DTIC COPY

SPACECRAFT POTENTIAL CONTROL

V. A. Davis
B. M. Gardner
G. A. Jungeward
M. J. Mandell

Science Applications International Corporation
10260 Campus Point Drive
San Diego, CA 92121

July 2004

Final Report

AIR FORCE RESEARCH LABORATORY
Space Vehicles Directorate
29 Randolph Rd
AIR FORCE MATERIEL COMMAND
Hanscom AFB, MA 01731-3010

This technical report has been reviewed and is approved for publication.

/ signed /   /signed /

David Cooke   Katharine Kadinsky-Cade, Acting Chief
Contract Manager   Space Weather Center of Excellence

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE (DD-MM-YYYY) July 2004	2. REPORT TYPE Final Technical Report	3. DATES COVERED (From - To)	
4. TITLE AND SUBTITLE Spacecraft Potential Control		**5a. CONTRACT NUMBER** F19628-98-C-0074	
		5b. GRANT NUMBER	
		5c. PROGRAM ELEMENT NUMBER 63410F	
6. AUTHOR(S) V. A. Davis B.M. Gardner G.A. Jongeward M.J. Mandell		**5d. PROJECT NUMBER** 2822	
		5e. TASK NUMBER GC	
		5f. WORK UNIT NUMBER MX	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Science Applications International Corporation 10260 Campus Point Drive, MS A-1A San Diego, CA 92121		**8. PERFORMING ORGANIZATION REPORT NUMBER** SAIC 04/2040	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory Space Vehicles Directorate 29 Randolph Road Hanscom AFB, MA 01731-3010		**10. SPONSOR/MONITOR'S ACRONYM(S)**	
		11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-VS-HA-TR-2004-1152	

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The objective of this contract is to develop improved numeric algorithms for the computation of spacecraft charging on Earth-orbiting spacecraft. This work is part of the Nascap-2k program, which is a joint program with the Space Environment Effects (SEE) Program at NASA/MSFC. The end result of the program is a user friendly computer code that computes spacecraft charging in dense and tenuous plasma environments. The primary focus of the SEE Program's contribution was funding development of the graphical user interface and user documentation, as well as a related program to measure relevant material properties.

15. SUBJECT TERMS

C/NOFS	DynaPAC	Nascap 2-K	Potentials	Space environment
Spacecraft	Spacecraft charging			

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON David Cooke
a. REPORT UNCLASS	**b. ABSTRACT** UNCLASS	**c. THIS PAGE** UNCLASS	SAR		**19b. TELEPHONE NUMBER** (Include area code) (781) 377-2931

CONTENTS

FIGURES

TABLES

1. INTRODUCTION

The objective of this contract is to develop improved numeric algorithms for the computation of spacecraft charging on Earth-orbiting spacecraft. This work is part of the *Nascap-2k* program, which is a joint program with the Space Environment Effects (SEE) program at NASA/MSFC. The end result of the program is a user friendly computer code that computes spacecraft charging in dense and tenuous plasma environments. The primary focus of the SEE program's contribution was funding development of the graphical user interface and user documentation, as well as a related program to measure relevant material properties.

Nascap-2k is a spacecraft charging and plasma interactions code designed to be used by spacecraft designers, aerospace and materials engineers, and space plasma environments experts to study the effects of both the natural and spacecraft-generated plasma environment on spacecraft systems. Survival in the plasma environment is a concern for virtually all Earth orbiting satellites, be they in low-Earth orbit (LEO), geostationary orbit (GEO), polar or other Earth orbit, as well as for interplanetary missions. Increased power requirements have pushed spacecraft subsystem design parameters, such as solar array voltage and power, to higher values than ever before, while demand for resources, especially in the commercial telecommunications industry, results in the need for longer mission lifetimes. Additionally, electric propulsion, which is critical to the success of many exploratory and commercial missions, produces a high-energy, high-density plasma in which can cause serious erosion and contamination problems for spacecraft surface coatings and for sensitive instruments.

NASCAP/GEO[1,2,3] (NASA Charging Analyzer Program for GEosynchronous Orbit) was the standard tool for the computation of spacecraft charging in tenuous plasmas for more than two decades. The fully three-dimensional computer codes *NASCAP/LEO*[4,5] (NASA Charging Analyzer Program for Low-Earth Orbit), *POLAR*[6] (Potentials Of Large objects in the Auroral Region), and *DynaPAC*[7,8] (Dynamic Plasma Analysis Code) were developed to address various other spacecraft-plasma interactions issues. Computer modeling of flight experiments (such as SCATHA[2,3], the SPEAR[9,10] series and CHAWS[11]) demonstrated excellent ability to predict both steady-state and dynamic interactions between high-voltage spacecraft and the ambient plasma.While each of these codes works well for the range of problems for which it was designed, by today's standards these codes are awkward to use and require expertise to be used properly. In addition, *NASCAP/GEO* and *POLAR* have highly restrictive geometrical capabilities.

Nascap-2k builds on the capabilities of the older codes, giving the spacecraft designer much-improved modeling capabilities by taking advantage of a greater understanding of the pertinent phenomena, employing more advanced algorithms, and implementing a state-of-the-art user interface, including three-dimensional post-processing graphics. The surface charging physical models developed for *NASCAP/GEO* have been incorporated in a boundary element model (BEM), which permits implicit treatment of electric fields. The *DynaPAC* code has been incorporated within the *Nascap-2k* framework to treat potentials and plasmas external to the spacecraft. With the incorporation of *DynaPAC*, *Nascap-2k* features arbitrarily nested grids to provide good spatial resolution, strictly continuous electric fields for accurate particle tracking, models for sheath and wake structures, and the ability to do particle-in-cell (PIC) calculations.

Nascap-2k is an interactive toolkit for studying plasma interactions with realistic spacecraft in three dimensions. The *Nascap-2k* interface employs an index-tab metaphor. Several of these tabs can be seen in Figure 1. The graphical user interface is designed to help less experienced users easily solve moderately complex plasma interactions problems. *Nascap-2k* also enables plasma interactions specialists to perform realistic analyses with direct application to engineering problems. The core capabilities of *Nascap-2k* are:

1. Define spacecraft surfaces and geometry and the structure of the computational space surrounding the spacecraft;
2. Solve for time-dependent potentials on spacecraft surfaces;
3. Solve the electrostatic potential around the object, with flexible boundary conditions on the object and with space-charge computed either fully by particles, fully analytically, or in a hybrid manner;
4. Generate, track, and otherwise process particles of various species, represented as macroparticles in the computational space; and
5. View surface potentials, space potentials, particle trajectories, and time-dependent potentials and currents.

An overview of Nascap-2k structure is shown in Figure 2. There are three main programs: *Nascap-2k*, *Object ToolKit*, and *GridTool*.
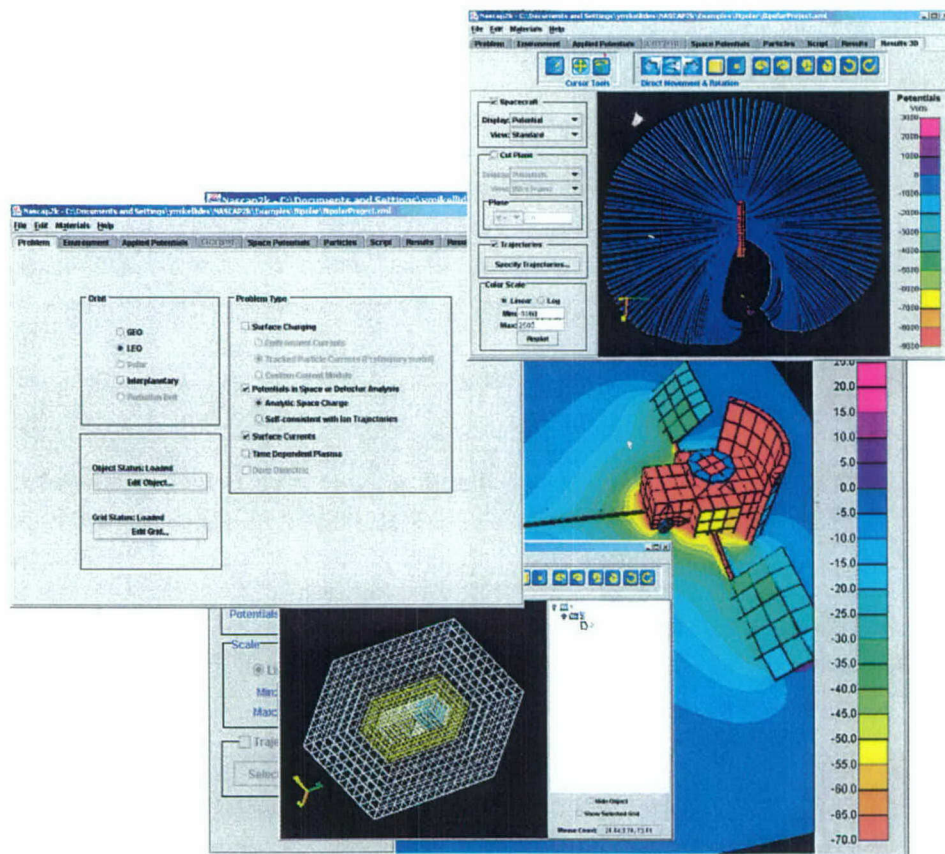
Figure 1. Selected Views of the *Nascap-2k* User Interface.
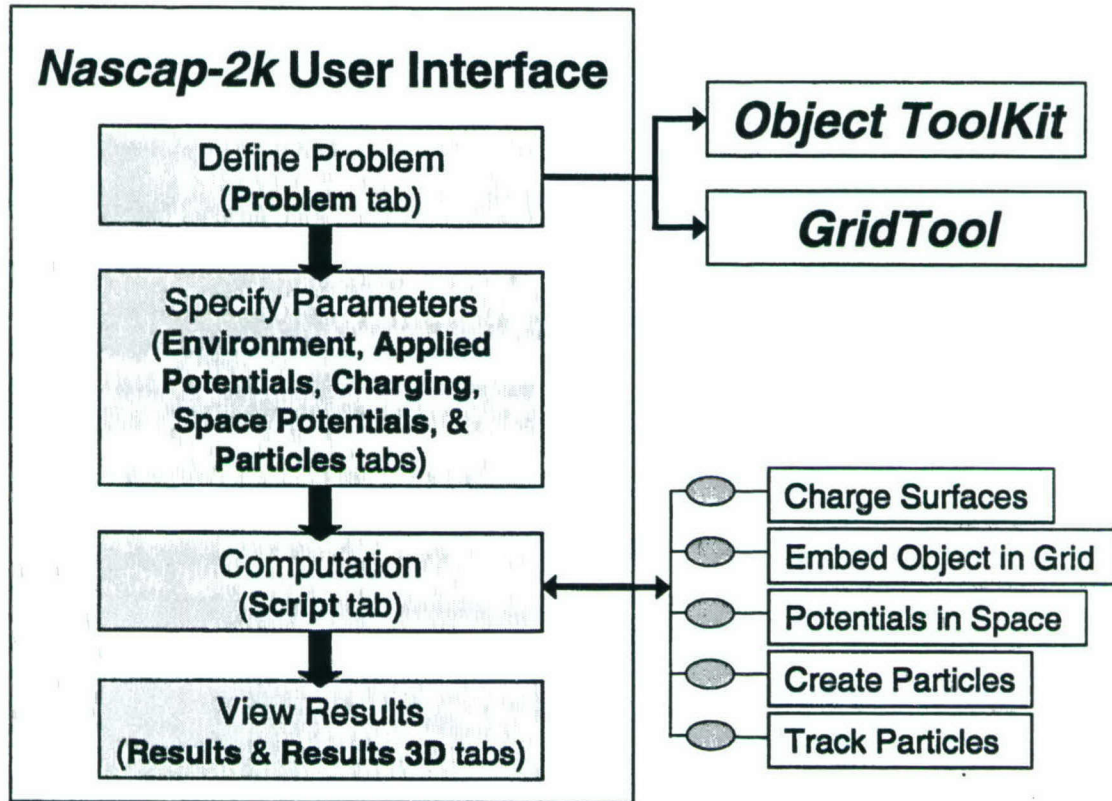
Figure 2. *Nascap-2k* **Structure.**

Object ToolKit is a three dimensional object generator tailor-made for spacecraft modeling. It is used to create finite-element representations of spacecraft surfaces for *Nascap-2k* (and other environmental interactions computer codes, such as *EPIC*.). It also has materials editing capability and can import objects from other standard finite-element preprocessors such as *PATRAN*. In this way the spacecraft geometry can be realistically represented and existing finite element models of spacecraft constructed for other purposes can be adapted for use in *Nascap-2k*. *Object ToolKit* Output (in XML) contains the recipe for recreating/reassembling the object, object definition by nodes and surface elements, and material definitions.

The computational space around the spacecraft is gridded interactively using the *GridTool* module. Arbitrarily nested subdivision allows resolution of important object features while including a large amount of space around the spacecraft.

The main *Nascap-2k* user interface uses an index-tab metaphor, and contains tabs for problem selection, initial conditions, parameter specification, script writing, time-dependent results analysis, and two- and three-dimensional display of surface potentials and fields.

Nascap-2k calculates surface charging in Geosynchronous Earth Orbit, in the Solar Wind, or in other tenuous plasma environments using the Boundary Element Method (BEM)[12], which permits implicit treatment of electric fields.

Nascap-2k uses a high-order, finite-element representation for the electrostatic potential that ensures electric fields are strictly continuous throughout space. The electrostatic potential solver, originally developed for *DynaPAC*, uses a conjugate gradient technique to solve for the potentials and fields on the spacecraft surface and through the surrounding space. Space charge density models presently include Laplacian, Linear, Non-linear, Frozen Ions, Full Trajectory Ions, Full PIC (Particle in Cell), and Hybrid PIC (appropriate to the several microsecond timescale response to a negative pulse).

Particle tracking is used to study sheath currents, to study detector response, or to generate space charge evolution for dynamic calculations. *Nascap-2k* generates macroparticles (each of which represents a collection of particles) either at a "sheath boundary", the problem boundary, or throughout all space. Particles are tracked for a specified amount of time, with the timestep automatically subdivided at each step of each particle to maintain accuracy. The current to each surface of the spacecraft is recorded for further processing.

The **Results** tab of the *Nascap-2k* user interface displays generate time histories of potentials and surface currents. The **Results 3D** tab displays object surface potentials, space potentials, particle positions, and/or particle trajectories. Contour levels and other plotting attributes are modified through the user interface.

The computational modules of *Nascap-2k* employ a database manager. This database manager is a library of routines capable of making large arrays of information contained in disk files accessible to computational modules. It has a programmer-friendly language for defining data types and for retrieving and storing data. This strategy enables *Nascap-2k* to be operable on, and portable among, modern high-power workstations, which have proven to be more cost-effective than supercomputers for this type of code development and analysis.

The user interface is written in Java, the sciences modules are in C++ and Fortran, and the utility routines are written in C. All information is stored in the multi-file database inherited from the *DynaPAC* code or as XML. The modules communicate using XML files, keyword text input files, direct subroutine calls (DLL import/export), JNI subroutine calls, and a proprietary database. XML files and text input files can be manually edited with a text editor or XML editor.

Nascap-2k inherits the core computational modules and database from the most modern of these codes, DynaPAC. The DynaPAC computational modules were converted to DLLs (dynamic link libraries) to run within *Nascap-2k*. In addition, the following modules were developed for *NASCAP-2k*:

1. *BEMDLL* (written in C++) performs the Boundary Element Method analysis. It reads the object definition output file (XML), converts the object information to the DynaPAC structure, and stores it in the database. It exports standard methods and JNI methods (called by the Java Applications to request calculations and retrieve results). It uses the Boundary Element Method[12] (BEM) for calculating surface charging in Geosynchronous Earth Orbit (GEO), in the Solar Wind, or in other tenuous plasma environments. It can also use surface currents computed by the Tracker module to compute surface charging for dense plasma

environments. Discussion of the algorithms appears in the software documentation, Scientific Report #2 for this contract.

2. *Object Toolkit* (written in Java) is used to create finite-element representations of spacecraft surfaces. It also has materials editing capability, and can import objects in PATRAN neutral file format. It can also import objects from the DynaPAC database. Output (in XML) contains the recipe for recreating/reassembling the object, object definition by nodes and elements, and material definitions.

3. *GridTool* (written in Java) is used to define gridding of the space surrounding the spacecraft. (This module supersedes an older DynaPAC module.)

4. *Nascap2K GUI* (written in Java) is the main user interface for *NASCAP-2K*. It is based on an index-tab metaphor, and contains tabs for problem selection, initial conditions, environment specification, runscript creation and execution, "TermTalk-like" results analysis, and three-dimensional display of surface potentials and fields. It creates and runs default and user specified scripts, and a "project file" (XML) to save its state. This module was developed under contract with NASA and extended under this contract.

5. *DynaBase* (Fortran Windows DLL) is the C++ callable gateway to the DynaPAC database.

6. *Lapack* (C++ Windows DLL) is a custom implementation of matrix solver/inverter programs needed by *NASCAP-2K*.

We implemented the capabilities of the POLAR code in the *Nascap-2k* GUI and in the BEM charging module. The auroral charging model and its validation is reported in Scientific Report #3 for this contract.

In addition to writing the software, we prepared documentation of *Nascap-2k*. It appears as Scientific Technical Report #2 for this contract.

Since Scientific Technical Report #2 was published, we wrote documentation for a general *Nascap-2k* user describing how to create and use a custom DLL to model surface currents using models other than the *Nascap-2k* standard one. This documentation is included as Section 1 of this report. We cleaned up the template for a custom current DLL and added it to the *Nascap-2k* install.

We validated and documented the disjoint grid capability of *Nascap-2k*. This capability is intended for modeling objects which are disjoint and distant but electrically connected. A description of the validation and the documentation are included as Section 3 of this report. To use this capability, the CombineGrids Java application is needed. This application is also included in the *Nascap-2k* install.

The software is being delivered for both the Win32 platform (supporting Windows 2000, Windows XP Home Edition, and Windows XP Professional Edition) and for the LINUX platform. The Windows version is fully tested, and the LINUX version has undergone only limited testing. Some notes regarding the port to LINUX are included as Section 1.
We prepared a written description of how the charge stabilization algorithm is implemented in *Nascap-2k*. This description appears as Section 5.

5

We performed C/NOFS calculations using *Nascap-2k* under contract to Spectrum Astro. As these calculations are of more general interest, we wrote a summary of the non-obvious, undocumented features of *Object Toolkit* and *Nascap-2k* used to create the geometric model and run surface potential calculations for the C/NOFS spacecraft. This summary is included in Section 1 of this report.

We simulated the electron dynamics in the sheath of a VLF antenna. We estimated the sheath size and did one-dimensional calculations for both sine wave and square wave excitation. The results show strong electrostatic plasma oscillations at the sheath edge. We used *Nascap-2k* to duplicate the square wave results through the first maximum in the plasma oscillation, obtaining excellent agreement with the one-dimensional results. This opens the door to fully three-dimensional dynamic VLF antenna calculations. The results were presented at the Spacecraft Charging Technology Conference (October 2003). A description of these calculations is included as Section 7 of this report.

Under direction of Dr. David Cooke, we supported the STEREO mission by doing spacecraft charging calculations. This provided an opportunity to test and expand the capabilities of the *Nascap-2k* computer code and its algorithms. This work is described in a presentation made at the STEREO/Impact SWG Meeting in Berkeley, CA. This presentation is included here as Section 8 below.

In addition, this contract supported development of a Rapid Alert Charging Tool. This work is described in 1 below and in the publication I. Katz, V. A. Davis, M. J. Mandell, D. L. Cooke, R. Hilmer, L. Habash Krause, *Forecasting Satellite Charging: Combining Space Weather and Spacecraft Charging*, AIAA 2000-0369.

The scientists and other researchers who contributed to this work are as follows: Dr. Myron. J. Mandell, Dr. Ira Katz, Mr. Jeffery M. Hilton, Dr. Victoria A. Davis, Mr. David Monjo, Mr. Dale Lovell, and Ms. Barbara M. Gardner.

This contract is a follow-on to work performed under earlier contracts F19628-91-C-0187, Space System-Environment Interactions Investigation, F19628-93-C-0050 Modeling and Post Mission Data Analysis, and F19628-89-C-0032 Analysis of Dynamical Plasma Interactions with High Voltage Spacecraft. NASA supported related work under contracts NAS8-98220 and NAS8-02028.

The following publications were supported in total or in part by this contract:

M. J. Mandell, I. Katz, D. L. Cooke, *Towards a more robust spacecraft charging algorithm*, AIAA Paper AIAA-99-0379, presented at the 1999 Aerospace Sciences Meeting in Reno, NV.

I. Katz, V. A. Davis, M. J. Mandell, D. L. Cooke, R. Hilmer, L. Habash Krause, *Forecasting satellite charging: Combining space weather and spacecraft charging*, AIAA Paper AIAA-2000-0369, presented at the 2000 Aerospace Sciences Meeting in Reno, NV.

M.J. Mandell, I. Katz, J.M. Hilton, J. Minor, D.L. Cooke, *NASCAP-2K--A spacecraft charging analysis code for the 21st century,* AIAA Paper AIAA-2001-0957, presented at the 2001 Aerospace Sciences Meeting in Reno, NV.

M.J. Mandell, I. Katz, J.M. Hilton, D.L. Cooke, J. Minor, 2001, *Nascap-2k* spacecraft charging models: Algorithms and applications, *Proceedings of the 7th Spacecraft Charging Technology Conference,* ESA SP-476, p. 499.

V.A. Davis, L.F. Neergaard, M.J. Mandell, I. Katz, B.M. Gardner, J. M. Hilton, J. Minor, *Spacecraft charging calculations: Nascap-2k and SEE spacecraft charging handbook,* AIAA Paper AIAA-2002-0626, presented at the 2002 Aerospace Sciences Meeting in Reno, NV.

M.J. Mandell, D.L. Cooke, V.A. Davis, G.A. Jongeward, B.M. Gardner, R.A Hilmer, K.P. Ray S.T. Lai, L.H. Krause, 2002, *Nascap-2k* calculations of spacecraft charging on interplanetary spacecraft *Proceedings of 34th COSPAR Scientific Assembly and the 2nd World Space Congress.*

M.J. Mandell, V.A. Davis, B.M. Gardner, I.G. Mikellides, D.L. Cooke, J. Minor, 2003, *Nascap-2k* an overview, *Proceedings of the 8th Spacecraft Charging Technology Conference,* NASA/CP-2004-213091.

V.A. Davis, M.J. Mandell, B.M. Gardner, I.G. Mikellides, L.F. Neergaard, D.L. Cooke, J. Minor, 2003, Validation of *Nascap-2k* spacecraft-environment interactions calculations, *Proceedings of the 8th Spacecraft Charging Technology Conference,* NASA/CP-2004-213091.

M.J. Mandell & D.L. Cooke, 2003, *Nascap-2k* as a PIC code, *Proceedings of the 8th Spacecraft Charging Technology Conference,* NASA/CP-2004-213091.

M.J. Mandell & D.L. Cooke, *Nascap-2k simulations of spacecraft charging in tenuous plasma environments,* AIAA Paper AIAA-2004-0986, presented at the 2004 Aerospace Sciences Meeting in Reno, NV.

Three Scientific Reports were prepared under this contract.

