Boltzmann-Electron Model in Aleph

Thomas P. Hughes and Russell W. Hooper

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

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.
Boltzmann-Electron Model in Aleph

Thomas P. Hughes
Applied Science and Technology Maturation
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-MS0878
thughe@sandia.gov

Russell W. Hooper
Multiphysics Applications
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-MS1318
rhoope@sandia.gov

Abstract

We apply the Boltzmann-electron model in the electrostatic, particle-in-cell, finite-element code Aleph to a plasma sheath. By assuming a Boltzmann energy distribution for the electrons, the model eliminates the need to resolve the electron plasma frequency, and avoids the numerical “grid instability” that can cause unphysical heating of electrons. This allows much larger timesteps to be used than with kinetic electrons. Ions are treated with the standard PIC algorithm. The Boltzmann-electron model requires solution of a nonlinear Poisson equation, for which we use an iterative Newton solver (NOX) from the Trilinos Project. Results for the spatial variation of density and voltage in the plasma sheath agree well with an analytic model.
Acknowledgments

We thank Jeremy Boerner (Dept. 1516) for providing Aleph input and output files, and files containing analytic results from his recent study of plasma sheaths[1].

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-AC04-94AL85000.
Executive Summary

Fully-kinetic numerical simulations of dense plasmas using the particle-in-cell (PIC) method have to deal with high-frequency electron oscillations and short-wavelength numerical heating of electrons due to the so-called “grid instability”. As a result, one has to either use sufficiently small timesteps and mesh sizes (which may result in prohibitively long and large calculations), use an implicit-PIC treatment of the electrons, or use a non-PIC treatment of the electrons that removes these constraints. The latter can be a good alternative when the high-frequency/short-wavelength electron physics can be neglected for the application of interest, and a approximate non-kinetic electron model exists. In this report, we examine the Aleph implementation of the Boltzmann-electron model, where ions are treated using the standard (explicit) PIC algorithm, but electrons appear only as a charge-density term in the equation for the electric potential. The resulting Poisson equation is nonlinear, and requires a different solver than that used for the standard linear Poisson equation. We use the NOX solver developed for the SNL Trilinos Project, and test the model by applying it to a plasma-sheath problem previously developed to test the fully-kinetic PIC model in Aleph [1]. In Fig. 1, we show good agreement between an analytic sheath model and the Boltzmann-electron simulation using a timestep 10 times that used in the fully-kinetic study. Timesteps of 100 and 1000 times the fully-kinetic timestep give similar results, though in the latter case, the ion density is more noisy. The upper limit on the timestep is determined by the ion velocity. An input file using a timestep 100 times the fully-kinetic value is in the Aleph repository in directory examples/boltzmann_sheath_report.
Figure 1. Comparison of Boltzmann-electron calculation for a plasma sheath with an analytic model. Plot (a) compares the ion density and (b) compares the electric potential (the lines are almost indistinguishable). The Aleph simulation used a timestep 10 times that used in a recent fully-kinetic study [1]. The spatial mesh size is $\lambda_D/4$ (where $\lambda_D$ is the Debye length) the same as for the fully-kinetic simulations.
Contents

1 Introduction ................................................................. 9
2 Boltzmann-electron Model ............................................... 11
3 Nonlinear Solution Algorithm in Aleph .............................. 13
4 One-dimensional Results ................................................. 17
5 Discussion and Further Work .......................................... 23

Appendix

A Cubit input ................................................................. 27
B Aleph input ................................................................. 28
1 Introduction

In kinetic simulations of a collisionless plasma the spatial and temporal resolution needed for numerical stability are often set by the electron Debye length, $\lambda_D$, and the electron plasma frequency, $\omega_{pe}$, respectively (see Fig. 2). For applications where physical phenomena at these length- and time-scales can be neglected, these constraints can lead to prohibitively large meshes and computation times. Various schemes have been devised to avoid this problem. Implicit-kinetic schemes [2–4] retain kinetic electrons, but filter their high-frequency motion. Hybrid schemes [5, 6] treat the ions kinetically and replace kinetic electrons with a fluid momentum-equation in which the electron inertia term is usually neglected. One of the simplest examples of the latter is the ambipolar approximation, whose implementation in Aleph was examined in Ref. [7]. This approximation assumes (a) quasineutrality ($n_e \approx n_i$, where $n_e$, $n_i$ are the electron and ion densities, respectively, assuming singly-ionized ions) and (b) that the electron pressure gradient is balanced by an electric field created by charge-separation between fast-moving electrons and slower ions. In this report, we examine another hybrid model, the Boltzmann-electron model, which treats the electrons as a charged, hot gas in thermal equilibrium with the electric potential, but does not assume quasineutrality. The electrons appear only as a negative-charge-density term in the Poisson equation. Like the ambipolar model, this eliminates the electron plasma oscillations. Unlike the ambipolar model, the Boltzmann-electron model is able to treat plasma sheaths since it allows strongly non-neutral regions. These properties make it potentially much more efficient than the full-kinetic algorithm for modeling ion-beam extraction from a plasma.

