SANDIA REPORT
SAND2019-XXX
Unlimited Release
Printed March 2019

SGEMP Algorithm Development and Demonstration for ICEPIC

John J. Watrous, Ph.D.,
David B. Seidel, Ph. D.,
John W. Luginsland, Ph.D.
Confluent Sciences, LLC

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology and Engineering Solutions of Sandia, LLC.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods/

SAND2019-3165
Printed March 2019
Unlimited Release

SGEMP Algorithm Development and Demonstration for ICEPIC

John J. Watrous, Ph.D.,
David B. Seidel, Ph. D.,
John W. Luginsland, Ph.D.

Confluent Sciences, LLC
Albuquerque, New Mexico 87112

Abstract

Algorithms designed to facilitate calculation of gas-modified System-Generated Electromagnetic Pulse (SGEMP) phenomena have been developed for use in ICEPIC. Improvements and extensions of these and related models are considered for helping them be more widely applicable for SGEMP modeling. Confluent Sciences has developed a Buneman-Villasenor unit test tool for assessing particle merge algorithms and has applied that to several candidate algorithms. Statistical analysis of Maxwellian and near-Maxwellian velocity distribution functions has been used to understand limits and tradeoffs of particle merging. Collisional phenomena and plasma models that relieve the burden of a fully kinetic approach have also been developed and assessed.

ACKNOWLEDGMENTS

The authors would like to acknowledge the interest and support provided for this work by Drs. Keith Cartwright and Tim Pointon of Sandia National Laboratories through PO1919367.

TABLE OF CONTENTS

I. INTRODUCTION

The work presented in the following is an examination of several possible means of addressing difficulties encountered in modeling SGEMP phenomena, particularly gas-modified SGEMP phenomena.

Particle merging provides a means of controlling the number of macroparticles present within a calculation, but as it involves throwing away particle information, it must be done in a manner that will preserve as much of the original population as possible. A unit test tool has been developed to quickly assess candidate merge methods in terms of a variety of metrics on a variety of initial conditions. This has been used to assess several different candidate merge algorithm.

Different approaches to the interation of electrons with the background gas have been examined – these provide aspects of the interaction that a fully kinetic model provides without all of the expense, so can help a calculation model a challenging situation by avoiding some of the difficulties that a fully kinetic model can present.

II. BUNEMAN-VILLASENOR UNIT TEST TOOL

To provide an expeditious means for checking how well various merge algorithms perform with respect to conservation of charge, current, energy, etc., a simple unit test has been created which will be referred to as "BVcomp.". The unit test pushes a collection of particles in two dimensions though a single force-free timestep on a 3x3 cell array with all particles initially located within the central cell of the 3x3 array (a cell which will be referred to as the "home cell"). This is done twice: once for the original set of particles and a second time for the modified set where the modification is the trial merge algorithm applied to the original set of particles. The total charge, total kinetic energy, current on the grid as calculated by the Buneman-Villasenor algorithm (Villasenor & Buneman, 1992) as well as other metrics are collected for the original and the modified set of particles. This ignores motion in the third direction.

Figure 1 illustrates the Buneman-Villasenor unit check tool. The plot shows primary cell boundaries with dashed lines and dual cell boundaries with solid lines. Red dots indicate the initial locations of the original data set. Green dots indicate their final positions after a single timestep of force-free motion. The initial positions were chosen at random from positions within the central primary cell. The velocity components were also chosen at random from the range $-c/2 < V_x < c/2$, $-c/2 < V_y < c/2$. This is a simple initial condition which can be replaced easily by some other initial condition as desired by the user. The timestep is the Courant-condition timestep for the Yee algorithm:

$$\Delta t = 1 \Big/ \left(c \sqrt{ {1}/{(\Delta x)^2} + {1}/{(\Delta y)^2} } \right)$$

The arrows in the figure indicate the positions at which the Buneman-Villasenor (BV) algorithm will calculate components of the current density.

The observation by Buneman and Villasenor that the motion of a particle through a single time-step excites either 4, 7 or 10 current density components at specific sites on the grid can be simplified by breaking any path down into subpaths each of which is contained entirely within a single primary cell. Each subpath then excites only 4 current density components. The procedure used in BVcomp, which is functionally equivalent to that used in ICEPIC, is to calculate the start and end points of each subpath for a given path, and hand these end points to the Buneman-Villasenor current density calculator. Analysis of the geometry of an arbitrary path corresponding to a single timestep of motion has shown that 13 unique path-types exist, each of which has either 1, 2, or 3 subpaths with each subpath contained entirely within a single primary cell. These 13 cases are listed in Table 1 with the start and end points of each subpath and the indices of the primary cell, relative to the primary cell that owns the initial particle position, listed, as well. Figure 2 provides a definition of the various points that form the start and end points for the subpaths. Following this is a series of diagrams that help define the 13 types of paths. The 13 paths are each given a unique 3-digit code that helps identify them (referred to in the BVcomp code as "IMI" for Intersection Multi-Index). The left-most digit indicates whether the path final point is in a primary cell that is to the right or to the left of the primary cell that owns the path start point (0: left, 1: right, 9: neither). The middle digit indicates whether the path intersects first with a vertical or with a horizontal primary cell boundary element (0: vertical, 1: horizontal, 9: no such intersection). The right-most digit indicates whether the primary cell that owns the path end point is above or below the primary cell that owns the path start point (0: above 1: below, 9: neither). The value of 6 is assigned to the middle digit to denote that the path end-point resides in the same primary cell as the path start-point. This codification of the path types was useful for debugging and was retained as output.

Figure 3 shows a structure chart for the BVcomp tool (boxes are functions with lines indicated superior and subordinate relations). The various functions perform the following tasks:

- BVcomp: overall master function coordinating the entire set of calculations, from the actual Buneman-Villasenor algorithm to the various metrics and reduce/merger algorithms
- CreateSet: creates the original particle set. Particle positions are chosen randomly over the central primary cell. Particle velocity components are chosen from several different options: drifting Maxwellian, exponential ($f(E) = f_0 \exp(-E/E_0)$), random between $\pm V_0$, beam-like; calculates final particle positions assuming force-free motion through a single time-step.
- GetDualCellIndex: Finds the dual cell indices of each particle's initial and final position
- AnalyzePath: performs the decomposition of the particle path into subpaths each of which reside entirely within a single primary cell; also determines the path class and the initial and final points of the sub-paths
- Jcalc: coordinates the calculation of current density components using the Buneman-Villasenor algorithm
- CalculateJs: calculates the 4 components of current density for each sub-path for each particle using the Buneman-Villasenor algorithm
- ProdRate: collides the given particle set against a collisional cross section as a metric to compare the original and the post-merge particle sets; currently uses the nitrogen ionization cross-section
- AnalyzeSet: coordinates analysis of the particle set

- CalcVdfs: calculates velocity distribution functions for x- and y-components of velocity
- Weights: calculates the bilinear distribution of a single particle's shape over the four primary cells with which its shape intersects
- ClipSet: in reference to the particle charge-energy product, removes the top n^{th}-percentile of particles from the original particle set and injects them unchanged (with exception of particle charge) into the reduced particle set; this "clip-set" is not considered when processing the original set through the chosen particle merging algorithm
- ParticleMergeMaster*:* calls the user-requested merge algorihm
- InvertCVdfs: inverts the x- and y-velocity distribution function to determine particle velocity components
- ReduceCheck: calculates and outputs various diagnostic results

Metrics calculated for both the original and the modified (modified implying the result of applying the given particle merging algorithm to the original particle set) include total charge, charge allocated to the various grid points, total energy, and current density components. The algorithm produces 12 x-components of current density (for *i-1, i, i+1* and *j-1, j, j +1,* and *j+2*) and 12 y-components of current density (for *i-1, i, i+1, i+2* and *j-1, j,* and *j +1*). A "perfect" particle merge algorithm would conserve these 24 current density components identically while also identically conserving total charge, charge at the grid points and total energy. It is probably true that an algorithm that identically conserves the 24 current density components automatically conserves the charge and energy identically, as well. Comparing the 24 components of current density can be time-consuming, so a single number based on the current density was created. This calculates the charge transported out of the home cell due to the motion of the particles through the single timestep that the BVcomp tool puts them through and is calculated by use of the current density components calculated by the Buneman-Villasenor algorithm. The current density components more naturally refer to the transport of current into and out of dual cells, but since in the BVcomp tool, all of the particles are initially resident only within the central primary, or home cell, calculating the charge transported out of the four dual cells that intersect with the home cell and summing the result gives the total charge transported out of the home cell. Thus, letting ΔQ_0 represent the change in charge within the home cell due to motion of particles in a single timestep,

$$\Delta Q_0 = \Delta Q_1 + \Delta Q_2 + \Delta Q_3 + \Delta Q_4$$

Where subscripts 1-4 refer to the four dual cells that overlap the home cell. Using the current density components, this can be expressed as

$$\begin{aligned}
\Delta Q_0 = \quad & \Delta y \left\{ Jx_{i+2,j+1} - Jx_{i+1,j+1} \right\} + \Delta x \left\{ Jy_{i+1,j+2} - Jy_{i+1,j+1} \right\} + \\
+ \quad & \Delta y \left\{ Jx_{i+1,j+1} - Jx_{i,j+1} \right\} + \Delta x \left\{ Jy_{i,j+2} - Jy_{i,j+1} \right\} + \\
+ \quad & \Delta y \left\{ Jx_{i+1,j} - Jx_{i,j} \right\} + \Delta x \left\{ Jy_{i,j+1} - Jy_{i,j} \right\} + \\
+ \quad & \Delta y \left\{ Jx_{i+2,j} - Jx_{i+1,j} \right\} + \Delta x \left\{ Jy_{i+1,j+1} - Jy_{i+1,j} \right\}
\end{aligned}$$

This gives a single number that describes the result of the Buneman-Villasenor algorithm. A particle merge algorithm that produces precisely the same ΔQ_0 result as was produced by the original particle set does not necessarily have the same 24 current density components, but it is nonetheless a useful metric for quickly assessing a merge algorithm.

Another metric has been developed to provide a means of quickly assessing how well the velocity distribution function has been recreated by the merge algorithm. This metric was suggested by the likelihood that the calculation in question involves the interaction of electrons with background species. The reaction rates that govern the various interactions, ionization, for example, are very sensitive to the electron velocity distribution function, so to minimize the impact on interaction rates, the merge algorithm should reproduce the original distribution function with as little error as possible. Calculating the interaction rate for a typical interaction using the original particle set, then comparing that to the interaction rate produced by the post-merger particle set gives a single number for assessing the impact that the merge algorithm has on the velocity distribution function. As with the ΔQ_0 metric, the interaction rate can turn out to be the same even if the distribution functions are not the same. The current implementation uses the electron-N_2 ionization cross-section to calculate the nitrogen ionization rate per background nitrogen atom.

Additional metrics include the charge allocated to the nodes, particle momentum allocated to the nodes, and total kinetic energy of the particle set.

A repetition capability has been implemented that allows the user to readily generate a set of results by creating N initial data sets that differ, say, by the seed used for the random number generator that creates them. The N sets of diagnostic results can then be subjected to statistical analysis, generating, for example, the mean, the median, and the standard deviation of the results.

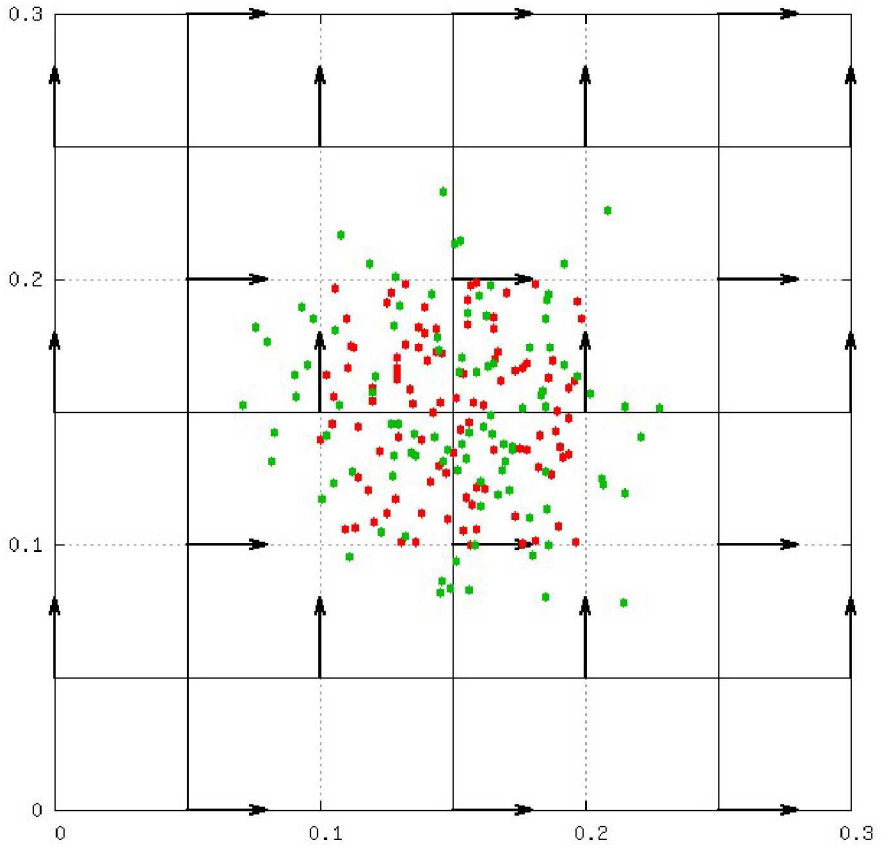


Figure 1: The Buneman-Villasenor unit test

Table 1: Path decomposition into sub-paths

Path Code	Sub-Path	Initial Point	Final Point	Primary Cell
000	1	(x_i, y_i)	(X(i), yINT)	(i, j)
(class 0)	2	(X(i), yINT)	(xINT, Y(j+1))	(i-1, j)
	3	(xINT, Y(j+1))	(x_f, y_f)	(i-1, j+1)
001	1	(x_i, y_i)	(X(i), yINT)	(i, j)
(class 1)	2	(X(i), yINT)	(xINT, Y(j))	(i-1, j+1)
	3	(xINT, Y(j))	(x_f, y_f)	(i-1, j-1)
010	1	(x_i, y_i)	(xINT, Y(j+1))	(i, j)
(class 2)	2	(xINT, Y(j+1))	(X(i), yINT)	(i, j+1)
	3	(X(i), yINT)	(x_f, y_f)	(i-1, j+1)
011	1	(x_i, y_i)	(xINT, Y(j))	(i, j)
(class 3)	2	(xINT, Y(j))	(X(i), yINT)	(i, j-1)
	3	(X(i), yINT)	(x_f, y_f)	(i-1, j-1)
100	1	(x_i, y_i)	(X(i+1), yINT)	(i, j)
(class 4)	2	(X(i+1), yINT)	(xINT, Y(j+1))	(i+1, j)
	3	(xINT, Y(j+1))	(x_f, y_f)	(i+1, j+1)
101	1	(x_i, y_i)	(X(i+1), yINT)	(i, j)

(class 5)	2	(X(i+1), yINT)	(xINT, Y(j))	(i+1, j)
	3	(xINT, Y(j))	(x_f, y_f)	(i+1, j-1)
110	1	(x_i, y_i)	(xINT, Y(j+1))	(i, j)
(class 6)	2	(xINT, Y(j+1))	(X(i+1), yINT)	(i, j+1)
	3	(X(i+1), yINT)	(x_f, y_f)	(i+1, j+1)
111	1	(x_i, y_i)	(xINT, Y(j))	(i, j)
(class 7)	2	(xINT, Y(j))	(X(i+1), yINT)	(i, j-1)
	3	(X(i+1), yINT)	(x_f, y_f)	(i+1, j-1)

009	1	(x_i, y_i)	(X(i), yINT)	(i, j)
(class 8)	2	(X(i), yINT)	(x_f, y_f)	(i-1, j)
109	1	(x_i, y_i)	(X(i+1), yINT)	(i, j)
(class 9)	2	(X(i+1), yINT)	(x_f, y_f)	(i+1, j)
910	1	(x_i, y_i)	(xINT, Y(j+1))	(i, j)
(class 10)	2	(xINT, Y(j+1))	(x_f, y_f)	(i, j+1)
911	1	(x_i, y_i)	(xINT, Y(j))	(i, j)
(class 11)		(xINT, Y(j))	(x_f, y_f)	(i, j-1)
969 (class 12)	1	(x_i, y_i)	(x_f, y_f)	(i, j)

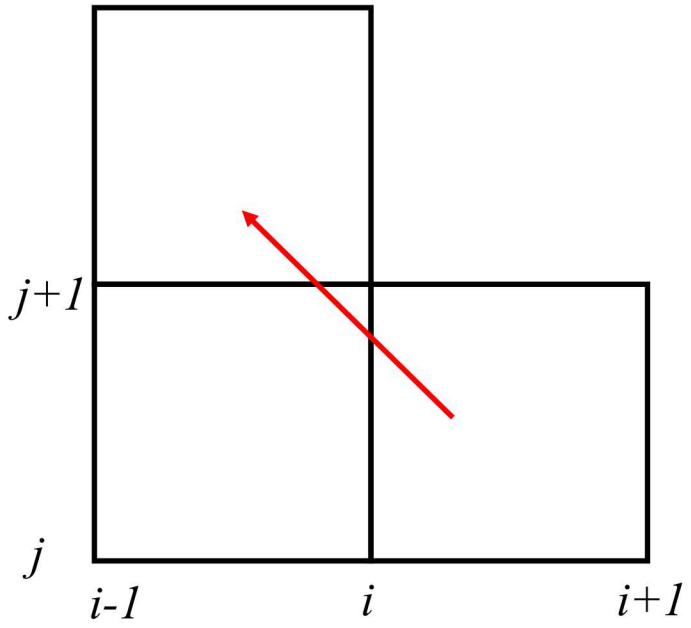


Figure 2: Showing a path that crosses passes through three primary cells

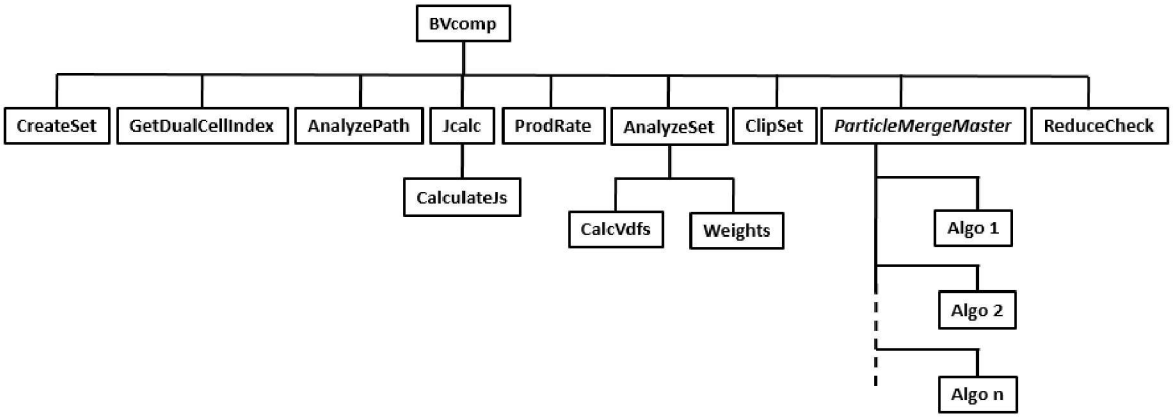


Figure 3: Structure chart for BVcomp Buneman-Villasenor unit test code

III. CANDIDATE MERGE ALGORITHMS

Several different merge algorithms have been developed and tested. Not all are necessarily considered as serious candidates for us. Some were useful in exploring specific aspects of different approaches to merging. In the following, we present each algorithm that has been incorporated into BVcomp. Each algorithm has been subjected to a uniform set of tests that help to show its strengths and weakness. The algorithms are each run on four different initial velocity loading described in Section III.A.2 below. The test results have been useful in suggesting possible improvements, and these have spawned some of the candidate algorithms that have been tested.

A. ICEPIC REDUCE Algorithm

1. Introduction

The ICEPIC REDUCE algorithm was developed to deal with rapidly growing particle populations in calculations dealing with volumes of plasma excited by relatively low intensity RF excitation. The nature of the calculation drove some of the choices made in constructing the algorithm; not all of these choices are necessarily the most appropriate for SGEMP applications. These will be pointed out and discussed within the context of possible improvements to the algorithm.

2. Description

In its current implementation in ICEPIC, the REDUCE algorithm is triggered on a global basis, yet acts on a cell-by-cell basis. The user defines a trigger level as the total number of particles of the species in question. When the total number of particles of this species reaches the trigger level, the algorithm is activated.

The algorithm uses trilinear weighting to allocate to each cell the weight that each particle in the original set contributes to that cell. This total weight is used to determine the weight of the replacement particles for that cell. The algorithm also calculates the velocity distribution function in each direction for each cell that contains any of the original particles.

For each cell that contained any of the original particles, a replacement set for that cell is created with half as many particles as the original set. The new particles all are assigned the same weight calculated as the total weight of the original particles allocated to this cell divided by the number of replacement particles. The positions of the replacement particles are determined randomly over the volume of the cell. The velocities of the replacement particles are determined by inverting the velocity distribution functions calculated for the original particles.

A weakness of the algorithm is how it handles the particle weight or charge. The total weight to be divided among the new particles in a given cell is determined using the first-order trilinear interpolation of the original particles to the given cell. But because the replacement particles are distributed randomly over the volume of the cell, if the total weight allocated to the given cell by the replacement particles were to be calculated, it would not necessarily be equal to the total weight allocated to the cell by the original set of particles. The difference in charge is not lost. It can be found in the neighboring cells, but this process seems likely to have a diffusive effect that could cause harm to existing density gradients.

The greatest strength of the algorithm is its replication of the pre-existing velocity distribution function – it reproduces what was there by construction, and for cells that are richly loaded with original particles, it does a very good job of it.

3. Examples

The algorithm was used to reduce an initial loading of 10^4 particles to a replacement set containing 5000 particles. This was done for 4 different initial loadings. These 4 different initial loadings all distribute the particles randomly within the home cell, but use different velocity distributions. The first (IC1) was a random velocity distribution defined by

$$\gamma = 1 + \xi E_0 / mc^2 ; V_0 = c\sqrt{1 - 1/\gamma^2}$$

* MERGEFORMAT (1.1)

$$v_{x,n} = V_0 \cos(2\pi \times \zeta)$$

$$v_{y,n} = V_0 \sin(2\pi \times \zeta)$$

Where, thoughout, ζ and ξ are random numbers selected from the range $0 < \zeta, \xi < 1$ and E_0 is a user-defined energy. The second is a slight randomized drifting population defined by (IC2)

$$v_{x,n} = V_0 + \xi \Delta V$$

* MERGEFORMAT (1.2)

$$v_{y,n} = \zeta \Delta V$$

$$\Delta V = V_0 / 10^3$$

The third is an exponential electron energy distribution ($f(E) = f_0 \exp(-E/E_0)$ for $E > E_{min}$) combined with a cos(θ) angular distribution with particle velocities determined according to the inversion of $f(E)$ for velocity components (IC3):

* MERGEFORMAT (1.3)

$$E = E_{\min} - E_0 \ln(1-\zeta); \quad \gamma = 1 + E/mc^2$$
$$V_0 = c\sqrt{1-1/\gamma^2}; \quad \theta = \sin^{-1}(\xi)$$
$$v_{x,n} = V_0 \cos(\theta)$$
$$v_{y,n} = V_0 \sin(\theta)$$

The fourth is a drifting Maxwellian, with velocities determined by numerical inversion of the velocity distribution functions defined by (IC4)

* MERGEFORMAT (1.4)

$$f(E) = f_0 \exp(-E/kT)$$
$$E = mc^2(\gamma - 1)$$
$$\gamma = 1/\sqrt{1-\beta^2}$$
$$\beta^2 = c^2[(v_{x,n} + V_D)^2 + v_{y,n}^2]$$

The REDUCE algorithm was run 1000 times with the seed for the random number generator changing with with each iteration. The various metrics discussed in connection with the BVcomp code description above were collected and run through a simple statistical analysis tool that calculates the frequency distribution and the cumulative probability distribution functions as well as the mean and standard deviation of the distributions. The median of the distribution was then determined by finding the value of the variable for which the normalized median equals ½. The statistical analysis was performed on the relative error between the metric as determined for the original particle distribution and metric as determined for the replacement set. The relative error is defined here as

* MERGEFORMAT (1.5)

$$RE(x_{replace}) = \frac{\left|x_{orig} - x_{replace}\right|}{\frac{1}{2}\left(x_{orig} + x_{replace}\right)}$$

Execution times were monitored for all the algorithms, but it should be noted that no attempt to optimize execution time has been undertaken. In this regard, it is clear by simple inspection of the manner in which the algorithms are constructed that a simple reworking could result in significantly quicker execution. Table 2 shows the execution times of REDUCE on the four different initial velocity loadings. The median values for the relative error distribution in the various metrics from the 1000 calculations using the REDUCE algorithm are shown in Table 3. The "Jx11" metric simply pulls out a selected current density component for before and after comparison. This is the x-component of J at grid location (1,1).

BVcomp calculates the x- and y-velocity distribution functions for both the original and the replacement sets. The Vx-distribution functions are compared for the various initial velocity loadings in Figure 4 through Figure 7. The distribution function for the replacement set is shown reflected through the horizontal axis to facilite comparison. The plots were constructed by placing a red or green dot at the value of the distribution function for each of the 1000 different

distributions generated in a single run. For the original sets of particles, the breadth in the vertical dimension of the distribution of dots created by superimposing 1000 slightly different velocity distribution functions is a result of the sampling of the prescribed velocity or energy distribution. This breadth increases in all cases for the replacement set. The increase in breadth is consistent with the increase in statistical variability due to decreasing the number of particles used to build the distribution function from 10000 to 5000: the breadth increases approximately by a factor of 1.4 to 1.5, which is what one would expect on the basis of statistical noise, where the breadth of a distribution varies with $1/\sqrt{n}$, where n is the number of samples, resulting in a $\sqrt{2}$ increase in the breadth of the replacement set velocity distribution function plots.

Table 2: Execution time for ICEPIC REDUCE for 1000 repetitions starting with a set containing10^4 particles

Initial Loading Method	Time Required (s)
Random V	10.30
Translate w/slight randomization	8.82
Exponential	9.94
Drifting Maxwellian	10.71

Table 3: Mean value of the relative error in the various BVcomp metrics produced by the ICEPIC REDUCE algorithm working on the various initial conditions

Initial Loading	ΔQ_0	Ionization Rate	Kinetic Energy	Nodal Q	$J_x(1,1)$
Random-V	0.18356	0.02511	0.007906	0.01227	4.252*
Drift	0.07479	0.003648	6.327e-06	0.01204	0.008189
Exponential	0.07837	0.016010	0.007885	0.01227	0.01135
Drift Max	0.04153	0.006546	0.007117	0.01231	0.01020

*the frequency distribution of relative error was severely distorted by the presence of a few dozen outliers

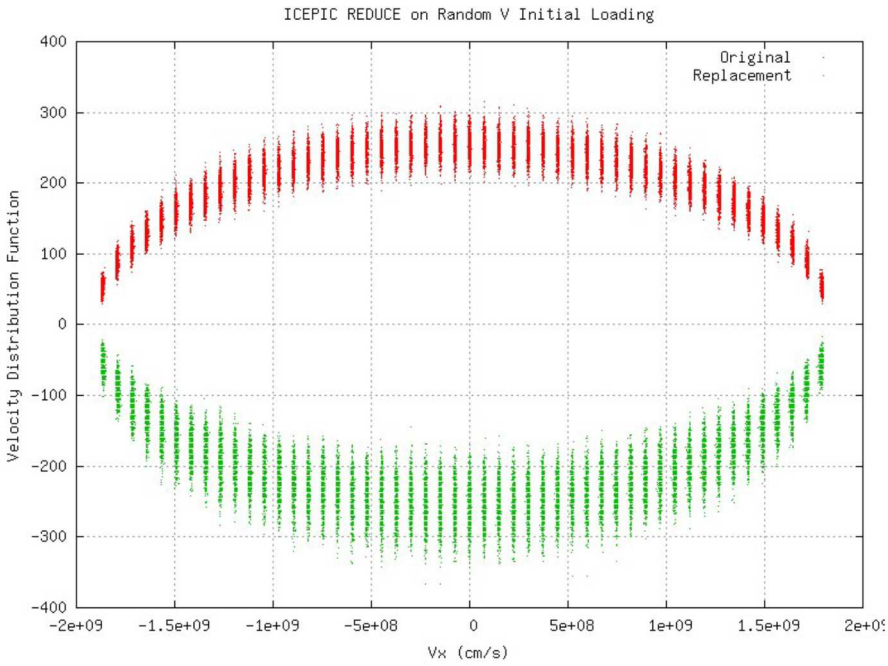


Figure 4: Original (red) and Replacement (green) V_x distribution functions for the random-velocity initial loading (Replacement reflected through the horizontal axis to facilitate comparison in this and the following velocity distribution function plots).

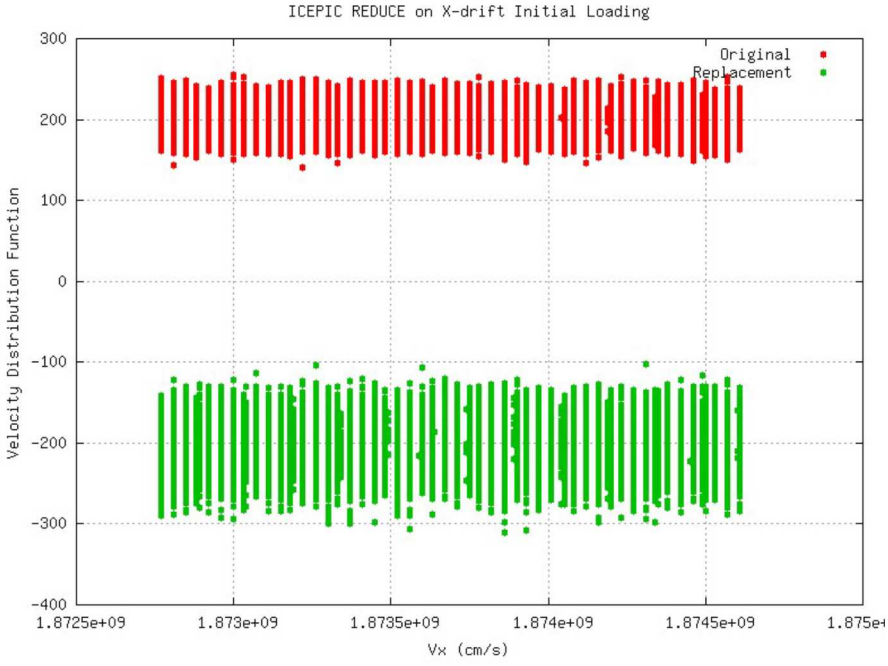


Figure 5: Original and replacement V_x distribution functions for the lightly-randomized drifting particle loading

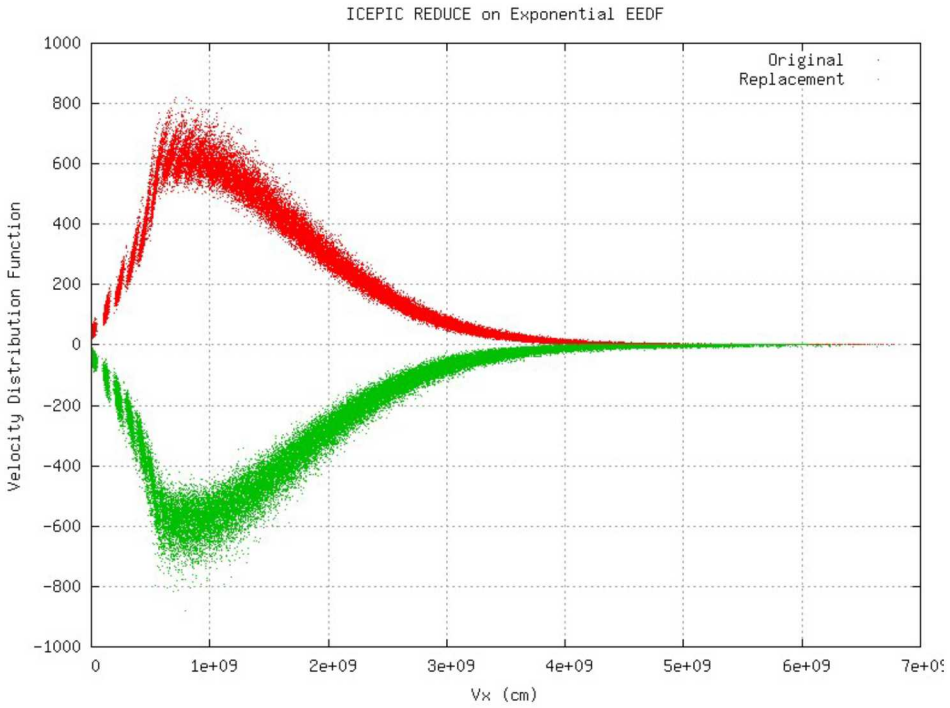


Figure 6: Original and replacement V_x distribution functions for the exponential energy distribution with cosθ angular distribution

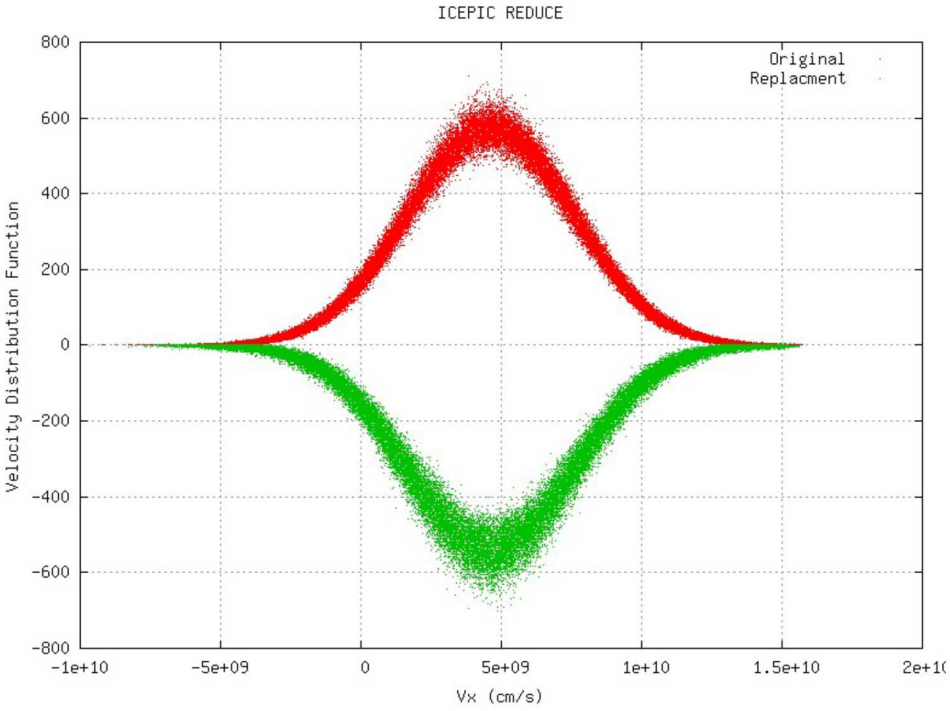


Figure 7: Original and replacement V_x distribution functions for the drfting Maxwellian initial velocity loading

Figure 8 through Figure 12 show the relative error in various metric for the four initial velocity loading. All of the algorithms happened to perform remarkably well in terms of kinetic energy on initial loading IC2 - so well that the relative error frequenccy distribution cannot be meaningfully plotted on the same linear plot used for the other results. At the other extreme, all algorithms performed remarkably poorly on the random-velocity (IC1) initial velocity loading in terms of the $J_x(1,1)$ metric. In these results, several dozen values of $J_x(1,1)$ that were 2 to 3 orders of magnitude greater that what, in their absence, would have been the mean value were produced. These tended to distort the relative error frequency distribution to an extreme degree. These results are also not plotted.

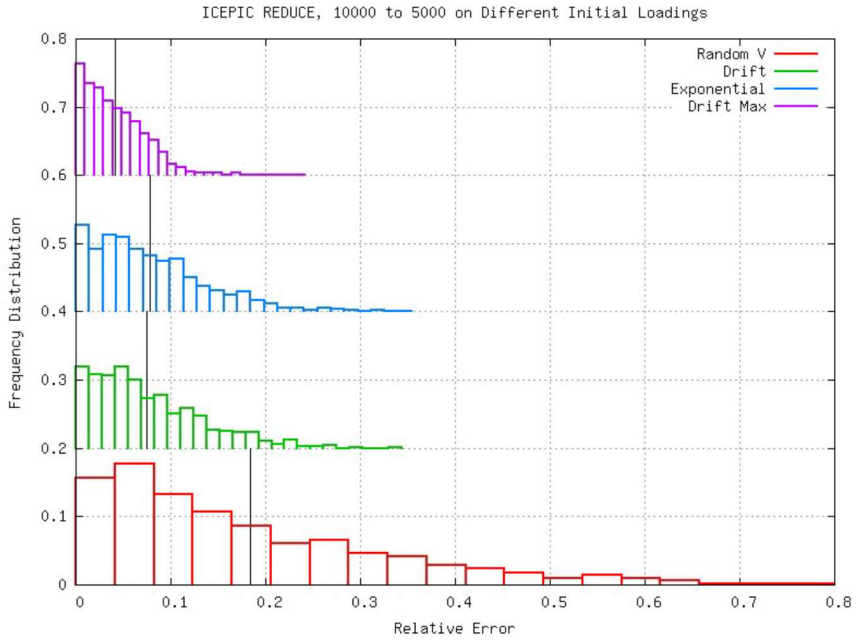


Figure 8: Relative error in the ΔQ_0 metric by the ICEPIC REDUCE algorithm on the four initial velocity loadings. Thin black vertical line indicates the median value of the distribution.

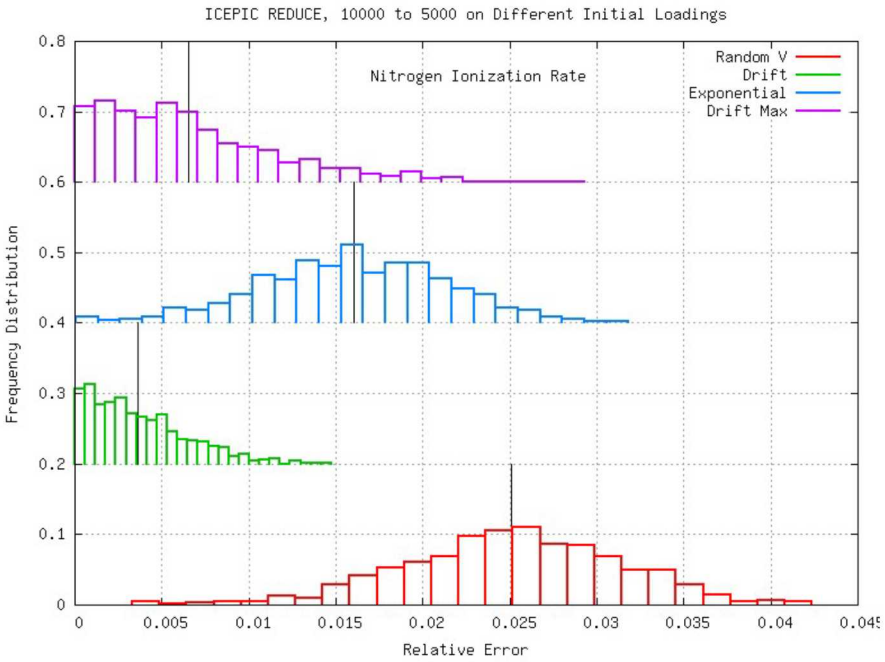


Figure 9: Relative error in the nitrogen ionization metric by the ICEPIC REDUCE algorithm on the four initial velocity loadings

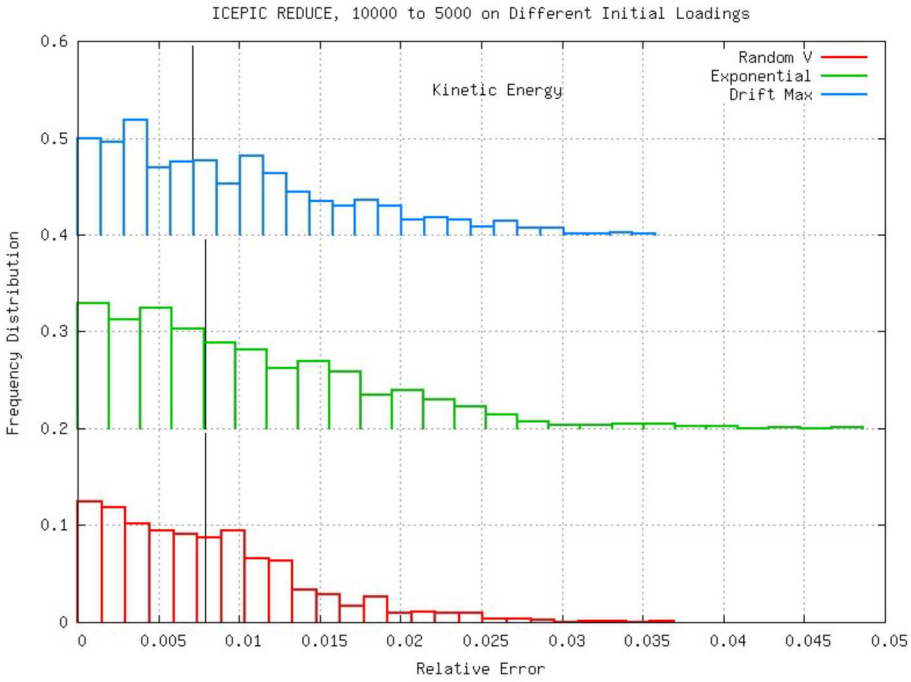


Figure 10: : Relative error in the kinetic energy metric by the ICEPIC REDUCE algorithm on three of the four initial velocity loadings; the results of the drift initial loading were not included here as they would not be visible given that their mean is 3 orders of magnitude smaller than that of the distributions shown here.

