To the Graduate Council:

I am submitting herewith a dissertation written by Quan Zhang entitled "Particle Simulation for 10.6 μm Picosecond Laser-Induced Breakdown in Gases."I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Engineering Science and Mechanics.

Dennis Keefer, Major Professor

We have read this dissertation
and recommend its acceptance:

Frank A. Collins

Horace W. Crater

Dwayne McCay

Charles C. Limbaugh

Y. C. L. Sun Wu

Accepted for the Council:

Associate Vice Chancellor
and Dean of The Graduate School

PARTICLE SIMULATION FOR 10.6 μm PICOSECOND LASER-INDUCED BREAKDOWN IN GASES

A Dissertation

Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Quan Zhang

May 1992

Copyright © Quan Zhang , 1992

All rights reserved

To My Children

Mong-Mong and Aaron

ACKNOWLEDGEMENTS

The author wish to express her sincere gratitude and appreciation to her major advisor Dr. Dennis Keefer for his guidance, outstanding professional advises, encouragement and abiding support.

Thanks are also expressed to committee members, Drs. F. G. Collins, H. W. Crater, C. C. Limbaugh, T. D. McCay and Y. C. L. Wu for their assistance and helpful comments.

The cordial assistance provided by the UTSI and CLA staff is gratefully acknowledged.

A very special thanks is given to her husband Xiaohang Peng for his constant assistance, encouragement and patience during the course of this study. Also, appreciation is extend to her parents for their encouragement.

The work reported herein was supported in part by the Air Force Office of Scientific Research under Grant AFOSR -86-0317 and by the Center for Laser Applications through funding provided by the Tennessee Higher Education Commission.

ABSTRACT

A theoretical model was developed to explain experimental observations in free electron laser experiments: argon plasmas were initiated and sustained by a free electron laser operating at a wavelength of 10.6 μm with a sequence of 10 ps pulses, and attempts to obtain optical breakdown with the same laser in nitrogen and hydrogen proved unsuccessful. The particle simulation method was used in this model to calculate the electron density increase and the electron energy distribution during the gas breakdown process. A Monte Carlo collision method was incorporated into the particle simulation method to calculate collisions between particles. The theoretical model also included three-body recombination and gasdynamic expansion effects. The calculated results showed that the argon plasma breakdown in the free electron laser experiments was the result of a cumulative growth of the electron density during 27 pulses. Hydrogen and nitrogen did not reach breakdown at the free electron laser experiment conditions because the electron energy decayed more rapidly in these gases during the interval between pulses. The reduction of electron energy permits a much faster electron recombination rate which depletes the electron density during the interval between pulses. The particle simulation predicts that hydrogen and nitrogen can reach breakdown at a power threshold which is higher than that which was used in the free electron laser experiments.

TABLE OF CONTENT

CHAPTER . . . . . PAGE

1. INTRODUCTION 1

1.1 Background of laser-induced breakdown 2

1.2 Background of laser-sustained plasma 6

1.3 The theoretical models and numerical methods used in this study 10

2. THEORY AND MODELS 15

2.1 Introduction 15

2.2 Particle simulation method 17

2.2.1 Simulation procedure 17

2.2.2 Weighting factor 18

2.2.3 Field equations and solver 20

2.2.4 Equations of motion 23

2.3 Monte Carlo method 25

2.3.1 Free flight time and time step cycle 26

2.3.2 Collision process 29

2.4 Recombination and expansion 38

2.4.1 Recombination 38

2.4.2 Expansion 41

2.5 Summary 44

3. RESULTS AND ANALYSIS 45

3.1 Physical and numerical conditions 45

3.2 The evolution of the electron density and distribution 48

3.3 Recombination and expansion 54

3.4 Breakdown threshold 59

4. CONCLUSION AND RECOMMENDATIONS 62

BIBLIOGRAPHY 65

APPENDIXES 72

APPENDIX A. FORTRAN PROGRAM 73

APPENDIX B. INPUT AND OUTPUT DATA 98

VITA 100

List of Figures

Figure . . . . . **Page**

2.1 Sketch of leap-frog integration method showing time-centering of force **F** while advacing **v**, and of **v** while advacing x (from Birdsall and Langdon 1985) 25

2.2 The self-scattering and real collision rates in argon 28

2.3 A typical electron orbit for a high scattering rate, showing several free flight per field-adjusting time step. Scatters are indicated by a sharp change in the orbit. There is a slight change in the orbit every DT seconds due to the recalculation of the electric field (from Hockney and Eastwood 1988) 29

2.4 Coordinates in configuration and velocity space for jump from v' to v 36

3.1 Electron density increase for a sequence of 30 laser pulses in argon at a pressure of 1 atmosphere 49

3.2 Electron density increase for a sequence of 40 laser pulses in hydrogen and nitrogen at a pressure of 1 atmosphere 51

3.3 Electron energy distribution within focal volume at end of laser pulse (10 ps) and at 40 ps. Maxwellian distribution shown for comparison 52

3.4 Electron and ion densities on the grid points in argon after the 27th pulse 53

3.5 Three-body recombination coefficient 54

3.6 Relaxation of the electron energy in argon at one atmosphere during the 46 ns interval between laser pulses 55

3.7 Relaxation of the electron energy in hydrogen at one atmosphere during the 46 ns interval between laser pulses 56

3.8 Relaxation of the electron energy in nitrogen at one atmosphere during the 46 ns interval between laser pulses 57

3.9 Density ratio at 46 ns after pulse 27 in argon 58

3.10 Breakdown thresholds of hydrogen and nitrogen as a function of the pressure 61

CHAPTER 1. INTRODUCTION

Experiments were performed by the University of Tennessee Space Institute (UTSI) at Los Alamos National Laboratory (LANL) in August 1988 to study quasi-steady plasmas (Keefer et al. 1990). The beam from a RF Linac free electron laser (FEL) operating at a wavelength of 10.6 μm contained approximately 1739 pulses 10 ps in width and were separated by 46 ns. Argon plasmas were easily initiated and sustained with this laser at an intensity of $1.0 \times 10^{11} W/cm^2$. Attempts to initiate and sustain similar plasmas in hydrogen and nitrogen using an intensity of $4.0 \times 10^{11} W/cm^2$ were unsuccessful. These experimental results were different from predictions made by Rockwood (1975) more than 10 years earlier. He predicted that for a laser beam with a 10 ps pulse duration and a wavelength of 10.6 μm, threshold intensities of laser induced-breakdown should be less than $10^{11} W/cm^2$ for many gases, including Ar, H_2 and N_2 at a pressure of one atmosphere. The purpose of this dissertation was to develop a theoretical model to explain these unexpected experimental observations.

Laser-induced plasmas have been investigated since the first Q-switched ruby laser was invented in 1962 (McClung and Hellwarth, 1962). The laser-induced breakdown and the laser-sustained plasma are only two branches of this study. Laser-induced breakdown can be defined as the generation of a "totally "ionized gas(plasma). The experimental criterion is usually taken to be the observation of

a glow or intense spark at the focus of a lens. The objective of investigations of laser-induced breakdown is to find the mechanism leading to breakdown and the relation between threshold power and various parameters such as wavelength, focal diameter, pressure, pulse duration and material (Weyl, 1989). The laser-sustained plasma is often referred to as a continuous optical discharge (Keefer, 1989). The primary objective of most work in laser-sustained plasmas is to find the influence on the sustained plasma of various factors such as laser power, flow and optical configuration.

1.1 Background of laser-induced breakdown

There are two main mechanisms for electron generation (Weyl, 1989). The first mechanism is cascade ionization in which electrons absorb the laser radiation through inverse-bremsstrahlung collisions until sufficient energy is gained to ionize an atom by an inelastic electron-atom collision. The second mechanism is the multiphoton ionization (MPI) in which a sufficient number of photons are absorbed simultaneously by an atom or molecule to cause it to ionize.

The first experiments of laser-induced breakdown were carried out by Damon and Thoulinson in 1963. Meyerand and Haught (1963) made the first measurements of thresholds for argon and helium at various pressures. Minck (1964) and Nelsen (1964) also made threshold measurements for N_2, H_2, He, Ar and Air. Tomlinson, Damon, and Buscher (1965) measured the threshold power densities

as a function of pressure in several molecular and inert gases. The early experiments used the ruby laser or neodymium laser as a radiation source. The pulse duration for these lasers is in the range from 15-20 to 50-60 nsec. The pulse duration of a early used carbon dioxide laser is in the range from 200 to 1500 nsec. The first measurements of breakdown thresholds by using a carbon dioxide laser were reported by Generalov, Zimakov, Kozlov, Masyukov, and Raizer (1970). In the 1970's, lasers having picosecond pulse duration became available for experiments and experiments were carried out which showed a strong dependence of breakdown threshold on the pulse length for these short pulse duration lasers.

Many experiments have been carried out to measure the MPI cross sections for gases. The first measurements of multiphoton ionization probability in an optical field were carried out by Voronov and Delone in 1965. The results of these work were listed in the table I of Grey-Morgan's (1975) review, table VI of Delone's (1975) review and table 1.1 of Weyl's (1989) review.

The results of the experiments mentioned above were summarized as follows: The breakdown threshold increases with decreasing gas pressure. The breakdown threshold becomes smaller with increased focal volume. There is a peak value of the breakdown threshold with wavelength. The breakdown threshold is inversely proportional to the pulse length. Although MPI plays a role in the electron ionization process, cascade ionization dominates the breakdown process when the wavelength is larger than $0.53\mu m$. The detailed study of experimental data for

laser-induced breakdown can be obtained in several good reviews (Raizer, 1977; Smith and Meyerand, 1976; Grey-Morgan, 1975; Weyl, 1989).

The theoretical study of laser-induced breakdown is based on classical theories which have been well developed for microwave and electric arc breakdowns. The major difference between the microwave and the optical frequency breakdown is the magnitude of the photon energy from laser sources compared to the ionization potential of gases. Therefore, one needs to consider two additional effects: the multiphoton ionization and the quantum nature of the interaction in the cascade ionization process.

The growth of the electron density, n_e, for laser-induced breakdown is governed by the following equation (Weyl, 1989):

$$\frac{dn_e}{dt} = \nu_i n_e + W_m I^m n - \nu_a n_e - \nu_R n_e + \nabla(D\nabla n_e), \tag{1.1}$$

where n_e is the electron density and n the neutral density, I is the laser irradiance, m is the number of photons for ionizing an atom, W_m is the multiphoton ionization rate coefficient, ν_i, ν_a, ν_R are the impact ionization, attachment, and recombination rates, respectively, and D the diffusion coefficient for electrons. Gasdynamic expansion can also significantly reduce the electron density when the gas heating rate is sufficiently high. This process needs to be considered at later times in the gas breakdown process.

The pioneering calculations of multiphoton ionization were carried out in the mid-60s (Keldysh, 1965; Bebb and Gold, 1965; Gontier and Trahin, 1968; Morton,

1967; Voronov, 1967). Most treatments involved the use of mth-order perturbation theory in order to evaluate the transition matrix element ($\langle i \mid [(e/c)P \cdot A]^m \mid f \rangle$), where $\langle i \mid$ is the initial (bound) state, $\mid f \rangle$ the final (free) state, P is the electron momentum, and A is the vector potential of the EM field associated with the laser beam. Several other general expressions for the multiphoton ionization rate, W_m, were derived by Keldysh (1965), Bebb and Gold (1965), and Grey-Morgan (1975).

The first term of right hand side in Eq.(1.1) is the total cascade ionization rate in the focal volume. The calculation of this term requires the electron energy distribution, which is a function of time. A typical way to find the electron energy distribution is to solve the Boltzmann equation within an optical field. When the photon energy $h\nu$ is much less than the average electron energy $\bar{\epsilon}$ ($\bar{\epsilon} \simeq 3$ to 6 eV), the absorption of the laser radiation in the Boltzmann equation can be treated classically. When $h\nu \geq \bar{\epsilon}$, the assumption of continuous energy absorption is no longer valid and the absorption of photons must be treated quantum-mechanically. Zel' dovich and Raizer (1965) in their pioneering paper derived a quantum kinetic version of the Boltzmann equation that takes into account the absorption of energy in discrete quanta. The inverse-bremsstrahlung absorption coefficient K_a and stimulated emission coefficient K_e were introduced to the energy absorption term in the Boltzmann equation. Several calculations of K_a and K_e were made by Geltman (1973), Dalgarno and Lane (1966), Kroll and Watson (1972), Kas'yanov and Starostin (1965), and Phelps (1966). Measurements of K_a were carried out

by Batenin and Chinov (1972), and Yamabe et al. (1982).

The detailed theoretical study of the gas breakdown threshold caused by laser irradiation have been carried out by a number of authors [Phelps (1966), Nighan (1970), Lowke et al. (1973), Canavan et al. (1972), Kroll and Watson (1972), Weyl (1979), Weyl and Rosen (1985), and Rosen and Weyl (1987)]. A good review paper was given by Weyl (1989).

1.2 Background of laser-sustained plasma

The experimental and the theoretical study of the laser-sustained plasma (LSP) also started in the 1960's. Until recently, most of the experimental studies were directed toward defining combinations of laser power, plasma material, and pressure for which the LSP could be successfully operated. Within the past few years, more detailed studies have been made that better elucidate the interactions among optical geometry, pressure, and flow that control the characteristics of the LSP (Keefer, 1989). Some of these studies were conducted by Keefer et al. at UTSI to determine the suitability of the LSP for laser thermal propulsion. A primary objective of these studies has been to determine the absorption characteristics and stability of the plasma under the influence of forced convective flow. The FEL experiment mentioned in the beginning of this chapter is only a part of these studies at UTSI.

A detailed experimental study of the LSP was carried out recently by Keefer et al.(1986) and Welle et al.(1987) at the University of Tennessee Space Institute. They measured plasma temperatures above 10,000 K by using a spectroscopic method for the plasma which was sustained in a vertical flow of argon, by a 1.5 kW CO_2 laser. They found that the pressure dependence of the plasma was considerably different for two lenses of different focal length and that experimentally determined threshold values were specific to the optical geometry used. They also found that the flow had little influence on the characteristics of the plasma for incident flow velocities within the range of 0.8 to 32 cm/sec. But at the higher flow velocities (up to 4.5 m/sec), the plasma shape, size, and position within the focal volume were significantly influenced. With an increase in the incident flow velocity along the laser beam, the stable plasma position was much nearer the focal point and the region of maximum temperature was more compact and positioned within the region of maximum beam intensity. That results in two effects: the radiation loss decreased from 53 to 45 % of the incident radiation, and the power absorption increased from 67 to 78 % of the incident beam power. These detailed experimental studies of the laser-sustained plasmas indicated that stable laser-sustained plasmas may be operated over a wide range of conditions.

A full two-dimensional theoretical model for laser-sustained plasma was developed recently by Jeng and Keefer (1986). It is based on a Navier-Stokes description of the flow field, geometric ray-tracing for laser beams of arbitrary radial profiles,

and a radiative transfer model that divided the radiative transport into optically thick and optically thin regions. The solutions were found to be in good agreement with detailed measurements of the LSP experiments.

The models have shown that the f/number of the focusing lens has an important effect on the fraction of laser power that is absorbed by the plasma, and predicted that in hydrogen plasmas, it should be possible to sustain a plasma that absorbs virtually 100 % of the incident laser beam power and that radiates no more than 35 % of that power out of the plasma (Jeng et al., 1987).

The models were also used to investigate the effects of using different wavelengths to sustain the plasma. They predicted that the fraction of incident power absorbed at $3.9\mu m$ was 28%, considerably less than the fraction of 75 % for $10.6\mu m$.

The models also indicate that the plasma can be sustained over a wide range of flow velocities and that it is difficult to "blow out", although the flow velocity has a profound influence on the shape of the plasma and its position within the beam. It is found that the fractional power absorption increases with incident flow velocity. The fractional power that is lost from the plasma through radiation decreases as the incident flow velocity is increased.

The development of detailed numerical models has made it possible to study and interpret the complex interactions among optical geometry, flow, and pressure that occur within the LSP. It is clear that the LSP can be produced in a variety of gases and controlled within a wide range of conditions using appropriate com-

binations of laser power, flow, and optical configuration. The position of stability for the plasma will be located where the beam intensity is just sufficient that the absorbed power will balance the losses due to convection, thermal conduction, and thermal radiation. The thermal radiation plays a significant role in determining the detailed structure of the plasma (Keefer, 1989).

Since the LSP can be operated in a small, isolated volume at relatively high pressures and temperatures, a number of potential applications have been proposed, including laser thermal propulsion. The laser thermal propulsion method that was proposed by Keefer et al. (1984) uses a LSP to absorb power from a laser beam and to convert it to enthalpy in the propellant gas. Advantages of laser thermal propulsion are that the heated propellant provides large values of specific impulse (1000 to 1500 sec) with the associated high fractional payloads and small propulsion system masses. For many applications, this makes it superier to other propulsion schemes, *i.e.*, chemical, nuclear and electric.

The application of the LSP to laser thermal propulsion requires very large power (1 to 1000 MW), but research on the free electron laser suggested that these large laser powers were possible. The experiment performed by UTSI at LANL in August, 1988 was a study of quasi-steady plasmas sustained with the beam from a RF Linac free electron laser (FEL). This laser produced a train of pulses having a wavelength of 10.6 micrometers with a pulse width of 10 ps. These pulses were separated by 46 ns, and each pulse train consisted of approximately 1739 pulses.

The total energy in the pulse train ranged from 10 to 300 mJ. Argon plasmas were easily initiated and sustained with this laser. Numerous attempts were also made to initiate and sustain similar plasmas in hydrogen and in nitrogen, but these were unsuccessful. A detailed theoretical model of laser-induced breakdown has been developed by the author in an attempt to explain the experiment results.

1.3 The theoretical models and numerical methods used in this study

Since only $10.6\mu m$ wavelength radiation from the laser beam is considered in this study and the photon energy is much less than the average electron energy at this wavelength, the cascade ionization process leads to gas breakdown and the MPI process can be neglected for atoms and molecules in their ground state. Also, due to the small photon energy, the absorption of the laser radiation by electrons in the cascade ionization process can be treated classically. Since only the initial breakdown was investigated in this study and the population of excited states is much less than that of the ground state at this initial time, impact ionization and photoionization of these excited states can be neglected. Since the calculation of the total cascade ionization rate requires the electron energy distribution, the main problem is to find models which can calculate the electron energy distribution during the breakdown process.

The previous theoretical models for the laser-induced breakdown process were based on the Eulerian approach. In the Eulerian model the plasma is assumed

to behave like a fluid and to obey certain kinetic equations. The appropriate macroscopic differential equations for average quantities are then solved on the computer.

In this study, Lagrangian models were used to calculate the electron energy distribution. The Lagrangian model (also called particle simulation) follows the motion of a large number of plasma particles as they move in time under their mutual electromagnetic interactions. This type of model is more basic, since it only assumes the form of the interaction between individual plasma particles. However, limitations of computer time and storage usually restrict this type of model to a few ten-thousand computational particles instead of the 10^{11} or 10^{12} which nature provides within the breakdown volume.

The development of simulation techniques using particles started about 1960 (Buneman, 1959; Birdsall and Bridges, 1961; Dawson, 1962; Eldridge and Feix, 1962; Burger, 1965; and Hockney, 1965). This method has been successfully used for the areas of plasma physics and controlled fusion. It has also been used in the study of astrophysics and as a design tool for semiconductor devices. Recently, with the rapid development of high speed computers with large memory capacity, this method has become more and more useful. Comprehensive books which describe particle simulation methods are Birdsall and Langdon (1985) and Hockney and Eastwood (1988).

To the author's knowledge, this is the first use of the particle simulation method to calculate the electron energy distribution for the laser-induced breakdown process. The major difficulty with using particle simulation in the breakdown process is that the electron density increases very quickly over a large range (10^3 to $10^{17} cm^{-3}$) during short times. Therefore, the number of computational particles needs to be reduced very often during the whole calculation. Since the electron density is high when the laser-induced breakdown occurs, the plasma frequency can be high for this electron density and then the time step must be short to avoid numerical instability. With this high electron density, the Debye length is small and the cell size needs to be small to decrease noise. The short time step and the small cell size will result in long computational times.

A multi-pulse effect is investigated for the FEL experiments in this study. During the 10 ps duration of the laser pulse, the electron energy increases very quickly through elastic collisions, and its distribution is far from Maxwellian. In an additional 30 ps, although the electron density still increases due to the high energy tail of the electron energy distribution, the electron energy decreases through the large number of elastic and inelastic collisions and its distribution relaxes to approximately Maxwellian. The particle simulation method was used to calculate the electron energy distribution during this 40 ps. Since the cascade ionization process leads to gas breakdown in this study, collision processes dominate the calculation during the 40 ps. They are calculated by a Monte Carlo collision

method (MCC) which is incorporated into the particle simulation method.

After the first 40 ps, the electron energy distribution is more nearly a Maxwellian distribution with an average energy which decreases until the end of 46 ns. During the long relaxation time of 46 ns between pulses, it is not necessary to calculate the electron energy distribution since rate equations for average quantities can be used during this time. The average electron density and the average electron energy are calculated by these rate equations during this time. Due to the large number of collisions, the electron energy will decrease considerably, making it possible for the two processes – electron-ion recombination and gasdynamic expansion – to become important.

The electron-ion recombination is a process that reduces the electron density. This process is determined by the recombination coefficient which is a function of the electron density and temperature. When the electron density is high enough but the electron temperature is relatively low, the three-body (electron) recombination effect can be very significant during the 46 ns. A rate equation for the electron density is used to calculate this effect. A first-order finite differencing method was used to solve the rate equation.

The gasdynamic expansion may result in a shock wave propagating away from the focal volume depending on the gas heating rate. When the electron density is high enough, the gas heating rate by electrons through elastic collisions with neutral particles will be high. Then the resulting gasdynamic expansion will

considerably reduce the electron density. The unsteady quasi-one-dimensional Euler equations were used to describe this process. These equations were solved by the Euler implicit method with a second-order flux-differencing upwind scheme and total variation diminishing (TVD) technique (Hsieh, 1989).

The calculation steps described above are repeated for a sequence of 10 ps laser pulses until either the electron number density saturates at a maximum value below breakdown or until breakdown is achieved. The detailed theoretical models are described in Chapter 2. The calculated results and discussion are given in Chapter 3. Conclusions are stated in Chapter 4.

CHAPTER 2. THEORY AND MODELS

2.1 Introduction

When a laser beam is focused into a gas, a plasma can be created with production of charged particles and absorption of the laser light. If a threshold irradiance has been achieved, a gas breakdown will occur. Simultaneously, a bright spark can be observed at the focus of the laser beam. The shape of the spark will rapidly extend along the laser beam from a spherical to a long cylindrical form once breakdown has been achieved at the focal point. Since only the initial breakdown was investigated in this study, it is assumed that the electrons are confined to a spherical volume centered on the focal point of the laser beam and that the plasma is spherically symmetric. This permits a one-dimensional description of the problem in r direction.

There are two main mechanisms which lead to gas breakdown – cascade ionization and multiphoton ionization (MPI). Since only $10.6\mu m$ wavelength radiation is considered in this study and the photon energy is much less than the average electron energy at this wavelength, the cascade ionization process leads to gas breakdown and the MPI process can be neglected.

