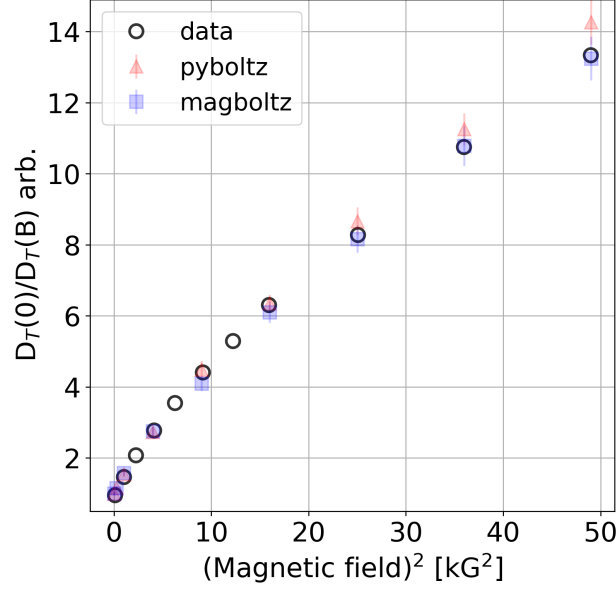


Figure 4: Validation of B-field suppression of transverse diffusion in Ar-CH<sub>4</sub> mixtures, **MagBoltz** and **PyBoltz** compared to data of Ref. [?] ]

For the case of electron transport in magnetic fields, data from Ref [?] ] were used as a benchmark. When a magnetic field is applied parallel or anti-parallel to the electric field, the diffusion in the transverse direction is reduced due to cyclotron motion of drifting electrons which prevents their lateral spreading [? ]. Reduction of the transverse diffusion is critical for retaining signal information when drifting electrons over a long distance in a TPC, and we consider this case here for illustration. The ratio  $\frac{D_T(0)}{D_T(B)}$  is used as a measure to quantify B-field assisted diffusion suppression. Simulations were run in a similar manner to the no B-field simulations, but using an electric field of 115 V/cm and a magnetic field in the  $z$  direction between 0 and 0.7 Tesla. The gas was set to 91%Argon with 9%CH<sub>4</sub> to match the measurement conditions of Ref [?] ]. The transverse diffusion in  $\frac{cm^2}{s}$  was taken from both programs and the ratio  $\frac{D_T(0)}{D_T(B)}$  calculated for each field point. Once again, the **PyBoltz** and **MagBoltz** calculations are consistent with each other and with measured data within statistical uncertainties, as shown in Fig 4.

## 5. Conclusion

The widely used and influential gaseous simulation detector code **Magboltz** has been refactored into **PyBoltz**, a **Cython** based code that has a modular structure allowing for improved flexibility and potentially a widened developer base. **PyBoltz** has demonstrated performance that is comparable to **MagBoltz** in both computation time and precision, with reproduction of results calculated with **MagBoltz** to a high degree of accuracy. The extendability of a modular, **Python**-based code has also opened up the potential to develop extension packages for new applications. We provide two examples in Appendix A of this work. We hope that in, the form of **PyBoltz**, the invaluable and seminal contributions made by **MagBoltz** to the field of gaseous detectors will continue as the code-base is embraced and extended by further generations of detector physicists.