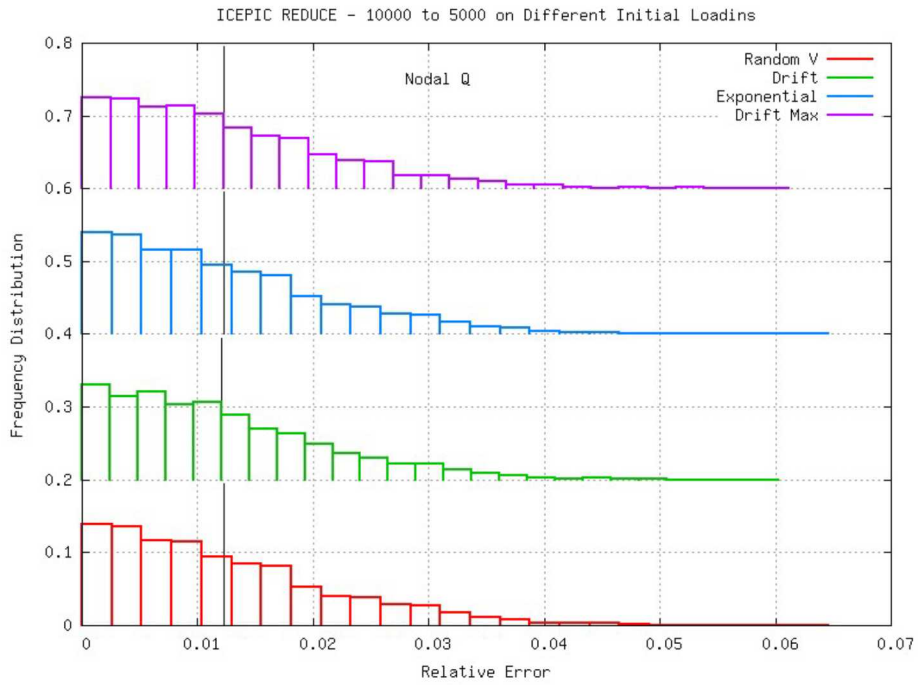


Figure 11: Relative error in the nodal Q metric by the ICEPIC REDUCE algorithm on the four initial velocity loadings

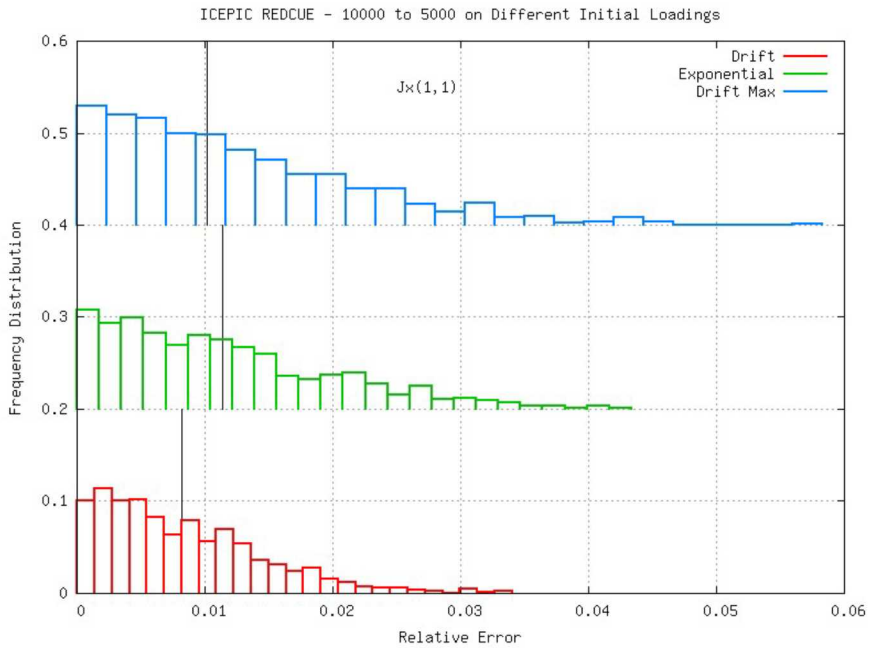


Figure 12: : Relative error in the $J_x(1,1)$ metric by the ICEPIC REDUCE algorithm on three of the four initial velocity loadings; the results for the random-V distribution are not included as the distribution was severely distorted by a non-trivial subset of outliers.

4. Summary of the REDUCE method

Other than the unavoidable statistical increase in the breadth of the collected velocity distribution functions, the REDUCE algorithm does a good job of reproducing the original velocity distribution function. This is to be expected, since rebuilding the original velocity distribution function is part of the definition of this algorithm. Hand-in-hand with its capability to rebuild the original velocity distribution function is its low degree of relative error in reproducing the ionization rate produced by the original set of particles, along with its low degree of relative error in reproducing the kinetic energy. These suggest that a good regime for applicability of this method is in calculations where the response of the background gas to ionizing and collision reactions is of high importance. On the less positive side, the relative error in nodal Q has a mean value of just over 1% with a consistently large standard deviation for each of the four initial velocity loadings tested. This suggests a systematic error that could have consequences in calculations where frequent applications of the merge algorithm could produce a cumulative deviation in the computed electromagnetic behavior of the system being modeled.

Two possible improvements to this method have been considered. The first is to change the manner in which the algorithm is invoked. In its current implementation in ICEPIC, the trigger is the total number of particles of the species in question, which is a global quantity. All cells containing particles of that species are then subjected to the REDUCE algorithm. While there would be no direct impact on the metrics BVcomp provides, changing this approach to a purely local approach, where the algorithm trigger is evaluated on a cell-by-cell basis,would help focus the use of the algorithm to just the cells where it is needed. The second possible improvement is to use the conservation of nodal-Q, as used in the Sandia merge algorithm (discussed in detail below), for the determination of the weights to assign to each of the new particles in the replacement set. This would not force any other changes in the algorithm, so would be a simple and straightforward way in which to eliminate the systematic error in nodal Q observed for this method. This would also help the algorithm to avoid the potential erosion of density gradients noted above in relation to the weight assignment rule for the existing algorithm.

B. The Reduce-By-Path-Class Algorithm

1. Introduction

While implementing the Buneman-Villasenor algorithm in BVcomp, the reduction of the total particle path into sub-paths, each of which is contained entirely within a single primary cell, pointed out an interesting relationship between the set of particles within the home cell and the manner in which the Buneman-Villasenor algorithm operates on that set. In two dimensions, a particle has only 9 possible outcomes in terms of the cell that contains the terminal point of its path: it can remain in the cell it started in, it can go to any of the four neighboring cells to the right, left, above, or below, or it can go to any of the four diagonally-neighboring cells. In determining how these various outcomes play out in terms of the decomposition of the total path into sub-paths, 13 distinct decompostions of the total path exist that describe all of the possible path/sub-path decompositions were identified. These are enumerated in Table 1. These path-classes contain the correlation between particle position and velocity that determine how the Buneman-Villasenor algorithm works on a given particle. This prompted the idea that were the merge algorithm to respect the membership of particles within these 13 distinct path classes, perhaps a better outcome in terms of the current density as calculated by the Buneman-Villasenor algorithm would result.

2. Description

The first thing that this algorithm, which will be referred to as the "reduce by path class" algorithm (RBPC), must do is to identify to which path class each particle in the original set of particles belongs. As this task is already performed in the execution of the Buneman-Villasenor algorithm, it was necessary only to add an integer array to the particle data structure that would carry along each particle's path-class identity. Second, the algorithm must assign the position and velocity of each particle in the replacement set such that the new particles belong to the same path class as the particles that they are replacing. For this, the method described by Cambier (REF) appeared to have promise, so it was adopted for use in the RBPC method.

Cambier's method calculates replacement set particle velocities by calculating center-of-momentum velocities and "thermal velocities". To that, our implementaton added an assignment of positions that did not have a clear counterpart in Cambier's approach. The description of the details of the position and velocity assignments follow.

The assignment of replacement set velocities and positions starts with the division of the set of original particles contained within a given cell into m distinct subsets, with each subset containing $N_{tot,m}$ particles. In Cambier's work, the subsets are determined from a velocity binning method (octree binning) that causes the subsets to be grouped closely together in velocity, but ignores their initial positions. Our interpretation of Cambier's method is that the octree velocity binning incorporates all particles within the entire calculation, rather than within, say, just a single cell, a choice which clearly breaks the correlation between position and velocity. In the BVcomp implementation, the m subsets are the path classes determined for the particles residing only within the home cell. Each subset is then divided into groups of four particles, each of which are to be replaced by two new particles. Each of the two new particles are assigned weight and velocity using the center-of-momentum and thermal speed of the 4 particles they are replacing:

* MERGEFORMAT (1.6)

$$w_{n,new} = \tfrac{1}{2}\sum_{i=1}^{4} w_{4n+i}; \quad n = 0,1,2,...,\mathrm{int}\left(\frac{N_{tot,m}}{4}\right) - 1$$

$$\vec{v}_{n,new} = \left\langle \vec{v} \right\rangle_n \pm v_{th,n}\left(\hat{x}\cos\theta + \hat{y}\sin\theta\right); \quad \theta = 2\pi\chi_n$$

where χ_n is a random number over the range 0 to 1, and the center-of-momentum velocity and thermal speed are calculated using

* MERGEFORMAT (1.7)

$$\left\langle \vec{v} \right\rangle_n = \frac{1}{w_{n,new}} \sum_{i=1}^{4} w_{4n+i} \vec{v}_{4n+1}$$

$$v_{th,n}^2 = \frac{1}{w_{n,new}} \sum_{i=1}^{4} w_{4i+1} (\vec{v}_{4i+1} - \left\langle \vec{v} \right\rangle_n)^2; \quad n = 0,1,2,...,\mathrm{int}(\frac{N_{tot,m}}{4})$$

This assignment of weights and velocities is intended to help conserve weight (*aka* charge), momentum, and energy.

For position assignment, Cambier goes to some lengths to make choices guided by considerations appropriate for use in an electrostatic code. As our focus is on electromagnetic codes, we looked for another choice that would help to preserve the path-class identity of the particles being replaced. The starting point was to use the positions of the four particles to be replaced as a guide for the positions of the two replacement particles. At the time of devising this algorithm, observations had been made that using averages to determine replacement set quantities tends naturally to pull the replacment set quantities toward the average values. We sought a choice that would tend to work in the opposite direction, in other words, to maintain the full spatial footprint of the original four particles being replaced. The obvious choice was to use the positions of the two particles from the set of four old particles that were the closest to and farthest from the center of the home cell. This helps to counter the erosion of the extremes that averaging processes otherwise inevitably produce. When the subset of old particles contains just 3 particles, or just 3 particles remain in the subset after having processed the subset into groups of 4, the positions of the two new particles are both set equal to the center-of-mass position of the 3 old particles, but the velocities for the 2 replacement particles are assigned just as outlined above for the sets of 4 original particles. When just one or two original particles remain in the subset, they are brought straight over to the replacement set without changing the positions or velocities, but letting the weight be reassigned to ensure charge conservation.

The weights of the new particles are determined by summing the weights of the original particles (using a simple summation of the particle weight rather than allocation by bilinear interpolation to the home cell), and dividing that sum by the total number of replacement particles.

3. Examples

As was done with the REDUCE method, the RBPC method was exercised on each of the four initial velocity loadings described above. Each initial velocity loading was subjected to 1000 repetitions of the merge algorithm, each time starting with a new original particle set containing 10^4 particles. The time required for 1000 repetitions for each initial velocity loading is listed in Table 4. The mean values of the various diagnostic metrics over this set of 1000 repetitions is shown for each initial velocity loading in Table 5.

The initial and replacement set v_x velocity distribution functions are collected for all 1000 repetitions in Figure 13 through Figure 16. The random-V and drift initial loadings (IC1 and IC2 - Figure 13 and Figure 14) show an extension to the range of velocities in the replacement set over that present in the original set. This is not clearly seen in the rebuilding of velocity distribution functions for the exponential EEDF and drifting Maxwellian initial velocity loading (IC3 and IC4 - Figure 15 and Figure 16), but is likely associated with the somewhat diffuse nature of the rebuilt IC3 velocity distribution function relative to that of the original set. The rebuilt drifting Maxwellian velocity distribution function is quite similar to that of the original, modulo vertical width increase due to increased statistical noise. Figure 17 through Figure 21 show the frequency distribution of relative error between the various metrics as measured for the replacement set in comparison to those measured for the original set.

Table 4: Execution time for RBPC for 1000 repetitions starting with a set containing10^4 particles

Initial Loading Method	Time Required (s)
Random V	10.32
Drift	8.67
Exponential	10.4
Drifting Maxwellian	10.67

Table 5: Mean value of the relative error in the various BVcomp metrics produced by the RBPC algorithm working on the various initial conditions

Initial Loading	ΔQ_0	Ionization Rate	Kinetic Energy	Nodal Q	$J_x(1,1)$
Random-V	0.8845	0.01195	0.086608	0.006885	5.16478*
Drift	0.04968	0.01249	6.665e-08	0.01765	0.004837
Exponential	0.1346	0.01088	0.03143	0.01547	0.007283
Drift Max	0.3060	0.02831	0.04881	0.02668	0.01693

*the frequency distribution of relative error was severely distorted by the presence of a few dozen outliers

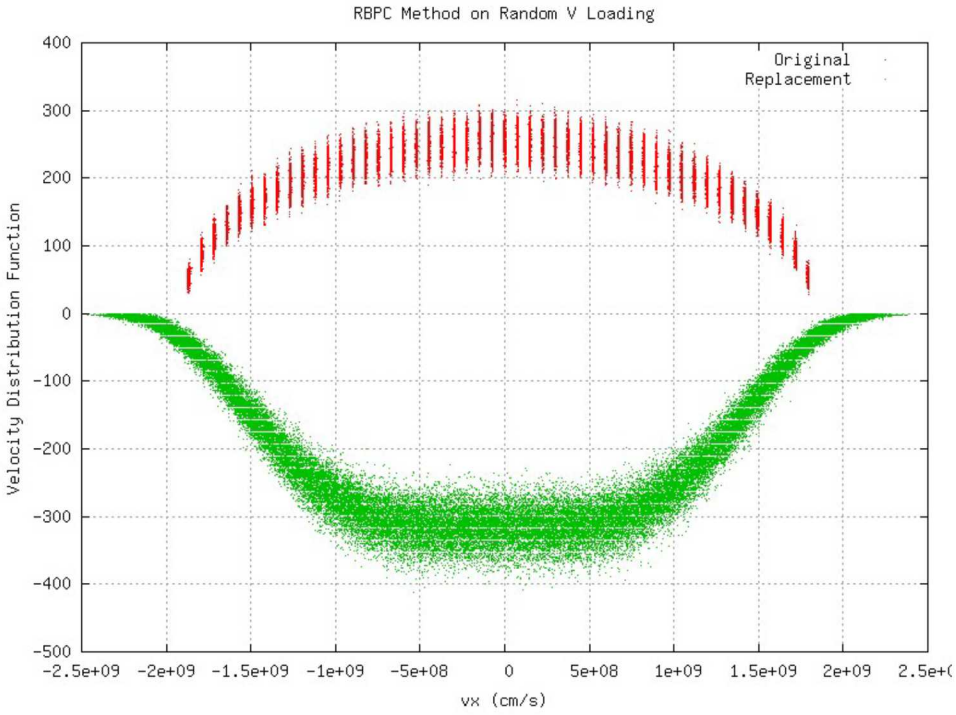


Figure 13: Comparison of original and replacment x-component velocity distribution function for the random-V initial loading by the RBPC method

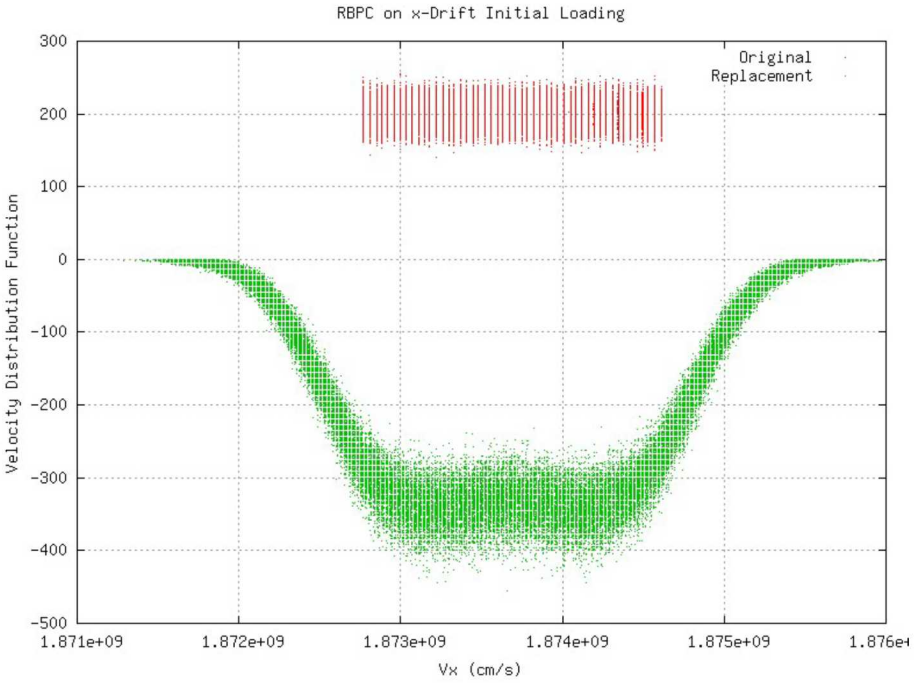


Figure 14: Comparison of original and replacment x-component velocity distribution function for the drifting initial loading by the RBPC method

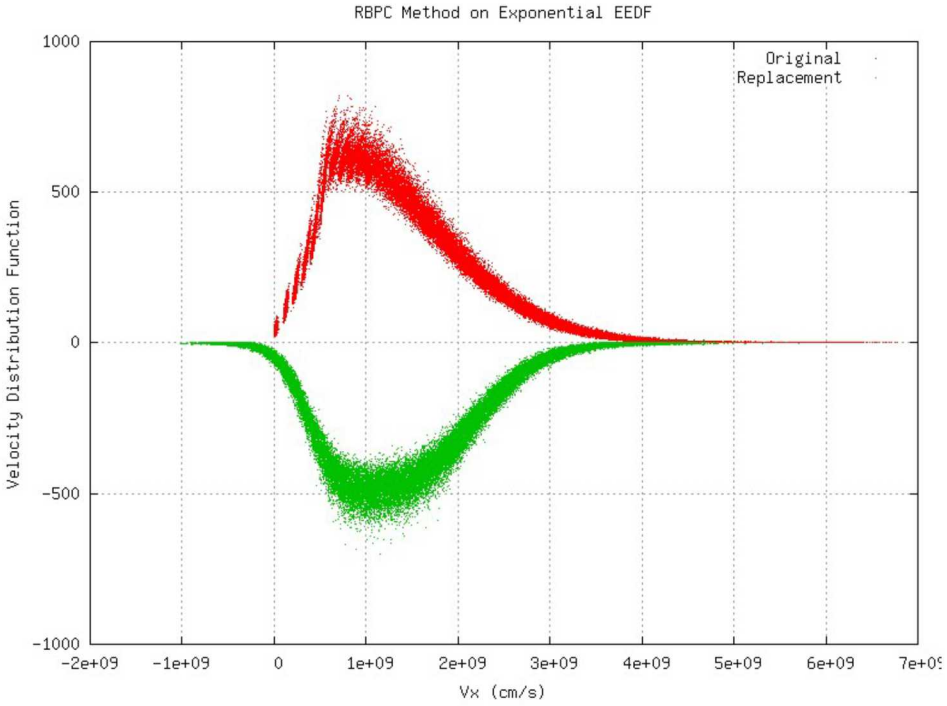


Figure 15: Comparison of original and replacment x-component velocity distribution function for the exponential EEDF initial loading by the RBPC method

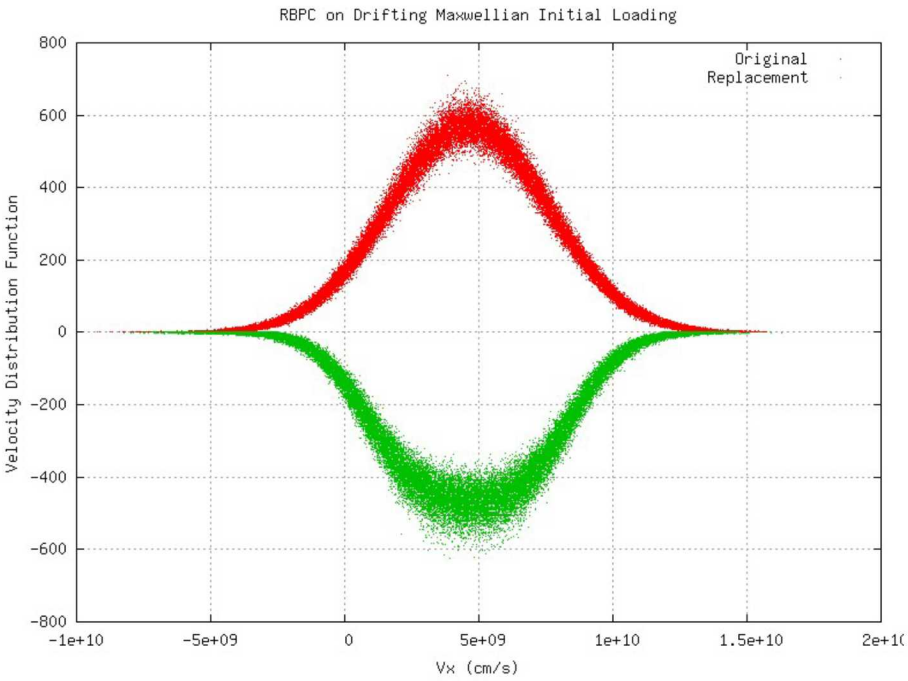


Figure 16: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellian by the RBPC method

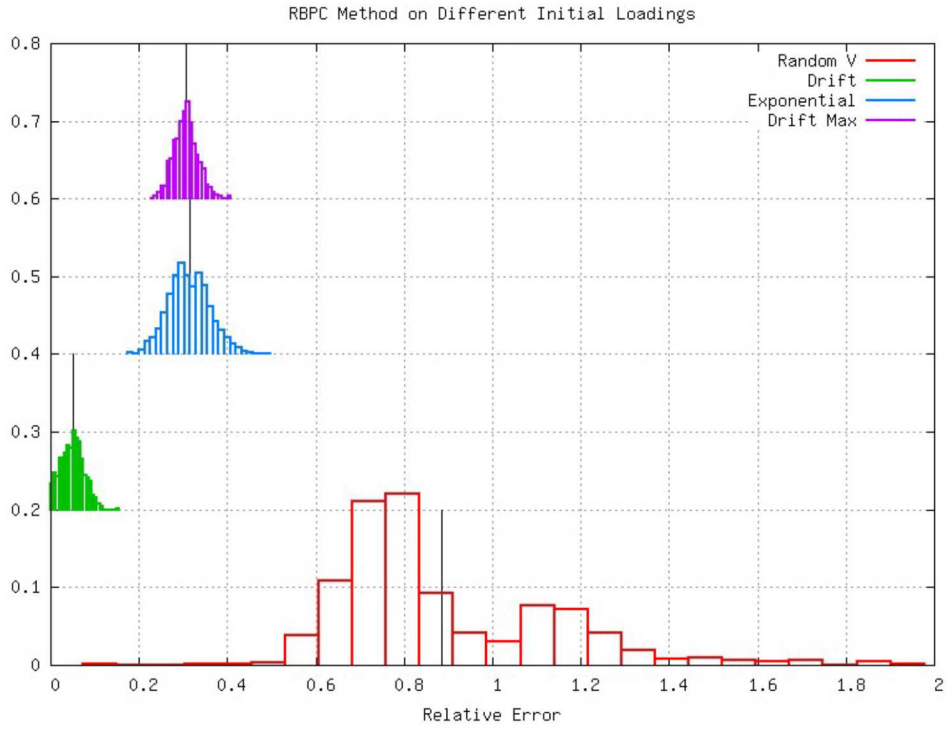


Figure 17: Relative error in the ΔQ_0 metric for the various initial loadings produced by the RBPC method

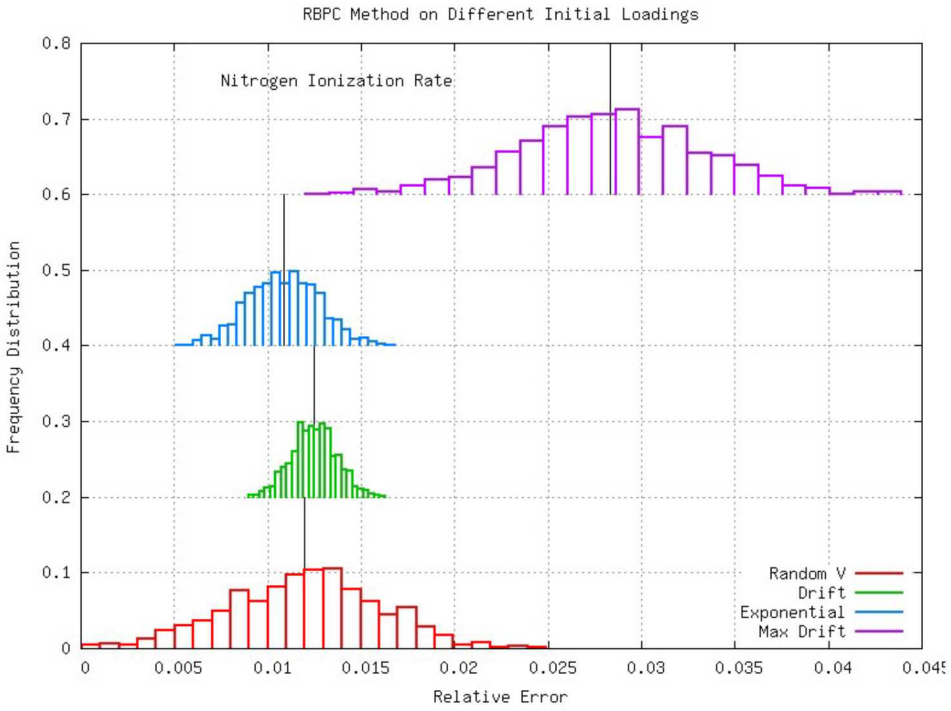


Figure 18: Relative error in the nitrogen ionization metric for the various initial loadings produced by the RBPC method

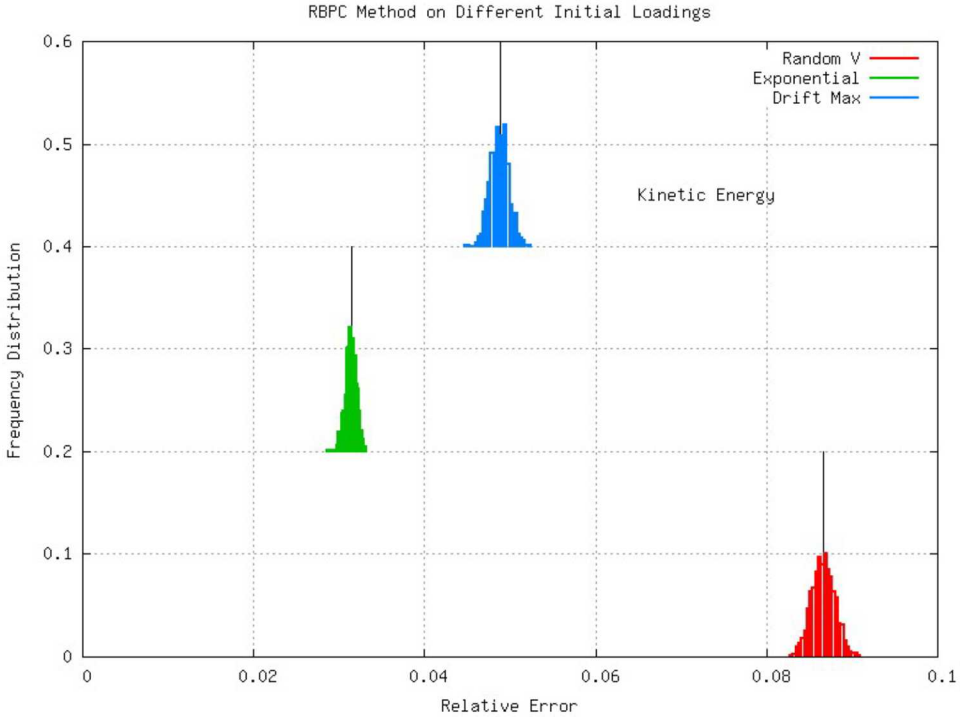


Figure 19: Relative error in the kinetic energy metric for the various initial loadings produced by the RBPC method

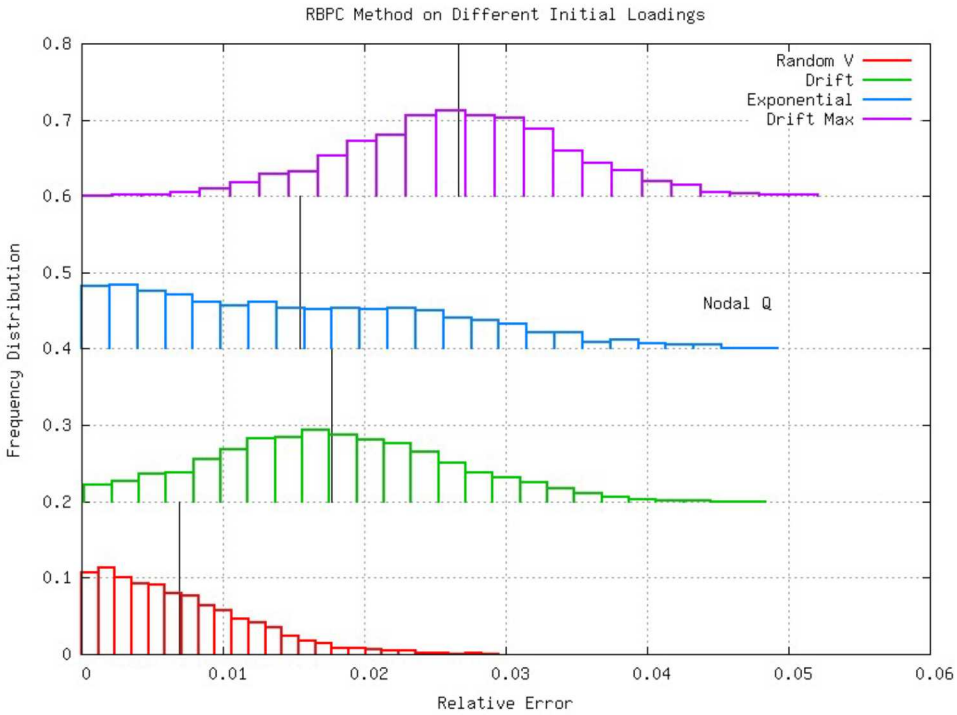


Figure 20: Relative error in the nodal Q metric for the various initial loadings produced by the RBPC method

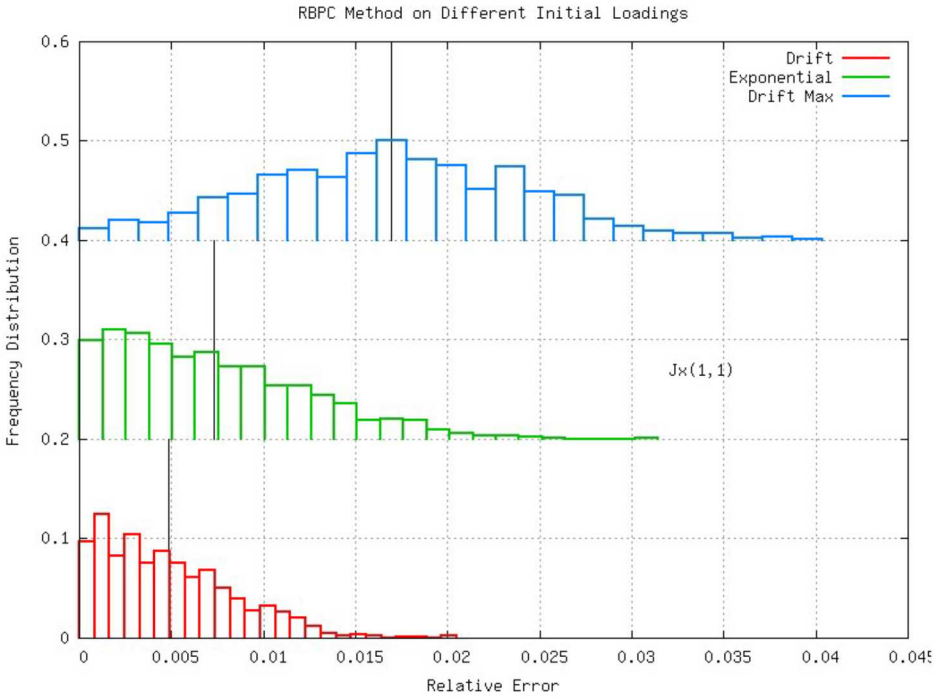


Figure 21: Relative error in the $J_x(1,1)$ metric for the various initial loadings produced by the RBPC method

4. Summary

The results of testing the RBPC metric do not bear out any great advantage to having attempted to respect the path-class phase-space distribution of the particles. It is possible that the potential improvements have been masked by choices made in implementation. Specifically, the 4 to 2 reduction picks four particles sequentially from the list of particles in the path class; this does not necessarily correlate to any aspect of spatial positioning. Thus, while the basic path-class structure is being respected, the merge can violate the velocity-position correlation with the class. This idea is taken up again in the modifed RBPC method, discussed next.

C. Modified Reduce-By-Path-Class Algorithm

1. Introduction

An aspect of the RBPC method that appeared to leave some room for improvement was the manner in which the sets of 4 particles that were to replaced by 2 particles were created. In the Buneman-Villasenor algorithm, the decomposition of a particle's complete path into sub-paths results in the determination of the path class to which this particular particle belongs. That information is written to an array in the particle data structure. When the RBPC method is executed, each path class is processed separately. The entire particle set is iterated for each path class to build local path class arrays that store the particle weight, position, and velocity data. Each path class is then processed separately. In so doing, the four particles that will be replaced by two particles are selected from the local arrays in the sequence in which the were placed in those local arrays. This sequence is directly related to the sequence in which the original set of particles was created, which is based on use of random spatial positioning. Thus, in pulling particles off the local data arrays four at a time, the positions of those four particles are random over the spatial extent of the path class to which they belong. Figure 22 shows the spatial extents path classes 9, 10, 11, and 12 for the case of initial velocity loading IC2. While path classes 9, 10, and 11 do have some degree of restriction over the area of the home cell, all of the path classes shown have at least one spatial coordinate that spans the entire home cell. Thus, each set of 4 particles pulled off the path class arrays will have random spatial positions over the spatial support of the given path class.

The modified method calculates both spatial and velocity distribution functions for each path class using the original particle set, then samples from those distribution functions to determine both positions and velocities of the replacement particles. It turns out that frequently several of the path classes are sparsely populated to the degree that the position and velocity distribution functions are extremely noisy. This resulted in the option to retain the 4:2 position and velocity assignment developed for the original RBPC method for sparsely populated path classes while employing the construction and inversion of position and velocity distribution functions for path classes that contained more than a user-defined threshold number of original particles.

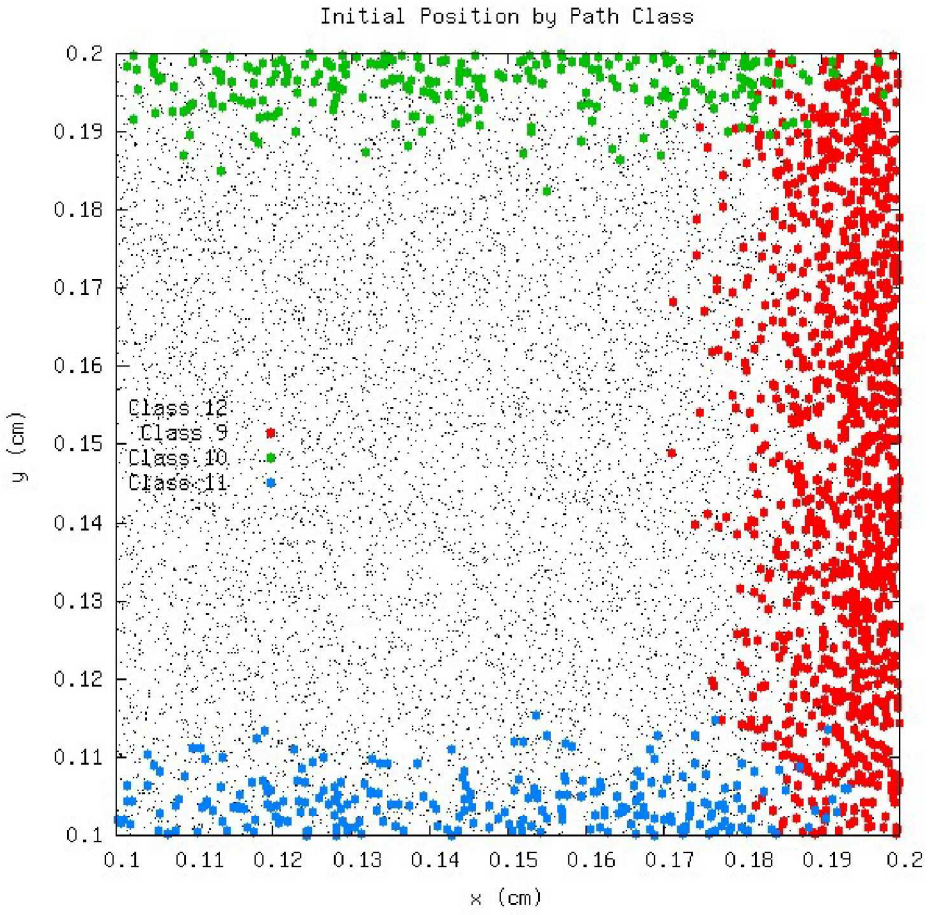


Figure 22: Spatial distribution over the home cell of several path classes for the "Random V" (IC2) initial loading

2. Description

The 4:2 merge is described in detail in Section III.B.2. The more richly populated path classes are merged using positional and velocity distribution functions. The original set of particles is sliced in the x-direction to build the x-position distribution function and is sliced in the y-direction to build the y-position distribution function. All of the original particles within the given path class are used to build a single v_x distribution function and a single v_y distribution function for that path class. When applied to the more populous path classes, the method manages to rebuild the path class fairly well. Figure 23 shows both the x-v_x (red=original, green=rebuilt) and y-v_y distributions (blue=original, magenta=rebuilt) of both the original and the rebuilt particle sets for Path Class 9 and initial velocity loading IC3. This is the path class for which particles' final positions are in the cell immediately to the right of the home cell. The rebuilt path class respects the sharp line that marks the spatially-dependent termination of the x- and y-velocity distributions.

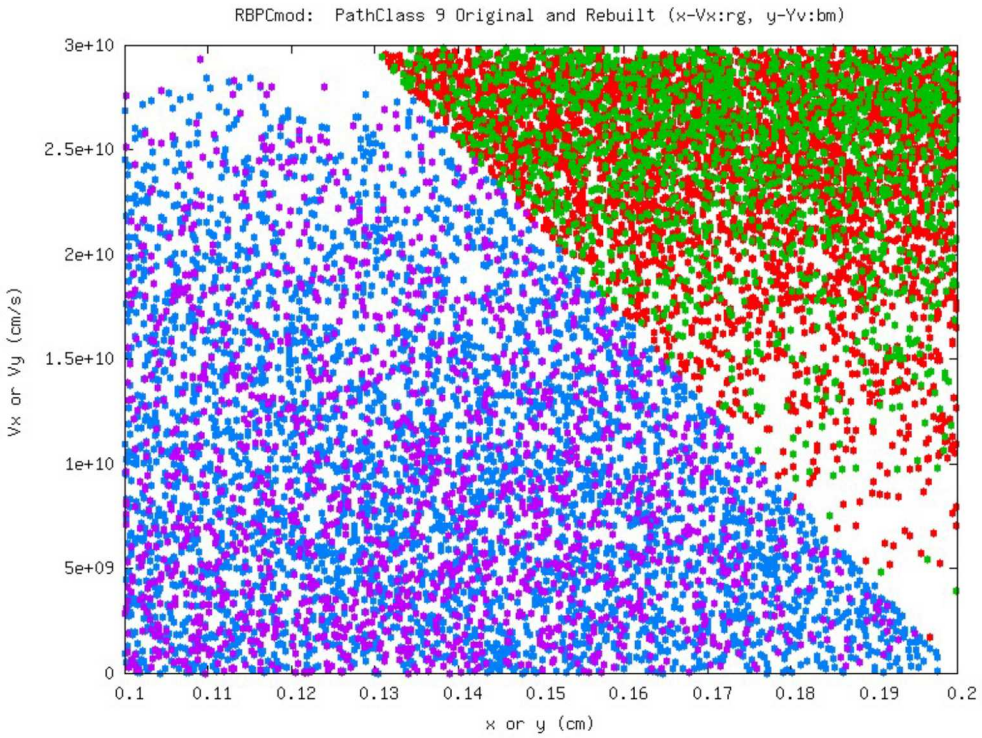


Figure 23: Original (red & green) and replacement (blue & magneta) particles for path class 9 for initial velocity loading IC3.