Both the Boltzmann-electron and ambipolar-field models specify an electron temperature that must be provided by the user. It is usually derived from experimental measurements.
Figure 2. Numerical stability plane for an explicit, electrostatic, fully-kinetic simulation of a stationary thermal plasma. Above $\omega_{pe}\Delta t = 2$, electron oscillations are unstable. The other boundaries are not sharp. The numerical heating-rate of electrons increases as $\Delta x/\lambda_D$ increases. Accuracy of orbit integration for faster electrons decreases as $v_{the}\Delta t/\Delta x$ increases, where $v_{the}$ is the thermal-electron speed. Kinetic-electron simulations are typically carried out in the shaded triangle. The kinetic-electron simulation in Ref. [1] is indicated by the red dot ($\omega_{pe}\Delta t_{ke} = 0.027$, $\Delta x_{ke}/\lambda_D = 0.25$), and two of the Boltzmann-electron calculations in this report are indicated by blue dots: (a) has timestep $10\Delta t_{ke}$ and (b) has $100\Delta t_{ke}$, where $\Delta t_{ke}$ is the kinetic-electron timestep.
2 Boltzmann-electron Model

The Boltzmann-electron model is arrived at by assuming that the electrons have an equilibrium Maxwell-Boltzmann distribution of the form

\[ f_e(x, v) = A \exp \left[ - \left( \frac{1}{2} m_e v^2 - e\phi(x) \right) / kT_e \right] \]  

where \( x, v \) are position and velocity vectors, respectively, \( A \) is a normalization constant, \( m_e, -e \) are the electron mass and charge, respectively, \( \phi \) is the electrostatic potential, and \( T_e \) is the electron temperature. The potential is usually measured with respect to a location, \( x_0 \), where it reaches its maximum value, and for simplicity we set \( \phi(x_0) = 0 \). Integrating Eq. 1 over velocity gives the electron density at any point in space:

\[ n_e(x) = n_{e0} e^{\phi(x)/kT_e} \]  

where \( n_{e0} \) is the electron density at \( x_0 \). Equations 1 and 2 describe an electron gas with temperature \( T_e \) everywhere and whose density varies exponentially with the electric potential. It may be useful in some cases to make \( T_e \) a function of position, \( x \). Equation 2 can be used to describe the electron population of a plasma where the bulk of the plasma is quasi-neutral and charge-separation is created, e.g., by electrons escaping at the boundaries or by inserting biased electrodes. The net charge-density, \( \rho \), in the plasma is then given by

\[ \rho = \rho_i - \rho_e = e \left( n_i(x) - n_{e0} e^{\phi(x)/kT_e} \right) \]  

where we have assumed singly-charged ions. Typically, we set \( n_{e0} = n_i(x_0) \) using the assumption of quasineutrality. Note that in regions where \( e\phi(x)/kT_e \ll 1 \) and \( \rho \approx 0 \), Eq. 3 yields

\[ \nabla n_i \approx \frac{n_{e0} e}{kT_e} \nabla \phi = -\frac{e n_i E}{kT_e} \]  

which gives the expression for \( E \) used in the ambipolar approximation (see Eq. (3) in Ref. [7]). Thus, in quasi-neutral regions, the ambipolar and Boltzmann-electron models are equivalent. The Boltzmann-electron model does not assume quasineutrality, however, and so can be used in non-neutral regions such as plasma sheaths. This makes it applicable to the extraction of ion beams from a plasma source. It also means that a nonlinear Poisson equation must be solved to obtain \( E \) instead of using the algebraic expression, Eq. 4. The solution method is discussed in the following section.
3 Nonlinear Solution Algorithm in Aleph

Using Eq. 3 in Poisson’s equation gives the equation for the electrostatic potential in the Boltzmann-electron model (in SI units):

\[ \nabla \cdot (\epsilon \nabla \phi) = -\rho = -e \left( Z \cdot n_i(x) - n_e 0 e^{(\phi - \phi_0)/kT_e} \right) \]  