Several collision processes are dominant for the electron generation in the cascade ionization breakdown. Electrons immersed in a laser field will gain energy through electron-neutral inverse bremsstrahlung collisions. Electrons will

lose energy by elastic and inelastic collisions (rotation, vibration, excitation, and ionization) with neutral particles. New electrons will be generated by ionizing collisions. Some electrons will be lost by electron-ion recombination and ambipolar diffusion processes. These processes are all considered in this study.

A multi-pulse effect is investigated for the FEL experiments investigated. The calculation for the electron generation is carried out through the 10 ps pulse duration and the 46 ns between pulses for each pulse period. During the 10 ps duration of the laser pulse, the electron energy increases very quickly through elastic collisions, and the electron distribution is far from Maxwellian distribution. During the next 30 ps after the laser pulse is off, electrons lose energy by elastic and inelastic collisions, and the electron distribution relaxes to approximately a Maxwellian distribution. During this 40 ps period, the particle simulation method was used to calculate the electron energy distribution. This method is discussed in Sec. 2.2.

At the same time, A Monte Carlo collision (MCC) method was incorporated into the particle simulation method to calculate the collisions between particles. The detailed discussion of the MCC is given in Sec. 2.3.

After the first 40 ps, the electron energy distribution is more nearly a Maxwellian distribution with an average energy which decreases until 46 ns. Thus, the rate equations for the macroscopic quantities – electron density, electron energy, and the gas properties, are used during this time. This calculation is described in Sec.

2.4.

2.2 Particle simulation method

The particle simulation method utilizes a large number of computational particles to simulate the plasma by following their motions. Each computational particle may represent a number of real particles and has a set of attributes, such as mass, charge, position, and velocity. The state of the physical system is defined by the attributes of a finite ensemble of particles and the evolution of the system is determined by the interactions between the particles and Newton's law of motion.

In this study, a one-dimensional model was used for the particle simulation by assuming that the plasma was spherically symmetric, *i.e.*, only the spatial coordinate r is considered. A 3-D velocity space model (v_r, v_θ, v_ϕ) was assumed in order to consider the random collisions of particles.

2.2.1 Simulation procedure

The plasma particle simulation method used in this study has four principle steps in each computational cycle:

1. Assigning charges to the computational grid points.
2. Solving the field equations at each grid point.
3. Interpolating mesh-defined forces at particle positions.
4. Advancing particles through the field.

The computational cycle starts from initial values of positions and velocities of particles. The particle quantities, such as velocity and position, are known for each particle and may take on continuous values in $\bar{v}$ and r space, called phase space. Each simulation particle is assigned an array index i, such as v_i, and r_i. The field quantities are obtained only at the spatial grid points, known only at discrete points in space labeled with index j, such as $\bar{E}_j$. The connections between the particle position and velocities and the field quantities are made by first calculating the charge densities on the spatial grid points. This step requires a distribution of the particle charge among the neighboring grid points. This process of charge assignment implies some weighting to the grid points that is dependent on particle position. Once the densities are established on the grid points, then a method is used to obtain the electric fields. With the fields known at the grid points, but with the particles scattered around within the grid, the fields are interpolated from the grid points to the particles in order to determine the force at the particle. This is again performed by using a weighting. Having the force on the particle, Newton's second law is used to move the particles and obtain new positions and velocities of the particles.

2.2.2 Weighting factor

As described above, it is necessary to calculate the charge density at the discrete grid points from the particle positions and, after the fields are obtained,

to calculate the force at the particles from the fields at the grid points. These calculations are called weighting, which implies some form of interpolation among the grid points nearest the particle. It is desirable to use the same weighting in both density and force calculations in order to avoid a self-force (*i.e.*, a particle accelerates itself) (Birdsall and Langdon 1985).

There are several different particle simulation methods which are classified according to their different weighting schemes, such as Nearest Grid Point method (NGP) (Burger et al. 1965), Cloud in Cell method (CIC) (Birdsall and Fuss 1969), and Particle in Cell method (PIC) (Morse and Nielson 1969). The PIC method was used in this study. Usually, the PIC method uses a linear interpolation to assign a point charge to its nearest grid points in one dimension. It produces a smoother density and field distribution than the Nearest Grid Point method. Higher-order weighting in the PIC method can be obtained by the use of quadratic and cubic interpolation. This reduces density and field noise, but requires more computation time.

There are two requirements for a weighting scheme. The first is charge conservation. That is, for each particle, the sum of the fractional charges assigned to the grid must be equal to the total charge carried by the particle. The second one is charge density conservation. That is, a uniform distribution of particles should render a uniform charge density on the grid upon particle-to-grid weighting. In spherical coordinates, these two requirements result in an interpolation which is

at least third order (Ruyten 1992).

For the PIC method in one-dimensional spherical coordinates, the weighting factor of a particle with the position r between grid points r_{j-1} and r_{j+1} is defined by,

$$S_j = \begin{cases} S_j^-(r) & r_{j-1} < r \leq r_j, \\ S_j^+(r) & r_j \leq r < r_{j+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

Ruyten's (1992) expression for a third-order weighting factor is assumed:

$$S_j^+(r) = \frac{r_{j+1} - r}{r_{j+1} - r_j}[1 + ar(r - r_j)], \tag{2.2}$$

where the parameter a is determined by the charge density conservation:

$$a = -\frac{5}{2}\frac{1}{2r_{j+1}^2 + r_{j+1}r_j + 2r_j^2}. \tag{2.3}$$

Once the weighting factor $S_j^+(r)$ is known, the $S_{j+1}^-(r)$ can be obtained by charge conservation:

$$S_{j+1}^-(r) = 1 - S_j^+(r). \tag{2.4}$$

For the special case: $r = r_j$,

$$S_j^-(r_j) = S_j^+(r_j) = 1. \tag{2.5}$$

2.2.3 Field equations and solver

In one-dimensional spherical coordinates, Gauss' law can be used directly at each grid point. Let the charges of the particles i be assigned to the grid points j

using a weighting which guarantees charge conservation exactly

$$\sum_{i=1}^{N} q_i = q_{\text{total}} = \sum_{j=0}^{N_r-1} Q_j, \tag{2.6}$$

where N and N_r are the number of particles and grid points, respectively. Using Gauss' law guarantees flux conservation. Applied at the spherical surface $j = 1/2$, Gauss' law produces the radial electric field from the charges,

$$\frac{Q_o}{\epsilon_o} = 4\pi r_{1/2}^2 E_{1/2} \tag{2.7}$$

where Q_o is the total net charge on the center grid point, $r_{1/2}$ is the radius of the spherical surface $j = 1/2$, and $E_{1/2}$ is the electric field at the radius $r_{1/2}$.

When $r_{j+1/2}$ is some radius in the interval, such as $(r_{j+1}+r_j)/2$, and between the j and $j+1$ surface, Gauss' law produces,

$$\frac{Q_j}{\epsilon_o} = 4\pi r_{j+1/2}^2 E_{j+1/2} - 4\pi r_{j-1/2}^2 E_{j-1/2} \tag{2.8}$$

where $j = 1, ...N$. In order to obtain an equation in ϕ, one can use single cell differencing

$$E_{j+1/2} = -\frac{\phi_{j+1} - \phi_j}{\Delta r_j} \tag{2.9}$$

where $\Delta r_j \equiv r_{j+1} - r_j$, so that Gauss' law reads

$$\frac{Q_j}{\epsilon_o} = 4\pi r_{j+1/2}^2 \frac{\phi_{j+1} - \phi_j}{\Delta r_j} + 4\pi r_{j-1/2}^2 \frac{\phi_j - \phi_{j-1}}{\Delta r_{j-1}} \tag{2.10}$$

Then equations (2.7) and (2.10) become

$$\begin{cases} \dfrac{Q_0}{\epsilon_o}\dfrac{1}{4\pi} = -\dfrac{r_{1/2}^2}{\Delta r_o}(\phi_1 - \phi_0), & j = 0 \\ \dfrac{Q_j}{\epsilon_o}\dfrac{1}{4\pi} = -\dfrac{r_{j+1/2}^2}{\Delta r_j}(\phi_{j+1} - \phi_j) + \dfrac{r_{j-1/2}^2}{\Delta r_{j-1}}(\phi_j - \phi_{j-1}), & 1 \le j \le N \end{cases} \tag{2.11}$$

Only one boundary condition is needed,

$$\phi_{N+1} = 0. \tag{2.12}$$

The linear system of equations, (2.11), can be written in matrix form,

$$\begin{pmatrix} b_o & a_o & & & & \\ c_1 & b_1 & a_1 & & & \\ & c_2 & b_2 & a_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & c_{n-1} & b_{n-1} & a_{n-1} \\ & & & & c_n & b_n \end{pmatrix} \cdot \begin{pmatrix} \phi_o \\ \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{n-1} \\ \phi_n \end{pmatrix} = \begin{pmatrix} s_o \\ s_1 \\ s_2 \\ \vdots \\ s_{n-1} \\ s_n \end{pmatrix}, \tag{2.13}$$

where:

$$a_j = -r^2_{j+1/2}/\Delta r \qquad j = 0, ..., n,$$

$$b_j = (r^2_{j+1/2} + r^2_{j-1/2})/\Delta r \qquad j = 1, ..., n, \qquad b_o = r^2_{1/2}/\Delta r,$$

$$c_j = -r^2_{j-1/2}/\Delta r \qquad j = 1, ...n,$$

and

$$s_j = \frac{Q_j}{4\pi\epsilon_o} \qquad j = 0, ...n.$$

The matrix in Eq.(2.13) is tridiagonal. This linear system can be solved efficiently by the Crout Reduction algorithm for tridiagonal linear systems (Burden and Faires, 1985).

Once potentials at the grid points are known, the electric fields can be obtained by the difference method,

$$E_j = -\frac{\phi_{j+1} - \phi_{j-1}}{2\Delta r} \tag{2.14}$$

The force acting on a particle is produced only by the electric field. The electric field at the grid point is weighted to an individual particle in the same manner as in the charge assignment, *i.e.*,

$$E_i(r) = E_j S_j^+(r) + E_{j+1}(1 - S_j^+(r)) \tag{2.15}$$

2.2.4 Equations of motion

In the particle simulation method, the particles move according to Newton's laws of motion,

$$\frac{d\bar{v}}{dt} = \frac{\bar{F}}{m} \tag{2.16}$$

$$\frac{d\bar{r}}{dt} = \bar{v} \tag{2.17}$$

The force $\bar{F}$ on a particle with charge q moving in electric field $\bar{E}$ is $q\bar{E}$. Since the electric field is assumed to be in the r direction only, the detailed expression of Eq.(2.16) in three perpendicular directions of spherical coordinates is given by

$$\left(\frac{d\bar{v}}{dt}\right)_r = \frac{qE}{m} = \frac{dv_r}{dt} - \frac{v_\theta^2 + v_\phi^2}{r} \tag{2.18}$$

$$\left(\frac{d\bar{v}}{dt}\right)_\theta = 0 \tag{2.19}$$

$$\left(\frac{d\bar{v}}{dt}\right)_\phi = 0 \tag{2.20}$$

Introduce the perpendicular velocity $v_\perp = \sqrt{v_\theta^2 + v_\phi^2}$ and calculate $v_\perp$ by a conservation of angular momentum during the move and Eq.(2.18), (2.19) and

(2.20) are replaced by

$$\frac{dv_r}{dt} = \frac{qE}{m} + \frac{v_\perp^2}{r} \tag{2.21}$$

$$L = r \times v_\perp = rv_\perp = constant \tag{2.22}$$

For one-dimensional spherical coordinates, Eq.(2.17) can be written as

$$\frac{dr}{dt} = v_r \tag{2.23}$$

Replacing Eq.(2.21), (2.22) and (2.23) by finite-difference equations according to the leap-frog scheme (Birdsall and Langdon, 1985),

$$v_{r_{\text{new}}} = v_{r_{\text{old}}} + (\frac{qE}{m} + \frac{v_{\perp_{\text{old}}}^2}{r_{old}})\delta t \tag{2.24}$$

$$r_{new} = r_{\text{old}} + v_{r_{\text{new}}}\delta t \tag{2.25}$$

$$v_{\perp_{\text{old}}} = \frac{L}{r_{new}} \tag{2.26}$$

Then, the velocity components in the direction θ and ϕ are chosen at random,

$$v_\theta = v_\perp \times R_f \tag{2.27}$$

$$v_\phi = \sqrt{v_\perp^2 - v_\theta^2} \tag{2.28}$$

where R_f is a uniform random number between 0 and 1. The computer will advance $\bar{v}_{t-\Delta t/2}$ and $\bar{r}_t$ to $\bar{v}_{t+\Delta t/2}$, $\bar{r}_{t+\Delta t}$. The flow in time and notation is shown in Figure 2.1, which makes clear the time centering.

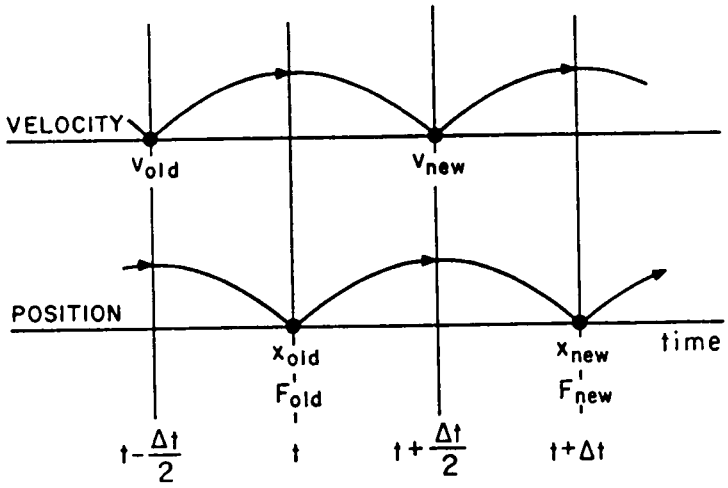


Figure 2.1. Sketch of leap-frog integration method showing time-centering of force **F** while advancing **v**, and of **v** while advancing **x** (from Birdsall and Langdon 1985).

2.3 Monte-Carlo method

To incorporate collisions into the PIC code, a Monte-Carlo collision method is used to directly simulate the binary collisions between particles. Since the collision frequencies are time dependent (they are energy-dependent, and, consequently, time-dependent), there is no simple expression for calculating the free flight time between collisions. Therefore, a self-scattering concept is used to resolve this difficulty. This idea was first introduced by Rees (1968). In the PIC code, all the different collision processes are modeled by a selection rule which is determined by their collision frequencies.

2.3.1 Free flight time and time step cycle

The probability that a particle free flight time is between t and $t+dt$ is $p(t)dt$ (Holt and Haskell 1965):

$$p(t)dt = \nu e^{-\nu t} dt, \tag{2.29}$$

where $p(t)$ is the probability density function and ν is the total collision rate of a particle. The random number generated on the computer is a uniform distribution, while the probability density function $p(t)$ is a nonuniform function. The problem is to find an expression which relates the random number to free flight time.

Define the probability that a particle free flight time is less than t,

$$P(t) = \int_0^t p(t')dt' \tag{2.30}$$

and hence

$$dP(t) = p(t)dt = \nu e^{-\nu t} dt. \tag{2.31}$$

If a constant collision rate ν is assumed during time step, then Eq.(2.30) becomes

$$P(t) = 1 - e^{-\nu t} \tag{2.32}$$

where

$$0 \leq P < 1 \qquad \text{and} \qquad P(\infty) = 1.$$

If the value of P is chosen as a random number with a uniform distribution ($0 \leq r < 1$), then the probability of selecting a value in the range dP is dP, which is also the probability that a particle free flight time is between t and $t + dt$.

Introduce a random number r and change t to δt, then Eq.(2.32) becomes

$$r = P(t) = 1 - e^{-\nu \delta t}. \tag{2.33}$$

Rewrite Eq.(2.33),

$$\delta t = -\frac{1}{\nu} ln(r_1) \qquad r_1 = 1 - r, \tag{2.34}$$

where r_1 is a random number. This relates δt, the free-flight time, to a random number, r_1.

In general, the total collision rate for a real process is not constant. It is a function of energy. Rees(1968,1969) suggested the introduction of a dummy "self-scattering" process that makes no change to the velocity of the particle. The self-scattering rate is indicated by the shaded area of Fig.2.2. If we now include the energy dependent self-scattering as a process, the total collision rate becomes a constant which is equal to the sum of the real collision and self-scattering rates. With the addition of self-scattering, free flight times can be chosen according to Eq.(2.34).

After the free flight, the selection of the collision process is made with a second uniformly distributed random number r_2 in the range $0 \leq r_2 < \nu$, where ν is a constant collision rate, composed of the sum of the real processes plus self scattering.

The collision process m is selected if

$$\sum_{i=1}^{m-1} \nu_i(e_k) \leq r_2 < \sum_{i=1}^{m} \nu_i(e_k) \tag{2.35}$$

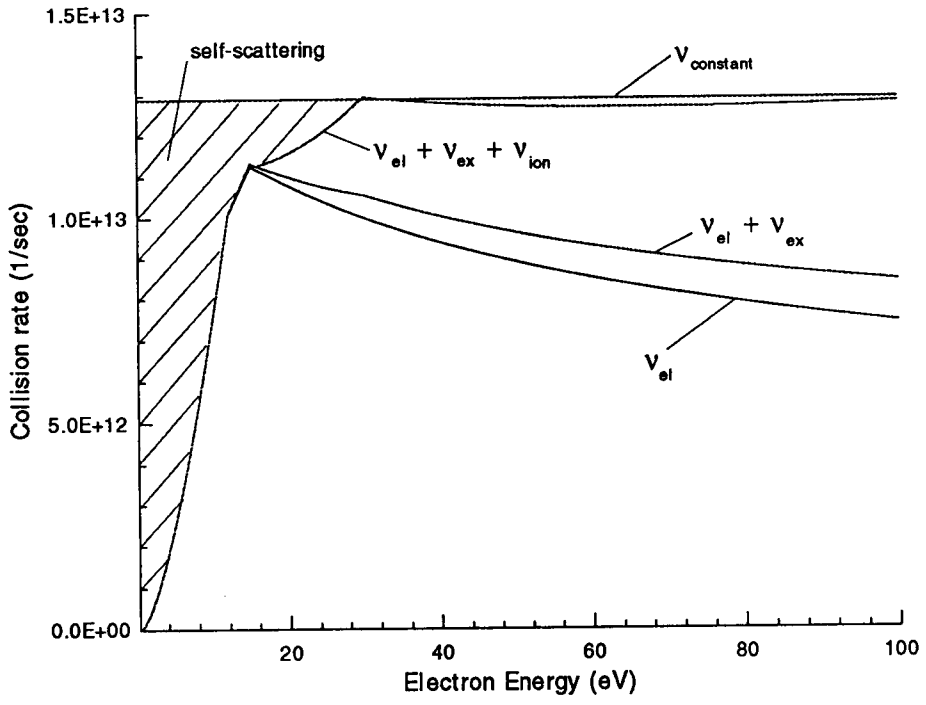


Figure 2.2. The self-scattering and real collision rates in argon.

where $\nu_i(e_k)$ is the collision rate for process i (Fig.2.2).

Now, the normal time step cycle in the particle simulation needs to be modified in order to incorporate Monte-Carlo collisions (Eq.(2.35)). Figure 2.3 shows several collision processes occurring for each field solution (Hockney and Eastwood, 1988). After a field adjustment, an inner loop of Monte-Carlo collisions is entered, during which the electric field remains unchanged. In this inner loop each particle is taken in turn and subjected to Monte-Carlo collisions until the sum of its free-flight times exceeds the time at which the next field-adjustment is

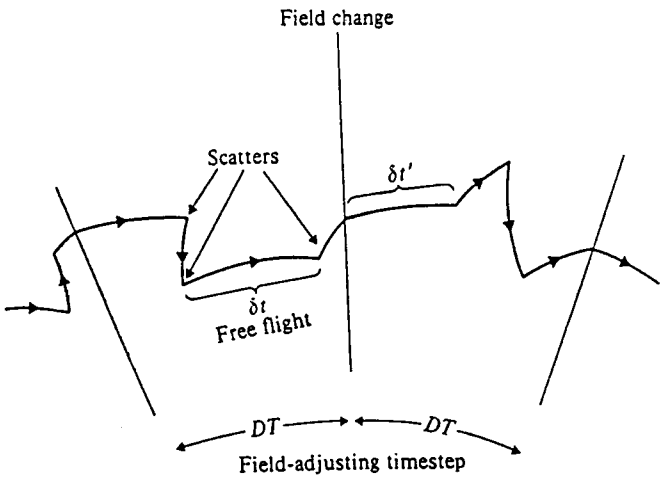


Figure 2.3. A typical electron orbit for a high scattering rate, showing several free flight per field-adjusting time step. Scatters are indicated by a sharp change in the orbit. There is a slight change in the orbit every DT seconds due to the recalculation of the electric field (from Hockney and Eastwood 1988).

to take place. The amount of the excess is stored as the coordinate $\delta t'$, so that the free flight may be correctly continued with the new field after the next field adjustment. During the inner loop each of the simulation particles is scattered in three-dimensional velocity space as though it were a single particle, with the effective mass and charge appropriate to its state.

2.3.2 Collision process

There are many real collision processes which can occur in the plasma. The importance of each process depends on the particular gas (Table 2.1). For example, in argon, only three scattering processes are considered for electrons with neutrals, i.e., elastic, excitation, and ionization processes. For molecular hydrogen and ni-

Table 2.1 Collision Processes in Argon and Hydrogen

Argon	
elastic	$A + e \xrightarrow{+E_q} A + e$
excitation	$A + e \xrightarrow{-11.2eV} A^* + e$
ionization	$A + e \xrightarrow{-15.57eV} A^+ + 2e$
coulomb	$A^+ + e \longrightarrow A^+ + e$
i-A scattering	$A + A^+ \longrightarrow A + A^+$
charge exchange	$A + A^+ \longrightarrow A^+ + A$
Hydrogen	
elastic	$H_2 + e \xrightarrow{+E_q} H_2 + e$
vibration & rotation	$H_2 + e \longrightarrow H_2^* + e$
dissociation	$H_2 + e \xrightarrow{-9.0eV} 2H_2 + e$
ionization	$H_2 + e \xrightarrow{-15.54eV} H_2^+ + 2e$
coulomb	$H_2^+ + e \longrightarrow H_2^+ + e$
i-H_2 scattering	$H_2 + H_2^+ \longrightarrow H_2 + H_2^+$
charge exchange	$H_2 + H_2^+ \longrightarrow H_2^+ + H_2$

trogen, there are five electron collision processes with neutrals; elastic, rotation, vibration, dissociation, and ionization. In high density plasmas the coulomb collisions between electrons and ions are important for both gases. Ion scattering and charge exchange processes between ions and neutral particles are also considered.

Having selected a scattering process, the post-collision velocity needs to be calculated for the particles. One must look at each process separately and then determine the energy partitioning and the scattering angle.

Electron-neutral collisions

The electron-neutral collision rate for process i can be expressed as

$$\nu_i = nvQ_i,$$

where Q_i is the cross section for process i, n is the neutral number density, and v is the electron velocity. In this study, the cross sections for elastic, excitation and ionization collisions in Ar were taken from Mitchner and Kruger (1973). The cross sections for elastic, rotation, vibration, dissociation, and ionization collisions in H_2 were taken from Massey (1969) and Kieffer (1973). The cross sections for elastic, electronic excitation, vibration, dissociation, and ionization collisions in N_2 were taken from Huxley and Crompton (1974) and Kieffer (1973).

