PyBoltz – a New Python Swarm Simulation Code

B. Al Atoum. S. Biagi, D. Gozalez Diaz, B.J.P. Jones and A.D.McDonald

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

B. Al Atoum[a], S. Biagi[b], D. González-Díaz[c], B.J.P. Jones[a], A.D. McDonald[a]

[a] *Department of Physics, University of Texas at Arlington, Arlington, TX 76019, USA*
[b] *University of Liverpool, Physics Department, Liverpool L69 7ZE, United Kingdom*
[c] *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.

To arXiv today

Electron Drift and Longitudinal Diffusion in High Pressure Xenon-Helium Gas Mixtures

---

The NEXT Collaboration

A.D. McDonald,[1,a] K. Woodruff,[1] B. Al Atoum,[1] D. González-Díaz,[2] B.J.P. Jones,[1] C. Adams,[10]

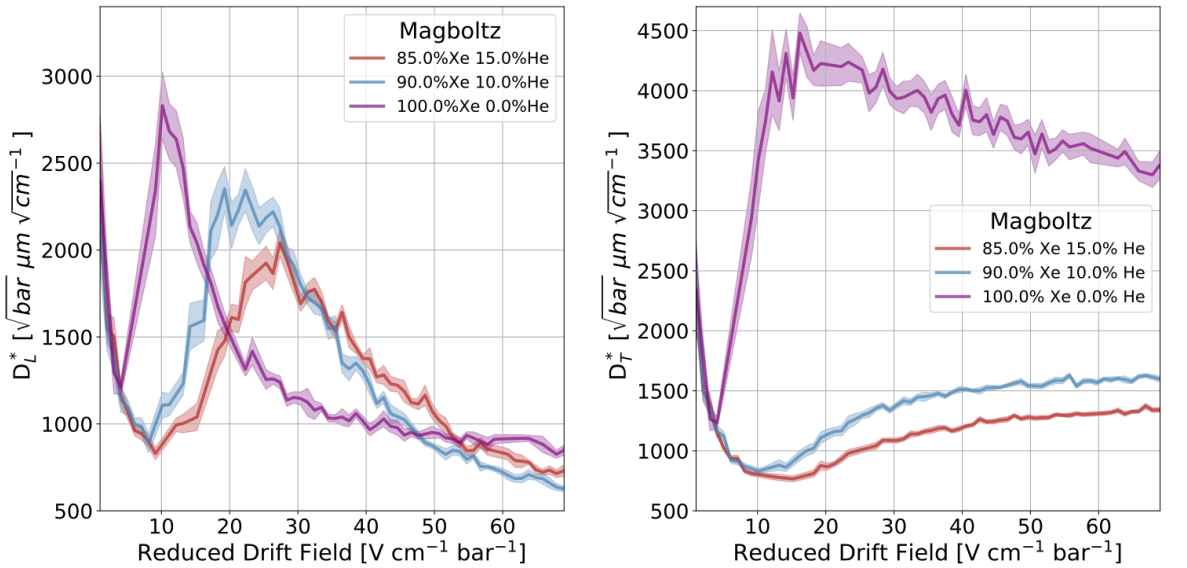


Figure 1. MagBoltz predictions of the dependence of the reduced longitudinal (left) and transverse (right) diffusion coefficients(equation 3.1) on helium concentration. The line represents the exact value from the simulation and the shaded region is the error in the simulation.

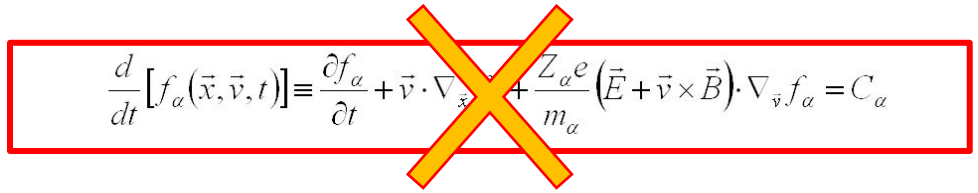
For example, JINST 14 (2019) no.08, P08009

What does MagBoltz Do?

- Predicts electron transport properties including velocity, longitudinal and transverse diffusion, attachment, Townsend coefficients, mean energy, for electron swarms drifting in gas mixtures.

- Since ~1999 MagBoltz has been a workhorse code for the field of gas detectors

What does MagBoltz Do?



$$\frac{d}{dt}[f_\alpha(\vec{x},\vec{v},t)] \equiv \frac{\partial f_\alpha}{\partial t} + \vec{v}\cdot\nabla_{\vec{x}} + \frac{Z_\alpha e}{m_\alpha}\left(\vec{E}+\vec{v}\times\vec{B}\right)\cdot\nabla_{\vec{v}} f_\alpha = C_\alpha$$

Monte Carlo calculation of electron transport coefficients in counting gas mixtures: II. Mixtures containing neon and carbon dioxide

G.W. Fraser, E. Mathieson

⊞ Show more

https://doi.org/10.1016/0168-9002(86)90418-3 Get rights and content

- You would be forgiven for thinking that MagBoltz has something to do with the Boltzmann Equation, potentially in a magnetic field.
- Actually this has not been true since early releases in 1990s.
- Since 2001 it uses methods of Frasier and Mathieson and collision-by-collision simulations.