3. Examples

Each initial velocity loading was subjected to 1000 repetitions of the merge algorithm, each time starting with a new original particle set containing 10^4 particles. The time required for 1000 repetitions for each initial velocity loading is listed in Table 6. The mean values of the various diagnostic metrics over this set of 1000 repetitions is shown for each initial velocity loading in Table 7. Figure 13 through Figure 21 show the results of rebuilding velocity distribution functions and results on the various metrics.

Table 6: Execution time for the modified RBPC method for 1000 repetitions starting with a set containing 10^4 particles

Initial Loading Method	Time Required (s)
Random V	11.39
Drift	9.89
Exponential	11.48
Drifting Maxwellian	11.90

Table 7: Mean values of the relative errors in the various metrics for the four different initial velocity loadings by the modified RBPC method

Initial Loading	ΔQ_0	Ionization Rate	Kinetic Energy	Nodal Q	$J_x(1,1)$

Random-V	1.0681	0.02770	0.008800	0.009885	*
Drift	0.03033	0.0030838	6.861e-06	0.009971	0.006513
Exponential	0.5513	0.01780	0.01063	0.01011	0.01325
Drift Max	0.4491	0.007842	0.01268	0.009777	0.02925

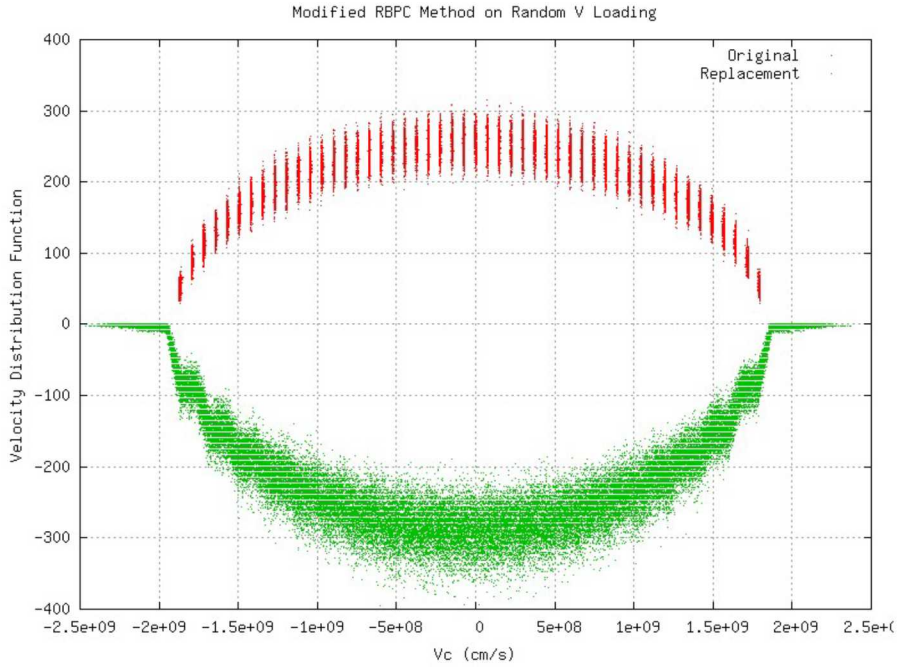


Figure 24: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellian by the RBPC method

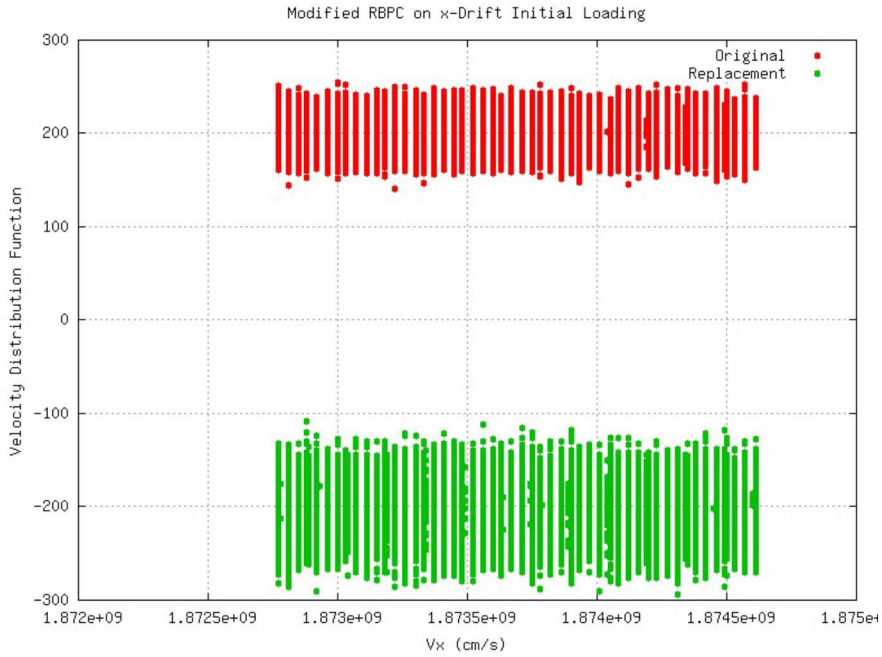


Figure 25: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellian by the RBPC method

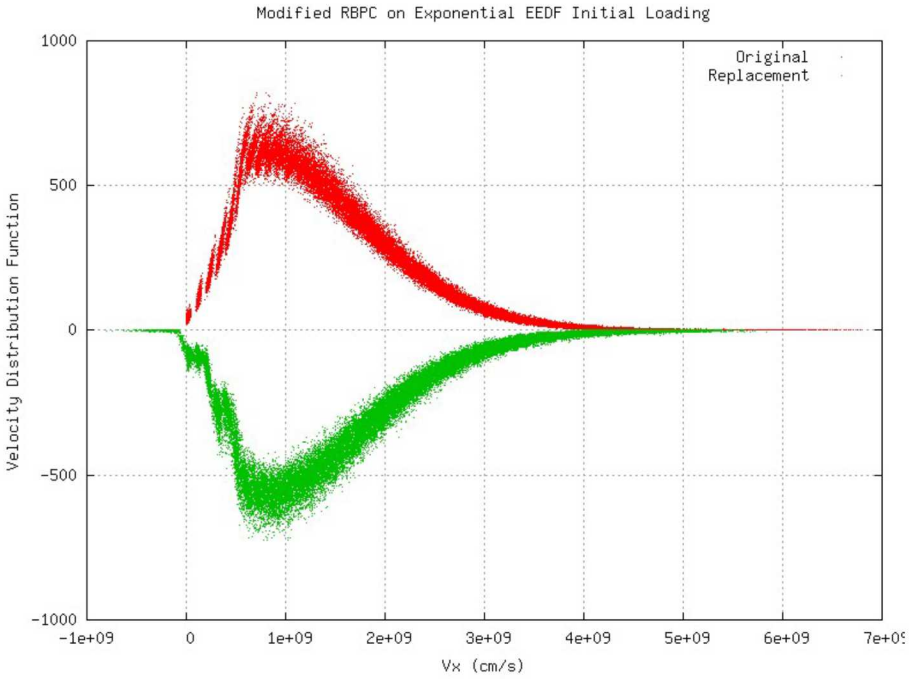


Figure 26: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellian by the RBPC method

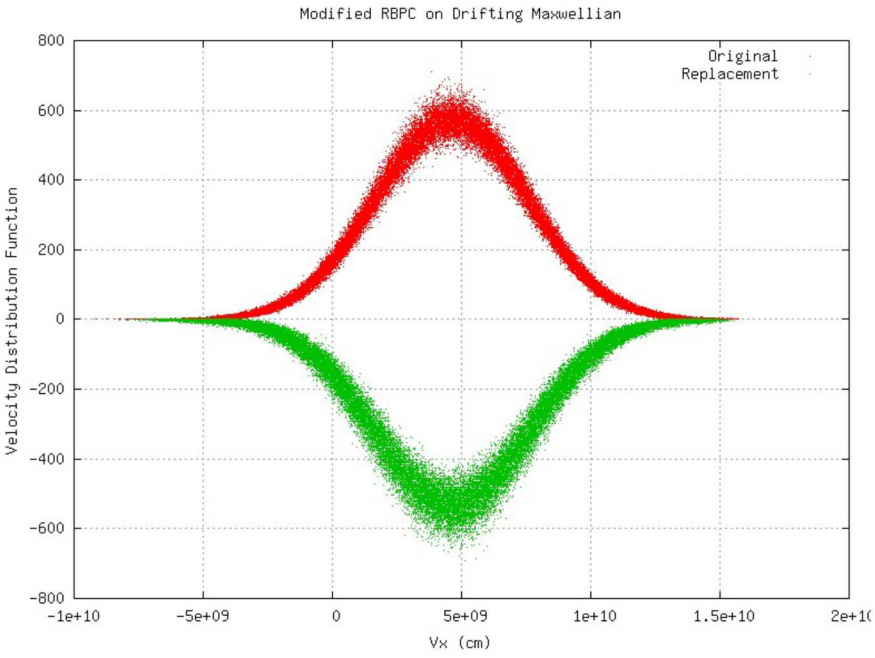


Figure 27: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellian by the RBPC method

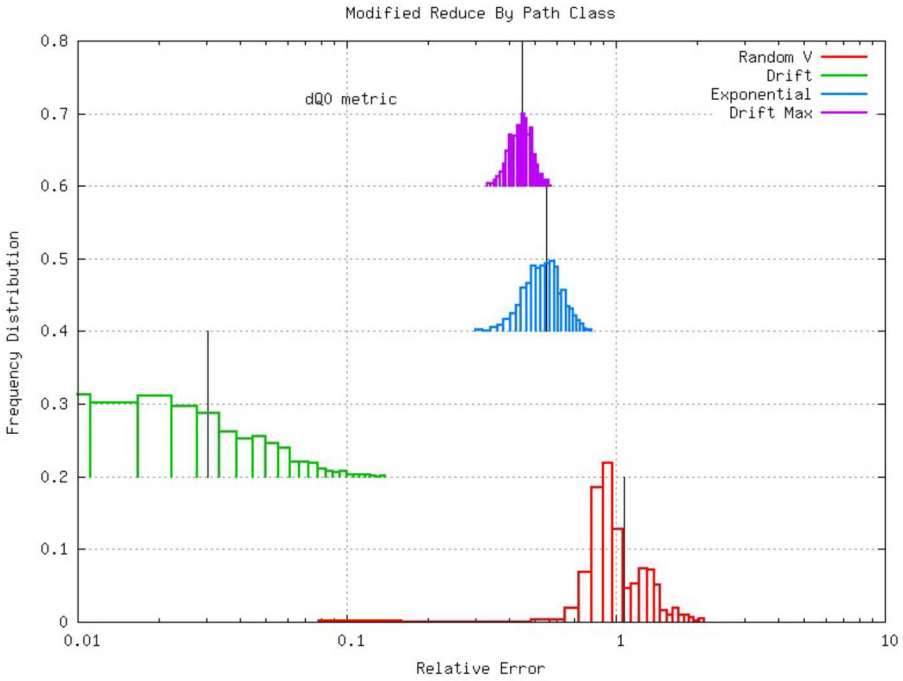


Figure 28: Relative error in the ΔQ_0 metric for the various initial loadings produced by the modified RBPC method

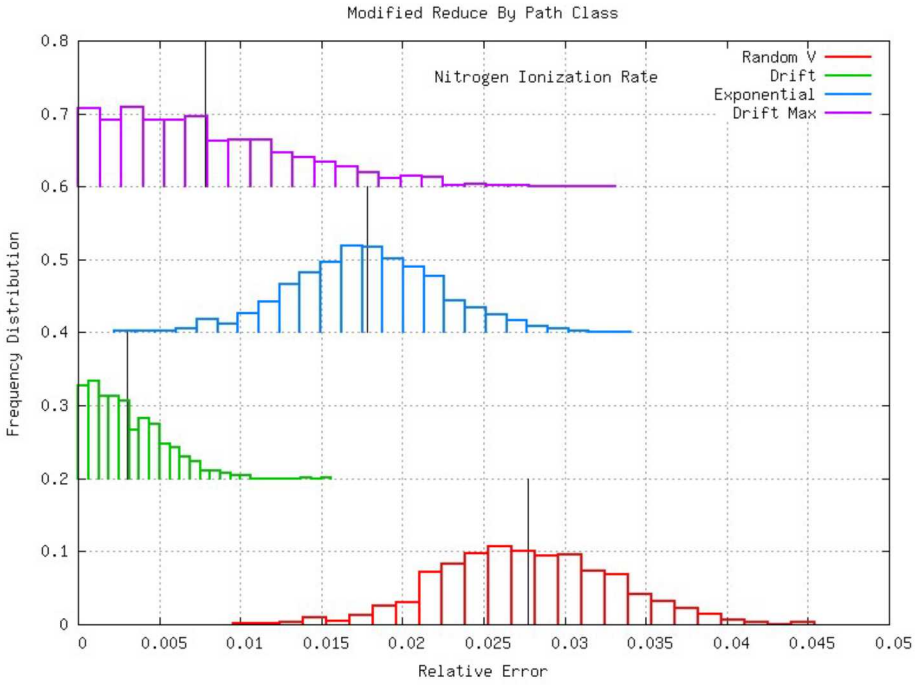


Figure 29: Relative error in the nitrogen ionization rate metric for the various initial loadings produced by the modified RBPC method

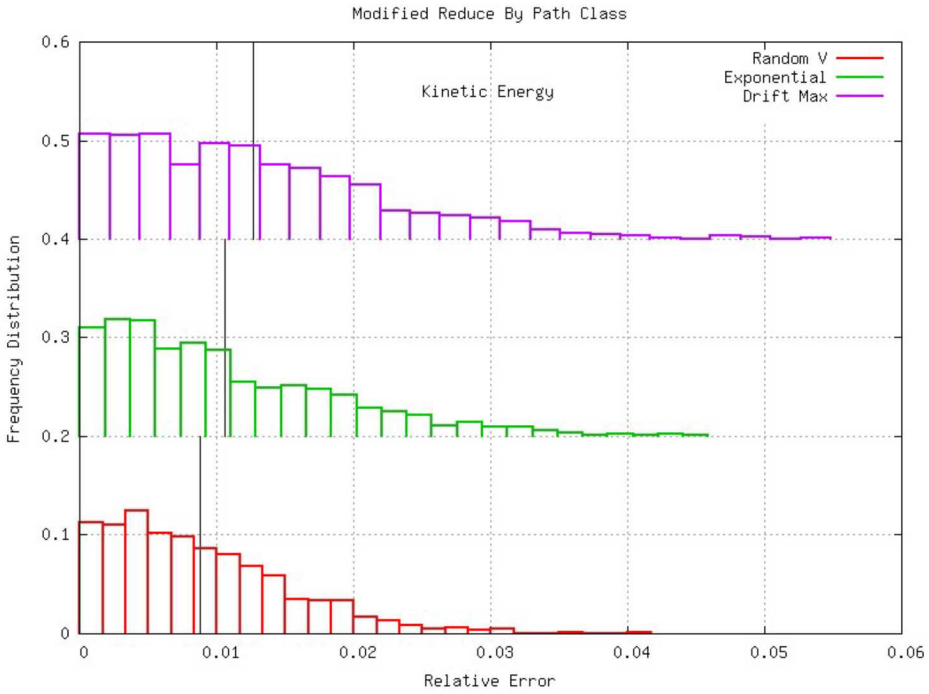


Figure 30: Relative error in the kinetic energy metric for the various initial loadings produced by the modified RBPC method

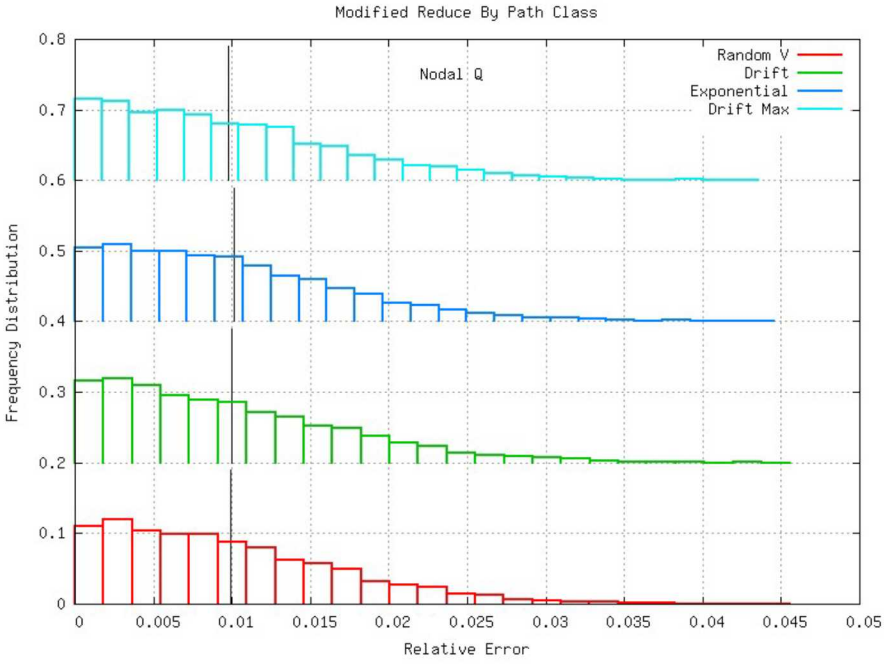


Figure 31: Relative error in the nodal Q metric for the various initial loadings produced by the modified RBPC method

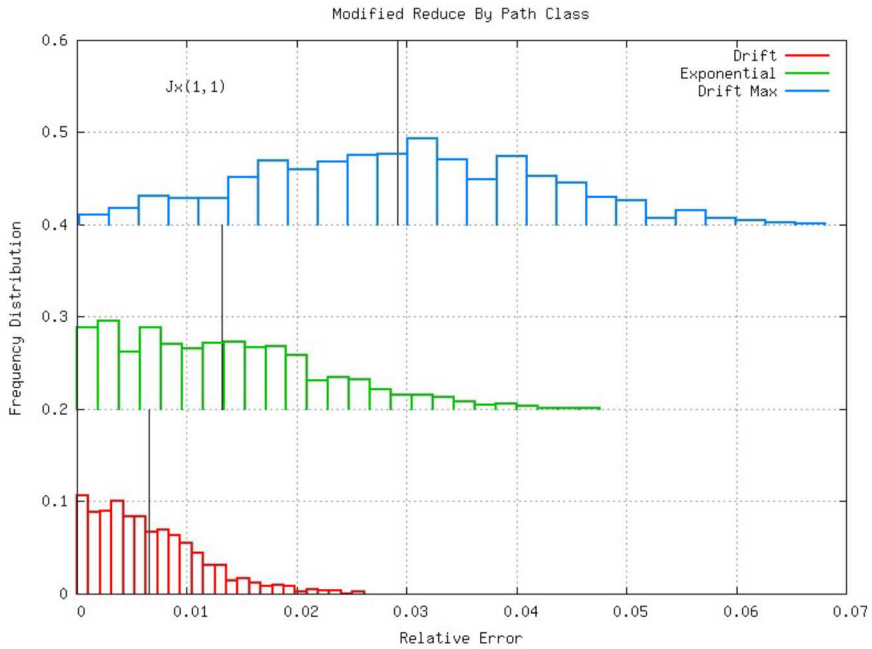


Figure 32: Relative error in the $J_x(1,1)$ metric for the various initial loadings produced by the modified RBPC method

4. Summary

The degree to which the modification introduced here actually improves the RBPC method can be judged by comparing its performance directly against the original RBPC method. The modified method required slightly longer execution time for all four initial velocity loadings (Table 4 *vs* Table 6), but as noted earlier, no attempt to optimize the algorithm for speed has been made. The rebuilt velocity distribution functions appear to be distinctly improved in the cases of initial velocity loadings IC1 and IC2 improved (compare Figure 13 and Figure 14 to Figure 24 and Figure 25). The results for IC3 are a slight improvement to an already fair result, and for IC4, little change to an adequate result can be seen. Improvement over the original RBPC method is also evident in terms of the reaction rate metric, the kinetic energy metric, and the nodal Q metric. Comparison to the ICEPIC REDUCE method shows mean relative errors in most of the metric that are close to one another. The modified RBPC method is thus an improvement over the original RBPC method, but does not appear to offer a distinct advantage over the REDUCE method.

D. Particle-Velocity Correlation Method (PVCM)

1. Introduction

The modification to the reduce-by-path-class method provides a modest improvement on the basis of the nodal charge metric, yet room for improvement certainly remains. The modified RBPC method built x-, y-, v_x- and v_y-distribution functions for each path class with number of particles exceeding a user-input threshold value. However, as shown in Figure 23, the spatial range of a given path class with respect to the home cell can be significant. That suggests that

producing homogenous velocity distribution functions for the entire class is leaving something on table with respect to position-velocity correlation.

2. Description

The obvious next step to that is to localize both v_x and v_y velocity distribution functions within subregions of the home cell, for example by imposing a grid structure on the home cell and building velocity distribution functions for each subcell. In putting this algorithm together, the segregation of particles into path classes was considered an overly burdensome, so was, at least temporarily, put aside. Thus, the PVCM method works by imposing a subgrid over the home cell, building both v_x and v_y velocity distribution functions for each resulting subcell, then sampling these distribution functions to obtain assignments for the replacement set. The positions of the replacement particles are chosen randomly over the area spanned by the given subcell.

3. Examples

Each initial velocity loading was subjected to 1000 repetitions of the merge algorithm, each time starting with a new original particle set containing 10^4 particles. The time required for 1000 repetitions for each initial velocity loading is listed in Table 8. The mean values of the various diagnostic metrics over this set of 1000 repetitions is shown for each initial velocity loading in Table 9. Figure 24 through Figure 32 show the results of rebuilding velocity distribution functions and results on the various metrics.

Table 8: Execution time for the PVCM method for 1000 repetitions starting with a set containing 10^4 particles

Initial Loading Method	Time Required (s)
Random V	10.54
Drift	9.05
Exponential	10.61
Drifting Maxwellian	10.76

Table 9: Mean values of the relative errors in the various metrics for the four different initial velocity loadings by the PVCM method

Initial Loading	ΔQ_0	Ionization Rate	Kinetic Energy	Nodal Q	$J_x(1,1)$
Random-V	0.1672	0.01911	0.03573	9.757e-04	*
Drift	0.04433	4.357e-04	6.610e-05	9.706e-04	1.339e-03
Exponential	0.1760	7.7712e-03	0.1936	9.757e-04	0.06890
Drift Max	0.1148	0.04754	0.1300	9.741e-04	0.07179

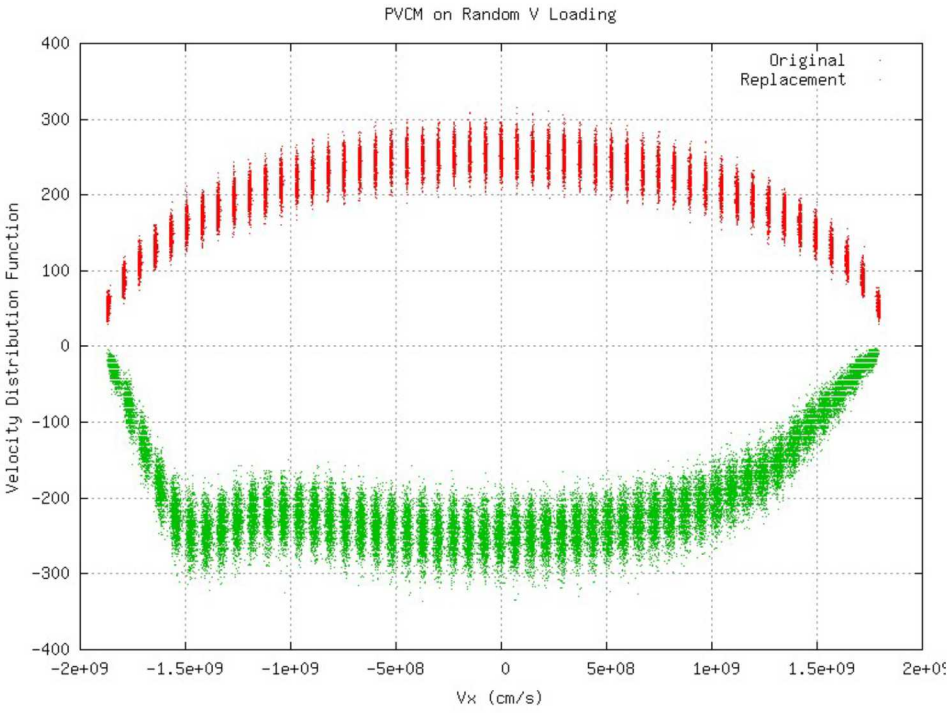


Figure 33: Comparison of original and replacment x-component velocity distribution function for the random V initial loading by the PVC method

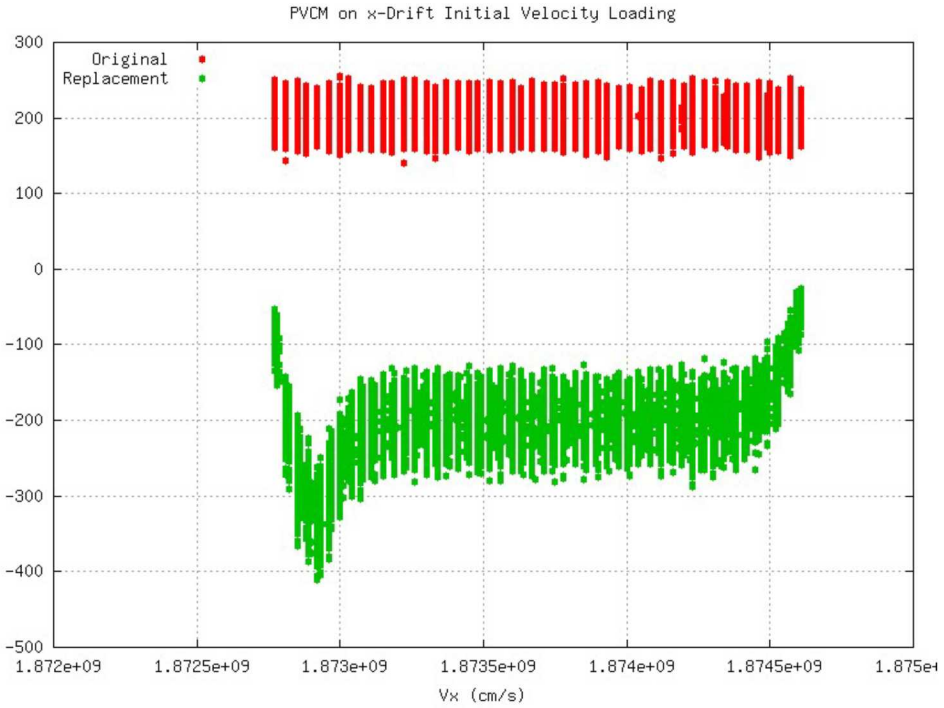


Figure 34: Comparison of original and replacment x-component velocity distribution function for the x-drift initial loading by the PVC method

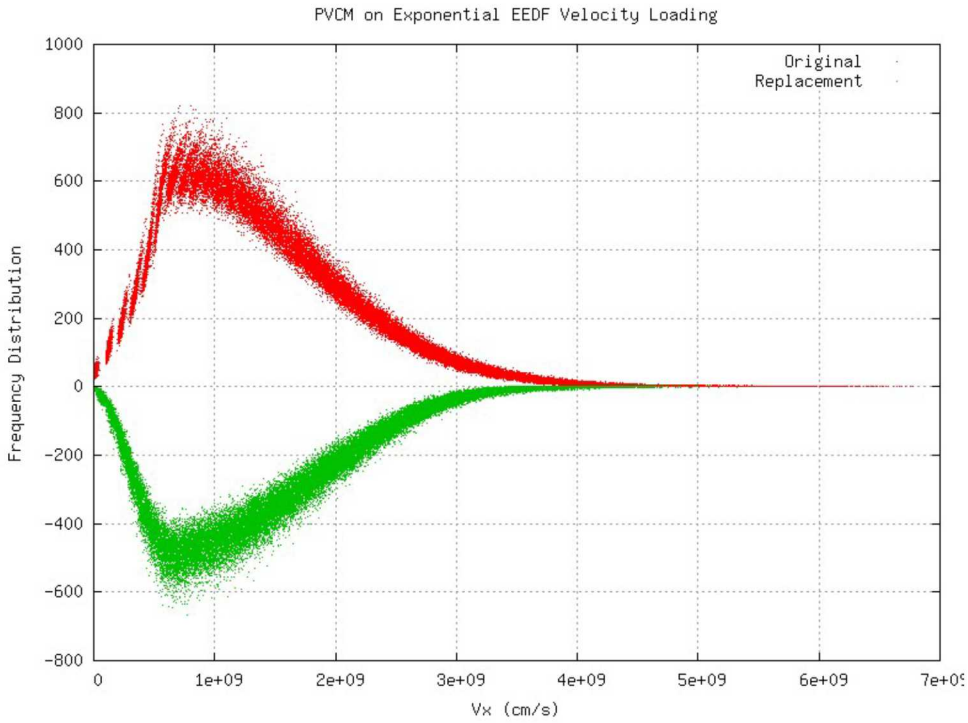


Figure 35: Comparison of original and replacment x-component velocity distribution function for the exponential EEDF initial loading by the PVC method

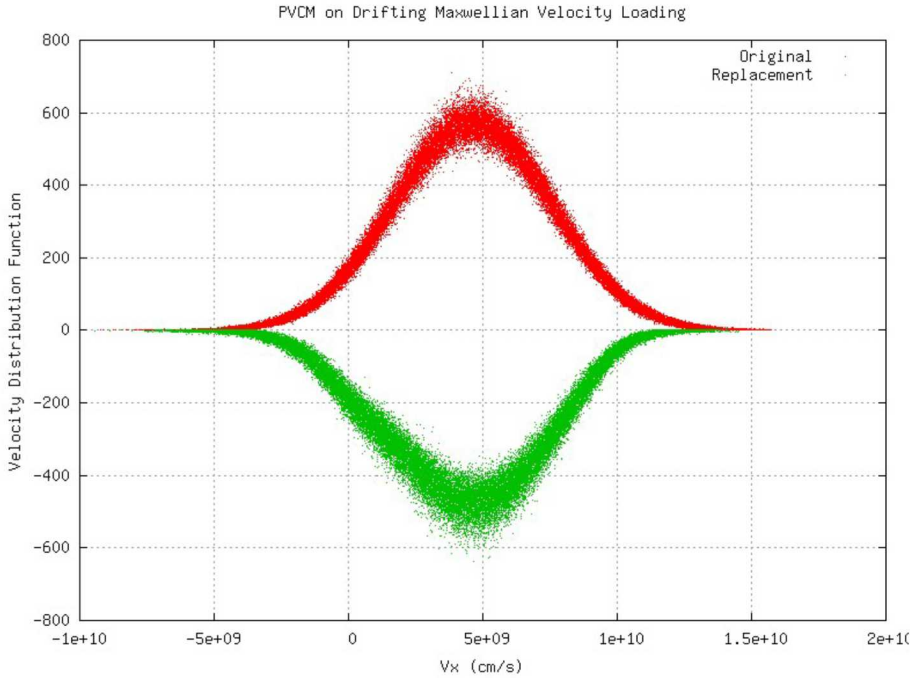


Figure 36: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellian initial loading by the PVC method

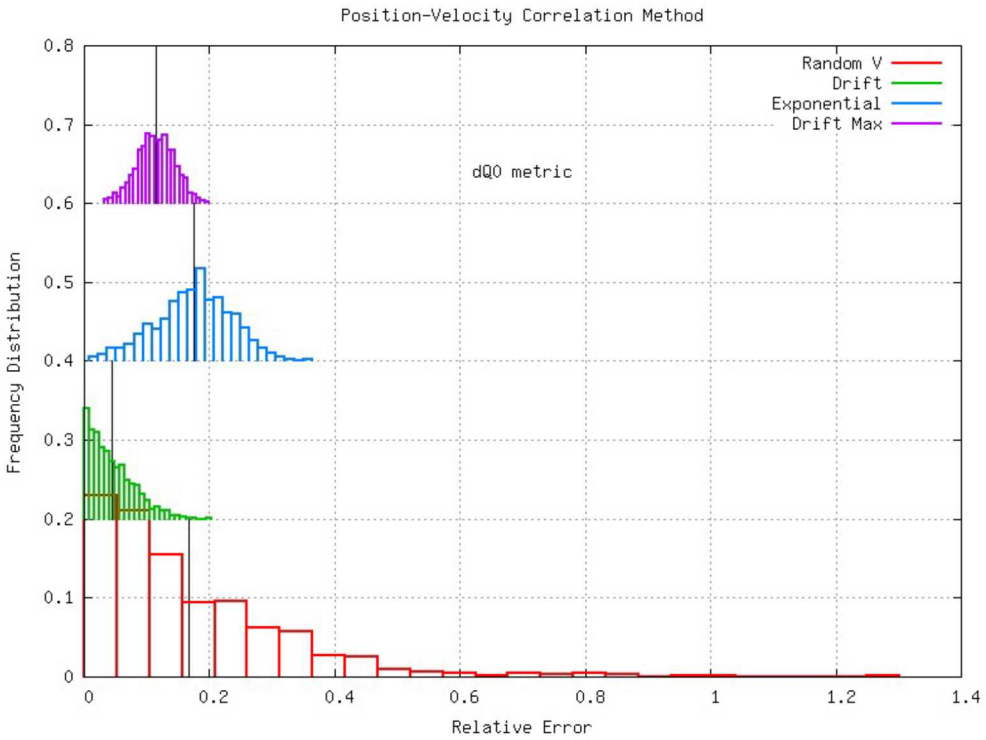


Figure 37: Relative error in the ΔQ_0 metric for the various initial loadings produced by the PVC method

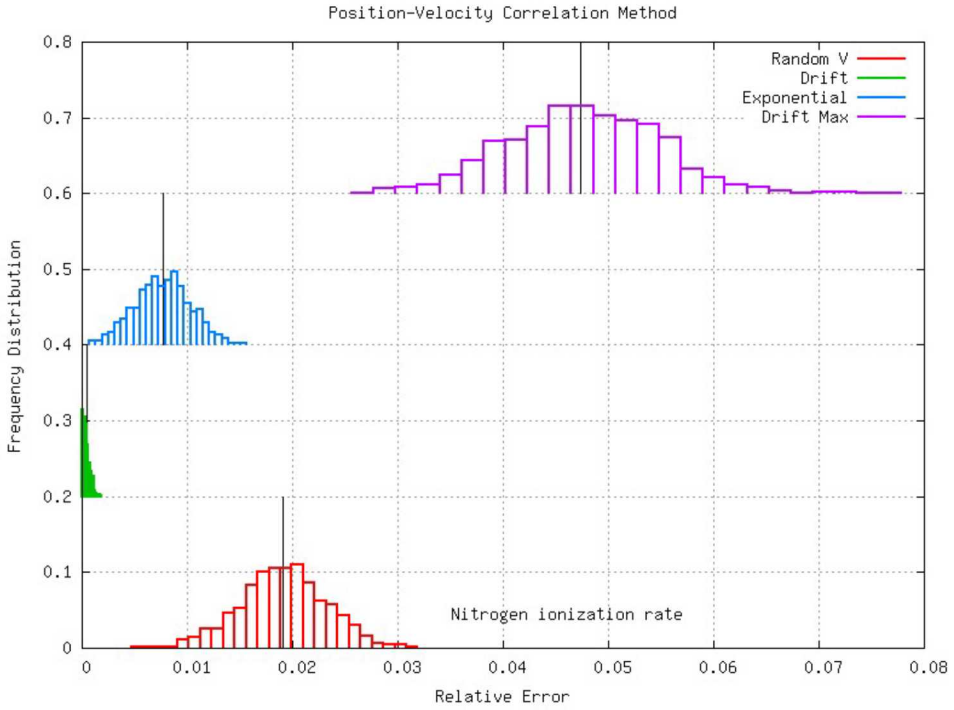


Figure 38: Relative error in the nitrogen ionization metric for the various initial loadings produced by the PVC method

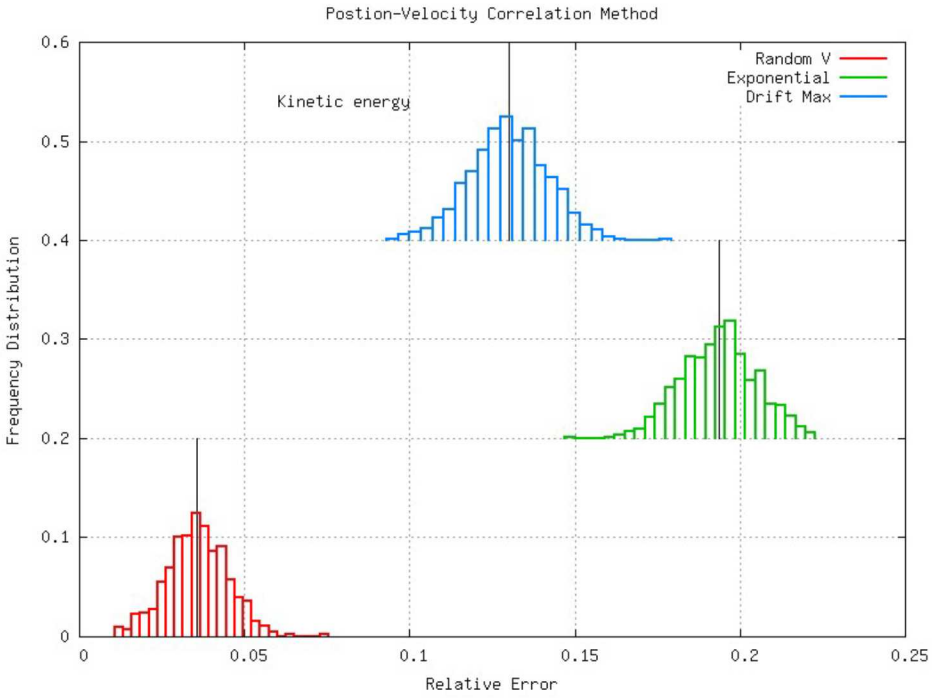


Figure 39: Relative error in the kinetic energy metric for the various initial loadings produced by the PVC method

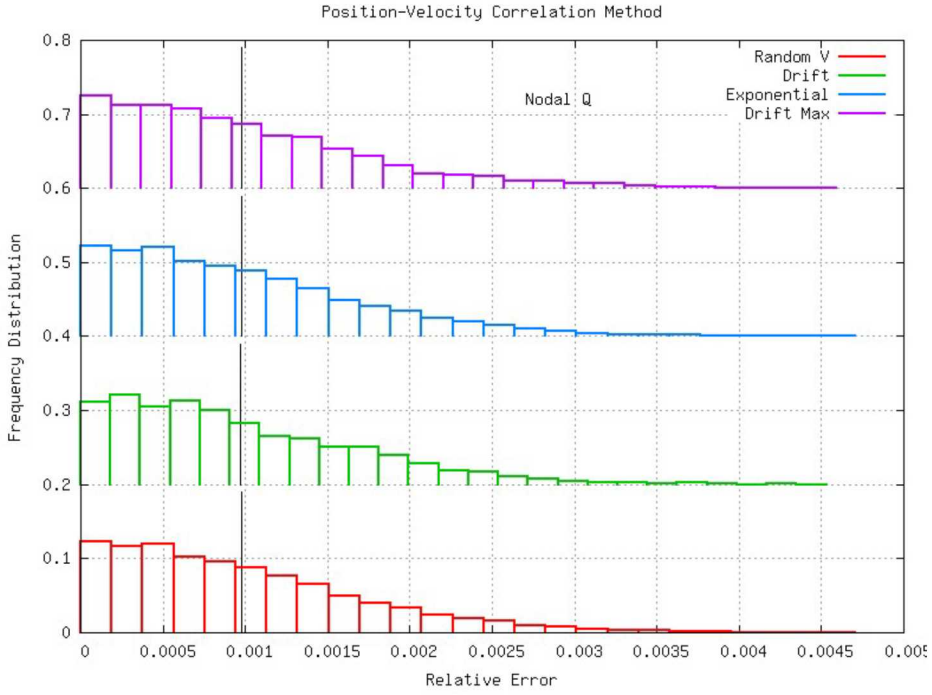


Figure 40: Relative error in the nodal Q metric for the various initial loadings produced by the PVC method

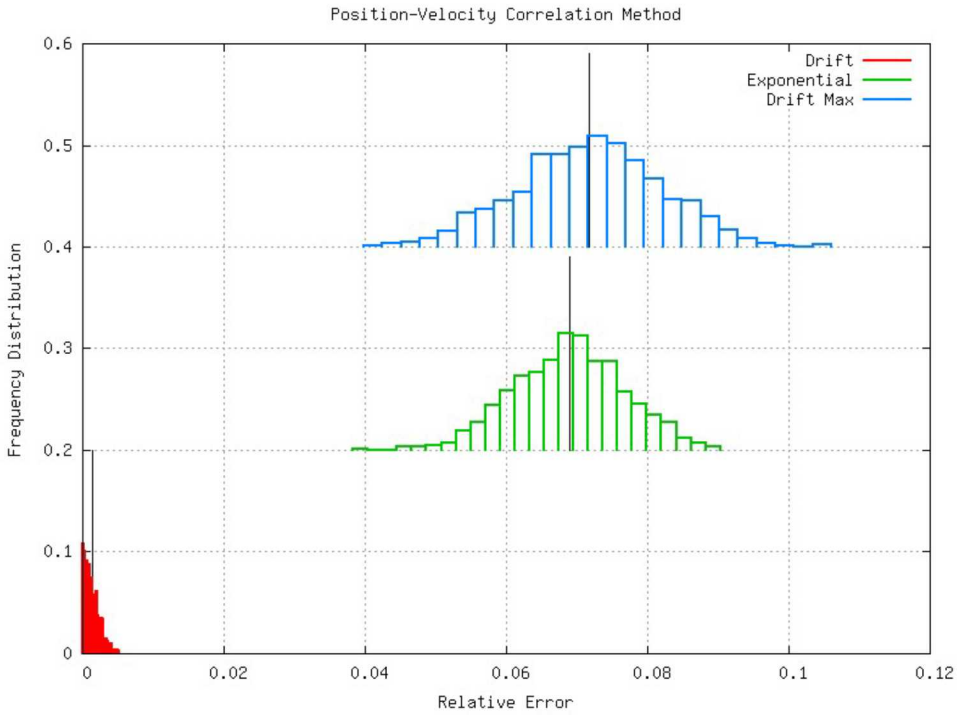


Figure 41: Relative error in the $J_x(1,1)$ metric for the various initial loadings produced by the PVC method

4. Summary

This method clearly had some difficulty rebuilding the IC1 and IC2 velocity distribution functions. The results for IC3 and IC4 look better, but this is likely a case of the more complex distribution functions hiding the errors made so clear in the simple functions of IC1 and IC3. The comparison of performance on the various metrics show superior results in terms of the reaction rate and the nodal-Q metrics, but poorer results in terms of kinetic energy and the Jx11 metric.