For the electron-neutral elastic collision, the electron loses a small fraction of its energy by collision due to its small mass relative to the neutral, and may lose a small fraction of energy by radiation in the bremsstrahlung process. However, the electron absorbs energy from the optical field during the pulse duration by

inverse bremsstrahlung. Since the calculation of the lost energy by the radiation in the bremsstrahlung between electron and atom is too complex, (in the initial breakdown process, the gas density is greatly higher than the ion density) and this lost energy is much smaller than the energy absorbed by electron in the inverse bremsstrahlung, this lost energy is neglected during the calculation. The amount of absorbed energy is derived using purely classical ideas as follows.

The equation of motion for an average electron in the optical electric field ($E_{op} = E_o \cos \omega t$) is given as

$$m_e \frac{d\bar{v}}{dt} = -q\bar{E}_{op} - m_e \nu_m \bar{v} \tag{2.36}$$

This equation includes the effect of collisions on the motion of an average electron. ν_m is the momentum-transfer collision rate of the electron with neutrals. Solving Eq.(2.36) for the velocity $\bar{v}$ in one-dimension, we obtain

$$v(t) = -\frac{qE_o}{m_e} \frac{1}{\omega^2 + \nu_m^2} (\omega \sin \omega t + \nu_m \cos \omega t) \tag{2.37}$$

The ν_m can be neglected from Eq.(2.37) by comparing the optical frequency ω with ν_m for the laser wavelength of $10.6 \mu m$. Then the velocity becomes

$$v(t) \approx -\frac{qE_o}{m_e \omega} \sin \omega t \tag{2.38}$$

or

$$v(t) = -v_m \sin \omega t \tag{2.39}$$

where $v_m = qE_o/m_e\omega$.

Thus, the oscillatory kinetic energy of an electron in an optical field can be obtained from

$$E_k(t) = \frac{1}{2} m_e v^2(t) = \frac{1}{2} m_e v_m^2 \sin^2 \omega t \tag{2.40}$$

For an optical frequency cycle, the average kinetic energy, referred as the "quiver energy", is written as,

$$E_q = \frac{1}{4} m_e v_m^2 = \frac{1}{4} m_e \left(\frac{qE_o}{m_e \omega} \right)^2. \tag{2.41}$$

An elastic collision between an atom and an electron considerably alters the direction of motion of the latter (this motion consists of the translational component and of an oscillatory component directed at some angle with respect to the direction of translation) (Raizer, 1977). The new motion of the electron begins practically from the same kinetic energy as just before the collision because the large difference between the atomic and electron masses results only in a very small transfer of energy to an atom experiencing an elastic collision. However, the new translational motion now has a completely different direction relative to the field and, in the average field, the oscillations of an electron are built up again at a new angle relative to the translational velocity. In this way, the electron acquires, on the average, an additional energy of the order of the quiver energy. This discussion of the acquisition of energy is based on an important assumption $\omega \gg \nu_m$. If the $E_q \gg \hbar\omega$, the process is classical. This is true in this study. In this way, the electron rapidly increases its energy with each elastic electron-neutral collision.

For an electron-neutral excitation collision, the energy of the scattered electron is simply:

$$E_{\text{scattered},e} = E_{\text{incident},e} - E_{\text{excitation}} \tag{2.42}$$

(e.g., for argon, $E_{\text{excitation}} = 11.55$ eV). The electron-neutral vibration, rotation, and dissociation processes are all similar to this scattering process.

For an electron-neutral ionization collision, the electron energies after the collision are equal to that before the collision minus the ionization energy, or

$$E_{\text{scattered},e} + E_{\text{created},e} = E_{\text{incident},e} - E_{\text{ionization}}. \tag{2.43}$$

The energy of the created ion is given the energy associated with the neutral. The energy is divided between the two electrons by inverting the differential cross section (Birdsall,1991).

$$E_{\text{scattered},e} = B \tan\left[r_3 \tan^{-1}\left(\frac{E_{\text{incident},e} - E_{\text{ionization}}}{2B}\right)\right] \tag{2.44}$$

where B is known (e.g., for argon, $B \cong 10$ eV over our range of interest) and r_3 is a random number. The energy of the electron created is now simply:

$$E_{\text{created},e} = E_{\text{incident},e} - E_{\text{ionization}} - E_{\text{scattered},e}. \tag{2.45}$$

After colliding with a neutral particle, the electron velocity is changed not only in magnitude but also in direction. The new direction of the electron velocity is determined by an angle χ, between the electron velocity $\bar{v}$ (before collision) and

$\bar{v}'$ (after collision), and an angle ϕ defined by Fig.2.4. The angle χ is calculated from (Birdsall, 1991)

$$\cos\chi = \frac{2 + E - 2(1+E)^{r_4}}{E} \tag{2.46}$$

and the angle ϕ is chosen by $2\pi r_5$, where r_4 and r_5 are random numbers. E is the electron energy.

For calculation, it is assumed that the electron changes its energy first, then changes its direction second during collision. Thus, only the unit vectors of the electron velocities before and after collision are required. In calculation, the velocity $\hat{v}'$ is projected first on the three mutually perpendicular, known directions $(\hat{v},\ \hat{v}\times\hat{\theta},\ (\hat{v}\times\hat{\theta})\times\hat{v})$. Then these components are transformed to the regular axes $\hat{r}$, $\hat{\theta}$, and $\hat{\phi}$. The expression for the new velocity components can be written as

$$\begin{aligned}\hat{v}'_r &= \hat{v}'\cdot[\hat{v} + \hat{v}\times\hat{\theta} + (\hat{v}\times\hat{\theta})\times\hat{v}]\cdot\hat{r} \\ &= [\cos\chi\hat{v} + \sin\chi\cos\phi(\hat{v}\times\hat{\theta}) + \sin\chi\sin\phi(\hat{v}\times\hat{\theta})\times\hat{v}]\cdot\hat{r}\end{aligned} \tag{2.47}$$

similarly,

$$\hat{v}'_\theta = [\cos\chi\hat{v} + \sin\chi\cos\phi(\hat{v}\times\hat{\theta}) + \sin\chi\sin\phi(\hat{v}\times\hat{\theta})\times\hat{v}]\cdot\hat{\theta} \tag{2.48}$$

$$\hat{v}'_\phi = [\cos\chi\hat{v} + \sin\chi\cos\phi(\hat{v}\times\hat{\theta}) + \sin\chi\sin\phi(\hat{v}\times\hat{\theta})\times\hat{v}]\cdot\hat{\phi} \tag{2.49}$$

Finally, we have:

$$v'_r = v_r\cos\chi - v_\phi\sin\chi\cos\phi - v_r v_\theta\sin\chi\sin\phi \tag{2.50a}$$

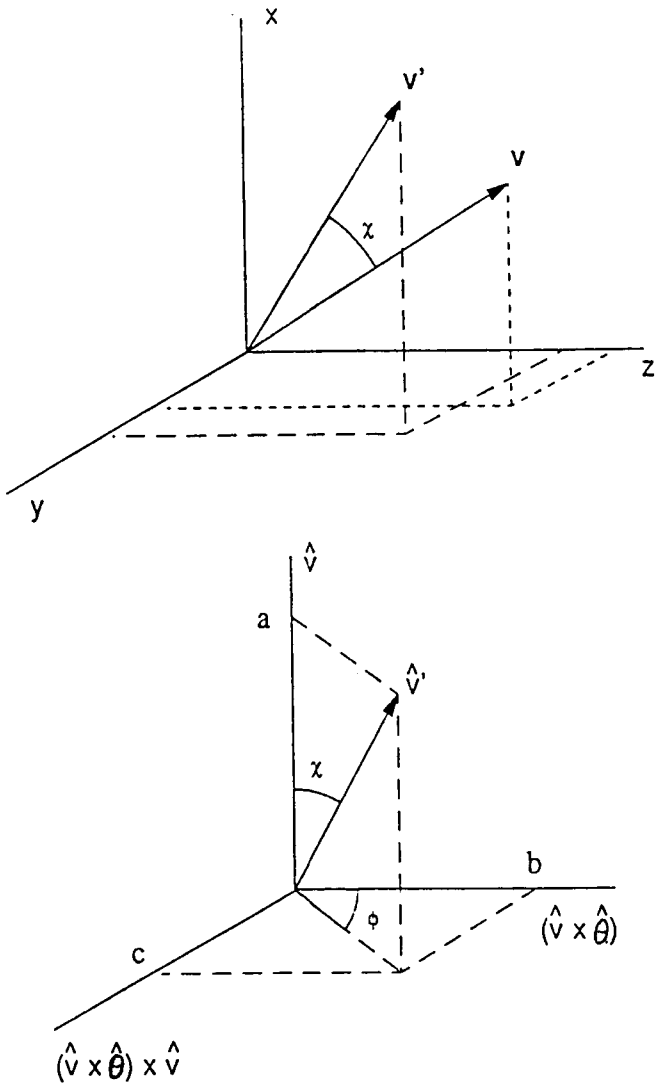


Figure 2.4. Coordinates in configuration and velocity space for jump from v' to v.

$$v'_\theta = v_\theta \cos\chi + (v_r^2 + v_\phi^2) \sin\chi \sin\phi \tag{2.50b}$$

$$v'_\phi = v_\phi \cos\chi + v_r \sin\chi \cos\chi - v_\theta v_\phi \sin\chi \sin\phi \tag{2.50c}$$

Ion-neutral collisions

The ion-neutral collision cross sections are specified simply by:

$$\sigma = a + b/\sqrt{E} \tag{2.51}$$

with a and b given separately for elastic and charge exchange (Birdsall, 1991). (Note that $\sigma v \sim b$ for $E \to 0$, *i.e.*, there is no singularity in σv as $E \to 0$.) The colliding ions and neutral atoms have essentially the same mass (*e.g.*, Ar^+ and Ar). In this study, the a and b were calculated by the experiment data taken from McDaniel (1964) for Ar, H_2 and N_2.

For elastic scattering (hard sphere, equal masses) we use:

$$E_{\text{scattered},i} = E_{\text{incident},i} \cos^2 \chi \tag{2.52}$$

where $2\chi = \Theta$, the center of mass scattering angle. If we assume isotropic scattering, χ is obtained from:

$$\cos\Theta = \cos 2\chi = 1 - 2r_6 \qquad (\text{uniform} - 1 \quad \text{to} \quad 1), \tag{2.53}$$

$$\cos\chi = \sqrt{1 - r_6} \tag{2.54}$$

where r_6 is a random number between 0 and 1. The scatter in ϕ is uniform 0 to 2π ($\phi = r_7 2\pi$), where r_7 is a random number between 0 to 1.

For charge exchange, the incident fast ion picks up an electron from a slow gas atom (becoming a fast atom) and leaving behind a slow scattered ion. The

particles exchange forward momentum. The slow ion is scattered isotropically; χ is uniform 0 to $\pi(r_8\pi)$ and ϕ is uniform 0 to $2\pi(r_9 2\pi)$, where r_8 and r_9 are random numbers between 0 and 1. The speed of the scattered ion is chosen to match the room temperature.

In summary, during each real scattering process, the incident particle undergoes changes in its energy and velocity direction.

2.4 Recombination and expansion

The heavy particles in the focal volume are heated during the 46 ns time between laser micropulses as elastic and inelastic collisions relax the electron energy. There are two important effects that result from this relaxation during the time between pulses. If the average electron temperature relaxes rapidly enough it will allow significant reduction of electron density by 3-body recombination, and the energy gained from the electrons by the heavy species can result in a gasdynamic expansion of the focal volume. A set of coupled rate equations were used to calculate the rate of relaxation and the reduction of electron density. The gasdynamic expansion of this heated gas was calculated using a one-dimensional CFD code.

2.4.1 Recombination

The recombination effect becomes significant when the average electron temperature is less than several thousand degrees (about 0.5 eV).

The macroscopic recombination coefficient α is defined by the relation,

$$\frac{dn_e}{dt} = -\alpha n_e n_i \tag{2.55}$$

where n_i denotes the number density of ions with which the electrons are recombining.

Recombination may occur through any of several possible mechanisms; radiative, three-body (electron or heavy particle), dissociative, and dielectronic recombination. In this study, the electron energy can decrease to very low values for N_2 and H_2 through elastic and inelastic collisions during the long relaxation time. When the electron density is high enough during this time, the three-body (electron) recombination effect is significant. Only the three-body (electron) recombination process is considered in this calculation.

The average electron density in the computational volume is described by

$$\frac{dn_e}{dt} = -\alpha n_e^2 + \beta n_e, \tag{2.56}$$

where β is the ionization rate ($cm^3 sec^{-1}$) and n is the neutral density. The Thomson model was used to calculate the recombination coefficient (Clark and Incropera, 1969),

$$\alpha = C n_e T_e^{-9/2} \qquad (cm^3 sec^{-1}), \tag{2.57}$$

where T_e is electron temperature and C is a constant obtained from experiment. The value of C chosen for this work was measured by Hinnov and Hirschberg

(1962),

$$C = 5.6 \times 10^{-9} (K^{9/2} cm^6 sec^{-1}). \tag{2.58}$$

The ionization rate in Eq.(2.56) is obtained from Th. Bornath et al. (1988) as a smooth interpolating function from following data,

$$\beta(T) = \begin{cases} 3. \times 10^{-17}, & T = 8000K; \\ 10^{-12}, & T = 16000K; \\ 10^{-10}, & T = 32000K; \\ 2. \times 10^{-9}, & T = 64000K. \end{cases} \tag{2.59}$$

As shown in Eq.(2.57) and Eq.(2.59), the recombination coefficient and ionization rate are functions of the electron energy. During the 46 ns between pulses, the electron energy decreases through elastic and inelastic collisions and a simple energy balance is used to determine the average electron temperature,

$$\frac{dE}{dt} = -\nu_{el} \Delta E_{el} - \nu_{vib} \Delta E_{vib} - \beta n_a \Delta E_{ion}, \tag{2.60}$$

where:

ν_{el} – elastic collision frequency

ν_{vib} – vibration collision frequency.

βn_a – ionization collision frequency.

$\Delta E_{el} = \frac{2m_e}{M} E$.

ΔE_{vib} – vibration energy.

ΔE_{ion} – ionization energy.

Initially, when the electron density is small there is too little energy absorbed for any significant gasdynamic expansion and Equations (2.56) and (2.60) are

solved together. A Euler's method for initial value problems was used to solve these equations. The initial values of particle number density and electron energy are obtained from the PIC method at 40 ps, since the average electron energy at that time is high and the recombination rate is small. Having solved the equations at the end of the 46 ns, the new particle density and the new average electron energy can be obtained. The average ratio between the particle densities at the 40 ps and 46 ns is calculated within the focal region, and the ratio between the average electron energy at 40 ps and 46 ns is also calculated. These ratios are then used in the PIC method to modify the corresponding quantities. These modified quantities are used as the initial values to begin the PIC calculation for the next pulse.

2.4.2 Expansion

After several pulses, when the electron density in the focal volume has reached approximately $10^{17}\ cm^{-3}$, the gas heating between pulses is sufficient to produce a gasdynamic expansion which reduces the electron density prior to the arrival of the next pulse. In one-dimensional spherical coordinates, the gas expansion is governed by Euler equations which are expressed as,

$$\frac{\partial Q}{\partial t} + \frac{\partial E}{\partial r} = F, \tag{2.61}$$

where

$$Q = \begin{pmatrix} \rho \\ \rho u \\ \rho \epsilon \end{pmatrix} a \qquad E = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ (\rho \epsilon + p) u \end{pmatrix} a \qquad F = \begin{pmatrix} 0 \\ p \frac{da}{dx} \\ \dot{s} a \end{pmatrix} \tag{2.62}$$

and

ρ – the mass density of gas.

u – the velocity of gas.

ϵ – total internal energy.

p – pressure.

$a = r^2$ – proportional to the cross-section area.

$\dot{s}$ – rate of change of energy source.

A CFD model was used to calculate the gasdynamic expansion during the 46 ns between pulses. The numerical scheme in the model was described by Hsieh (1989), where Eq.(2.61) was solved by the Euler implicit method with a second-order flux-differencing upwind scheme using a total variation diminishing (TVD) technique. The Jacobian matrices in the upwind scheme were evaluated using "Roe-averaged" quantities.

The source term in Eq.(2.62) is determined by the gas heating rate obtained from electron-neutral elastic collisions. It is assumed that the gas has no hydrodynamic expansion during the first 40 ps of a pulse because the energy which goes into gas heating is not sufficient to cause expansion. This energy is added to the energy term in Eq.(2.62) at the beginning of the gas expansion. The gas heating rate from electron-neutral elastic collisions at the grid point j is determined by

the total loss of electron energy at the grid point j during the time step dt,

$$\frac{ds(j,t)}{dt} = \frac{n_e(t)\Delta E(t)f(j)}{dt} \tag{2.63}$$

where

$\Delta E(t)$ – the loss energy of a electron during time step dt.

$f(j)$ – the energy distribution function at the grid j.

dt – time step.

$n_e(t)$ – the number density of electron at time t.

and $\Delta E(t)$ is determined by Eq.(2.60).

The electron density in Eq.(2.63) should be reduced simultaneously by two effects – recombination and gasdynamic expansion. Another equation needs to be used to calculate the new electron density $n'_e(t)$ due to the gasdynamic expansion during dt,

$$n'_e(t) = n_e(t)\frac{\sum_{j=1}^{N_f}\rho_j(t)}{\sum_{j=1}^{N_f}\rho_j(t-dt)} \tag{2.64}$$

where $\rho_j(t)$ is the gas density on the grid point j at the time t and N_f is the total grid number in the focal volume.

In the calculation for each time step, the $n_e(t)$ is first reduced by Eq.(2.56). Then it is brought into Eq.(2.63) to calculate the gasdynamic expansion. Finally, it is reduced again by Eq.(2.64). These steps are repeated at each time step until the end of the 46 ns between pulses.

2.5 Summary

A particle simulation method is used to calculate the electric field and to simulate the motion of the particles. A Monte Carlo collision method is used to calculate the effects due to collisions between particles. The rate equations for recombination and gasdynamic expansion are used to calculate the reduction of the electron density during the interval between laser pulses. These computational steps are repeated for a sequence of only 10 ps laser pulses since the electron number density either saturates at a maximum value below breakdown or breakdown is achieved.

CHAPTER 3. RESULTS AND ANALYSIS

The codes, called PCSR (Plasma Computational Simulation in one-dimension r with PIC and MCC method) and ROE (for the gasdynamic expansion), were used to simulate the charged particle behavior and the gasdynamic expansion phenomena for the gas breakdown process observed in the FEL experiment at the Los Alamos National Laboratory (Keefer et al., 1990). These codes were developed in an attempt to understand why plasma breakdown in argon could be easily initiated using the Los Alamos FEL, but plasma breakdown could not be achieved in either hydrogen or nitrogen. The physical and numerical conditions are stated in Sec.3.1. The evolution of electron distribution and density are discussed in Sec.3.2. The decreases of electron energy and electron density due to the recombination and gasdynamic expansion are described in Sec.3.3. Breakdown thresholds are evaluated in Sec.3.4.

3.1 Physical and numerical conditions

The physical parameters used in PIC code are based on the FEL experiments at the Los Alamos National Laboratory. A RF linac free electron laser with a 10.6 micron wavelength was focused into the gases by a $127mm$ focal length lens. The pulse format of the RF linac laser consisted of an 80 microsecond long burst. Each 80 μs burst consisted of approximately 1739 micropulses that had a duration of

10 picoseconds and were spaced 46 nanoseconds apart to provide a peak power of about 17 MW. The radius of the laser beam in the focal region was $37\mu m$.

The experiments were simulated for three separate gases (Ar, H_2, N_2) at a pressure in the range 1 to 5 atm. The radius of the computational volume was two times the radius of the laser beam to allow consideration of the particle diffusion effect, and a uniform optical field distribution was assumed in the focal region. It was also assumed that the neutral particles were uniformly distributed through the entire computational domain. Only electron and ion trajectories were followed explicitly.

A necessary condition for cascade ionization is that there be an initial electron in the focal volume. Usually, an initial electron can be obtained several ways: multiphoton ionization (for short wavelength, $\lambda \leq 1.06\mu m$), partial preionization of the gas, tunnel effect through which the electric field may be strong enough to pull an electron from an outer shell of its orbital (for irradiance $> 4 \times 10^{12} W/cm^2$), and cosmic rays ($n_e = 10^3 cm^{-3}$). In this study, we simply assume that an initial electron exists within the focal volume.

The calculation begins with a single electron placed near the center of the focal region. This electron undergoes elastic collisions until its energy exceeds that required to suffer an inelastic collision with a neutral molecule. This inelastic collision can result in an excited molecule, dissociation of the molecule into atoms, or an ionization event in which a second electron is produced. When a second

electron is produced, it too gains energy from the optical electric field.

Each electron or ion is tracked in both space and time until the total number of electrons reaches 60,000. Then the number of electrons retained in the computation is reduced by a factor of approximately 3, and each computational electron then represents 3 real electrons. This process is repeated over a sequence of 10 ps laser pulses until either no further increase in electron density occurs or the breakdown electron density is reached. Many investigators use the condition $(\ln(n_{ef}/n_{ei}) \simeq 40)$ as criterion for breakdown, where n_{ef} and n_{ei} are the final and initial electron densities, respectively. This criterion would correspond to an electron density of $10^{17} cm^{-3}$ by the end of the pulse if one started out with $n_{ei} = 1\ cm^{-3}$ (Weyl, 1989). In this study, breakdown is assumed to occur when the electron density reaches one percent ionization, or approximately $2.45 \times 10^{17} cm^{-3}$ at one atmosphere.

Four stability requirements for the particle simulation method must be satisfied during the calculation (Birdsall and Langdon, 1985, and Hockney and Eastwood, 1988):

1. time step Δt.

$$\nu_p \Delta t \sim 1$$

where ν_p is the plasma frequency which is defined by

$$\frac{\nu_p}{2\pi} = 9000\sqrt{n_e} \qquad (n_e - cm^{-3})$$

2. Cell size Δr.

$$\frac{\Delta r}{\lambda} \leq \pi$$

where λ is the Debye length defined by

$$\lambda = 6.9(\frac{T}{n_e})^{1/2} \qquad (T - K, n_e - cm^{-3})$$

3. The number of cells crossed by a particle in every time step must be less then 2.

4. The minimum number of computational particles per cell must be larger than 10.

The purpose of the first three stability requirements is to build an electric field as real as possible and to prevent electrons from moving too far. The last requirement is to smooth the charge density and potential. For a more effective calculation, the time step and cell size were changed with the charge density according to the stability requirements during the whole calculation. In order to save computational time and storage, the cell size was uniform over the entire computational domain. Hence, the volume of each cell was unequal in spherical coordinates.

