




# chombo-discharge: An adaptive code for gas discharge simulations in complex geometries

Robert Marskar <sup>1</sup>

<sup>1</sup> SINTEF Energy Research, Norway

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Open Journals](#) 

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## In partnership with



AMERICAN  
ASTRONOMICAL  
SOCIETY

This article and software are linked with research article DOI [10.3847/xxxxx](https://doi.org/10.3847/xxxxx) <- [update this with the DOI from AAS once you know it.](#), published in the *Astrophysical Journal* <- The name of the AAS journal..

## Summary

chombo-discharge is 2D and 3D adaptive code for simulating low-temperature gas discharges in complex geometries. Such discharges occur when electrons accelerate in strong electric fields and ionize the gas, and further evolution is often determined by the space charge set up by electrons and ions. Streamers, for example, are filamentary plasma dominated by space charge effects and are precursors to leader, sparks, and lightning.

Gas discharge problems typically involve simulations over multiple scales in time and space. chombo-discharge reduces the cost of such simulations by using Cartesian Adaptive Mesh Refinement (AMR). It also provides support for multi-material complex geometries (gas phase, electrodes, and solid dielectrics) through an embedded boundary (EB) formulation. The code uses a solver-centered modular design where larger applications are constructed by strongly or weakly coupling numerical solvers shipped with the chombo-discharge base code. Depending on their needs, users can therefore enter the code through several layers: E.g. they only need to learn pre-existing text interfaces when using existing applications; use solver C++ APIs when typing up new applications, or use the EB AMR infrastructure when writing entirely new solvers.

Several solvers exist in chombo-discharge, all of which are parallelized and compatible with EBs and AMR:

- Advection-diffusion-reaction solvers.
- Helmholtz equation solvers, using geometric multigrid.
- Electrostatic solvers (with support for discontinuous coefficients).
- Kinetic Monte Carlo solvers.
- Radiative transfer solvers.
- Various particle solvers, e.g. for Monte Carlo radiative transfer, tracer particles, Brownian walkers, or kinetic Particle-In-Cell.
- Volumetric and cut-cell ODE solvers.

All solvers exist as stand-alone applications, but many of these solvers are also coupled through more complex physics applications that aim at resolving different types of discharge phenomena (e.g. statistical inception models or streamer discharges). The interaction of these solvers occurs through a common AMR core, which can also use dual grids where e.g. fluid and particle kernels are load-balanced separately. chombo-discharge uses the Chombo infrastructure for the AMR and EB infrastructure, and is parallelized using MPI. However, chombo-discharge supplies its own solver discretizations.

## Statement of need

There is already a number of discharge simulation codes currently available. Commercial codes used for simulating discharge include COMSOL, ANSYS Fluent, OpenFOAM, PLASIMO,

41 VSIM/VPIC, and Vizglow. Many non-commercial codes also exist, such as Afivo-streamer  
42 (Teunissen & Ebert, 2017), which also uses Cartesian AMR.

43 While chombo-discharge is not the only open-source discharge simulation code, it has a  
44 number of unique features. In particular, chombo-discharge supports complex geometries  
45 and is therefore useful for lightning initiation investigations (from hydrometeors), in high-  
46 voltage technology, plasma medicine and plasma-assisted combustion. The code is also quite  
47 performant, and its design pattern permits a flexible and extensible approach to numerically  
48 solving various discharge problems, even when these require many thousands of CPU cores.  
49 For example, coupling a drift-diffusion Particle-In-Cell model together with an electrostatics  
50 and Monte Carlo based radiative transfer solvers required about 3500 lines of C++ code, and  
51 this includes the work for dual-grid load-balancing up to at least 8000 CPU cores, and all  
52 integration and I/O functionality.

### 53 Application examples

54 Originally, the code was written for studying pre-breakdown phenomena in high-voltage  
55 equipment (Marskar, 2019), but over time it has been adapted to fit a wider category of  
56 discharge problems.

- 57 ■ Studying upper atmospheric lightning (sprite discharges).