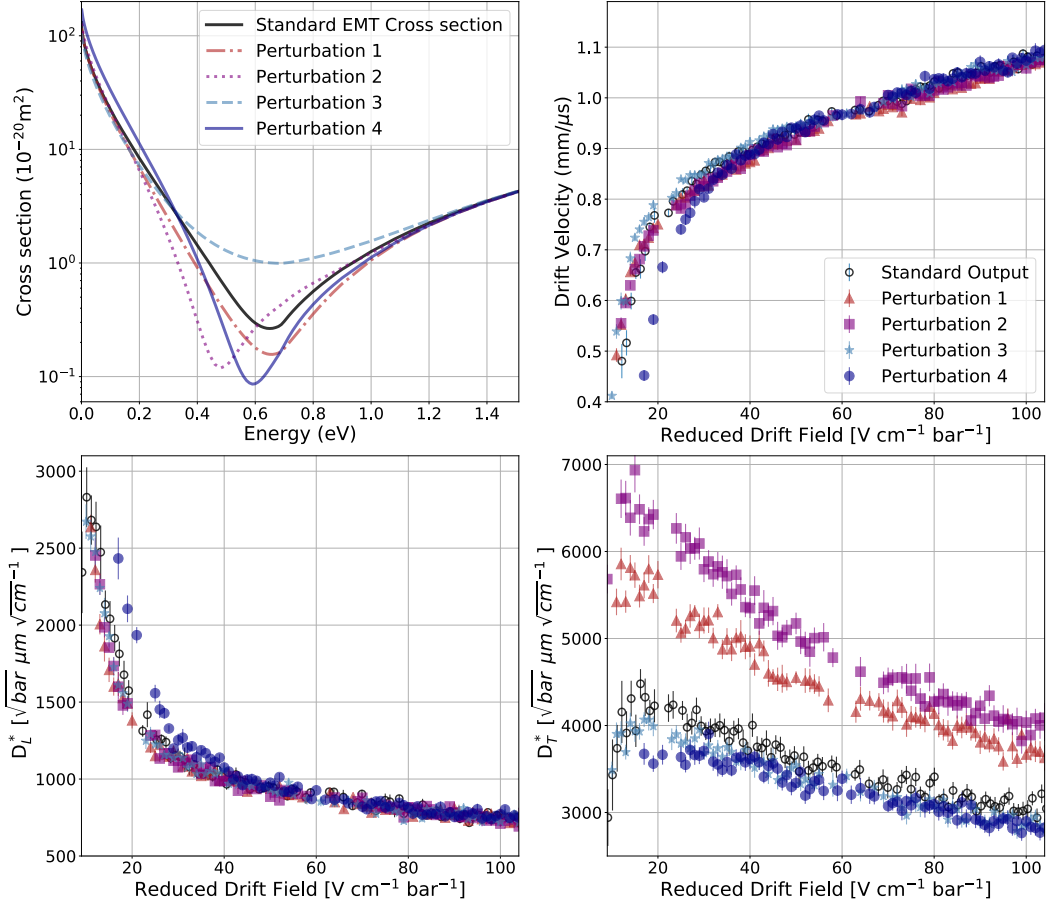


Figure 5: Top left: Default momentum transfer cross section in `PyBoltz` alongside different parametrizations consistent with the MERT formalism. Top right, bottom left, and bottom right display the effect of changing cross sections on drift velocity, longitudinal, and transverse diffusion respectively.

## Appendix A: Examples of Extended `PyBoltz` Applications

One goal of developing `PyBoltz` has been to modularize the code and enable extensibility. We present two examples in this section: first, the implementation of tunable cross sections with Modified Effective Range Theory (MERT); and second a wide exploration of properties of high pressure noble gases with additives (the “Plus Anything 2.0” program).

### 5.1. Tunable Cross Sections with Modified Effective Range Theory

Modified Effective Range Theory (MERT) utilizes the phase-shift representation of scattering cross sections in order to parameterize scattering behaviours that are consistent with angular momentum conservation in quantum mechanics. The original MERT method was developed by O’Malley [?], is explained in detail by Raju in Ref. [?], and has been used to obtain phenomenological fits to various low energy cross sections in various more recent works. While the cross sections can be fitted with electron beam data as in Ref. [?], it is also common to attempt to fit the cross section, using electron drift parameters as demonstrated in Refs. [? ? ?].

PyBoltz allows for scattering parametrizations such as MERT to be directly implemented into its modular structure. Our recent work has focused on testing perturbations of the xenon cross in the vicinity of its pronounced Ramsauer minimum, and will be the subject of future publications. The second order MERT formalism in [?] is presently available within PyBoltz, with modern extensions such as MERT5/6 from [?] being implemented for future releases. The zeroth and  $l$ 'th MERT phase shifts are described by:

$$\tan(\eta_0) = -Ak[1 + \frac{4\alpha}{3a_0}k^2\ln(3a_0)] - \frac{\pi\alpha}{3a_0}k^2 + Dk^3 + Fk^4 \quad (7)$$

$$\tan(\eta_1) = \frac{\pi}{15a_0}\alpha k^2[1 - (\frac{\varepsilon}{\varepsilon_1})^{\frac{1}{2}}] \quad (8)$$