3.2 The evolution of the electron density and distribution

The calculated electron density for argon is shown in Fig.3.1 for a sequence of 30 pulses at a laser irradiance of $2.0 \times 10^{11} W/cm^2$. The time scale between

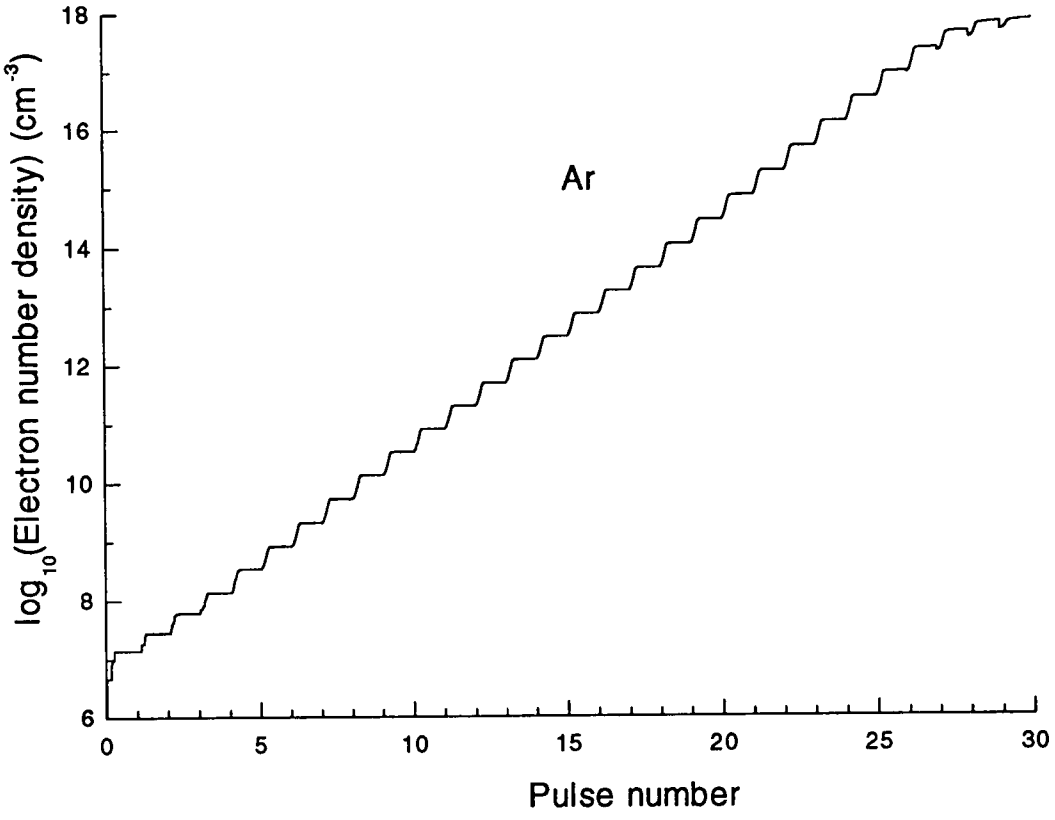


Figure 3.1. Electron density increase for a sequence of 30 laser pulses in argon at a pressure of 1 atmosphere.

successive laser pulses has been contracted to 40 ps for clarity. Fig.3.1 describes the evolution of the electron density from a single electron placed in the focal volume ($2 \times 10^{-7} cm^3$) until electron ionizing collisions have produced an avalanche of electrons that reaches plasma breakdown. Breakdown is assumed to occur when the electron density reaches approximately $2.45 \times 10^{17} cm^{-3}$ at one atmosphere.

The result in Fig.3.1 indicates that the breakdown in the FEL experiment did not occur during a single 10 ps pulse, but was the result of a cumulative growth of the electron density resulting from the absorption of approximately 27 pulses in argon. The average electron density increased by a factor of approximately 2.5 per pulse for argon at this power.

The effect of gasdynamic expansion appears only after the electron density reaches $10^{17} cm^{-3}$. The electron density is reduced by this effect after 26 pulses. The recombination process does not significantly change the electron density for argon because the average electron energy during the 46 ns interval between pulses is higher than 1.45 eV, keeping the recombination coefficient small. This is due to the large atomic mass in elastic collisions and the lack of vibrational or dissociative collisions.

The calculated electron densities for hydrogen and nitrogen are shown in Fig.3.2 for a sequence of 40 pulses at the laser irradiance $4.0 \times 10^{11} W/cm^2$ (the maximum irradiance obtained in the FEL experiments). It can be seen that the electron densities in both gases reach a plateau of approximately $2.15 \times 10^{14} cm^{-3}$ for H_2 and $9.08 \times 10^{15} cm^{-3}$ for N_2 after several pulses, and never achieve breakdown. These results are consist out with the FEL experiment results. This saturation effect is caused by the rapid recombination of electrons which occurs during the 46 ns between the laser pulses. The detailed explanation of the experiment results is discussed in Sec. 3.3.

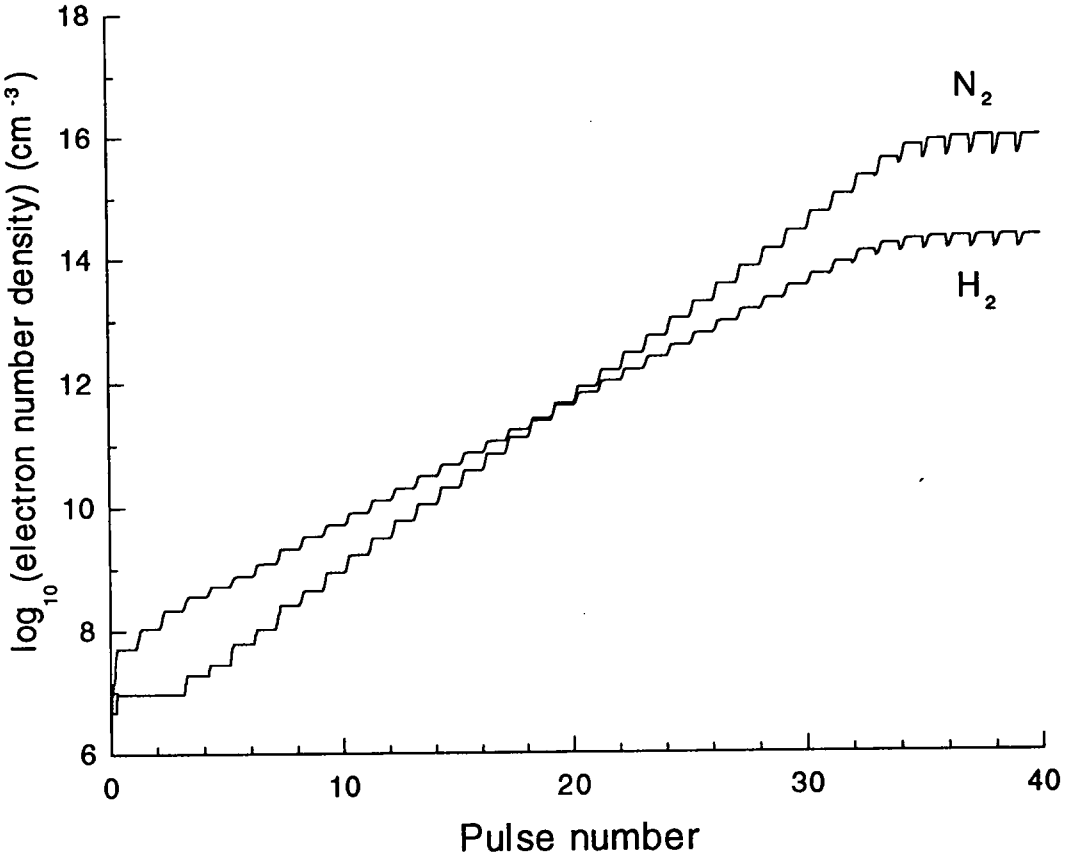


Figure 3.2. Electron density increase for a sequence of 40 laser pulses in hydrogen and nitrogen at a pressure of 1 atmosphere.

The calculated electron energy distribution function for argon during the 27th pulse in the sequence is shown in Fig.3.3. This figure plots the total number of electrons in the $2 \times 10^{-7} cm^3$ focal volume rather than the electron density. Note the relatively large number of electrons in the high energy (10 - 40 eV) tail of the distribution at 10 ps, the end of the laser pulse. This high energy tail will lead to a continuing increase of electron density after termination of the

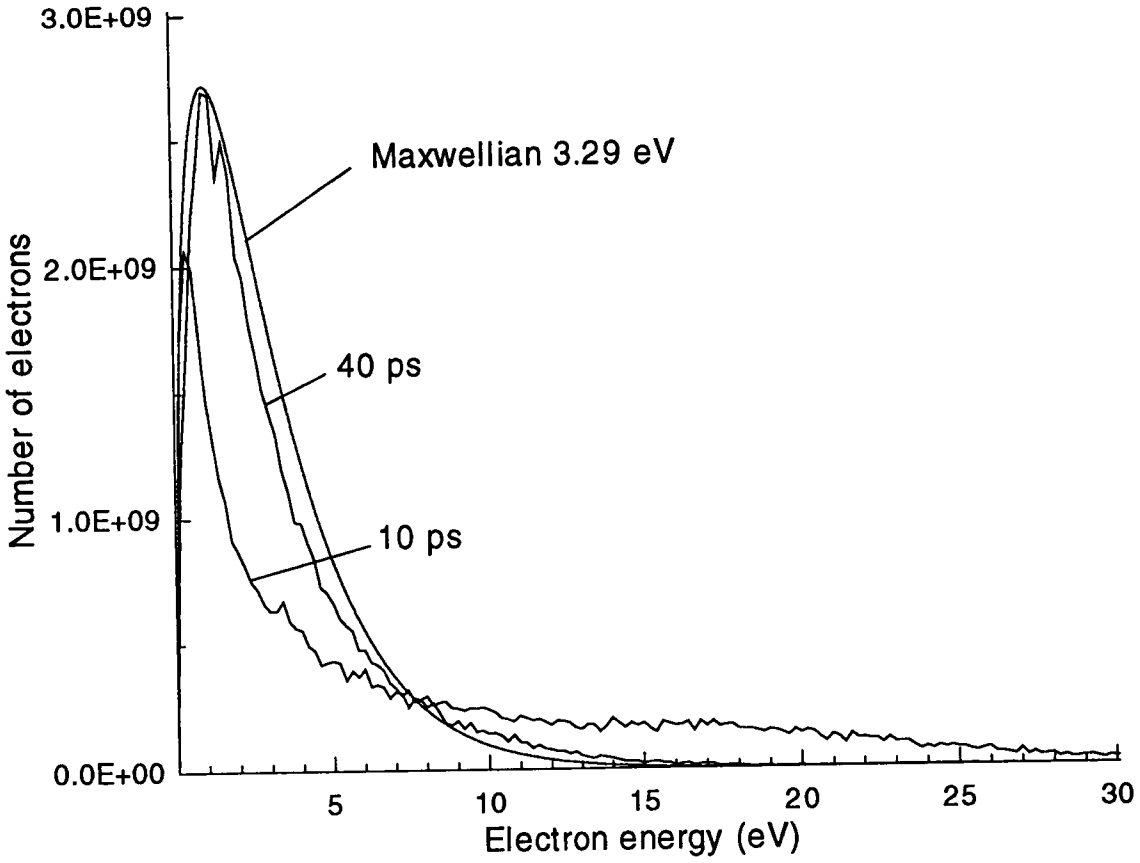


Figure 3.3. Electron energy distribution within focal volume at end of laser pulse (10 ps) and at 40 ps. Maxwellian distribution shown for comparison.

pulse. Note the relaxation of the distribution function to a nearly Maxwellian form after 40 ps. This is why the particle simulation method is used only during the first 40 ps and rate equations are applied for the rest of the 46 ns between pulses. The total number of electrons in the focal volume increases as the high energy electrons undergo ionizing collisions during the relaxation. A Maxwellian distribution having the same mean energy of 3.29 eV is also shown for comparison.

The electron and ion densities at the grid points are shown in Fig.3.4 for argon after the 27th pulse. Fig.3.4 is based on 900 data points which come from 9000 original data points on the 9000 grid points. Each of the 900 data points is the average value of 30 original data points. It can be seen that the charged particle densities decrease slowly with the increase of the spatial r. This result is caused by electron and ion ambipolar diffusion for a uniform optical field distribution in the focal region. Since the electron and ion densities have small fluctuations along the grid points, the net charges oscillate on the grid points and the electric field also oscillates on the grid points. This oscillation effect retards the motion of charged particles over a long distance.

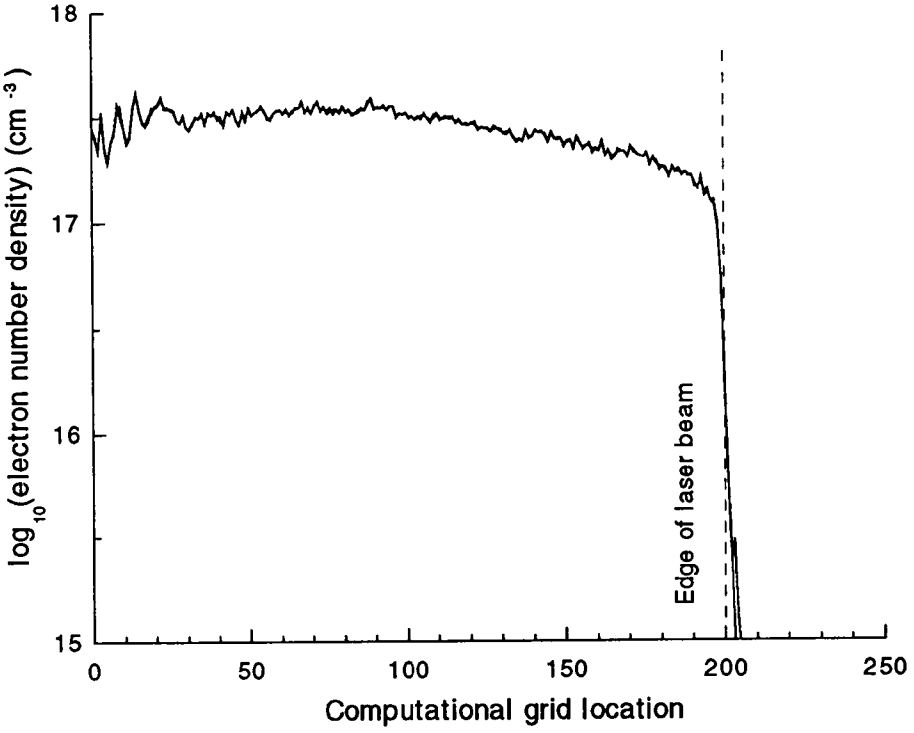


Figure 3.4. Electron and ion densities on the grid points in argon after the 27th pulse.

3.3 Recombination and expansion

The 3-body recombination coefficient is a strong function of electron temperature, and is largest for low energy electrons as shown in Fig.3.5. The recombination effect becomes significant when the temperature of the electrons is less than about five thousand degrees.

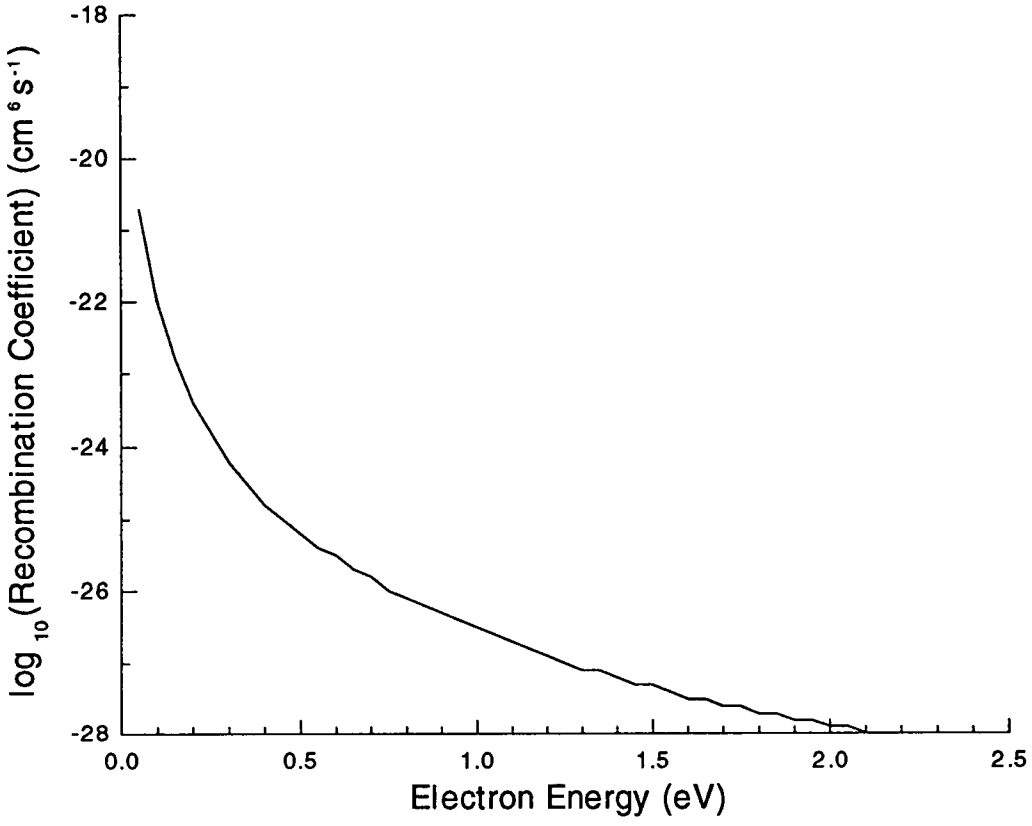


Figure 3.5. Three-body recombination coefficient.

During the 46 ns between laser pulses the average electron energy decreases due to elastic and inelastic collisions between the electrons and neutrals. The rate of electron energy loss due to inelastic collisions is relatively small for argon because there are no vibrational or dissociation collisions, and the loss due to elastic collisions is small because of the large atomic mass. The relaxation of the average energy in argon is shown in Fig.3.6. Since the electron energy remains relatively high during the 46 ns between pulses, recombination does not significantly reduce the electron density.

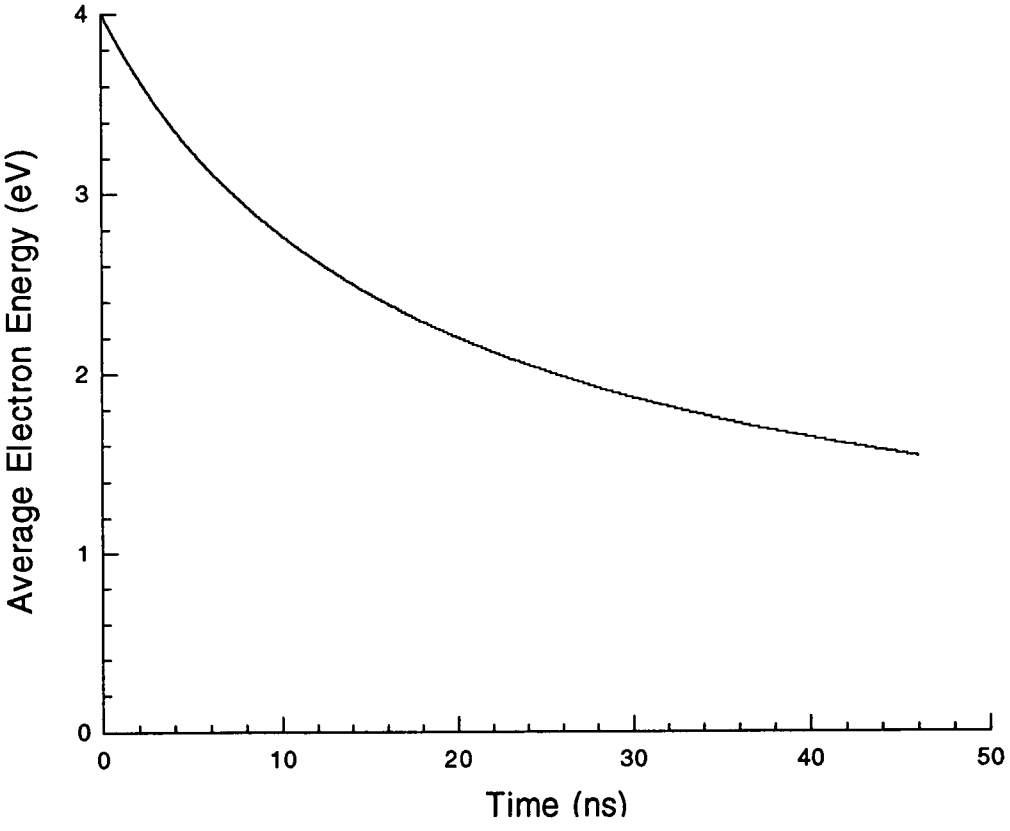


Figure 3.6. Relaxation of the electron energy in argon at one atmosphere during the 46 ns interval between laser pulses.

On the other hand, vibration and dissociation collisions and a small molecular mass combine to rapidly reduce the electron energy between pulses for hydrogen and nitrogen. The relaxation of the average electron energy in the two gases is shown in Fig.3.7 and Fig.3.8 (note the difference in time scales). The reduced electron energy permits electron recombination to significantly reduce the electron density in the focal volume during the 46 ns between laser pulses.

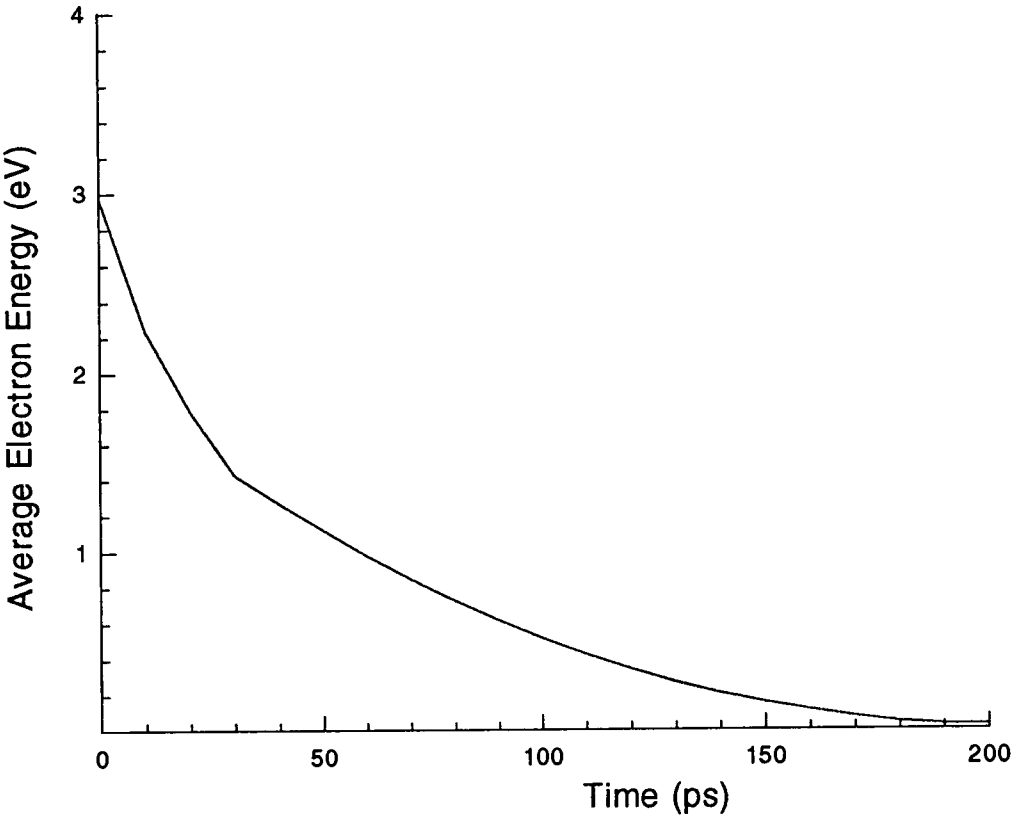


Figure 3.7. Relaxation of the electron energy in hydrogen at one atmosphere during the 46 ns interval between laser pulses.