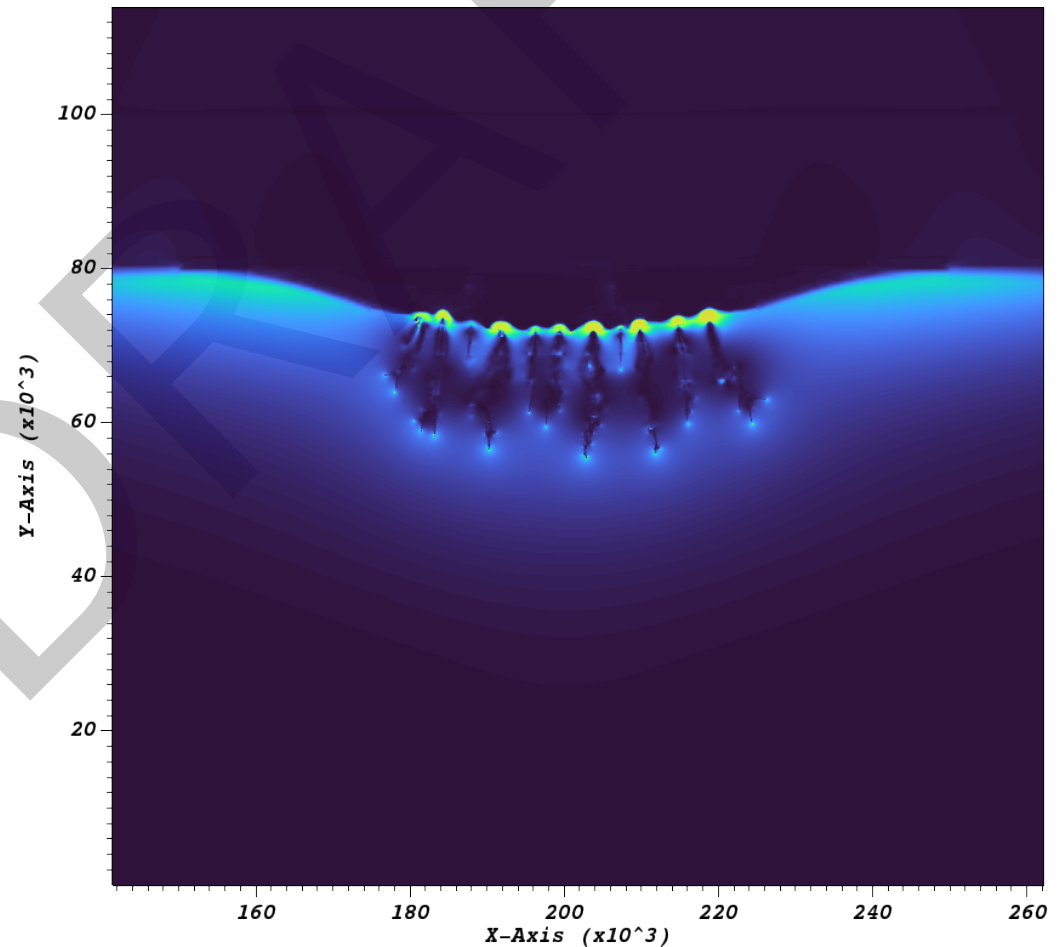


Figure 1: Simulation of laboratory-scale streamer corona

- 58 ■ Simulation of laboratory-scale using non-kinetic Particle-In-Cell.