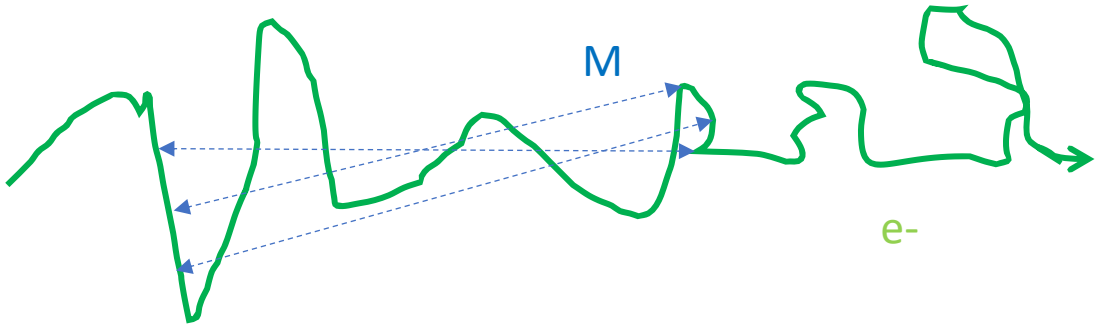
Basic idea

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

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

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

Decorrelation distance M



- A well-simulated electron will explore configuration space ergodically, and eventually be uncorrelated with its past self.
- Measuring separation between an electron and itself far enough back in time is approximately the same as measuring separation between two different electrons.
- Thus MagBoltz makes a swarm prediction by simulating just one electron, for a lot of collisions (typically tens of millions).
- Cross sections must include energy dependent elastic, excitation, ionization, attachment, and have appropriate angular distributions.
- Simulations possible in generic configuration of E and B fields (which influence how the electron moves between collisions.

The Real Magic

- 1) Doing it all very fast
  - *Skulleruds Null Collision method*
  - *… and other tricks*
- 2) Ensuring a comprehensive database of accurate cross sections
  - *Periodically uploaded to LXCat, tuned by magic hands of S. Biagi*

The stochastic computer simulation of ion motion in a gas subjected to a constant electric field

H. R. SKULLERUD

Electron and Ion Physics Research Group, Physics Department, Norges Tekniske Høgskole, Trondheim, Norway

MS. received 24th July 1968

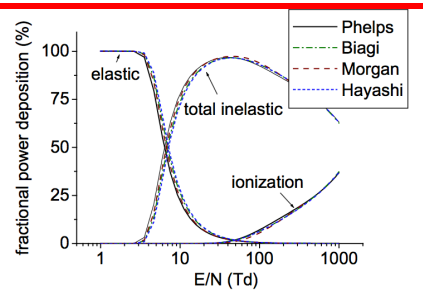


Fig. 3. Fractional power deposited into elastic, inelastic and ionization in argon vs *E/N* calculated using the different cross section sets as indicated.

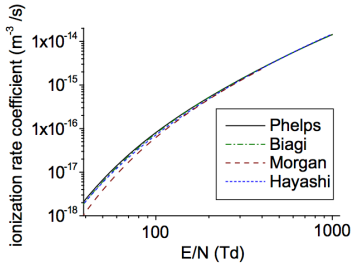


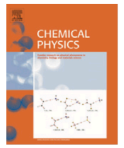
Fig. 4. Same as for Figs. 1 and 2, but showing the calculated ionization rate coefficients.