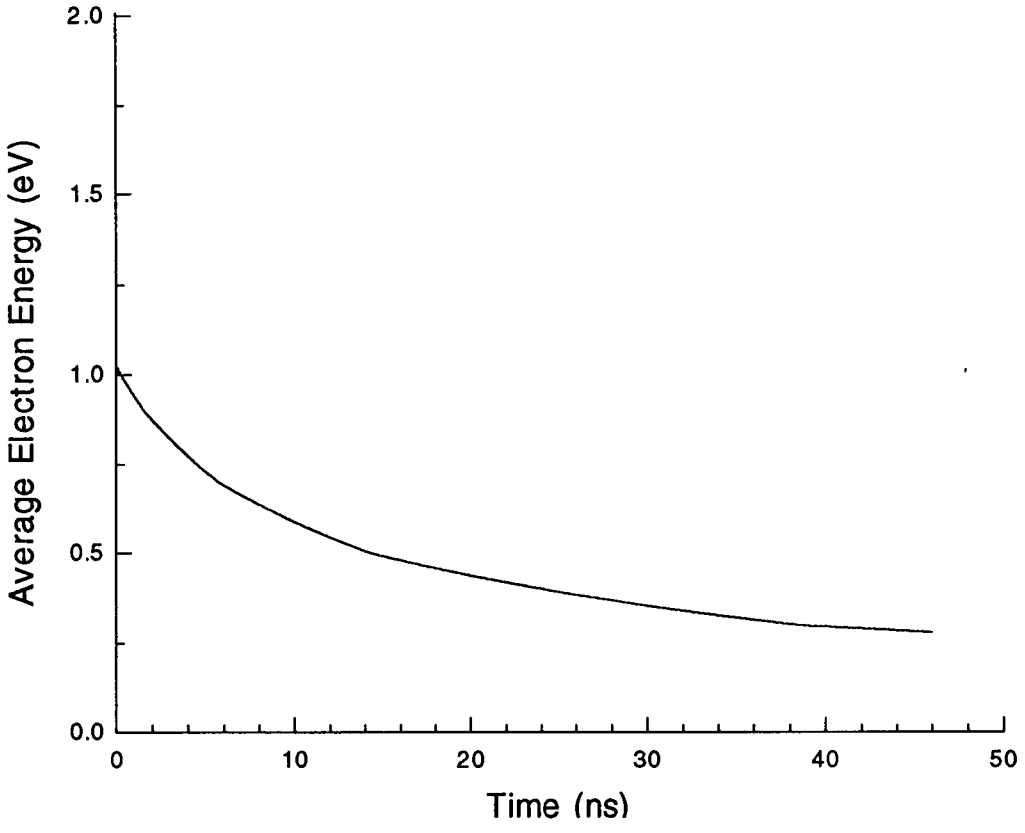


Figure 3.8. Relaxation of the electron energy in nitrogen at one atmosphere during the 46 ns interval between laser pulses.

For nitrogen, the average electron energy rapidly decreases from 4 eV to 1 eV (the threshold of inelastic collisions) by inelastic collisions during a few picoseconds, but the electron density is not significantly reduced during this short time. After these few picoseconds, the average electron energy decreases only though elastic collisions. During the long relaxation time (46 ns), the average electron energy decreases gradually to 0.28 eV, and the recombination effect is significant.

The result of this recombination is that, for hydrogen and nitrogen, the electron density saturates at a maximum value which is below plasma breakdown.

When the electron density reaches approximately $10^{17} cm^{-3}$, enough energy is added to the gas through elastic collisions that gasdynamic expansion reduces the density in the focal volume. The density ratio after pulse 27 due to the gasdynamic expansion is shown in Fig.3.9. For this pulse, the density of gas has been reduced to approximately 86% by gasdynamic expansion during the 46 ns time between pulses. During the gasdynamic expansion, the electron density is also reduced with gas density.

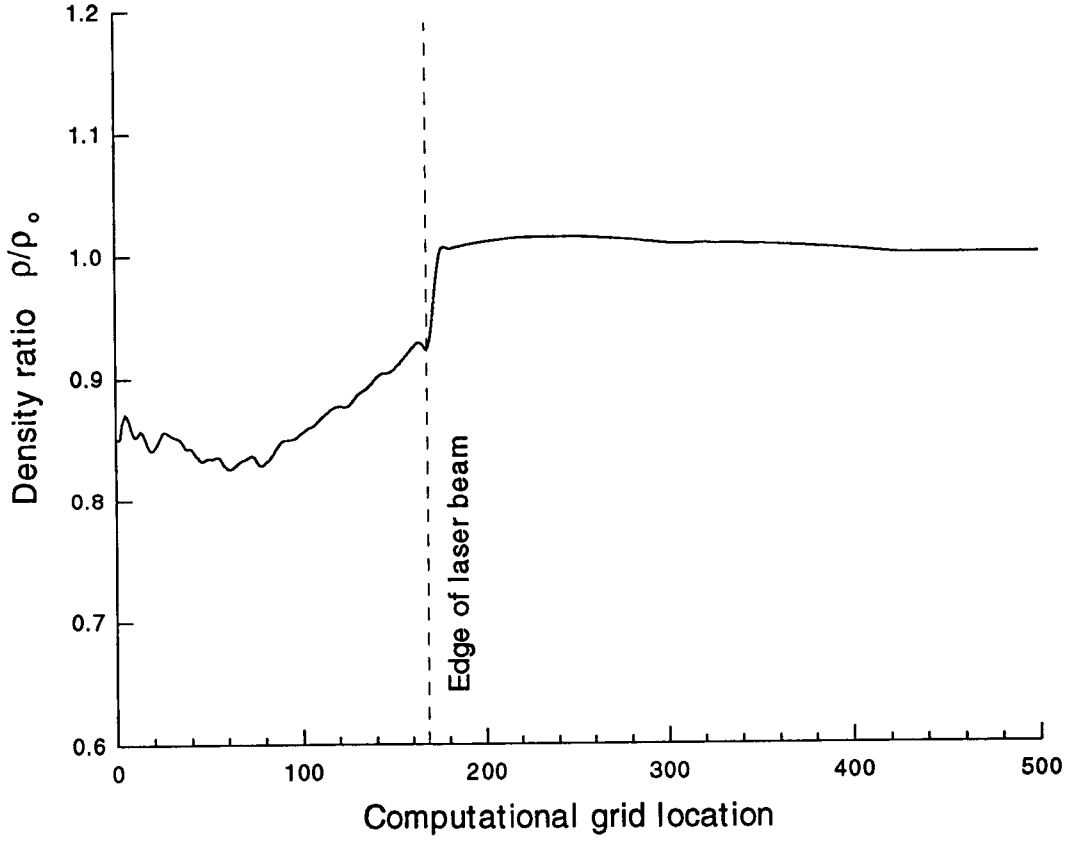


Figure 3.9. Density ratio at 46 ns after pulse 27 in argon.

3.4 Breakdown threshold

Threshold calculations were performed for argon, hydrogen and nitrogen at the FEL experimental conditions. The calculated results for argon were compared with the experimental results in order to verify the PCSR code developed by author.

The breakdown threshold of a gas depends on the combined effect of several parameters: pressure, wavelength, pulse duration, focal volume and materials. In this calculation, the breakdown threshold was defined as the minimum energy required to cause electron cascade growth at a certain rate. Since there is no significant recombination effect for argon in the initial breakdown, the breakdown threshold is determined by finding a minimum energy at which the number of electrons continues to increase for several pulses. However, due to the significant recombination effect for hydrogen and nitrogen, the breakdown threshold is a minimum energy at which the rate of increase of electron density is greater than its loss rate due to recombination during the 46 ns between pulses.

Breakdown threshold of argon

In the free electron laser experiments, only one energy threshold for argon was measured. This was approximately 50 mJ for a pulse train at a pressure of approximately 3 atm. This threshold corresponds to an irradiance of $6.69 \times 10^{10} W/cm^2$ for a focal spot size 74 microns in diameter. At the same condition, the

calculated threshold is 45 mJ for a pulse train. The relative error (10%) between the experimental and calculated results indicate the code PCSR is reliable.

Breakdown threshold of hydrogen and nitrogen

Free electron laser experiments were carried out with a maximum available energy (300 mJ for a pulse train) for H_2 and N_2 at different pressures (1 atm to 5 atm) and no breakdown was observed for these gases. Therefore, the threshold calculation for these two gases were made using a larger laser energy but holding the other experimental conditions constant. During the calculation, breakdown was assumed to occur when the electron density reaches one percent ionization at a given pressure and continues to reach one percent ionization for subsequent pulses.

For example, due to the large number of elastic and inelastic collisions during the 46 ns between pulses, the average electron energy, if it is higher than 1 eV at the end of 40 ps, will decrease to approximately 0.049 eV for H_2 and 0.28 eV for N_2 for one atmosphere at the end of 46 ns. Hence, the electron density, if it is higher than $10^{17} cm^{-3}$ at the end of 40 ps, will be reduced to $1.7 \times 10^{14} cm^{-3}$ for H_2 and $6.3 \times 10^{15} cm^{-3}$ for N_2 at the end of 46 ns due to the recombination effect. Since breakdown is assumed to occur when the electron density reaches approximately $2.45 \times 10^{17} cm^{-3}$ for one atmosphere, the breakdown threshold of H_2 is the energy at which the electron density growth rate is about 1400 per pulse, and the breakdown threshold of N_2 is the energy at which the electron density

growth rate is about 40 per pulse. Using these criteria, the calculated threshold for H_2 is 2.5 J (or $3.4 \times 10^{12} W/cm^2$) and the calculated threshold for N_2 is 730 mJ (or $1 \times 10^{12} W/cm^2$) for a pulse train at a pressure of 1 atm. The calculated thresholds for H_2 and N_2 are shown in Fig.3.10. These calculated thresholds are higher than those which were used in the FEL experiments. Since no breakdown was observed for hydrogen and nitrogen, the particle simulations are consistent with the experiments.

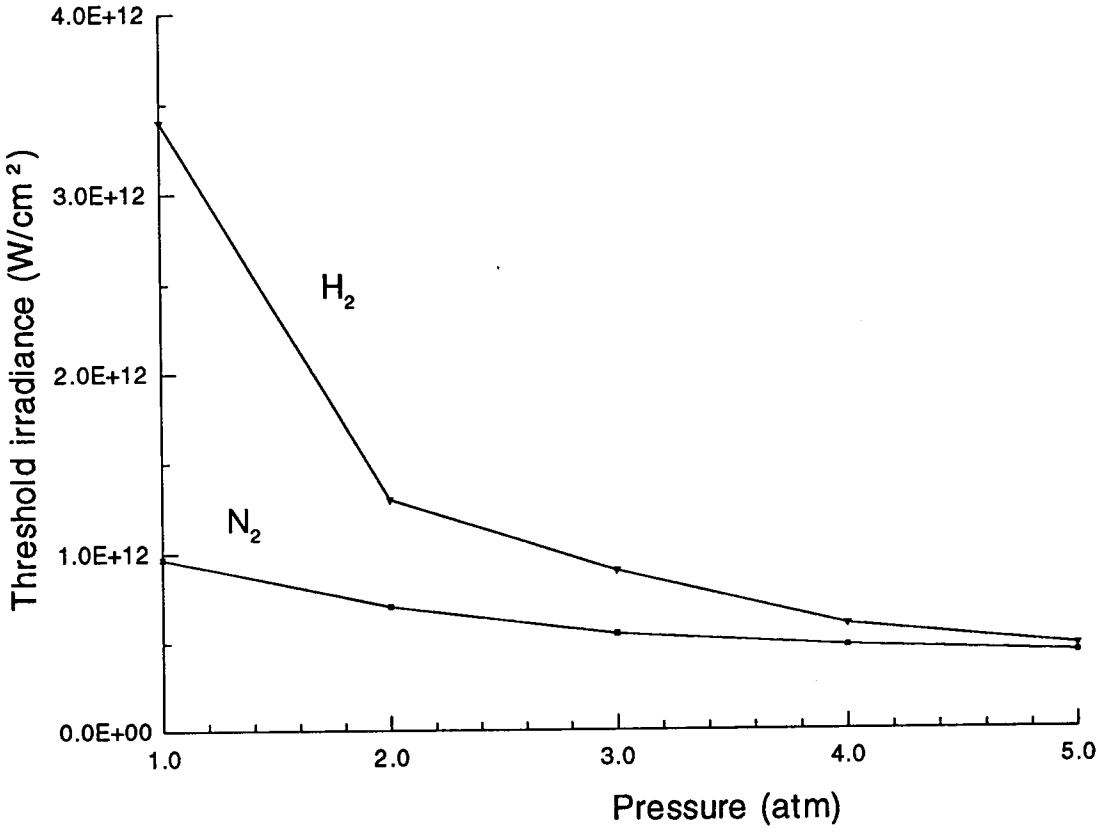


Figure 3.10. Breakdown thresholds of hydrogen and nitrogen as a function of the pressure.

CHAPTER 4. CONCLUSION AND RECOMMENDATIONS

The purpose of this dissertation is to provide a valid explanation for the FEL experimental results where attempts to obtain plasma breakdown in hydrogen and nitrogen were unsuccessful. A particle simulation method was used to simulate the charged particle behavior during the gas breakdown process. The gasdynamic expansion effects were included by numerically solving Euler's equations. The calculated results are consistent with the FEL experimental observations and are summarized below:

1. The argon plasma breakdown in the free electron laser experiments did not occur during a single 10 picosecond pulse, but was the result of a cumulative growth of the electron density during approximately 27 pulses. During those pulses, both recombination and gasdynamic expansion effects can be neglected for argon due to the relatively high energy of the electrons and low energy absorption by the gas.

2. Hydrogen and nitrogen did not reach breakdown because the electron energy decayed more rapidly in these gases during the 46 ns time between pulses. The rapid decay of electron energy in nitrogen occurs because of inelastic collisions. The rapid decay of electron energy in hydrogen occurs because of inelastic vibration collisions and a larger energy loss in elastic collisions due to the small mass of the hydrogen molecule. The reduction of electron energy

permits a much faster electron recombination rate which depletes the electron density during the 46 ns between pulses.

3. The heating of the gas through elastic and inelastic collisions during this 46 ns off time can cause a shock wave to form in argon when the electron number density is greater than $10^{17} cm^{-3}$. This shock effect reduces the electron number density in the focal volume and also reduces the collision frequency between electrons and neutral particles, but this effect becomes significant only after plasma breakdown has been achieved.

It is planned to extend this study by including the MPI process in the theoretical models. Then these models can be used to predict gas breakdown induced by a picosecond pulse laser over a wide range of conditions. Another interesting improvement would be to change the optical field distribution in the focal volume from a uniform distribution to a gaussian form such as was used in the FEL experiments. Since the gaussian beam produces a higher irradiance at the center of the focal volume than the edge of the laser beam, the electron density would be much higher at the center of the focal volume. At breakdown conditions, if the cell size is uniform over the entire computational domain, the small cell volume near the center would result in a very high electron density at that cell, which would be much higher than the average electron density in the focal volume. This will cause the potential from the PIC method at that cell to be abnormally high. To avoid this problem, a nonuniform cell size should be used in the breakdown

calculation using the particle simulation method with a gaussian distribution of the laser beam. This improvement would necessarily increase the computational time. In addition, the rate of the electron density diffusion and the electron energy diffusion will increase due to the high electron energy and high electron density at the center of the focal region.

BIBLIOGRAPHY

BIBLIOGRAPHY

Agostini, P., Barjot, G., Mainfray, G., Manus, C., and Morellec, J., "Multiphoton ionization of hydrogen and rare gases," *IEEE J. Quant. Electron., QE-4*: 667-669, (1968).

Batenin, V. and Chinov, V., "Electron bremsstrahlung in the field of argon or helium atoms," *Sov. Phys. JEPT*, **34**: 30-33, (1972).

Bebb, H. and Gold, A., "Multiphoton ionization of hydrogen and rare gas atoms," *Phys. Rev.*, **143**: 1-24, (1965).

Birdsall, C., "Particle-in-Cell Charged-Particle Simulations, Plus Monte Carlo Collisions With Neutral Atoms, PIC–MCC," , *IEEE Translations on plasma science*, vol 19. no. 2, pp. 65-85, (1991).

Birdsall, C. and Bridges, W., "Space-Charge instabilities in electron diodes and plasma converters," *J. Appl. Phys.*, vol. 32, pp. 2611-2618, (1961).

Birdsall, C. and Fuss, D., "Clouds-in-Clouds, Clouds-in-Cells physics for many-body plasma simulation," *Journal of Computational Phsics* **3**: 494-511, (1969).

Birdsall, C. and Langdon, A., *Plasma Physics via Computer Simulation,* McGraw-Hill Book Company, New York, (1985).

Bornath, Th., Schlanges, M., Kremp, D., "Influence of nonideality effects on the ionization and recombination coefficients of a reacting hydrogen plasma," *Contib. Plasma Phys.* 28, 1: 57-66, (1988).

Buneman, O., "Dissipation of currents in ionised media," *Phys. Rev.*, vol. 115, pp. 503-517, (1959).

Burden, R. L. and Faires, J. D., "Numerical analysis," Third edition, Prindle, Weber & Schmidt, Boston, (1985).

Burger, P., "Theory of large-Amplitude oscillation in the one-dimensional low-pressure cesium thermionic converter," *J. Appl. Phys.*, vol. 36, pp. 1938-1943, (1965).

Burger, P., Dunn, D., and Halsted, A., "Computer experiments on the randomization of electrons in a collisionless plasma," *The Physics of Fluids*, vol. 8, No. 12, pp. 2263-2272, (1965).

Canavan, G., Proctor, W., Nielson, P., and Rockwood, S., "CO_2 laser air breakdown calculations," *IEEE J. Quant. Electron.*, **8**: 564-565, (1972).

Clark, K. and Incropera F., "Electron-ion recombination - a critical review of theories and experimental results for an argon plasma," In American Society of Mechanical Engineers, Symposium on Thermophysical Properties, 5th, Newton, Mass., September 30-October 2, 1970, Proceedings. (A70-44426 23-23) Edited by C. F. Bonilla. New York, American Society of Mechanical Engineers, p.331-341, (1970).

Dalgarno, A. and Lane, N., "Free-free transitions of electrons in gases," *Astrophys. J.*, **145**: 623-633, (1966).

Damon, E. and Thomlinson, R., "Observation of ionization of gases by a ruby laser," *Appl. Opt.*, **2**: 546-547,(1963).

Dawson, J., "One-dimensional plasma models", *Phys. Fluids*, vol. 5, pp. 445-459, (1962).

Delone, N.B., "Multiphoton ionization of atoms", *Sov. Phys. Usp.*, 18: 169-189, (1975).

Eldridge, O. and Feix, M., "One-dimensional plasma model at thermodynamic equilibrium," *Phys. Fluids*, vol. 5, pp. 1076-1080, (1962).

Geltman, S., "Free-free radiation in electron-neutral atom collisions," *J. Quant. Spectr. Rad. Trans.*, **13**: 601-613, (1973).

Generalov, N., Zimakov, V., Kozlov, G., Masyukov, and Raizer, Yu., *JETP Lett.*, vol. 11, No. 7, pp. 228-231, (1970).

Gontier, Y. and Trahin, M., "Multiphoton ionization of atomic hydrogen in the ground state," *Phys. Rev.*, **172**: 83-87, (1968).

Grey-Morgan, C., "Laser induced breakdown of gases," *Rep. Prog. Phys.*, **38**: 621-665, (1975).

Hinnov, E. and Hirschberg, J., "Electron-ion recombination in dense plasmas," *Phys. Rev.*, vol. 125, No. 3, pp. 797-801, (1962).

Hockney, R., "A fast direct solusion of poisson's equation using Fourier analysis," *J. Assoc. Comput. Mach.*, vol. 12, pp. 95-113, (1965).

Hockney, R. and Eastwood, J., *Computer simulation using particles,* Adam Hilger, Bristol and New York, (1988).

Holt, E. and Haskell, R., *Foundations of Plasma dynamics*, The Macmillan Company, New York, (1965).

Hsieh, K. C., "Assessment of numerical techniques for unsteady flow calculations," AIAA Computational Fluid Dynamics Conference, 9th, Buffalo, NY, June 13-15, 1989, technical paper A89-4177618-02 and Washington, DC, AIAA, 1989, pp. 280-291, AIAA Paper 89-1956.

Huxley, L. and Crompton, R. W., " The diffusion and drift of electrons in gases," Jhon Wiley & Sons, New York, (1974).

Jeng, S. M. and Keefer, D., "Theoretical investigation of laser-sustained argon plasmas," *J. Appl. Phys.*, **60**: 2272, (1986).

Jeng, S. M., Keefer, D., Welle, R., and Peters, C., Laser-sustained plasmas in a forced convective argon flow. Part II: Comparison of numerical model with experiment," *AIAA J.*, **25**: 1224-1230, (1987).

Kas'yanov, V. and Starostin, A., "On the theory of bremssthahlung of slow electrons on atoms," *Sov. Phys. JETP*, **21**: 193-198, (1965).

Keldysh, T., "Ionization in the field of a strong electromagnetic wave," *Sov. Phys. JETP*, **20**: 1307-1314, (1965).

Keefer, D., "Laser-sustained plasmas," in *Laser-induced plasmas and applications* (L. J. Radziemski and D. A. Cremers ed.), Marcel Dekker Inc., New York and Basel, pp. 169-206, (1989).

Keefer, D., Elkins, R., Peters, C., and Jones, L., "Laser thermal propulsion," Orbit-Raising and Maneuvering Propulsion: Research Status and Needs, *AIAA Progress in Astronautics and Aeronautics series*, Vol. 89 (L. H. Caveny, ed.), AIAA, Washinhton, D.C., p. 129, (1984).

Keefer, D., Sedghinasab, A., Wright, N., and Zhang, Q., "Laser propulsion using free electron lasers," AIAA/DGLR/JSASS, 21st International Electric Propulsion Conference, July 18-20, 1990, Orlando, FL, AIAA 90-2636, (1990).

Keefer, D., Welle, R., and Peters, C., "Power absorption in laser- sustained argon plasmas", *AIAA J.*, **24**: 1663, (1986).

Kieffer, L. J., "A compilation of electron collision cross section data for modeling gas discharge lasers," University of Colarado, (1973).

Kroll, N. and Watson, K., "Theoretical study of ionization of air by intense laser pulses," *Phys. Rev.*, **A5**: 1883-1905, (1972).

Lowke, J., Phelps, A. and Irwin, B., "Predicted electron transport and operating characteristics of CO_2, N_2, He laser mixtures," *J. Appl. Phys.*, **44**: 4664-4671, (1973).

Massey, H., "Electron collisions with molecules and photon-ionization," in Electronic ionic impact phenomena (H. Massey, E. Burhop, and H. Gilbody ed.), vol. 2, Oxford, at the Clarendon press, (1969).

McClung, F. J. and Hellwarth, R. W., "Giant optical pulsations from ruby," *J. Appl. Phys.*, **33**: 828-829, (1962).

McDaniel, E. W., " Collision phenomena in ionized gases," John Wiley & Sons, Inc., New York, (1974).

Meyerand, Jr., and Haught, A. F., *Phys. Rev. Lett.*, **11**: 401, (1963).