(5)

where \( \epsilon \) is the scalar dielectric permittivity, \( Z \) is the integer ion charge-state, and \( n_i \) is the density of ions treated using a kinetic particle description. Equation 5 is a nonlinear partial differential equation (PDE), sometimes referred to as the Poisson-Boltzmann equation, in the unknown potential, \( \phi \). We have introduced an explicit reference potential, \( \phi_0 \), as a convenience when setting up a calculation. For a simulation where the source-plasma is injected through a boundary, for example, the user will usually set \( \phi_0 \) in the input parameters for the Boltzmann-electron algorithm equal to the value of the Dirichlet boundary-condition for the potential at the plasma injection boundary (see lines 11 and 29 in the input file listing in Appendix B).

To solve Eq. 5, Aleph uses an iterative “approximate Newton” algorithm which leverages the existing infrastructure for the linear Poisson equation used when both ions and electrons are treated kinetically, i.e., when there is no explicit \( \phi \)-dependence in the source term. We employ an existing nonlinear-solver package, NOX, developed for the Trilinos Project by a team that included one of the authors of this report (RWH) [8].

We begin by casting the PDE into residual form,

\[ R(\phi) = \nabla \cdot (\epsilon \nabla \phi) + e n_i - e n_e e^{(\phi - \phi_0)/kT_e} = 0. \]  

(6)

The Newton algorithm is based on expanding the residual, \( R(\phi) \), about the approximate nonlinear solution \( \phi^k \) at the \( k \)th iteration:

\[ R(\phi) = R(\phi^k) + \frac{\partial R}{\partial \phi} \bigg|_{\phi^k} (\phi - \phi^k) + O \left( ||\phi - \phi^k|| \right)^2. \]  

(7)

If we truncate second order and higher terms, define \( J \equiv \partial R/\partial \phi \), define the correction to \( \phi^k \) at iteration \( k \) as \( \Delta \phi^k \equiv \phi - \phi^k \), and note that the solution must satisfy \( R(\phi) = 0 \), we obtain a linear PDE to be solved for \( \Delta \phi^k \):

\[ J^k \Delta \phi^k = -R^k, \]  

(8)

where the superscript \( k \) on \( J^k \) and \( R^k \) indicate that they are evaluated at \( \phi^k \). The solution to this equation is then used to correct the solution to the nonlinear PDE:

\[ \phi^{k+1} = \phi^k + \Delta \phi^k. \]  

(9)

We now perform another Newton iteration by solving Eq. 8 again, with \( J \) and \( R \) evaluated at \( \phi^{k+1} \), and this gives us a further correction to the numerical solution. This process is repeated until the user-specified measure of convergence is achieved (see below).
In the numerical scheme, Eq. 8 is discretized using a Galerkin finite-element method on an unstructured triangular (2D) or tetrahedral (3D) mesh. Equation 8 becomes a matrix equation in which \( \Delta \phi, R \) are vectors \( \Delta \phi, R \) of length \( N \), respectively, where \( N \) is the number of unknown values of \( \phi \) on the mesh, and \( J \) is a sparse matrix \( J \) of size \( N \times N \). If we number the equations so that the \( n^{th} \) element of \( R \) contains the source term located at the \( n^{th} \) node of the mesh, then the \( n^{th} \) row of \( J \) will in general have the coefficient of \( \Delta \phi_n \) on the diagonal, and several (but \( \ll N \)) non-zero off-diagonal coefficients resulting from the spatial derivatives in the original PDE.

Efficient parallel methods have been developed for solving linear matrix equations using “direction-set” methods (see Sec. 10.5 of Ref. [9]). These methods are based on the geometrical idea of generating a set of orthogonal directions in the column-space of the matrix, and searching along these directions to find the vector that solves the matrix equation. For the equation \( A \cdot x = b \), where \( A \) is a non-singular matrix of rank \( n \), the column space is spanned by the \( n \) vectors of the “Krylov sequence” \( \{ b, A \cdot b, ..., A^{n-1} \cdot b \} \). Using these vectors, the direction-generating and search processes can be carried out using only the operation of \( A \) on vectors, i.e., matrix-vector products, for which fast parallel algorithms exist. In a Newton method, there is an additional advantage: the product \( J^k \cdot p \) can be obtained approximately without computing \( J^k \) itself by noting that

\[
R(\phi^k + \epsilon p) = R(\phi^k) + J|_{\phi^k} (\epsilon p) + O(||\epsilon p||^2),
\]

where the parameter \( \epsilon \) is a small number typically of \( O(10^{-6}) \). This gives a first-order accurate approximation for \( J^k \cdot p \) at iteration \( k \):

\[
J^k \cdot p \approx \frac{R(\phi^k + \epsilon p) - R(\phi^k)}{\epsilon}.
\]