E. Subgrid M:2 Method

1. Introduction

This was our initial forray into merge algorithms beyond the ICEPIC REDUCE method. It helped drive the development of BVcomp in terms of bringing together the basic tools and metrics needed for assessment.

2. Description

This method imposes a rectangular N×N subgrid over the home cell and within each subgrid cell, performs the M:2 reduction described in Section III.B.2.

3. *Examples*

Execution times for this method on the various initial velocity loadings are listed in Table 10. The times are significantly longer that those for the other algorithms. This reflects the comment initially made in regard to the timing data, namely that no attempt to optimize with respect to execution speed had been made. The long execution time in this case results from processing the set of original particles at least once for each of the N×N subgrid cells. In the following method, which is structurally very similar, the algorithm was structured such that the pass over the original particle set was the external loop and the iteration of the N×N subgrid cells was the inner loop. This resulted in a large decrease in execution times as can be seen by comparing the data in Table 10 to that in XXX.

Table 10: Execution time for the SubGrid M:2 method for 1000 repetitions starting with a set containing 10^4 particles

Initial Loading Method	Time Required (s)
Random V	103.55
Drift	102.71
Exponential	104.41
Drifting Maxwellian	104.24

Table 11: Mean values of the relative errors in the various metrics for the four different initial velocity loadings by the PVCM method

Initial Loading	ΔQ_0	Ionization Rate	Kinetic Energy	Nodal Q	$J_x(1,1)$
Random-V	0.1672	0.01911	0.03573	9.757e-04	*
Drift	0.04433	4.357e-04	6.610e-05	9.7056e-04	1.339re-03
Exponential	0.1760	7.7712e-03	0.1936	9.757e-04	0.06890
Drift Max	0.1148	0.04754	0.1300	9.741e-04	0.07179

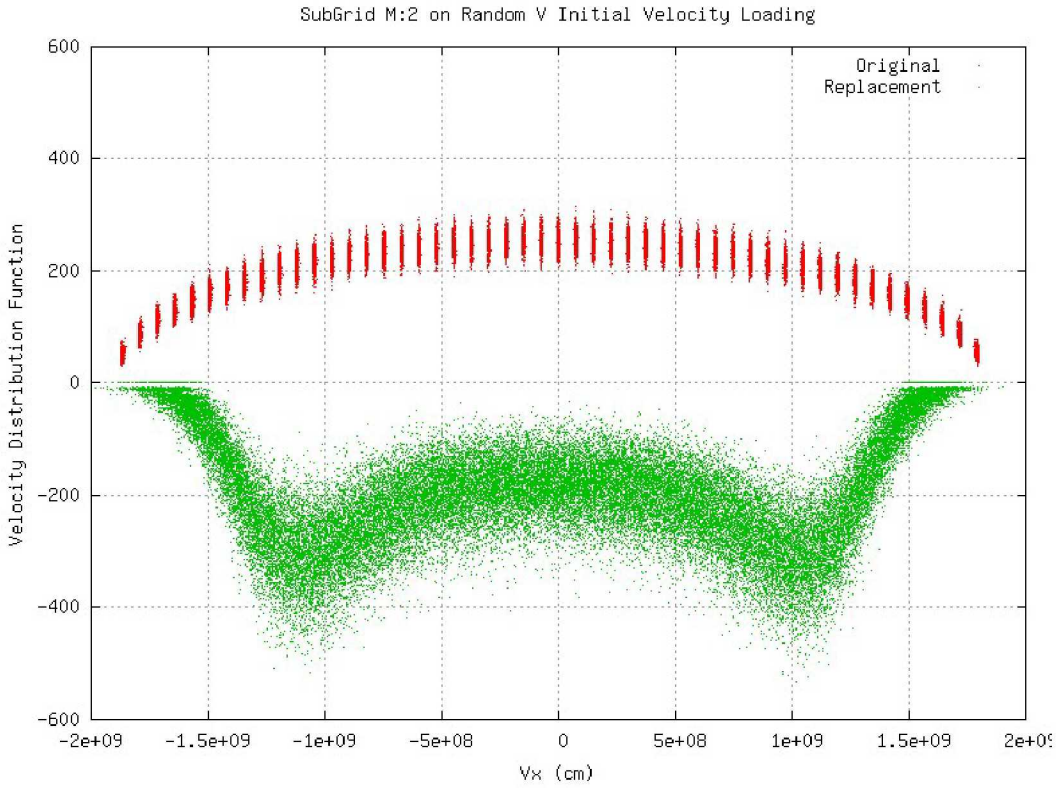


Figure 42: Comparison of original and replacment x-component velocity distribution function for the random V initial loading by the subgrid M:2 method

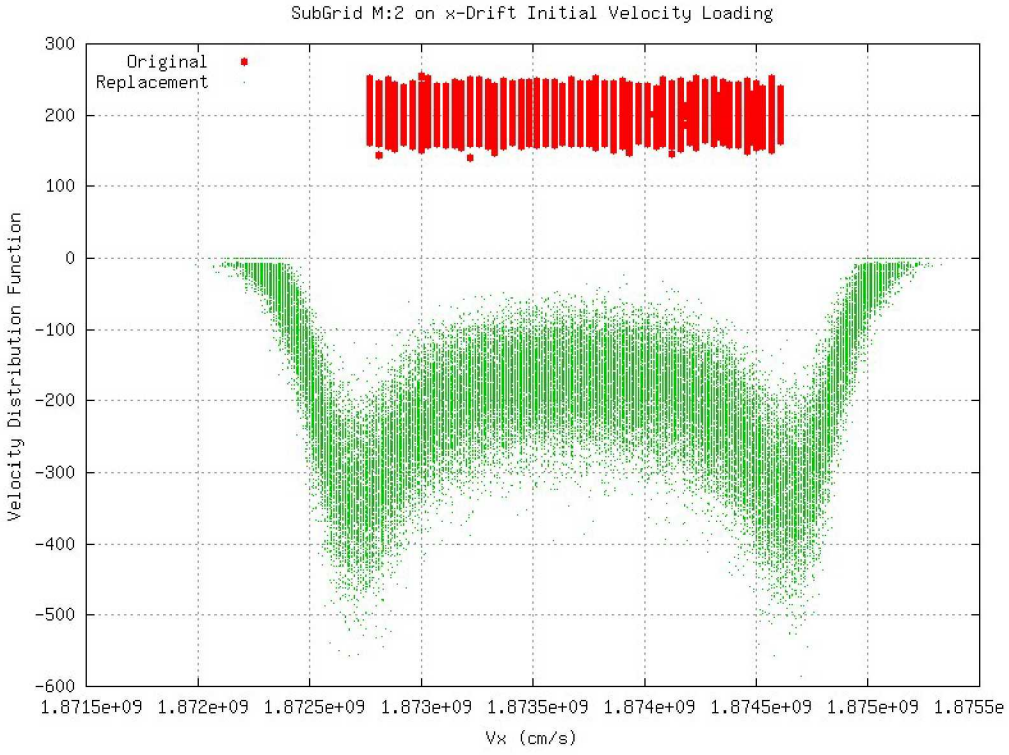


Figure 43: Comparison of original and replacment x-component velocity distribution function for the x-drift initial loading by the subgrid M:2 method

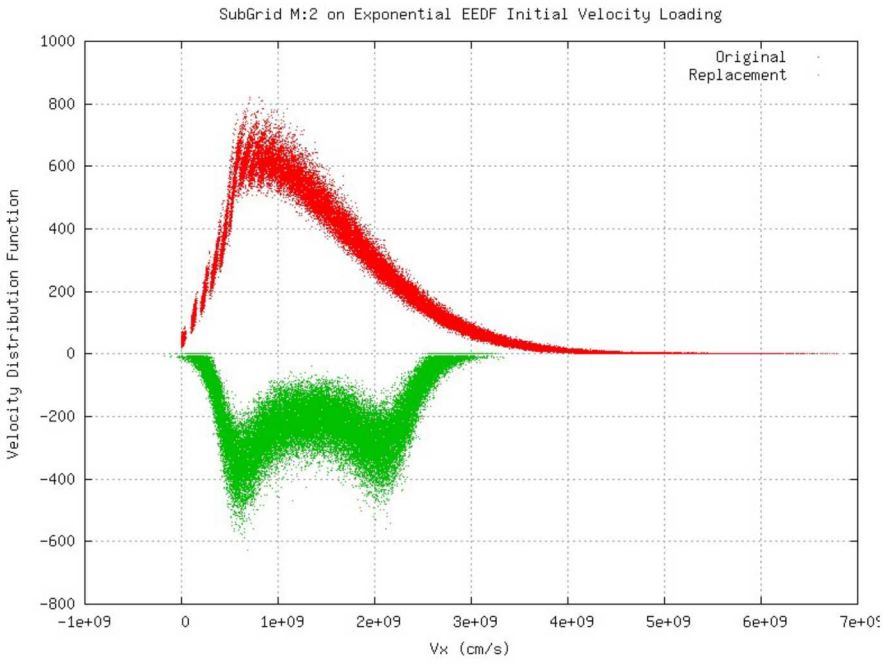


Figure 44: Comparison of original and replacment x-component velocity distribution function for the exponential EEDF initial loading by the subgrid M:2 method

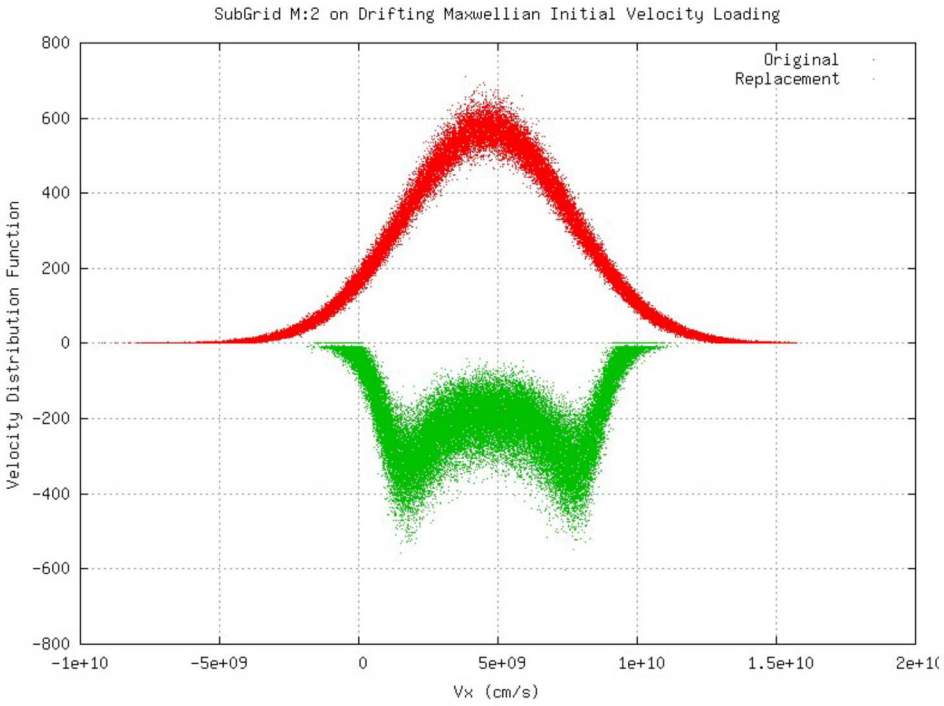


Figure 45: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellian initial loading by the subgrid M:2 method

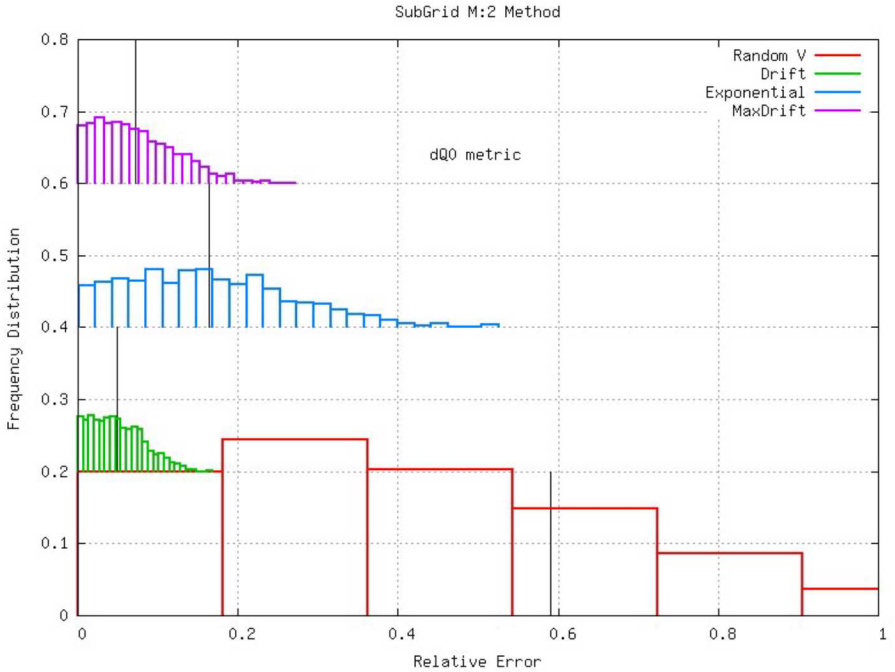


Figure 46: Relative error in the ΔQ_0 metric for the various initial loadings produced by the SubGrid M:2 method

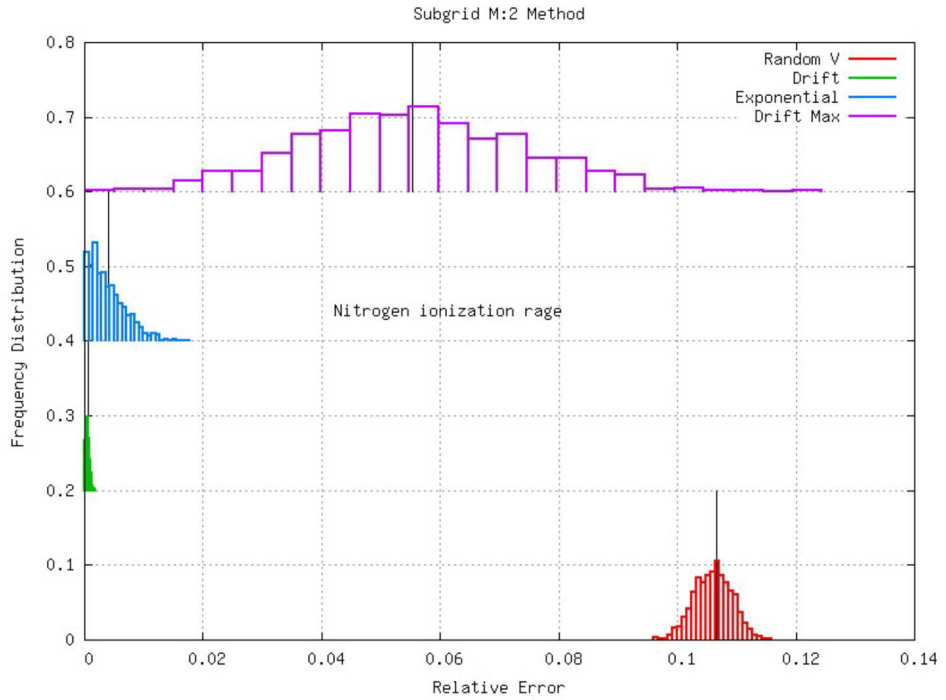


Figure 47: Relative error in the nitrogen ionization rate metric for the various initial loadings produced by the SubGrid M:2 method

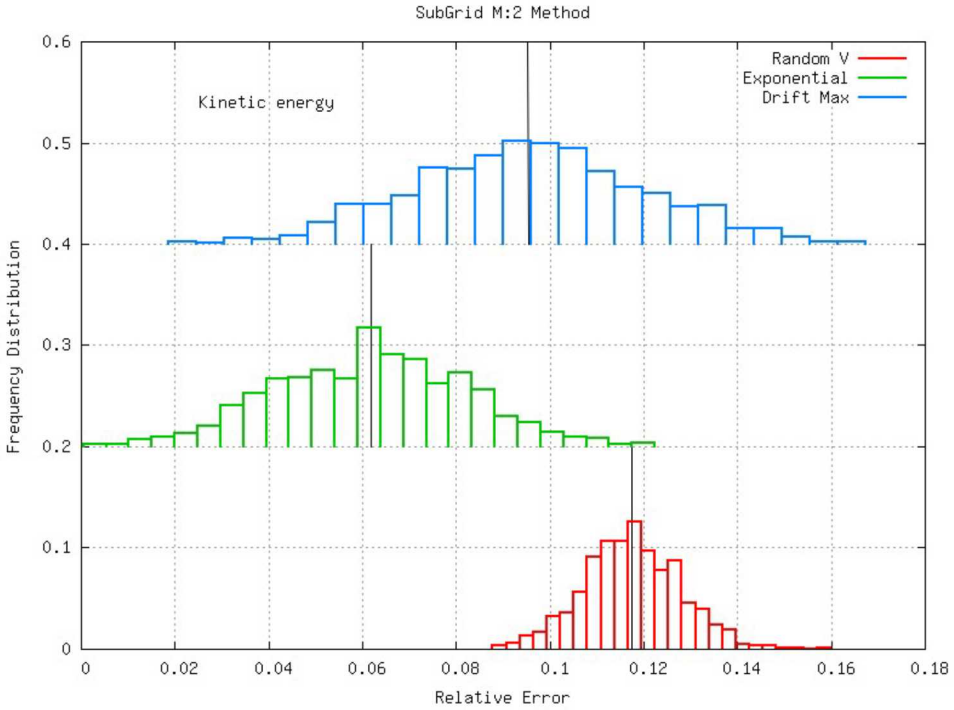


Figure 48: Relative error in the kinetic energy metric for the various initial loadings produced by the SubGrid M:2 method

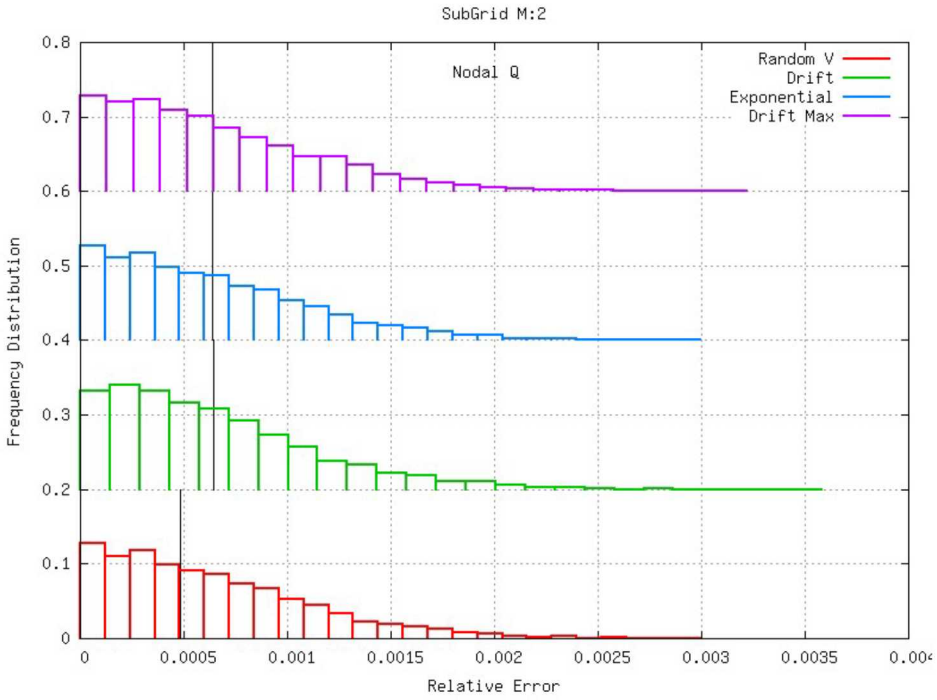


Figure 49: Relative error in the nodal Q metric for the various initial loadings produced by the SubGrid M:2 method

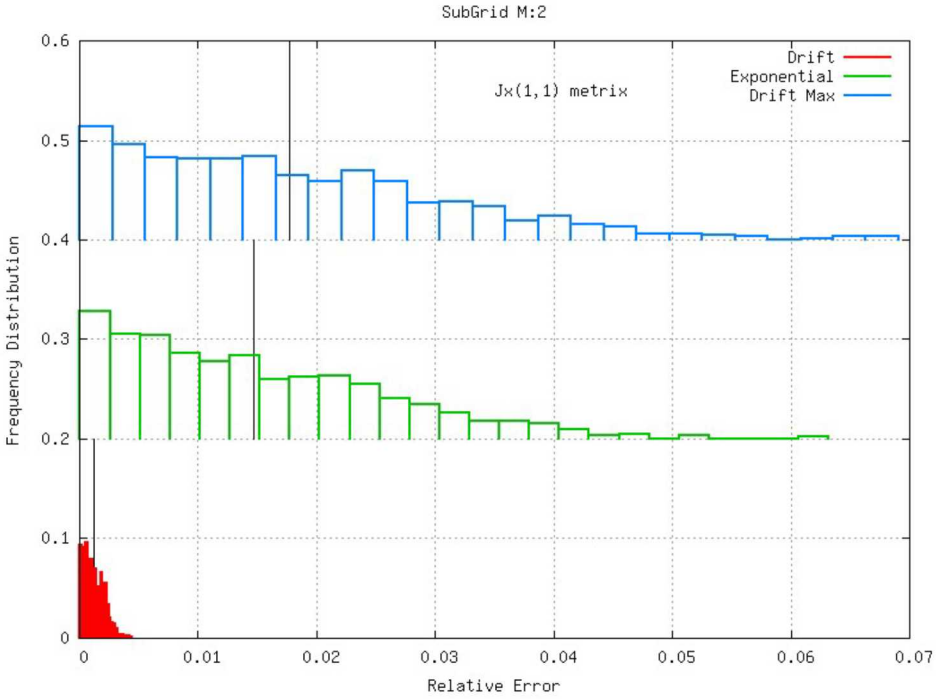


Figure 50: Relative error in the $J_x(1,1)$ metric for the various initial loadings produced by the SubGrid M:2 method

4. Summary

This method performed fairly well in terms of the nodal Q metric, but not so well in terms of the kinetic energy metric. Other results were mixed. The execution speed can probably be improved. The performance in terms of rebuilding the velocity distribution functions, however, is not good. This would seem to be reason enough not to use this method as is.

F. Modified Subgrid Method

1. Introduction

The subgrid method did fairly well on the nodal Q metric. Further consideration of this method led to the notion that velocities of the two particles that would be replacing the particles within any given subgrid cell could be chosen to conserve kinetic energy identically. Whether this would help or hurt the method's performance with respect to rebuilding velocity distribution functions was not clear, so the decision was made to build it and test it.

2. Description

The M:2 method calculates the center-of-momentum velocity for the original particles within each subgrid cell as well as the thermal speed for those particles. These velocities are then used to determine the velocities of the two new particles. The notion investigated here was that these quantities could be used so that the total kinetic energy of the two replacement particles would be identical to the total kinetic energy of the original particles within that subgrid cell. To do so, a parameter ρ was introduced such that

* MERGEFORMAT (1.8)

$$
\begin{aligned}
v_{x,1} &= (1-\rho)\langle V_x \rangle + \rho V_{th} \\
v_{y,1} &= (1-\rho)\langle V_x \rangle - \rho V_{th} \\
v_{x,2} &= (1-\rho)\langle V_y \rangle - \rho V_{th} \\
v_{y,2} &= (1-\rho)\langle V_y \rangle + \rho V_{th}
\end{aligned}
$$

The relativistically correct kinetic energy is calculated from these velocity components and compared to the total kinetic energy of the original particles within the given subgrid cell. The contributions of the two particles to the new total kinetic energy are weighted by their weights which were calculated using the nodal Q conservation method used in the Sandia merge algorithm (see Section III.G.2.a). A search-by-bisection method is used to determine that value of ρ required to make the total kinetic energies equal to one another. It was found that this works almost always, but there are times when the old total kinetic energy is less that the total new kinetic energy as expressed through the formulation of new velocity components for all values of ρ. In such a case, this particular sub-grid is simply emptied of particles. This represents a loss of kinetic energy and of charge, but can be reduced through choice of the number of subgrid divisions. Fewer divisions result in more richly populated subgrid cells, tending to increase the

total subgrid cell total kinetic energy and reducing the likelihood that the parametric formulation of kinetic energy will fail.

3. Examples

This method is structurally very similar to the original SubGrid M:2 method, but in creating it, it was observed that reordering the sequence of loops over the original particle set and the subgrid cells could provide a significant reduction in the work have to be done. The runtimes, listed in Table 12, decreased by roughly a factor of four in comparison to those of the original SubGrid M:2 method.

The original and rebuilt velocity distribution functions for the various initial velocity loadings are shown in Figure 51 through Figure 54. These results are so poor that there seemed to reason to conduct the full assessment of performance on the standard slate of metrics.

The highly structured rebuilt velocity distribution functions, particularly as exhibited in Figure 52, does not qualify the entire method for rejection based solely on the esthetics of the rebuilt function. The introduction of a velocity distribution function with regions of positive slope ($\frac{\partial f}{\partial v} > 0$) could violate the Penrose criterion, which guarantees that a velocity distribution function with negative slope is stable with respect to exchange of energy between particles and waves, yet could be unstable if the slope is anywhere positive. Thus, using the highly structured velocity distribution functions such as those shown here could result in local instability growth. This could push the physics away from where it would have otherwise evolved, and is thus a sound basis for rejection of the method.

Table 12: Execution times for the modified subgrid model on the various initial velocity loadings

Initial Loading Method	Time Required (s)
Random V	24.23
Drift	22.73
Exponential	26.39
Drifting Maxwellian	24.76

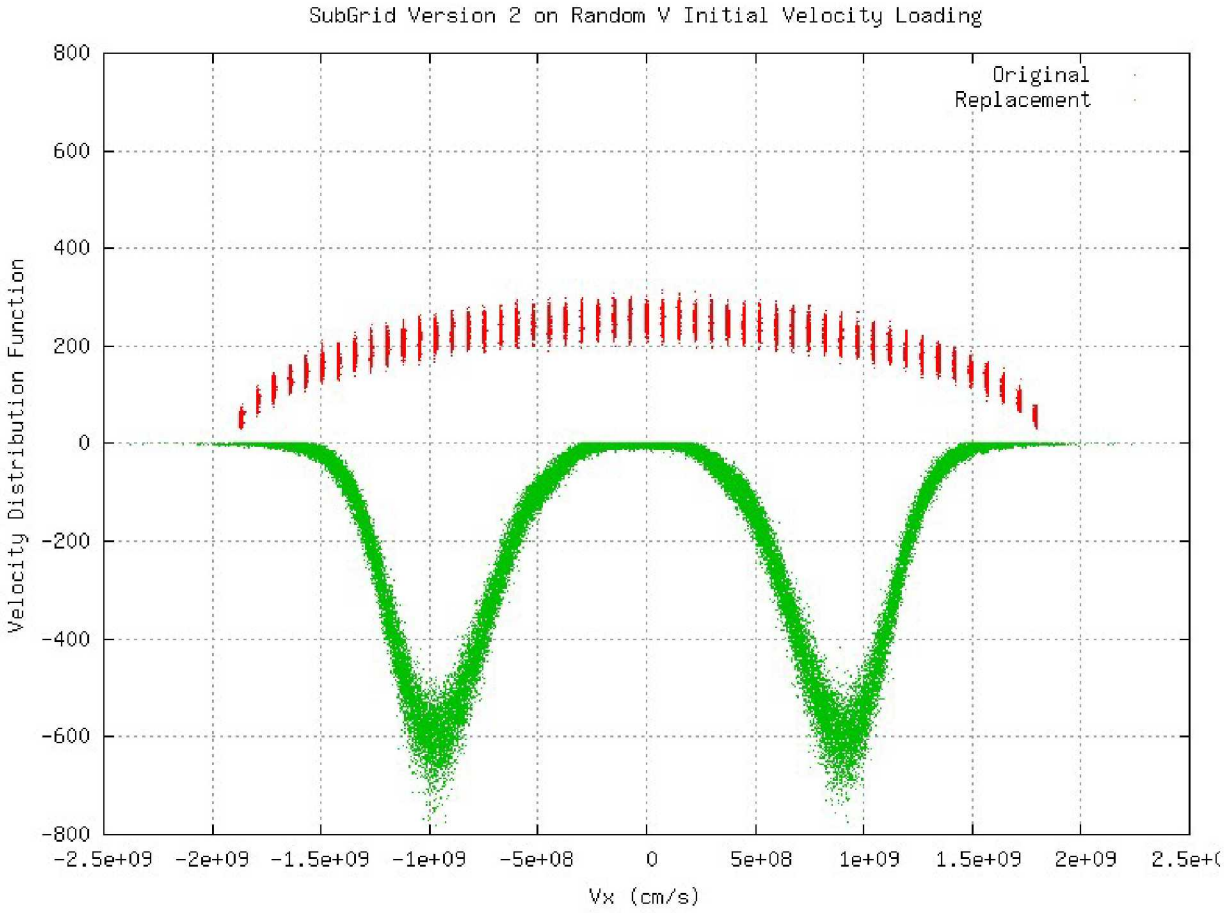


Figure 51: Comparison of original and replacment x-component velocity distribution function for the exponential EEDF initial loading by the subgrid v2 method

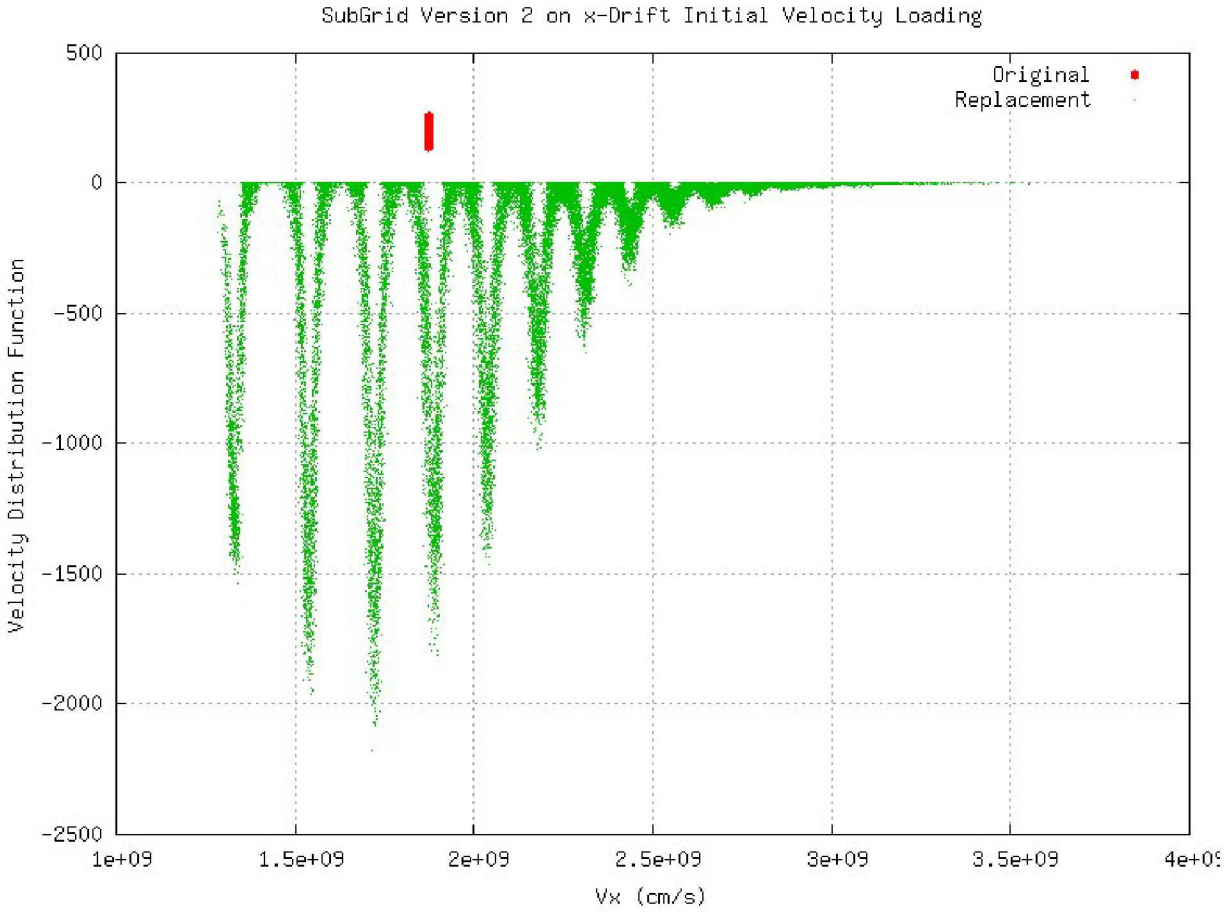


Figure 52: Comparison of original and replacment x-component velocity distribution function for the x-drift initial loading by the subgrid v2 method

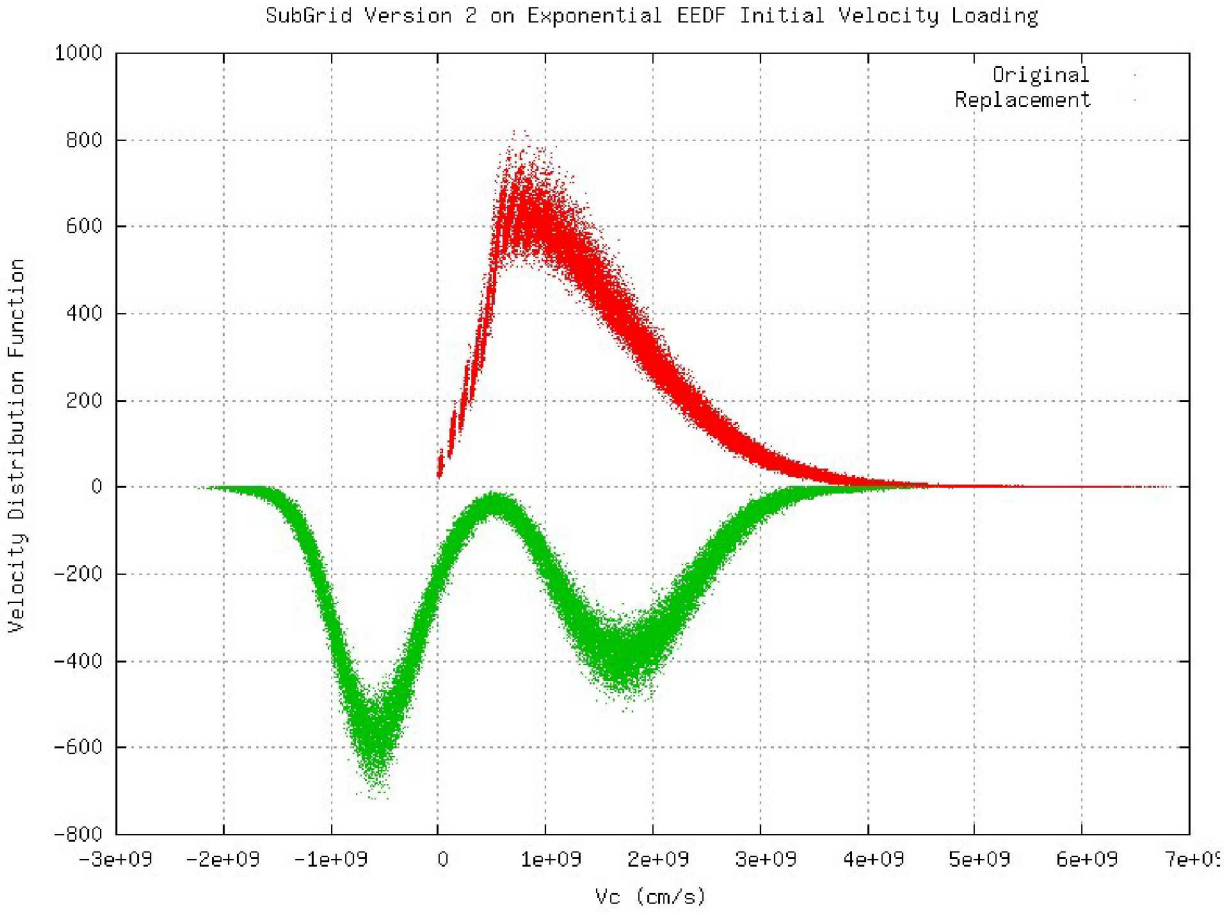


Figure 53: Comparison of original and replacment x-component velocity distribution function for the exponential EEDF initial loading by the subgrid v2 method

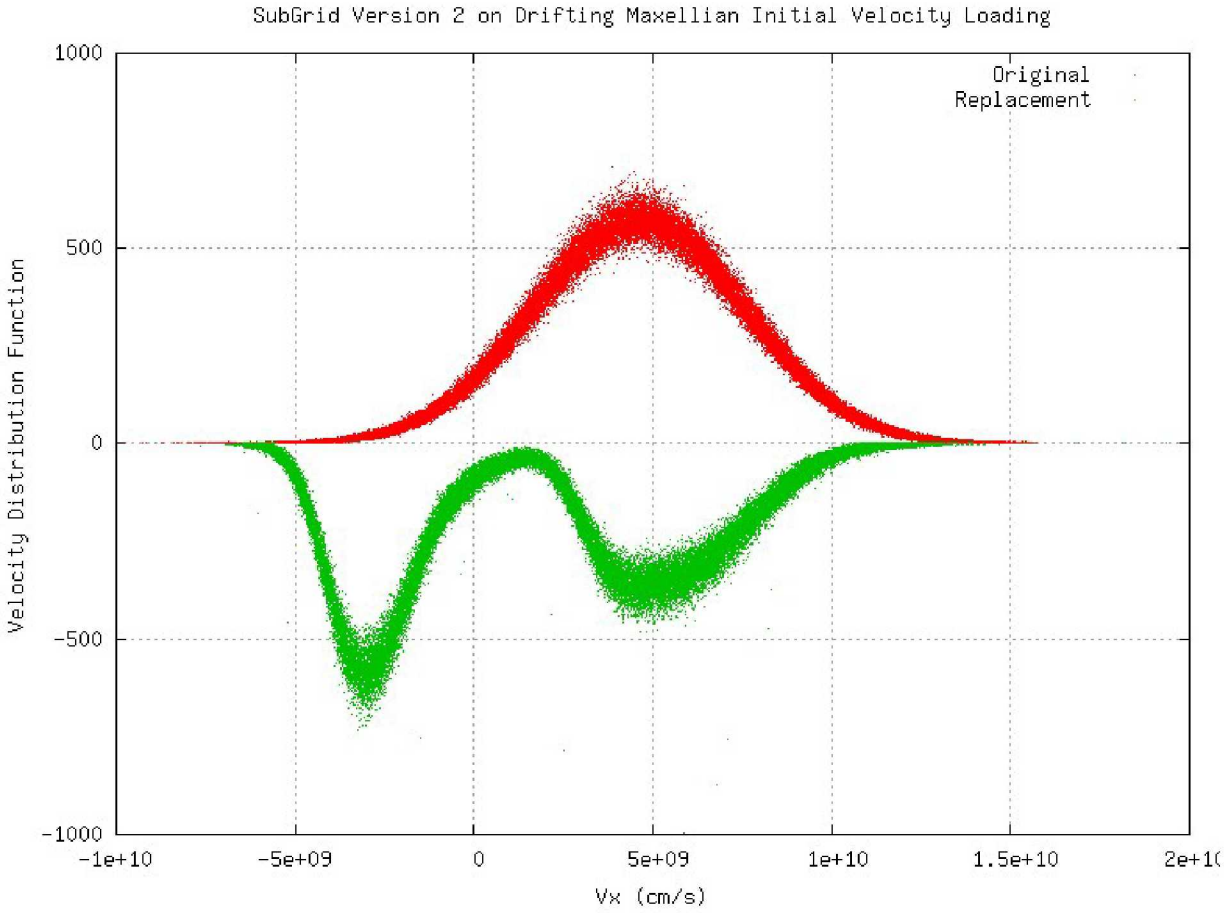


Figure 54: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellianinitial loading by the subgrid v2 method

G. Sandia Merge Method

1. Introduction

The roots of this algorithm are from a paper by Dale Welsh, but with several modifications by Tim Pointon to improve the chances of a successful merge for any given cell at any given time step. Much of this document is based on a set of Pointon's hand-written notes, with the rest filled in based on an analysis of the implemented algorithms in Quicksilver and Emphasis.