Minck, R. W., *J. Appl. Phys.*, **35**: 252, (1964).

Mitchner, M. and Kruger, C., "Partially ionized gases," John Wiley & sons, New York, (1973).

Morse, R. L. and Nielson, C. W., "Numerical simulation of warm two-beam plasma," *The Physics of Fluids*, vol. 12, No. 11, pp. 2418-2425, (1969).

Morton, V., "Multiphoton absorption in monatomic gases," sl Proc. Phys. Soc. (London) **92**: 301-309, (1967).

Nelsen, P., Veyrie, M., and Durand, Y., *Phys. Lett.*, **13**: 226, (1964).

Nighan, W., "Electron energy distributions and collision rate in electrically excited N_2, CO, and CO_2," *Phys. Rev.*, **A2**: 1989-1999, (1970).

Phelps, A. V., "Theory of Growth of ionization during breakdown," in *Physics of Quantum Electronics* (P. Kelley, B. Lax, and P. Tannenwald, eds.), McGraw-Hill, New York, pp. 538-547, (1966).

Raizer, Yu. P., *Laser induced discharge phenomena*, Consultants Bureau, New York, (1977).

Rees, H. D., "Calculation of steady-state distribution function by exploting stability," *Phys. Lett.*, vol. 26A, pp. 416-417, (1968).

Rees, H. D., "Calculation of distribution functions by exploting the stability of the steady state," *J. Phys. Chem. Solids*, vol. 30, pp. 643-655, (1969).

Rockwood, S. D., "A comparison of laser induced gas breakdown thresholds in various gases," Unpublished paper, (1975).

Rosen, D. and Weyl, G., "Laser induced breakdown in N_2 and the rare gases at 0.53 and 0.35 μm," *J. Phys. D. Appl. Phys.*, **20**: 1264-1276, (1987).

Ruyten, W. M., "Density-conserving shape factors for particle simulations in cylindrical and spherical coordinates," submmited to the *Journal of computational physics*, (1992).

Smith, D. C. and Meyerand, R. G., "Laser radiation induced gas breakdown," in *Principles of laser Plasma* (G. Bekefi, ed.), Willey, New York, pp. 457-507, (1976).

Tomlinson, R. G., Damon, E.K., and Buscher, H. T., "Physics of Quantum Electronics" (Proc. Intern. Conf., San Juan, Puerto Rico, 1965), ed. by P.L. Kelley et al., McGraw-Hill, New York (1966), p. 520.

Voronov, G., "Dependence of the probability of multiphotoionization of atoms on the photon flux intensity," *Sov. Phys. JETP*, **24**: 1009-1011, (1967).

Voronov, G. S. and Delone, N. B., *JETP Lett.*, **1**(2): 66-68 (1965).

Welle, R., Keefer, D., and Peters, C., "Energy conversion effiviency in high-flow laser-sustained argon plasmas," *AIAA J.*, **25**: 1093, (1987).

Weyl, G., "Ionization path formation in gases using a laser with a retract able focus," *J. Phys. D. Appl. Phys.*, **12**: 33-49, (1979).

Weyl, G., "Physics of Laser-induced breakdown: An update," in *Laser-induced plasmas and applications* (L. J. Radziemski and D. A. Cremers ed.), Marcel Dekker Inc., New York and Basel, pp. 1-67, (1989).

Weyl, G. and Rosen, D., "Laser induced breakdown in argon at 0.35 μm, theory and experiments," *Phys. Rev.* **A31**: 2300-2313, (1985).

Yamabe, C., Buckman, C. S., and Phelps, A. V., "Measurement of free-free emission from low-energy electron collisions with argon," *Phys. Rev.* **A27**: 1345-1352, (1982).

Zel'dovich, Y. B. and Raizer, Y. D., "Cascade ionization by a light pulse," *Sov. Phys. JETP*, **20**: 772-780, (1965).