M.J. Mandell, V.A. Davis, B.M. Gardner, J.M. Hilton, I. Katz, *Spacecraft potential control,* Scientific Report No. 1, AFRL-VS-TR-2001-1619, 2001.

M.J. Mandell, V.A. Davis, I.G. Mikellides, *Nascap-2k preliminary documentation,* Scientific Report No. 2, AFRL-VS-TR-2002-1676, 2002.

V.A. Davis, M.J. Mandell, G.A. Jongeward, *Computing surface charging in the auroral environment using Nascap-2k,* Scientific Report No. 3, AFRL-VS-TR-2003-1606, 2003.

2. TEMPLATE FOR *NASCAP-2K* CUSTOM CURRENT DLL

2.1 Purpose

The purpose of a "Custom Current" DLL is to calculate currents to surfaces in a manner the analyst considers more appropriate to his problem than the formulations built in to *Nascap-2k*'s surface charging DLL, BEMDLL. (Naturally, the analyst feels that his or her formulation is worthy of incorporation into the standard DLL, but that topic is beyond the scope of our discussion here.) For example, the template contains the "EWB Plate" formulation, which is appropriate to a low-Earth orbiting spacecraft with no highly biased surfaces, and takes into account ram ions and wake effects.

2.2 Mechanics of Use

The custom current DLL is loaded dynamically by BEMDLL, which expects to find the two entry points described below. The analyst assigns the DLL an appropriate filename and places it where Windows can find it (in Windows, Windows\System32, or in a directory contained in the PATH environment variable). The filename is specified to BEMDLL in the SetCustomCurrentDLL script item.

2.3 Template

The Template contains the C++ and project files needed to create a custom current DLL. The supported programming environment is Microsoft Visual Studio 6 with Service Pack 5.

2.4 Entry Points

The DLL contains two entry points, which are called by BEMDLL.

> CALLBACK setEnvironmentParams(double* den, double* te, double* ti, Vector3* objvel, double* ionamu)

is called once each timestep to set the calculation parameters, which are:

- den – the first electron density of a GEO environment;
- te – the first electron temperature of a GEO environment;
- ti – the first ion temperature of a GEO environment;
- objvel – the spacecraft velocity;
- ionamu – the ion atomic mass, currently hardwired to be 16.

The analyst may hardwire additional parameters into the DLL, read parameters from a file, or otherwise obtain additional parameters.

> CALLBACK getCustomCurrent(element* elem, double* I0, double* I1)

is called for every surface element at each timestep. The element structure's public member variables (listed below) are accessible to the analyst. Element public methods are NOT

8

accessible, as the source code is not provided. (The header file element.h must be identical to the file used in building BEMDLL.) The analyst is responsible for calculating and returning I0 – the cell current divided by ε_0 (=JA/ ε_0, units of Volt-meter/second), and I1 – the derivative of I0 with respect to surface potential. (Note that I1 must be non-positive.)

2.5   The Vector3 Class

The Vector3 class, implemented in Vector3.cpp, is used to encapsulate three-vectors and numerous useful methods. The analyst can easily discover these methods through inspection of the code (provided). Note that many of the Vector3 methods return pointers to new Vector3 objects; it is the analyst's responsibility to delete these objects.

2.6   Other Files

The Include directory contains several include files, of which the most noteworthy is element.h, whose properties are described below. The file derf.cpp is used in the "EWB Plate" formulation, and is not generally required.

2.7   Element Properties

The following element properties in Table 1 are accessible to the analyst via the elem->*propertyname* construct:

Table 1. Element Properties.

Type	Variable Name	Description
double	area	Cell area (m^2)
double	capacitance	Cell Capacitance/ε_0 (meters)
Vector3*	center	Location of cell center (meters)
conductor*	conduc	Structure with info about associated conductor
int	conductorIndex	Index of the associated conductor
double	field	Normal component of electric field (V m^{-1})
int	index	The fortran-style index of the cell
double	initialcurrent	Current at the beginning of the timestep
double	initialPotential	Cell potential to be used in SetInitialPotentials (V)
char	materialName[32]	Name of the cell's material
material*	matl	Pointer to the associated material structure
double	maxpotential	The maximum potential on the object (V)
double	minpotential	The minimum potential on the object (V)
element*	next	Pointer to the next element to be iterated over
node*	nodes[4]	An array of pointers to the four nodes in counterclockwise order
Vector3*	normal	The unit outward normal to the cell
double	normalfield	The normal component of electric field (V m^{-1})
double	potential	The cell potential (V)

Table 1 (Continued)

Type	Variable Name	Description
element*	prev	Pointer to the previous cell iterated over
Projection	proj	Structure describing the projection of the cell onto a plane normal to the velocity vector
Vector3*	ram	Unit vector in the velocity direction
double	speed	Magnitude of the spacecraft velocity (m s^{-1})
Vector3*	sundir	Unit vector from the spacecraft toward the sun
double	sunIntensity	Ratio of sun intensity to the usual sun intensity at 1 AU
BOOL	sunlit	True if the surface normal has a positive scalar product with the sun direction.

For the node object, the only potentially useful public variable is the index. For the material object the name and the arrays of input property values (pInput) and processed values (pProps) are publicly available. (Note that the property array indices in C++ are one less than their fortran indices.) The properties of the conductor object are all publicly accessible.

3. DISJOINT GRIDS IN *NASCAP-2K*

3.1   Overview

Some time ago we developed the capability of defining objects in disjoint grids for *Nascap-2k* calculations. The idea is that, while the grid and object models are disjoint, the objects are electrically connected, so that meaningful calculations may be done. In the course of this description we will illustrate a simple example of such a calculation.

Note that the scope of this task is to retest and document the capability as it presently exists. Where the capability is unnecessarily restrictive or puts unnecessarily onerous burdens on the user to work outside the GUI, this will be noted. Perhaps in the future some or all of these problems will be fixed.

3.2   Defining the Objects and Grids

We start by defining each of the disjoint objects together with its grid structure in the usual way. At present, it is necessary that both outer grids have the same mesh spacing. The apparent reason for this is that PotentDLL (the potential solver) and ScannerDLL (the potential plotter) seem to calculate the local mesh spacing by dividing the first primary grid spacing by the subdivision ratio, rather than reading the local mesh spacing from the grid info structure or recognizing the second primary grid. Using different primary grid spacings will cause potentials to be both wrongly calculated and displayed.

For this example, we define a "Lower" object as a three meter 6×6×6 gold cube, and an "Upper" object as a one meter 4×4×4 aluminum cube. Both use a 0.8 meter primary grid, with a three grid structure for the "Lower" object and a four grid structure for the "Upper" object. Figures 1 and 2 show GridTool pictures of these two objects and grid structures.

10

3.3 Combining the Grids

The two grid files must be combined using the CombineGrids Java application. The jar for this application should be included in the *Nascap-2k* installation; otherwise it can be obtained from SAIC. Figure 5 shows the CombineGrids interface. The full pathnames to the primary, secondary, and combined grid structures should be entered. In the "Offset Vector" field, enter the vector distance from the center of the primary grid structure to the center of the secondary grid structure. In this case, we place the "Upper" object twenty meters above (+Z) the "Lower" object. (This relatively modest distance is chosen so that the results can be easily seen in the *Nascap-2k* GUI. If more realistic multikilometer distances are used, displaying the secondary grid will be difficult, and displaying both grids impossible.)

Clicking the "Combine" button will write out the combined grid (.grd) file. In this case, grids 1-3 will be the grids of the "Lower" object, grid 4 will be the primary grid of the "Upper" object, and grids 5-7 (descended from 4) will be the refined grids of the "Upper" object. Details of this will be printed to the console window. The last quantity printed on the console window is "Object Center Offset." If both objects are centered in their respective grids, this will be the same as the grid offset. Otherwise, the numbers should be noted for later use.

3.4 Creating the Combined Project and Database

The new project and database should be created in a directory containing the new grid file (Combined.grd), possibly the two component objects, possibly a set of schemas, and no other "Combined" files. (Note that we use the prefix "Combined" in this documentation, but, of course, the actual project name is at the user's discretion.)

3.4.1 Create a new project.
Launch the *Nascap-2k* GUI and select "Create New Project." Uncheck "Create New Folder," click "Set Location" to place the project in the folder containing "Combined.grd." Assign a prefix (in our case, "Combined") to the project. Click "OK."

3.4.2 Load the primary object.
Click the menu item "File|Load Object ...." Navigate to the primary object definition file (in our case "LowerObject.xml"), select it, and click "Open." The "Grid Status" should show the grid already loaded.

3.4.3 Select problem type and parameters.
You must choose something. For this example, we choose "LEO" orbit and leave the "Problem Type" with the corresponding defaults. On the "Environment" tab, set Density=1e11, Temperature=0.3, B=(0., 2.5e-5, 0.) (northward), V=(7500, 0, 0) (eastward). (Correspondingly, V×B will be in the Z direction, which we consider upward.) Add the Oxygen species.

3.4.4 Build the script.
The script (as it appears in the Script window) is shown in Figure 6a. Alternatively, you may write the XML version of the script (shown in Figure 6b) to a file (*e.g.*, AppendScript.xml) and load it using the "File|Load Script…" menu item. Note that the (x,y,z) coordinates appearing in the AppendXML command correspond to the "Object Center Offset" printed by CombineGrids, and is not necessarily the same as the grid offset.

11

After running the script, the combined object can be viewed on the "Results3D" tab, as shown in Figure 7.

3.5 Calculating Potentials

The script used for calculating potentials is shown in Figure 8. Note that specification of the magnetic field is problematic in both the "Charge_Surfaces" commands and the potential input file. In the "Charge_Surfaces" commands, the SetBField command does not exist on the menu, so it must be inserted into the external XML version of the script and re-loaded. In the potential solver input file, the GUI does not write the BFIELD keyword and values. So, it is necessary to save the files, add the BFIELD keyword and values (a convenient place is immediately following the OBJVEL keyword), and uncheck the "Automatically overwrite files" checkbox. The potential solver input file is shown in Figure 9.

The "Value" in the "SetVXBPotentials" command is the maximum (most positive or least negative) potential to appear on the object. If timesteps are run, object potentials will be adjusted from this initial condition based on the charging currents.

Note also that *Nascap-2k* previously shipped with wrong signs for the magnetic field induced potential in both the BEM and Potent modules.

When running the script, if asked about overwriting files, click "No to All."

Figure 10 shows a display of the resulting object and space potentials. Note the magnetically induced potential variation on the object surfaces, and the fact that the potentials are split between the two disjoint grid structures. Figure 11 is a blow-up of the "upper" part of Figure 10, showing that potentials have been correctly calculated and plotted in this region.

3.6 Displaying Trajectories

Particle calculations can be done, but care must be taken to ensure that the particle generator and tracker input files are as intended. Make sure that the particle tracking and plotting limits are large enough to include the secondary grid. Make sure that the magnetic field is included correctly in the tracker input file. Check the output files to make sure the charge and mass of the tracked species is correctly specified. If necessary, reduce the number of tracking steps and tracking iterations to avoid a "Java Out Of Memory Error" when plotting.

Figure 12 shows "Visualization" of trajectories for electrons generated at the intersection of the 0.5 volt contour with the Y=0 plane in the secondary grid. To get extended potentials, the density (in the potential solver input file) has been reduced to 1.e6. The potentials in Figure 12 are on the X=0 plane. As expected, the particles E×B drift along the potential contour, and none hit the object. Figure 11 shows "Visualization" of trajectories for electrons generated at the intersection of the 0.5 volt contour with the X=0 plane in the secondary grid. The potentials in Figure 13 are on the Y=0 plane. In this case, the electrons whose Y-values pass through the object are rapidly collected, while the remainder bounce parallel to the Y-axis while E×B drifting.
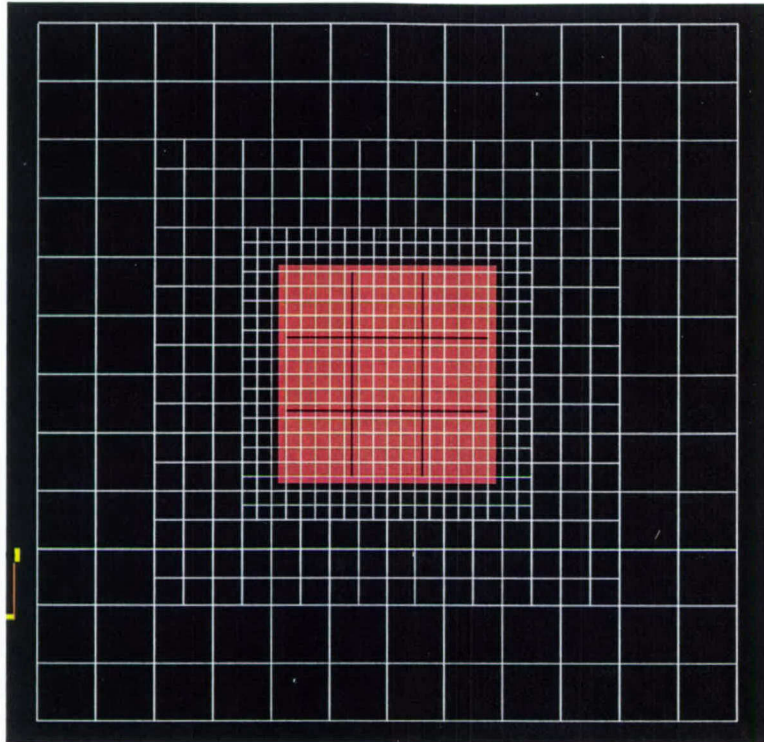
Figure 3. The "Lower" Object and Grid.
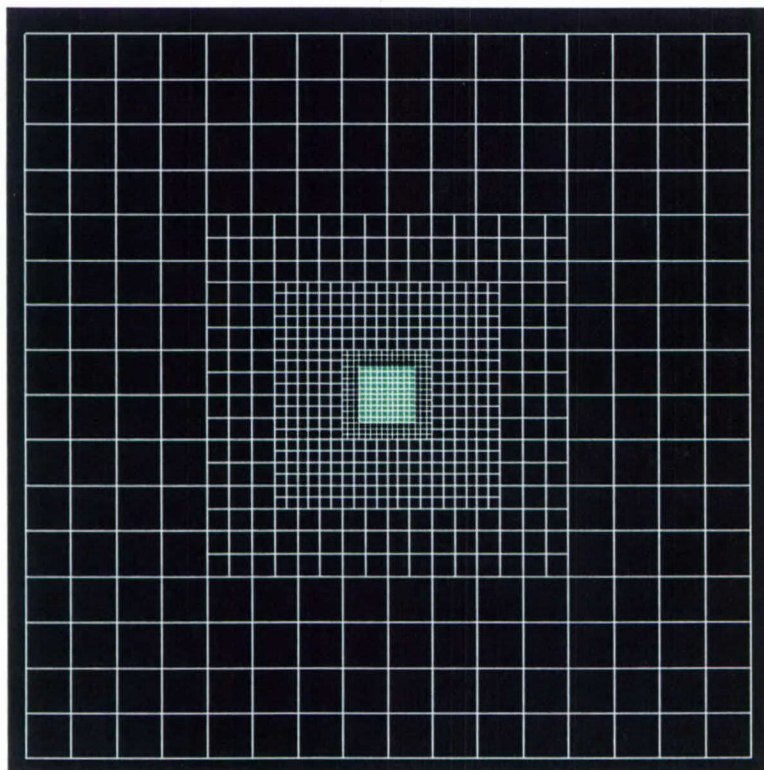
Figure 4. The "Upper" Object and Grid.

13
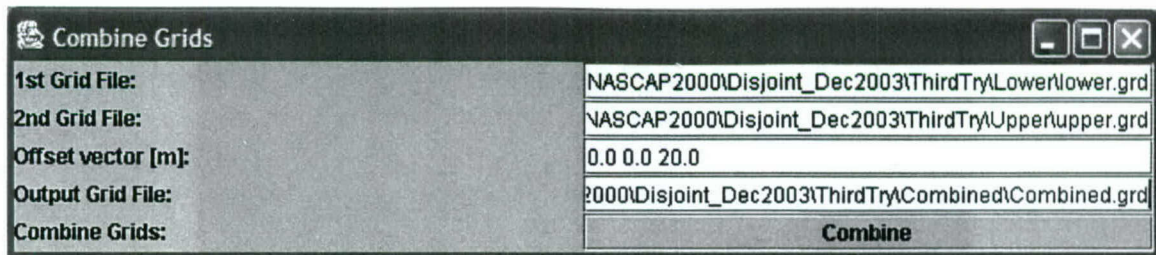
Figure 5. The CombineGrids User Interface.

(a)
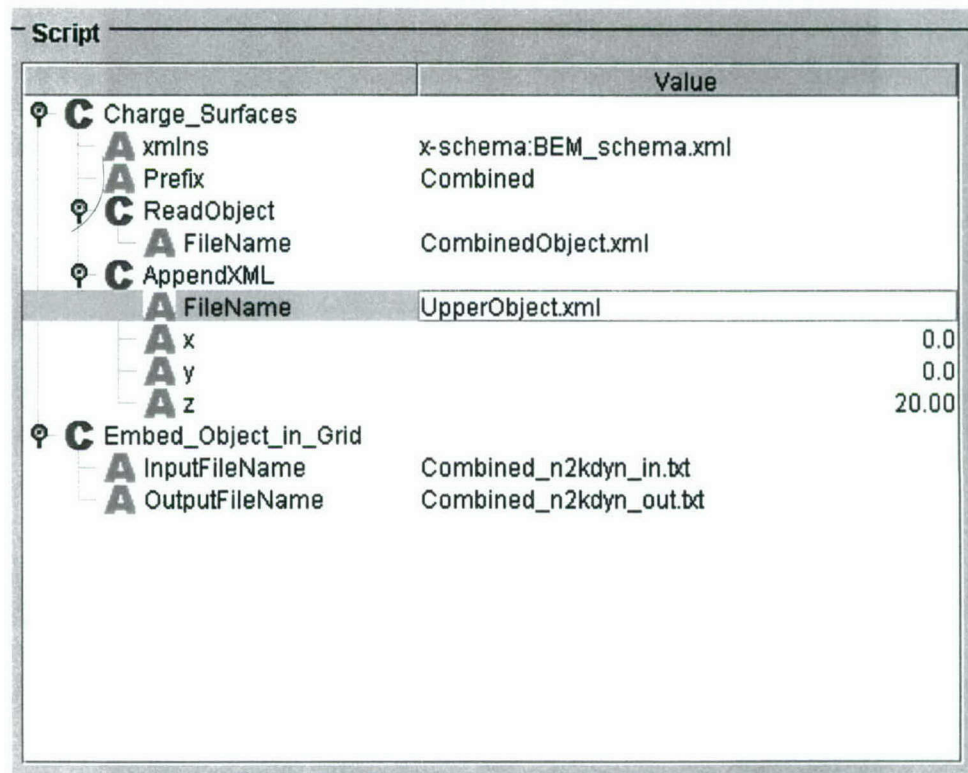
```xml
<?xml version="1.0" encoding="UTF-8" ?>
E:\NASCAP2000\Disjoint_Dec2003\ThirdTry\Combined\AppendScript.xml - # <SCRIPT>
E:\NASCAP2000\Disjoint_Dec2003\ThirdTry\Combined\AppendScript.xml - #        <COMMAND
    cmd="Charge_Surfaces" xmlns="x-schema:BEM_schema.xml" Prefix="Combined">
            <COMMAND cmd="ReadObject"  FileName=CombinedObject.xml" />
            <COMMAND cmd="AppendXML" FileName=UpperObject.xml" x="0.0" y="0.0" z="20.0" />
    </COMMAND>
      <COMMAND cmd="Embed_Object_in_Grid" InputFileName="Combined_n2kdyn_in.txt"
      OutputFileName="Combined_n2kdyn_out.txt" />
</SCRIPT>
```

(b)

Figure 6. (a) Script Used to Append Upper Object. (b) XML Version of Append Object Script.

14
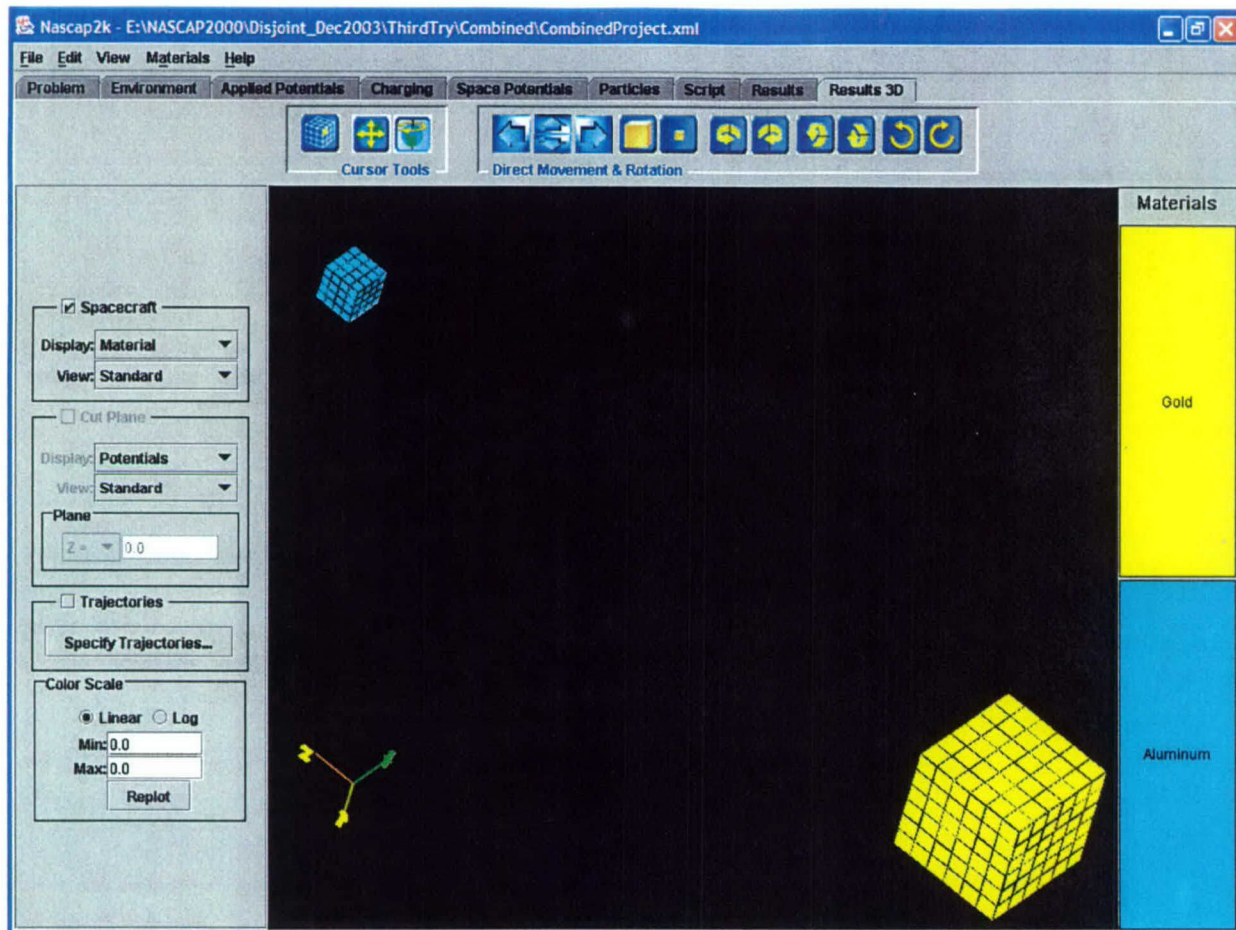
Figure 7. View of Combined Object After Running the Append Object Script (Figure 6).