$$\tan(\eta_l) = \pi\alpha k^2/[(2l+3)(2l+1)(2l-1)a_0] \quad (9)$$

where  $A$ ,  $D$ ,  $F$ ,  $\varepsilon_1$  are the adjustable MERT parameters,  $\alpha$  is the polarizability of the atom ( $27.292a_0^3$  for Xenon),  $a_0$  is the Bohr radius,  $k$  is the electron wavenumber that is related to the energy via  $\epsilon = 13.605(ka_0)^2$ ,  $l$  is the angular momentum quantum number, and  $\eta_0$ ,  $\eta_1$  represent the zeroth and first phase shift respectively. With this parameterization the momentum-transfer and total cross sections are given by:

$$\sigma_m = \frac{4\pi a_0^2}{k^2} \sum_{l=0}^{\infty} (l+1) \sin^2(\eta_l - \eta_{l+1}), \quad (10)$$

$$\sigma_t = \frac{4\pi a_0^2}{k^2} \sum_{l=0}^{\infty} (2l+1) \sin^2(\eta_l). \quad (11)$$

Given a set of MERT parameters, the elastic and momentum-transfer cross sections are calculated in PyBoltz in a consistent manner prior to swarm evolution. The cross section for xenon are only modified below 1 eV where MERT is applicable, and merged smoothly into the standard Biagi cross section at high energy, as described in Refs [? ?].

Simulating data sets in reduced field space with various cross sections allows to fit the cross sections with experimental data. A grid search over various cross sections can then be used to re-fit cross sections to data. An example showing the drift properties obtained with various MERT parameters is presented in Fig. 5.

## 5.2. Argon or Xenon “Plus Anything” 2.0

The PyBoltz code can be easily parallelized to run using distributed computing resources. In the past we explored combinations of Xenon “Plus Anything” gas mixtures, studying admixtures between  $10^{-6}\%$  to 20% in xenon at 10 bar and room temperature to survey the diffusion reducing properties of molecular and light noble gases [?]. Using our new package we have re-calculated these data points, which can be found at Ref. [?].

A recent application of high pressure gas mixtures concerns the DUNE MultiPurpose Detector [?]. R&D is required to optimize the gas mixture for this device, which is under way at various institutions, with the goal of enabling fast, low-diffusion, quenched and HV-stable drift of electrons in a gas of predominantly 10 bar argon. We have surveyed “Argon Plus Anything” gas mixtures at concentrations between  $10^{-6}\%$  to 20% of  $\text{CF}_4$ ,  $\text{CH}_4$ , DME,  $^4\text{He}$ ,  $\text{H}_2$ , Isobutane, Krypton, Neon, Propane and Xenon, among others. studying mean electron energy, drift velocity, longitudinal and transverse diffusion and attachment. The simulations were performed with electric fields between