Chemical Physics 398 (2012) 148–153

Contents lists available at ScienceDirect

Chemical Physics

journal homepage: www.elsevier.com/locate/chemphys



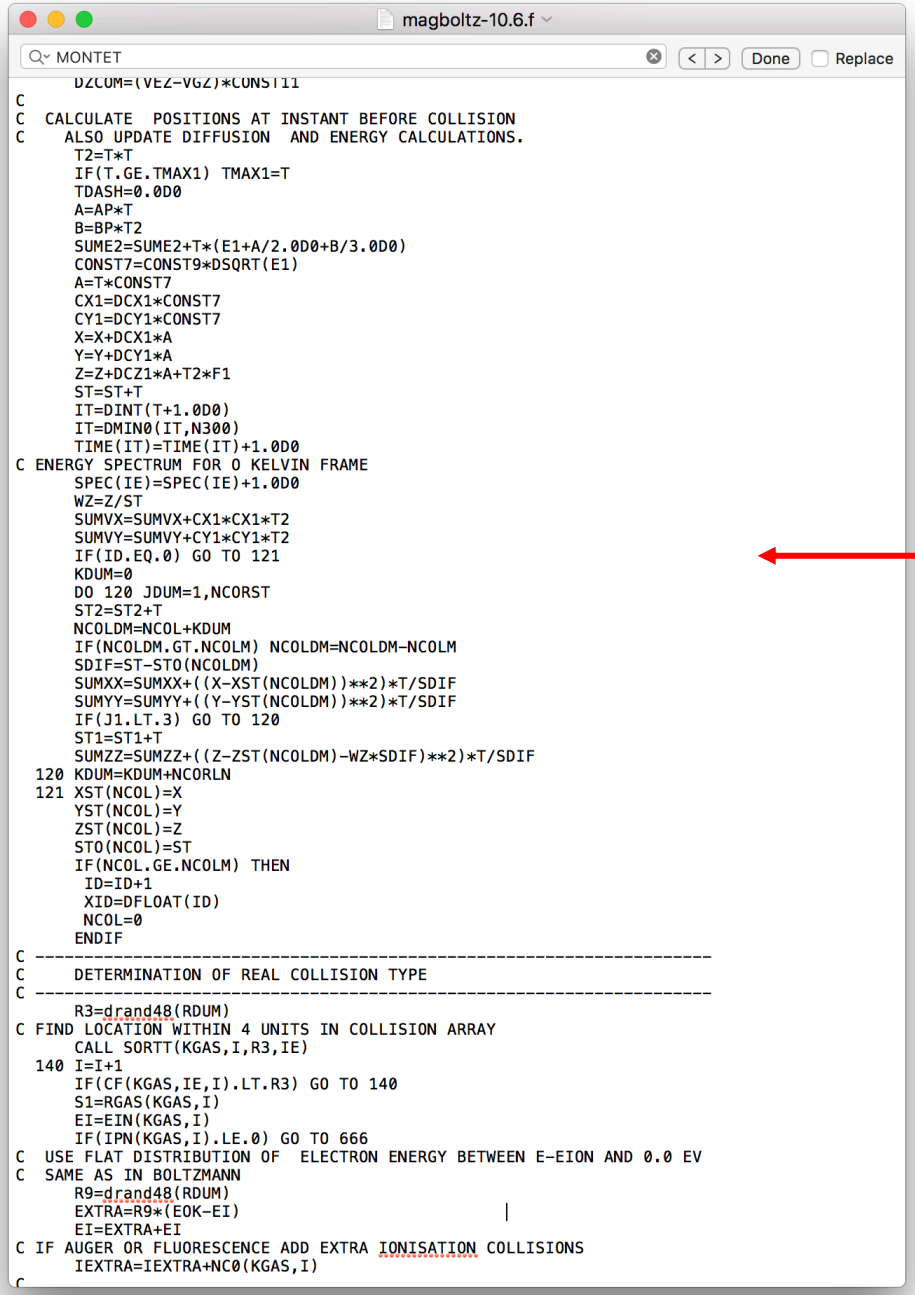
The LXCat project: Electron scattering cross sections and swarm parameters for low temperature plasma modeling

S. Pancheshnyi [a,b], S. Biagi [c], M.C. Bordage [a,b], G.J.M. Hagelaar [a,b], W.L. Morgan [d], A.V. Phelps [e], L.C. Pitchford [a,b,*]

[a] *Université de Toulouse, UPS, INPT, LAPLACE (Laboratoire Plasma et Conversion d'Energie), 118 route de Narbonne, F-31062 Toulouse cedex 9, France*
[b] *CNRS, LAPLACE, F-31062 Toulouse, France*
[c] *University of Liverpool, Oliver Lodge Laboratory, Oxford St., Liverpool L69 7ZE, UK*
[d] *Kinema Research and Software, P.O. Box 1146, Monument, CO 80132, USA*
[e] *JILA, NIST and University of Colorado, Boulder, CO 80309-0440, USA*