(a)

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<SCRIPT>
        <COMMAND cmd="Charge_Surfaces" xmlns="x-schema:BEM_schema.xml" Prefix="Combined">
                <COMMAND cmd="OpenDatabase" Prefix="Combined" />
                <COMMAND cmd="SetVelocity" x="7500.0" y="0.0" z="0.0" />
                <COMMAND cmd="SetEnvironment">
                        <Environment type="LEO" ne1="1.12E11" te1="0.3" ni1="1.12E11" ti1="0.3"
                        from="0.0" to="-1.0" />
                </COMMAND>
                <COMMAND cmd="SetBField" x="0.0" y="2.5e-5" z="0.0" />
                <COMMAND cmd="SetInitialPotentials" />
                <COMMAND cmd="SetVXBPotentials" Value="3.0" />
                <COMMAND cmd="WritePotentials" />
        </COMMAND>
        <COMMAND cmd="Potentials_in_Space" InputFileName="Combined_potent_0_in.txt"
        OutputFileName="Combined_potent_0_out.txt" Iteration="0" />
</SCRIPT>
```

(b)

Figure 8. (a) Script for Calculating Potentials (GUI View). (b) Script for Calculating Potentials (XML View).

16

```
Comment -- Potential Solver Input File
Comment
Comment -- File Prefix ...
PREFIX      Combined
Comment
Comment -- New or Continue run ...
RUN     NEW
Comment
Comment -- Time parameters (NEW run only) ...
TIME START   0.0000E00   seconds
TIME RISE    0.0000E00   seconds
TIME FALL    1.0000E30   seconds
Comment
Comment -- Convergence criteria ...
MAXITS        20  max space charge iterations
RMSMIN      1.0000E-04  min RMS error
MAXITC        50  max potential iterations
POTCON      2.0000E00   SCG Convergence - orders of magn.
RDRMIN      1.0000E-04  min rdotr
DEBLIM      2.0000E00   debye per zone limit
Comment
Comment -- environment ...
DEBYE       1.2167E-02  debye length (meters)
TEMP      3.0000E-01  plasma temperature (eV)
DENSITY     1.1200E11   plasma density (1/m**3)
Comment
Comment -- algorithm ...
ALGORITHM    32_NODE    32-node algorithm
Comment
Comment -- problem type ...
PROBLEM     NON_LINEAR  Nonlinear screening
Comment
Comment -- other options ...
DEBYE_SCALE   LOCAL    Scaled by local xmesh
CONV_EFFECT    ON     analytic convergence
Comment
Comment -- range of loop over grids ...
GRIDLOW       0  lower limit
GRIDHIGH      0   upper limit
Comment
Comment -- mixing old and new solutions ...
SOLUTION_MIX   0.0000E00   old solution fraction
Comment
Comment -- Wake parameters (NEW run only) ...
WAKE      OFF
OBJVEL     7.5000E03 0.0000E00 0.0000E00
BFIELD     0.0000E00 2.5e-5 0.0000E00
RMASS        16   AMU
NADD       1
NPHI      36
NTHETA       180
Comment
Comment -- diagnostics ...
DIAG INIT      1  PSinit
DIAG FINAL     1  PSfinal
DIAG SCG       1  PSscg
DIAG SCREEN      0  PSscrn
DIAG MATRIX     0  PSmtrx
DIAG INTERFACE    1  PSgrds
DIAG WAKE      1  Wake Diags
Comment
Comment -- miscellaneous ...
TIMER       1   Timer Level
END
```

Figure 9. Potential Solver Input File.
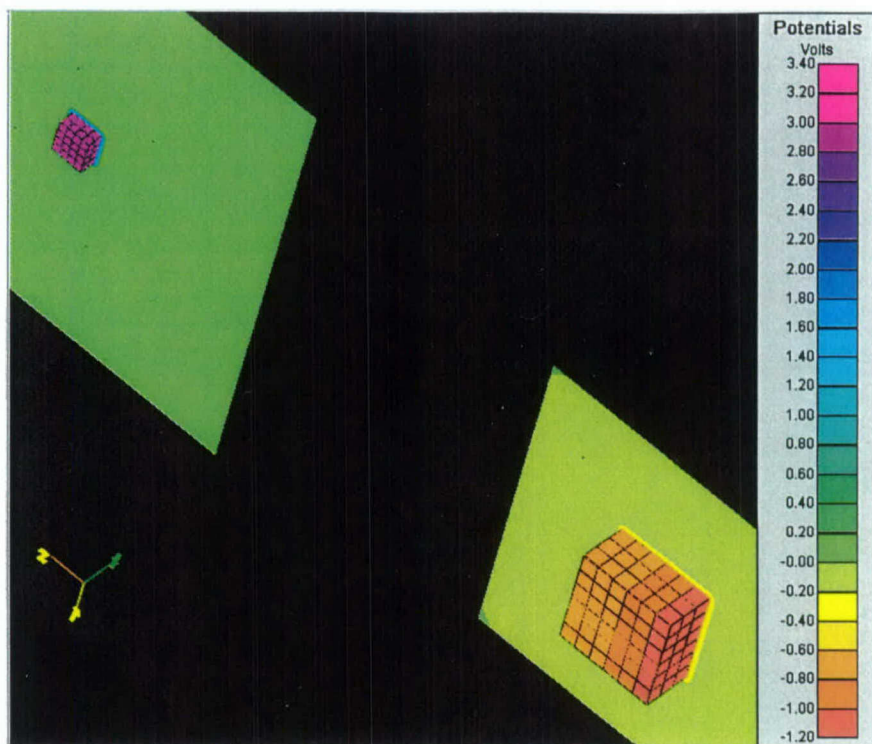
Figure 10. Results3d Picture After Running Potential Script. Note Magnetically Induced Potential Variations on Object.
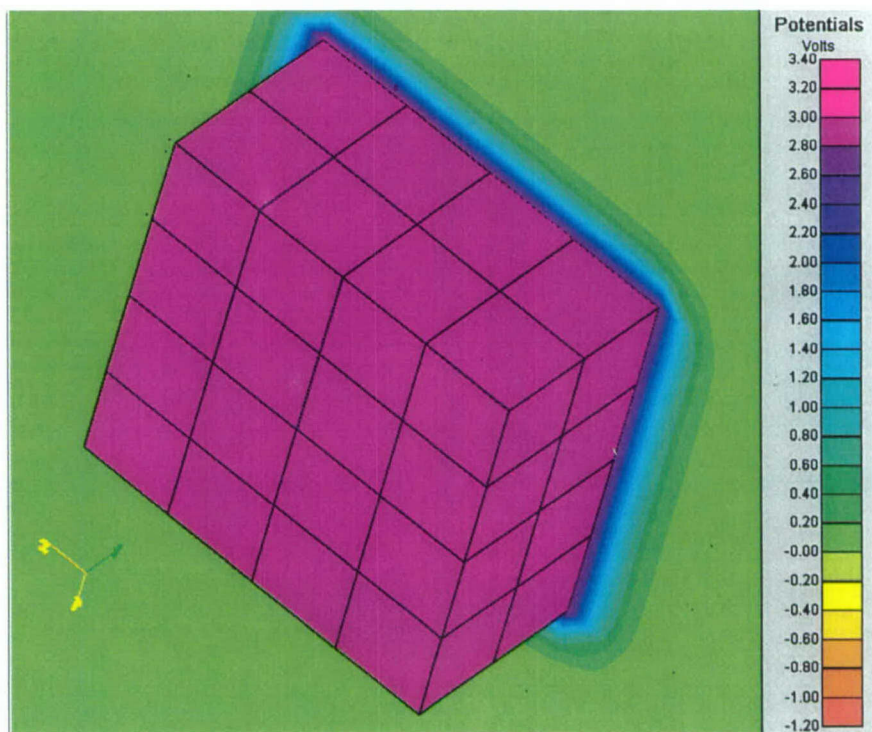
Figure 11. Blow-up of the "Upper" Part of Figure 8, Showing That the Potentials Have Been Correctly Calculated and Plotted.
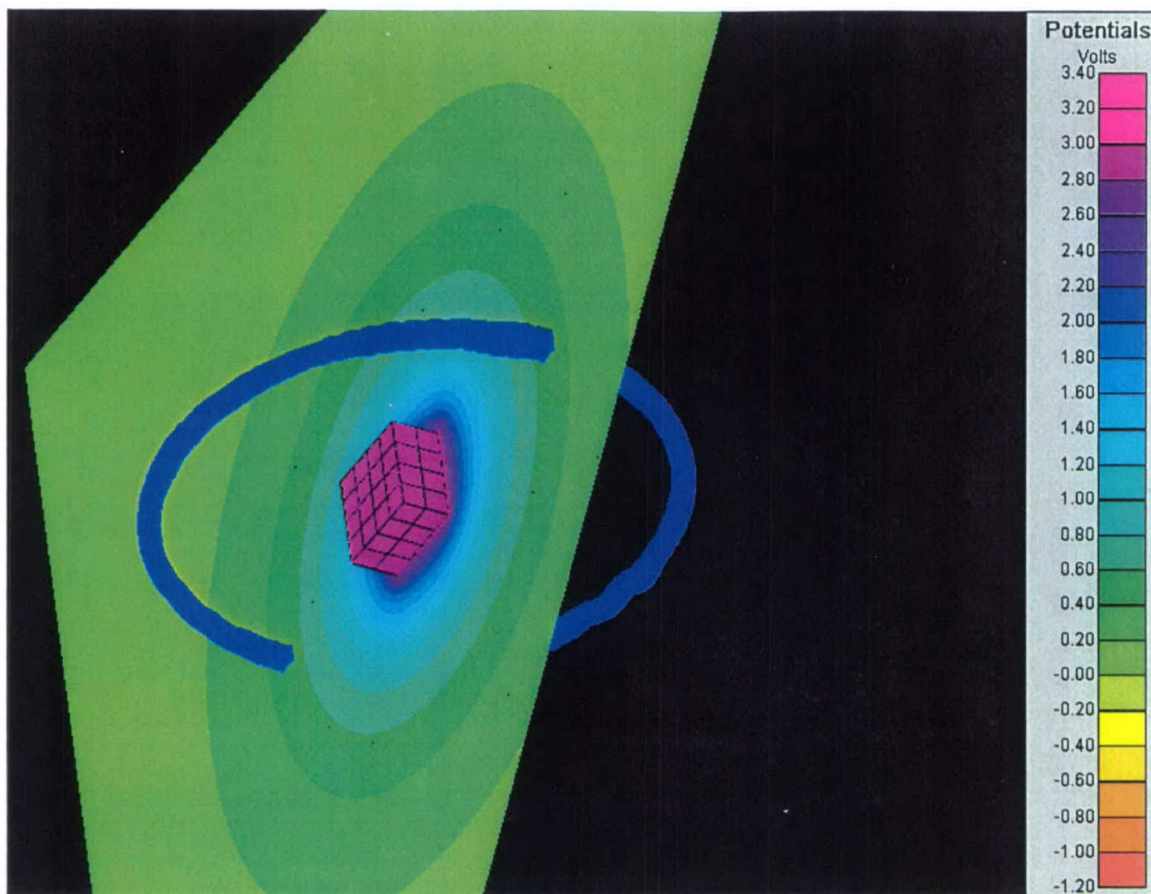
Figure 12. Trajectories of Particles Generated at the Intersection of the 0.5 Volt Contour and the Y=0 Plane. These Particles E×B Drift Along the Potential Contour.
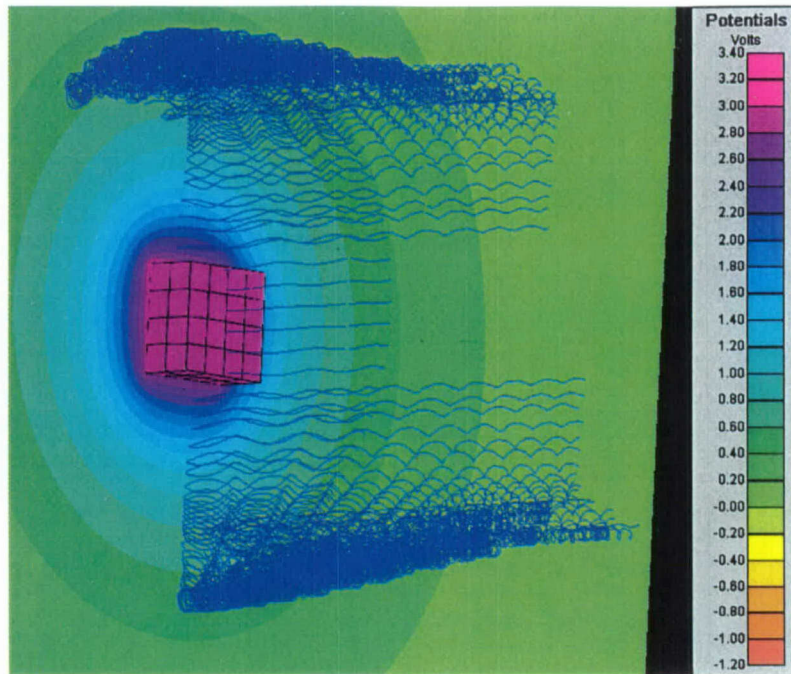
(a)
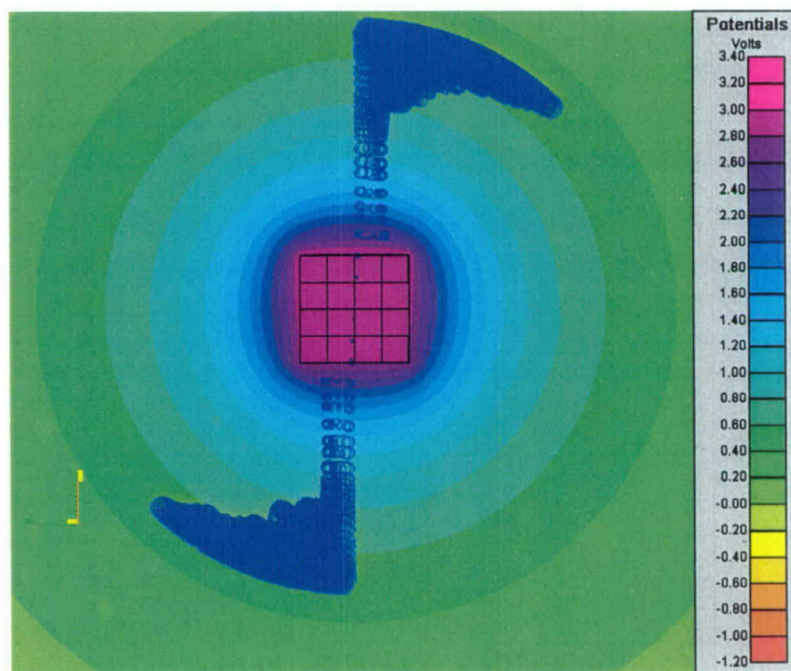
(b)

Figure 13 (a) Trajectories of Electrons Generated at the Intersection of the 0.5 Volt Contour and the Plane X=0, Superimposed on Potential Contours on Y=0 Plane. Electrons Follow Magnetic Field Lines (parallel to Y) to Hit or Miss the Object. (b) Trajectories of Electrons Generated at the Intersection of the 0.5 Volt Contour and the Plane X=0, Superimposed on Potential Contours on Y=0 Plane. Magnetic Field Direction (Y-direction) is Normal to Paper. Electrons That Miss the Object E×B Drift Along the Potential Contour in a Clockwise Direction Until the Calculation Runs Out of Time.

4. PORT OF *NASCAP-2K* TO LINUX

We have modified *Nascap-2k* so that it can be built and run under either a Win32 or a LINUX operating system. The "DynaPAC database" files are cross platform. Calculations can be run under LINUX and the files transferred to Windows (or vice versa) if desired. The graphical user interfaces, which are written in Java, are built under Windows and executed under either operating system.

The Fortran, C, and C++ computational portion of the code is now cross platform. The build instructions are contained in the dsp project files under Windows and in the makefiles under LINUX. IFDEF structures are used as needed for operating system specific code. Some features are not yet functional under LINUX. The ones identified are given in the Caveats section of this report. Only basic testing has been performed.

The *Nascap-2k* Makefile structure is built on the older DynaPAC Makefile structure. No effort to detangle the old code from the present makefiles has been made. The advantage to this is that executables for the DynaPAC codes are built along with the *Nascap-2k* libraries.

We found that on our older LINUX computer (P2-233 running RedHat 6.5), we are able to compile and link the shared objects (SO), but are, in practice, unable to run the *Nascap-2k* GUI. When opening a project for which the database has already been created and then clicking on the "Results 3D" tab, *Nascap-2k* dies with a segkil error message. This does not occur on our newer LINUX computer (P3-??? running RedHat 8.0).

The versions of the various third party software used to build and execute *Nascap-2k* are given in Table 2.

Table 2. Third Party Software.

	Version compiled with	Version executed with
LINUX	6.5	8.0
Portland Group Fortran compiler	3.2	5.1
gnu c compiler and cpp preprocessor	egcs-2.91.66	3.2
JAVA 2 for LINUX	1.4.2	1.4.0
JAVA 3D for LINUX/OpenGL	1.3.1	1.3.1
Xerces XML parser	2.4.0	2.4.0

4.1 Conversion

There were several things that needed to be done to allow *Nascap-2k* to run under LINUX. *Nascap-2k*'s Java code talks through JNI (Java Native Interface) to the calculation routines, which are written in Fortran and C++. On Windows, this is done by writing C++ DLLs that can be loaded and called from Java. The C++ DLLs include the C++ or Fortran routines that perform the calculations. On LINUX the C++ DLLs need to be converted to shared object files. In order

21

to use the same code wherever possible, we built C++ shared objects by using #IFDEF to remove the Windows specific code, such as the DllMain procedures that are used on Windows when DLLs are loaded. The Microsoft generated DLL codes contain many #DEFINES that work under Windows. Wherever possible we have redefined them so that they will work under LINUX.

Codes generated in Microsoft's Visual Studio use a predefined header called stdafx.h that includes some Windows specific code along with some non-standard definitions. We created a stdafx.h file for LINUX that makes LINUX versions of the definitions and have written a few routines to replace the standard Windows routines.

The convention for calling Fortran from C++ and C++ from Fortran is also different between Windows and LINUX. We added #IFDEFs for non-Windows machines to handle these differences.

In addition to the problems mentioned above, we have found and fixed various problems that arise when porting to LINUX. An example is the fact that in order to create a *.o from a *.F, the Makefiles first run the file through cpp, which interprets a single quote mark as an uncompleted character literal and generates an error message. All single quote marks (usually apostrophes) were removed from comment statements. Another issue is that the Portland group Fortran has a more limited set of options on open statements than does the Compaq Fortran. The open statements for the input and output files where changed to be status=UNKNOWN and rewind statements added just after the open.

There are two areas of Windows specific code for which we need an alternative for LINUX. One is the lapack DLL used on Windows. We built a lapack shared object on LINUX using the same procedure that we used for the other DLLs to get the lapack functionality on LINUX.

The other difference is the parsing of XML in C++. In the Windows version, we use Microsoft's MSXML parser which is accessed as a COM object. On LINUX we use the Xerces XML parser. The Xerces XML parser is the standard open source XML parser for C++ and is widely used. Xerces is a Document Object Model (DOM) parser like the MSXML parser. The syntax for accessing the parsed documents in Xerces was completely different from MSXML syntax, so we wrote new versions of the code to parse the XML files in C++ on LINUX. We use readdynapac.cpp on Windows and readdynapacxe.cpp on LINUX. To keep this working we need to make sure that all changes that are made to readdynapac.cpp for Windows are also made to readdynapacxe.cpp. Note readdynapac.cpp also has routines that do not call the XML parser. ReadDynapac.cpp is not a full class implementation, but, rather, it implements a small number of methods belonging to the GeomModel class.

Although the Microsoft and Xerces parsers are both DOM compliant parsers, the syntax for interacting with the XML is drastically different. This is partly because we are interacting with the Microsoft parser via COM and interacting with the Xerces parser in standard C++; partly because the DOM specification refers to the abilities a parser must have but does not specify the API's that you use to provide the abilities; and partly because the Microsoft parser offers some abilities outside of the DOM specification that we use in readdynapac.cpp. Because of this, the parsing codes in the two cpp files (readdynapac.cpp and readdynapacxe.cpp) are quite different.

We built a library to access the Xerces library at a higher level and it should be straightforward to transcribe any future changes in readdynapac.cpp to readdynapacxe.cpp). readdynapacxe.cpp uses the ScXercesLib. This Lib can be built in either LINUX or Windows.

The LINUX file readdynapacxe.cpp can be used on Windows (this is a good way to test it). To do this BEMDLL.DLL must be made using readdynapacxe.cpp. This is done by deleting readdynapac.cpp from the BEMDLL project and adding readdynapacxe.cpp. Then readdynapacxe.cpp is set to not use precompiled headers. The Xerces includes and libs must be in the path. To run it the Xerces.DLLs must be in the path.

On LINUX the Xerces shared object files must be in the user's path to run the code. To make it work on the older LINUX computer we made them there (they did not have binaries for that version of LINUX to download). The SO files can be downloaded for most versions of LINUX, so that they do not have to be made. The source and the binaries are available from http://xml.apache.org/xerces-c/index.html. We made the SO files exactly as suggested on the web.

4.2 Building Nascap-2k on LINUX

Java and Fortran should be in the user's path.

In addition to setting the SCRENPKG and DYNAPAC variables, the XERCESDIR variable in Nascap2k.setup should be set to point to top level directory where the Xerces files are located and JAVA_HOME should be set to the top level Java directory. A new make command n2kmake sos was added. The sos stands for shared objects. The command n2kmake all also executes n2kmake sos.

The LINUX version of *Nascap-2k* includes the executables that were originally part of DynaPAC, including the MIRIAD based graphical user interfaces, DynaPre, DynaPost, Scanner, and GridTool (original version). Because of this, it is necessary to also build the screen package. The screen package requires that screenpkg/inputs/makeincl have the correct path for the variable INSTALLPATH.

The makefiles require that each directory of Fortran files have a subdirectory named Dbx and that a top level directory lib exists. If they do not exist, they need to be created. There are files in the src/GridTool, src/DynaPost, src/DynaPre, and src/Scanr directories that need to be writable. (ddisub.f, and io*.f files)

Before building *Nascap-2k* for LINUX, the Xerces shared objects must exist. Build instructions are included at the apache web site.

To build *Nascap-2k* for LINUX, go to the head *Nascap-2k* directory, make the appropriate path changes in the heading of the Nascap2k.setup file and screenpkg/inputs/makeincl, source the file Nascap2k.setup, and then type "n2kmake scrnpkg" and then "n2kmake all".

To run *Nascap-2k* the SOs that are called from Java must be in the Java lib path and all of the SOs must be in the load library path for the C++ code. The command "n2kmake all" places the *Nascap-2k* SOs in the bin directory.

The Jar files, which are created under Windows, (Nascap2k.jar, ObjectToolkit.jar, and GridTool.jar) should be placed in the bin directory. The help files, GridToolHelp.html, Nascap2kDocumentation.htm, ObjectToolkitHelp.htm and associated folder (Nsacap2kDocumentation) with graphics files should also be placed in the bin directory. The lib directory containing the Apache jar files (axis.jar, commons-discovery.jar,…) should also be placed in the bin directory.

4.3   Running Nascap-2k on LINUX

Java should be in the user's path.

To execute, source the Nascap2k.setup file and then type N2K, OTK, or GT as desired.

4.4   Caveats

The buttons on the "Problem" tab in the *Nascap-2k* GUI that bring up Object ToolKit and GridTool, do not work under LINUX.

When running, the absolute path of the *Nascap-2k*/bin directory must be the same as it was when the code was built.

5. CHARGE STABILIZED POISSON ITERATION IN *NASCAP-2K*

The Poisson equation can be written dimensionlessly as

$$-\nabla^2\Phi = \left(n_i - n_e\right)/L^2 \qquad (1)$$

where

$$\Phi = \frac{eV}{kT}, \text{ and } L^2 = \varepsilon_o \frac{kT}{N_o e^2 h^2} = \frac{\lambda_D^2}{h^2}$$

L is the dimensionless Debye length, N_o is the ambient plasma density, $n_i = N_i/N_o$, $n_e = N_e/N_o$, and the Laplacian is also normalized by h^2. *Nascap-2k* solves this equation on its discrete mesh of uniform spacing h, using the finite element method.

The traditional approach to the solution of equation (1) has been an explicit iteration of the form

$$-\nabla^2\Phi^v = L^{-2}\left(n_i\left(\Phi^{v-1}\right) - n_e\left(\Phi^{v-1}\right)\right) \qquad (2)$$

where v is the iteration index and the charge density is determined using the potentials of the previous iteration. This method can be shown to be unstable[13] when the Debye length, λ_D, becomes small with respect to other scale length of the problem. This can be understood by considering that a smooth potential variation over a distance of, say, 1000, would require a smooth $\nabla^2\phi$ (the 'second derivative') which is, in turn, given everywhere by the charge density. Maintaining a smooth charge density distribution is difficult when any errors in determining (n_e - n_i) are multiplied by a huge L^{-2}. There is one effective remedy to this dilemma due to Parker[13], but the process reported here appears to be more efficient in the short Debye length limit. This method involved the combination of two concepts. One uses a partial implicitization of the repelled density[14]. The other simply reduces the charge density to an acceptable level whenever the first method is inadequate.

Nascap-2k uses six different expressions to describe the charge density for different kinds of problems. A summary of the options is given in Table 3. Each expression uses implicitization and charge limiting to a different extent. As there is no space charge for the Laplace space charge option, the following discussion does not apply to that case.

5.1 Implicitization

The right hand side of Equation 2 is the dimensionless charge density.

$$q\left(\mathbf{r},\Phi^v\left(\mathbf{r}\right)\right) = L^{-2}\left(n_i\left(\Phi^v\right) - n_e\left(\Phi^v\right)\right) \qquad (3)$$

The charge density at the present iterate may be linearized about the previous potential iterate

$$q\left(\Phi^v\right) \cong q\left(\Phi^{v-1}\right) + q'\left(\Phi^{v-1}\right)\left(\Phi^v - \Phi^{v-1}\right)$$

25

where $q' = \dfrac{\partial q}{\partial \Phi}$, and the **r** dependence has been dropped for clarity. With this expression, we may write the implicit Poisson iteration scheme

$$-\nabla^2 \Phi^\nu - q'\left(\Phi^{\nu-1}\right)\Phi^\nu = q\left(\Phi^{\nu-1}\right) - q'\left(\Phi^{\nu-1}\right)\Phi^{\nu-1} \qquad (4)$$

Though it is not immediately obvious, the implicit character of Equation 4 makes it more stable than scheme (2) provided q́ is constrained to be non-negative. This can be understood by realizing that in Equation 3 the charge density is treated as an independent variable, whereas in Equation 4 the charge density is determined simultaneously with the potential.
The finite element approximation to Equation 4 produces the matrix equation

$$\sum_e \left(W^{(e)} - \overline{S}'^{(e)}V^{(e)}\right)\Phi^\nu = S - \overline{S}'^{(e)}\Phi^{\nu-1} \qquad (5)$$

where (e) refers to each element and S is derived from q by the following analysis.

Table 3. Charge Density Options for *Nascap-2k*.

Laplace	Zero space charge
Linear	Net space charge linear with potential; appropriate for low potentials. ("Debye approximation.")
Non-linear	Non-linear expression that reduces to the charge density of the accelerated particles at high potentials and Debye (linear) shielding at low potentials. Includes a correction for converging particles.