2. Description

To use the particle merging algorithm in either Quicksilver or Emphasis, the user provides a list of species that will be subject to merging, as well as a time step interval ($k_{\Delta t}$) upon which the merge will be performed. Typically, the species specified will be a list of the electron-neutral ionization products, i.e., secondary electrons and ions, that are created by the kinetic gas model. In particular, primary electrons should typically not be merged. On selected time steps, and for each selected species, each cell in the simulation is checked to see if the cell's particle

count N_c for that species exceeds a user-specified threshold N_T. If that threshold is exceeded, the merge algorithm will be activated, with the goal of creating M_{targ} particles to replace the original N_c particles in the cell. Here, M_{targ} is a user-specified target for the number of particles in the cell after the successful completion of the merge operation. It turns out that there are many problems that can occur during this process which will cause the merge operation to fail, and many of the fine details of the algorithm are a consequence of trying to reduce the number of such failures.

This algorithm attempts to reduce the number of particles of a given species (N_c) present in a single cell by replacing them with a completely new set of M particles, where $M < N_c$. This is done is such a way that the nodal charge due to the particles is preserved; consequently, the continuity equation will be satisfied (assuming a charge-conserving current-allocation algorithm, like Villasenor-Buneman, is being used). The algorithm is also designed to preserve the mean momentum of the particles in the cell, which will force the edge-allocated currents to be continuous in time before and after the merge operation. Finally, the total energy of all particles within the cell is conserved, and to the extent possible, the algorithm attempts to preserve the electron energy distribution (EEDF) within the cell.

Notationally, we define the merge operation as the following state transformation:

$$\left\{q_i, \mathbf{x}_i, \mathbf{u}_i, i \in \left[1, N_c\right]\right\} \rightarrow \left\{q'_j, \mathbf{x}'_j, \mathbf{u}'_j, j \in \left[1, M\right]\right\}, \quad \backslash^* \text{ MERGEFORMAT (9)}$$

where q_i and $\mathbf{x}_i$ are the charge and position of the i^{th} particle, respectively, and $\mathbf{u}_i = \gamma \mathbf{v}_i$ is the mass-normalized momentum of the i^{th} particle.

One of the primary reasons for a merge failure turns out to be caused by attempting to merge particles that are significantly different than the bulk of the particles in the cell. In particular, particles whose weight (charge) is much larger than the average weight of particles in the cell, or particles whose thermal velocity greatly exceeds the mean thermal velocity within the cell, can cause these problems. In order to reduce such failures, particles with these undesirable properties are identified and are then removed from the pool of particles to be merged. Such particles will instead be saved, and ultimately be added directly to the set of final particles resulting from the merge of the remaining particles in the pool.

First, the algorithm rejects any particles for which $q_i > f_{reject}^{qtot} Q_{tot}$, where Q_{tot} is the total charge of all particles within the cell and f_{reject}^{qtot} is a user-specified parameter. After these particles have been removed from the input merge pool, the mean thermal velocity (actually mean thermal mass-normalized momentum) is computed, i.e.,

$$\left\langle u_{therm} \right\rangle^2 = \frac{\sum_{i=1}^{N} u_i^2}{Q_{tot}} - \left\langle \mathbf{u} \right\rangle \mathbf{g} \left\langle \mathbf{u} \right\rangle, \text{ where } \left\langle \mathbf{u} \right\rangle = \frac{\sum_{i=1}^{N} \mathbf{u}_i}{Q_{tot}} \quad \backslash^* \text{ MERGEFORMAT (10)}$$

and N is the number of particles remaining in the input merge pool. Any particle for which $u_i - \left\langle \mathbf{u} \right\rangle \mathbf{g} \left\langle \mathbf{u} \right\rangle > f_{reject}^{utherm} \left\langle u_{therm} \right\rangle$ will be removed from the pool, where f_{reject}^{utherm} is specified by the user. Since removal of any such particles will somewhat reduce the value of $\left\langle u_{therm} \right\rangle$, the algorithm will repeat this process up to n_{reject}^{utherm} iterations (user-specified) as needed.

After all such particles have been removed, the merge algorithm will be applied to the remaining N particles if $N \geq N_{T2}$, where $N_{T2} < N_T$ is user-specified. If not, the merge will not be attempted.

a) Charge Conservation

For a charge-conserving current-allocation scheme, we need to preserve the charge allocated to each of the nodes to which any particle in the cell will allocate its charge. Using n to denote the number of these nodes, we have $n = 8$ for a 1st-order algorithm; $n = 27$ for a 2nd-order algorithm. For an unstructured mesh of 1st-order tetrahedral elements, $n = 4$. If we define the weight of the i^{th} particle to the k^{th} node as $W_k(\mathbf{x}_i)$, then the total charge due to the original particle set at the k^{th} node is given by

$$Q_k = \sum_{i=1}^{N} q_i W_k(\mathbf{x}_i) \qquad \text{* MERGEFORMAT (11)}$$

To conserve charge at each node, we then require

$$\sum_{i=1}^{M} q_i' W_k(\mathbf{x}_i') = Q_k \text{ for } k \in [1, n]. \quad \text{* MERGEFORMAT (12)}$$

To facilitate satisfying * MERGEFORMAT (12), we specify a particle's charge in terms a of a function of its position, i.e., $q_i' = G(\mathbf{x}_i)$, where

$$G(\mathbf{x}) = \sum_{j=1}^{n} g_j W_j(\mathbf{x}), \qquad \text{* MERGEFORMAT (13)}$$

where n is the number of nodes. Note that if we choose the particle locations $\mathbf{x}_i'$ well, we expect that $g_j : q_{ave}'$. For the moment, we will defer further discussion of the choice of $\mathbf{x}_i'$. Substituting * MERGEFORMAT (13) into * MERGEFORMAT (12), we obtain

$$\sum_{j=1}^{n} g_j \left[\sum_{i=1}^{M} W_j(\mathbf{x}_i') W_k(\mathbf{x}_i') \right] = Q_k \text{ for } k \in [1, n], \qquad \text{* MERGEFORMAT (14)}$$

a simple linear system of order n to compute the g_j's. Once this is solved, we obtain the charge for each new particle via

$$q_i' = \sum_{j=1}^{n} g_j W_j(\mathbf{x}_i'). \qquad \text{* MERGEFORMAT (15)}$$

Note that although * MERGEFORMAT (15) provides a formal solution for the charge of the new particles, it does not guarantee that these charges have reasonable values, or even all have the correct sign. If any of the charges do in fact have the wrong sign, the particle merge will be abandoned for this timestep. In fact, if

$$\frac{q'_{\min}}{Q_{tot}/M} < \alpha^{q}_{\min},$$

where $\alpha^{q}_{\min}$ is user-specified, the merge is considered to have failed.

To reduce the number of merge failures due to this problem, it makes sense to replace the original set of particles with a new set that closely matches the spatial distribution of the original set. To do this, a sub-grid is defined within the cell, and then we compute the total charge residing in each sub-grid for the original distribution. In Quicksilver, which has rectilinear cells, the sub-grid is constructed as a $m_x \times m_y \times m_z$ uniform lattice within the cell. The values of each particle's three 1D nodal weighting functions are used to determine in which sub-grid cell it resides. For Emphasis, which has unstructured tetrahedral cells, an analogous procedure is used to divide the cell into m^3 tetrahedral sub-grid cells. Similarly, the four nodal basis functions for each particle's position can be used to determine in which sub-grid cell the particle resides. We define $N_S \equiv m_x m_y m_z$ (m^3 for a tetrahedral cell) to be the number of cells in the sub-grid, and Q^S_l to be the total charge residing in the ℓ^{th} sub-grid cell. Additionally, while computing Q^S_l, the center of mass of all the particles within the sub-grid cell can also be computed, which will be used later when choosing the location of new particles with in each sub-grid cell.
If an algorithmic parameter is introduced that reflects a desired range of particle charges, i.e.,

$$R_q = \frac{q'_{p,max}}{q'_{p,min}}, \qquad \text{* MERGEFORMAT (16)}$$

then Q_{tot} ; $\frac{M_{t\arg}}{2}\left(q_{p,min} + q_{p,max}\right)$. This leads to

$$q_{p,min} = \frac{2Q_{tot}}{M_{t\arg}\left(1+R_q\right)} \quad \text{and} \quad q_{p,max} = \frac{2R_q Q_{tot}}{M_{t\arg}\left(1+R_q\right)}. \qquad \text{* MERGEFORMAT (17)}$$

Particles are now created within each sub-grid cell, using the following prescription for the number placed in the ℓ^{th} sub-grid cell:

$$N^S_l = \begin{cases} 0, & Q^S_l < q_{p,min} \\ 1, & q_{p,min} \le Q^S_l \le q_{p,max} \\ 1 + \text{int}\left(Q^S_l / q_{p,max}\right), & Q^S_l > q_{p,max} \end{cases} . \text{* MERGEFORMAT (18)}$$

Note that this approach does not guarantee that $M \equiv \sum_{l=1}^{N_S} N^S_l = M_{t\arg}$. The worst-case scenario is that in which every sub-grid cell but one has charge $q_{p,min}$, and the final cell has the remaining charge. In this case, it's easy to show that

$$M_{max} = \text{int}\left[\frac{\frac{M}{2}\left(1+R_q\right)-N_S+1}{R_q}\right]+N_S\,. \quad \text{* MERGEFORMAT (19)}$$

For example, if $M_{targ} = 40$, $N_S = 27$ (a 3x3x3 sub-grid), and $R_q = 3$, $M_{max} = 45$. As long as $N_S < M$, there is still a big reduction in the particle count per cell. The minimum number of particles created occurs when every sub-grid cell but one has charge infinitesimally less than $q_{p,min}$, and the final cell has the remaining charge. In which case

$$M_{min} = \text{int}\left[\frac{M\left(1+R_q\right)}{2R_q}\right]+1\,. \quad \text{* MERGEFORMAT (20)}$$

For the same example ($M_{targ} = 40$, $R_q = 3$), $M_{min} = 27$.

For the ℓ^{th} sub-grid cell, if N_ℓ^S is non-zero, the positions of the new particles within the sub-grid cell must be determined. if N_ℓ^S is one, the new particle is simply placed at the center-of-mass computed earlier. Otherwise the each of the new particles is randomly placed in a parallelepiped centered at the center-of-mass and whose edge lengths are determined by a user-provided fraction (f_{sg}, where $0 \le f_{sg} \le 1$) of the sub-grid cell size in each dimension, with the caveat that the parallelepiped is truncated so as to not extend into adjacent sub-grid cells.* For tetrahedral cells, a similar approach is used, except that the supplied fraction is relative to the minimum height of the tetrahedral sub-element.

b) Mean Momentum

In order to preserve the mean momentum of the original particles in a single cell, we will use the same weighting to the nodes that we used to conserve charge in * MERGEFORMAT (11). Since the momentum of the i^{th} particle is $m_i\mathbf{u}_i$, and $m_i = R_{me}q_i$, where is the mass-to-charge ratio for the particle species being merged, it is sufficient to preserve the product of the particles' charge and mass-normalized momentum, i.e.,

$$\left(Q\mathbf{u}\right)_k = \sum_{i=1}^{N} q_i\mathbf{u}_i W_k\left(\mathbf{x}_i\right). \quad \text{* MERGEFORMAT (21)}$$

If we define each new particle's mean mass-normalized momentum† $\langle\mathbf{u}_i\rangle$ as a function of its position, i.e.,

$$\langle\mathbf{u}_i\rangle \equiv \mathbf{H}(\mathbf{x}_i) = \sum_{j=1}^{n}\mathbf{h}_j W_j\left(\mathbf{x}_i\right), \quad \text{* MERGEFORMAT (22)}$$

---

* Actually, the Quicksilver algorithm computes the three 1D nodal weighting coefficients, e.g., for the x coordinate, $\omega_x^{+}(k) = \left(x_i' - X_k^{fg}\right)\Big/\Delta X_k$. Only at the end of the merge are these values converted to the components of $\mathbf{x}_i'$.

† Throughout the remainder of this document, we will refer to the mass-normalized momentum simply as the momentum, unless explicitly stated otherwise.

we obtain the following vector matrix equation:

$$\sum_{j=1}^{n}\mathbf{h}_j\left[\sum_{i=1}^{M}q_i'W_j\left(\mathbf{x}_i'\right)W_k\left(\mathbf{x}_i'\right)\right]=\left(Q\mathbf{u}\right)_k \text{ for } k\in\left[1,n\right], \qquad \text{* MERGEFORMAT (23)}$$

which can be solved for the values $\mathbf{h}_j$. Finally, each particle's mean momentum can be determined using * MERGEFORMAT (22).

c) Thermal Momentum and Energy

At this point, we have only determined the mean momentum component, $\langle\mathbf{u}_i\rangle$, of the new particles. Now the thermal energy of the initial set of particles in the cell must be accounted for. Toward that end, we express the momentum of the i^{th} particle as

$$\mathbf{u}_i'=\langle\mathbf{u}'\rangle+\beta\mathbf{u}_i^{ran}, \qquad \text{* MERGEFORMAT (24)}$$

Where

$$\mathbf{u}_i^{ran}=\mathbf{u}_i^{sam}+\sum_{j=1}^{n}\mathbf{b}_jW\left(\mathbf{x}_i'\right). \qquad \text{* MERGEFORMAT (25)}$$

In * MERGEFORMAT (25), $\mathbf{u}_i^{sam}$ is a thermal momentum that will be sampled from the thermal distribution of the initial set of particle within the cell. The second term in * MERGEFORMAT (25) is added to conserve momentum. Finally, the factor β will be adjusted to conserve energy.

For the moment, we will defer the discussion of how the sampled thermal distribution $\mathbf{u}_i^{sam}$ is obtained other than to say that it is chosen with the goal that the distribution resulting from all members of the set of new particles best matches the salient features of the distribution obtained from the initial set of particles. Of course, since the reduction in the number of particles between the initial and final sets results in a loss of information, one cannot expect to exactly reproduce this distribution.

Since the set of $\langle\mathbf{u}_i\rangle$ already preserves the momentum aggregated to the n nodes of the cell, $\mathbf{u}_i'$ will also preserve nodal momentum if we now require that $\mathbf{u}_i^{ran}$ when aggregated to the nodes vanishes, i.e.,

$$\sum_{i=1}^{M}q_i'W_k\left(\mathbf{x}_i'\right)\mathbf{p}_i^{ran}=0 \text{ for } k\in\left[1,n\right]; \qquad \text{* MERGEFORMAT (26)}$$

consequently,

$$\sum_{j=1}^{n}\mathbf{b}_j\left[\sum_{i=1}^{M}q_i'W_j\left(\mathbf{x}_i'\right)W_k\left(\mathbf{x}_i'\right)\right]=-\sum_{i=1}^{M}q_i'W_k\left(\mathbf{x}_i'\right)\mathbf{p}_i^{sam} \text{ for } k\in\left[1,n\right]. \qquad \text{* MERGEFORMAT (27)}$$

Note that this is the same matrix as found in * MERGEFORMAT (23). Solving * MERGEFORMAT (27) for the values of $\mathbf{b}_j$, * MERGEFORMAT (25) can be used to find $\mathbf{u}_i^{ran}$.

Finally, energy must be conserved. Non-relativistically‡, $E_i = m_i u_i^2 = R_{me} q_i u_i^2$, so we have

$$\sum_{i=1}^{N} q_i u_i^2 = \sum_{i=1}^{M} q' u_i'^2 = \sum_{i=1}^{M} q_i' \langle u' \rangle_i^2 + 2\beta \sum_{i=1}^{M} q_i' \langle \mathbf{u}_i' \rangle \mathbf{g}_i^{ran} + \beta^2 \sum_{i=1}^{M} q_i' \left(u_i^{ran} \right)^2 . \qquad \text{* MERGEFORMAT (28)}$$

Defining

$$a = \sum_{i=1}^{M} q_i' \left(u_i^{ran} \right)^2$$

$$b = \sum_{i=1}^{M} q_i' \langle \mathbf{u}_i' \rangle \mathbf{g}_i^{ran} \qquad \text{* MERGEFORMAT (29)}$$

$$c = \sum_{i=1}^{N} q_i' u_i^2 - \sum_{i=1}^{M} q_i' \langle u' \rangle_i^2 \ ,$$

* MERGEFORMAT (28) becomes

$$a\beta^2 + 2b\beta - c = 0 . \qquad \text{* MERGEFORMAT (30)}$$

where $a > 0$ and $c \geq 0$. Thus, β has two real roots which, unless $c = 0$, have opposite sign. We will choose the root nearest unity.

d) Generation of the Sampled Thermal Momentum Distribution

We would like, at least to the extent possible, have the set of new replacement particles match the thermal distribution of the initial set of particles. To construct the thermal momentum distribution for the initial set of particles, we need to extract the mean drift momentum from each particle, i.e.,

$$\mathbf{u}_i^{therm} = \mathbf{u}_i - \langle \mathbf{u} \rangle \qquad \text{* MERGEFORMAT (31)}$$

Where

$$\langle \mathbf{u} \rangle = \frac{1}{Q_{tot}} \sum_{i=0}^{N} q_i \mathbf{u}_i \text{ and } Q_{tot} = \sum_{i=0}^{N} q_i . \qquad \text{* MERGEFORMAT (32)}$$

Using the standard notational convention that $v = |\mathbf{v}|$, we can find a minimum and maximum value of u_i^{therm} for the initial particle set, i.e., $u_{\min}^{therm}$ and $u_{\max}^{therm}$. We will construct a logarithmically-uniform distribution as a histogram by dividing $\log u^{therm}$ into N_b uniform bins in such a way that $\log u_{\min}^{therm}$ and $\log u_{\max}^{therm}$ are at the centers of the first and last bins respectively. In each bin, we then accumulate the charge of all the initial particles whose values of u^{therm} fall within the range of the bin. Finally,

‡ Since the merger will presumably only used ionization products (secondary electrons and ions), this is a reasonable approximation.

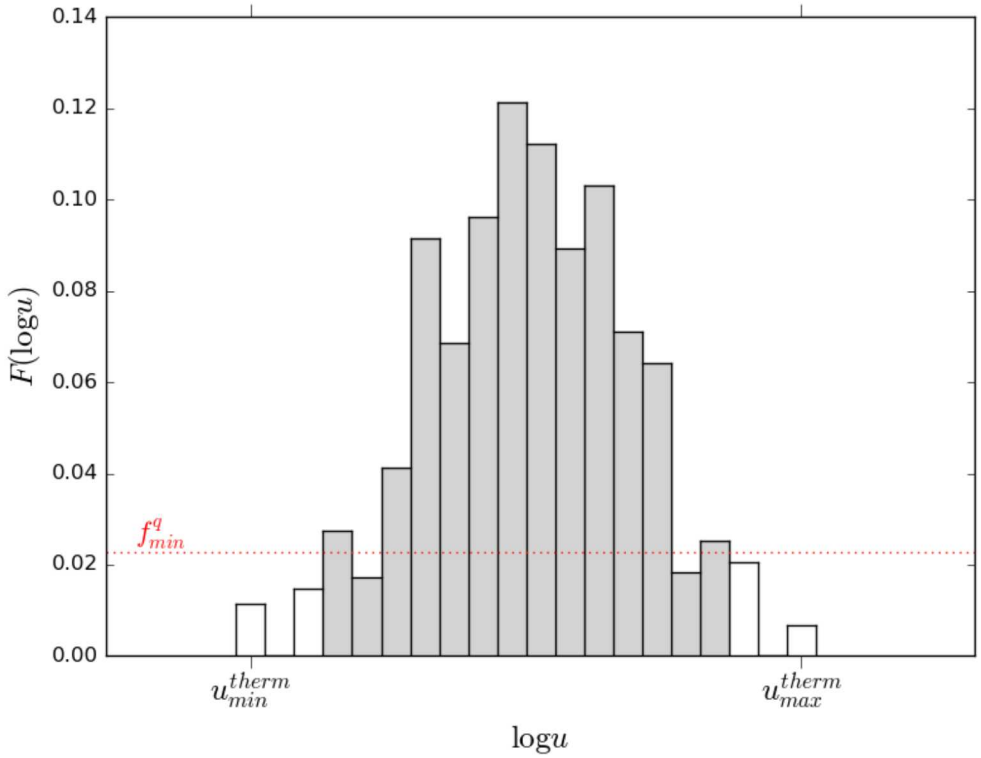


Figure 55: Example thermal momentum distribution, showing its core region in gray.

the accumulated charge in each bin is divided by Q_{tot}, resulting in a normalized distribution function $F(\log u)$. The histogram can be described by $\{F_k\}$, for $k \in [1,N_b]$. To avoid creating new particles in bins with very small values of F_k, F_k is set to zero for any bin in which $F_k < F_{min}$, where $F_{min} << 1$ is a tuning parameter for the algorithm. If necessary, $\{F_k\}$ is then renormalized so that $\sum F_k = 1$. Figure 1 shows an example of such a distribution.

The goal now is to determine $\mathbf{u}_i^{sam}$ for the final set of M particles in such a way as to preserve this distribution function F. The algorithm attempts to anticipate the shape of the distribution, assuming that it will be smallest at its lower and upper ends, with larger values in the midrange of the distribution.

By construction, the total charge in this set of particles in Q_{tot}, so we can define

$$f_i^q = q_i' / Q_{tot} \text{ for } i \in [1, M] \qquad \backslash^* \text{ MERGEFORMAT (33)}$$

to be the charge fraction contributed by each new particle. We also assume that $\{f_i^q\}$ has been sorted[§] in ascending order, i.e., from lightweight to heavyweight. Consequently, the minimum weight of any particle in the set is given by $f^q_{\min} = f_1^q$.

[§] The codes do not actually sort the particles, but instead do an "indirect" sort, which provides an indirection array which allows ordered access to the unsorted set of particles.

In order to ensure that bins in the two tails of the distribution with $F_k < f_{\min}^q$ will be represented, the algorithm searches from the two ends of the distribution until it finds the first bin for which $F_k \geq f_{\min}^q$, which yields the range of the distribution's "core", $\{k_{core}\} \in [k_l, k_u]$. The example shown in Figure 1 shows this core region of the distribution shaded in gray, with the corresponding value of $f_{\min}^q$ shown as a dashed red line. For each bin k that is in one of the two non-core tails of the distribution (shown as unshaded bins in Figure 1), we will randomly select one of the heavier-weight particles and split it into two identical particles, except for their charge. One of the two resulting particles is designated as "additional" and its charge fraction is set to F_k. The original "donor" particle's charge is correspondingly reduced so that sum of the charge fractions for the two particles equals that of the original donor particle before being split. Also, the magnitude of the sampled thermal momentum (u_i^{sam}) for the additional particle is randomly selected to be somewhere the range of the k^{th} bin. The original donor particle's sampled thermal momentum is deferred at this point and will be set later in the process, contributing to the core of the distribution. The algorithm will only allow a donor particle to be split once, and currently has two parameters that impact this process: 1) the fraction of a donor particle's charge that can be given to an additional particle is limited to a maximum value $f_{\max}^{split}$, and 2), the number of particles that can be split is limited to a maximum number M_{max}^{extra}. Note that the first restriction allows for the possibility, albeit remote, that more than one particle may need to be split to satisfy the needs of a non-core bin, and the algorithm needs to be able to handle this situation. Finally, if M_{max}^{extra} is exceeded, the merge operation is deemed to have failed for this cell and time step.

Now we will use the M original particles (albeit some have reduced charge due to splitting) to populate the remaining core of the distribution. At this point a cumulative distribution function $S(\log u)$ is formed by first renormalizing the remaining core of the distribution such that

$$\sum_{k=k_l}^{k_u} F_k = 1 \qquad \text{* MERGEFORMAT (34)}$$

and then integrating the core distribution over $\lambda = \log u$, yielding

$$S(\lambda_k) = \sum_{i=k_l}^{k-1} F_i \text{ for } i \in [k_l, k_u + 1], \text{ where } \lambda_k = \log u_{\min}^{therm} + \left(k - \tfrac{3}{2}\right)\Delta\lambda, \qquad \text{* MERGEFORMAT (35)}$$

where

$$\Delta\lambda = \frac{\log u_{\max}^{therm} - \log u_{\min}^{therm}}{N_b - 1}. \qquad \text{* MERGEFORMAT (36)}$$

This defines a piecewise-linear function of λ. Figure 56 shows a plot of this function, based on the example of Figure 55.

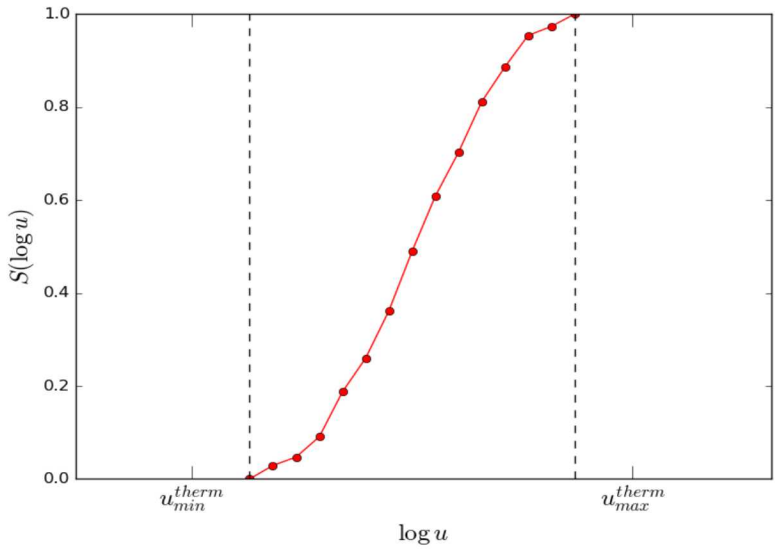


Figure 56: Cumulative distribution function obtained by integrating the example distribution in Figure 55. Dashed vertical lines indicate the bounds of the distribution's core.

A variation of the standard inverse cumulative distribution approach will be used to choose u_i^{sam} for each particle to populate the core of the distribution. Recognizing that the core distribution is still likely to be smaller at the limits of its momentum range, the algorithm attempts to preferentially use the lighter-weight particles to populate core distribution's "wings", leaving the heavier particles to populate its central region. To do this, the set of new particles is divided into two groups, one of which, containing the M_{wing} lightest-weight particles, will be used to populate the upper and lower "wings" of the distribution. Here, M_{wing} = int($f_{wing}M$), where f_{wing} is an algorithm parameter. The remaining particles will be used to populate the remaining central portion of the distribution. The particles are reordered** into a list using the following prescription:
the M_{wing} lightweight particles are successively assigned, with 50-50 probability, to the either the beginning or the end of the list, progressively working in from either end, then the M - M_{wing} remaining heavy particles are randomly assigned to the remaining unfilled central portion of the list. Using this reordering, we define

$$\sigma_k = \begin{cases} 0\,, & k = 0 \\ \sum_{j=1}^{k} f_j^q\,, & k \in [1, M] \end{cases}, \quad \text{* MERGEFORMAT (37)}$$

and then determining u_j^{sam} for each particle using

** The algorithm actually uses an indirection array to implement this reordering.

$$s_j^{part} = \sigma_{j-1} + R\left(\sigma_j - \sigma_{j-1}\right) \qquad \text{* MERGEFORMAT (38)}$$

where R is a random number between zero and one. Then $\lambda_j \equiv \log\ u_i^{sam}$ is the value of λ for which $s_j^{part} = S(\lambda)$ and consequently, $u_j^{sam} = \exp\lambda_j$.

At this point, we have a value of u_i^{sam} for our M original new particles, plus M_{extra} additional particles split off from one or more of the original particles. Assuming this sampled thermal momentum is isotropic, each $\mathbf{u}_i^{sam}$ is determined using

$$\begin{aligned}
\mathbf{u}_j^{sam} \cdot \hat{\mathbf{x}} &= u_j^{sam} \sin\theta\cos\phi \\
\mathbf{u}_j^{sam} \cdot \hat{\mathbf{y}} &= u_j^{sam} \sin\theta\sin\phi \quad , \qquad \text{* MERGEFORMAT (39)} \\
\mathbf{u}_j^{sam} \cdot \hat{\mathbf{z}} &= u_j^{sam} \cos\theta
\end{aligned}$$

where $\cos\theta = 1 - 2R_1$, $\phi = 2\pi R_2$, and R_1 and R_2 are random numbers between zero and one.

e) Algorithm Failure Mechanisms

As seen throughout the algorithm, there are several problems that can arise during the merging process. This section lists all such problems that will cause the algorithm to fail.

- If due to the exclusion of outlying particles, the number of particles available to be merged (N) falls below N_{T2}.
  If the nodal charge weights are not adequately preserved, i.e.,

$$\sum_{j=1}^{n}\left|\sum_{i=1}^{N} q_i W_j\left(\mathbf{x}_i\right) - \sum_{i=1}^{M} q_i' W_k\left(\mathbf{x}_i'\right)\right| \geq \varepsilon_{\max}^{q} Q_{tot}$$

- If the charge computed for any of the replacement particles has the wrong sign, or it is too close to zero, i.e., $\dfrac{q'_{\min}}{Q_{tot}/M} < \alpha_{\min}^{q}$.
- If the drift kinetic energy of the new particles exceeds the total kinetic energy of the old particles, i.e.,

$$\sum_{i=1}^{M} q_i' \left\langle \mathbf{u}_i' \right\rangle \cdot \left\langle \mathbf{u}_i' \right\rangle > \left(1 + \varepsilon_{\max}^{ke}\right) \sum_{i=1}^{N} u_i^2 \quad .$$

- If the particle-splitting process needs to split more than M_{max}^{extra} heavyweight particles to fill the non-core bins of the thermal distribution.

- If both a and b in * MERGEFORMAT (29) vanish.

- If the total energy of any particle in the set of new particles is significantly larger than the maximum energy any particle in the original set of particles, i.e., $u'^2_{\max} > \alpha^{ke}_{\max} u^2_{\max}$, where here the "*max*" subscripts denotes the maximum value over the new $\{u'_i\}$ and original $\{u_i\}$ sets of particles, respectively.

f) Algorithm Parameters

This section provides a table of all of the parameters used by the algorithm. Most can be set by the user, but a few are presently hardwired into the two codes. A non-empty entry in the "Value" column indicates that the parameter is set to a fixed value and cannot be selected though user input.

Table 13. Merge Algorithm Parameters

Parameter	Description	Type	Range	Value
$k_{\Delta t}$	Timestep interval for applying merge algorithm	Integer	$k_{\Delta t} > 0$	
N_T	Threshold particle count to trigger particle merge	Integer	$0 < M_{targ} < N_{T2} < N_T$	
N_{T2}	Min. particle count to proceed w/ merge after rejection tests	Integer	$0 < M_{targ} < N_{T2} < N_T$	
M_{targ}	Target particle count after merge	Integer	$0 < M_{targ} < N_{T2} < N_T$	
f^{qtot}_{reject}	Fraction of Q_{tot} above which particle will be rejected	Real	$0 < f^{qtot}_{reject} = 1$	
f^{utherm}_{reject}	Multiple of mean thermal velocity above which particle will be rejected	Real	$f^{utherm}_{reject} > 1$	
n^{utherm}_{reject}	Max. interation count for rejecting velocity outliers	Integer	$n^{utherm}_{reject} > 0$	
$\alpha^{q}_{\min}$	Min. new particle charge relative to mean particle charge	Real	$\alpha^{q}_{\min} < 1$	
m_x, m_y, m_z or m	Sets the sub-grid lattice within the cell	Integer	all ≥ 1	
R_q	Nominal max. ratio of particle charges after merge	Real	$R_q \geq 1$	
f_{sg}	Fraction of sub-grid cell in which to create replacement particles around the center-of-mass of initial particles in sub-grid cell	Real	$0 \leq R_q \leq 1$	
N_b	Number of uniform log u bins used for thermal momentum distribution	Integer	$N_b > 1$	20
$F_{\min}$	Min. distribution value (F_k) below which F_k will be set to zero.	Real	$0 < F_{\min} = 1$	10^{-3}
$f^{split}_{\max}$	Max. fraction of a heavyweight donor particle that will be split off to form an additional particle	Real	$0 < f^{split}_{\max} < 1$	0.8

M_{max}^{extra}	Max. number of particles that will be added by splitting heavyweight particles	Integer	$M_{max}^{extra} > 0$	10
f_{wing}	Fraction of new particles that are used to populate the wings of the thermal distribution	Real	$0 < f_{sg} < 1$	0.5
ε_{max}^{q}	Max. relative error allowed in preserving nodal charge weights	Real	$\varepsilon_{max}^{q} = 1$	
ε_{max}^{ke}	Max. fraction the drift kinetic energy of the new particles can exceed the total kinetic energy of the old particles	Real	$\varepsilon_{max}^{ke} = 1$	10^{-5}
α_{min}^{ke}	Max. multiplicative factor that the kinetic energy of any new particle can exceed the max. kinetic energy of any original particle.	Real	$\alpha_{min}^{ke} > 1$	

3. Examples

Execution times and means of relative errors the comparison of new to old metrics are shown in Table 14 and Table 15, respectively. Comparisons of new to original velocity distribution functions and the frequency distributions of relative error on the various metrics are shown in Figure 57 through Figure 64. Note that the nodal Q error is not included in these tables nor figures as it is conserved by construction thereby causing the relative error between new and original to essentially be zero.

Table 14: : Execution time for the SubGrid M:2 method for 1000 repetitions starting with a set containing 10^4 particles

Initial Loading Method	Time Required (s)
Random V	17.65
Drift	13.00
Exponential	17.23
Drifting Maxwellian	17.33

Table 15: Mean values of the relative errors in the various metrics for the four different initial velocity loadings by the Sandia merge method

Initial Loading	ΔQ_0	Ionization Rate	Kinetic Energy	Nodal Q	$J_x(1,1)$
Random-V	0.8722	0.01008	3.619e-05	small	*
Drift	0.7095	8.635e-04	4.1612e-07	"	0.01608
Exponential	0.6321	0.01339	3.120e-04	"	0.01417
Drift Max	0.05358	8.922e-03	6.6645e-04	"	0.009367:

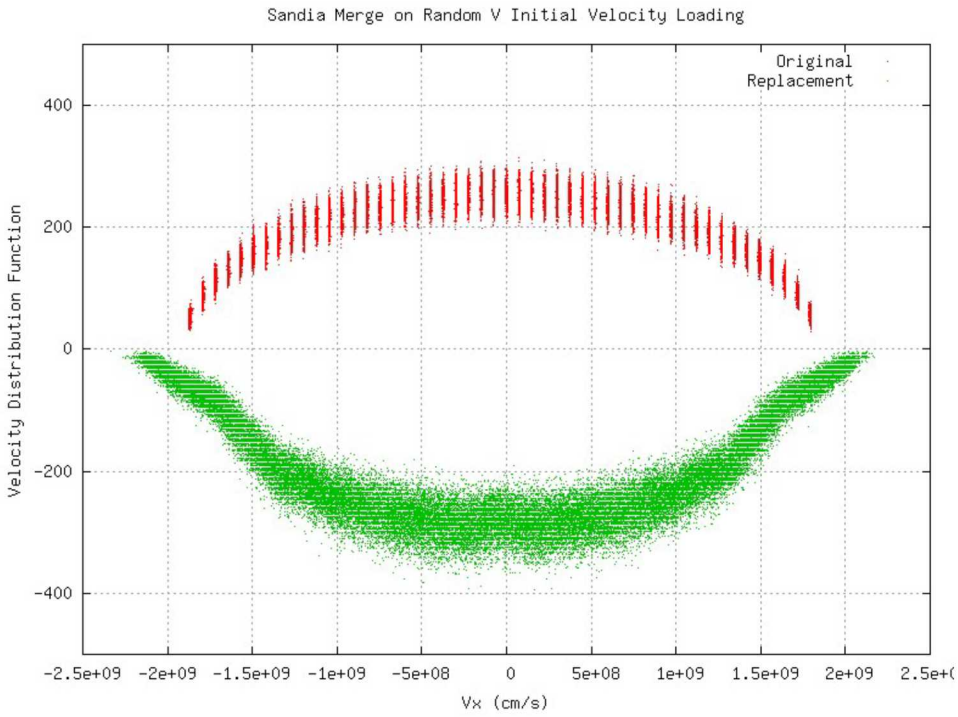


Figure 57: Comparison of original and replacment x-component velocity distribution function for the random V initial loading by the Sandia merge method

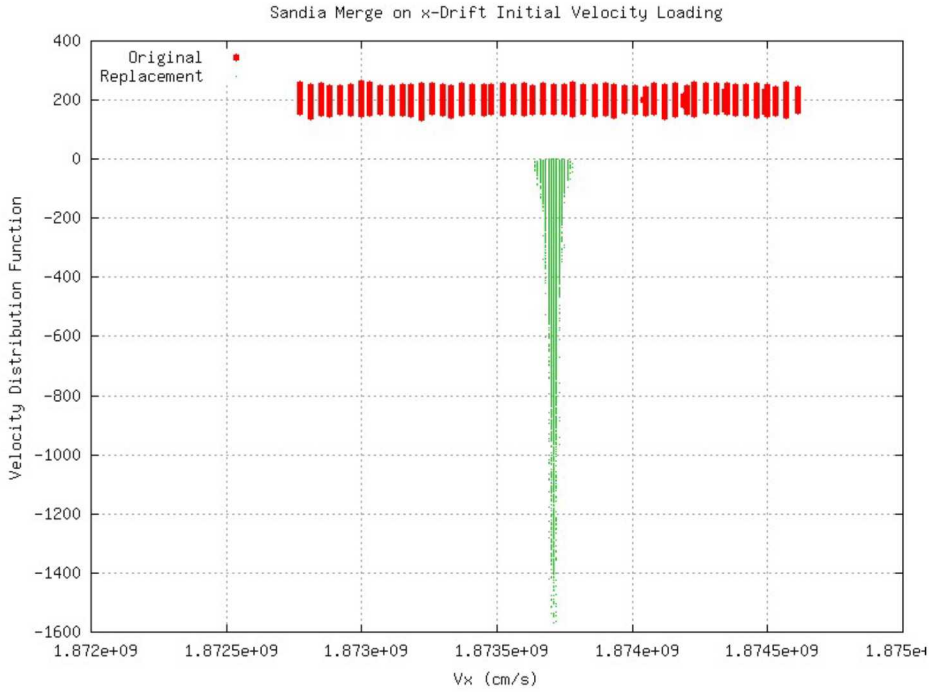


Figure 58: Comparison of original and replacment x-component velocity distribution function for the x-drift initial loading by the Sandia merge method

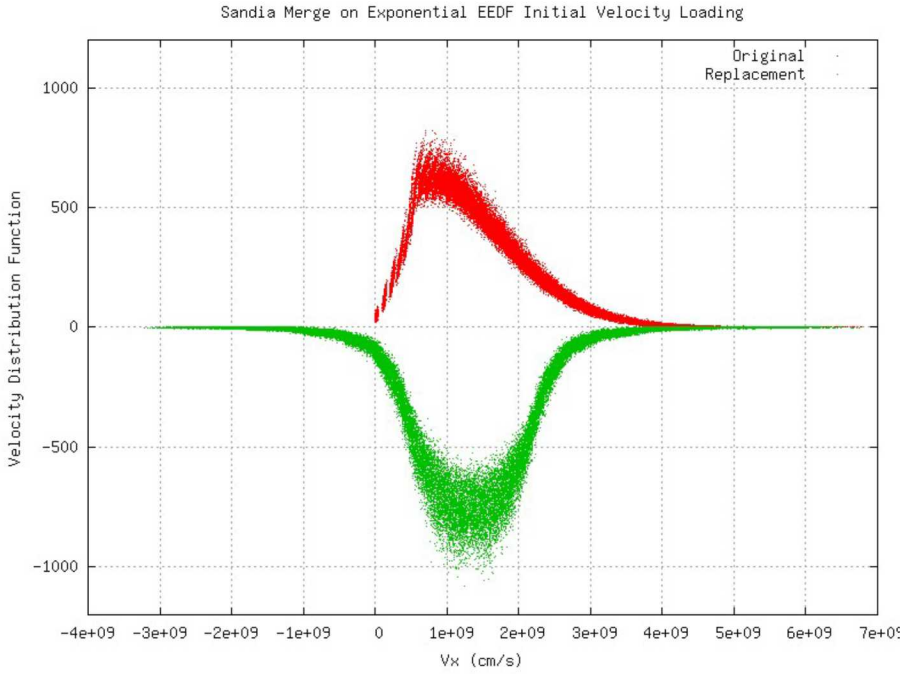


Figure 59: Comparison of original and replacment x-component velocity distribution function for the exponential EEDF initial loading by the Sandia merge method