The efficiency of direction-set methods can be greatly increased in some cases by using preconditioning to improve the numerical properties of \( J^k \). Preconditioning is designed to transform \( J^k \) to a more orthogonal matrix with a condition number closer to 1, i.e., to approximate the identity matrix. Without preconditioning it is not uncommon for the number of iterations required to solve the linear system to approach the number of unknowns in the problem. Good preconditioners reduce the iteration count by orders of magnitude. The standard Poisson-solver in Aleph is a case where the extra operations needed to carry out one-time preconditioning result in a large reduction in the number of iterations required to recompute \( \phi \) on each timestep as the source-term changes. This gives a large net reduction in total computation-time spent in the field-solver for a run with many timesteps. If preconditioning is required on every timestep, the net benefit may need to be measured.

In the context of a Newton method the best preconditioner matrix would be the Jacobian matrix, but since \( J^k \) depends on \( \phi^k \) because of the nonlinearity of Eq. 6, this would require repeating the preconditioning operations on each time \( \phi^k \) changes. Instead, Aleph currently uses a constant matrix, \( M \), obtained from the Laplacian operator (i.e., dropping the contribution from the exponential term in Eq. 6) as the preconditioner matrix:

\[
M = \int_\Omega \epsilon \nabla \xi \cdot \nabla \psi d\Omega
\]
where $\xi$, $\psi$ are FEM test functions, and integration is over the problem domain $\Omega$. This matrix is computed only once and then used for all the Newton iterations, and has been found to perform reasonably well in practice.

Some alternatives to using $M$ as the preconditioner that could be tried include:

1. Use $J^k$, so that preconditioning is performed each time $\phi^k$ changes.
2. Use $J$ from the previous timestep.
3. Use $M - 1/\lambda_0^2$ instead of $M$ (see Eq. 16 below).

Using the default parameters set in Aleph, the Newton iterations stop when either (a) 20 iterations have been performed, or (b) when all three of the following conditions are met:

1. The RMS value of the components of the residual vector $R^k$ is small:
   \[
   \sqrt{\frac{1}{N} \sum_{i=1}^{N} (R^k_i)^2} < \delta_r. 
   \]  
   where $\delta_r = 10^{-8}$.

2. The change in the RMS value of the components of the $\phi^k$ vector from one Newton iteration to the next is small:
   \[
   \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left( \frac{\phi^k_i - \phi^{k-1}_i}{RTOL |\phi^{k-1}_i| + ATOL} \right)^2} < 1. 
   \]  
   where $RTOL = 10^{-2}$ and $ATOL = 10^{-8}$ are relative and absolute tolerances, respectively.

3. The RMS value of the components of the update vector $\Delta \phi^k$ is small:
   \[
   \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\Delta \phi^k_i)^2} < \delta_u. 
   \]  
   where $\delta_u = 10^{-5}$.

We conclude this section by mentioning some previous work involving Eq. 5 and related equations. Reference [10] solved Eq. 5 to obtain plasma and beam equilibria in 3D ion-beam systems. They used an iterative Newton scheme where complete ion orbits are traced through the entire system each time Eq. 5 is solved. They appear to use a direct method, which they call “modified Gauss-Cholesky” factorization, to solve the linearized equation inside the Newton iteration loop (GMRES was not invented until 1986). An equation similar to Eq. 5 arises in the Debye-Hückel treatment of electrolytes [11], and Ref. [12] describes a numerical solution method that uses a multilevel solver for the linearized equation rather than the direction-set method used in the present calculations.
4 One-dimensional Results

In many plasma applications, the region where Eq. 5 is most nonlinear is either near boundaries, where the plasma is in contact with a particle-absorbing wall or diagnostic probe (a plasma sheath), or in a region where ions are extracted from the plasma to form an ion beam (a plasma meniscus). In the bulk of the plasma, deviations from quasineutrality are typically small, and the exponential term can be linearized, as described in Sec. 2. Equation 5 then becomes a linear “modified Poisson” equation

\[ \nabla \cdot (\epsilon_r \nabla \phi) - \frac{\phi - \phi_0}{\lambda_{D0}^2} = -\frac{e}{\epsilon_0} (Z_i n_i (x) - n_{e0}) \]  