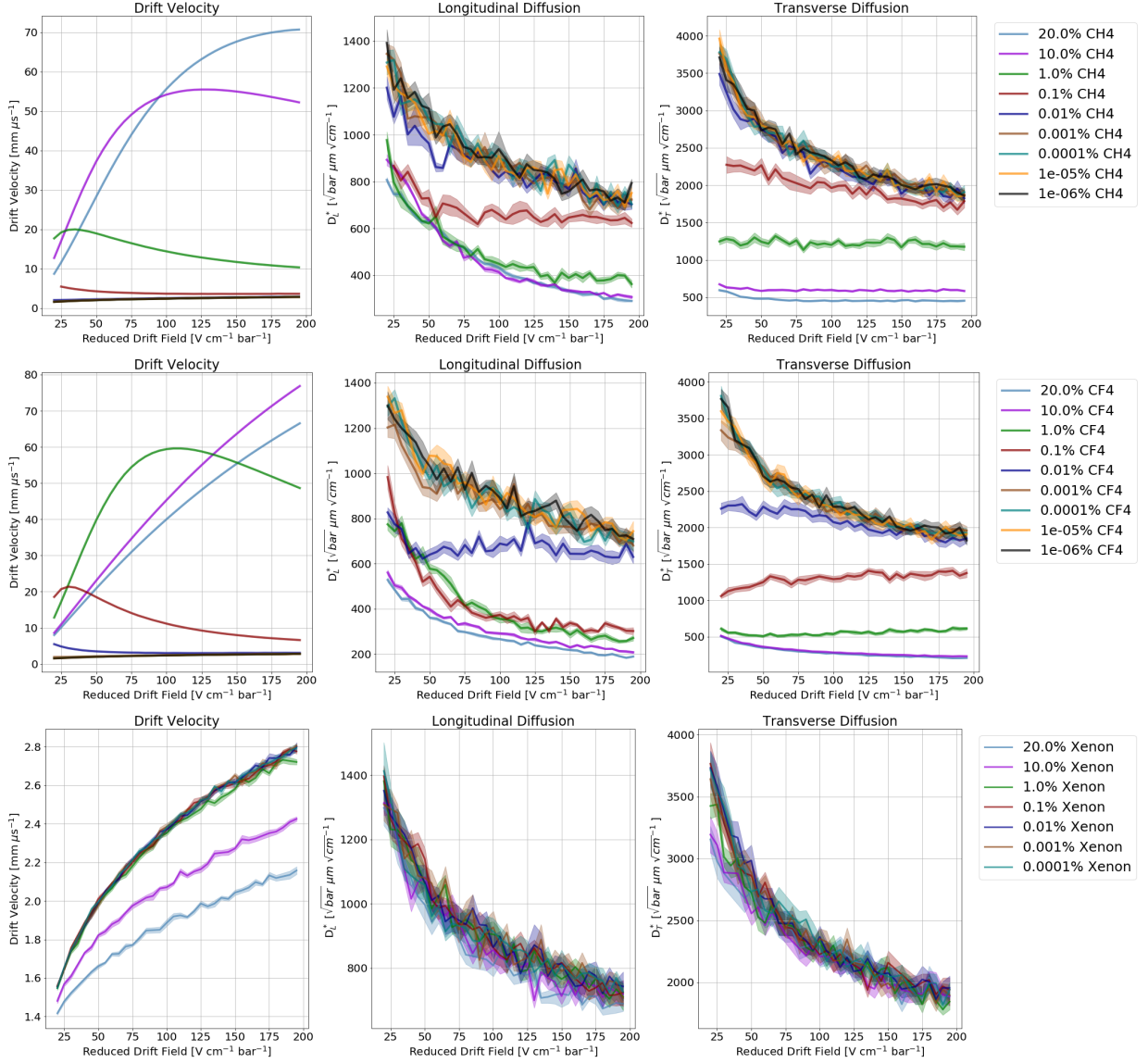


Figure 6: Example plots from the “Argon Plus Anything” project using PyBoltz. This example uses argon with various concentrations of CH<sub>4</sub>, CF<sub>4</sub> and xenon. We show here drift velocity, longitudinal and transverse diffusion coefficients.

20-200V/cm at a pressure of 1 bar, with simple extrapolation to any other field by the well known universality of transport parameters at a given reduced field  $E/N$  (this universality was verified in preliminary studies with a subset of mixtures). A compilation of the simulation results are available at [?], with a few examples shown in Fig. 6.

## Appendix B: Default Parameters in PyBoltz

Parameter	Default	Description
Gases	["NEON", "CO2"]	List of named gases to use.
Fractions	[90,10]	Percentages of each gas to use
Max_Collisions	$4 \times 10^7$	Total number of collisions to simulate
NumSamples	10	Number of points to statistically sample drift parameters
Temperature_C	23	Gas temperature in Celcius
Pressure_Torr	750.062	Gas pressure in Torr
BField_Tesla	0	Applied magnetic field in Tesla
BField_angle	0	Angle of magnetic to electric field
Enable_penning	0	Switch on or off Penning transfer effects
Enable_thermal_motion	1	Switch on or off thermal motion of the gas
ConsoleOutputFlag	0	Print output messages to console
Decor_Colls	0	Distance swarm must evolve to begin counting <sup>†</sup>
Decor_LookBacks	0	How many points to correlate electron with itself <sup>†</sup>
Decor_Step	0	Decorrelation number parameter M over which correlations vanish <sup>†</sup>
Max_electron_energy	0	Upper limit of binned energy spectrum <sup>†</sup>
		<sup>†</sup> implies automatically estimated if zero

## Acknowledgement

We would like to thank Roxanne Guenette and the Harvard FAS Research Computing center for the use of the Odyssey cluster, and members of the NEXT collaboration including Neus Lopez March and Ryan Felkai for illuminating conversations. The UTA group acknowledges support from the Department of Energy under Early Career Award number DE-SC0019054, and by Department of Energy Award DE-SC0019223. DGD acknowledges the Ramon y Cajal program (Spain) under contract number RYC-2015- 18820.