Frozen ion	Charge density due to ions at ambient density and electrons at barometric equilibrium; appropriate for very short timescales. ("Ion matrix" approximation.)
Full trajectory ions	Charge density is the sum of the ion density from steady-state ion trajectories and the barometric electron density. Also used when the ion density is determined by an external code, such as EPIC.
Hybrid PIC	Charge density is the sum of the ion density from tracking ion macroparticles and the barometric electron density.
Full PIC	Electron and ion densities from tracking macroparticles.

5.2 Charge Limiting

For $L \geq 1$, S is simply the total charge associated with each node, q. (Physically, this means that an entire grid cube of plasma with one species eliminated alters the potential by no more than the temperature.) However, for $L \ll 1$, numerical noise and features like a sheath edge, which may span only a few λ_D, become incorrectly amplified when the q determined at a point becomes multiplied by the entire nodal volume. When it is not possible to reduce the zone size, stability can be preserved by replacing Q (and Q´) with a reduced value S, (and S´) which is calculated to be the maximum allowable charge for the element.

Because of the artificial amplification argument, S is often the more realistic total for an element, in the sense that it produces potential variation appropriate to the spatial resolution rather than causing unphysical overshoots. Before deriving S, we define the barometric potential $\Phi_b = \ln(n_i)$, which (in cases where electron density is governed by the Boltzmann factor, e^{Φ}) is the potential for which Q = 0. For the Linear and Non-linear space charge density formulations, the ion charge density is also dependent on the potential. For these cases, the barometric potential as used here is zero, $\Phi_b = 0$. Note that it is important that S→ Q as $\Phi \to \Phi_b$ if quasineutral regions are to be modeled correctly. To determine S, consider a capacitor with potential difference $\Phi_b - \Phi$, area h^2, and a separation of h. The charge q_c on this capacitor is given by

$$q_c = C\Delta V = \frac{\varepsilon_o h^2}{he}(\Phi_b - \Phi)kT \tag{6}$$

In the units of our previous q, q_c becomes

$$q_{Limit} = \alpha(\Phi_b - \Phi) \tag{7}$$

which is the maximum allowable charge per element, with the parameter α, set to the maximum value consistent with the stability of the Poisson solver. Thus at each node, we choose for the charge

$$|S| = \min\left(|q_{Limit}|, |q|\right) \tag{8}$$

with

$$S' = \begin{cases} -\alpha, & \text{for } S = q_{Limit} \\ \dfrac{\partial q}{\partial \Phi}, & \text{for } S = q \end{cases} \tag{9}$$

The way in which S and S′ are computed for each charge density formulation appears in the section below titled "Charge density and derivative in *Nascap-2k*."

The effect of this algorithm is as follows: If a problem has been specified where a boundary potential would be screened in less than a zone or two (the limit of any code's resolution), sufficient sheath charge is redistributed to allow the potential to be screened over the minimum number of zones that is consistent with stability.

5.3 Analysis of the Charge Stabilized Poisson Method.

The charge stabilized Poisson method calculates for each node the maximum allowable charge that is consistent with the stability of a linearly interpolating Poisson solver. This method is developed above, but a further analysis is presented here to help the user interpret its impact. *Nascap-2k*'s charge stabilization is accomplished through the process of charge limiting, illustrated in Figure 14. This figure shows two charge versus potential curves for the case where the ion density is fixed (tracked or equal to the background density) and the electron density is barometric. Equation 3 is rewritten as

$$q = \alpha \exp(-\Phi_m)(\exp\Phi_b - \exp\Phi) \tag{10}$$

27

where $\Phi_m = \ln\left(\alpha\lambda_D^2/h^2\right)$, α is user specified, λ_D is the Debye length, h is the mesh spacing, and Φ_b is the barometric potential, $\Phi_b = \ln(n_i)$. For curve q_1, $\Phi_b = -3.0$ ($n_i = 0.05$); for curve q_2, $\Phi_b = 0.0$ ($n_i = 1.0$) and for both curves $\Phi_m = -2.2$, and $\alpha = 1$. For each curve, the limiting charge as given by Equation 7 is also shown. The limiting charge is rewritten here as

$$q_{Limit} = \alpha\,(\Phi_b - \Phi) \tag{11}$$

which intersects the "natural" charge curves at Φ_c and Φ_b. The charge stabilization method reduces the charge to the limiting value when $\Phi > \Phi_c$ and uses the natural charge for $\Phi < \Phi_c$. The parameter Φ_m provides a good measure of the limiting process. From Figure 14 it can be seen that Φ_m is the point at which the slope of the natural charge curve equals that of the limiting charge line. Figure 15 shows a family of curves giving the dependence of the cutoff potential Φ_c on the barometric potential for various values of Φ_m. These curves were obtained by numerically solving for the zeros of the difference between q and q_{Limit}. This difference equation always has two solutions, one at Φ_c and one at Φ_b, with the exception of a degeneracy at $\Phi_c = \Phi_b = \Phi_m$, which is indicated in the figure. This figure shows that the charge limiting is minimal for $\Phi_m > -1$, and quite severe for $\Phi_m < -6$ or so.
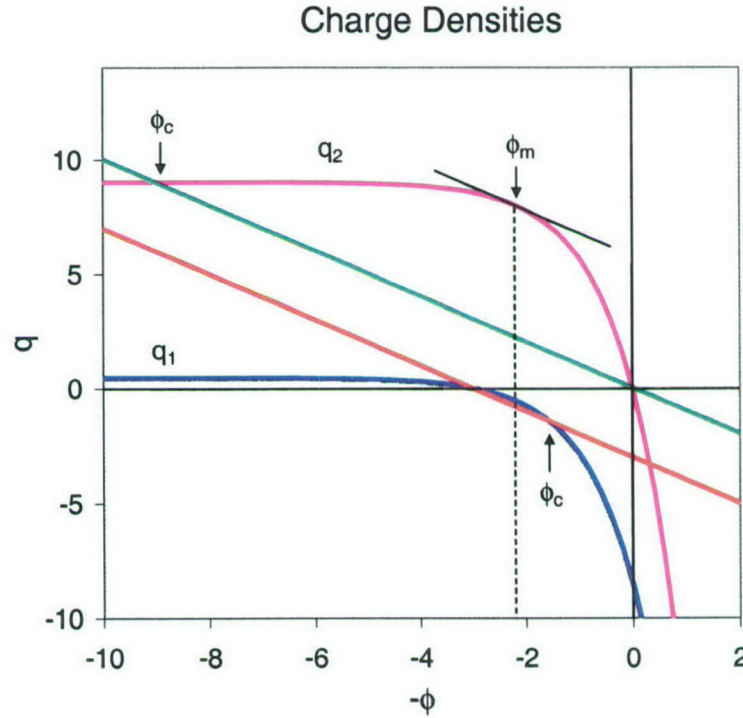

Figure 14. Plots of Space Charge (Curves q_1 and q_2 as a Function of Potential as Given by Equation 3. The Straight Lines Represent the Maximum Allowable Charge for Non-Oscillatory Potentials. The "Natural" Space Charge, q_1 or q_2 is Acceptable for Which Slopes of the Curves and the Corresponding Line are Equal.
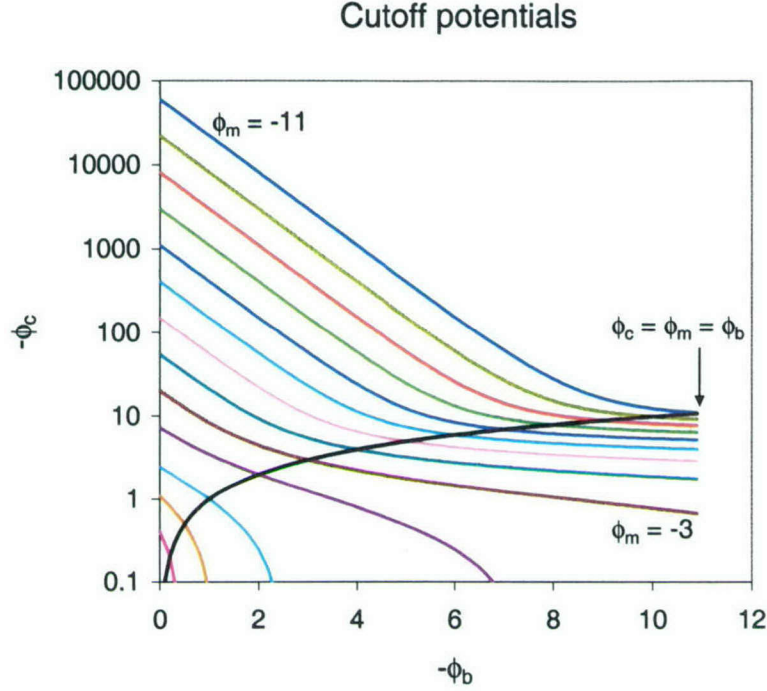
Cutoff potentials

Figure 15. Plot of the Space Charge Cutoff Potential, Φ_c, Versus Barometric Potential (Φ_b = ln n_i) for a Series of Φ_m Values (-02, -0.5, -1.0, -2.0, -3.0, --4.0 ... -11.0). The Point at Which $\Phi_m = \Phi_b = \Phi_c$ is Also Indicated.

5.4   Sheath Boundary Potential

Consider the first zone of a sheath to satisfy the laws of Child and Langmuir (planar space charge limiting). At the sheath edge (z=0) and one zone in (z = Δx) the potential and electric field are given in Table 4.

Table 4. Potential and Electric Field Variation Given by Planar Space Charge Limiting.

Position	0	Δx
Potential	0	$K(\Delta x)^{4/3}$
Electric field	0	$(4/3)K(\Delta x)^{1/3}$

By Gauss's law, the charge per unit area in this zone is given by

$$\frac{Q}{\varepsilon_o A} = \frac{4}{3} K \left(\Delta x \right)^{1/3} \tag{12}$$

Nascap-2k computes the charge density to be limited by $(\alpha/(\Delta x)^2)$ times the mean potential (assuming linear interpolation), and so gets

$$\frac{Q}{\varepsilon_o A} = \Delta x \left(\frac{\alpha}{\left(\Delta x \right)^2} \right) \frac{1}{2} \left(0 + K \left(\Delta x \right)^{4/3} \right) \tag{13}$$

Equating these two expression gives $\alpha = 8/3$.

29

Because of the economics of running a three-dimensional code, *Nascap-2k* is frequently operated at high Φ_m values, i.e., a coarse mesh with respect to the Debye length. In these cases charge is removed at almost all points. Ideally, the charge that was removed was excess charge generated by the coarse gridding. This is the artificial charge amplification argument. However, since *Nascap-2k* must be reliably stable, the result is that too much charge is often removed. This results in an enlarged sheath thickness for high negative Φ_m problems. In order to compensate for this sheath enlargement, the sheath boundary potential must be adjusted.

Equating the above equation to the code resolution, *xmesh*, gives

$$\phi_x = 5.1 \times 10^{-6} \, xmesh^{4/3} \theta^{1/3} n^{2/3} = 0.74 (xmesh / \lambda_D)^{4/3} \theta \tag{14}$$

The potential ϕ_x may be interpreted as the potential below which *Nascap-2k* underestimates screening. At best, beyond the ϕ_x contour, the potential drops about one order of magnitude per element. For $\theta = 0.1$ eV, $n = 10^{11} m^{-3}$, and *xmesh* = 0.2 m, we find $\phi_x = 6$ V. If the 6 V contour is correctly placed, the 0.6 V contour lies at least one element beyond (at the approximate sheath location), and the default sheath contour (0.07 V) is yet another element farther. This would produce a sheath area that is too large. The suggested criterion for the sheath boundary potential is

$$\phi_{SB} = Max(\theta \ln 2, 0.24 \phi_x) \tag{15}$$

where $0.24 = e^{-1.0/0.7}$ is the planar screening per element allowed by *Nascap-2k*. Note, however, that ϕ_x depends strongly on the grid in which the sheath is found, so that if an increase in object potential moves the sheath from grid 3 to grid 2, a corresponding larger value of ϕ_{SB} should be used.

5.5 Charge Density and Derivative in Nascap-2k

Nascap-2k has a number of techniques for the computation of space charge density used in the computation of potentials throughout space (Poisson's equation). In addition, as described above the implicit approach used in solving Poisson's equation uses the derivative of the charge density with the local potential. The formulas used for the various charge density formulations are given below. The symbols used to describe the charge density models are:

ρ = space charge (C m^{-3})
ε_0 = permittivity of vacuum (8.854 x 10^{-12} F m^{-1})
e = magnitude of the electron charge (1.60 x 10^{-19} C)
n = background plasma density (m^{-3})
λ_{Debye} = plasma Debye length (m)
θ = plasma temperature (eV)
n_{min} = max of user input value and 10^{-6} x n, defaults to n/100
ϕ = local space potential (V)
E = local space electric field (V m^{-1})
L = local mesh spacing (m)
D_{lim} = DEBLIM parameter on advanced screen, defaults to 2
$D = L/(D_{lim})$
g = Maximum of plasma density reduction factor computed by neutral wake model (0<g<1) and 10^{-6}

Laplace. The Laplacian space charge option specifies that the charge density is zero.

$$\frac{\rho}{\varepsilon_o} = 0 \tag{16}$$

i.e., charge exists only on object surfaces and external boundaries, as determined by the boundary conditions. "Space charge" iterations may still be required, however, due to the treatment of surface electric fields.

$$\frac{d(\rho/\varepsilon_o)}{d\phi} = 0 \tag{17}$$

Linear (Debye Shielding). The Linear space charge option solves the Helmholtz or Debye-Huckel equation:

$$\frac{\rho}{\varepsilon_o} = -\frac{\phi}{\lambda_{nl}^2} \tag{18}$$

$$\lambda_{nl}^2 = \max(\lambda_{Debye}^2/g, D^2)$$

$$\frac{d(\rho/\varepsilon_o)}{d\phi} = \frac{1}{\lambda_{nl}^2}$$

Non-Linear. Non-linear space charge is appropriate for most low-earth-orbit type plasmas. Poisson's equation is solved with space charge given by:

$$\rho/\varepsilon_0 = -\left(\phi/\lambda_{nl}^2\right)\frac{\max\left(1, C(\phi, E)\right)}{1 + \sqrt{4\pi}\left|\phi/\theta_{nl}\right|^{3/2}} \tag{19}$$

$$C(\phi, E) = \min\left(\left(R_{sh}/r\right)^2, 3.545\left|\phi/\theta_{nl}\right|^{3/2}\right)$$

$$\left(R_{sh}/r\right)^2 = 2.29\left|E\lambda_{nl}/\theta_{nl}\right|^{1.262}\left|\theta_{nl}/\phi\right|^{0.509}$$

$$\lambda_{nl}^2 = \max(\lambda_{Debye}^2/g, D^2)$$

$$\theta_{nl} = \theta(\lambda_{nl}^2 g/\lambda_{Debye}^2)^{2/3}$$

The term $C(\phi, E)$ (analytic focusing) comes from fitting a finite temperature spherical (Langmuir-Blodgett) sheath. If analytic convergence is turned off (on the Potentials Advanced screen) then $C(\phi, E)$ is set to zero. Additional adjustments are made to the convergence portion of this formula for significant values of velocity.

$$\frac{d(\rho/\varepsilon_0)}{d\phi} = \text{Max}\left(\left|\frac{\rho/\varepsilon_0}{\phi}\right|, (1/\lambda_{nl}^2)\frac{\max(1,C(\phi,E))}{1+\sqrt{4\pi}|\phi/\theta_{nl}|^{3/2}}\left(1-\left(\frac{3}{2}\right)\frac{\sqrt{4\pi}|\phi/\theta_{nl}|^{3/2}}{1+\sqrt{4\pi}|\phi/\theta_{nl}|^{3/2}}\right)\right)$$

Note that the derivative of the convergence contribution to the charge density with respect to the potential is assumed to be negligible.

Frozen Ion. The **Frozen Ion** formulation is intended for short timescale (typically sub-microsecond) problems for which it is a good approximation to assume that ions remain stationary and at ambient density ("ion matrix" approximation), but electrons achieve barometric equilibrium. The space charge function depends on the mesh-dependent potential, $\phi_1 < 0$, which satisfies

$$1 - \exp(\phi_1/\theta) = -(\lambda/D)^2 (\phi_1/\theta) \tag{20}$$

for $D > \lambda$. Otherwise, $\phi_1 = -1\times 10^{-6}$. The space charge is then given by

$$\rho/\varepsilon_0 = \begin{cases} (\phi_1/D^2)(1-\exp(\phi/\phi_1)) & \phi \geq 0 \\ -\phi/D^2 & 0 \geq \phi \geq \phi_1 \\ -\phi_1/D^2 + (\theta/\lambda_{Debye}^2)(\exp(\phi_1/\theta)-\exp(\phi/\theta)) & \phi_1 \geq \phi \end{cases}$$

$$\frac{d(\rho/\varepsilon_0)}{d\phi} = \begin{cases} \text{Max}\left(-\frac{\rho/\varepsilon_0}{\phi}, \exp(\phi/\phi_1)/D^2\right) & \phi \geq 0 \\ 1/D^2 & 0 \geq \phi \geq \phi_1 \\ \text{Max}\left(-\frac{\rho/\varepsilon_0}{\phi}, \exp(\phi/\theta)/\lambda_{Debye}^2\right) & \phi_1 \geq \phi \end{cases}$$

Full Trajectory Ions. Ion densities are calculated from steady-state ion trajectories. Electrons are barometric.

$$\frac{\rho}{\varepsilon_0} = \frac{\rho_i^{ft}}{\varepsilon_0}\left(1-\exp\left(\min\left(10,(\phi-\phi_b)/\theta_{ft}\right)\right)\right) \tag{21}$$

$$\rho_i^{ft} = \max\left(\rho_{ion}^{tracked}, e\,n_{min}\right)$$

$$\phi_b = \theta \ln\left(\frac{\rho_i^{ft}}{en}\right)$$

$$\theta_{ft} = \max\left(\theta, D^2\frac{\rho_i^{ft}}{\varepsilon_0}\right)$$

$$\frac{d(\rho/\varepsilon_0)}{d\phi} = \begin{cases} -\dfrac{\rho}{\varepsilon_o}\dfrac{1}{\phi-\phi_b} & (\phi-\phi_b)/\theta_{ft} > 1 \\[2ex] \dfrac{\rho_i^{ft}}{\varepsilon_o\theta_{ft}} & abs\left((\phi-\phi_b)/\theta_{ft}\right) \le 10^{-3} \\[2ex] \max\left(-\dfrac{\rho}{\varepsilon_o}\dfrac{1}{\phi-\phi_b}, \dfrac{\rho_i^{ft}\exp\left(\min\left(10,(\phi-\phi_b)/\theta_{ft}\right)\right)}{\varepsilon_o\theta_{ft}}\right) & \text{otherwise} \end{cases}$$

Hybrid PIC. This algorithm is used for timescales (typically sub-millisecond) on which it is practical to treat ion motion, but electrons may be considered in barometric equilibrium. The ion density is computed from actual ion macroparticles as computed by the TrackerDLL. The electron charge density is,

$$\rho_e/\varepsilon_0 = \begin{cases} -(\phi+\theta_{eff})/\lambda_{eff}^2 & \phi > 0 \\ -(\theta_{eff}/\lambda_{eff}^2)\exp(\phi/\theta_{eff}) & \phi \le 0 \end{cases} \tag{22}$$

$$\lambda_{eff} = \max\left(\lambda_{Debye}, D\right)$$

$$\theta_{eff} = \theta\lambda_{eff}^2/\lambda_{Debye}^2$$

$$\rho = \rho_e + \rho_{ion}^{tracked}$$

$$\frac{d(\rho/\varepsilon_0)}{d\phi} = \begin{cases} 1/\lambda_{eff}^2 & \phi > 0 \\ -(1/\lambda_{eff}^2)\exp(\phi/\theta_{eff}) & \phi \le 0 \end{cases}$$

Full PIC. For this option, it is assumed that the TrackerDLL has stored both the electron and ion charge densities.

$$\rho = \rho_{electron}^{tracked} + \rho_{ion}^{tracked} \tag{23}$$

$$\frac{d(\rho/\varepsilon_o)}{d\phi} = 0$$

6. UNDOCUMENTED FEATURES USED IN C/NOFS CALCULATION

6.1 Grounding Nodes and Edges

Though not used in the final calculations, grounding nodes and edges were used in the early design of the C/NOFS panels. By default, *Nascap-2k* (in the BEM module) treats both surface and bulk conductivity for insulators. The treatment of bulk conductivity through an insulator to its underlying conduction is fairly obvious, and will not be further discussed. Surface conductivity operates between insulating cells of a common material and with a common edge, thus covering transport over a wide expanse of such material. To ground the material, we can specify grounding by a strip at a cell edge, or by a circular contact located at a node. These grounding elements can be specified only for "Primitive" components, because other types of components frequently have their meshes recreated. A grounding edge is created by selecting a cell edge of a Primitive, and clicking the menuItem "Mesh|Conductivity|Add Conducting Edge." The grounding edge establishes conductance to ground from each of the two neighboring cells of $L/D\kappa$, where L is the length of the edge, D is the distance from the center of the cell to the center of the edge, and κ is the surface resistivity of the material. Similarly, a grounding node or grounding dot is created by selecting a node of a Primitive, and clicking the menuItem "Mesh|Conductivity|Add Conducting Node." The user must specify the radius of the grounding dot. Conductance from each of the neighboring cells to the grounding dot is $(\theta/\ln(d0/r))/\kappa$, where θ is the angle the cell subtends at the node, d0 is the distance from the cell center to the node, and r is the node radius.

6.2 Magnetically Induced Potentials

Magnetically induced ($v\times B$) potentials (henceforth VXB potentials) are of interest in LEO for very large spacecraft or for spacecraft with very high electrostatic cleanliness requirements. Typically, the most positive exposed conducting surfaces make contact with the plasma via electron collection, leaving the most negative exposed conducting surfaces far more negative than they might otherwise be.

The VXB potentials must be set both in the BEM module (via script command) and in the Potential Solver module (in the input file). Needless to say, both V and B must be available (in both places) to set the surfaces correctly.

In the script input to the BEM module, VXB potentials are typically set by a command sequence such as