(16)

where \( \epsilon_r = \epsilon/\epsilon_0 \) is the relative permittivity (\( \epsilon_0 \) is the permittivity of free space) and \( \lambda_{D0} = \sqrt{e^2 n_{e0}/\epsilon_0 kT_e} \) is a constant equal to the electron Debye-length at \( x_0 \). (Note that neglecting the Laplacian term in Eq. 16 and taking the gradient of each side again leads to the ambipolar expression for \( E \) (cf. Eq. 4). Thus, the ambipolar field is obtained in the “long-wavelength limit” of Eq. 16.) In an ion-extraction configuration, on the other hand, there are essentially no electrons in the ion beam downstream of the meniscus, and so there is no explicit dependence on \( \phi \) on the RHS of Eq. 5. In sheath/meniscus regions, however, the exponential term is neither negligible nor linearizable, and a nonlinear solver like that described in Sec. 3 must be used.

In Ref. [1], J. Boerner obtained good agreement between an analytic model and Aleph calculations of a collisionless sheath using kinetic ions and electrons. Figure 3 shows the geometry and results for his “Reference” case [1]. The bulk plasma has equal electron and proton number densities of \( 10^{19} \text{ m}^{-3} \). The electrons are given an isotropic temperature \( kT_e = 5 \text{ eV} \), while the ions are essentially cold. A potential difference of 15 V is imposed between the left and right boundaries. Ions are injected from the right boundary with a velocity corresponding to a kinetic energy of 5 eV, which is twice the minimum or Bohm energy, \( kT_e/2 \), needed to obtain a stable sheath. Electrons with \( kT_e = 5 \text{ eV} \) are co-injected with the ions. The timestep, \( \Delta t_{ke} = 1.5 \times 10^{-13} \text{ s} \), was chosen such that \( \omega_{pe} \Delta t_{ke} \approx 0.027 \), a factor of 75 smaller than the stability limit \( \omega_{pe} \Delta t < 2 \) (see Fig. 2) and such that \( v_{the} \Delta t/\Delta x \approx 0.1 \), where \( v_{the} \) is the electron thermal velocity and \( \Delta x \) is the spatial cell size (thus, a thermal electron travels one cell length in about 10 timesteps). This timestep is probably smaller than needed, but we have not investigated this question here. (Ref. [1] examined a case with \( \omega_{pe} \Delta t \approx 0.1 \), \( v_{the} \Delta t/\Delta x \approx 0.4 \), and coarser spatial resolution, \( \Delta x = \lambda_{D0} \), obtaining results similar to, but noisier than, the Reference case.) The cell size is \( \Delta x \approx \lambda_{D0}/4 \), and the length of the simulation region is \( 15 \lambda_{D0} \). In steady state, a quasi-neutral region is established over roughly the right 2/3 of the simulation region and a non-neutral sheath over the left 1/3. The electric field in the sheath region is such as to accelerate ions to the left while retarding electrons.

We use this same one-dimensional geometry to test the Boltzmann-electron model. For these calculations, the ions are specified in exactly the same way as in Ref. [1], but there are no electron particles: the electron gas is completely specified by a reference density \( n_{e0} \), reference potential \( \phi_0 \), and an isotropic temperature \( T_e \) (see Eq. 5). For the Boltzmann-
**Figure 3.** Figure 5 of Ref. [1], comparing results from a 1D, fully-kinetic simulation of a plasma sheath to an analytic model. Plot (a) shows the electric potential and (b) shows the ion and electron densities. Plasma is injected from the right boundary.
electron model, $\omega_{pe} \Delta t < 2$ is no longer a constraint on the timestep. The ion orbits need to be integrated accurately through the sheath electric field, so we require $v_i \Delta t_{be} < \Delta x$, where $v_i$ is the ion velocity, $\Delta t_{be}$ is the Boltzmann-electron timestep, and $\Delta x$ is sufficiently small to resolve the spatial variation of the sheath field, i.e., $\Delta x \ll \lambda_{D0}$. These two conditions give the constraints on the timestep and mesh size: $\Delta t_{be} < \Delta x/v_i \ll \lambda_{D0}/v_i$. We use the same mesh size as the Reference case in [1], i.e., $\Delta x = \lambda_{D0}/4$. The condition $\Delta t_{be} < \Delta x/v_i$ then allows us to go up to $\Delta t_{be} = 280\Delta t_{ke} = 4.2 \times 10^{-11}$ s.