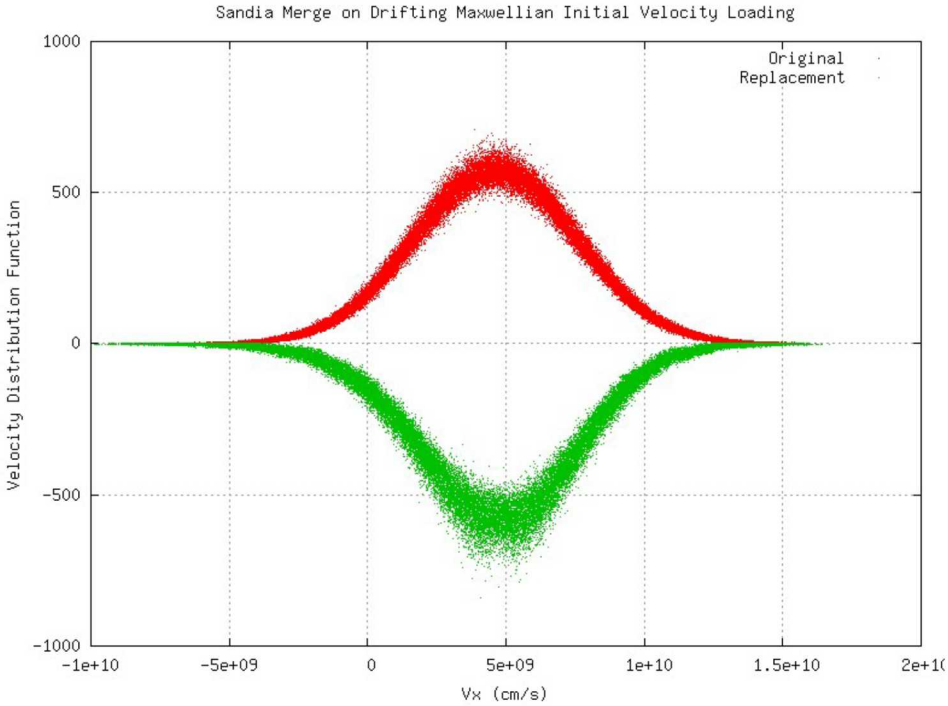


Figure 60: Comparison of original and replacment x-component velocity distribution function for the drifting Maxwellian initial loading by the Sandia merge method

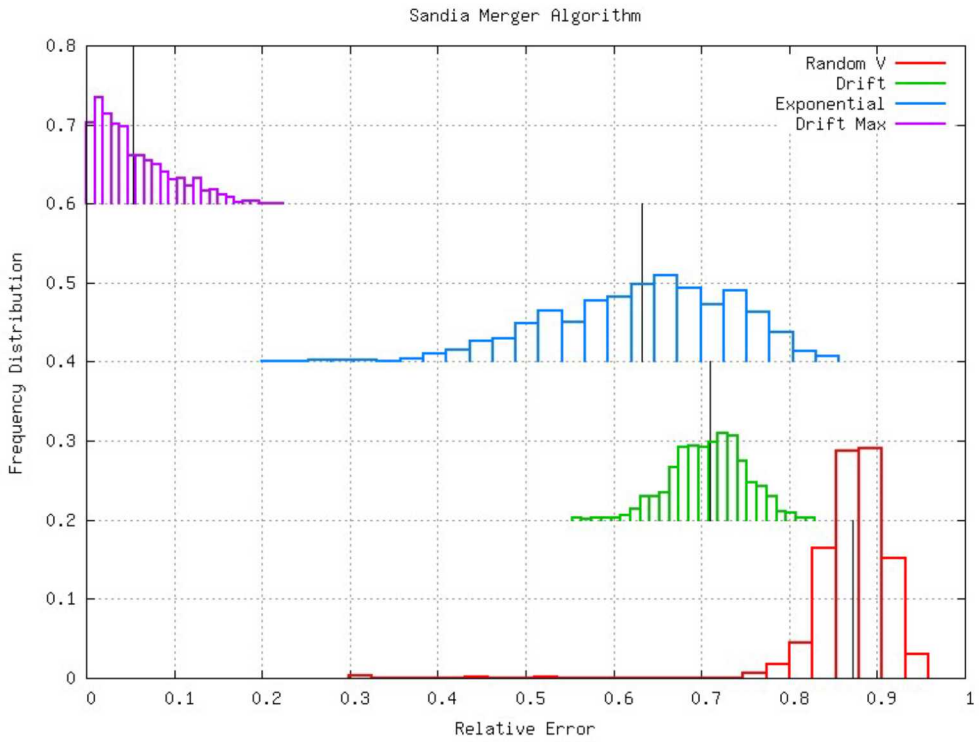


Figure 61: Relative error in the ΔQ_0 metric for the various initial loadings produced by the Sandia merge method

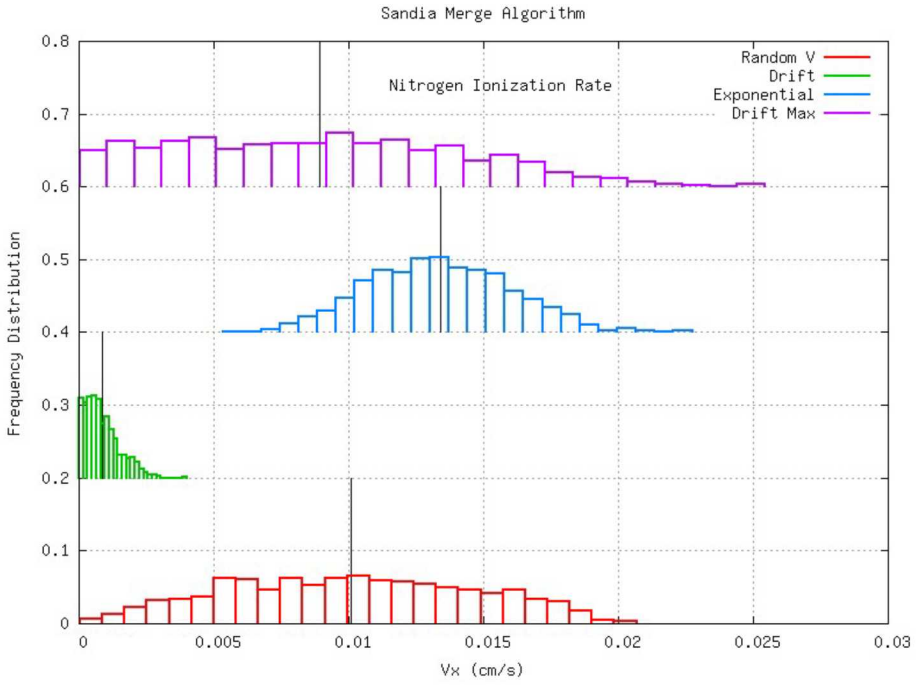


Figure 62: Relative error in the nitrogen ionization rate metric for the various initial loadings produced by the Sandia merge method

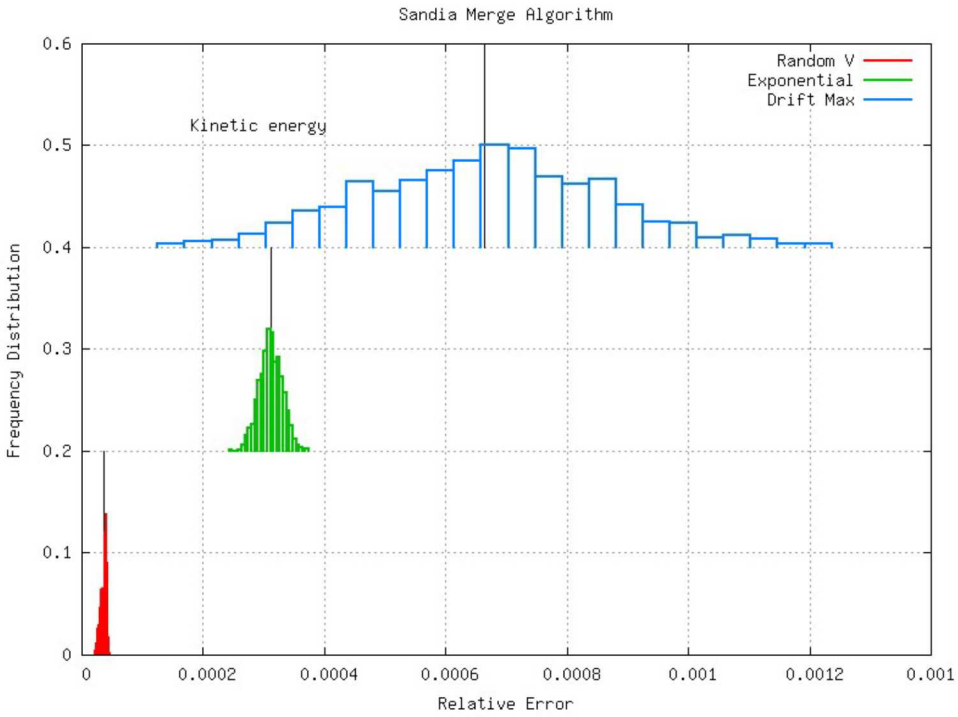


Figure 63: Relative error in the kinetic metric for the various initial loadings produced by the Sandia merge method

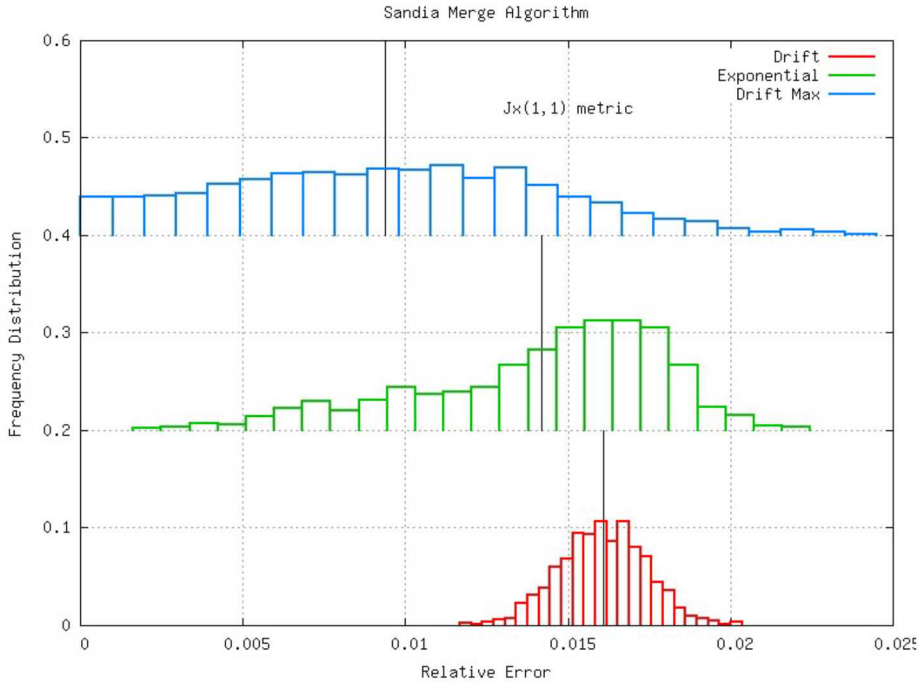


Figure 64: Relative error in the $J_x(1,1)$ metric for the various initial loadings produced by the Sandia merge method

4. Summary

The results of the analysis of the Sandia merge algorithm and comparison to the results from the other merge algorithms show it to be as good or better than the other merge algorithms with the exception of how well it rebuilds the original velocity distribution function. Figure 58, in particular, is odd. It could be that our implementation of the method in BVcomp has done some harm to the algorithm in terms of the user-adjustable parameters that could improve relevant performance.

In consultation with Sandia personnel, it was speculated that the Sandia merge algorithm's constraints on the preservation of mean momentum might be overly stringent, and it would be interesting to relax somewhat them with the hope that this would allow higher fidelity preservation of the thermal velocity distribution of the particles in the cell. The idea that emerged from this discussion was to attempt to preserve the mean drift velocity in the cell rather than the momentum weighted to the cell's nodes. There was some discussion on whether mean drift velocity for the particles in the merged set should be equal to the mean drift velocity of all the particles in the original unmerged set ($<\mathbf{u}>$), or whether it should be determined by a more spatially-localized method. Since the algorithm already divides the cell into multiple sub-grid cells, it was relatively easy to add tje computation of the mean velocity for particles in the original set specific to each sub-grid cell ($<\mathbf{u}_\ell>$ for the ℓ^{th} sub-grid cell), and then set the mean velocity of each new merged particle to the value computed for the sub-grid cell into which it was inserted.
In order to explore this idea, a new modified algorithm was implemented. It started with the original algorithm, but with the following specific changes:

- The constraints involving mean velocity were removed.
- They were replaced by the constraint that the mean velocity of each merged particle located in the ℓ^{th} sub-grid cell would be set to $f<\mathbf{u}_\ell> + (1-f)<\mathbf{u}>$, where $0 \leq f \leq 1$ is a user-provided parameter. This "smoothing" factor should help mitigate issues with sub-grids with very few particles; additionally, by just changing f, it's easy to look at using just the "whole-cell" mean velocity, just the sub-grid mean velocities, or any combination thereof.
- Discard the sum term from , so Eq 25. that $\mathbf{u}^{ran}$ is simply $\mathbf{u}^{sam}$.

Initial testing of this modified algorithm has been disappointing, and we do not see the hoped-for improvements in preserving the thermal velocity distribution. Interestingly, the resulting currents look quite reasonable, and the error in the node-weighted momentum conservation turns out to be quite small. However, the fit to the remaining thermal velocity distribution is quite poor. The problem is that to conserve energy, the multiplicative β factor for the thermal component of the momentum which one would like to be near one, is instead ~0.2-0.3. This was true for a variety of values of the newly-introduced f parameter. The effect of this small value for β is to reduce the range of the thermal velocity distribution by this factor. For the same test cases, but using the original algorithm, β is typically ~1. Although there may be promise in the general idea of sacrificing preservation of momentum to obtain better preservation of the thermal velocity distribution, it does not appear that this particular approach is the way to accomplish it.

Exposure to and experience in implementing and using the conservation of nodal Q method has shown that it is an easy algorithm to implement. There would seem to be no excuse not to implement it in any algorithm in which the positions of the replacement particles can be assigned independently of the assignment of velocities. The impact of use of the nodal Q conservation method in the other merge algorithms, particularly the ICEPIC REDUCE algorithm, would be interesting to study.

The notion of producing an algorithm that identically produces the 24 separate components of the current density that the Buneman-Villasenor algorithm produces in two dimensions was considered here, as well. But unlike the nodal Q conservation algorithm, which uses the linear weighting functions to allocate particle weights to nodal Qs, the analogous set of "basis functions" which a J-conservation algorithm would have to use, would be considerably more complex. The nodal Q basis functions depend only upon the particles' initial positions while the J basis functions would depend on both intial and final particle positions. This would thus be a nonlinear problem with as many unknowns as the number of replacement particles – a problem that might be approachable with an iterative method. The complexity of such a method did not tend to invite further investigation within the scope of this effort.

Finally, while the path-class idea had at least conceptual promise, no algorithm investigated within the scope of this effort has been able to make good on that promise. This is not proof that it does not warrant further investigation, nor is it proof that it does not have some merit, yet the question as to its applicability to codes that do not rely on Cartesian conformal gridding, which provides a foundation for the notion of the path class, must be considered. Perhaps the basic concept is readily extended to arbitrary meshes. This might warrant further investigation.

IV. STATISTICAL ANALYSIS OF MERGING PARTICLES

A. Comparisons of Velocity Distribution

In the interest of better understanding the impact of various particle merging algorithms, Confluent Sciences has undertaken research on the how to best characterize distributions of particles with a single number. This topic has also occupied other communities looking at questions of kinetic versus fluid simulation techniques, so much of the research looks at figures of merit for assessing how Maxwellian a distribution function may or may not be. This is clearly not the only distribution function that would be helpful for SGEMP calculations, where beam-like distributions are also of interest, but given the fact that many merger algorithms have the property that repeated application of the algorithm have the effect of thermalizing a given distribution function, in essence acting as a kind of pseudo-collision that removes the extremum of the distribution function, this class of figure of merit offers value to our SGEMP problem.

To understand this figure of merit, and following the work of Staton *at al.* (Stanton, 2018), assume a near equilibrium, normalized distribution function for each velocity component that is Maxwellian

$$f(v) = \sqrt{\frac{m}{2\pi T}} exp\left(-\frac{m}{2T}\bar{v}^{\,2} \right); \bar{v} = v - \langle v \rangle \qquad (1)$$

Here the <v> denotes the moment of the velocity distribution function. With this definition of a "centered" Maxwellian, the associated moments have the following form:

$$\langle \bar{v}^{\,n} \rangle = \begin{cases} 0 & n\ odd \\ \left(\frac{T}{m}\right)^{\frac{n}{2}} (n-1)!! & n\ even \end{cases} \qquad (2)$$

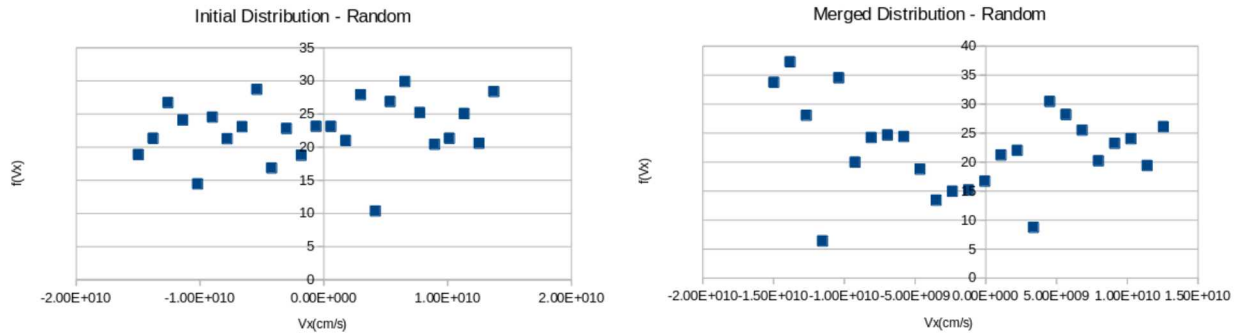
The main insight here is that the moments of a Maxwellian are connected and specific ratios of powers are constant. The "!!" denotes a double factorial (e.g. a!!=a(a-2)(a-4)…). This lets us define a metric, originally used to determine how "kinetic" a distribution function is, such as:

$$G_k \equiv \frac{\langle \bar{v}^{\,2k} \rangle}{(2k-1)!! \langle \bar{v}^{\,2} \rangle^{\,k}} \qquad (3)$$

If the distribution is near equilibrium, G_k should approach one for any integer k, and non-unity implies deviations from a Maxwellian. Calculated on a cell-by-cell basis in PIC, this offers the means to assess "kinetic-ness" as a function of space and time. Additionally, by choosing k, one can emphasize different parts of the distribution function. For example, higher powers emphasize the tail of the distribution. There does appear to be a practical issue that these higher order moments are reliant on kinetic information encoded in the calculation. It is well-known, however, that PIC, while preserving the volume of phase space, can have challenges with conserving both energy and moment to machine precision. The impact of these errors on the diagnostic are an area for future study, and for the time being we well only deal with k=1 and k=2.

To test this new figure of merit, Confluent Sciences used its new Buneman-Villasenor Comparison tool to generate initial distributions of particles and then merge these particles using the REDUCE algorithm (originally from ICEPIC, and implemented inside BVcomp). In the cases that follow, the initial distribution will have 1000 particles and the merged distribution will have 500 particles (2:1 reduction). We will then plot f(v) vs v for each case of an initially random distribution function, initially beam-like distribution function, and finally, an initially exponential distribution function while computing G_k for both the initial and merged distributions. Note that because none of these distribution are truly Maxwellian in nature, we do not expect G_k to be unity.

Figure 65 shows the distribution function f(Vx) versus Vx for the three distribution functions. The initial distribution function is shown on the left while the merged distribution function is given on the right for comparison:



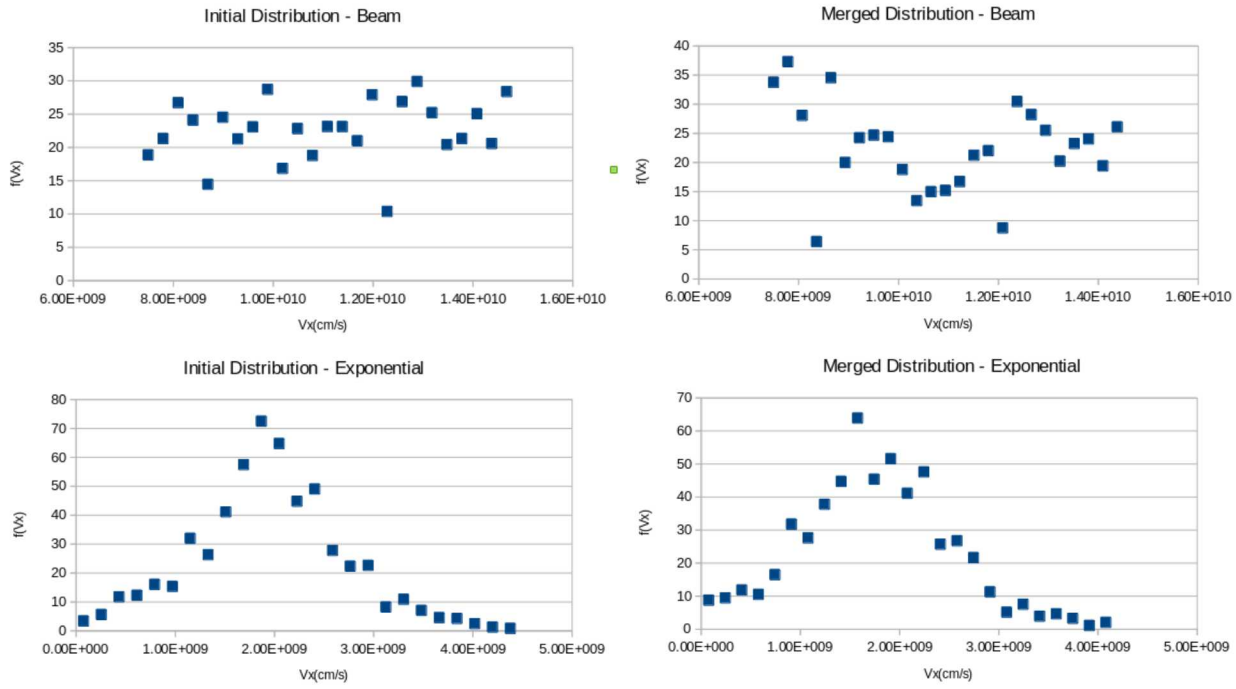


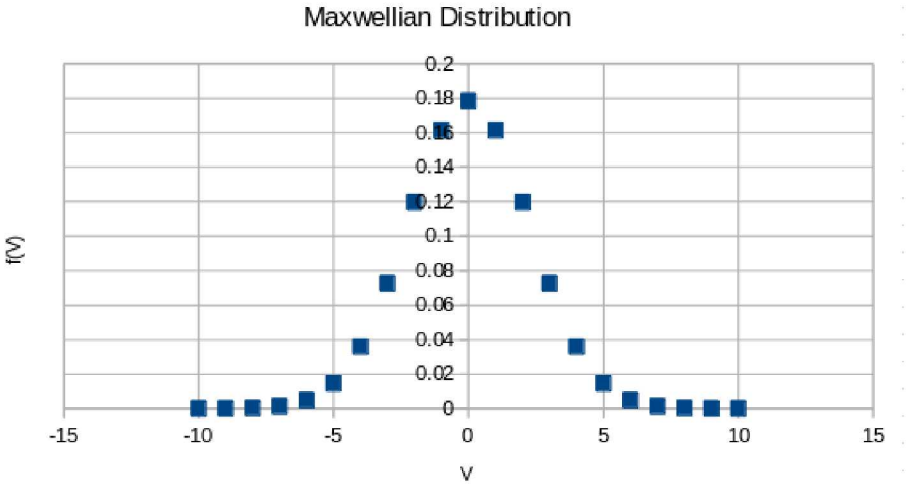
Figure 65: Plots showing the distribution function f(Vx) vs Vx for three classes of initial distribution functions. The plots on the left show the initial distribution while the plots on the right show the impact of the REDUCE algorithm for merging particles. There are 1000 particles on the left and 500 on the right.

For each of the cases shown in Figure 65, G_k has been calculated. These results are shown in Figure 66 There are several interesting features to note: First, all three distribution functions, upon merging, show movement in G_k away from unity, indicating a slight, but consistent characterization as less Maxwellian than the initial distribution functions. The source of this "anti-thermalization" is under study, and it is not clear if it is a consistent feature of the REDUCE algorithm in general. One can see in Figure 65, however, a systemic impact of the merger on the negative or low velocity "side" (the left hand side of the figure) of all three distributions. Second, the cycling of k=1 to k=2 clearly shows that both the random distribution (centered around zero) and the beam-like distribution (random, but centered with a bulk "left-to-right" velocity) are far from Maxwellian. This allows for an assessment of the nature of particle distributions in a PIC calculation. Third, the exponential algorithm is surprising close to unity at k=2. This agreement drops suddenly, however, at k=3, where G_3~0.001. If this is due to the fact that the higher moment is detecting the fact that the exponential isn't precisely Maxwellian, or if there are challenges with the higher order moments when one has sparse data, will be a question under study as this effort moves forward.

Table 16: G_k for three distribution functions.

Distribution	k	G_k(initial)	G_k(merged)
Random	1, 2	1.00, 0.60	1.00, 0.57
Beam-like	1, 2	1.00, 0.59	1.00, 0.55
Exponential	1, 2	1.00, 1.00	1.00, 0.93

For completeness, we also tested the G_k diagnostic against a pure Maxwellian distribution. This data was synthesized directly from a Maxwellian distribution function, including varying the mass and temperature of the Maxwellian. In all cases, G_k was very close to one (see Figure 66). An additional task will be to vary the Maxwellian data, adding skew and kurtosis to better understand how the G_k figure of merit behaves as one moves away from a pure Maxwellian.



k	G_k
1	1
2	0.999
3	0.997
4	0.991
5	0.977

Figure 66: Maxwellian test data and the performance of Gk.

For our SGEMP problem involving situations where beams have formed, and where the particles have not yet thermalized, our results show that this diagnostic still offers the means to characterize a distribution of particles by a single (perhaps single set) of numbers, which would probably not be unity, but would be a robust indication of the nature of the distribution function. For merging/splitting, the main focus would be that the algorithm does not change the value of G_k. Additionally, if we noted that merging pushed this figure of merit toward unity, that would indicate a thermalizing aspect to a given merging algorithms, allowing more subtle characterization of the performance of the method. Similarly, merging algorithms that preferentially emphasized beaming particles would be distinguishable by a trend away from unity for given distribution function. Clearly, experimentation will be necessary to make this a useful diagnostic in practice. It may also be necessary to re-work this diagnostic to include the bulk motion <v> that was removed in equation (1) to best capture the physics of SGEMP. One particularly interesting option comes from noting that for a cold beam, characterized by a delta function is velocity space, the moments of the distribution function are such that $\langle v^{2k} \rangle \sim \langle v^{2} \rangle^{k}$. Thus, G_k simply becomes "1/(2k-1)!!" thereby allowing ratio of G_k at different values of k to

offer a signature of cold beams. In principle, this allows the new diagnostic to allow identification of different classes of distribution (Maxwellian vs cold beam) that are critical to SGEMP physics. In the context of merging algorithms, this might allow one to use different merging algorithms adaptively depending on the physical situation in a given portion of a simulation.

B. Statistical Limits on Merging

Confluent Sciences also looked at the question of what are the statistical limits on merging particles. Because of the extensive literature on Maxwellian distribution, and its link to the Normal distribution, we have focused exclusively on this class of distribution function. It is well-known that the Maxwellian distribution is common in plasma physics, as it both represents a lowest energy state solution to the kinetic Vlasov equation, and is often the basis for deriving fluid equations based on taking moments of the distribution. In equation form, we start with

$$f(v) = \sqrt{\frac{m}{2\pi T}} exp\left(-\frac{m}{2T}\bar{v}^2 \right); \bar{v} = v - \langle v \rangle \quad (1)$$

where equation 1 is a centered Maxwellian, with the average drift <v> removed. Recall that f(v) is dimensionless, so we have some freedom in the units of m, T, and v. Here we use a normalized set of units to focus on the statistical issues of the function. The moments of this distribution function are often quantities that we wish to conserve in our simulations. The first three moments are (for a 1-D case to ease reading; the moments are generally vector or tensor quantities)

$$\rho := \int_v f(v)dv$$

Mass or Charge Density (2)

$$M := \int_v vf(v)dv$$

Momentum Density (3)

$$E := \int_v v^2 f(v)dv$$

Energy Density (4)

We can also define a general concept for statistical quantities, where if we want to know the average of some quantity X, we can write

$$\langle X \rangle = \frac{\int xf(v)dv}{\rho} \quad (5).$$

In (2)-(5), we assume in this report that the density is normalized to one, rather than the more typical link to density. These are standard formulations in plasma physics for both fluid and kinetic equations. For merging particles in a kinetic simulation, these are the fundamental definitions necessary for the analysis, as the numbers of particles can be thought of as discrete statistical measurements of the continuous phasespace represented by the distribution function f(v).

One advantage of starting our analysis with Maxwellian distributions is the relationship to the Normal, or Gaussian, distribution from statistics:

$$f(x) = \sqrt{\frac{1}{2\pi\sigma^2}} exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{6}$$

Where μ and σ are the mean and standard deviation of the Normal distribution function. In general, this function is written as $N(\mu, \sigma^2)$, where the standard deviation squared is called the variance. By inspection, it is clear that the Maxwellian distribution and the Normal distribution are the same once the variance is linked to the ratio of the temperature to the mass in the Maxwellian.

For the purposes of analysis, this direct link between the Maxwellian and the Normal distribution is helpful, in that there are a number of tools from the statistics community that can be brought to bear as we move from continuous distribution functions to discrete numbers of particles representing the distribution function. For example, it is a standard technique to define moments of a statistical distribution function in complete analogy with our physical moments equations (2)-(4). In general, the n^{th} moment is defined as

$$\int_{-\infty}^{\infty} (x-c)^n f(x) dx \tag{7}$$

By convention, the zeroth moment is normalized to one in statistics, the first moment or expected value retains the translation term "c" and the higher order moments are typically found around zero translation. This is often called a central moment, as we did this in defining equation 1 with <v>. Note that this can be used for any distribution function f(x). The first moment is the mean and the second moment is the variance in general, and for the Normal distribution (6) it is easy to show that mean is μ and the variance is σ^2.

Because of its role in many statistical methods, the Normal distribution's behavior under sampling has been extensively studied. If we have a set of discrete measurements, say the velocity of a particle in a given cell, it is easy to compute the mean and the variance of the N "measurements" or instances of the particle. In the standard statistical parlance:

$$\bar{v} = \frac{\sum v_i}{N}; S^2 = \frac{\sum (v_i - \bar{v})^2}{N-1} \tag{8}.$$

If we believe that our sampling has come from a Normal (Maxwellian) distribution, these measured quantities can be used to immediately define our distribution function $N(\bar{v}, S^2)$. The challenge is when we have finite numbers of particles, we know that there is an error associated with our measurement that is of the order of $\sqrt{n}$. For a Normal distribution, this error can be made explicit. If "t" is defined as

$$t = \frac{\bar{x} - \mu}{\sigma_{\bar{x}}} \tag{9}$$

which is the difference between the true mean and the measurement mean, normalized by the standard deviation of the measurement (or S in equation 8), it can be shown that t follows the following distribution function based on the number of measurements N at hand:

$$f(t) = \frac{\Gamma(\frac{N+1}{2})}{\Gamma(\frac{N}{2})\sqrt{\pi N}}\left[1 + \frac{t^2}{N}\right]^{-\frac{N+1}{2}} \quad (10).$$

This is called "Student's t-distribution" function, due to the fact that the researcher published under the pen name "Student." (Historical sidenote: Student is William Gosset, and he worked for Guinness Brewing. He was not allowed to publish under his real time for fear of identifying important corporate information). The t-distribution arises when we have a normally distributed population where the sample size is small. It is typically used in statistics when the variance of the distribution is unknown, and allows estimates of if the sample size N is sufficient to have statistical confidence in a result. However, we can repurpose the t-distribution to help us analyze the impact of merging particles. The t-distribution allows us to see the statistical implications of representing a distribution with only a few particles.

To this end, let us see what the t-distribution looks like when compared against a Normal (Maxwellian) distribution. As shown in Figure 67, the t-distribution has higher tails and a lower peak, but with the same mean value. It is the expected distribution if one sampled a Maxwellian with only 8 independent uniform measurement (N=8). This particular example is from Harvard's on-line statistic textbook, and therefore also serves as test of Confluent Science's implementation.

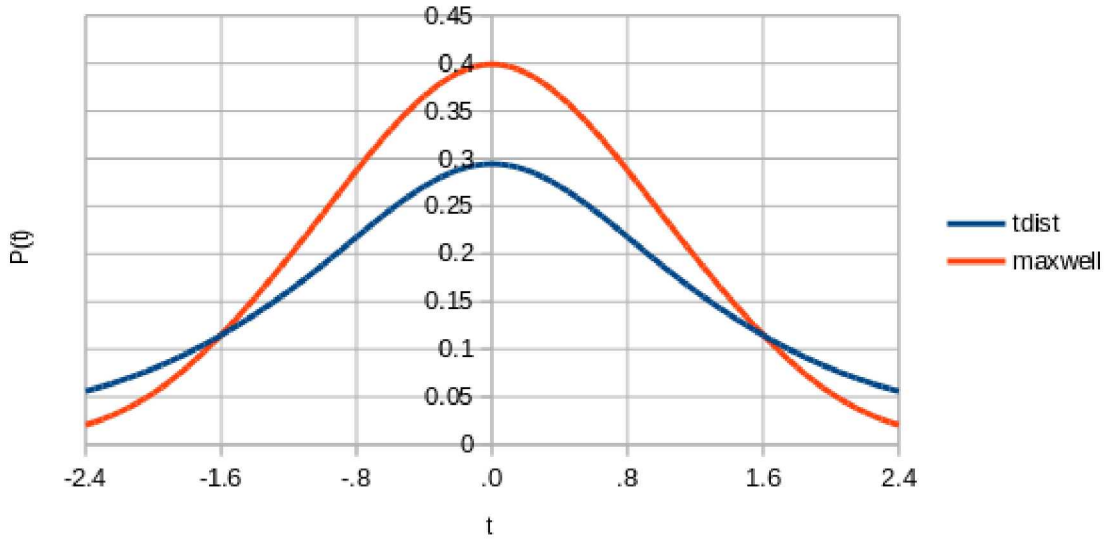


Figure 67: Comparison of Maxwellian distribution and a t-distribution with the same mean and variance, but only N=8 samples.

It can be formally shown as N goes to infinity, the t-distribution evolves to the Normal distribution. In general, N>30 makes the t-distribution identical to the Normal.

1. Application to a PIC model

Assume that we have a cell that has a finite, but large number of particles, and we wish to understand the impact of reducing the number of particles. This section can be thought of analyzing the Confluent Sciences BVComp model in light of the statistical issues outlined above. We are using equations (1)-(10), and we assume that the underlying physics has produced a Maxwellian distribution in one cell with normalized dimensions for mass, velocity and energy, such that equation 1 is consistent. For our Maxwellian, we will use m=1, T=5, and <v>=0 in equation 1. Likewise, our Normal distribution will have $\mu = 0$ and $\sigma^2 = 5$. We will then change the number of particles N in the cell and, by virtue of the t-distribution, we will estimate the impact on the mean and variance of the particles in the cell. Figure 68 shows the impact of going from N=250 to N=10 (or equivalently, a merger of 250:10). If we then measure the mean and variance that results from sampling particles pulled from a Maxwellian distribution, we see that the N=250 case, the mean is zero and the temperature is 5.04, in good agreement with our parent Maxwellian distribution with a mean of zero (to machine precision) and a temperature of 5. For the N=10 case, however, we see that while we get the same mean of nearly zero (small~10^{-6}, but nowhere near machine precision), the temperature has increased to 6.13, or an error of roughly 22%. This increase in variance simply from statistical variation is exactly what the t-distribution is designed to capture. The mean (1st statistical moment) and the variance (2nd statistical moment) of the normalized t-distribution is

$$Mean = \mu; \sigma_t^2 = \frac{N}{N-2} \qquad (11).$$

Therefore, the increase in the measured temperature is roughly (10/8). The increase in the merged distribution's temperature is simply a consequence of poor sampling. This suggests that unless more care is taken in the sampling, it is difficult to simultaneously sample in such way to retain both momentum and energy (first moment and second moment of the physical distribution function) in the merging process.

The situation, in practice, gets slightly worse with drifting beams. If we set up a Maxwellian with m=1, <v>=2, and T=5, and draw particles from this distribution, we see for the N=250 case a mean drift of 1.9998 and a temperature of 5.04. When we merge to N=10, however, the drift is 1.9810 and the temperature has again increased to 6.13. It is not clear why our drifting case seems to perform so much worse than the centered case.

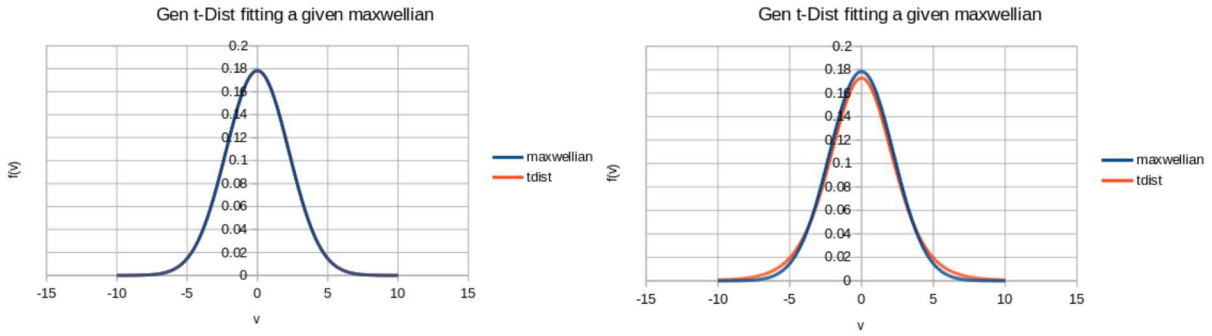


Figure 68: The left figure compares a Maxwellian and a t-distribution for N=250. The right figure has the same Maxwellian compared to a t-distribution of N=10. This is equivalent to a particle merge of 250:10.

These errors, while small, again reiterate the challenge with getting both the mean and the variance of a distribution correct with small number of samples. Furthermore, this drives home an additional problem that comes for normalizing a distribution in the process of merging. When we performed the 250:10 merge, we naturally also renormalized the zeroth moment (charge and/or mass - see equation 2) to account for the change in total number of particles so that we hold the total mass and charge the same. While the charge to mass ratio was maintained in this normalization, as is typically in PIC for macroparticles, this resulted in the charge on an individual particle increasing by a factor of 25. This had the odd effect of amplifying the error present in the current density as the roughly 1% error in drift velocity of the merged particles is multiplied by the charge increase per particle. Confluent Sciences will continue to work to understand if this interplay between the zeroth order and first order scaling results in the lack of current continuity in the merging process. It might be possible, in an electromagnetic PIC, to look at the error in the velocity (roughly 1% in the example here) and modify the merged particle charge distribution to compensate for the error to that the current density in continuous. An additional option might be to spend a great deal of effort creating particles to that drive the momentum error down to machine precision to minimize any error associated with the current density. This analysis drives home that current density is a correlation between charge and momentum. Future work will look to see if different kinds of statistical analysis might help clarify this physics.