## References

### References

- [1] F. Sauli, Gaseous radiation detectors: fundamentals and applications, no. 36, Cambridge University Press, 2014.
- [2] D. Gonzalez-Diaz, F. Monrabal, S. Murphy, Gaseous and dual-phase time projection chambers for imaging rare processes, *Nucl. Instrum. Meth.* A878 (2018) 200–255. [arXiv:1710.01018](#), [doi:10.1016/j.nima.2017.09.024](#).
- [3] R. Veenhof, Numerical methods in the simulation of gas-based detectors, *JINST* 4 (2009) P12017. [doi:10.1088/1748-0221/4/12/P12017](#).
- [4] S. Biagi, Monte carlo simulation of electron drift and diffusion in counting gases under the influence of electric and magnetic fields, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 421 (1-2) (1999) 234–240.
- [5] H. Schindler, R. Veenhof, Garfield++, <https://garfieldpp.web.cern.ch/garfieldpp/>.
- [6] E. Ruiz-Choliz, et al., Modelling the behaviour of microbulk Micromegas in Xenon/trimethylamine gas, *Nucl. Instrum. Meth.* A799 (2015) 137–146. [arXiv:1506.05077](#), [doi:10.1016/j.nima.2015.07.062](#).
- [7] C. D. R. Azevedo, S. Biagi, R. Veenhof, P. M. Correia, A. L. M. Silva, L. F. N. D. Carramate, J. F. C. A. Veloso, Position resolution limits in pure noble gaseous detectors for X-ray energies from 1 to 60 keV, *Phys. Lett. B* 741 (2015) 272–275. [doi:10.1016/j.physletb.2014.12.054](#).
- [8] W. Riegler, C. Lippmann, R. Veenhof, Detector physics and simulation of resistive plate chambers, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 500 (1-3) (2003) 144–162.
- [9] J. Burns, T. Crane, A. C. Ezeribe, C. Grove, W. Lynch, A. Scarff, N. J. C. Spooner, C. Steer, Characterisation of Large Area THGEMs and Experimental Measurement of the Townsend Coefficients for CF<sub>4</sub>, *JINST* 12 (10) (2017) T10006. [arXiv:1708.03345](#), [doi:10.1088/1748-0221/12/10/T10006](#).
- [10] O. Sahin, T. Z. Kowalski, Measurements and calculations of gas gain in Xe5% TMA mixture-pressure scaling, *JINST* 13 (10) (2018) P10032. [doi:10.1088/1748-0221/13/10/P10032](#).
- [11] A. Simon, et al., Electron drift properties in high pressure gaseous xenon, *JINST* 13 (07) (2018) P07013. [arXiv:1804.01680](#), [doi:10.1088/1748-0221/13/07/P07013](#).
- [12] C. A. O. Henriques, et al., Secondary scintillation yield of xenon with sub-percent levels of CO<sub>2</sub> additive for rare-event detection, *Phys. Lett. B* 773 (2017) 663–671. [arXiv:1704.01623](#), [doi:10.1016/j.physletb.2017.09.017](#).
- [13] C. A. O. Henriques, et al., Electroluminescence TPCs at the Thermal Diffusion Limit, *JHEP* 01 (2019) 027. [arXiv:1806.05891](#), [doi:10.1007/JHEP01\(2019\)027](#).
- [14] C. D. R. Azevedo, L. M. P. Fernandes, E. D. C. Freitas, D. Gonzalez-Diaz, F. Monrabal, C. M. B. Monteiro, J. M. F. dos Santos, J. F. C. A. Veloso, J. J. Gomez-Cadenas, An homeopathic cure to pure Xenon large diffusion, *JINST* 11 (02) (2016) C02007. [arXiv:1511.07189](#), [doi:10.1088/1748-0221/11/02/C02007](#).
- [15] B. J. P. Jones, XePA Project: The Xenon Plus Anything Project, <http://www-hep.uta.edu/~bjones/XePA/> (2016).
- [16] R. Felkai, et al. (The NEXT collaboratio), Helium-Xenon mixtures to improve the topological signature in high pressure gas xenon TPCs, *Nucl. Instrum. Meth.* A905 (2018) 82–90. [arXiv:1710.05600](#), [doi:10.1016/j.nima.2018.07.013](#).
- [17] A. D. McDonald, et al., Electron Drift and Longitudinal Diffusion in High Pressure Xenon-Helium Gas Mixtures, *JINST* 14 (08) (2019) P08009. [arXiv:1902.05544](#), [doi:10.1088/1748-0221/14/08/P08009](#).
- [18] A. F. M. Fernandes, et al., Electroluminescence Yield in low-diffusion Xe-He gas mixtures (2019). [arXiv:1906.03984](#).
- [19] N. Lopez-March, The NEXT-DEMO++ detector: R&D for low diffusion mixtures, talk presented at LIDINE 2019, Manchester, UK.