```
<COMMAND cmd="SetBField" x="0.0" y="2.5e-5" z="2.5e-5" />
<COMMAND cmd="SetVelocity" x="7500" y="0.0" z="0.0" />
<COMMAND cmd="SetInitialPotentials" />
<COMMAND Value="-0.3" cmd="SetVXBPotentials" />
```

The "Value" associated with the SetVXBPotentials command corresponds to the maximum (usually least negative) potential on the conductor. The value stored as the conductor potential is the potential referenced to the origin (where $v\times B\cdot R = 0$). Since the cell coordinates are the same in Potent and BEM, Potent sets the potential of a conducting cell by simply adding vxb.R (where R is the centroid of the cell) to the conductor potential.

6.3 Running the C/NOFS Model

To run the C/NOFS full model, we first do a charging calculation with the BEM module, using a custom current DLL that implements the "EWB Plate" formulation. Then, two POTENT runs are needed: first, a "NEW" run that calculates the wake ion densities (GIs) and uses the "NONLINEAR" space charge formulation; second a "CONTINUE" run that uses the "SHEATH_WAKE" formulation. The SHEATH_WAKE formulation, for negative potentials, uses the "GI" ion densities and barometric electron densities which gives a plausible wake structure, whereas the "NONLINEAR" formulation does a poor job in the wake for low potentials. (The "SHEATH_WAKE" charge density formula is *not* available from the interface.)

7. ANTENNA CALCULATIONS

There is current interest in generating VLF waves in space for the purpose of controlling the trapped electron population.[15] A transmitting antenna for this purpose would be several inches in diameter, tens of meters long and have bias amplitudes of hundreds of volts. It would interact with a large volume of the surrounding plasma, and be a major driver for spacecraft potential. Thus, it is of major interest to be able to simulate dynamically and in three dimensions the antenna together with its host spacecraft and the surrounding plasma.

Nascap-2k, through its DynaPAC heritage, contains the features needed to do this type of problem. However, these features have not been fully exercised since the days of SPEAR II.[16] In studying the CHAWS experiment,[11] ion trajectories were used to calculate steady-state, self-consistent charge densities and space potentials, and we have exercised the ability to do PIC ions with barometric electron densities. For the antenna problem we would like to do both electrons and ions using PIC. (Note, however, that *Nascap-2k* only solves Poisson's equation, rather than the full Maxwell equations, so it computes only curl-free, quasi-static fields.)

Below we describe the antenna problem, and, for baseline parameters, show the solution, as calculated by a one-dimensional PIC code, for electron dynamics in the sheath. We then pose a nearly identical problem for solution with *Nascap-2k*, demonstrate that the solutions agree, and show the graphics.

7.1 Statement of Problem

The objective is to simulate the dynamic sheath around a negative thin rod. We avoid positive polarity because the positive half of the antenna will collect copious electrons, so the maximum potential it can reach is determined by numerous unknown factors, such as the relative size of the spacecraft and antenna. We wish to do this with realistic values of plasma density, ion mass, applied voltage, frequency, magnetic field, and spacecraft velocity. The arbitrary directions of the latter two require a three-dimensional code. A one-dimensional (radial) code can handle magnetic field either parallel to the antenna or circumferential (as would be caused by current flowing in the antenna).

7.2 Sheath Size Estimate

The sheath (defined as the region from which electrons are excluded) can be quite large, even for a fairly modest potential of about 100 volts. To calculate the sheath size, we specify the electric field at the antenna radius, R_0. We assume that the external space between the antenna and the sheath is filled with ions at ambient density, ρ. The electric field at any radius, r, between the antenna and the sheath is

$$E(r) = \frac{\rho e(r^2 - R_0^2)}{2\varepsilon_0 r} - \frac{a}{r}E(R_0) \tag{24}$$

The sheath condition is $E(R_s)=0$, where R_s is the sheath radius. We then integrate the electric field from R_s to R_0 to determine the corresponding potential.

Figure 16 shows the relation between applied potential and sheath radius for a 10 cm diameter antenna. At a density of 10^{12} m^{-3} the sheath radius at 100 V bias is about 15 cm, and grows to nearly a meter at a density of 10^{10} m^{-3}. The calculations to follow assume a density of 3×10^{11} m^{-3}, giving a sheath radius of about 20 cm.
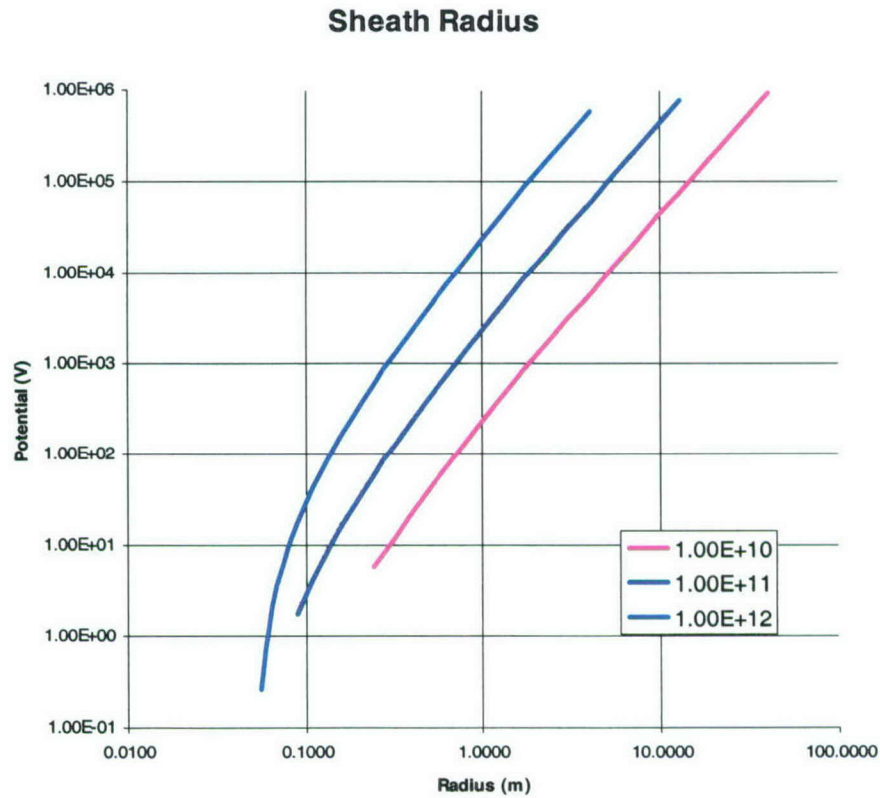
Figure 16. Potential vs. Sheath Radius for Negative Applied Potential on a 10 cm Diameter Antenna. Curves for Three Different Plasma Densities are Shown.

7.3 Baseline Parameters

Figure 16 shows the baseline parameters for the calculation. Density of 3×10^{11} m^{-3} is chosen so that the sheath is large compared with the wire but still very tractable computationally. In general the plasma is cold, but 0.1 eV is used in those places where temperature is required. The antenna frequency is set to 100 kHz, and the process is followed for a half-period of 5 µs. To see the effect of magnetic field, a field of 0.5 gauss is chosen. The ordering of the plasma frequency, electron gyrofrequency, and applied frequency is $\omega_p > \omega_c > 2\pi f$.

Table 5. Parameters for Baseline Calculations.

Plasma Density	3×10^{11} m^{-3}
Electron Temperature	0.0 or 0.1 eV
Plasma Frequency	3.1×10^7 s^{-1}
Antenna Frequency	100 kHz
Magnetic Field	0.0 or 0.5 gauss
Electron Gyrofrequency	0.0 or 8.8×10^6 s^{-1}
Ion Species	O^+

7.4 One-dimensional Calculations

A simple one-dimensional finite element code was written to simulate quasistatic plasmadynamics about a long cylindrical antenna. The computational domain extended out to one meter from an antenna radius of 5 cm, and was divided into 1000 zones in equal increments of r^2. Two ion macroparticles and two electron macroparticles were placed in each zone, with each macroparticle having equal charge. The simulation was run for 2000 timesteps of 2.5 ns each, making up the 5 µs half-period for the 100 kHz frequency. When electrons left the computational space they were replaced by thermal electrons.

Figure 17 shows the potential profile at various times in the calculation. As expected, the potential is rapidly screened to about the expected sheath radius as electrons are expelled from the sheath. At certain times a positive potential region appears. This is an effect of electron inertia, as the moving electrons do not stop of their own accord, but must be attracted back towards the sheath boundary. In Figure 18 through Figure 21 we plot (1) the maximum potential at times when a positive region appears; (2) the location of the maximum potential; and (3) the location of the sheath edge, indicated by a sharp drop in the charge density from nearly the ambient ion density to nearly zero.