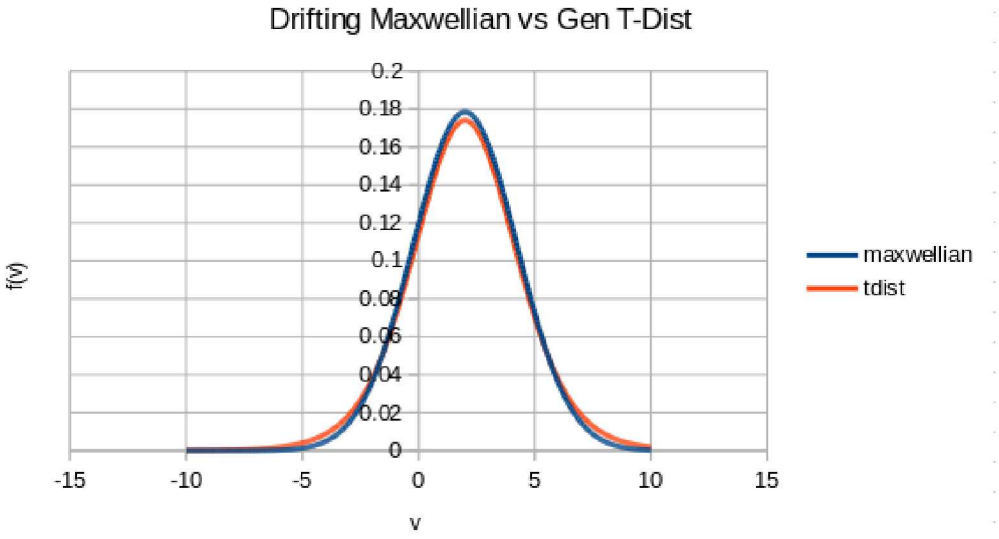


Figure 69: A drifting Maxwellian compared to an N=10 t-distribution.

2. Merger Algorithm Improvement

Consideration has also been given to a speculative way to exploit this new understanding of the t-distribution to construct new distribution functions from small numbers of particles that better mimic the parent Maxwellian distribution function without the increase in temperature that we are seeing due to the variance of small numbers. The t-distribution can be rewritten in the generalized way to allow for curve fitting to desired parameters. This is often used in both the Bayesian inference community and the machine learning community. In this formalism, equation 10 becomes

$$P(x|\nu,\mu,\sigma) = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})\sqrt{\pi\nu\sigma^2}}\left[1+\frac{1(x-\mu)^2}{\nu \quad \sigma^2}\right]^{-\frac{\nu+1}{2}} \tag{12}$$

Where P(x|ν, μ, σ) is the probability of the value x, given a sample number ν, a desired mean μ, and a desired variance σ^2. Note that in equation 12, μ and σ are desired quantities, and they should not be confused with physical quantities that we might have had in the previous section. Equation 12 allows the potential to develop small samples which meet the desired mean velocity requirements (important for conserving momentum or the first moment) and energy requirements (important for conserving the second moment).
Initial tests are promising. Using the same initial Maxwellian with no drift and a temperature T=5 as described above, we then use equation 12 to get μ=<v>=0 and T=5 by varying σ for given number of particles ν. As shown in Figure 70, choosing σ for a given ν allows one to get the correct temperature.

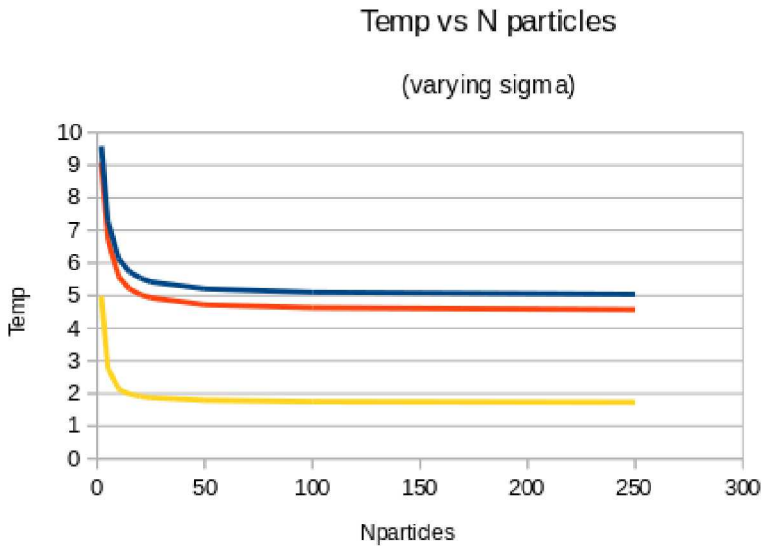


Figure 70: Using equation 12, one can vary σ(ν) to achieve a desired effective temperature with a low number of particles.

A large number of details remain to be worked out. First, for very small numbers of particles, one can see that the impact of the N/(N-2) scaling is going toward infinity – the effective temperature becomes very sensitive to model choices. Additionally, this is a kind of effective temperature that compensates for the statistical inadequacy of the population. Many kinetic methods, however, are formulated as forces laws acting on individual particles. It is not clear what impact this kind of effective temperature might have in a particle simulation that runs for many time steps with momentum and/or energy conserving algorithms also in play. At some level, this idea involves changing the actual energy of individual particles to achieve the correct property of a distribution function. If nothing else, how to implement a scheme such as the one proposed here with multiple cells, each with their own local temperature and drifts, might be challenging and might introduce errors in the current density that are unacceptable for SGEMP simulations. Confluent Sciences will continue to study these questions.

C. Physical and statistical limits of Merging Particles

In many cases that may be encountered, simulations of SGEMP will have the computational resources to have many particles even after merging, with counts significantly greater than the N>30 limit where the normal distribution significantly deviates from the t-distribution. With this in mind, Confluent Sciences decided to study the limit of large numbers of particles, where the central limit theorem applies, and better understand the limits of preserving physical moments during merging. As in the previous section, we will start with Maxwellian/Normal distributions, but we will then broaden out to study preserving other classes of statistical distribution functions to better understand the potential of merging in important physics in the SGEMP problem.

Reviewing equations (1)-(8), we are considering a discrete set of N particles. For the present analysis, we assume all of these particles are in one cell, and that we are not considering the effect of particles leaving or entering into a given cell, so that we can ignore the complications from spatial dependence. Also, we assume one-dimensional distributions here, but the analysis can be repeated for higher numbers of dimensions. With this in mind, and following equation (8), we can write

$$\bar{v} = \frac{1}{N}(v_1 + v_2 + ...v_N) = \frac{1}{N}\sum_1^N v_i;\ S^2 = \frac{1}{N-1}\sum_1^N (v_i - \bar{v})^2 \tag{13}$$

For the mean and variance of the sample, where we have use (N-1) convention where Bessel's correction is applied to the variance to find an unbiased estimator. This isn't particularly important in the large sample limit where N>>1, but it is helpful to denote the framework. While we anticipate v_i being drawn from a Maxwellian distribution, equation (13) holds for any set of particles. What changes is the physical meaning of the mean and variance if one can assign a distribution to the collection of particles.

This population of N particles can be merged into M particles (M<N) using any of the methods described in this report. We designate the mean and variance of the merged particle distribution as

$$\bar{v}_M = \frac{1}{M}(v_1 + v_2 + ...v_M) = \frac{1}{M}\sum_1^M v_i;\ S_M^2 = \frac{1}{M-1}\sum_1^M (v_i - \bar{v}_M)^2 \tag{14}$$

At this point in the analysis, we assume that particles of the parent population (N) are drawn from a Maxwellian distribution, and we further assign physical meaning to the mean and variance ala equations (1)-(6):

$$\bar{v} = \frac{1}{N}\sum_1^N v_i = \mu;\ S^2 = \frac{1}{N-1}\sum_1^N (v_i - \bar{v})^2 = \sigma^2 \equiv \frac{T}{m} \tag{15}$$

such that the mean velocity is the drift velocity of the Maxwellian and the variance is the ratio of the temperature to the particle mass. Thus, equation (13) gives the average velocity of the population and the variance is a measure of the temperature. In short, the statistical first (mean) and second (variance) moments are explicit linked to the physical average velocity (equation 3) and a measure of the energy (equation 4) parameterized by the temperature. As usual, velocity and temperature can be transformed into momentum and energy measures, and the momentum naturally leads to the current density. Although we do not comment on it here, we assume that the total charge and mass are persevered in any physical valid merge algorithm (equation 2).

This is necessary to avoid putting physical charge on the mesh of the electromagnetic algorithm. If equation 2 is not conserved, additional steps such as divergence cleaning may be necessary.
Now we can begin to assess the consequences of various features of any merge algorithm. Suppose that we force the merged set of particles (M) to have the same mean velocity as the original population. This is equivalent to conserving momentum between the parent and daughter populations and is a necessary, but not sufficient condition to conserving current density. We write this conservation as

$$\bar{v} = \frac{1}{N}\sum_{1}^{N} v_i = \bar{v}_m = \frac{1}{M}\sum_{1}^{M} v_i \tag{16}$$

Virtually all of the merger algorithms that we have investigated perform this step, and this is relatively easy to do by merging particles that are close to one another in velocity space for a given cell. Note, however, that this places constraint on the variance. Inspection of equations (13) and (14) show that these mean velocities enter into the variance. This allows us to write

$$S^2 = \frac{1}{N-1}\sum_{1}^{N}\left(v_i - \bar{v}\right)^2 = \frac{T}{m}; S_M^2 = \frac{1}{M-1}\sum_{1}^{M}\left(v_i - \bar{v}_m\right)^2 = \frac{1}{M-1}\sum_{1}^{M}\left(v_i - \bar{v}\right)^2 \tag{17}$$

In writing (17), we continue to identify the variance of the parent population (S^2) with T/m, and we also note that the variance of the daughter population (S^2_M) is constrained by the mean of the parent population (see the right most equation in (17)).

Variance can be rewritten in a number of ways. Here perform the following steps, using the dummy variables σ and x for clarity:

$$\sigma^2 = \frac{1}{N-1}\sum_{1}^{N}\left(x_i - \bar{x}\right)^2$$

$$\sigma^2 = \frac{1}{N-1}\sum_{1}^{N}\left(x_i^2 - 2\bar{x}x_i + \bar{x}^2\right)$$

But with the definition of the mean velocity (equation (16)), we can rewrite this as

$$\sigma^2 = \frac{1}{N-1}\left(\sum_{1}^{N} x_i^2 - 2N\bar{x}^2 + N\bar{x}^2\right) = \frac{1}{N-1}\left(\sum_{1}^{N} x_i^2 - N\bar{x}^2\right) \tag{18}$$

The form of (18) allows us to rewrite (17) as

$$S^2 = \frac{1}{N-1}\left(\sum_{1}^{N} v_i^2 - N\bar{v}^2\right); S_M^2 = \frac{1}{M-1}\left(\sum_{1}^{M} v_i^2 - M\bar{v}^2\right) \tag{19}$$

It is important in equation (19) to recall that the parent population (N) and the daughter population (M) need not be at all the same except for the same mean.
With equation (19), it is important to introduce Jensen's Inequality:

$$\frac{1}{N}\sum_{1}^{N} x_i^2 \geq \bar{x}^2 \equiv \left(\frac{1}{N}\sum_{1}^{N} x_i\right)^2 \tag{20}$$

This merely notes the sum of squared items is greater than or equal to the square of their sum. In addition to Jensen's Inequality, we also know that for our merger case, N>M and this allows one to write

$$N\bar{x}^2 \geq M\bar{x}^2 \tag{21}$$

Using equations (20)-(21), we can write (19) as

$$S^2 = \frac{T}{m} = \frac{1}{N-1}\left(\sum_1^N v_i^2 - N\bar{v}^2\right)$$

$$(N-1)\frac{T}{m} = \left(\sum_1^N v_i^2 - N\bar{v}^2\right)$$

$$(N-1)\frac{T}{m} \leq \left(N\bar{v}^2 - M\bar{v}^2\right)$$

$$(N-1)\frac{T}{m} \leq \left(\sum_1^M v_i^2 - M\bar{v}^2\right)$$

$$\frac{N-1\,T}{M-1m} \leq \frac{1}{M-1}\left(\sum_1^M v_i^2 - M\bar{v}^2\right)_{=S_M^2}$$

$$S_M^2 \geq \frac{N-1\,T}{M-1m} = \frac{N-1}{M-1}S^2 \qquad (22)$$

So, the daughter population (M) has a higher effective temperature if you fix the momentum between the two populations. In general, the sums present in these equations are only equalities when M=N, so that one can design an identical population with the same mean and variance only when the parent and daughter population are the same size. Additionally, when one recalls that the standard deviation scales as the square root of the variance, (S), we can see an intuitive scaling with the square root of the sampling between the two populations, such that we see the typical central limit behavior of

$$\bar{v} = \mu + \frac{\sigma}{\sqrt{N}} + \frac{S}{\sqrt{M}}. \qquad (23)$$

Alternatively, one could demand that the variance be the same between the parent population N and the daughter population M. If we look at equation (17), we would find

$$\bar{v} \neq \bar{v}_M$$

with the consequence that carful work must be done to ensure that the current density (J=Qv) is appropriate adjusted to ensure conservation of charge and accurate currents on the mesh. Yet another alternative to pick a merge scheme that does not preserve either the mean momentum or the temperature but seeks to minimize the different for both in merging the parent population to the daughter population. This last option would clearly involve iterations and might be prohibitively expensive. Additionally, the fact that neither momentum nor temperature/energy was preserved suggests additional challenges in any PIC time stepping algorithm.

The main takeaway at this point is that it seems to be very difficult, if not formally impossible, to preserve both momentum and temperature for a Maxwellian distribution if the parent and daughter population are not the same size. This is simply a consequence of building continuous functions out of a finite and discrete set of particles. This suggests that there is probably not one "silver bullet" merger algorithm, at least for situations where the Maxwellian distribution dominate.

D. Other Distributions

The rapid nature of SGEMP suggests potential for non-equilibrium physics. For these situations, it is inappropriate to use normal distribution functions. Other distribution functions are possible, especially in the range of challenges associated with the SGEMP problem. For

example, a uniform distribution is described by the endpoints in velocity space, might be used to represent a kind of beam population. If one uses a uniform distribution of v_{min} to v_{max} this implies mean μ and variance σ^2 of

$$\mu = \frac{1}{2}(v_{max} + v_{min}); \sigma^2 = \frac{1}{12}\left(v_{max}^2 - v_{min}^2\right) \quad (24).$$

This is independent of the number of points, assuming that we are in the central limit regime. There is a technical point of the Bessel correction in these cases leading to a factor of M/(M-1) and N/(N-1) in the variance of the daughter and parent population respectively, but for large M and large N, we do not see the challenges in matching both mean and variance that we see in the Maxwellian case.

Another function that exhibits this phenomena is the exponentials distribution where

$$f(v;\lambda) = \lambda e^{-\lambda v} \; for \; v \geq 0 \quad (25)$$

with

$$\mu = \frac{1}{\lambda}; \sigma^2 = \frac{1}{\lambda^2} \quad (26)$$

Using equation (19), it is easy calculate for exponentials

$$\frac{1}{\lambda^2} = \frac{N}{N-1}\left(\frac{2}{\lambda^2} - \frac{1}{\lambda^2}\right) = \frac{M}{M-1}\left(\frac{2}{\lambda^2} - \frac{1}{\lambda^2}\right) \quad (27)$$

where again, in the limit of large N and large M, we get both the mean and variance in both the parent and daughter population. It seems noteworthy that in both these cases, the mean and variance are determined by same set of parameters, whereas for a Maxwellian, the mean velocity and the temperature can be independent of each other. It is this independence that leads to challenges in crafting merged populations that conserve both momentum and energy.

In keeping with the speculation at the end of the section on the t-distribution, it is interesting to consider how to better use statistical knowledge when doing particle merging. For example, could we attempt to fit our particle populations to a distribution function and then draw samples to meet a given set of specifications? Additionally, perhaps it is possible to use statistical distribution to choose particles that could then be merged. That is develop a scheme that picks which particles to merge based on how well they approximate a given distribution function, say the exponential distribution, and then use our knowledge of mean and variance to match the first two moments of the parent population with our new merged daughter population. While this might be an expensive procedure, it would allow the potential of merging while conserving both energy and momentum. An additional benefit is it would rationally allow for particles that do not fit a given distribution to remain unmerged, thereby preserving their information content for the simulation. This is, in some sense, a generalization of schemes in the literature used to match fluid quantities (represented by Maxwellians) with kinetic information, but might be more robust given the statistical distributions naturally tie to physical moments that can be better parametrized by interdependent means and variances.

E. Summary

Confluent Sciences has applied basic statistics to the problem of particle merging in particle-in-cell software. In particular, it focuses on the link between the Maxwellian, the normal distribution, and the t-distribution for situations with low particle count, and then broadened our

investigation to consider other classes of distribution. These investigations have demonstrated that

- The mathematical nature of the Maxwellian allows the development of diagnostics to assess the equilibrium nature of a population, as well as measure the deviations from a Maxwellian via a single metric. This metric can be used to investigate if a merge algorithm preserves the Maxwellian nature of a population, although it does not provide the means to assess if the merger changed the populations temperature.
- Dramatic changes in the total number of particles during a merger can lead to a systematic change from a Maxwellian distribution to Student's t-distribution, especially when the particle count drops below approximately 30 particles per cell.
- If computational requirements force this dramatic reduction in the number of particles for Maxwellian distributions, it seems possible to use the t-distribution to craft merged populations that better approximate the parent population than one would anticipate from simple merger schemes.
- In the other limit of large parent populations (N) and large daughter populations (M), it is difficult, if not formally impossible to conserve both momentum and energy, or said another way, it is difficult to preserve the first and second moments of the distribution function (mean and variance).
- With M<N, and choosing to conserve mean momentum, we see an increase in the variance of the merged population. The effect of the merger is to increase the effective temperature of the population. This is probably most valid for momentum conserving PIC algorithms.
- It is possible to conserve temperature, but at the price of not preserving the average velocity of the distribution through the merger. This is probably most valid for energy conserving PIC algorithms. This will raise challenges for conserving charge and current density on the mesh, as conservation of momentum is a necessary but not sufficient step to conserve current density.
- It is possible to minimize the error in average velocity and temperature, but not formally preserve either. It is not clear the benefits and costs of this approach.
- There are other classes of distribution function where the mean and variance are not independent of each other. Examples include the exponential distribution function and the uniform distribution function. It is possible with these distribution functions to conserve both momentum and energy through a merger process.
- A merger algorithm might be possible that uses these interdependent distribution functions to choose particles to merge, thereby allowing the merge algorithm to preserve both the first and second moments of the distribution. It also offers the means to choose particles NOT to merge, thereby preserving information content in the simulation.

One unexamined area, especially for kinetic simulation where the Maxwellian equilibrium is not present is the potential role of higher order moments. It is well known in the fluid community that higher order moments can preserve important information in simulations. It is not atypical for fluid calculation to have 5 or even 13 moments to preserve non-equilibrium features of a simulation. It is not at all clear what a merger algorithm would do to a PIC calculation in this

regime. Given the challenges of preserving the zeroth, first, and second moments as we have discussed here, it may require completely new merger algorithms to simulate these situations.

V. STUDY OF THE LIMITATIONS OF THE NON-KINETIC ALGORITHM FOR ELECTRON SCATTERING

A. Introduction

An algorithm for modeling electron scattering from a background gas in non-kinetic PIC simulations has been developed for use in electromagnetic PIC codes and described in detail elsewhere. It was different from previous models in that it was not based on a null-collision algorithm, with its constraints on the timestep-gas density product, but could also deal with situations in which the chance of multiple collisions per timestep was significant. The previous document did not attempt to determine just how far (i.e., how many collisions per timestep (N_c) could be treated while retaining reasonable accuracy) the multi-collision algorithm could be pushed, and what might be done to mitigate any issues that arose as N_c became large. The effort summarized below addresses this question.

B. Possible Issues with the Algorithm

The multi-collision model makes the assumption that the parameters that determine the bulk characteristics of electron scattering do not change significantly any electron over a single timestep. There are two major parameters that effect scattering. First, the expected value of the number of collisions in a single timestep ($\mu_{total}(E) \equiv nv\sigma_{total}(E)\Delta t$), where n is the gas density, v is the electron velocity, σ_{total} is the sum of the cross sections of all collision processes, and Δt is the simulation timestep. Since both the cross section and the velocity are dependent on the electron's energy, this expected value is also dependent on energy. The second parameter influencing the scattering process is the $\xi(E)$ parameter, also energy-dependent, which defines the relationship between the differential and total cross sections. As this parameter varies over its range from -1 to 1, the scattering changes from complete back-scattering (at -1) to isotropic scattering (at 0) to complete forward scattering (at 1). Consequently, the assumption that these parameters do not change significantly over one timestep can be restated as the assumption that the energy of the electron does not change enough over the timestep to significantly change these parameters.

It is important to realize that the processes that cause μ_{total} at any given energy to be large are exactly the same as those that cause the drag force on the electron to be high, and consequently the energy lost by the electron over a timestep to be large. Consequently, one expects as this energy loss increases, it will eventually have a substantial effect upon the scattering of the electron, and consequently will limit the largest value of μ_{total} that can accurately be treated.

For a given gas mixture, density, and timestep, we have the information needed to quantify this situation. If we specify a limit on how much μ_{total} and ξ can change before the assumption that they are constant over a single timestep is no longer valid, we can define an associated limit on how much an electron's energy can change in that single timestep. In the

energy regime where μ_{total} is large, the dominant force on an electron is due to the decelerating drag force exerted by the gas. Consequently, we can get a fairly accurate value for the electron's energy loss over a timestep by integrating the drag force over the electron's path over the timestep. This loss is denoted as $E_{loss}(E)$. Note that this quantity is already available as a function of energy since it is needed to provide the impact ionization source term for the particular high-pressure gas model being used.

First, we will define two quantities, $\Delta E_{\mu}(E,\varepsilon)$ and $\Delta E_{\xi}(E,\varepsilon)$, to be the maximum change in energy, starting at E, for which the change in μ_{total} or ξ, respectively, is less than a maximum allowed change specified in some manner by the parameter ε. Specifically, define $\Delta E_{\xi}(E,\varepsilon)$ to be the minimum positive value of ΔE for which

$$\zeta(E)-\zeta(E-\Delta E)=\varepsilon\,. \qquad \text{* MERGEFORMAT (40)}$$

Similarly, define $\Delta E_{\mu}(E,\varepsilon)$ to be the minimum positive value of ΔE for which

$$\mu_{total}(E)-\mu_{total}(E-\Delta E)=\begin{cases}\varepsilon\mu_{low}, & \mu_{total}(E)<\mu_{low}\\ \varepsilon\mu_{total}(E), & \mu_{total}(E)>\mu_{low}\end{cases}\text{.* MERGEFORMAT (41)}$$

In * MERGEFORMAT (40), since the range of ξ is limited to -1 to 1, ε is chosen to be an absolute error tolerance, whereas in * MERGEFORMAT (41), it is a relative tolerance for all μ_{total} above some small value μ_{low}. While this particular choice could be debated, it is certainly a reasonable starting point for quantifying the effect of an electron's change in energy upon its scattering behavior. Finally, we will define $\Delta E_0(E,\varepsilon)$ to be the minimum of $\Delta E_{\mu}(E,\varepsilon)$ and $\Delta E_{\xi}(E,\varepsilon)$ for all values of E.

To demonstrate the relationship between these limits and the energy lost to drag forces, consider the case of dry air (79% N2, 21%O2) with a simulation timestep of 5.0 ps. The parameters associated with the limits defined by * MERGEFORMAT (40) and * MERGEFORMAT (41), ε and μlow, are chosen to be 1% and 1.0, respectively. Various regimes are explored by varying the gas pressure (i.e., its density n). For the study we use the default tabulation format used by Emphasis, i.e., logarithmically-uniform energy values between 0.1 and 106 eV, with 100 points/decade. First consider the relatively low-pressure case at 1.6 torr $(n\,;\ 5.2\times10^{22}\ \mathrm{m}^{-3})$, which has a maximum value of 0.1426, slightly higher that the peak value allowed for using the Nanbu-Sugawara (NS) algorithm when Perr is 1% (see [1]). Figure 71 shows the two energy loss limits ΔEξ and ΔEμ, as well as ΔE0, as a function of electron energy. Swanekamp's fit (see [1]) has been used to compute ξ(E). For comparison, Eloss is also plotted. The dotted gray line indicates the limit in which all of the electron's energy is lost. Note in this case, the energy loss never exceeds either loss limits. From this, we can draw the conclusion that exceeding these loss limits is never an issue in energy regimes for which the NS algorithm can be used with better that 1% accuracy.

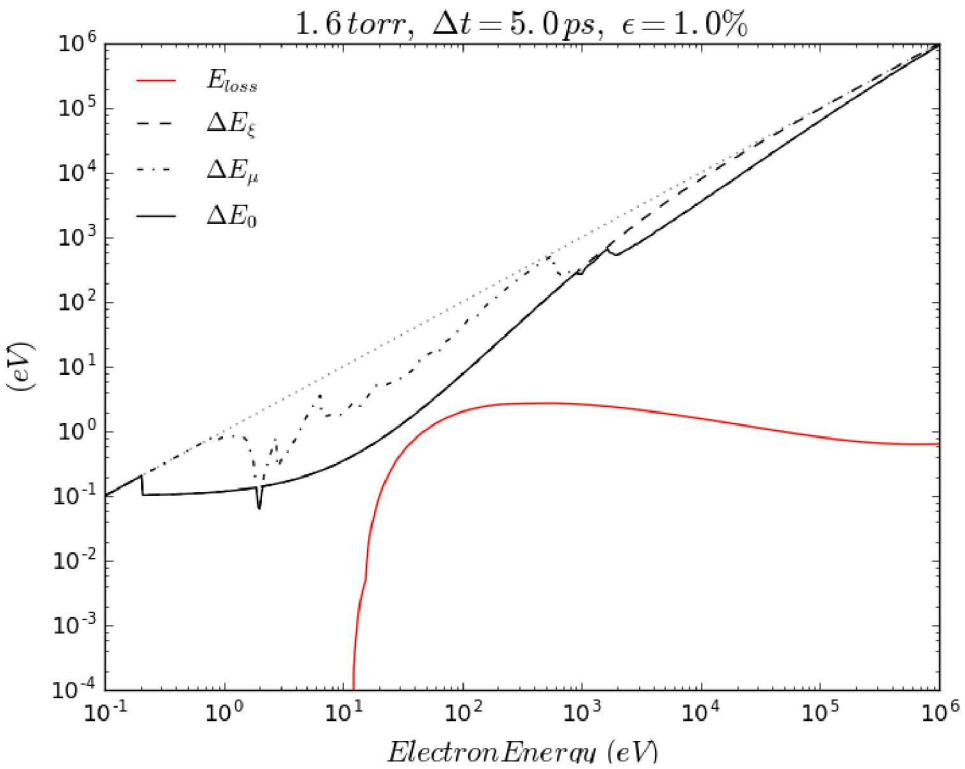


Figure 71. Energy loss limits and energy loss due to drag forces at 1.6 torr.

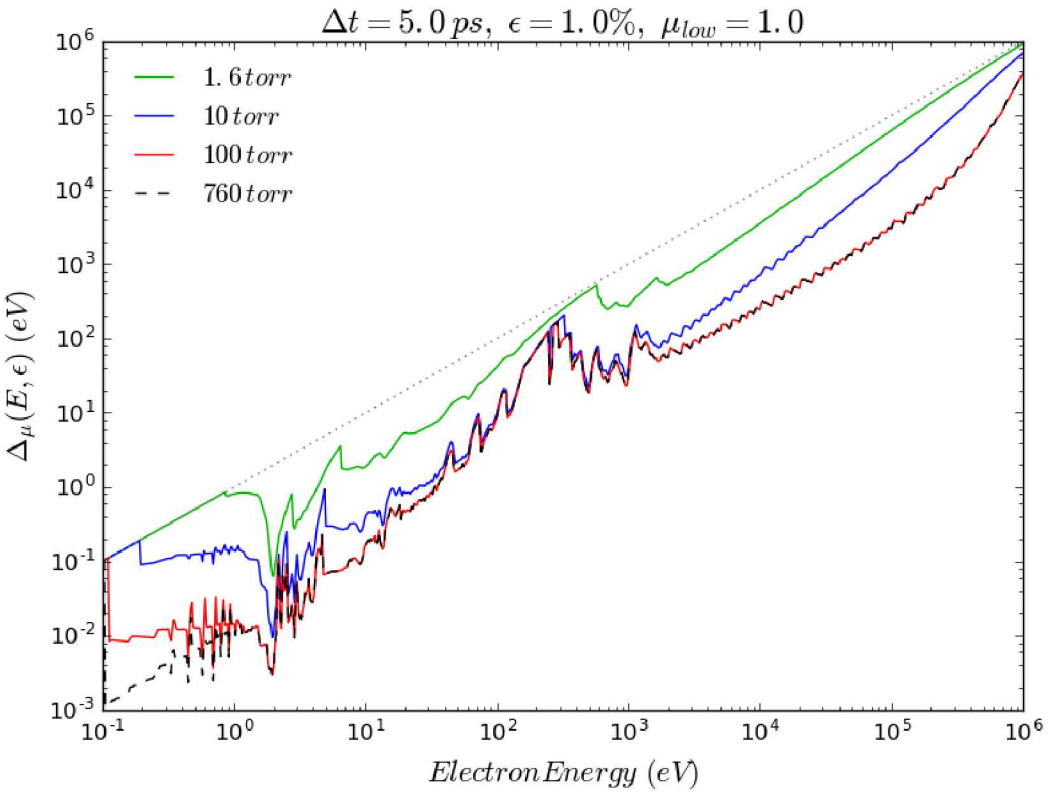


Figure 72. Variation of Δ_μ as a function of background gas density.

Examining * MERGEFORMAT (40), it is seen that the ξ-based energy limit is independent of the product nΔt, and since μtotal is directly proportional to this product, the μ–based limit is only dependent on it when μtotal is less than μlow. Figure 2 shows how Δμ changes as the gas density is increased. Since the peak value of μ_{total} at any energy does not exceed one for pressures less than ~12 torr, the first two curves have essentially no overlap, except at very low energies for which all the energy can be lost without exceeding the accuracy limit. As the density increases, it is seen that ΔE_μ overlaps for energies where μ_{total} exceeds μ_{low}.

In contrast to the ΔE limits, the energy lost due to the drag force is directly proportional to $n\Delta t$, and as this product increases, we expect to see an increasingly larger range of energies for which these limits are exceeded. For example, Figure 73 shows the relationship between ΔE_0 and E_{loss} at 10 torr. It is seen that ΔE_0 somewhat exceeds E_{loss} over a small portion of the energy range. Also shown is a "smoothed" version of ΔE_0, which was obtained by applying a boxcar (moving average) filter to the original values obtained using * MERGEFORMAT (41), i.e.,

$$f_i = \frac{1}{2k+1}\sum_{j=i-k}^{i+k} f_j, \text{ for } i \in [k, N-k], \text{ where } f_i \in (f_0, f_1, ..., f_N) \text{ . * MERGEFORMAT (42)}$$

Here, k is the "half-width" of the filter. For the smoothed curve shown in Figure 3, k = 10 (1/10 of a decade) was used. The "noise" in the unfiltered ΔE_0 is primarily due to rapid variations in the cross sections of the many excitation processes. However, below 400 eV, E_{loss} is approximated by the shape of just the ionization cross section. A more accurate approximation should probably include losses associated with the excitation processes as well, in which case the

"noise" in both quantities would tend to balance out. Consequently, this smoothing probably justified on physical grounds.

We can now define a quality factor as

$$Q(E,\varepsilon) = \frac{E_{loss}(E)}{\Delta E_0(E,\varepsilon)} \qquad \backslash^* \textbf{ MERGEFORMAT (43)}$$

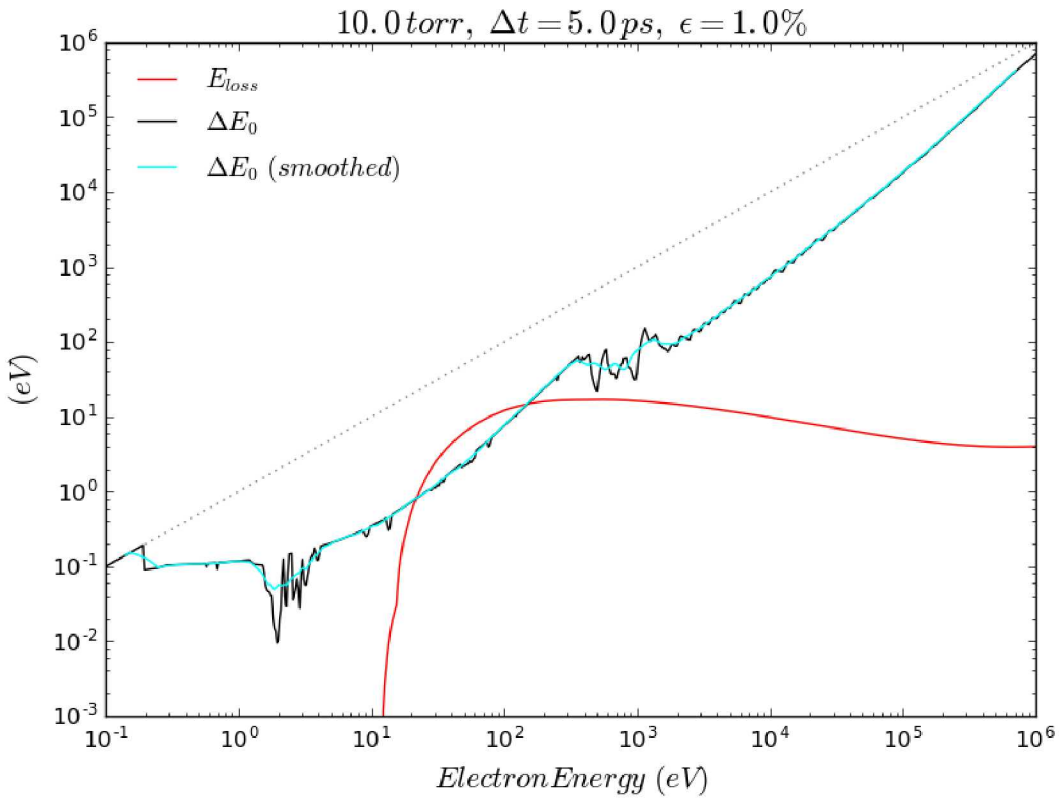


Figure 73: Energy loss limits and energy loss due to drag forces at 10 torr.

where Eloss(E) is the energy lost by an electron with initial energy E due to the drag force exerted by the background gas, as described earlier. If Q is greater than one, at least one of the two □E limits has been exceeded at that energy, and the timestep would need to be reduced by a factor of 1/Q in order to not exceed those limits. Figure 4 shows Q for the 10 torr case of . Here we have used the smoothed values of □E0 to compute Q using * MERGEFORMAT (43). Perhaps the most obvious solution to deal with energy regimes where Q > 1 is to subdivide the timestep, at least for the electron scattering, into multiple timesteps, and subcycle the scattering algorithm once for each of these sub-timesteps using the electron's energy centered in each substep. Since it is desirable to have an integral number of substeps in each timestep, we will convert Q to an integer value indicating the number of subcycles needed to meet our chosen accuracy constraints. Define

$$N_{substep}(E) = \max\left[1,\ \text{int}\left(Q(E)+\delta\right)\right], \qquad \backslash^* \textbf{ MERGEFORMAT (44)}$$

where δ is a number less than or equal to one. The accompanying plot shows such a conversion for the 10 torr case, indicating that as many as three substeps would be required. For this conversion, δ was arbitrarily chosen to be 0.9. Since μ_{total} is directly proportional to Δt, reducing the timestep used by the scattering algorithm means that the effective value of μ_{total} that will be used to compute the probability tables for the number of collisions, needs to be reduced accordingly. Consequently, we can define

$$\mu_{total,effective} \equiv \frac{\mu_{total}}{N_{substep}}. \qquad \text{* MERGEFORMAT (45)}$$

Figure 4 also shows the original μ_{total} and the modified effective μ_{total} for the 10 torr example.

Figure 74 and Figure 75 show Q, $N_{substep}$, μ_{total}, and $\mu_{total,effective}$ for 100 and 760 torr, respectively. As can be seen, substep counts as high as 15 and 24 are seen for the two cases, respectively, with correspondingly lower values of μ_{total} at energies at which the substep count is greater than one.

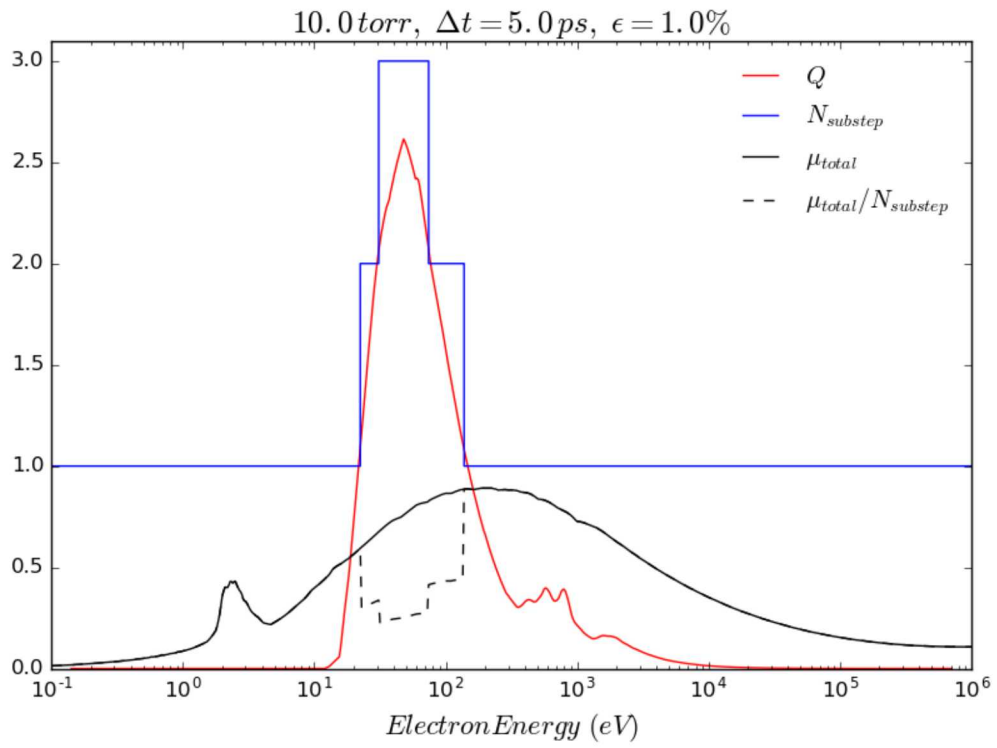


Figure 74: *Q* factor for 10 torr gas pressure. Also shown are $N_{substep}$, the original μ_{total}, and effective μ_{total}.

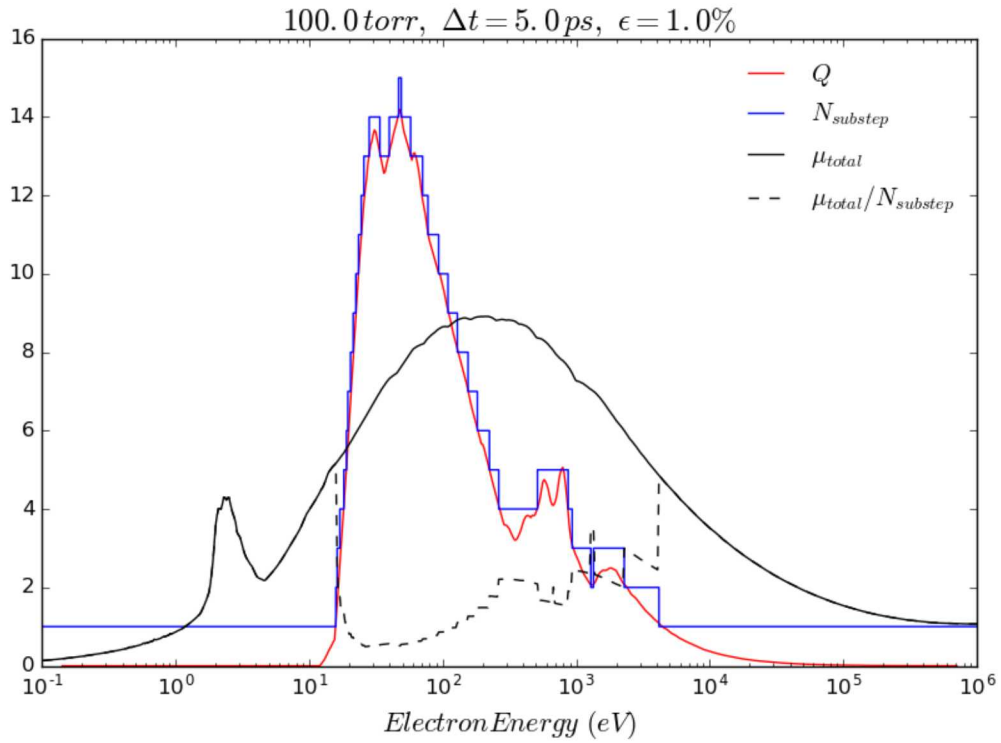


Figure 75: . *Q* factor for 100 torr gas pressure. Also shown are $N_{substep}$, the original

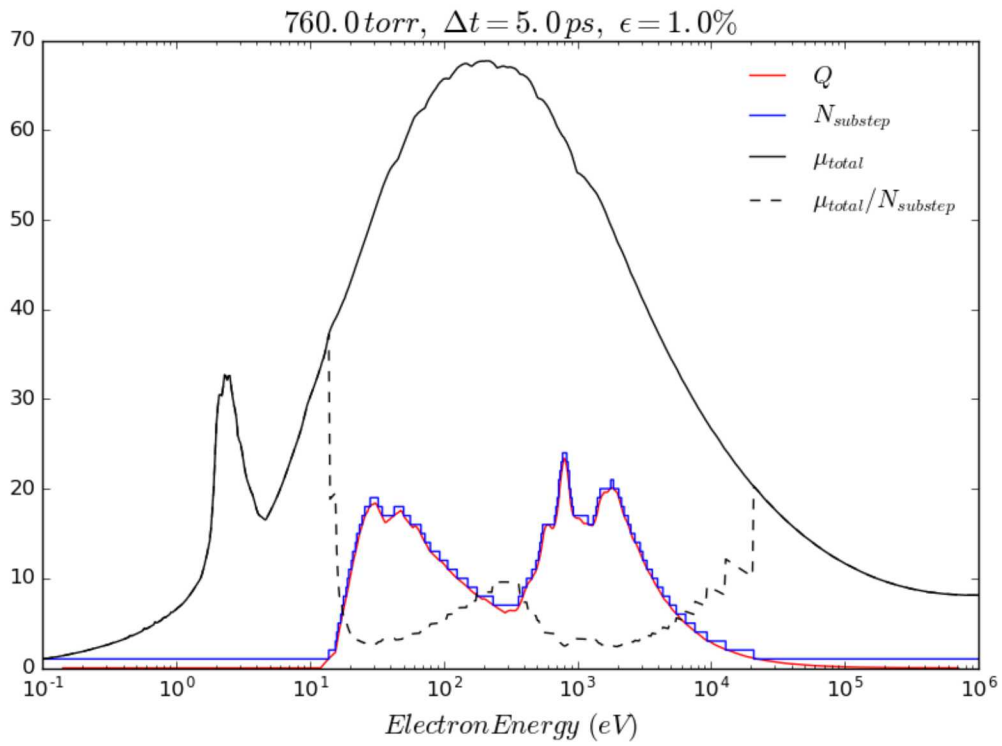


Figure 76: *Q* factor for 760 torr gas pressure. Also shown are $N_{substep}$, the original μ_{total}, and effective μ_{total}.