All the lines marked "Biagi" in LXCat are MagBoltz exported cross sections

https://fr.lxcat.net/instructions/categories.php



The Down-side of MagBoltz

55990 lines of FORTRAN

Everything hard coded and cryptically named

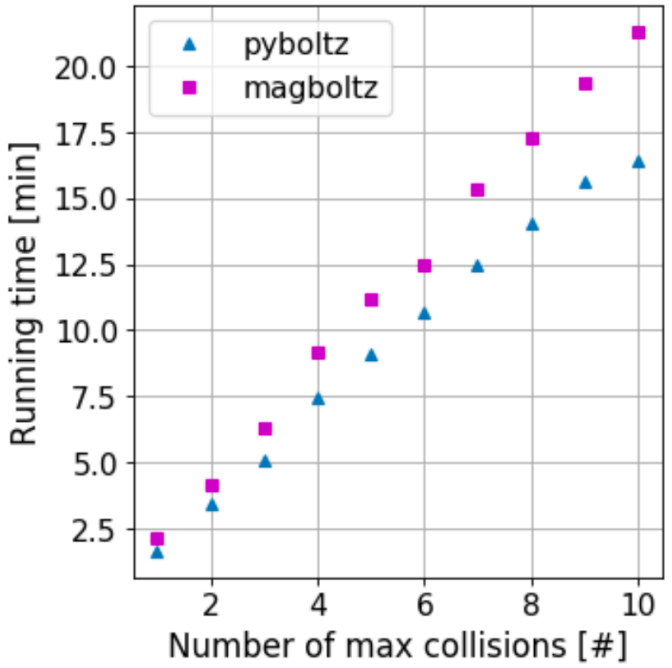
Easy enough to run

But almost impossible to modify / extend / interface / understand.

(Honorable mention: heroic attempts in interfacing w/ Garfield++)

PyBoltz: The New MagBoltz

- Full refactorization of entire MagBoltz code base into Python
- Modular structure, English variable names.
- Maintains performance (with a little help from Cython)




```
# We run collisions in NumSamples batches,
# evenly distributed between its MaxNumberOfCollisions collisions.
CollisionsPerSample = <long long> (Object.MaxNumberOfCollisions / Object.NumSamples)
for iSample in range(int(Object.NumSamples)):
    for iCollision in range(int(CollisionsPerSample)):
        while True:

            # Sample random time to next collision. T is global total time.
            RandomNum = random_uniform(RandomSeed)

            # This is the formula from Skullerud
            T = -log(RandomNum) / Object.MaxCollisionFreqTotal + TDash
            TDash = T

            # Apply acceleration.
            #
            #     VBefore = VBefore + a t
            #      EAfter = 1/2 m VAfter^2
            #             = 1/2 m(VAfterX^2 + VAfterY^2 + VAfterZ^2)
            #             = 1/2 m((VBeforeZ + at)^2 + VBeforeX^2 + VBeforeY^2)
            #             = EBefore + (dir_z)(AP + BP * T) * T
            #
            #  w/      AP = m VBefore a
            #          BP = 1/2 m a^2
            #
            # So here, F2 = sqrt(m / 2) EField e
            #          BP = 1/2 m EField^2 e^2
            #

            AP = DirCosineZ1 * F2 * sqrt(EBefore)
            EAfter = EBefore + (AP + BP * T) * T
            VelocityRatio = sqrt(EBefore / EAfter)
```

UTA-REST / PyBoltz

Unwatch 2 | Star 0 | Fork 1

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Security | Insights | Settings

Branch: Cython | PyBoltz / src / Cython / | Create new file | Upload files | Find file | History

Benjamin Jones and Benjamin Jones A few comments — Latest commit db8af3b 2 days ago

..		
C	Create README.md	13 days ago
Gases	Fixed isobutane	8 days ago
Monte	A few comments	2 days ago

https://github.com/UTA-REST/PyBoltz

README.md

PyBoltz

This software package is a translation of the Fortran based Magboltz code (Biagi, 2001) into Cython. This project was built to allow for more productive work to be done with magboltz.

General information.

About Magboltz.

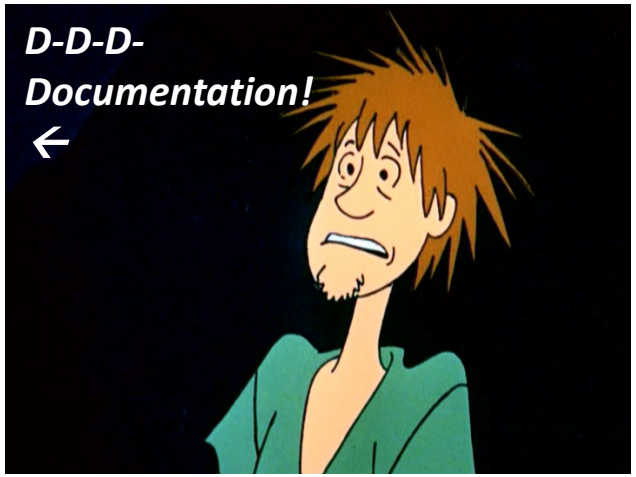
The Magboltz program computes drift gas properties by "numerically integrating the Boltzmann transport equation"-- i.e., simulating an electron bouncing around inside a gas. By tracking how far the virtual electron propagates, the program can compute the drift velocity. By including a magnetic field, the program can also calculate the Lorentz angle. Read more.

Why Cython?

Cython's static typing improves the speed of python code by about a hundred times. In other words, Cython provides us with the simplicity of python and the speed of Fortran/C. Read more.

Setting up and running instructions.

Setting up.



PyBoltz has extensive GitHub readme and full Sphinx documentation.