- [20] S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. S. Seljebotn, K. Smith, Cython: The best of both worlds, *Computing in Science & Engineering* 13 (2) (2011) 31.
- [21] B. Al Atoum, B. J. P. Jones, A. D. McDonald, Pyboltz, <https://github.com/UTA-REST/PyBoltz>.
- [22] S. Biagi, Accurate solution of the boltzmann transport equation, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 273 (2-3) (1988) 533–535.
- [23] S. Biagi, A multiterm boltzmann analysis of drift velocity, diffusion, gain and magnetic-field effects in argon-methane-water-vapour mixtures, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 283 (3) (1989) 716–722.
- [24] G. Fraser, E. Mathieson, Monte carlo calculation of electron transport coefficients in counting gas mixtures: I. argon-methane mixtures, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 247 (3) (1986) 544–565.
- [25] H. Skullerud, The stochastic computer simulation of ion motion in a gas subjected to a constant electric field, *Journal of Physics D: Applied Physics* 1 (11) (1968) 1567.
- [26] A. Okhrimovskyy, A. Bogaerts, R. Gijbels, Electron anisotropic scattering in gases: A formula for monte carlo simulations, *Physical Review E* 65 (3) (2002) 037402.
- [27] M. Capitelli, C. Gorse, S. Longo, D. Giordano, Collision integrals of high-temperature air species, *Journal of thermophysics and heat transfer* 14 (2) (2000) 259–268.
- [28] R. Brockmann, et al., Development of a time projection chamber with high two track resolution capability for collider experiments: RD-32 status report (1994).
- [29] S. Amendolia, M. Binder, W. Blum, M. Cherney, A. Farilla, F. Fidecaro, J. Izen, R. Jared, I. Lehraus, P. Marrocchesi, et al., Dependence of the transverse diffusion of drifting electrons on magnetic field, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 244 (3) (1986) 516–520.
- [30] J. N. Marx, D. R. Nygren, The Time Projection Chamber, *Phys. Today* 31N10 (1978) 46–53. doi:10.1063/1.2994775.
- [31] T. F. O’Malley, Extrapolation of electron-rare gas atom cross sections to zero energy, *Physical Review* 130 (3) (1963) 1020.
- [32] G. G. Raju, *Gaseous electronics: theory and practice*, CRC Press, 2005.
- [33] M. Kurokawa, M. Kitajima, K. Toyoshima, T. Kishino, T. Odagiri, H. Kato, M. Hoshino, H. Tanaka, K. Ito, High-resolution total-cross-section measurements for electron scattering from ar, kr, and xe employing a threshold-photoelectron source, *Physical Review A* 84 (6) (2011) 062717.
- [34] S. Hunter, J. Carter, L. Christophorou, Low-energy electron drift and scattering in krypton and xenon, *Phys. Rev. A; (United States)* 11 (38) (12 1988). doi:10.1103/PhysRevA.38.5539.
- [35] J. Pack, R. Voshall, A. Phelps, L. Kline, Longitudinal electron diffusion coefficients in gases: Noble gases, *Journal of applied physics* 71 (11) (1992) 5363–5371.
- [36] B. J. P. Jones, A. D. McDonald, B. Al Atoum, Argon and xenon plus anything, [https://github.com/UTA-REST/ArXe\\_plus\\_anything](https://github.com/UTA-REST/ArXe_plus_anything).
- [37] J. Martin-Albo, A pressurized argon gas TPC as DUNE near detector, *J. Phys. Conf. Ser.* 888 (1) (2017) 012154. arXiv:1610.07803, doi:10.1088/1742-6596/888/1/012154.