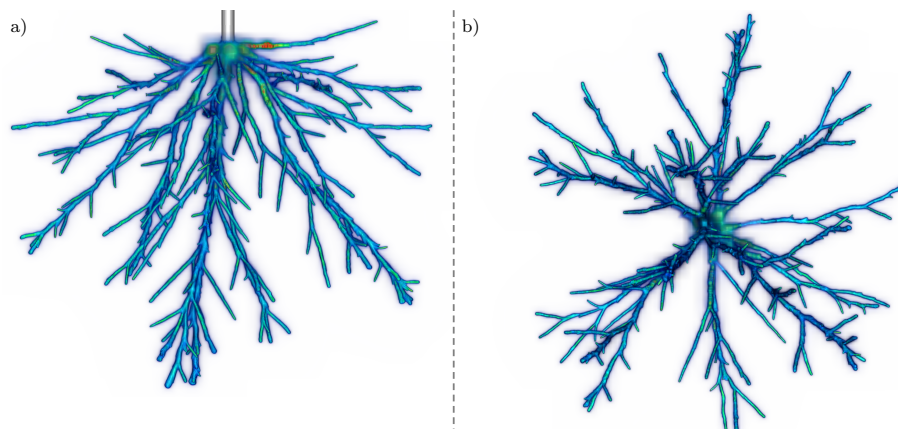


Figure 2: Simulation of laboratory-scale streamer corona

- 59 ■ Surface discharge evolution over complex surfaces.

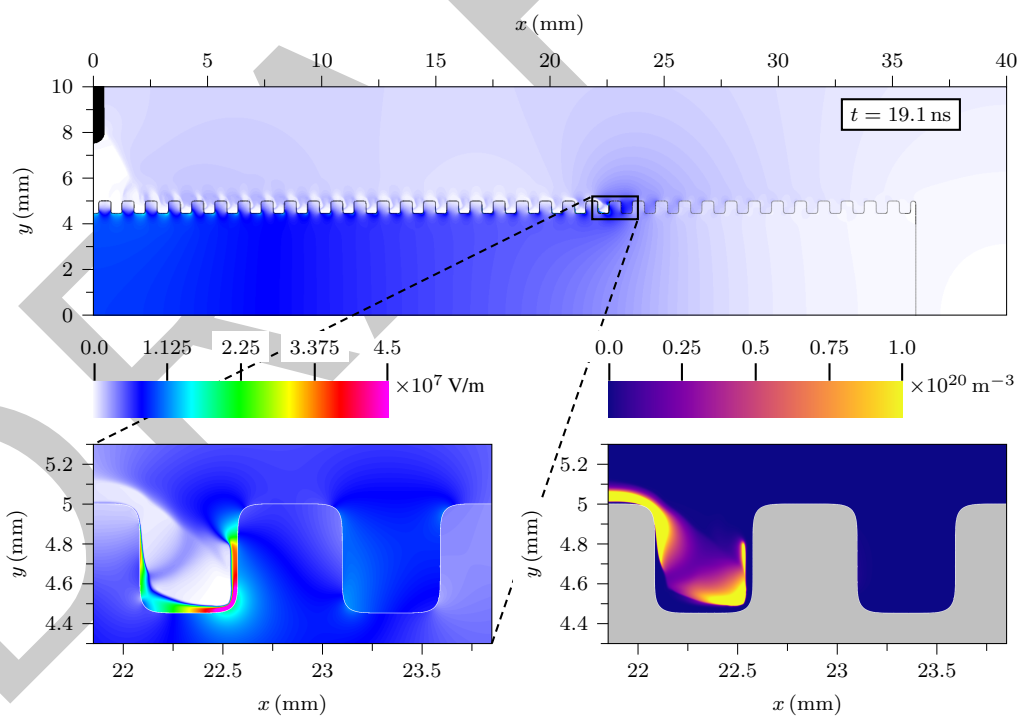


Figure 3: Simulation of laboratory-scale streamer corona

## 60 Acknowledgements

## 61 References

- 62 Marskar, R. (2019). An adaptive cartesian embedded boundary approach for fluid simulations  
63 of two- and three-dimensional low temperature plasma filaments in complex geometries.

<sup>64</sup> *Journal of Computational Physics*. <https://doi.org/10.1016/j.jcp.2019.03.036>

<sup>65</sup> Teunissen, J., & Ebert, E. (2017). Simulating streamer discharges in 3D with the parallel  
<sup>66</sup> adaptive afivo framework. *Journal of Physics D: Applied Physics*, 50, 474001. [https:](https://doi.org/10.1088/1361-6463/aa8faf)  
<sup>67</sup> [//doi.org/10.1088/1361-6463/aa8faf](https://doi.org/10.1088/1361-6463/aa8faf)

DRAFT