APPENDIXES

APPENDIX A

FORTRAN PROGRAM

```
      program pcsr
c     Plasma Computer Simulation 1-D(r)
      common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1              sextmax,extengy0,extengy1,extengy2,
     1              sionmax,ionengy0,ionengy1,ionengy2,
     1              achrgx,bchrgx,ascat,bscat
      common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
      common /randm1/ ranb,rans
      common /temp/ nt,sab,ncent,omiga1
      common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
      common /abc/ ax(9001),bx(9001),cx(9001),volum(9001)
      common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
      common /kkk/ keyn,keya,kcom
      common /exp/ e0,ef,ssde,sdevs(501),a0(501),npr
      common /main/ nt2,n1,n4,ne
      common /phic/ phi(9001)
      common /ff/ fm,fm1,fmi,fm2
      real*8 ranb,rans,ranf
      real length,m,ionengy0,ionengy1,ionengy2,ivscale,m1,nc2p
c
      read(99,*) key
c
      if(key.eq.1) then
c
cccccccc  input initial data
      call start
c
cccccccc  calculate cross section
      call xsect
c
      n1=0
      s_ext=0.
      keya=1
      kcom=1
      keyn=0
      else
      call resta1
      end if
c
cccccccc  circulation for pulse
  50  ntt=0
      timet=0.
      sab=0.
      ncent=0
      n1=n1+1
      print *, 'pulse number',n1
      print *, '   time(ps)        computational particles   real particles'
      na=nt2*(n1-1)
      do 60 i=1,ng
  60  sde(i)=0.
c
cccccccc  circulation for time step in one pulse
 100  ntt=ntt+1
      nt=na+ntt
      timet=timet+dt*1.e+12
c
cccccccc  calculation of collision and particle movement
      call time(ntt,ss12)
c
      sn1=0.
      do i=1,np(1)
      sn1=sn1+rep(i,1)
      end do
      if(mod(nt,n4).eq.0) then
```

```
        print *, timet,np,sn1
        end if
        call check1(ss12)
c
cccccccc  recombination
c       call check1(ss12)
c       call recomb(sn1,ss12,ntt)
c
cccccccc  calculation of electric field
          call field(ntt,n1)
c
cccccccc  calculate energy distribution
        if(ntt.eq.nt1) call check2(n1)
        if(ntt.eq.nt2) call check2(n1)
c
cccccccc  pulse end
        if(ntt.ge.nt2) goto 110
c
cccccccc  regroup particles
        if(np(1).gt.maxlen) then
201     format(6e12.5)
        call reduce(ne)
        call field(ntt,n1)
        end if
        goto 100
c
c
  110   continue
        ssde=0.
        do 111 i=1,ng
        sdev(i)=sde(i)/volum(i)
  111   ssde=ssde+sde(i)
        print *,'average electron energy (eV)',ss12
c
cccccccc  pulse contral
       if(n1.ge.nn) goto 120
c
cccccccc  gas breakdown
        if(sn1 .gt. 1.e+12) then
        print *, 'stop 2'
        goto 120
        end if
c
        call check1(ss12)
        if(sn1.lt.1.e+10) then
        call reca(sn1,ss12)
        else
c       call connect(sn1,ss12)
        end if
c
c
        if(sn1.gt.1.e+07.and.keyn.eq.0) then
         keyn=1
cccccccc  reduce time step
         call tstep(nt2)
         n4=50
        else if(sn1.gt.2.e+08.and.keyn.eq.1) then
         keyn=2
         call tstep(nt2)
         maxlen=60000
         call xgrid
         call field(ntt,n1)
         ne=20
         kcom=12
         n4=100
        else if(sn1.gt.5.e+09.and.keyn.eq.2) then
         keyn=3
cccccccc  reduce grid spacing
         call tstep(nt2)
         call xgrid
         call field(ntt,n1)
         maxlen=90000
         ne=10
         kcom=36
        else if(keyn.eq.3) then
         keyn=4
         call tstep(nt2)
        else
        end if
        if(keyn.ge.1) call output
        goto 50
c
cccccccc  output results
  120   continue
c       call output
```

```
      stop
      end
c
c
c
      subroutine start
      common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1              sextmax,extengy0,extengy1,extengy2,
     1              sionmax,ionengy0,ionengy1,ionengy2,
     1              achrgx,bchrgx,ascat,bscat
      common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
      common /ff/ fm,fm1,fmi,fm2
      common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
      common /abc/ ax(9001),bx(9001),cx(9001),volum(9001)
      common /randm1/ ranb,rans
      common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
      common /main/ nt2,n1,n4,ne
      common /gauss/ der(9001),power
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      common /recom/ dent,e0,beta
      common /res/ nend2,dt1
      real*8 ranb,rans,ranf,rot
      real ivscale,m1,nc2p
      real length,m,ionengy0,ionengy1,ionengy2
c
c
      open (unit=1,file='pcsr.dat',status='unknown')
      rewind 1
      read(1,*) nt1,nt2,nn,lran,n4,ne
      read(1,*) nsp,nc,nc2p,length,area,epsilon
      read(1,*) dt,de,bk,gtemp,gden,q,m
      read(1,*) pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      read(1,*) selsmax,elsengy0,elsengy1,elsengy2
      read(1,*) sextmax,extengy0,extengy1,extengy2
      read(1,*) sionmax,ionengy0,ionengy1,ionengy2
      read(1,*) achrgx,bchrgx,ascat,bscat
      read(1,*) fm,fm1,fmi,fm2
      read(1,*) power
      read(1,*) nend2,evk
      read(1,*) temp
      read(1,*) yl
      close (unit=1)
c
c
      do 50 l=1,lran
      ra=ranf(0.d0)
50    continue
c
      dx=length/float(nc)
      xnc=float(nc)
      xnm=xnc*2./3.
      ng=nc+1
      evscale=abs(.5*m(1)/q(1))
      ivscale=abs(.5*m(2)/q(2))
      vele=sqrt(3.*bk*gtemp/m(1)+de/evscale)
      veli=sqrt(3.*bk*gtemp/m(2))
c
      achrgx=achrgx*gden
      bchrgx=bchrgx*gden*sqrt(2.*q(2)/m(2))
      ascat=ascat*gden
      bscat=bscat*gden*sqrt(2.*q(2)/m(2))
c
cccccccc  create initial electron and ion (position and velocity)
c
c     xr=ranf(0.d0)*xnc
      xr=1.
      x(1,1)=xr
      x(1,2)=xr
      do 100 isp=1,nsp
      ra=ranf(0.d0)
      vx(1,isp)=ra
      ra=ranf(0.d0)
      vy(1,isp)=ra
      ra=ranf(0.d0)
100   vz(1,isp)=ra
      dum=sqrt(vx(1,1)*vx(1,1)+vy(1,1)*vy(1,1)+vz(1,1)*vz(1,1))
      vele=vele/dum
      dum=sqrt(vx(1,2)*vx(1,2)+vy(1,2)*vy(1,2)+vz(1,2)*vz(1,2))
      veli=veli/dum
      vx(1,1)=vx(1,1)*vele
```

```
      vy(1,1)=vy(1,1)*vele
      vz(1,1)=vz(1,1)*vele
      vx(1,2)=vx(1,2)*veli
      vy(1,2)=vy(1,2)*veli
      vz(1,2)=vz(1,2)*veli
c
c
      do 200 j=1,ng
  200 e(j)=0.
c
      do 300 isp=1,nsp
      np(isp)=1
      q1(isp)=q(isp)*nc2p
c     q1(isp)=q(isp)*nc2p/area
      do 350 i=1,20000
      rep(i,isp)=1.
  350 rtc(i,isp)=0.
      do 360 i=1,ng
  360 dens(i,isp)=0.
  300 m1(isp)=m(isp)*nc2p
c
      dddx=dx*dx*dx
      ncm1=nc-1
      do 80 i=1,ncm1
      h1=float(i)+.5
      h2=h1-1.
      h3=h1**3-h2**3
      a1=3.*h1*h1/h3
      b1=3.*h2*h2/h3
      ax(i+1)=-b1
      bx(i+1)=a1+b1
      cx(i+1)=-a1
      volum(i+1)=2.*twopi*h3*dddx/3.
   80 continue
      ax(1)=0.
      bx(1)=3./.5
      cx(1)=-bx(1)
      volum(1)=2.*twopi*(.5)**3*dddx/3.
      volum(ng)=2.*twopi*(float(nc)**3-(float(nc-1)+.5)**3)
     1          *dddx/3.
c     print *,'ax,bx,cx'
c     write(6,10) (ax(i),i=1,ng)
c     write(6,10) (bx(i),i=1,ng)
c     write(6,10) (cx(i),i=1,ng)
c     write(6,10) (volum(i),i=1,ng)
c
      do 400 isp=1,nsp
      qm(isp)=q(isp)/m(isp)
      do 400 i=1,ng
      vq(i,isp)=q1(isp)/volum(i)
  400 eqm(i,isp)=0.
      do 450 i=1,ng
  450 vn(i)=nc2p/volum(i)
      se=dx*dx/epsilon
      sm=m(2)/(m(1)+m(2))
      frengi=3.*bk*gtemp/(2.*q(2))
      cm2=2.*m(1)/m(2)
      cm2c=cm2
c     print *,'cm2c',cm2c
c
      dt1=4.6e-08/float(nend2)
c
      do 15 i=1,3
      beb(i)=2.*yl(i)-yl(i+1)
   15 bea(i)=(yl(i)-beb(i))/temp(i)
c
c
      ww=length*2./3.
      delta=ww
      dd=delta*delta
      dr=ww/float(nc*2/3)
      ai0=2.*power/(pi*dd)
c     print *,'ai0=',ai0
c
      do 105 i=1,nc
      r=float(i-1)*dr
      yr=2.*r*r/dd
      der(i)=5.e-12*ai0*exp(-yr)
  105 continue
c     write(6,10) (der(i),i=1,nc)
c
c
   10 format(6e12.5)
      return
      end
```

```
c
c
c
      subroutine move(tc,isp,ntt,i)
      common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
      common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
      common /temp/ nt,sab,ncent,omigal
      common /randm1/ ranb,rans
      dimension a(9001)
      real*8 ranb,rans,ranf,rot
      real length,m,ivscale,m1,nc2p
c
c
c     dum=vx(i,isp)*vx(i,isp) +vy(i,isp)*vy(i,isp)
c    1          +vz(i,isp)*vz(i,isp)
c     engy     = evscale*dum
c     print *, 'engy,x(i,isp),1',i,isp,engy,x(i,isp)
      if(x(i,isp).gt.0..and.x(i,isp).lt.xnc) then
      j=ifix(x(i,isp))
      a(j+1)=eqm(j+1,isp)*tc
      a(j+2)=eqm(j+2,isp)*tc
      sj=float(j)
      sjp=sj+1.
      s1=x(i,isp)-sj
      s=x(i,isp)*s1
      s=2.5*s/(2.*sjp*sjp+sjp*sj+2.*sj*sj)
      s=(sjp-x(i,isp))*(1.-s)/(sjp-sj)
      atemp=a(j+1)*s + a(j+2)*(1-s)
ccccccccc  velocity changed by electric field
      vptemp=sqrt(vy(i,isp)*vy(i,isp)+vz(i,isp)*vz(i,isp))
      cthe=x(i,isp)*vptemp
      vx(i,isp)=vx(i,isp)+atemp+vptemp*vptemp*tc/(x(i,isp)*dx)
c     vx(i,isp)=vx(i,isp)+atemp
      if(x(i,isp).lt.0.) print *,'x  =  ',x(i,isp)
ccccccccc  movement of particles
      x(i,isp)=x(i,isp)+vx(i,isp)*tc/dx
c     x1=x(i,isp)+vx(i,isp)*tc/dx
c     y1=vptemp*tc/dx
c     x(i,isp)=sqrt(x1*x1+y1*y1)
c     if(x(i,isp).eq.0.) then
c       cosa=1.
c       sina=0.
c     else
c       cosa=x1/x(i,isp)
c       sina=y1/x(i,isp)
c     end if
c     v1=vx(i,isp)*cosa+vptemp*sina
c     v2=-vx(i,isp)*sina+vptemp*cosa
c     vx(i,isp)=v1
c     vptemp=v2
      vptemp=cthe/abs(x(i,isp))
      s=ranf(0.d0)
      vy(i,isp)=s*vptemp
      vz(i,isp)=sqrt(1.-s*s)*vptemp
      end if
c
ccccccccc  geometry symmetric
      if(x(i,isp).lt.0.) then
      print *, 'x,vx,atemp',x(i,isp),vx(i,isp),atemp
      x(i,isp) = -x(i,isp)
      vx(i,isp) = -vx(i,isp)
      vy(i,isp) = -vy(i,isp)
      vz(i,isp) = -vz(i,isp)
      end if
c
      return
      end
c
c
c
      subroutine field(ntt,n1)
      common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
      common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
      common /abc/ ax(9001),bx(9001),cx(9001),volum(9001)
      common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
```

```
      common /temp/ nt,sab,ncent,omiga1
      dimension srho(9001,2),rho(9001),gam(9001),phi1(9001)
      common /phic/ phi(9001)
      real length,m,m1,ivscale,nc2p
c
c
c     if(ntt.eq.55) print *,'2'
c
      do 50 i=1,ng
      phi1(i)=phi(i)
      phi(i) = 0.
 50   rho(i)=0.
c
cccccccc  accumulate charge density linearly to grids
      do 100 isp=1,nsp
      do 110 i=1,ng
 110  srho(i,isp)=0.
      do 120 i=np(isp),1,-1
      if(x(i,isp).gt.0..and.x(i,isp).lt.xnc) then
      j=ifix(x(i,isp))
      if(j.le.3.and.isp.eq.1) ncent=ncent+1
      sj=float(j)
      sjp=sj+1.
      s1=x(i,isp)-sj
      s=x(i,isp)*s1
      s=2.5*s/(2.*sjp*sjp+sjp*sj+2.*sj*sj)
      s=(sjp-x(i,isp))*(1.-s)/(sjp-sj)
      srho(j+1,isp)=srho(j+1,isp)+s*rep(i,isp)
      srho(j+2,isp)=srho(j+2,isp)+(1.-s)*rep(i,isp)
      end if
 120  continue
      do 150 i=1,ng
cccccccc  particle density
      dens(i,isp)=srho(i,isp)/volum(i)
 150  srho(i,isp)=srho(i,isp)*vq(i,isp)
      do 160 i=1,ng
      rho(i)=rho(i)+srho(i,isp)
 160  continue
 100  continue
c
cccccccc  solve poisson equation
      phi(1)=se*rho(1)/bx(1)
      phi(ng)=0.
      gam(1)=-1.
      cx(nc)=0.
c
      do 200 i=2,nc
      bet=bx(i)-ax(i)*gam(i-1)
      gam(i)=cx(i)/bet
 200  phi(i)=(se*rho(i)-ax(i)*phi(i-1))/bet
      do 300 i=nc,1,-1
 300  phi(i)=phi(i)-gam(i)*phi(i+1)
c
c
cccccccc  calculate E(x) from phi(x)
      s=0.5/dx
      do 400 i=2,nc
 400  e(i)=(phi(i-1)-phi(i+1))*s
      e(1)=0.
      e(ng)=(phi(nc)-phi(ng))/dx
c     e(ng)=(phi(nc)-phi(ng))/dx + rho(ng)*.5*dx/epsilon
c
      do 500 isp=1,nsp
      do 500 i=1,ng
 500  eqm(i,isp)=e(i)*qm(isp)
c
c     if(nt1.ge.50) goto 520
      if(nt1.ge.50) goto 515
      if(mod(ntt,200).eq.0) write(16,*) n1,nt,np
      if(mod(ntt,200).eq.0) write(16,501)
      if(mod(ntt,200).eq.0) write(16,502) (phi(i),i=1,ng)
      if(mod(ntt,200).eq.0) write(16,504)
      if(mod(ntt,200).eq.0) write(16,502) (dens(i,1),i=1,ng)
      if(mod(ntt,200).eq.0) write(16,502) (dens(i,2),i=1,ng)
      if(mod(ntt,200).eq.0) write(16,505)
      if(mod(ntt,200).eq.0) write(16,502) (e(i),i=1,ng)
      goto 520
 510  if(nt1.ge.1000) goto 515
      if(mod(ntt,2000).eq.0) write(16,*) n1,nt,np
      if(mod(ntt,2000).eq.0) write(16,501)
      if(mod(ntt,2000).eq.0) write(16,502) (phi(i),i=1,ng)
      if(mod(ntt,2000).eq.0) write(16,504)
      if(mod(ntt,2000).eq.0) write(16,502) (dens(i,1),i=1,ng)
      if(mod(ntt,2000).eq.0) write(16,502) (dens(i,2),i=1,ng)
      if(mod(ntt,2000).eq.0) write(16,505)
```

```
        if(mod(ntt,2000).eq.0) write(16,502) (e(i),i=1,ng)
        goto 520
  515   if(nt1.ge.2000) goto 516
        if(mod(ntt,4000).eq.0) write(16,*) n1,nt,np
        if(mod(ntt,4000).eq.0) write(16,501)
        if(mod(ntt,4000).eq.0) write(16,502) (phi(i),i=1,ng)
        if(mod(ntt,4000).eq.0) write(16,504)
        if(mod(ntt,4000).eq.0) write(16,502) (dens(i,1),i=1,ng)
        if(mod(ntt,4000).eq.0) write(16,502) (dens(i,2),i=1,ng)
        if(mod(ntt,4000).eq.0) write(16,505)
        if(mod(ntt,4000).eq.0) write(16,502) (e(i),i=1,ng)
  516   if(mod(ntt,8000).eq.0) write(16,*) n1,nt,np
        if(mod(ntt,8000).eq.0) write(16,501)
        if(mod(ntt,8000).eq.0) write(16,502) (phi(i),i=1,ng)
        if(mod(ntt,8000).eq.0) write(16,504)
        if(mod(ntt,8000).eq.0) write(16,502) (dens(i,1),i=1,ng)
        if(mod(ntt,8000).eq.0) write(16,502) (dens(i,2),i=1,ng)
        if(mod(ntt,8000).eq.0) write(16,505)
        if(mod(ntt,8000).eq.0) write(16,502) (e(i),i=1,ng)
  520   continue
c
c
ccccccccc  particle density contral
        sn1=0.
        do i=1,np(1)
        sn1=sn1+rep(i,1)
        end do
        if(phi(1).gt.1.e+6.or.sn1.gt.1.e+12) then
        write(6,501)
        write(6,502) (phi(j),j=1,ng)
        write(6,504)
        write(6,502) (dens(j,1),j=1,ng)
        write(6,502) (dens(j,2),j=1,ng)
        print *,'e'
        write(6,502) (e(j),j=1,ng)
        ssde=0.
        do 550 i=1,ng
        sdev(i)=sde(i)/volum(i)
  550   ssde=ssde+sde(i)
        print *,'ssde',ssde
        print *,'sdev'
        write(6,502) (sdev(i),i=1,ng)
        print *,'s_ext',s_ext
        print *, 'sab', sab
        print *,'stop 5'
        stop
        end if
c
501     format(5h phi )
502     format(6e12.5)
503     format(5h rho )
504     format(6h dens )
505     format(3h e )
c       if(ntt.eq.55) print *,'2'
        return
        end
c
c
c
        subroutine time(ntt,ss12)
        common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
        common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1                pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
        common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1                sextmax,extengy0,extengy1,extengy2,
     1                sionmax,ionengy0,ionengy1,ionengy2,
     1                achrgx,bchrgx,ascat,bscat
        common /cs1/ sels(500),sext(500),sion(500)
        common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
        common /ff/ fm,fm1,fmi,fm2
        common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
        common /randm1/ ranb,rans
        common /temp/ nt,sab,ncent,omiga1
        common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
        real*8 ranb,rans,ranf
        real length,m,m1,ivscale,ionengy0,ionengy1,ionengy2,nc2p
c
        do 50 isp=1,nsp
        kkn=np(isp)
        j=0
  100   j=j+1
        if(j.le.np(isp)) then
        if(j.gt.kkn) goto 101
```

```
cccccccc  judge old free fly time in new dt
        if(rtc(j,isp).ge.dt) then
        rtc(j,isp)=rtc(j,isp)-dt
        call move(dt,isp,ntt,j)
        goto 100
        end if
        if(rtc(j,isp).eq.0.) goto 101
        call move(rtc(j,isp),isp,ntt,j)
        if(x(j,isp).gt.xnc) goto 100
        call collis(ntt,isp,j,ss12)
  101   random1=ranf(0.d0)
        if(random1.eq.0.) random1=1.e-05
cccccccc  give new free fly time
        if(isp.eq.1) then
        i=ifix(x(j,1))+1
        if(dens(i,1).ge.2.e+23) then
        dum=vx(j,1)*vx(j,1) +vy(j,1)*vy(j,1) +vz(j,1)*vz(j,1)
        engy     = evscale*dum
        call colfr(engy,dens(i,1),fm1)
        tc=-alog(random1)/fm1
        else
        tc=-alog(random1)/fm
        end if
        else
        tc=-alog(random1)/fmi
        end if
        tcp=tc+rtc(j,isp)
        if(tcp.lt.dt) then
        call move(tc,isp,ntt,j)
        if(x(j,isp).gt.xnc) goto 100
        call collis(ntt,isp,j,ss12)
        rtc(j,isp)=tcp
        goto 101
        else
        dtb=dt-rtc(j,isp)
        rtc(j,isp)=tcp-dt
        call move(dtb,isp,ntt,j)
        end if
        goto 100
        end if
   50   continue
c
c
cccccccc  eliminate outside electron
        do 200 isp=1,nsp
        nnp=np(isp)
        nnpp=nnp
        i=0
  201   i=i+1
        if(i.le.nnp) then
        if (x(i,isp).ge.xnc) then
        x(i,isp) = x(nnp,isp)
        vx(i,isp) = vx(nnp,isp)
        vy(i,isp) = vy(nnp,isp)
        vz(i,isp) = vz(nnp,isp)
        rtc(i,isp)=rtc(nnp,isp)
        rep(i,isp)=rep(nnp,isp)
        nnp=nnp-1
        i=i-1
        end if
        goto 201
        end if
        np(isp)=nnp
c       if(nnp.ne.nnpp) print *,'isp,nnpp,nnp',isp,nnpp,nnp
  200   continue
c
        return
        end
c
c
c
        subroutine collis(ntt,isp,j,ss12)
        common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
        common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1                pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
        common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1                sextmax,extengy0,extengy1,extengy2,
     1                sionmax,ionengy0,ionengy1,ionengy2,
     1                achrgx,bchrgx,ascat,bscat
        common /cs1/ sels(500),sext(500),sion(500)
        common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
        common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
        common /ff/ fm,fm1,fmi,fm2
```

```
        common /randm1/ ranb,rans
        common /temp/ nt,sab,ncent,omiga1
        common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
        common /gauss/ der(9001),power
        common /main/ nt2,n1,n4,ne
        real*8 ranb,rans,ranf
        real length,m,m1,ivscale,ionengy0,ionengy1,ionengy2,nc2p
c
        if(isp.eq.2) goto 100
c determine if a collision occurs
         dum=vx(j,1)*vx(j,1) +vy(j,1)*vy(j,1) +vz(j,1)*vz(j,1)
        engy     = evscale*dum
        vel      = sqrt(dum)
        ien      = ifix(engy*5. +.5)
        if(ien .ge.nemax) ien = nemax - 1
        if(ien .lt.1) ien = 1
        stot = sels(ien) + sext(ien) + sion(ien)
        fm2=fm
        l=ifix(x(j,1))+1
ccccccccc  E-N collision frequency
        fnu=vel*stot*gden
ccccccccc  E-I collision frequency
        if(dens(l,1).ge.2.e+23) then
        call coul(ss12,dens(l,1),fc)
        call colfr(engy,dens(l,1),fm1)
        fm2=fm1
        else
        fc=0.
        end if
        fac=(fnu+fc)/fm2
c       fnu=vel*stot*gden
c       fac=fnu/fm
        random1 = ranf(0.d0)
c-------  if collision occur
        if(random1.lt.fac) then
c       ijk=ijk+1
c       print *,'ijk',ijk
c       print *,'engy,ien,stot,fac,random1'
c       print *,engy,ien,stot,fac,random1
        random1=random1/fac
        fac1=fnu/(fnu+fc)
c-------  if collision is E-I
        if(random1.gt.fac1) then
        encm=engy*cm2c*rep(j,1)
        encm1=engy*cm2c
        sde(l)=sde(l)+encm
        engy=engy-encm1
        call newdir(engy,vel,vx(j,1),vy(j,1),vz(j,1))
c-------  if collision is E-N
        else
        random1=random1/fac1
c determine the type of collision and calculate new velocities, etc
        sig1 = sels(ien)/(stot +DBL_MIN)
      sig2 = (sels(ien) +sext(ien))/(stot +DBL_MIN)
c*******************************c
c    normalize velocity to get direction
        if(vel.gt. 1.e-30) then
        vx(j,1)=vx(j,1)/vel
        vy(j,1)=vy(j,1)/vel
        vz(j,1)=vz(j,1)/vel
        end if
c*******************************c
c   if the collision is elastic */
       if (random1 .le. sig1) then
        i=int(x(j,isp))+1
        encm=engy*cm2c*rep(j,1)
        encm1=engy*cm2c
        sde(i)=sde(i)+encm
        engy=engy-encm1
c   engy = engy*(1 - (1 - coschi)*2*m(1)/m(1+1)); */
        if(ntt.gt.nt1.or.x(j,1).gt.xnm) then
c       if(ntt.gt.nt1) then
        call newvel (engy,vel,vx(j,1),vy(j,1),vz(j,1))
        else
c       sab=sab+rep(j,1)*der(i)
c       engy=engy+der(i)
        sab=sab+rep(j,1)*de
        engy=engy+de
        vel=sqrt(abs(engy)/evscale)
       call     newvel(engy, vel, vx(j,1), vy(j,1), vz(j,1))
       end if
c*********************************/
c     if the collision is excitation */
      else if (random1 .gt. sig1 .and. random1 .le. sig2) then
        s_ext=s_ext+rep(j,1)
```

```
                engy = engy - extengy0
                  if(engy.lt.0.) engy=-engy
                vel  = sqrt(abs(engy)/evscale)
      call newvel(engy, vel, vx(j,1), vy(j,1), vz(j,1))

c*************************************/
c     if the collision is ionization,
c     add the released electron and ion first */
           else
         engy = engy - ionengy0
         if(engy.lt.0.) engy=-engy
c        engy = engy - ionengy0 -gtemp*alog(ranf(0.d0))
         random1 = ranf(0.d0)
         sengy = 10.0*tan(random1*atan(engy/20.0))
         engy = engy - sengy

         eengy = sm*engy
         vel = sqrt(abs(eengy)/evscale)
         k = np(1) +1
         vx(k,1) = vx(j,1)
         vy(k,1) = vy(j,1)
         vz(k,1) = vz(j,1)
         rtc(k,1)=rtc(j,1)
         call newvel(eengy, vel, vx(k,1), vy(k,1), vz(k,1))
               x(k,1) = x(j,1)
                rep(k,1)=rep(j,1)
         cengy =  engy - eengy
         vel = sqrt(abs(cengy)/ivscale)
         k = np(2) +1
         vx(k,2)= ranf(0.d0)
         vy(k,2)= ranf(0.d0)
         vz(k,2)= ranf(0.d0)
         rtc(k,2)=rtc(j,1)
         dum= sqrt(vx(k,2)*vx(k,2) +vy(k,2)*vy(k,2)+vz(k,2)*vz(k,2))

         vel = vel/dum
         vx(k,2) = vx(k,2)*vel
         vy(k,2) = vy(k,2)*vel
         vz(k,2) = vz(k,2)*vel
        x(k,2) = x(j,1)
         rep(k,2)=rep(j,1)


         np(1)=np(1)+1
         np(2)=np(2)+1
c        if(np(2).gt. 100000 .or. np(1).gt. 40000) print *,np
c        if(np(2).gt. maxlen .or. np(1).gt. maxlen) then
c        print *, 'stop 1'
c        stop
c        end if
c   then scatter the incident electron */
          vel = sqrt(abs(sengy)/evscale)
         call newvel(sengy, vel, vx(j,1), vy(j,1), vz(j,1))
         end if
         end if
         end if
         goto 200
c
  100   continue
c     determine if a collision occurs */
        vel = sqrt(vx(j,2)*vx(j,2) +vy(j,2)*vy(j,2) +vz(j,2)*vz(j,2))
     1        +DBL_MIN
         sig1= ascat*vel  +bscat
         sig2= achrgx*vel  +bchrgx

         stot= sig1 +sig2
         fnu=stot/fmi
         random1= ranf(0.d0)

         if(random1 .lt. fnu) then
c    determine the type of collision and calculate new velocities, etc */
         random1 = random1/fnu
         sig1 = sig1/(stot +DBL_MIN)

c*********************************/
c     if the collision is scattering */
         if(random1 .le. sig1 ) then
         random2 = ranf(0.d0)
         coschi = sqrt(1.-random2)
         sinchi = sqrt(1. - coschi*coschi)

         random2 = ranf(0.d0)
         tra=twopi*random2
         cosphi = cos(tra)
         if(tra.le.pi) then
```

```
          sinphi = sqrt(1. - cosphi*cosphi)
          else
          sinphi = -sqrt(1. - cosphi*cosphi)
          end if

          tempvx= vx(j,2)/vel
          tempvy= vy(j,2)/vel
          tempvz= vz(j,2)/vel

          sci=sinchi*cosphi
          ssi=sinchi*sinphi
         vx(j,2)= tempvx*coschi-tempvz*sci
        1           -tempvx*tempvy*ssi
         vy(j,2)= tempvy*coschi+(tempvx*tempvx+tempvz*tempvz)*ssi
         vz(j,2)= tempvz*coschi+tempvx*sci
        1           -tempvy*tempvz*ssi

          vel = vel*coschi
          vx(j,2) = vx(j,2)*vel
          vy(j,2) = vy(j,2)*vel
          vz(j,2) = vz(j,2)*vel

c ************************************/
c       if the collision is charge exchange */
          else
          engy = frengi
         vel = sqrt(abs(engy)/ivscale)

          vx(j,2)= ranf(0.d0)
          vy(j,2)= ranf(0.d0)
          vz(j,2)= ranf(0.d0)
          rtc(j,2)=0.
          vel = vel/sqrt(vx(j,2)*vx(j,2)+vy(j,2)*vy(j,2)
        1           +vz(j,2)*vz(j,2))

          vx(j,2) = vx(j,2)*vel
          vy(j,2) = vy(j,2)*vel
          vz(j,2) = vz(j,2)*vel
          end if
          end if
c
  200    continue
          return
          end
c
c
c
          subroutine newvel(energy,vel,vx,vy,vz)
          common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
        1               dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
        1               pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
          common /randm1/ ranb,rans
          real*8 ranb,rans,ranf
          real nc2p

          tempvx= vx
          tempvy= vy
          tempvz= vz

          if(energy .lt. 1.e-30) print *,'stop newvel',energy
          if(energy .lt. 1.e-30) then
          coschi = 1.
          else
          en=energy+1.
          ra=ranf(0.d0)
          coschi = (energy +2. -2.*en**ra)/energy
          if(coschi.gt.1..or.coschi.lt.-1.) then
          print *,'coschi=',coschi
          coschi=coschi/abs(coschi)
          end if
          end if
          sinchi = sqrt(1. - coschi*coschi)

          ra=ranf(0.d0)
          tra=twopi*ra
          cosphi = cos(tra)
          if(tra.le.pi) then
          sinphi = sqrt(1.-cosphi*cosphi)
          else
          sinphi = -sqrt(1.-cosphi*cosphi)
          end if
c

          sci=sinchi*cosphi
          ssi=sinchi*sinphi
```

```
        vx= tempvx*coschi -tempvz*sci
     1      -tempvx*tempvy*ssi
        vy= tempvy*coschi +(tempvx*tempvx+tempvz*tempvz)*ssi
        vz= tempvz*coschi +tempvx*sci
     1      -tempvy*tempvz*ssi

        vel=vel/sqrt(vx*vx+vy*vy+vz*vz)
        vx = vx*vel
        vy = vy*vel
        vz = vz*vel
c
        return
        end
c
c
c
        subroutine xsect
        common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
        common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1                pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
        common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1                sextmax,extengy0,extengy1,extengy2,
     1                sionmax,ionengy0,ionengy1,ionengy2,
     1                achrgx,bchrgx,ascat,bscat
        common /cs1/ sels(500),sext(500),sion(500)
       real ionengy0,ionengy1,ionengy2,nc2p
c
        dee=0.2
c    elastic cross section */
        do 100 i=1,nemax
        xe=float(i)*dee
        if (xe .le. elsengy0)  then
          sels(i) = 0.
        else if (xe .gt. elsengy0 .and. xe .le. elsengy1) then
       sels(i) = selsmax*(xe - elsengy0)/(elsengy1 - elsengy0)
        else if (xe .gt. elsengy1 .and. xe .le. elsengy2) then
        sels(i) = selsmax
        else
        sels(i) = selsmax*elsengy2/alog(elsengy2)
     1             *alog(xe)/xe
        end if
  100   continue
c
c       /* excitation cross section */
        do 200 i=1,nemax
        xe=float(i)*dee
        if (xe .le. extengy0) then
        sext(i) = 0.
        else if (xe .gt. extengy0 .and.  xe .le. extengy1) then
        sext(i) = sextmax*(xe - extengy0)/(extengy1 - extengy0)
        else if (xe .gt. extengy1 .and. xe .le. extengy2) then
        sext(i) = sextmax
        else
        sext(i) = sextmax*extengy2/alog(extengy2)
     1           *alog(xe)/xe
        end if
  200   continue
c
c       /* ionization cross section */
        do 300 i=1,nemax
        xe=float(i)*dee
        if (xe .le. ionengy0) then
        sion(i) = 0.
        else if (xe .gt. ionengy0 .and. xe .le. ionengy1) then
        sion(i) = sionmax*(xe - ionengy0)/(ionengy1 - ionengy0)
        else if (xe .gt. ionengy1 .and. xe .le. ionengy2) then