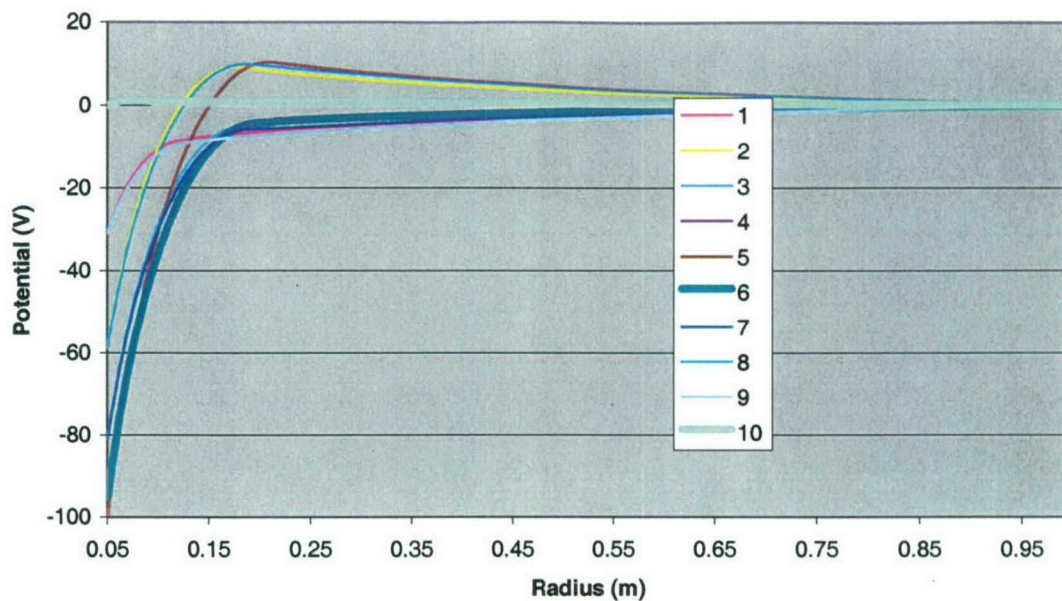
Figure 17. Potential Profile at Various Times During the One-dimensional Calculation.
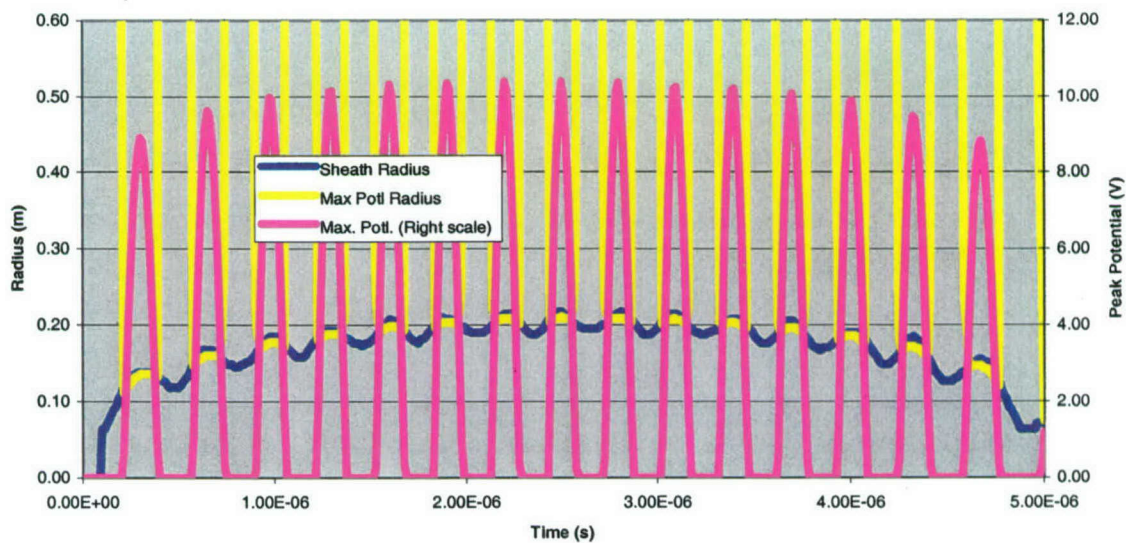

Figure 18. Simulation Results for Half Sine Wave and no Magnetic Field, Showing Peak Positive Potential (Magenta Curve, Right Scale), Location of Peak (Yellow Curve, Left Scale) and Location of Sheath Edge (Dark Curve, Left Scale).
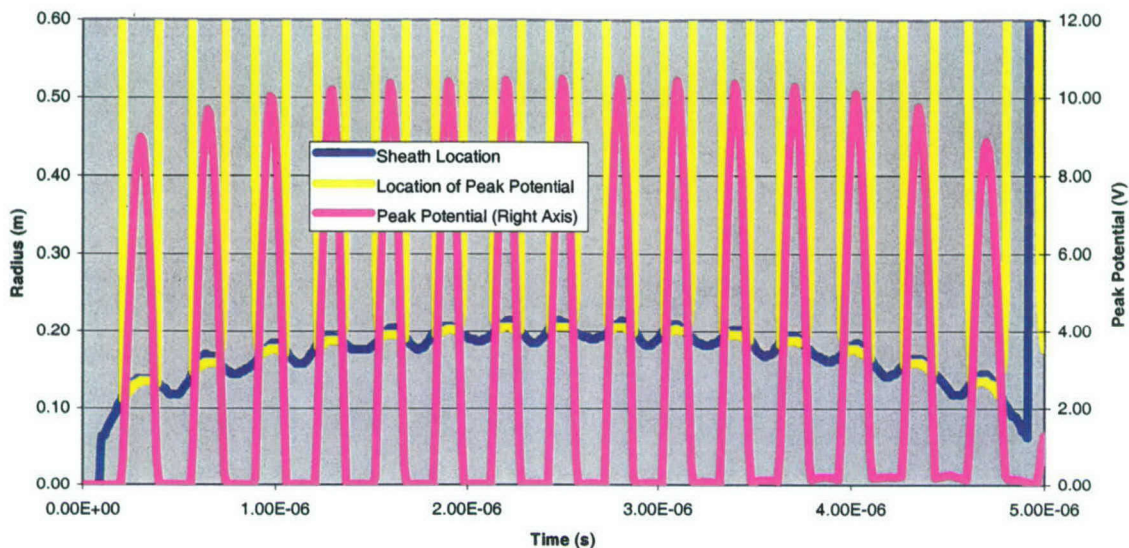
38

Figure 19. Same as Figure 18, for Half Sine Wave with Magnetic Field of 0.5 Gauss.

Figure 18 and Figure 19 show results for a half sine wave, for which the applied negative potential continuously rises and returns to zero. The magnitude of the potential maximum is 8 to 10 V, and the oscillation frequency is somewhat less than the electron plasma frequency. The sheath edge occurs at a radius of about 20 cm as calculated above, with oscillations of about two cm. The potential maximum, when it occurs, is just inside the sheath edge.
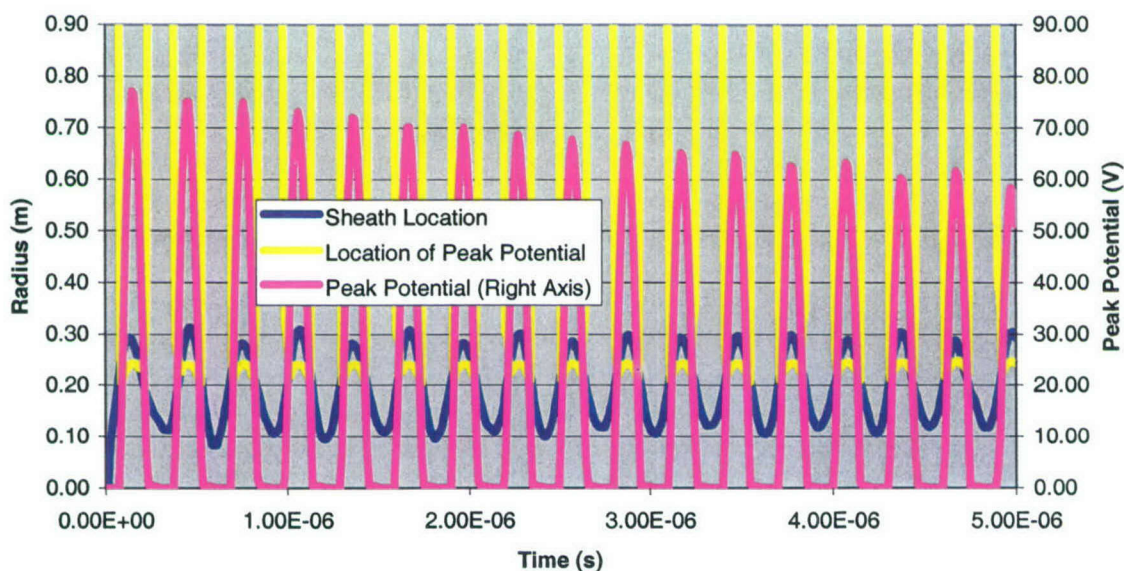
Figure 20. Same as Figure 18, for Square Wave and Zero Magnetic Field.
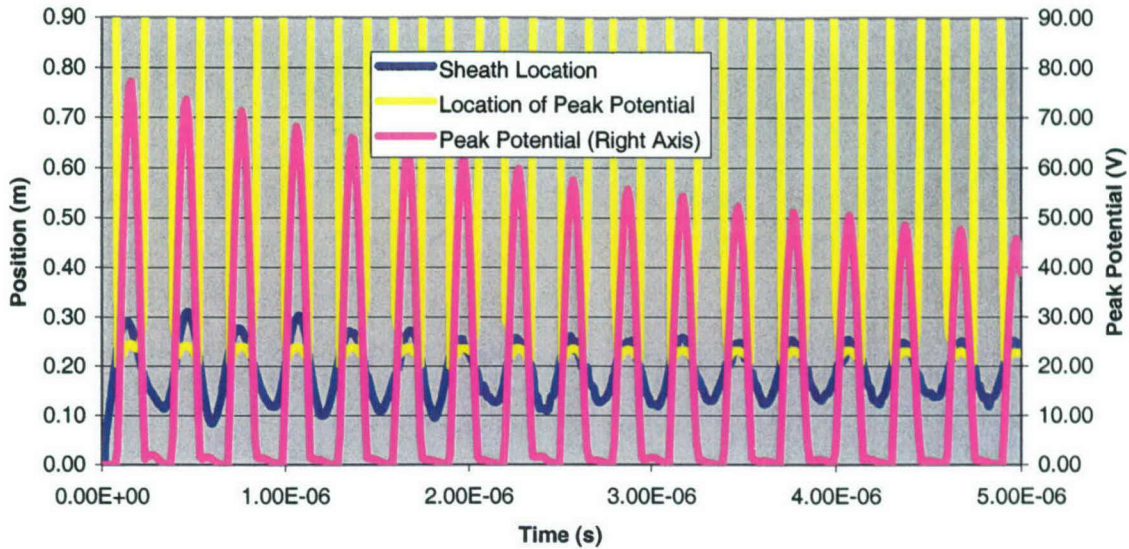
Figure 21. Same as Figure 18, for Square Wave With 0.5 Gauss Magnetic Field.

Two differences can be noted between the unmagnetized (Figure 18) and magnetized cases. First, in the unmagnetized case the potential goes completely non-positive between peaks, whereas in the magnetized case a positive peak forms inside the outer boundary. This occurs because the magnetic field inhibits inward diffusion of the thermal electrons. Second, in the unmagnetized case the sheath remains at the end of the pulse, even though the potential goes to zero, while in the magnetized case the sheath disappears. This occurs because the magnetic field traps inbound electrons in the sheath region, whereas in the absence of magnetic field, inbound electrons collide with the antenna.

Square wave calculations were done in one-dimensional calculations to compare with the *Nascap-2k* calculations below, and are shown in Figure 20 and Figure 21. Much stronger plasma oscillations are seen, with the initial oscillation at 75 V. The sheath edge oscillates 10 cm on either side of its average position at 20 cm, and the peak potential occurs well inside the sheath. Application of the 0.5 gauss magnetic field doubles the rate of decay of the oscillations.

7.5   Three-dimensional Calculations

Three-dimensional calculations were done with *Nascap-2k* to demonstrate the feasibility of such calculations. Figure 22 shows the *Nascap-2k* antenna model embedded in a nested grid. The antenna consists of two square rods, each 10 cm on a side and 4 m long. The outer boundary of the grid is a square 1.32 m on a side. The coarse resolution is 11 cm, with 5.5 cm resolution near most of the antenna, and 2.75 cm resolution in a limited region. Initially, 8 electron macroparticles and 8 ion macroparticles were placed in each zone, positioned to represent a uniform charge distribution in the context of the nonlinear interpolants. A negative 100 V square wave was applied to half of the antenna, and each timestep consisted of: (1) tracking the particles for 2.5 ns, (2) sharing the particle charge to the nodal coefficients in accordance with the nonlinear interpolants, and (3) recalculating the potential in preparation for the next tracking phase.
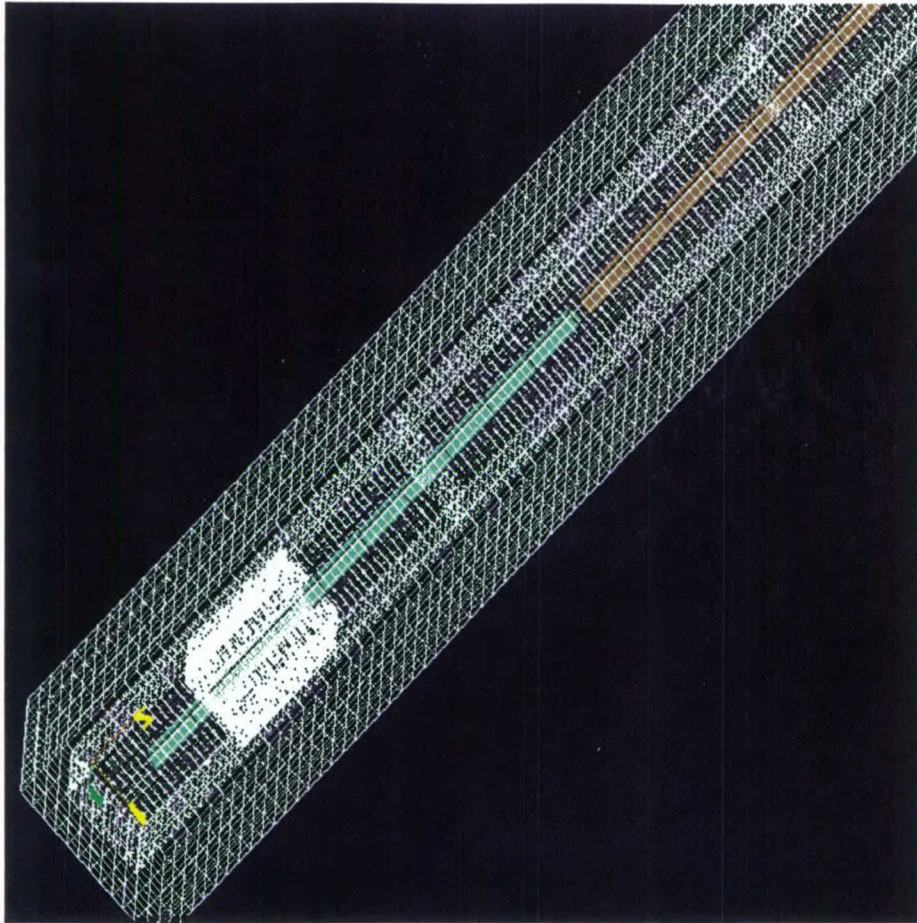
Figure 22. *Nascap-2k* **Antenna Model, Showing Antenna and Gridding.**

The initial state is represented in a three-dimensional view in **Figure** 23, and a planar view in Figure 24. Figure 24 through Figure 27 shows a plane of potentials with a plane of electron macroparticles just above it, positioned as shown in Figure 8. (Ion macroparticles have the same initial configuration, but move negligibly during the simulation time.) Note that the apparent high density of particles in the subdivided region is balanced by correspondingly reduced particle weight. Comparing Figure 25 with Figure 24, we see that the potentials change little in the first 25 ns, but they change considerably in the next 25. Figure 26 shows the electron motion, leading to sheath radii of 9 cm at 25 ns and 18 cm at 50 ns. Figure 26 shows the effects of the square cross-section. Particles that started out near the flat, low-field region have moved considerably less than those that started out near the high-field corners.
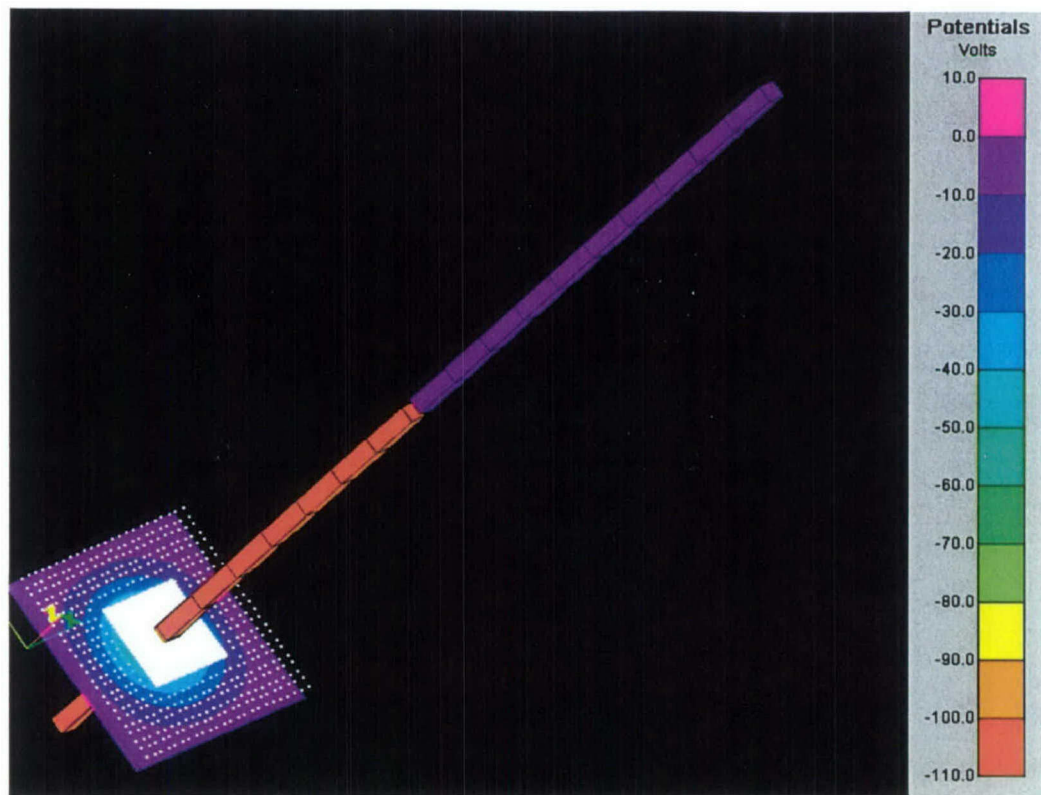
Figure 23. *Nascap-2k* **Antenna Model Showing Potentials and Particle Positions After 2.5 ns.**
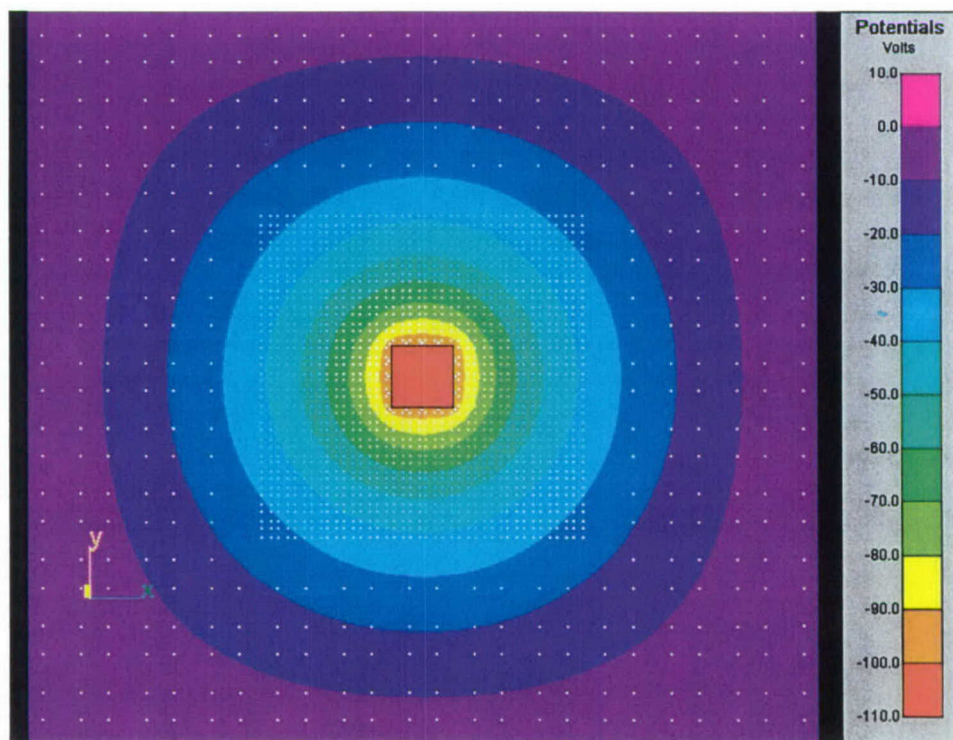

Figure 24. Planar View of Initial Potentials and Particles, as Shown in Figure 23.
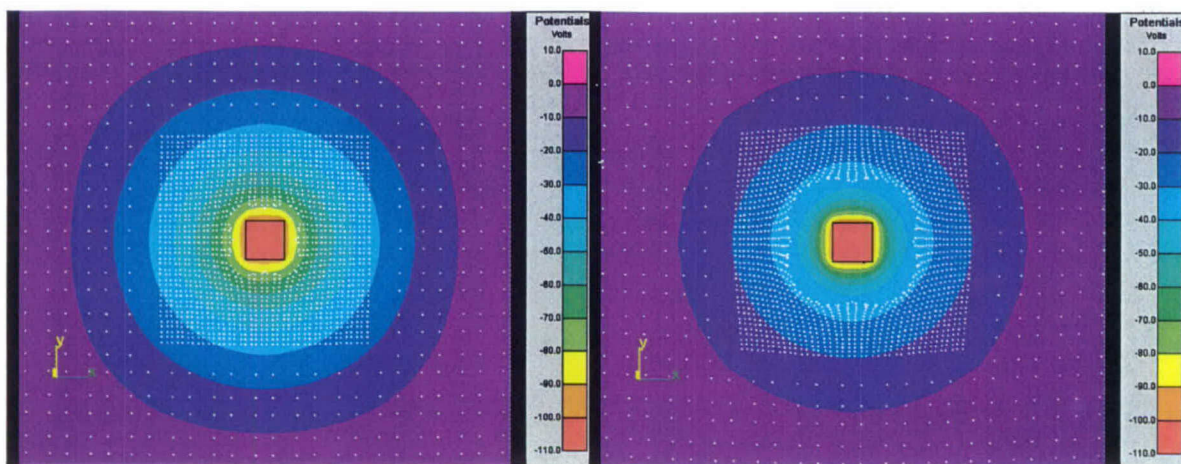
Figure 25. Particles and Potentials After 25 ns (left) and 50 ns (right).
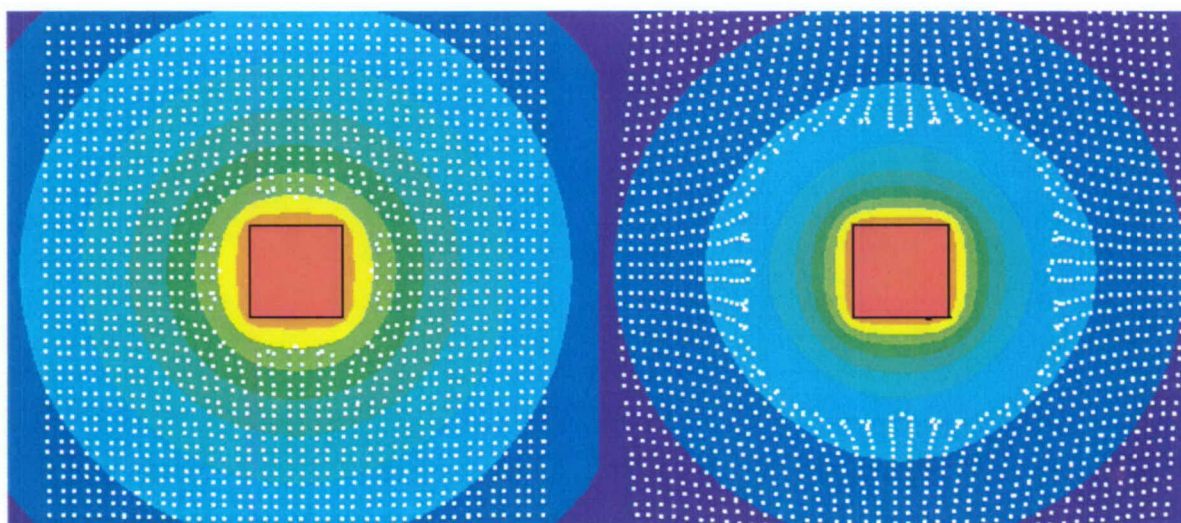

Figure 26. Blowup of Figure 25, Showing a Sheath Radius of 9 cm After 25 ns (left), and About 18 cm After 50 ns (right).
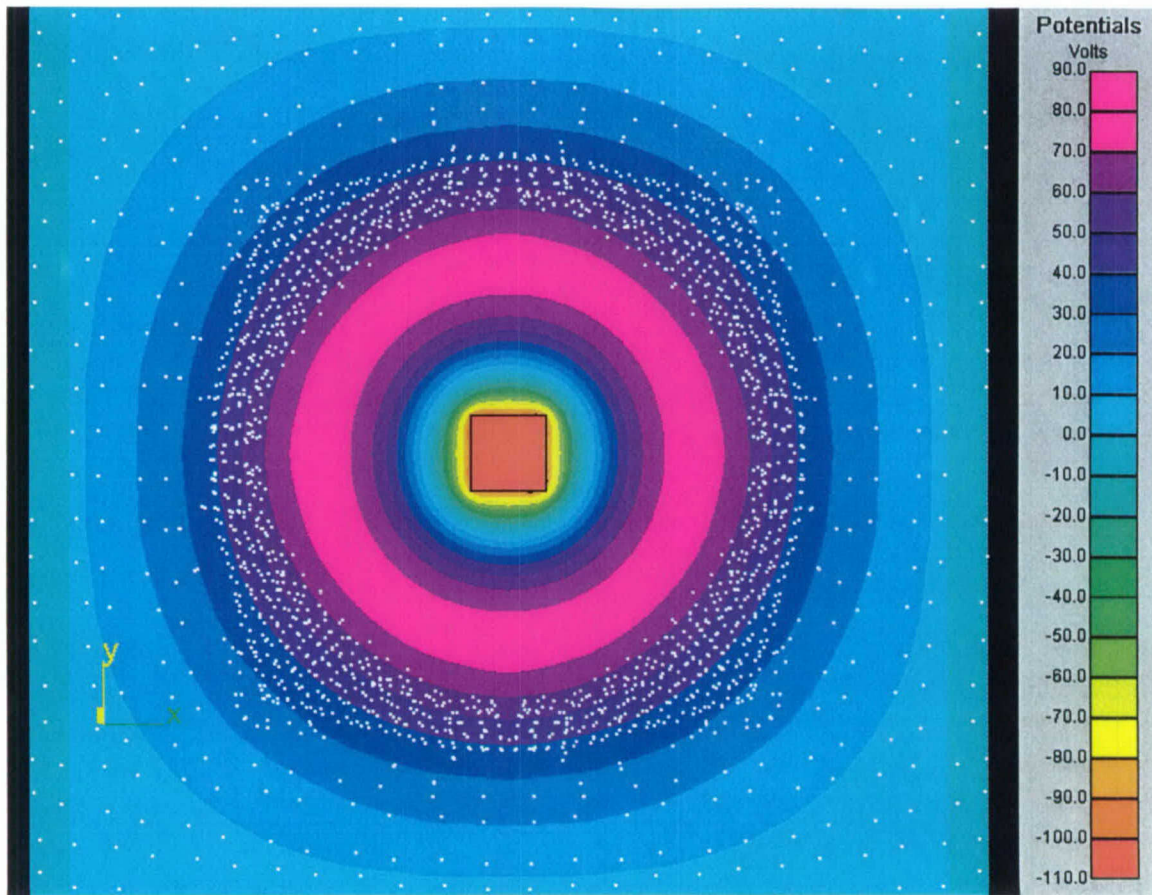
Figure 27. Potentials and Particles at Time of Maximum Positive Potential.

The simulation was run up to the first potential maximum, which occurred at 137.5 ns. Figure 27 shows that at this time, there is a high, broad maximum in the potential, with electrons excluded from a region that extends well beyond the location of the potential maximum. Figure 28 shows another view of the final configuration, with potentials in a plane containing the antenna. Note that there is no apparent difference between the potentials in the highly resolved region and in the less resolved region, suggesting that the highest level of resolution may not be needed.
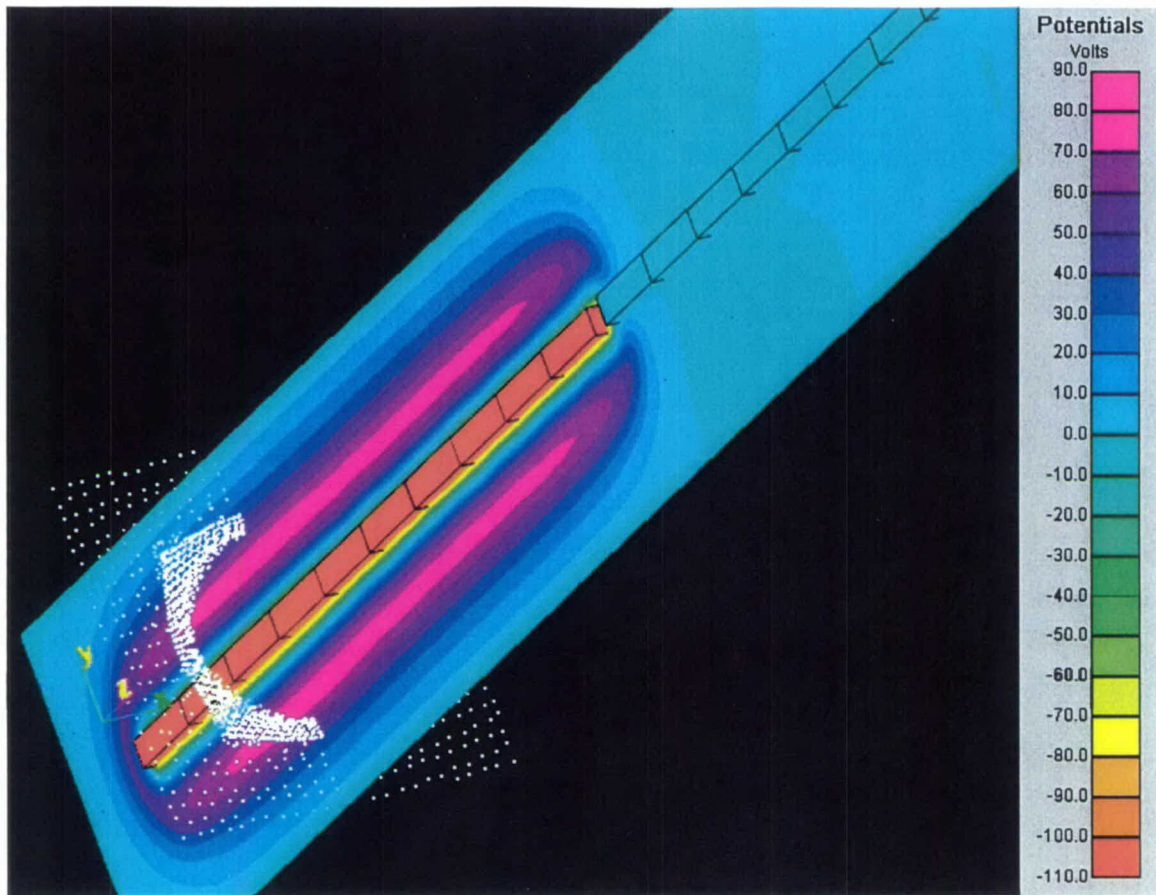
Figure 28. Another View of the Final Configuration (at 137.5 ns), Showing Potentials in a Plane Containing the Antenna.

Figure 29 shows the development of the sheath radius and maximum potential calculated by *Nascap-2k*, compared with the one-dimensional sheath radius result. The maximum potential of 75 V, as well as the time of first maximum (140 ns) is in excellent agreement with the one dimensional result. The maximum sheath radius calculated by *Nascap-2k* is larger than the one-dimensional result in proportion to the effective larger size of the 10 cm square antenna vs. the 10 cm diameter round antenna.
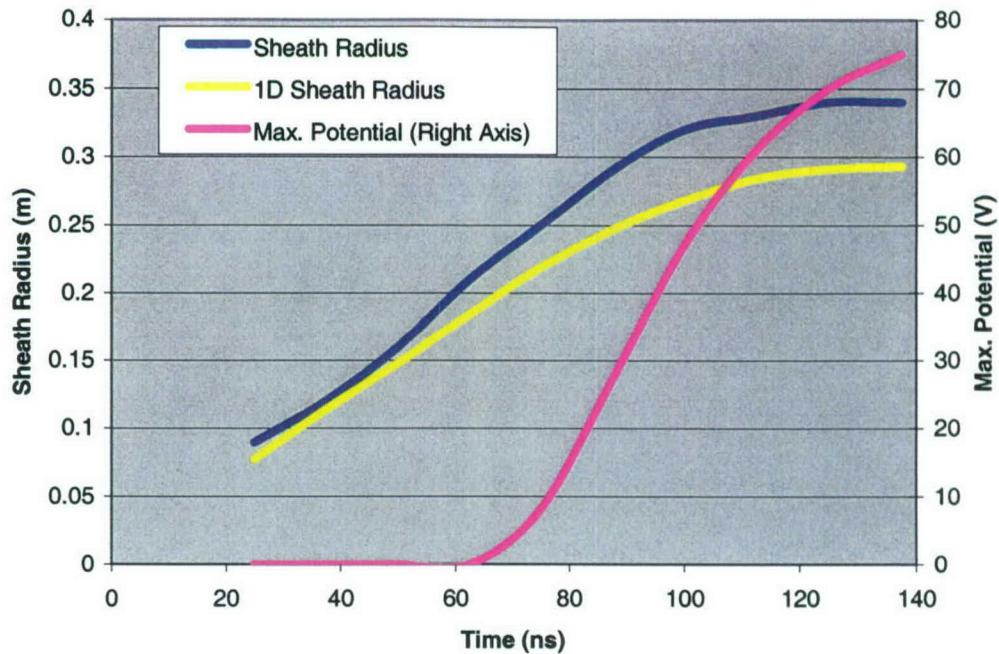
Figure 29. *Nascap-2k* **Results for Sheath Radius (Dark Curve) and Maximum Potential (Magenta Curve, Right Scale) Compared With One-dimensional Sheath Radius Results (Yellow Curve).**

7.6 Additional Calculation