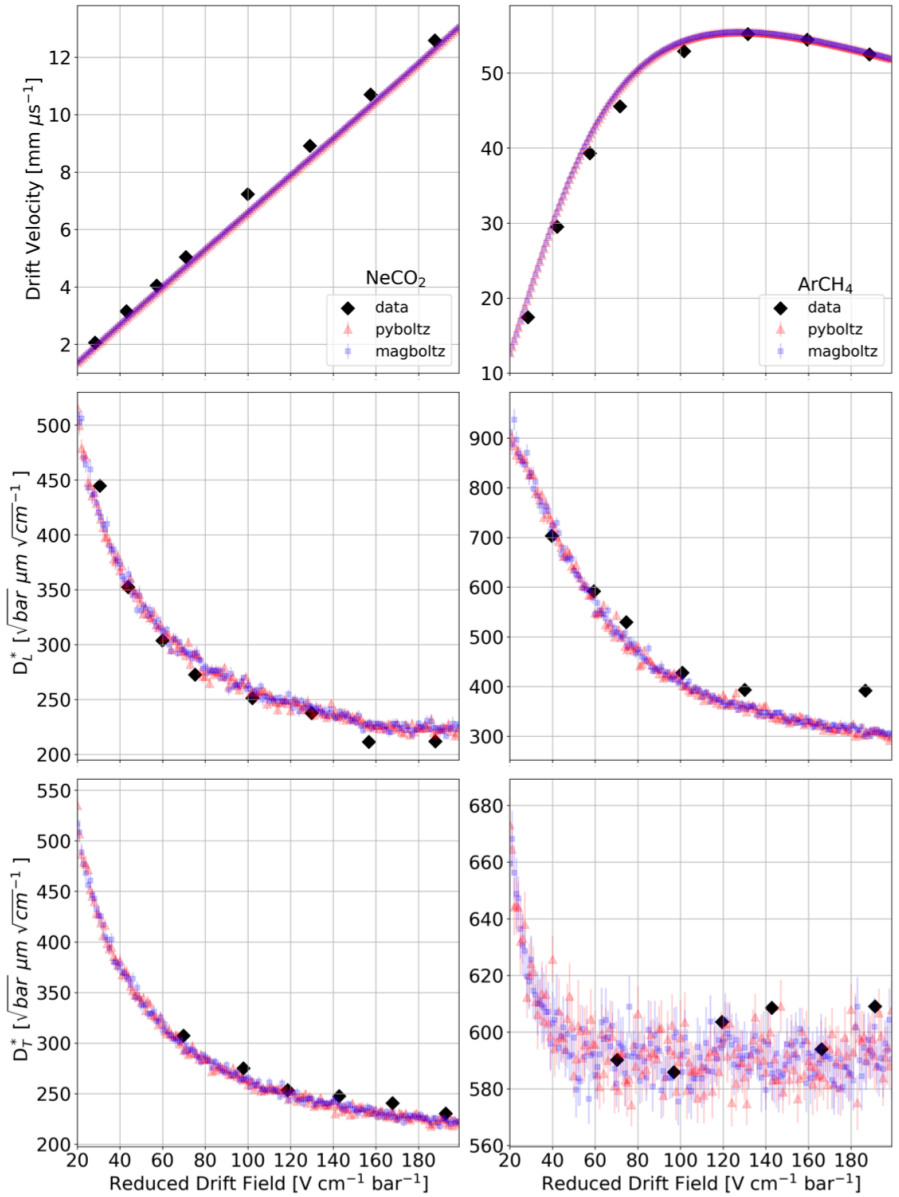


Figure 3: Left: the drift velocity, longitudinal and transverse diffusion of 91%Ne 9%CO_2. Right:The drift velocity, longitudinal and transverse diffusion of 90%Argon 10%CH_4. All calculations are at standard temperature and pressure. Data sets were taken from Ref [28].

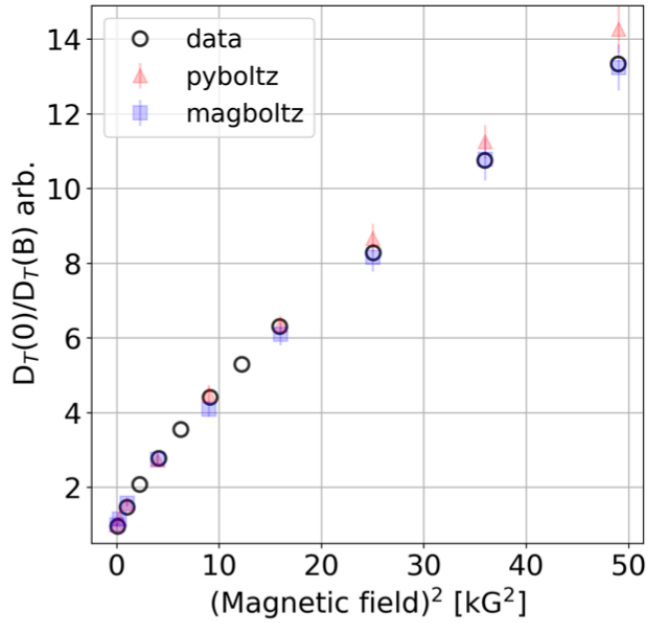


Figure 4: Validation of B-field suppression of transverse diffusion in Ar-CH_4 mixtures, `MagBoltz` and `PyBoltz` compared to data of Ref. [29]

Validated against MagBoltz and against data with and without B fields in complex gas mixtures

- Finally, we can ***develop*** for MagBoltz.
- Our first extension is incorporation of modified effective range theory for cross section tuning.