We ran three simulations, with $\Delta t_{ke} = 10\Delta t_{ke}$, $100\Delta t_{ke}$, and $1000\Delta t_{ke}$. The first two values are plotted in Fig. 2. The last value exceeds the condition $\Delta t_{be} < \Delta x/v_i$ given above by a factor of about 3.6, but this condition has to do with accuracy rather than stability. Results for the ion density and electric potential using $\Delta t_{be} = 10\Delta t_{ke}$ are shown in Fig. 4. The agreement with the analytic results is very good.

The results for all three Boltzmann-electron calculations are compared in Fig. 5. There is little difference between the ion densities and energies in the $10\Delta t_{ke}$ and $100\Delta t_{ke}$ calculations. The ion energies show the correct behavior, increasing from the injected value of 5 eV to 20 eV as they traverse the sheath. At $1000\Delta t_{ke}$, an injected ion is traveling about 3.6 cells per timestep and the density is clearly noisier than at the smaller timesteps. Even under these conditions, the ion energies are reasonably close to the analytic values.

In general, the maximum ratio between the timesteps for a Boltzmann-electron and a kinetic-electron simulation is given by

$$\frac{\Delta t_{be}}{\Delta t_{ke}} = f \sqrt{\frac{M_i}{2m_e}} \sqrt{\frac{kT_e}{Z_i eV_i}} \approx 30f \sqrt{\frac{A_i}{Z_i}} \sqrt{\frac{T_e(eV)}{V_i(V)}}$$

(17)

where $f$ is a factor that depends on how well the electron orbit-accuracy constraint $\Delta t_{ke} < \Delta x/v_{the}$ is fulfilled in the kinetic-electron simulation, viz., $f = \Delta x/v_{the}\Delta t_{ke}$. Here, $M_i$ is the ion mass, $Z_i eV_i$ is the ion kinetic energy, and $A_i$ is the ion mass in AMU. For the Reference parameters in [1], $Z_i = 1 = A_i$, $T_e = 5$ eV, $V_i = 5$ V, and $f = 9.3$. Equation 17 then gives a maximum Boltzmann-electron timestep of $\Delta t_{be} \approx 280\Delta t_{ke}$, as we found above. At this timestep, the ions travel about 1 cell per timestep.
Figure 4. Results for (a) ion density and (b) electric potential, comparing Boltzmann-electron model with analytic model. The simulation used a timestep 10 times larger than using to generate Fig. Ions are injected from the right boundary.
Figure 5. Comparison of three Boltzmann-electron calculations with $\Delta t = 10$ (red), 100 (blue), and 1000 (green) times the timestep used to generate Fig. 3. Plot (a) compares the ion density, (b) compares the electric potential and (c) compares the ion energies. Ions are injected from the right boundary with kinetic energy of 5 eV.
The Boltzmann-electron model avoids two constraints that often determine the maximum timestep and mesh-size one can use in explicit, fully-kinetic, electrostatic plasma simulations, namely, $\omega_{pe}\Delta t < 2$ (for stable electron oscillations) and $\Delta x/\lambda_D < 1$ (to avoid excessive electron heating). The ambipolar-field model also avoids these constraints, but assumes quasineutrality, which the Boltzmann-electron model does not. Thus, the Boltzmann-electron model can be applied to the problem of ion extraction from a plasma, where ions transition from a quasi-neutral to a completely non-neutral environment. The main computational hurdle in implementing the Boltzmann-electron model involves solving a Poisson-Boltzmann equation, which is very nonlinear in the sheath region at a plasma-wall boundary and at an ion-extraction meniscus. We have successfully applied a nonlinear solver, developed for the Trilinos project, to this equation. We have shown that the Boltzmann-electron model implemented in Aleph agrees with the analytic model for a 1D collisionless sheath. To demonstrate this, we used an input file and geometry used in a previous verification study of the fully-kinetic algorithms in Aleph. We ran the Boltzmann-electron algorithm with $\Delta t_{be}/\Delta t_{ke} =$10, 100, and 1000, where $\Delta t_{ke}$ is the timestep used in the “Reference” case of the fully-kinetic study. The latter two timesteps violate the $\omega_{pe}\Delta t < 2$ electron-stability constraint and are nevertheless stable (as expected) and give good agreement with analytic and fully-kinetic simulation results.