With a slight increase in the complexity of the structure of the lookup tables for collision count and process selection (P_c and P_{ps} in [1] , respectively), the algorithm described in [1] can be modified to selectively subcycle the scattering portion of the gas particle push algorithm. First, the quantities needed to compute $N_{substep}$ at each energy bin boundary would need to be computed. Note that all such quantities are readily available in Emphasis. Then, for each energy bin k, bounded by e_k and e_{k+1}, we need to compute and save the number of substeps required for electrons residing in the k^{th} bin, denoted as

$$N_{S,k} = \max\left(N_{substep}(e_k), N_{substep}(e_{k+1})\right). \qquad \backslash^* \textbf{MERGEFORMAT (46)}$$

The increased complexity in the table structure results from the fact that two columns (rather than one) will be required for any bin boundary whose adjacent bins have different values of $N_{S,k}$. Note that is conceptually no different than what the current algorithm requires for boundaries which separate adjacent NS and non-NS bins. Finally, the algorithm needs to be modified so that for an electron in bin k, the basic multi-collision algorithm must be called $N_{S,k}$ times, each with a different energy. Assuming that the force on the electron is roughly constant over the PIC timestep, the electron's momentum should vary linearly through the course of the timestep. So for the j^{th} substep, we will assume the electron's momentum is given by

$$\gamma v_j = \gamma v_{initial} + \frac{2j-1}{2N_{S,k}}\left(\gamma v_{final} - \gamma v_{initial}\right), \quad \text{* MERGEFORMAT (47)}$$

where $\gamma v_{initial}$ and γv_{final} are the electron's momentum at the beginning and end of the PIC timestep, respectively. Note that presently, Emphasis contains the code to subcycle the scattering step, although a fixed substep count is used for all energies, and it is assumed that the electron's energy is fixed to the electron's final energy for all substeps. Note that when $N_{S,k}$ is one, * MERGEFORMAT (47) yields a slightly different result than the existing Emphasis algorithm, since it uses an energy somewhere between the initial and final energies, rather than just the final energy. It could be argued that this is probably a better choice from a time-centering standpoint anyway.

C. Summary

For the case of dry air, it is clear that once the pressure-timestep product is ~50 torr-ps, energy losses over a single timestep can become significant and the accuracy of the multi-collision model comes into question. Note that this is less than an order of magnitude above the NS algorithm limit of ~8 torr-ps. One would expect this the basic behavior to be about the same for other gas mixtures of interest. Consequently, the modifications described in this document to the original algorithm should probably be considered as a required component if there is any hope of using the algorithm for problems for which the NS limit is exceeded by an order-of-magnitude or more.

In high-pressure (non-kinetic) gas models, such as the conductivity model implemented in Sandia's Emphasis code, simulation electrons are decelerated due to the drag force exerted on them by the gas, and the energy lost through this process provides an ionization source for the gas model. This approach accounts for the reduced velocity of the electrons, but does not introduce any change in their direction. This physical process clearly would introduce scattering of the electrons, which motivated Sandia to add a scattering model to the high-pressure gas model in Emphasis about ten years ago. In doing so, they chose to reuse those components of the Emphasis kinetic algorithm needed just for scattering with little or no modification. For reasons of algorithm efficiency, it turns out that this algorithm, known as a null-collision model, introduces a constraint on the maximum timestep size. This constraint is typically not a factor in regimes for which a kinetic treatment of the gas is required. However, this constraint can become the limiting factor in simulations with high gas pressure, and the problem is exacerbated for gases with a large number of collision processes, e.g., air.

Recently, Confluent Sciences personnel were tasked to add scattering to the non-kinetic gas models in the MEEC code. Since the MEEC code has no kinetic gas model, a scattering algorithm had to be developed without the benefit of reusing existing code. For this reason, a more general hybrid algorithm was developed that could use the more efficient null-collision model when its constraints were met, but could use a less restrictive (but also somewhat less efficient) multi-collision model which was developed as part of that task. This algorithm was successfully implemented in MEEC. Since Sandia now has some high-pressure regime applications for which their existing scattering model cannot practicably be used due to its timestep restrictions, modification of Emphasis's scattering algorithm to incorporate the new elements of the MEEC algorithm is highly desirable.

In the process of planning how these new elements would be incorporated into Emphasis, a decision first had to be made on how much of the existing Emphasis infrastructure could be reused. Many of the components needed for this algorithm could be used by the kinetic gas model – indeed, this is why Emphasis's existing high-pressure scattering model reused many component's of its already-existing kinetic model. However, it is not clear that anything other than the existing null-collision algorithm would ever be needed by the kinetic algorithm, since there are several other considerations that would prevent its usage at pressures beyond the constraints of the null-collision algorithm. For that reason, in consultation with Sandia personnel, the decision was made to develop a distinct class to implement high-pressure scattering, and only the code that is used to read gas cross section data from an external data file will be shared with the kinetic model. Additionally, the new model will initially exist in parallel with the existing high-pressure model, so that either could be used, which might be desirable, at least for the near future.

As of this writing, the class that will implement this enhanced scattering capability has been for the most part written, and has been extensively tested in a stand-alone test fixture. Full integration into Emphasis will require only a few minor modification, primarily dealing with the interface to the algorithm parameters that will be provided the Emphasis input file.

It had been expected that the algorithm would by this time be finished and fully integrated into Emphasis. However, there was one issue that had come up in the MEEC implementation, but due to lack of resources was never investigated. The issue was that there was no clear, quantitative understanding of how far the multi-collision model could be pushed in terms of pressure, which consequently increases the expected value of the number of collisions that will occur in a single timestep. Consequently, the decision was made to look into this issue in parallel with the Emphasis implementation effort. That study has been completed and the results have been documented above. The results can be summarized by saying that there is clearly a limit which is reached when the energy loss of an electron due to drag forces over a single timestep is large enough that the assumption that the characteristics of a scattering event do not change significantly is no longer valid. It was found that this limit occurs when the pressure-timestep product is ~50 torr-ps. To put this into context, the upper limit for the null-collision model now used in Emphasis is ~7 torr-ps, and for a problem at 1 atm. with a 5 ps timestep, a value of 3800 is needed. the obvious solution Tto this problem is to sub-cycle just the electron's scattering event, where the particle's energy is varied in each sub-cycle so as to be consistent with its deceleration over the timestep. Note that the number of sub-cycles required is a function of the electron's initial energy, peaking near the peak of the electron drag force, and in fact may be one over a significant portion of the remainder of the expected range of electron energies.

The modifications needed to extend the algorithm to incorporate sub-cycling is not extensive, and consequently are now in the process of being implemented in the new Emphasis class under development. It is felt that although this has somewhat delayed its completion, it will be worth the extra effort to have an algorithm that that can actually remain accurate in the pressure regimes needed for some of Sandia's applications. It remains possible, although unlikely, that finished new and modified Emphasis code might actually be delivered by the end of the period of performance for this contract. However, it is more likely that we will fall somewhat short of this goal, in which case the code in its existing state can be delivered, if so desired, along with a clear statement of what remains to be done. In any case, we expect the list of items in this statement to be relatively short.

VI. SUMMARY OF ICEPIC PLASMA CONDUCTIVITY MODEL

The plasma model provides a calculation of the electrical conductivity that is used in the solution of Maxwell's equations. The conductivity is driven by the electron number density. In the original version of the plasma model, the electron number density was driven exclusively by RF electric fields. In the DTRA-funded HiFSGEMP program, the model was extended by including an electron collisional model as an additional driver to the electron number density. Within the context of the plasma model, the electron number density is determined as the solution of a pair of coupled, time-dependent ordinary differential equations (ODE) in each cell governed by the plasma model, where the ODE includes various effects, such as losses due to attachment and recombination and gains from ionization of a background. The ODEs are:

$$\frac{dn_e}{dt} = \left[p\left(\alpha - s - \zeta p\right) - D \right] n_e - a n_e^2 + \gamma n_n n_i + S_c$$

$$\frac{dn_i}{dt} = psn_e - \gamma' n_n n_i$$

where n_e is the electron number density, n_n is the neutral species (background gas) number density, n_i is the negative ion (such as that which would result from attachment reactions) number density, p is the background gas pressure, s is a dissociation rate, α is the avalanche coefficient, a is the recombination rate. The electrical conductivity is calculated as

$$\sigma = bn_e / \nu + K_1 \Delta t$$

where b is a numerical constant, ν is the Spitzer collision rate, and K_1 is a user-input quantity that is useful for controlling the floor value of the conductivity.

The shortcoming was that this model did not have any direct connection to the actual electron population as represented by the macroparticles. Thus, it was insensitive to the intense emission of electrons that is characteristic of SGEMP. This shortcoming was fixed by providing the option of using the macroparticle population within each cell to calculate an ionization rate (the CIT or Collisional Ionization Term model). This does not actually go through with the process of causing ionization, it simply calculates a macroparticle-driven source term that is provided on a cell-by-cell/timestep-by-timestep basis to the plasma model. The collisional ionization term, S_c, is defined as

$$S_c = \sum_{n \in c} w_n \eta_{n,c} \left| \vec{v}_n \right| \sigma_i \left(\left| \vec{v}_n \right| \right) n_{b.c}$$

w_n is the macroparticle weight, $\eta_{n,c}$ is the fraction of macroparticle n present in cell c, $\vec{v}_n$ is the velocity of macroparticle n, σ_i is the ionization cross-section, and $n_{b,c}$ is the number density of the background gas present in cell c. The summation is over all macroparticles that have some overlap with cell c.

As this model was being implemented and tested, the possibility of augmenting the physics by including non-ionizing interactions with the background gas *via* CHIMP was considered. Without the avalanche ionization enabled by including ionizing collisions with the background *via* CHIMP, the impact on the overall computational burden of including CHIMP collisional interactions should be relatively small, yet the gain in making the physics more faithful to reality should be significant. This can be effected by setting up the calculation as thought the full CHIMP model were to be included in the calculation, then manually removing reference to the ionization reactions in the input file.

A test of the plasma model using CIT with and without the CHIMP scattering model shows that the inclusion of scattering can have a significant effect on the interaction of SGEMP emission and the background gas. A calculation was set up that consisted of a floating box that was to emit *via* the SGEMP emission model using a 10 keV black-body reverse emission from aluminum. The emission intensity was set such that in the absence of space-charge or background interaction effects, the current would rise at a constant rate from 0 at t = 0 to 900 A at t = 5 ns. The background gas was dry air (80% N_2, 20% O_2) with pressure 2000 mTorr. The calculation was run first with the plasma model (including the collisional ionization term), then with plasma model plus scattering interactions from CHIMP. Arguably an error, dissociative attachment ($e^- + O_2 \rightarrow O^- + O$) was included along with scattering interactions. The effect of this appears to have been minor, but further investigation is warranted going forward.

Figure 77 and Figure 78 show the electric field and current at the emission surface vs. time. The inclusion of scattering moderates the peak electric field significantly while keeping the current down, as well. With scattering, the electric field and current undergo damped oscillations with roughly a 90° phase difference between one and the other over the course of the calculation. Without the moderating effect of scattering, the plasma model predicts an emission current that is only slightly perturbed from the linear approach to 900 A at 5 ns. A check on the total inventory of electrons shows roughly ten times more electrons present in the calculation that included scattering (further checks confirmed that ionization reactions were indeed <u>*not*</u> taking place). The effect of scattering on the electron energy distribution function is also significant. Figure 79 shows the electron energy distribution calculated at the same point in time in the two calculations. Without scattering, the electron population remains concentrated above 1 keV whereas with scattering, the distribution function is much more complex and shows a significant fraction of the population with energies below 1 eV. Scattering is having the effect of confining electrons close to their point of emission, producing a higher density of electrons near the emitting surface that without collisions. This is shown in Figure 80, where a density greater that 10x that of the no-scattering calculation is seen at the emitting surface. Figure 81 and Figure 82 show the confining effect of collisions further – without collisions, electrons flow around the emitting body to meet on the side opposite to the emitting surface. This does not occur over the same timescale for the electrons affected by scattering with the background.

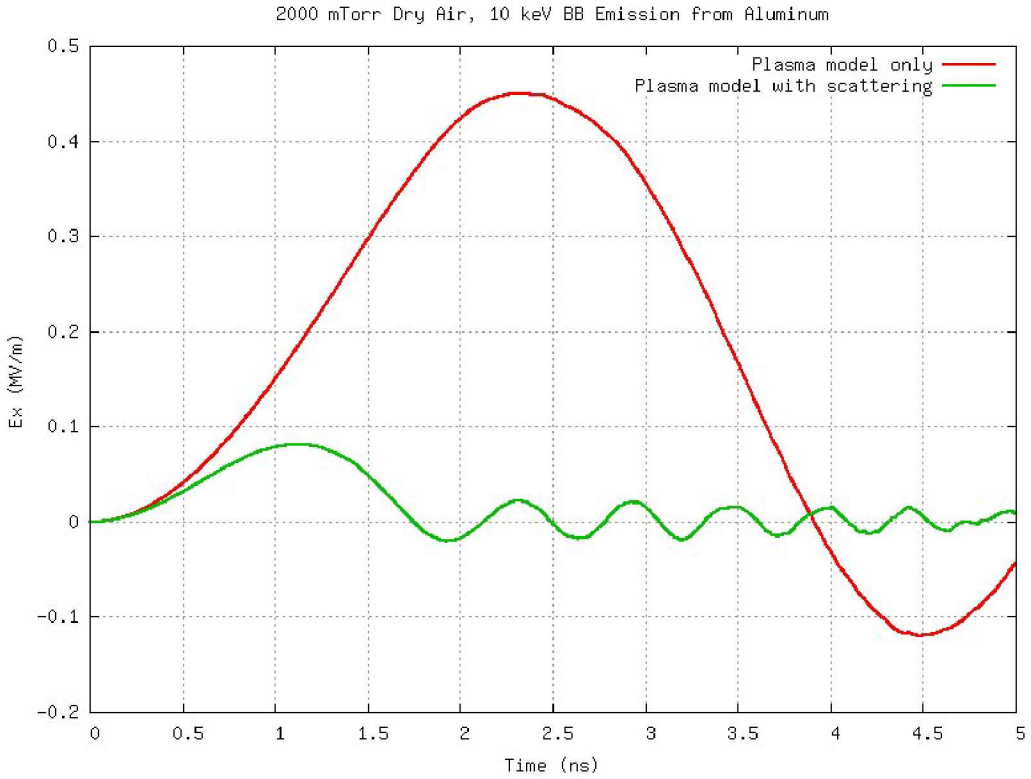


Figure 77: Electric field at the emission surface from the plasma model comparison tests

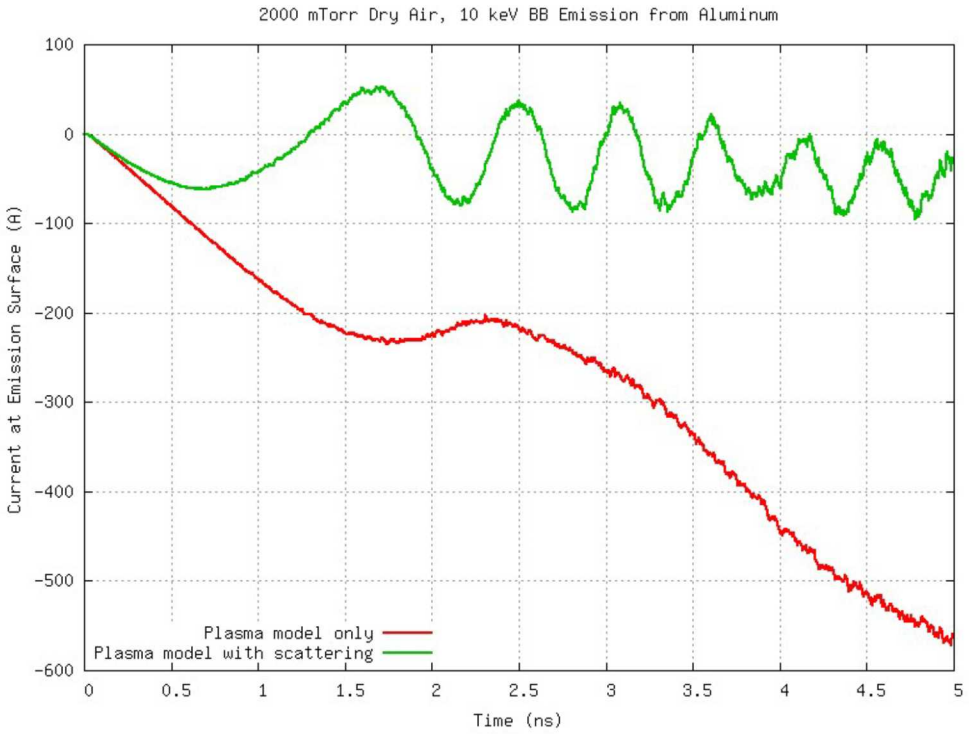


Figure 78: Current at the emission surface from the plasma model comparison tests

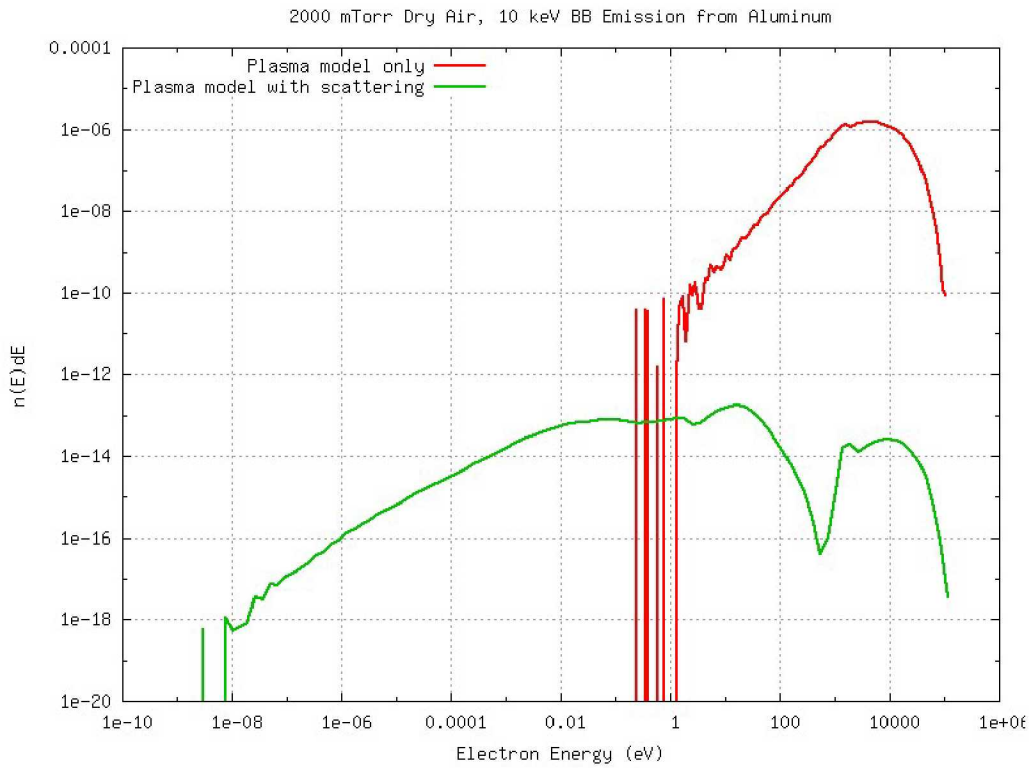


Figure 79: Electron energy distribution functions from the plasma model comparison tests

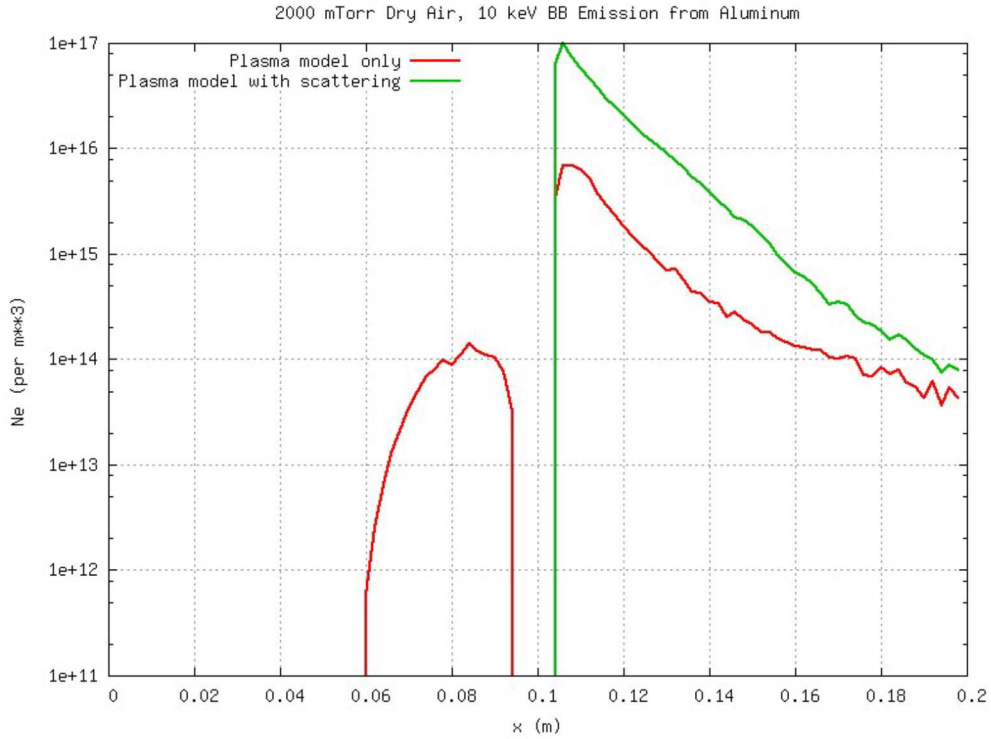


Figure 80: Electron number density along z=0 from the plasma model comparison tests

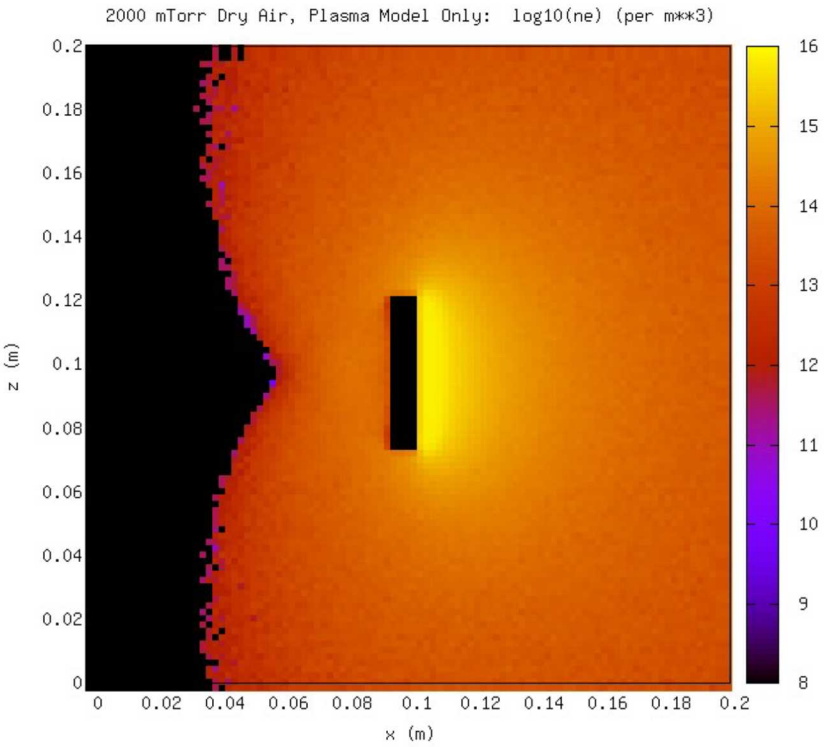


Figure 81: Electron number density from the plasma model without CHIMP scattering

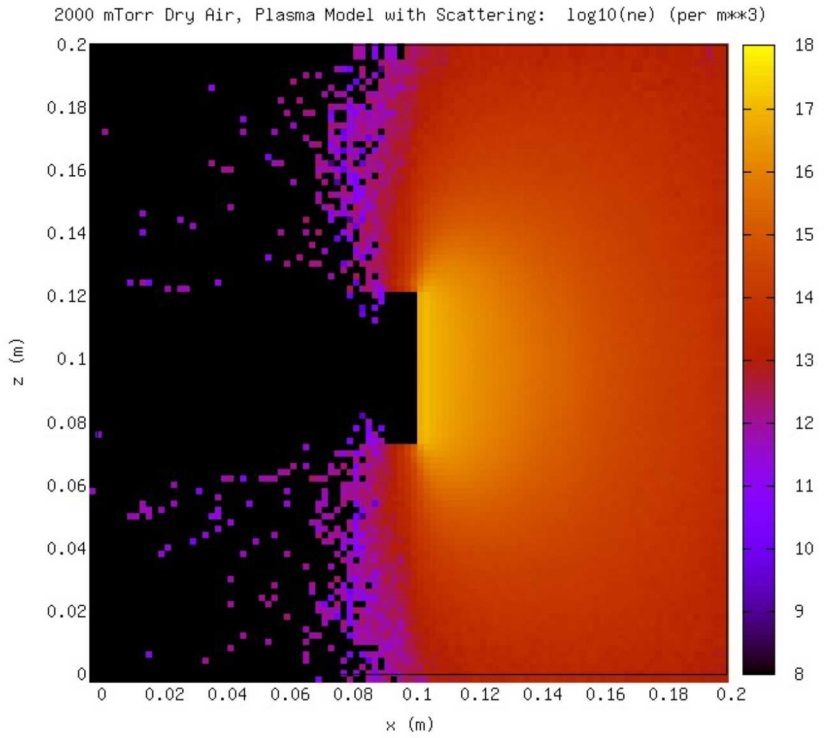


Figure 82: Electron number density from the plasma model including CHIMP scattering

As a further test of the model, the same calculation was rerun without CHIMP scattering. This provides a comparison of the PLASMA model with and without CIT, the collisional ionization term. Figure 83 shows the primary variable of the plasma conductivity model, the electron number density. Without CIT, the model is not driven at all, and the electron number density remains at its initial condition throughout the calculation. CIT drives the electron number density quite strongly. This has an effect on the electrical conductivity, as shown in Figure 84. With CIT driving the model, the electric conductivity increases by 6 orders of magnitude over what it is without CIT. This has a definite effect on the electromagnetic response, depicted though the electric field and current at the surface in Figure 85 and Figure 86, respectively.

The point of the PLASMA model is to provide a means of including aspects of the interaction with a background gas at high pressures without having the burden of macroparticles. The model has been exercised at relatively high pressures without any obvious difficulty. Figure 87 shows the same problem discussed above run at a number of high pressures. The PLASMA model including CIT was used, but CHIMP interactions were not included. The series of calculations went up to 100 Torr; higher pressures simply have not yet been explored. The effect, both physically and computationally, of including CHIMP scattering at these higher pressures has not yet been tested, but will be going forward.

The various options provided under the PLASMA model (plasma model alone, plasma model including CIT, plasma model including CIT and CHIMP scattering) should be tested against MEEC gas models and the scattering model being implemented in MEEC by Seidel.

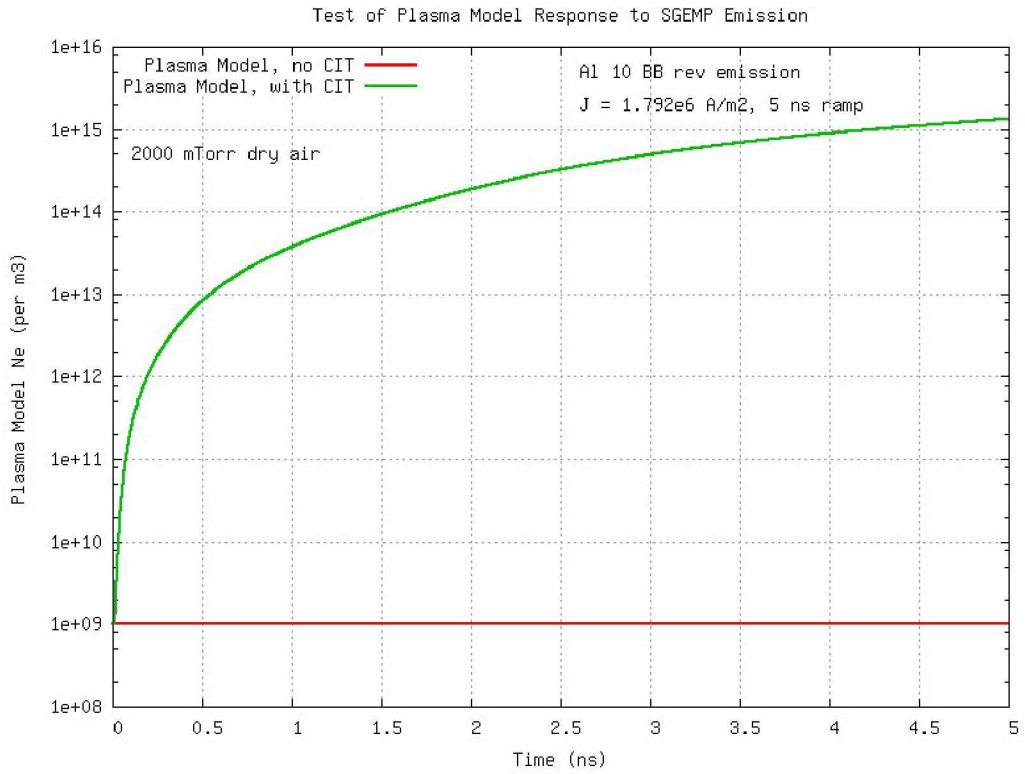


Figure 83: Plasma model prediction of electron number density with and without CIT

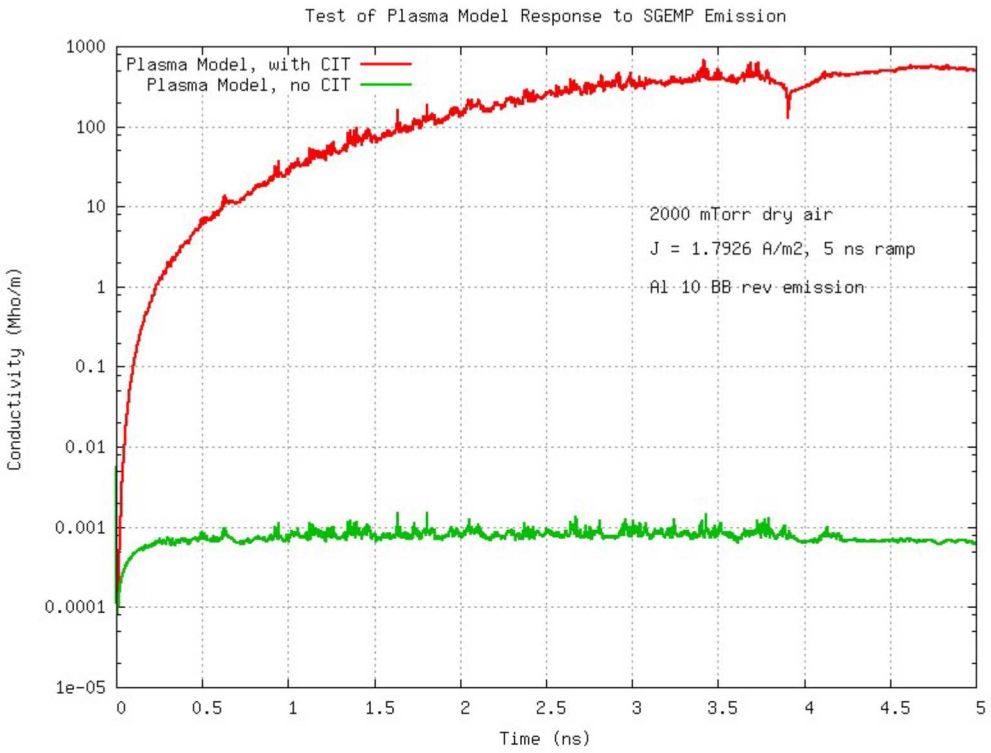


Figure 84: Plasma model prediction of electrical conductivity with and without CIT

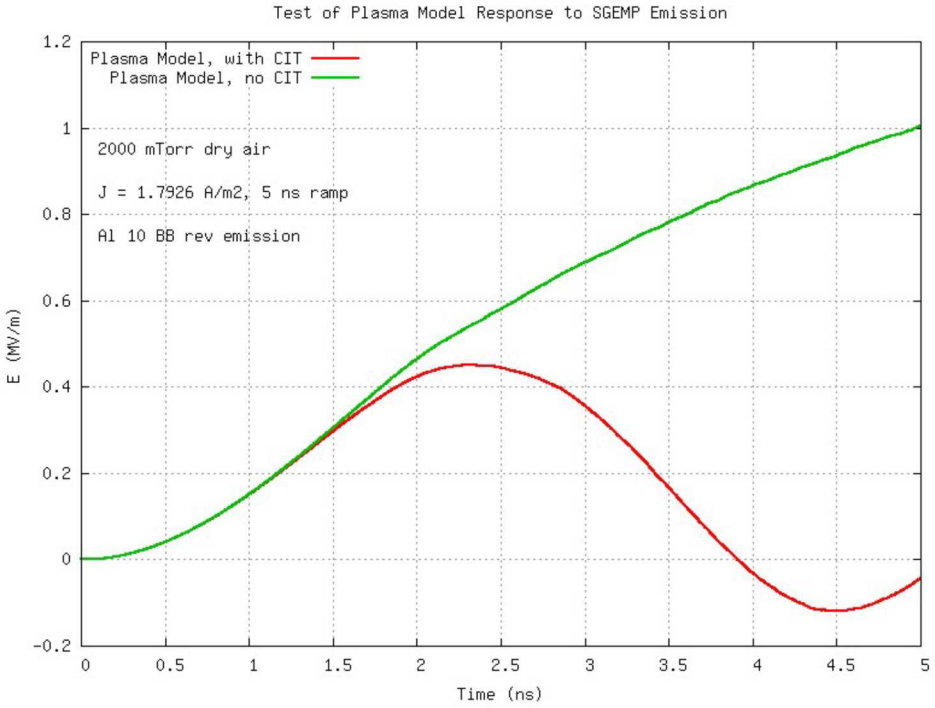


Figure 85: Plasma model prediction of electric field at emission surface with and without CIT

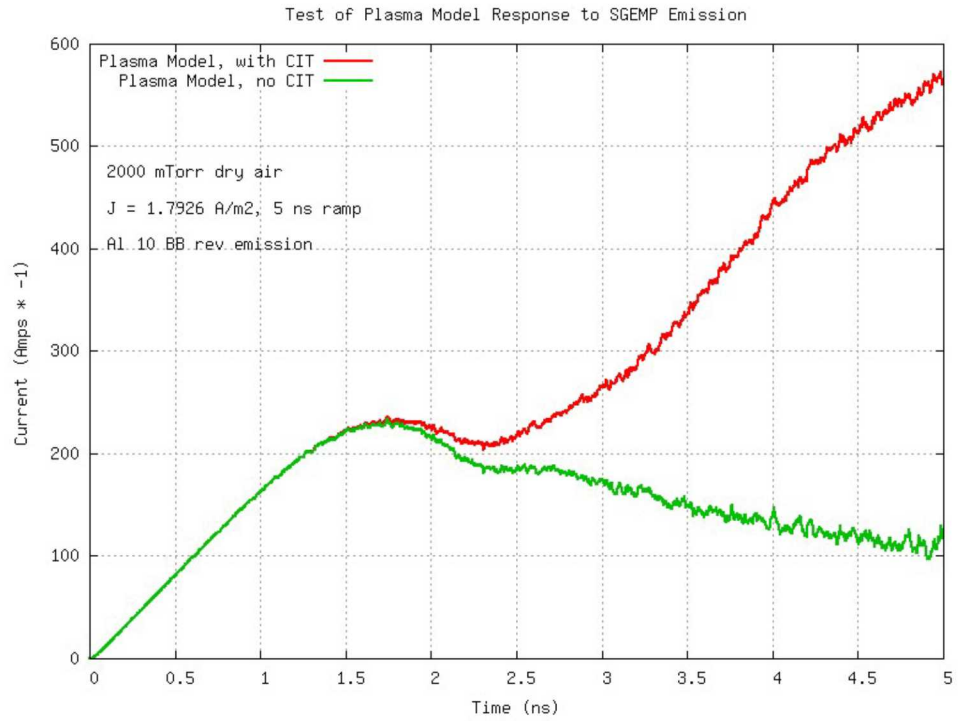


Figure 86: Plasma model prediction of current at emission surface with and without CIT

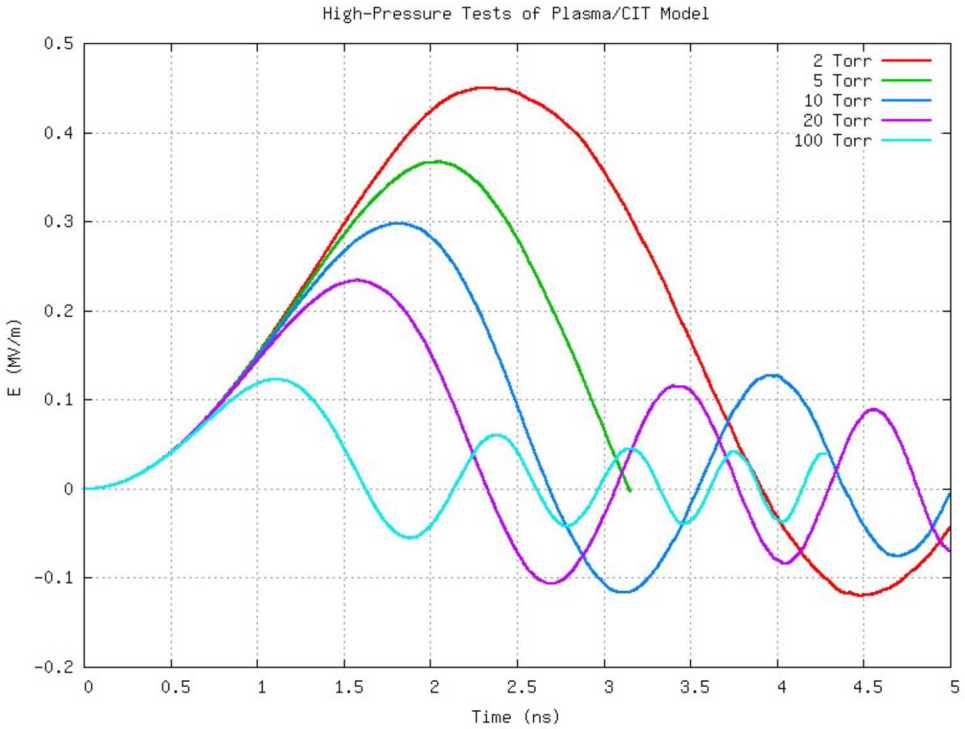


Figure 87: Plasma model at high pressures including CIT but no CHIMP interactions

REFERENCES [REFERENCE HEAD]

1.1.1.

Martin, R. S., & Cambier, J.-L. (2016). Octree particle management for DSMC and PIC simulations. *Journal of Computational Physics, 327*, 943-966.

Stanton, G. a. (2018). Multiscale Molecular Dynamics Model for Heterogeneous Charged Systems. *Physical Review X, 8*(021044).

Villasenor, J., & Buneman, O. (1992). Rigorous charge conservation for local electromagnetic field solvers. *Computer Physics Communications, 69*, 306-316.

DISTRIBUTION

4 Lawrence Livermore National Laboratory
Attn: N. Dunipace (1)
P.O. Box 808, MS L-795
Livermore, CA 94551-0808

1 MS0899 Technical Library 9536 (electronic copy)

Sandia National Laboratories