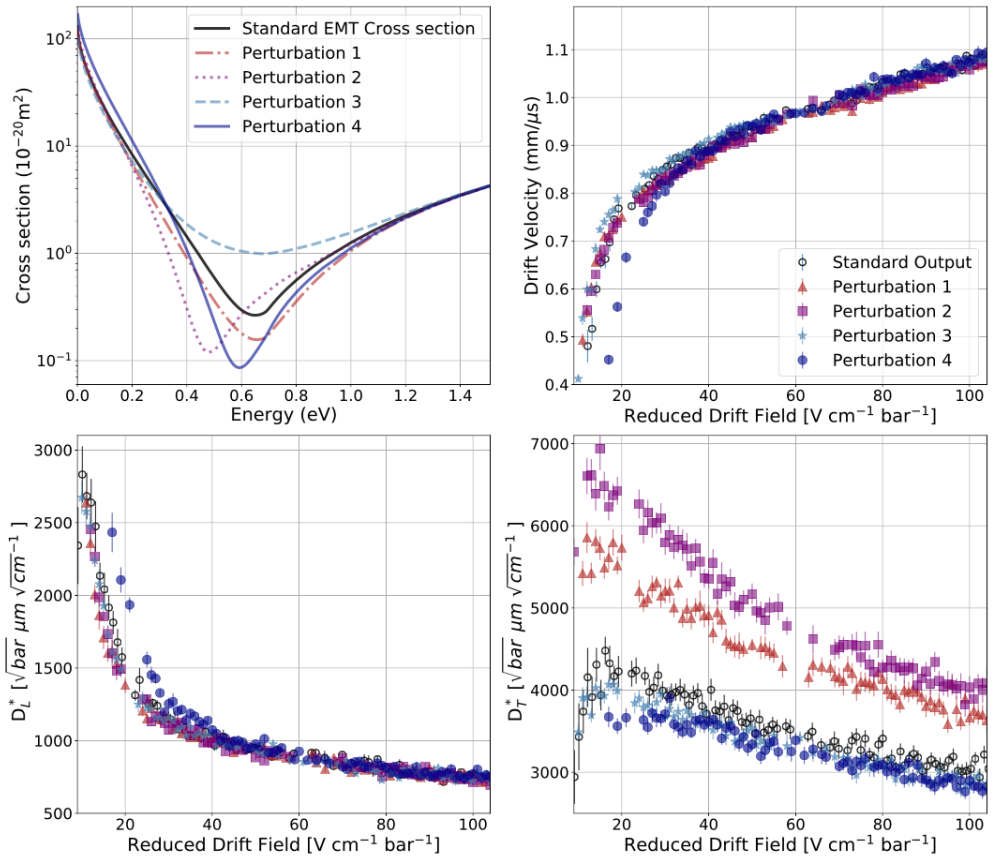


Figure 5: Top left: Default momentum transfer cross section in `PyBoltz` alongside different parametrizations consistent with the MERT formalism. Top right, bottom left, and bottom right display the effect of changing cross sections on drift velocity, longitudinal, and transverse diffusion respectively.

Method of cross section parametrization consistent with angular momentum conservation in quantum mechanics.