For these calculations, we used the same mesh size, $\Delta x = \lambda_{D0}/4$, as in the fully-kinetic calculation. In any calculation one must of course resolve the relevant length-scale, which in the sheath region is the Debye-length. Outside the sheath region, the Boltzmann-electron model should be able to use a mesh-size larger than $\lambda_{D0}$ with minimal effect on the results, but we have not yet verified this.

There are some downsides to the Boltzmann-electron model. As in the ambipolar-field model, the electron temperature is an input variable and not calculated by the code. The nonlinear Poisson-Boltzmann equation requires many more numerical operations to solve than the linear Poisson equation in the fully-kinetic model. For this reason, in the short, single-processor calculations for this study, the Boltzmann-electron calculation with $\Delta t_{be}/\Delta t_{ke} =$10 took about the same time ($\approx 60$ s) as the fully-kinetic case. However, the $\Delta t_{be}/\Delta t_{ke} =$100, 1000 cases were considerably faster than the fully-kinetic cases ($\approx 9$ s and $\approx 1.2$ s, respectively).

In Ref. [7], which applied the ambipolar-field model in Aleph to plasma expansion, we noted that the electric field was very noisy and produced a large energy-spread in the ions. By comparison, there is very little noise evident in the ion distribution in Fig. 5(c). As discussed following Eq. 16 above, the ambipolar-field expression is recovered in the long-wavelength limit of the Poisson-Boltzmann equation. It may be possible to improve on the results in Ref. [7] by using Eq. 16 instead of the ambipolar equation. In addition, it may be possible to take advantage of the fact that, in many applications, the region where the Poisson-Boltzmann equation is strongly nonlinear is a small fraction of the total volume.
In the coming year, we plan to apply the Boltzmann-electron algorithm to calculations of practical interest that have large 3D meshes and run on hundreds of parallel processors.
References


A Cubit input

The mesh for the calculations Figs. 4 and 5 was generated with Cubit 14.1. The following mesh information was written by Cubit:

Executive Exodus summary:
- Number of dimensions = 2
- Number of element blocks = 1
- Number of sidesets = 0
- Number of nodesets = 2
- Number of bc sets = 1
- Number of elements = 60
- Number of nodes = 61

The Cubit input file (courtesy of J. Boerner) is listed below:

```
1  # Mesh transform factor = {f = 5.229e-6} so 1 unit in Cubit becomes
1  # 1 Debye length (5.229e-6 m) in output mesh using SI units.
1  # Sheath length = {d_gap = 15} in Debye lengths
1  # Cell size dx = {dx = 1/4} in Debye lengths

1  #------ Geometry ------#
1  create curve 0 0 0 direction 1 0 0 length {d_gap}
1  curve 1 size {dx}
1  mesh curve 1
1 10  nodeset 1 vertex 1
1  nodeset 2 vertex 2
1  transform mesh output reset
1  transform mesh output scale {f}
1  export mesh {"'domain_" // tostring(d_gap) // "_LD_in_" // tostring(d_gap/dx) //
1  "_elems.e" "} dimension 2 overwrite
1  exit
```
B Aleph input

The calculations in Figs. 4 and 5 were carried out with Aleph version 4607. Aleph was run using the command:

```
./aleph boltz_sheath_ndx_060_dt_1.50e-12_wt_0.50e12.in >&alog10&
```

The following is the input file (based on kinetic-electron input from J. Boerner):

```
1 # BASIC CONTROL
sensitivity level = UUR
units = SI
timestep size = 1.50e-12
total number of timesteps = 5000
input mesh file name = ../../cubit/1d/domain_15_LD_in_60_elem.e

# FIELD SOLVER
apply electrostatic_ions_boltzmann_electrons model to block_1, \nwindow = discrete, size = 1, stride = 1, \nTe = 58022.5, n0 = 1.01e19, V0 = 0.0

# INACTIVE STRIDES
interaction stride = off
reweighting stride = off
restore energy stride = off
restart stride = off
rebalance stride = off

20 # PARTICLES
particle weighting for H+ = 0.50e12, reweighting_method = none

# BOUNDARY CONDITIONS
# Wall
BC for voltage on nodelist_1 is dirichlet V = -15.0
BC for particles on surface_1 is outflux for default

# Ambient plasma
BC for voltage on nodelist_2 is dirichlet V = 0.0

30 # 5 eV kinetic energy, 10 degrees K
BC for particles on surface_2 is influx of H+ with density = 1.01e19, \n vx = -30950.0, T = 10.0
BC for particles on surface_2 is outflux for default

# EXODUS OUTPUTS
exodus output stride = 200
exodus output file name = ./Results/boltz_sheath_ndx_060_dt_1.50e-12_wt_0.50e12.e

output elemental volumes, name = vols, window = discrete, size = 1
output nodal density for H+, name = n_N(Hp), window = discrete, size = 1
output elemental density for H+, name = n(Hp), window = discrete, size = 1
output elemental average_velocity for H+, name = v(Hp), \n window = discrete, size = 1
output elemental temperature for H+, name = T(Hp), window = discrete, \n size = 1
output elemental computational_count for H+, name = C(Hp), \n window = discrete, size = 1
output elemental charge_density, name = rho, window = discrete, size = 1
```
output solver charge density = true, window = discrete, size = 1