We also attempted a more ambitious antenna calculation than above. The intent was to show that a calculation could be performed for parameters more nearly resembling the frequency and plasma density of interest for VLF RBR. The old and new calculations are contrasted in Table 6.

Table 6. Contrasting Parameters for the Old and New Calculations.

Parameter	September 2003 Calculation	December 2003 Calculation
Frequency	100 kHz	20 kHz
Plasma Density	3×10^{11} m^{-3}	1×10^{10} m^{-3}
Plasma Frequency	4.92 MHz	898 kHz
Antenna Radius	5 cm	10 cm
Peak applied potential	100 V	1000 V
Waveform	Square wave	Sine Wave
Sheath radius	34 cm	2 m
Timestep	2.5 ns	100 ns
Time Simulated	0.140 μs	8.3 μs
Antenna Length	4 m	20 m
Spacecraft Body	4 m antenna	Cylinder 2.5 m diameter 0.8 m high
Initial No. of Electrons (Ions)	350,000	966,000

46

In both cases, the spacecraft body (or positive portion of antenna) was grounded, as static calculations indicated that electron currents would limit the body to negligible positive potentials. Both electrons and O^+ ions were tracked, and the potentials solved at each timestep.

The calculation was carried out to 8.7 μs, at which point the antenna was at negative 888 V (-888 V) on its way to its peak negative potential of 1 kV. Figure 30 through Figure 35 show a sequence of potentials and electron positions. Figure 36 shows the ion positions at the conclusion of the calculations.

Figure 30 through Figure 32 show a reasonably orderly growth of the sheath at early times, as evidenced by exclusion of electrons from a cylindrical region around the antenna. Figure 31 (at 1.2 μs) occurs about one plasma period into the calculation, and shows a large region of positive potential. The peak potential is about 45 V, which occurs near the inner edge of the positive potential region. Thus, all the electrons shown in the picture are being accelerated inward. The sheath is still well-defined at 3.9 μs (Figure 32), but the initial orderly pattern of electrons is no longer evident.

Figure 33 through Figure 35 no longer show a well-defined sheath. The effects of choosing too long a timestep now dominate the simulation, as electrons move substantial distances before their motion is reflected in a change in potential. A spike of returning electrons strikes the antenna at about 6.2 μs, with subsequent spikes at intervals approximating a plasma period. While a magnetic field might suppress these spikes, I believe they are unphysical even with zero magnetic field, but result from too long a timestep. Figure 35 (at the trailing edge of such a spike) shows the mechanism. A region more negative than the antenna itself forms. Necessarily, this region contains an excess of electrons. These electrons are then accelerated toward the antenna.

Figure 36 shows the ion positions at the conclusion of the calculation. As expected, ions within a few antenna diameters of the antenna have moved considerably, with just the beginnings of motion beyond. The original pattern is clearly visible. Ion macroparticles begin to strike the antenna at about three μs. Beyond this point the average ion current to the antenna is about a half milliampere.

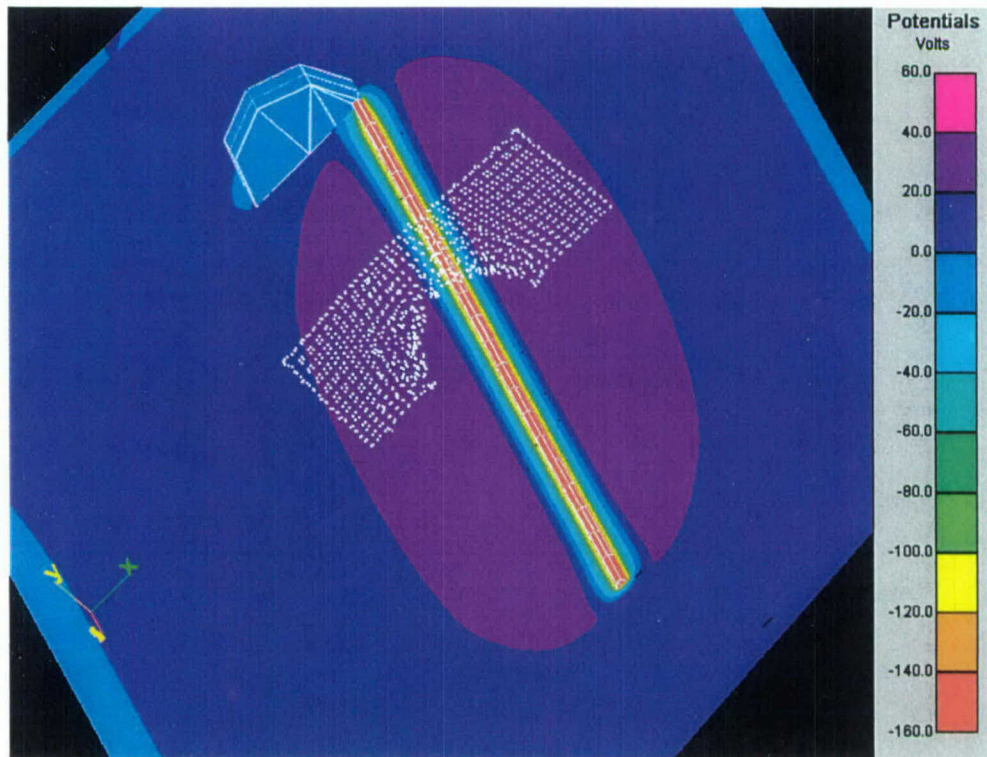**Figure 30. Potentials and Electron Positions at 0.3 µs, Antenna at 38 V.**

Figure 31. Potentials and Electron Positions at 1.2 µs, Antenna at 150 V.

48

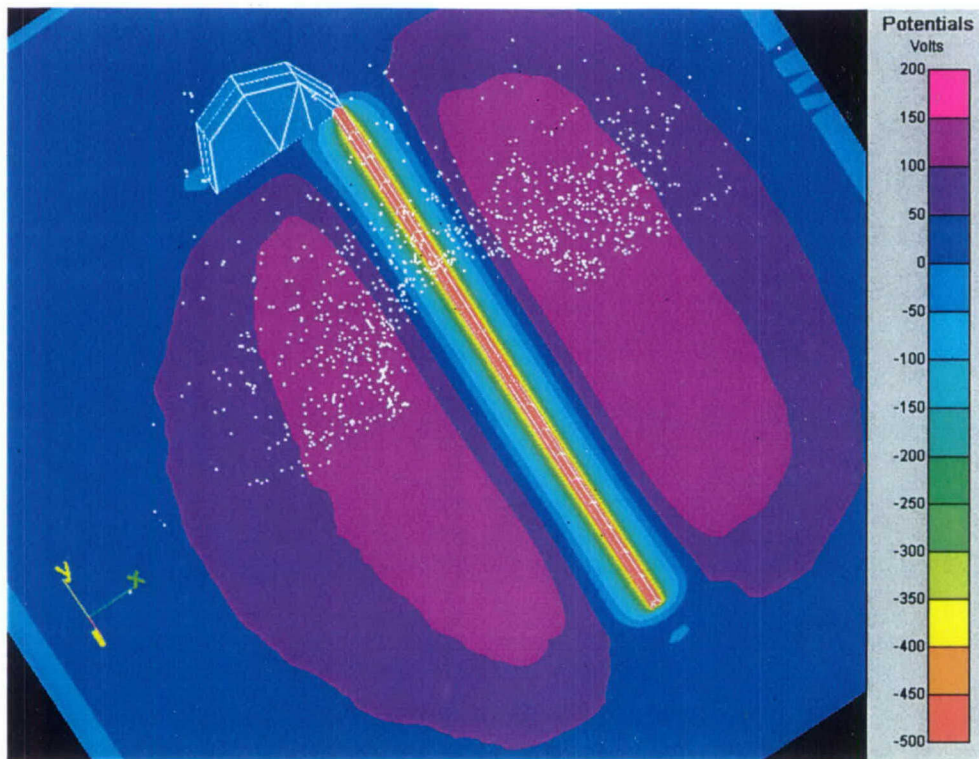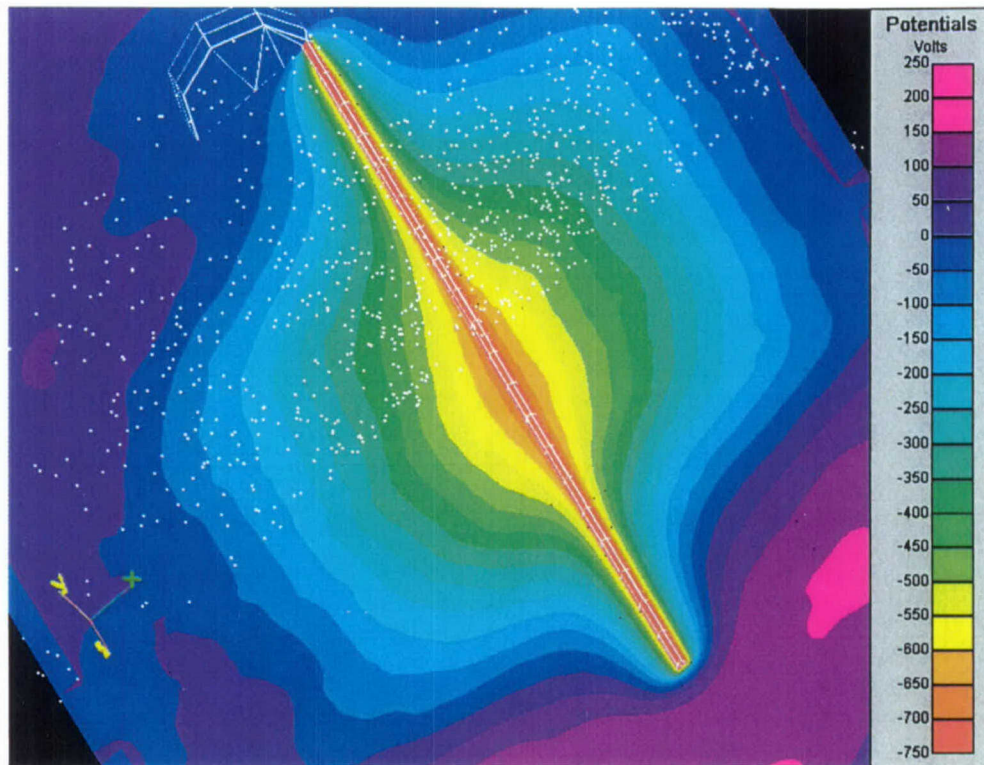**Figure 32. Potentials and Electron Positions at 3.9 μs, Antenna at 471 V.**

Figure 33. Potentials and Electron Positions at 5.1 μs, Antenna at 598 V.

49

Figure 34. Potentials and Electron Positions at 6.3 µs, Antenna at 712 V.
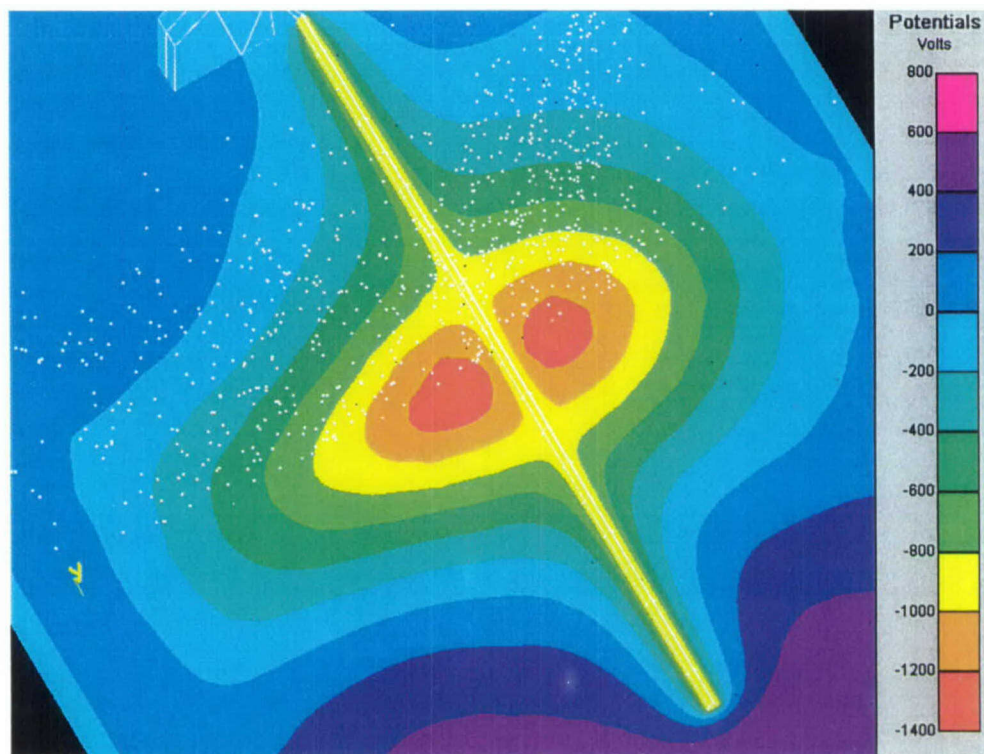

Figure 35. Potentials and Electron Positions at 8.7 µs, Antenna at 888 V.
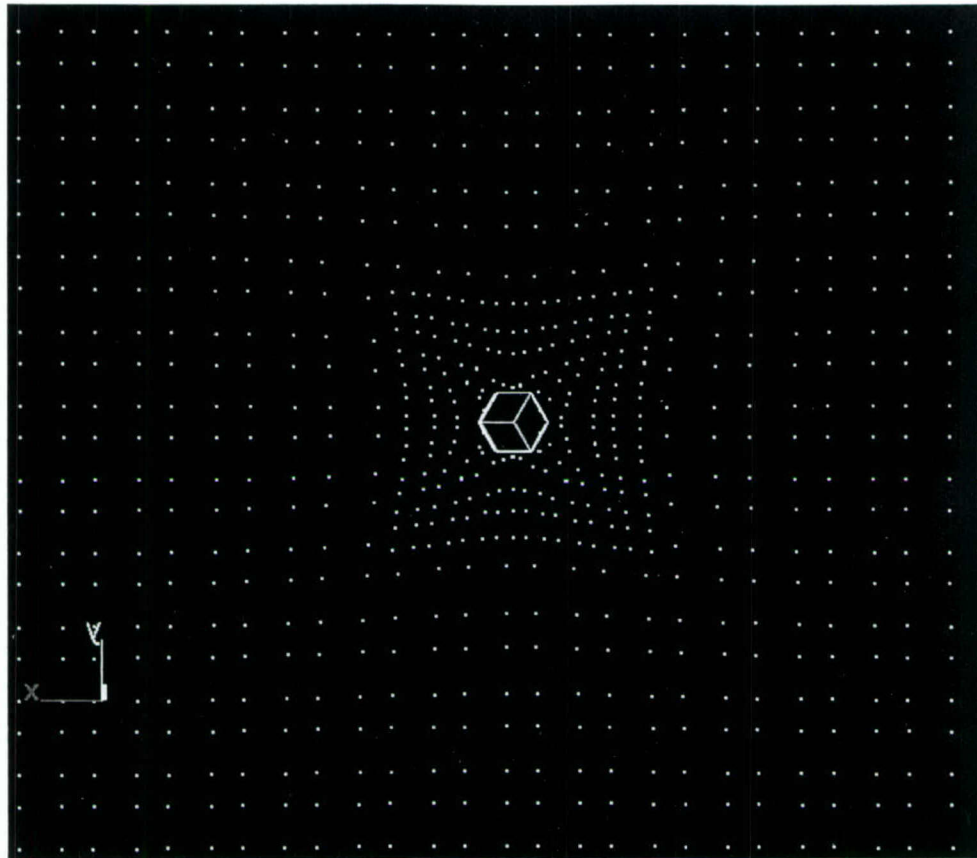
Figure 36. Ion Positions at 8.7 μs.

7.7   Conclusions

Electron dynamics are important in the very large sheaths of VLF antennas under space conditions. Large electrostatic oscillations are to be expected, producing at times positive potential regions about a negative antenna. The amplitude of the oscillations is waveform dependent.

Nascap-2k is shown to have the ability to perform quasi-electrostatic PIC simulations of sheath dynamics in fully three-dimensional geometry. This is important because such simulations can take into account the presence of the host spacecraft and arbitrary directions of magnetic field and spacecraft velocity.

The ultimate failure of the second set of calculations was due to the user's choosing too long a timestep. The chosen timestep of 0.1 μs is appropriate for the initial, low potential phase of the calculation (say, below 100 V), but the timestep needs to be reduced (say, to 0.025 μs) for the remainder of the simulation. Note that this simulation was done on a two year old (1.8 GHz) Windows PC. The availability of faster Windows computers, or perhaps computers running LINUX or UNIX, would also increase the feasibility of performing such simulations using *Nascap-2k*.

51

8. PRESENTATION ON STEREO

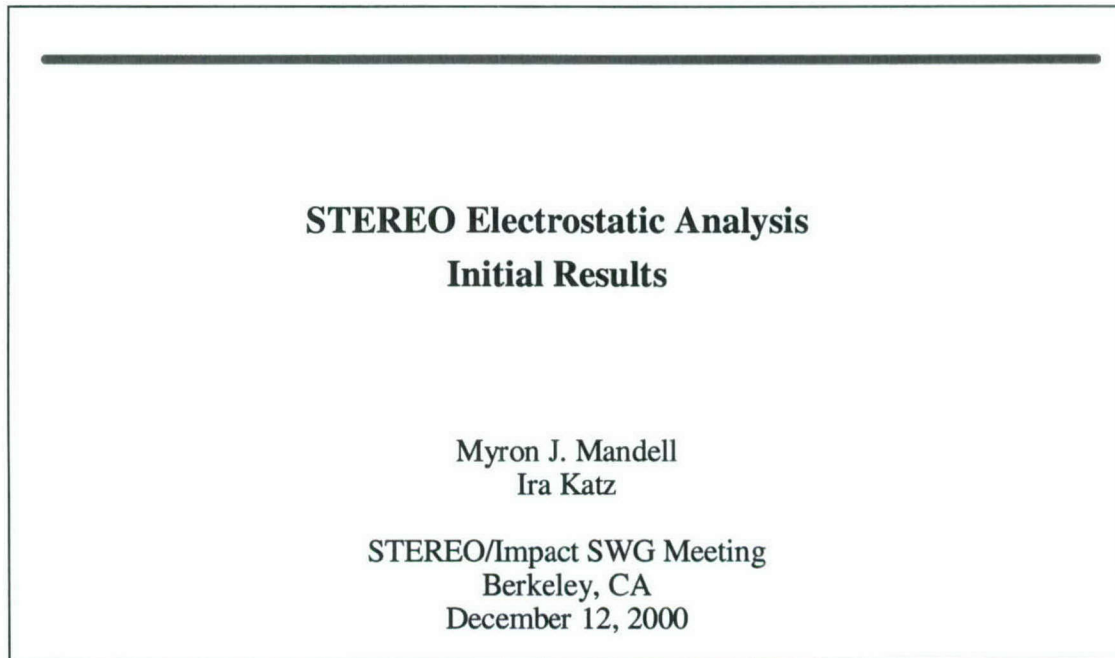The presentation shown in Figure 37 through Figure 50 summarizes our work on the STEREO spacecraft analysis.

STEREO Electrostatic Analysis
Initial Results

Myron J. Mandell
Ira Katz

STEREO/Impact SWG Meeting
Berkeley, CA
December 12, 2000

Figure 37. Presentation on STEREO Slide 1, Title.

Outline

- Solar Wind Charging Environment
- Nascap-2K Model for STEREO
- Overall and Differential Charging
- Conductivity of CMX Coverslips
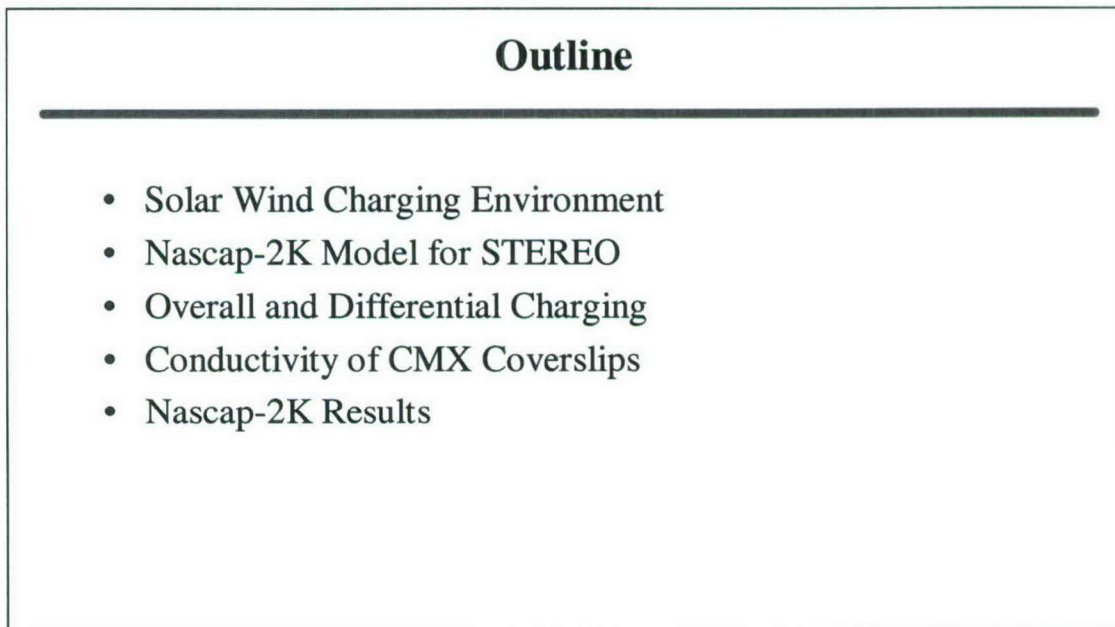- Nascap-2K Results

Figure 38. Presentation on STEREO Slide 2, Outline.

Solar Wind Charging Environment

- Sunlight causes photoemission
  - Effective photoemission decreases with potential
- Protons are beaming from sun direction
  - v ~ 400 km/s
  - E ~ 800 eV
- Electrons are isotropic
  - Plasma Density ~ 10^6 m^{-3}
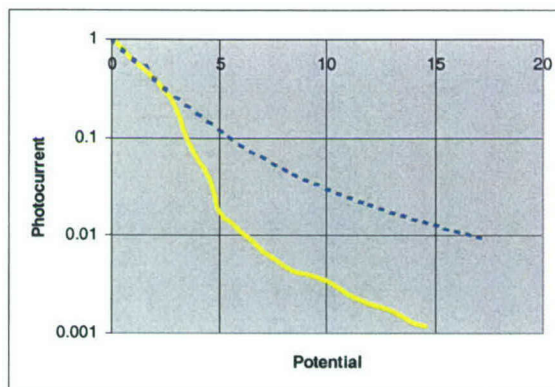 - Plasma Temperature ~ 3 eV
  - Debye Length ~ 10 m

Figure 39. Presentation on STEREO Slide 3, Solar Wind Charging Environment.
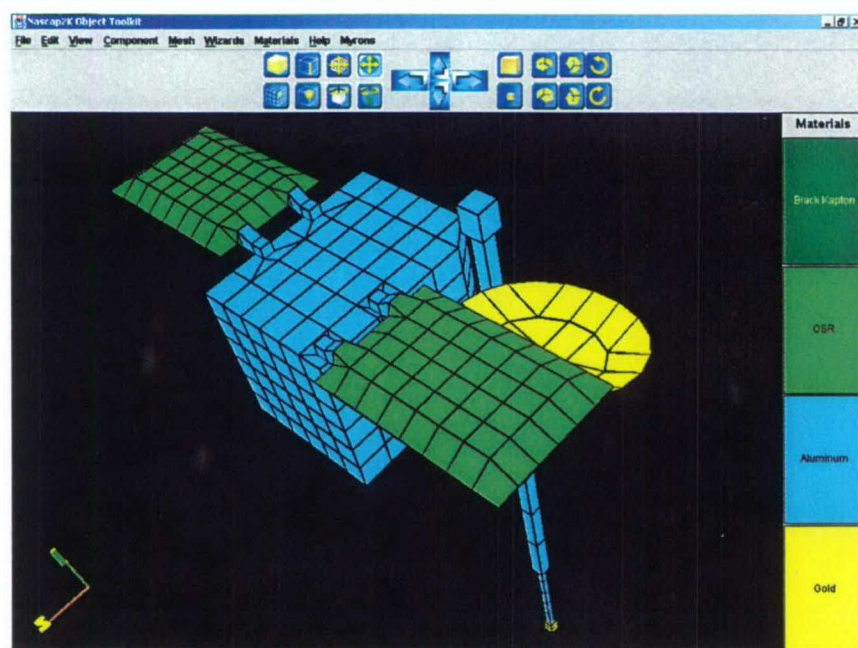
Nascap-2K Model

Figure 40. Presentation on STEREO Slide 4, Nascap-2k Model.
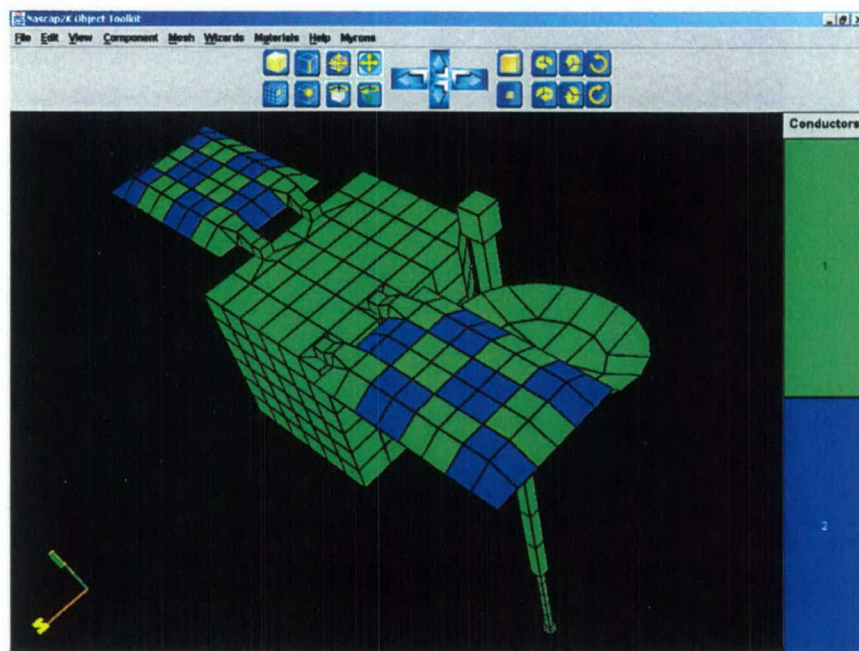
Nascap-2K Model

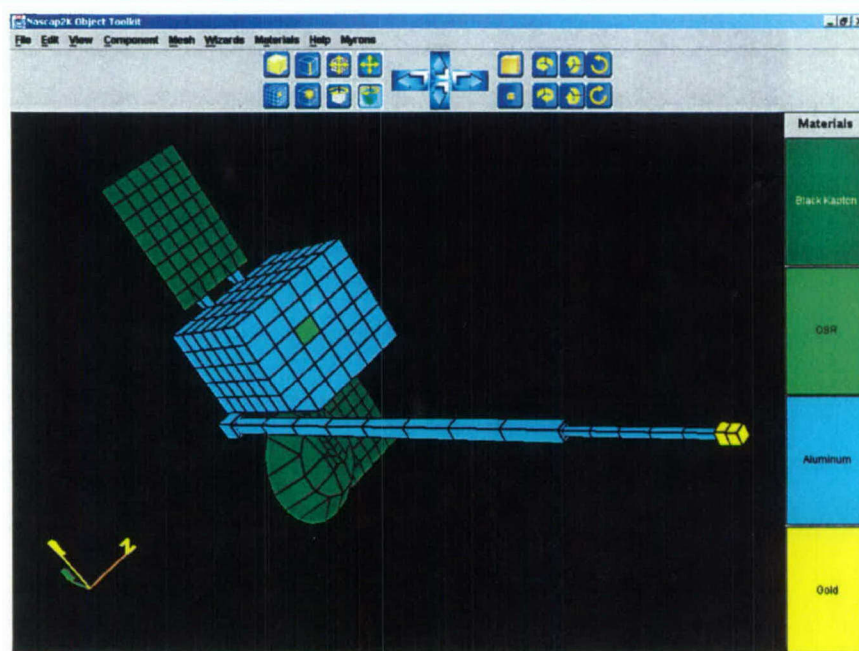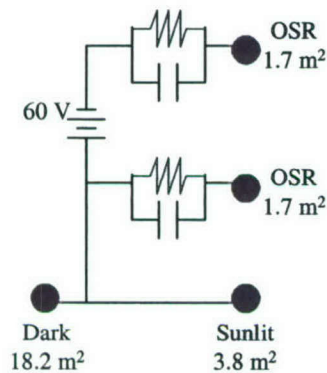Figure 41. Presentation on STEREO Slide 5, Nascap-2k Model View 2.

Nascap-2K Model

Figure 42. Presentation on STEREO Slide 6, Nascap-2k Model View 3.

Circuit Analysis

Overall Charging

Electron Current = $A_{total}J_e(1+V/T)$
Ion Current = $A_{sunlit}Nev$
Photocurrent = $A_{sunlit}I_{ph}(0)f(V,E)$

Capacitance is small, so s/c rapidly reaches potential for zero total current.

OSR 1.7 m²

60 V

OSR 1.7 m²
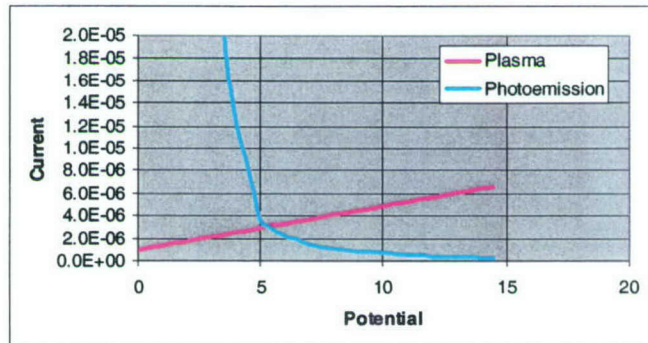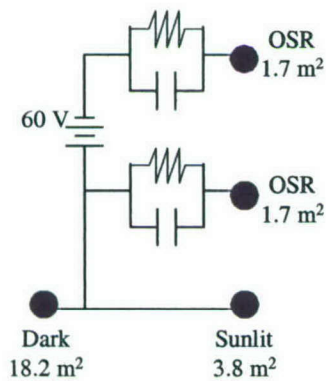
Dark 18.2 m²   Sunlit 3.8 m²