Spherical wave phase shifts

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

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

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

Momentum transfer and total cross sections

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

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

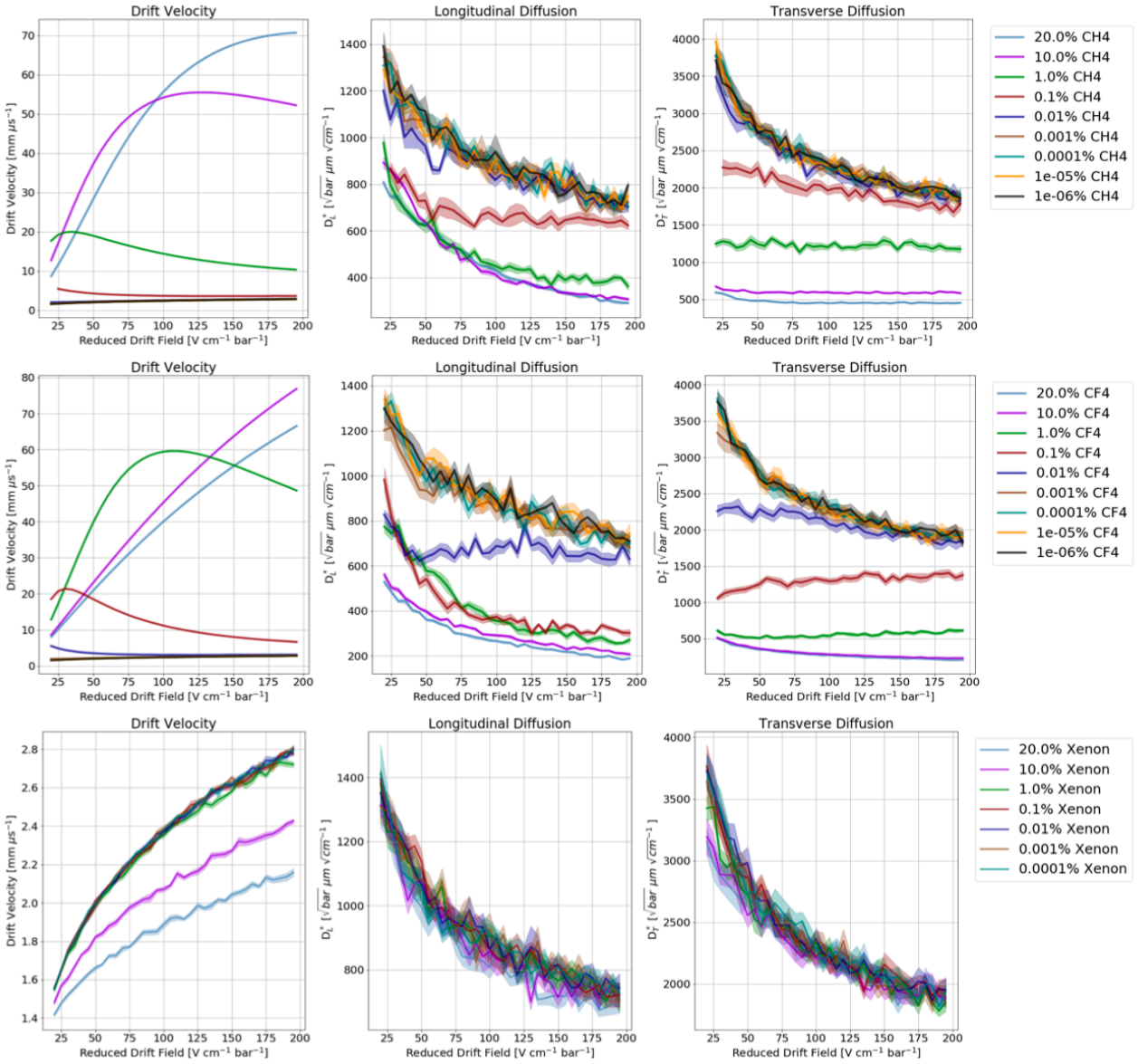
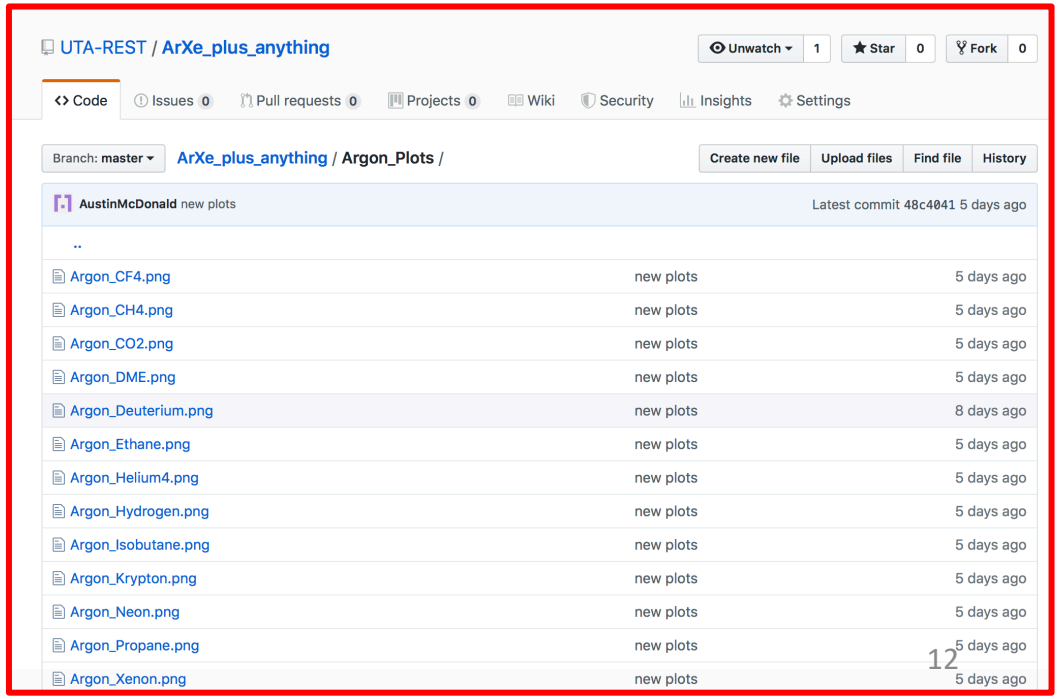


Figure 6: Example plots from the "Argon Plus Anything" project using PyBoltz. This example uses argon with various concentrations of CH_4, CF_4 and xenon. We show here drift velocity, longitudinal and transverse diffusion coefficients.