particle dump file name = ./Results/p-10dt.h5part, file type=h5part, \ attributes = local_id, type, x, y, z, vx, vy, vz, stride=200, \ fraction = 1

# STATUS REPORT OUTPUTS
output global computational_count for H+, name = C_H+, \ precision = 3, window = discrete, size = 1, stride = 20
output global current for all, on surface_1, name = I_inlet, \ precision = 9, window = discrete, size = 1, stride = 20
output global current for H+, on surface_1, name = I_inlet_H+, \ precision = 9, window = discrete, size = 1, stride = 20
output global current for all, on surface_2, name = I_wall, \ precision = 9, window = discrete, size = 1, stride = 20
output global current for H+, on surface_2, name = I_wall_H+, \ precision = 9, window = discrete, size = 1, stride = 20
status report name = stdout, stride = 20, columns = time_stamp, \ elapsed_time, timestep, C_H+

status report \ name = ./Results/boltz_sheath_ndx_060_dt_1.50e-12_wt_0.50e12.csv \ stride = 20, format = paraview, \ columns = simulation_time, C_H+, I_inlet, I_inlet_H+, I_wall, I_wall_H+

See also the files in the Aleph repository in directory examples/boltzmann_sheath_report.
DISTRIBUTION:

<table>
<thead>
<tr>
<th>MS</th>
<th>Author</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>0346</td>
<td>A. Grillet</td>
<td>1512</td>
</tr>
<tr>
<td>0828</td>
<td>A. Geller</td>
<td>1516</td>
</tr>
<tr>
<td>0836</td>
<td>J. Boerner</td>
<td>1516</td>
</tr>
<tr>
<td>0840</td>
<td>D. Rader</td>
<td>1513</td>
</tr>
<tr>
<td>0869</td>
<td>C. Renschler</td>
<td>2730</td>
</tr>
<tr>
<td>0878</td>
<td>S. Dirk</td>
<td>2735</td>
</tr>
<tr>
<td>0878</td>
<td>J. Howard</td>
<td>2735</td>
</tr>
<tr>
<td>0878</td>
<td>T. Hughes</td>
<td>2735</td>
</tr>
<tr>
<td>0878</td>
<td>G. Laity</td>
<td>2735</td>
</tr>
<tr>
<td>0878</td>
<td>M. Lovejoy</td>
<td>2735</td>
</tr>
<tr>
<td>1168</td>
<td>M. Bettencourt</td>
<td>1352</td>
</tr>
<tr>
<td>1168</td>
<td>K. Cartwright</td>
<td>1352</td>
</tr>
<tr>
<td>1168</td>
<td>C. Moore</td>
<td>1352</td>
</tr>
<tr>
<td>1177</td>
<td>L. Musson</td>
<td>1355</td>
</tr>
<tr>
<td>1318</td>
<td>S. Moore</td>
<td>1444</td>
</tr>
<tr>
<td>1318</td>
<td>R. Hooper</td>
<td>1446</td>
</tr>
<tr>
<td>1322</td>
<td>P. Crozier</td>
<td>1444</td>
</tr>
<tr>
<td>1423</td>
<td>M. Hopkins</td>
<td>1118</td>
</tr>
<tr>
<td>1423</td>
<td>E. Barnat</td>
<td>1118</td>
</tr>
<tr>
<td>1423</td>
<td>P. Miller</td>
<td>1118</td>
</tr>
<tr>
<td>0899</td>
<td>Technical Library</td>
<td>9536 (electronic copy)</td>
</tr>
</tbody>
</table>