Figure 43. Presentation on STEREO Slide 7, Circuit Analysis, Overall Charging.

Circuit Analysis

OSR 1.7 m²

60 V

OSR 1.7 m²

Dark 18.2 m²   Sunlit 3.8 m²

Differential Charging

At uniform floating potential
    Sunlit insulators have positive current
    Dark insulators have negative current
    Coverslips may have positive
                conduction current

$dV/dt = J/C$
$J \sim 3\times10^{-8}$ Am^{-2}
$C \sim 3\times10^{-7}$ Fm^{-2}
$dV/dt \sim 0.1$ Vs^{-1}
Charging rate will decrease with time

Figure 44. Presentation on STEREO Slide 8, Circuit Analysis, Differential Charging.

CMX Conductivity

- Conductivity of CMX coverslips will bring some coverslips to elevated potential.
- We tend to believe higher of two data sources.
- Nominal resistivity taken as 10^{13} ohm-m
- $J = 60 \text{ V} \times 10^{-13} / 1.5 \times 10^{-4}$

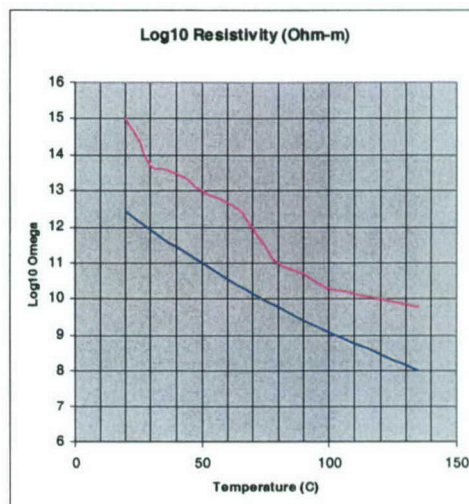- $J = 4 \times 10^{-8}$ A-m^{-2} (comparable to environment currents)

Figure 45. Presentation on STEREO Slide 9, CMX Conductivity.

Nascap-2K Results

Environment:

$n_e = 1 \times 10^6$ m^{-3} $E_i = 777$ eV

CMX Resistivity:	10^{17}	10^{13}	10^{12}	10^{13}
				$n_e = 1 \times 10^5$ m^{-3}
OSR/60	8.1	9.0	20.18	23.5
OSR/0	8.0	7.9	7.55	14.4
Ground	4.4	4.3	4.2	8.0

Figure 46. Presentation on STEREO Slide 10, Nascap-2k Results.

ITO Coating

- ITO Coatings are commonly grounded to local interconnects
- Surface potential variation reflects solar cell voltage pattern
- Positive cells will have enhanced electron collection, and may drive spacecraft negative

Figure 49. Presentation on STEREO Slide 13, ITO Coating.

Summary

- Under normal conditions
  - Chassis is a few volts positive
  - Coverslips have a few volts positive differential

- Get some differential charging if
  - CMX is conductive (resistivity $< 10^{13}$ ohm-m) (or hot)
  - Solar wind becomes tenuous

- Small insulating patches in the dark can charge to -5 volts or more. Thus far, this does not seem to indicate a problem.

- ITO coating predicted to have little or no benefit, and may be harmful to mission.

Figure 50. Presentation on STEREO Slide 14, Summary.
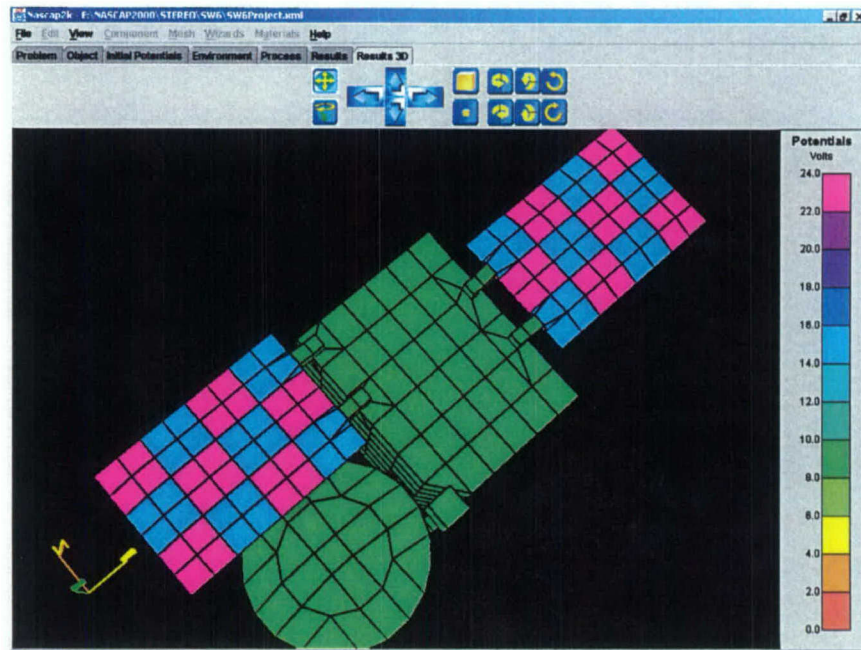
STEREO Potentials at Low Density

Figure 47. Presentation on STEREO Slide 11, STEREO Potentials at Low Density.

Low Density Charging History

Current Balance

$$\text{Conduction} = 45 \times 10^{-13} / 1.5 \times 10^{-4} = 3 \times 10^{-8}$$

$$\text{Electrons} = -4.6 \times 10^{-9} \times (1 + 24/3) = -4 \times 10^{-8}$$

$$\text{Protons} = 4 \times 10^5 \times 10^5 \times 1.6 \times 10^{-19} = 6.4 \times 10^{-9}$$

$$\text{Photocurrent} \sim 4 \times 10^{-9}$$
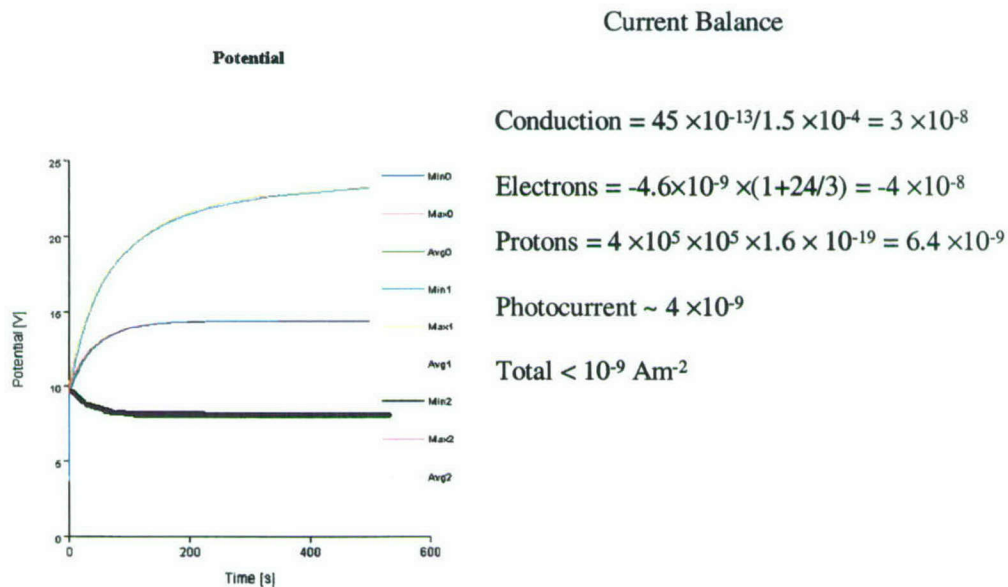
$$\text{Total} < 10^{-9} \text{ Am}^{-2}$$

Figure 48. Presentation on STEREO Slide 12, Low Density Charging History.

9. RAPID ALERT CHARGING TOOL

We used MSFM-generated fluxes to compute expected charging levels for a spacecraft at the location of DSCS for a known charging event. The net flux is given by the integral over the energy spectrum of the incident electrons and ions, along with secondary, and backscattered electrons. The figures below show a preliminary comparison for an event on day 217 of 1996. Yield parameters appropriate for carbon were used. A more complete description of our calculations appears in Reference 17.

$$\text{Net Flux} = \int_{0}^{\infty} \left(1 - Y_{see} - Y_{back}\right) j_e - \left(1 + Y_{sei}\right) j_i$$
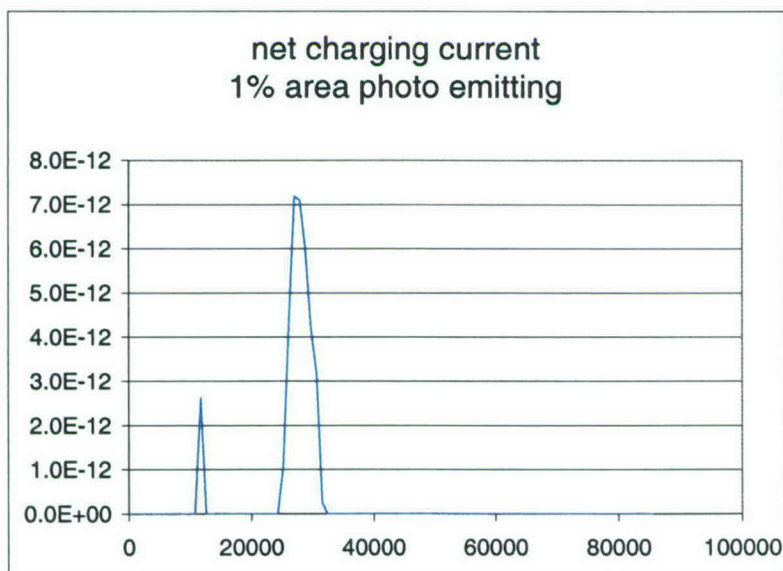

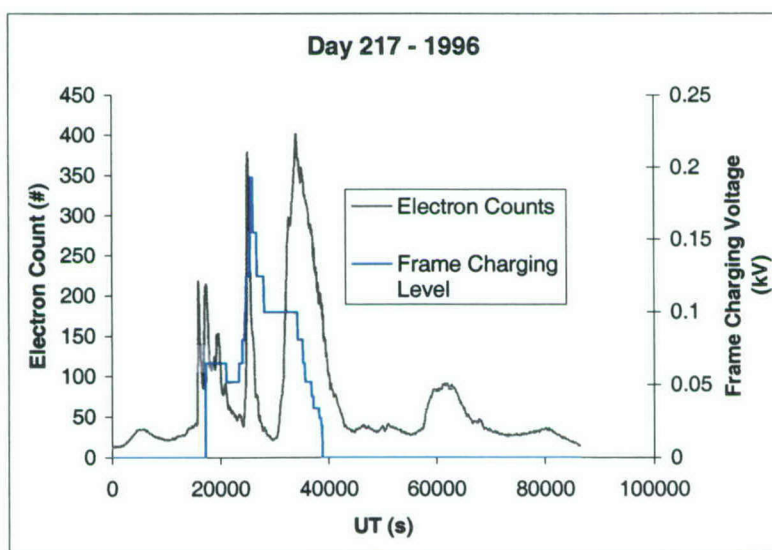**Figure 51. Net Charging Current.**

Figure 52. Electron Count.

59

REFERENCES

[1] M. J. Mandell, P. R. Stannard, I. Katz, 1993, *NASCAP Programmer's Reference Manual,* NASA CR 191044.

[2] P. R. Stannard, I. Katz, L. Gedeon, J.C. Roche, A.G. Rubin, M.F. Tautz, 1982, *Validation of the NASCAP Model using Spaceflight Data,* AIAA Paper AIAA-82-0269, presented at the 1982 Aerospace Sciences Meeting in Reno, NV.

[3] I. Katz, P. R. Stannard, L. Gedeon, J.C. Roche, A.G. Rubin, M.F. Tautz, 1983, NASCAP Simulations of Spacecraft Charging of the SCATHA Satellite, *Spacecraft/Plasma Interactions and Their Influence on Field and Particle Measurements,* ES SP-198, edited by A. Pedersen, D. Guyenne, and J. Hunt, European Space Agency, Noordwijk, The Netherlands, p. 190.

[4] M.J. Mandell and V.A. Davis, 1990, *User's Guide to NASCAP/LEO,* SCUBED report number SSS-R-85-730-R2.

[5] M.J. Mandell, and I. Katz, 1990, *High Voltage Plasma Interactions Calculations Using NASCAP/LEO,* AIAA Paper AIAA-90-0725, presented at the 1990 Aerospace Sciences Meeting in Reno, NV.

[6] J.R. Lilley, Jr., D.L. Cooke, G.A. Jongeward, I. Katz, 1989, *POLAR User's Manual,* GL-TR-89-0307, Oct.

[7] M.J. Mandell, I. Katz, J.M. Hilton, J. Minor, D.L. Cooke, 2001, *Nascap-2k, A Spacecraft Charging Analysis Code for the 21st Century,* AIAA Paper AIAA-2001-0957, presented at the 2001 Aerospace Sciences Meeting in Reno, NV.

[8] M.J. Mandell, T. Luu, J. Lilley, G. Jongeward, and I. Katz, 1992, *Analysis of Dynamical Plasma Interactions with High Voltage Spacecraft,* (2 volumes), Rep. PL-TR-92-2258, Phillips Lab., Hanscom Air Force Base, MA.

[9] I. Katz, G.A. Jongeward, V.A. Davis, M.J. Mandell, R.A. Kuharski, J.R. Lilley, Jr., W.J. Raitt, D.L. Cooke, R.B. Torbert, G. Larson, and D. Rau, 1989, Structure of the Bipolar Plasma Sheath Generated by SPEAR I, *Journal of Geophysical Research, 94,* A2, p. 1450.

[10] M.J. Mandell, G.A. Jongeward, D.L. Cooke, W.J. Raitt, 1998, SPEAR 3 flight analysis: Grounding by neutral gas release and magnetic field effects on current distribution, *Journal of Geophysical Research, 101,* A1, p. 439.

[11] V.A. Davis, M.J. Mandell, D.L. Cooke, and C.L. Enloe, 1999, High-voltage interactions in plasma wakes: Simulation and flight measurements from the Charge Hazards and Wake Studies (CHAWS) experiment, *Journal of Geophysical Research, 104,* A6, p. 12445.

[12] S.A. Brebbia, 1981, *Boundary Element Methods,* Springer Verlag, New York.

[13] L.W. Parker and E.C. Sullivan, 1974, NASA Report No. TN-D-7409.

[14] L.W. Parker, 1977, Calculation of sheath and wake structure about a pillbox-shaped spacecraft in a flowing plasma, *Proceedings of the Spacecraft Charging Technology Conference*, AFGL-TR-77-0051, NASA TMX-73567.

[15] U.S. Inan, T.F. Bell, J. Bortnik and J.M. Albert, 2003, Controlled precipitation of radiation belt electrons, *Journal of Geophysical Research 108*, No. A5, SMP 6-1, May.

[16] H.A. Cohen, F. M. Lehr and T. G. Engel (eds.), 1995, *Spear II – High Power Space Insulation*, Lubbock: Texas Tech University Press.

[17] I. Katz, V. A. Davis, M. J. Mandell, D. L. Cooke, R. Hilmer, L. Habash Krause, 2000, *Forecasting satellite charging: Combining space weather and spacecraft charging,* AIAA Paper AIAA-2000-0369, presented at the 2000 Aerospace Sciences Meeting in Reno, NV.