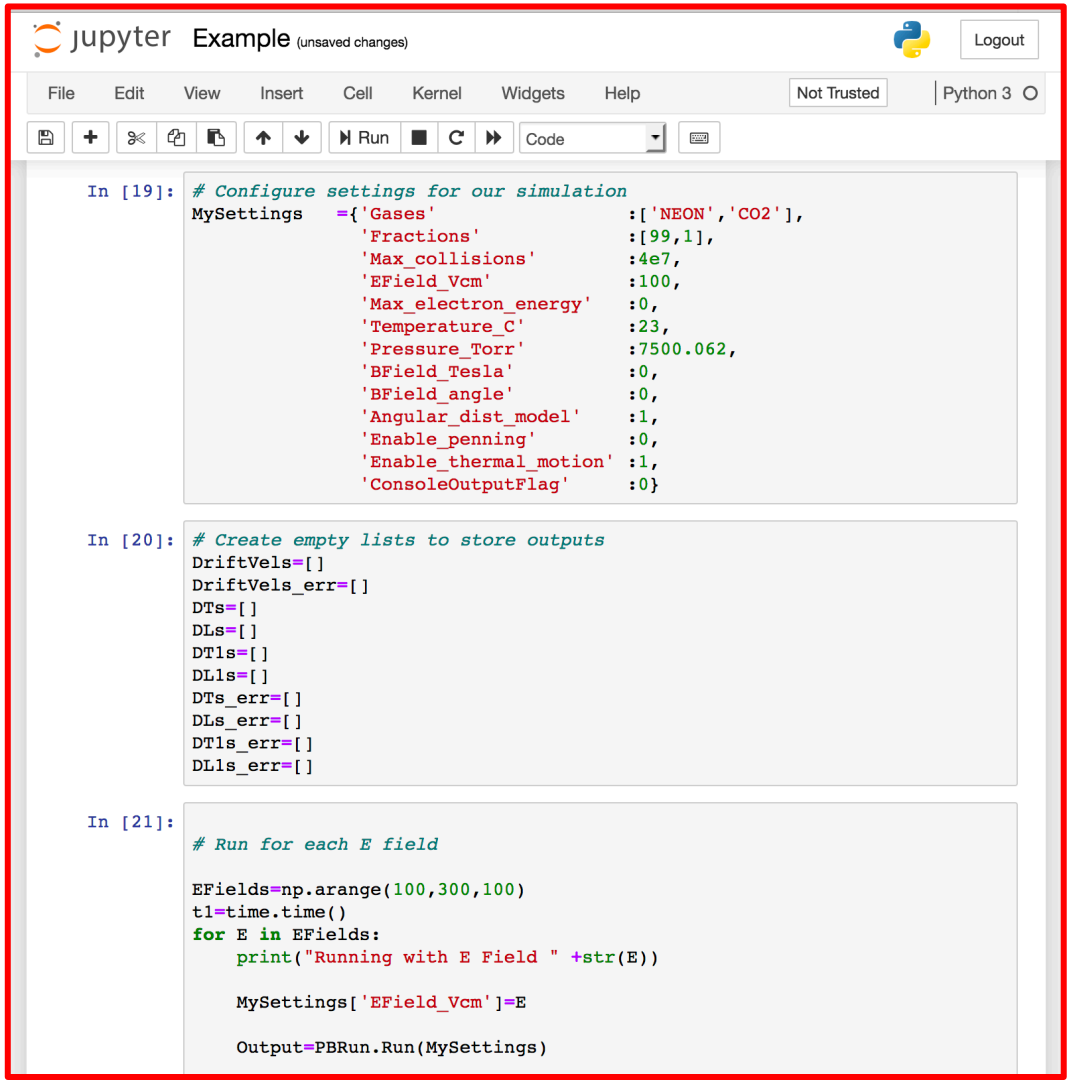
Argon- and Xenon- Plus Anything

- Rapid-fire exploration of diffusion reducing gas mixtures based on argon and xenon.
- Jumping off point for further UTA gas mixture studies targeted at DUNE MPD and NEXT.



Getting started

- Easy-to-use examples based on Jupyer notebooks and raw python in the Examples folder.
- We will support new users to get started – the sooner the better!
- Also open to ideas, thoughts, questions about interesting gas simulation studies for DUNE MPD. Please reach out if you have ideas!



Thanks!