        sion(i) = sionmax
        else
        sion(i) = sionmax*ionengy2/alog(ionengy2)*
     1            alog(xe)/xe
        end if
  300   continue
c
        return
        end
c
c
c
        subroutine xgrid
        common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
        common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
```

```
     1               pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
          common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
          common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
          common /abc/ ax(9001),bx(9001),cx(9001),volum(9001)
          common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
          common /gauss/ der(9001),power
          common /main/ nt2,n1,n4,ne
          common /kkk/ keyn,keya,kcom
          real*8 ranb,rans,ranf
          real length,m,ionengy0,ionengy1,ionengy2,ivscale,m1,nc2p
c
          print *,nc,keyn
          if(keyn.eq.2) aa=3.
          if(keyn.eq.3) aa=3.
          nc=nc*aa
          ng=nc+1
          xnc=float(nc)
          xnm=xnc*2./3.
          dx=dx/aa
          se=dx*dx/epsilon
          do 10 isp=1,nsp
          do 10 i=1,np(isp)
          x(i,isp)=x(i,isp)*aa
   10     continue
c
          dddx=dx*dx*dx
          ncm1=nc-1
          do 80 i=1,ncm1
          h1=float(i)+.5
          h2=h1-1.
          h3=h1**3-h2**3
          a1=3.*h1*h1/h3
          b1=3.*h2*h2/h3
          ax(i+1)=-b1
          bx(i+1)=a1+b1
          cx(i+1)=-a1
          volum(i+1)=2.*twopi*h3*dddx/3.
   80     continue
          ax(1)=0.
          bx(1)=3./.5
          cx(1)=-bx(1)
          volum(1)=2.*twopi*(.5)**3*dddx/3.
          volum(ng)=2.*twopi*(float(nc)**3-(float(nc-1)+.5)**3)
     1              *dddx/3.
          do 100 isp=1,nsp
          do 100 i=1,ng
          vq(i,isp)=q(isp)/volum(i)
   100    continue
c
c
          ww=length*2./3.
          delta=ww
          dd=delta*delta
          dr=ww/float(nc*2/3)
          ai0=2.*power/(pi*dd)
          print *,'ai0=',ai0
c
          do 105 i=1,nc
          r=float(i-1)*dr
          yr=2.*r*r/dd
          der(i)=5.e-12*ai0*exp(-yr)
   105    continue
c
          return
          end
c
c
c
          subroutine tstep(nt2)
          common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1    e(9001),rtc(150000,2),rep(150000,2)
          common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                  dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1                  pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
          common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
          common /kkk/ keyn,keya,kcom
          real length,m,ionengy0,ionengy1,ionengy2,ivscale,m1,nc2p
c
          k=10
          if(keyn.ge.2) k=2
          dt=dt/float(k)
          nt1=nt1*k
          nt2=nt2*k
          return
          end
```

```
c
c
c
      function ranf(dummy)
C
C =====================================================================
C
C     PORTABLE PSEUDO-RANDOM NUMBER GENERATOR, FROM TONY WARNOCK,
C     CRAY RESEARCH INSTITUTE.
C     USE DOUBLE PRECISION ON SHORT WORD LENGTH MACHINES.
C     QQ MUST BE GIVEN TO MACHINE SIGNIFICANCE (=5.960464477539E-8).
C     IF 2.**(-24) WON'T COMPILE ON CDC, USE 16704000000000000000B
C
C     ranf IS CALLED BY: inlet, move1 and freetime
C
C     ranf CALLS THE FOLLOWING SUBROUTINES AND FUNCTIONS:  (NONE)
C
C =====================================================================
C
       implicit real*8(a-h,o-z)
      PARAMETER (PP=2.**24, QQ=2.**(-24), GB=1., GS=14390069.)
c     parameter (pp=16777216.d0,qq=5.96046447539d-8,
c    1  gb=1.d0,gs=14390069.d0)
      common /randm1/ ranb,rans
C
C <><><><
```

```
c
      subroutine coul(ss12,denx,fc)
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
c
c     print *,'6'
c
      dena=denx/1.e+06
      temp=ss12*q(2)/(bk*1.5)
      if(temp.lt.4.2e+05) then
      xll=9.42+1.5*alog(temp)-.5*alog(dena)
      else
      xll=15.9+alog(temp)-.5*alog(dena)
      end if
      tc=.38*(sqrt(temp))**3/(dena*xll)
      if(xll.lt.0.25) then
      tc=tc*4.*xll
      end if
      fc=1./tc
c     print *,'temp,xll,dena,tc,fc'
c     print *,temp,xll,dena,tc,fc
      return
      end
c
c
c
      subroutine colfr(engy,denx,fm1)
c
      if(engy.gt.1.) then
      if(denx.lt.3.e+23) then
      fm1=1.34e+13
      else if(denx.ge.3.e+23.and.denx.lt.4.e+23) then
      fm1=1.35e+13
      else if(denx.ge.4.e+23.and.denx.lt.5.e+23) then
      fm1=1.36e+13
      else if(denx.ge.5.e+23.and.denx.lt.6.e+23) then
      fm1=1.37e+13
      else if(denx.ge.6.e+23.and.denx.lt.7.e+23) then
      fm1=1.39e+13
      else if(denx.ge.7.e+23.and.denx.lt.8.e+23) then
      fm1=1.40e+13
      else if(denx.ge.8.e+23.and.denx.lt.9.e+23) then
      fm1=1.41e+13
      else if(denx.ge.9.e+23.and.denx.lt.1.e+24) then
      fm1=1.42e+13
      else if(denx.ge.1.e+24.and.denx.lt.2.e+24) then
      fm1=1.81e+13
      else if(denx.ge.2.e+24.and.denx.lt.3.e+24) then
      fm1=2.22e+13
      else if(denx.ge.3.e+24.and.denx.lt.4.e+24) then
      fm1=2.56e+13
      else if(denx.ge.4.e+24.and.denx.lt.5.e+24) then
      fm1=2.86e+13
      else if(denx.ge.5.e+24.and.denx.lt.6.e+24) then
      fm1=3.14e+13
      else if(denx.ge.6.e+24.and.denx.lt.7.e+24) then
      fm1=3.39e+13
      else if(denx.ge.7.e+24.and.denx.lt.8.e+24) then
      fm1=3.63e+13
      else if(denx.ge.8.e+24.and.denx.lt.9.e+24) then
      fm1=3.85e+13
      else if(denx.ge.9.e+24.and.denx.lt.1.e+25) then
      fm1=4.05e+13
      else
      fm1=5.74e+13
      end if
      else
      if(denx.lt.3.e+23) then
      fm1=3.63e+13
      else if(denx.ge.3.e+23.and.denx.lt.4.e+23) then
      fm1=4.84e+13
      else if(denx.ge.4.e+23.and.denx.lt.5.e+23) then
      fm1=6.05e+13
      else if(denx.ge.5.e+23.and.denx.lt.6.e+23) then
      fm1=7.26e+13
      else if(denx.ge.6.e+23.and.denx.lt.7.e+23) then
      fm1=8.47e+13
      else if(denx.ge.7.e+23.and.denx.lt.8.e+23) then
      fm1=9.68e+13
      else if(denx.ge.8.e+23.and.denx.lt.9.e+23) then
      fm1=1.09e+14
      else if(denx.ge.9.e+23.and.denx.lt.1.e+24) then
      fm1=1.21e+14
      else if(denx.ge.1.e+24.and.denx.lt.2.e+24) then
```

```
        fm1=2.42e+14
        else if(denx.ge.2.e+24.and.denx.lt.3.e+24) then
        fm1=3.63e+14
        else if(denx.ge.3.e+24.and.denx.lt.4.e+24) then
        fm1=4.84e+14
        else if(denx.ge.4.e+24.and.denx.lt.5.e+24) then
        fm1=6.05e+14
        else if(denx.ge.5.e+24.and.denx.lt.6.e+24) then
        fm1=7.26e+14
        else if(denx.ge.6.e+24.and.denx.lt.7.e+24) then
        fm1=8.47e+14
        else if(denx.ge.7.e+24.and.denx.lt.8.e+24) then
        fm1=9.68e+14
        else if(denx.ge.8.e+24.and.denx.lt.9.e+24) then
        fm1=1.09e+15
        else if(denx.ge.9.e+24.and.denx.lt.1.e+25) then
        fm1=1.21e+15
        else
        fm1=2.42e+15
        end if
        end if
        return
        end
c
c
c
        subroutine reduce(ne)
        common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
        common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1                pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
        common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
        common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
        common /abc/ ax(9001),bx(9001),cx(9001),volum(9001)
        common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
        parameter (mm=7,nstack=50,fm=7875.,fa=211.,fc=1663.,fmi=1./fm)
        parameter (aln2i=1./0.69314718, tiny=1.e-5)
        dimension istack(nstack)
        dimension ek(9001),scale(4),nkc(200),npc(9000),x1(9000)
        dimension xc(9000,200),vxc(9000,200),vyc(9000,200),vzc(9000,200)
        dimension repc(9000,200)
        real length,m,ivscale,m1,nc2p
c
        scale(1)=evscale
        scale(2)=ivscale
c
c       print *,'np',np
902     format(6e12.5)
c
        do 100 isp=1,nsp
c
        do 10 i=1,nc
  10    npc(i)=0
c
        do 20 j=1,np(isp)
        k=int(x(j,isp))+1
        npc(k)=npc(k)+1
        npk=npc(k)
        xc(k,npk)=x(j,isp)
        vxc(k,npk)=vx(j,isp)
        vyc(k,npk)=vy(j,isp)
        vzc(k,npk)=vz(j,isp)
        repc(k,npk)=rep(j,isp)
  20    continue
c
        print *,'npc'
        write(6,904) (npc(i),i=1,10)
 904    format(10i8)
c
        inc=0
        do 30 i=1,nc
        if(npc(i).gt.ne) then
          do 31 j=1,npc(i)
          dum=vxc(i,j)*vxc(i,j)+vyc(i,j)*vyc(i,j)+vzc(i,j)*vzc(i,j)
          ek(j)=scale(isp)*dum
  31      continue
c
ccccccccc  sorting ek(i)
c
            lognb2=int(alog(float(npc(i)))*aln2i+tiny)
            m2=npc(i)
            do 7 nn1=1,lognb2
              m2=m2/2
              km=npc(i)-m2
```

```
              do 6 j=1,km
                 i1=j
  3              continue
                 l=i1+m2
                 if(ek(l).lt.ek(i1)) then
                   t=ek(i1)
                   t1=vxc(i,i1)
                   t2=vyc(i,i1)
                   t3=vzc(i,i1)
                   t4=repc(i,i1)
                   t5=xc(i,i1)
                   ek(i1)=ek(l)
                   vxc(i,i1)=vxc(i,l)
                   vyc(i,i1)=vyc(i,l)
                   vzc(i,i1)=vzc(i,l)
                   repc(i,i1)=repc(i,l)
                   xc(i,i1)=xc(i,l)
                   ek(l)=t
                   vxc(i,l)=t1
                   vyc(i,l)=t2
                   vzc(i,l)=t3
                   repc(i,l)=t4
                   xc(i,l)=t5
                   i1=i1-m2
                   if(i1.ge.1) goto  3
                 end if
   6           continue
   7       continue
c
           nk=int(npc(i)/ne)
           nkp=nk+1
           nem=npc(i)-nk*ne
           do 32 l=1,nem
  32       nkc(l)=nkp
           do 33 l=nem+1,ne
  33       nkc(l)=nk
           lnk=0
                do 34 l=1,ne
                sek=0.
                svx=0.
                svy=0.
                svz=0.
                srp=0.
                sx=0.
                do 35 j=1,nkc(l)
                   srp=srp+repc(i,j+lnk)
                   sek=sek+ek(j+lnk)*repc(i,j+lnk)
                   svx=svx+vxc(i,j+lnk)*repc(i,j+lnk)
                   svy=svy+vyc(i,j+lnk)*repc(i,j+lnk)
                   svz=svz+vzc(i,j+lnk)*repc(i,j+lnk)
                  sx=sx+xc(i,j+lnk)*repc(i,j+lnk)
 35             continue
                lnk=lnk+nkc(l)
                sek=sek/srp
                svx=svx/srp
                svy=svy/srp
                svz=svz/srp
                sx=sx/srp
                dum=svx*svx+svy*svy+svz*svz
                fac=sek/(scale(isp)*dum)
                fac=sqrt(fac)
                vx(inc+1,isp)=fac*svx
                vy(inc+1,isp)=fac*svy
                vz(inc+1,isp)=fac*svz
                x(inc+1,isp)=sx
                rep(inc+1,isp)=srp
  34          continue
              inc=inc+ne
         else
             do 36 j=1,npc(i)
             x(inc+j,isp)=xc(i,j)
             vx(inc+j,isp)=vxc(i,j)
             rep(inc+j,isp)=repc(i,j)
  36         continue
                  inc=inc+npc(i)
         end if
  30         continue
           np(isp)=inc
  100    continue
c
c
c        print *,'x,vx,vy,vz'
c        write(6,902) (x(j,1),j=1,10)
c        write(6,902) (x(j,1),j=np(1)-10,np(1))
c        write(6,902) (vx(j,1),j=1,10)
```

```
c       write(6,902) (vy(j,1),j=1,10)
c       write(6,902) (vz(j,1),j=1,10)
        do i=1,20
        dum=vx(i,1)*vx(i,1)+vy(i,1)*vy(i,1)+vz(i,1)*vz(i,1)
        x1(i)=scale(1)*dum
        end do
c       write(6,902) (x1(i),i=1,20)
c
        do 500 isp=1,nsp
        do 550 j=1,90000
  550   rtc(j,isp)=0.
  500   continue
c
c       print *,'np',np
        print *,'rep'
        write(6,902) (rep(i,1),i=1,10)
c       print *,'5'
        return
        end
c
c
c
        subroutine output
        common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
        common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1                pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
        common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1                sextmax,extengy0,extengy1,extengy2,
     1                sionmax,ionengy0,ionengy1,ionengy2,
     1                achrgx,bchrgx,ascat,bscat
        common /cs1/ sels(500),sext(500),sion(500)
        common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
        common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
        common /ff/ fm,fm1,fmi,fm2
        common /randm1/ ranb,rans
        common /temp/ nt,sab,ncent,omiga1
        common /abc/ ax(9001),bx(9001),cx(9001),volum(9001)
        common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
        common /main/ nt2,n1,n4,ne
        common /kkk/ keyn,keya,kcom
        common /datac/ temp(4),yl(4),bea(3),beb(3),evk
        common /recom1/ dent,betai,arol,gdent
        common /res/ nend2,dt1
        common /gauss/ der(9001),power
        real*8 ranb,rans,ranf
        real length,m,ionengy0,ionengy1,ionengy2,ivscale,m1,nc2p
c
        open(unit=2,file='pcsr.rst',form='unformatted')
        rewind 2
c
        write(2) nsp,ng
c
        do 10 isp=1,nsp
        write(2) np(isp)
        write(2) (x(i,isp),i=1,np(isp))
        write(2) (vx(i,isp),i=1,np(isp))
        write(2) (vy(i,isp),i=1,np(isp))
        write(2) (vz(i,isp),i=1,np(isp))
        write(2) (rep(i,isp),i=1,np(isp))
        write(2) (rtc(i,isp),i=1,np(isp))
   10   continue
c
        write(2) (ax(i),i=1,ng)
        write(2) (bx(i),i=1,ng)
        write(2) (cx(i),i=1,ng)
        write(2) (volum(i),i=1,ng)
        write(2) (der(i),i=1,ng)
        write(2) (e(i),i=1,ng)
        write(2) (sde(i),i=1,ng)
        write(2) (sdev(i),i=1,ng)
        write(2) (eqm(i,1),i=1,ng)
        write(2) (eqm(i,2),i=1,ng)
        write(2) (dens(i,1),i=1,ng)
        write(2) (dens(i,2),i=1,ng)
        write(2) (vq(i,1),i=1,ng)
        write(2) (vq(i,2),i=1,ng)
        write(2) (sels(i),i=1,500)
        write(2) (sext(i),i=1,500)
        write(2) (sion(i),i=1,500)
c
        write(2)      nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                dt,de,bk,gtemp,gden,nt1,nn,q,m,
     1                pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
```

```
      write(2)     selsmax,elsengy0,elsengy1,elsengy2,
     1              sextmax,extengy0,extengy1,extengy2,
     1              sionmax,ionengy0,ionengy1,ionengy2,
     1              achrgx,bchrgx,ascat,bscat
      write(2)     dx,xnc,evscale,ivscale,q1,m1,xnm,
     1              qm,se,sm,frengi,power,
     1              fm,fm1,fmi,fm2,cm2,cm2c,sab,ranb,rans,
     1              nt2,n1,n4,ne
      write(2)     keyn,keya,kcom
      write(2)     bea,beb,temp,yl,nend2,dt1,
     1              dent,betai,evk,aro1,gdent
      write(2) (x(i,1),i=1,np(1))
      close(2)
c
      return
      end
c
c
c
      subroutine restal
      common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1 e(9001),rtc(150000,2),rep(150000,2)
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1              sextmax,extengy0,extengy1,extengy2,
     1              sionmax,ionengy0,ionengy1,ionengy2,
     1              achrgx,bchrgx,ascat,bscat
      common /cs1/ sels(500),sext(500),sion(500)
      common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
      common /const/ qm(2),vn(9001),vq(9001,2),eqm(9001,2),se,sm,frengi
      common /ff/ fm,fm1,fmi,fm2
      common /randm1/ ranb,rans
      common /temp/ nt,sab,ncent,omiga1
      common /abc/ ax(9001),bx(9001),cx(9001),volum(9001)
      common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
      common /main/ nt2,n1,n4,ne
      common /kkk/ keyn,keya,kcom
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
     common /recom1/ dent,betai,aro1,gdent
      common /res/ nend2,dt1
      common /gauss/ der(9001),power
      real*8 ranb,rans,ranf
      real length,m,ionengy0,ionengy1,ionengy2,ivscale,m1,nc2p
c
      open(unit=3,file='pcsr.rst',form='unformatted')
      read(3) nsp,ng
      do 10 isp=1,nsp
      read(3) np(isp)
      read(3) (x(i,isp),i=1,np(isp))
      read(3) (vx(i,isp),i=1,np(isp))
      read(3) (vy(i,isp),i=1,np(isp))
      read(3) (vz(i,isp),i=1,np(isp))
      read(3) (rep(i,isp),i=1,np(isp))
      read(3) (rtc(i,isp),i=1,np(isp))
  10  continue
c
      read(3) (ax(i),i=1,ng)
      read(3) (bx(i),i=1,ng)
      read(3) (cx(i),i=1,ng)
      read(3) (volum(i),i=1,ng)
      read(3) (der(i),i=1,ng)
      read(3) (e(i),i=1,ng)
      read(3) (sde(i),i=1,ng)
      read(3) (sdev(i),i=1,ng)
      read(3) (eqm(i,1),i=1,ng)
      read(3) (eqm(i,2),i=1,ng)
      read(3) (dens(i,1),i=1,ng)
      read(3) (dens(i,2),i=1,ng)
      read(3) (vq(i,1),i=1,ng)
      read(3) (vq(i,2),i=1,ng)
      read(3) (sels(i),i=1,500)
      read(3) (sext(i),i=1,500)
      read(3) (sion(i),i=1,500)

      read(3)     nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q,m,
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      read(3)     selsmax,elsengy0,elsengy1,elsengy2,
     1              sextmax,extengy0,extengy1,extengy2,
     1              sionmax,ionengy0,ionengy1,ionengy2,
     1              achrgx,bchrgx,ascat,bscat
      read(3)     dx,xnc,evscale,ivscale,q1,m1,xnm,
     1              qm,se,sm,frengi,power,
```

```
     1                fm,fm1,fmi,fm2,cm2,cm2c,sab,ranb,rans,
     1                nt2,n1,n4,ne
          read(3)     keyn,keya,kcom
          read(3)     bea,beb,temp,yl,nend2,dt1,
     1                dent,betai,evk,aro1,gdent
c
          return
          end
c
c
c
          subroutine check1(ss12)
          common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1   e(9001),rtc(150000,2),rep(150000,2)
          common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1                dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1                pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
          common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
          common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
c
          ss1=0.
          ss2=0.
c
          do 200 j=1,np(1)
          dum=vx(j,1)*vx(j,1)+vy(j,1)*vy(j,1)+vz(j,1)*vz(j,1)
          engy=evscale*dum
          ss1=ss1+engy*rep(j,1)
          ss2=ss2+rep(j,1)
   200    continue
          ss12=ss1/ss2
c         print *,'ss12=',ss12
c
          return
          end
c
c
c
          subroutine reca(sn1,ss12)
c         This program is used to calculate electron density
c         and average electron energy under recombination
c         and ionization.
c
          common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1    e(9001),rtc(150000,2),rep(150000,2)
          common /datac/ temp(4),yl(4),bea(3),beb(3),evk
          common /recom/ dent,e0,beta
          common /res/ nend2,dt1
c
          e0=ss12
          e1=ss12
          dent=sn1*1.e+13/2.12
          d1=dent
c
          do 100 j=2,nend2
          call newne2(j)
          call newengy2(j)
   100    continue
c
          fac1=dent/d1
          fac2=sqrt(e0/e1)
c         print *,'dent,d1,e0,e1,fac1,fac2'
c         print *,dent,d1,e0,e1,fac1,fac2
          print *,'electron density (before,after recom.) (m-3)'
          print *,d1,dent
          print *, 'electron energy (before,after recom.) (eV)'
          print *,e1,e0
c
          nsp=2
          do 110 isp=1,nsp
          do 5 j=1,np(isp)
          rep(j,isp)=rep(j,isp)*fac1
     5    rtc(j,isp)=0.
   110    continue
c
          do 200 j=1,np(1)
          vx(j,1)=vx(j,1)*fac2
          vy(j,1)=vy(j,1)*fac2
          vz(j,1)=vz(j,1)*fac2
   200    continue
c
          return
          end
c
c
c
```

```
      subroutine newne2(j)
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      common /recom/ dent,e0,beta
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /res/ nend2,dt1
c
      TT=e0*evk
      alpha=5.6e-21*dent/TT**4.5
      call icoef2(TT)
      d1=-alpha*dent*dent+beta*dent*gden
      dent=dent+d1*dt1
c     if(j.le.2) print *,alpha,beta,gden,d1,dt1
c
      return
      end
c
c
c
      subroutine newengy2(j)
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      common /recom/ dent,e0,beta
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1              sextmax,extengy0,extengy1,extengy2,
     1              sionmax,ionengy0,ionengy1,ionengy2,
     1              achrgx,bchrgx,ascat,bscat
      common /cs1/ sels(500),sext(500),sion(500)
      common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
      common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
      common /res/ nend2,dt1
      real m,ionengy0,nu1,nu2,nu3,nu4,nu5
c
      if(e0.lt.2.8e-02) goto 10
      vel=sqrt(e0/evscale)
      ien=int(e0*5.+.5)
      if(ien.lt.1) ien=1
      nu1=gden*vel*sels(ien)
      nu3=gden*beta
      ef=e0-dt1*(nu1*cm2*e0+nu3*ionengy0)
      e0=ef
c     if(j.le.2) print *,'nu1,nu2,nu3,nu4,nu5',nu1,nu2,nu3,nu4,nu5
 10   continue
c
      return
      end
c
c
c
      subroutine icoef2(TT)
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      common /recom/ dent,e0,beta
      common /res/ nend2,dt1
c
      if(TT.lt.temp(2)) beta=bea(1)*TT+beb(1)
      if(TT.gt.temp(2).and.TT.le.temp(3)) beta=bea(2)*TT+beb(2)
      if(TT.gt.temp(3)) beta=bea(3)*TT+beb(3)
      beta=10.**beta*1.e-12
c
      return
      end
c
c
c
      subroutine recomb(sn1,ss12,ntt)
      common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1 e(9001),rtc(150000,2),rep(150000,2)
      common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      common /res/ nend2,dt1
c
      dent=sn1*1.e+13/2.12
      TT=ss12*evk
      alpha=5.6e-21*dent/TT**4.5
      d1=-alpha*dent*dent
      dent1=dent+d1*dt
      ratio=dent1/dent
c
      do 100 isp=1,nsp
      do 100 j=1,np(isp)
```

```
        rep(j,isp)=rep(j,isp)*ratio
  100   continue
c
        if(mod(ntt,100).eq.0) print *,'ratio=',ratio
        return
        end
c
c
        subroutine check2(n1)
        common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
        common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
        common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
        dimension edis(500)
c
 10     format(6e12.5)
c
        do 250 i=1,500
  250   edis(i)=0.
c
        do 200 j=1,np(1)
        dum=vx(j,1)*vx(j,1)+vy(j,1)*vy(j,1)+vz(j,1)*vz(j,1)
        engy=evscale*dum
        ien=ifix(engy*5.+.5)
        if(ien.lt.1) ien=1
        if(ien.gt.500) ien=500
        edis(ien)=edis(ien)+rep(j,1)
  200   continue
c       write(98,90) n1
c       write(98,10) (edis(i),i=1,500)
  90    format(7h pulse ,i5)
        return
        end
c
c
c
        subroutine connect(sn1,ss12)
        common np(2),x(150000,2),vx(150000,2),vy(150000,2),vz(150000,2),
     1  e(9001),rtc(150000,2),rep(150000,2)
        common /para/ nsp,nc,nc2p,ng,length,area,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
        common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
     1              sextmax,extengy0,extengy1,extengy2,
     1              sionmax,ionengy0,ionengy1,ionengy2,
     1              achrgx,bchrgx,ascat,bscat
        common /ff/ fm,fm1,fmi,fm2
        common /exp/ e0,ef,ssde,sdevs(501),a0(501),npr
c
        e1=ss12
        print *,'ss12,sn1,ssde',ss12,sn1,ssde
        call expa(sn1,ss12,rs)
        fac=sqrt(e0/e1)
c
        gden=gden*rs
        achrgx=achrgx*rs
        bchrgx=bchrgx*rs
        ascat=ascat*rs
        bscat=bscat*rs
        fm=fm*rs
        fm1=fm1*rs
        fmi=fmi*rs
        fm2=fm2*rs
c
902     format(6e12.5)
c
        do 100 isp=1,nsp
        do 5 j=1,np(isp)
        rep(j,isp)=rep(j,isp)*rs
   5    rtc(j,isp)=0.
  100   continue
c
        do 200 j=1,np(1)
        vx(j,1)=vx(j,1)*fac
        vy(j,1)=vy(j,1)*fac
        vz(j,1)=vz(j,1)*fac
  200   continue
c
        print *,'gden,fm,achrgx,e0',gden,fm,fm1,fmi,fm2,achrgx,e0
c
        return
        end
c
```

```
c
c
      subroutine expa(sn1,ss12,rs)
      COMMON /VECTOR/ DQ(501,3),RHOA(501),RHOUA(501),EA(501),
     >                U(501),P(501),EI(501),C(501),RM(501),
     >                DTC(501),source(501)
      COMMON /SPECS/  AREA(501),DADX(501),RL,DX,DDX,AIN,ATH
      COMMON /CONST1/  CFL,GAMMA,GM1,CP,CV,GGM1,U0,T0,P0,
     >                OMEGAX,THETA,PB,PHI,BETA,EPS
      COMMON /INTEGR/  IL,ILM1,NEND,NPRINT,MON,NADV,IBIN,IBEX,IBC
     >                 ,NORD
      common /para/ nsp,nc,nc2p,ng,length,area1,epsilon,s_ext,
     1              dt,de,bk,gtemp,gden,nt1,nn,q(2),m(2),
     1              pi,twopi,maxlen,nsmax,ncmax,nemax,DBL_MIN
      common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
      common /exp/ e0,ef,ssde,sdevs(501),a0(501),npr
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      common /recom1/ dent,betai,aro1,gdent
      common /kkk/ keyn,keya,kcom
      dimension  aro(501),aron(501),ratio(501)
      real m
c
      if(keya.eq.1) then
      call roe4(1,1)
      keya=2
      else
      call roe4(3,1)
      end if
c
      do 50 i=1,il
      a0(i)=rhoa(i)
      aro(i)=rhoa(i)/area(i)
      sdevs(i)=0.
 50   continue
      write(6,11) (aro(i),i=1,6)
c
      ilm2=ilm1*.5
      do 60 i=1,ilm2
      k=(i-1)*kcom
      do 61 kk=1,kcom
 61   sdevs(i)=sdevs(i)+sdev(k+kk)
      sdevs(i)=sdevs(i)/kcom
 60   continue
      do 65 i=1,8
 65   sdevs(i)=sdevs(9)
      write(6,11) (sdevs(i),i=1,16)
      write(6,11) (sdevs(i),i=il-7,il)
c
      do 100 i=1,il
      ea(i)=ea(i)+1.6e-19*sdevs(i)*area(i)
      ei(i)=ea(i)/rhoa(i)-u(i)*u(i)*.5
      p(i)=gm1*rhoa(i)/area(i)*ei(i)
      c(i)=sqrt(gamma/rhoa(i)*area(i)*p(i))
      rm(i)=u(i)/c(i)
 100  continue
      write(6,11) (ea(i),i=1,6)
      write(6,11) (ei(i),i=1,6)
      write(6,11) (c(i),i=1,6)
      write(6,11) (p(i),i=1,6)
c
      dent=sn1*1.e+13/2.12
      gdent=gden
      e0=ss12
      nend=46.e-09/dtc(1)
      npr=1.e-08/dtc(1)
      call roe4(2,1)
c
      rs=0.
      do 300 i=1,il
      aron(i)=rhoa(i)/area(i)
      ratio(i)=aron(i)/aro(i)
      aro(i)=aron(i)
 300  continue
      do 301 i=1,150
 301  rs=rs+ratio(i)
      rs=rs/150.
c
      print *, 'rs',rs
      write(10,12) rs
      write(10,11) (ratio(i),i=1,il)
c     print *,'p'
c     write(6,11) (p(i),i=1,il)
 11   format(6e12.5)
 12   format(7h ratio ,1e13.6)
c
```

```
      return
      end
c
c
c
      subroutine deft
      COMMON /VECTOR/ DQ(501,3),RHOA(501),RHOUA(501),EA(501),
     >                U(501),P(501),EI(501),C(501),RM(501),
     >                DTC(501),source(501)
      COMMON /SPECS/  AREA(501),DADX(501),RL,DX,DDX,AIN,ATH
      COMMON /INTEGR/  IL,ILM1,NEND,NPRINT,MON,NADV,IBIN,IBEX,IBC
     >                 ,NORD
      common /exp/ e0,ef,ssde,sdevs(501),a0(501),npr
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      dimension aro(501)
c
      if(nadv.eq.1) aro1=1.
      if(nadv.gt.1) then
      do 110 i=1,il
  110 aro(i)=rhoa(i)/a0(i)
c     if(mod(nadv,npr).eq.0) write(6,11) (aro(i),i=1,il)
c     dent=dent*aro(100)/aro1
c     aro1=aro(100)
      ar2=0.
      do 111 i=1,150
  111 ar2=ar2+aro(i)
      ar2=ar2/150.
      dent=dent*ar2/aro1
      aro1=ar2
c
      call newne(ar2)
      if(e0.lt.2.8e-02) then
      return
      else
      call newengy(ar2)
      end if
c
      fac=(e0-ef)*dent*1.e-13
      fac1=fac*1.6e-19/ssde
      fac2=fac2+fac*1.6e-19
c     write(50,*) nadv,e0,fac2
c
      do 100 i=1,il
      source(i)=fac1*sdevs(i)*aro(i)*area(i)/dtc(1)
  100 continue
c
      e0=ef
   11 format(6e12.5)
c
      end if
      return
      end
c
c
c
      subroutine newne(ar2)
      COMMON /VECTOR/ DQ(501,3),RHOA(501),RHOUA(501),EA(501),
     >                U(501),P(501),EI(501),C(501),RM(501),
     >                DTC(501),source(501)
      common /exp/ e0,ef,ssde,sdevs(501),a0(501),npr
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      common /recom1/ dent,betai,aro1,gdent
      dimension aro(501)
c
c
      TT=e0*evk
      dden=dent
      alpha=5.6e-21*dden/TT**4.5
      call icoef(TT)
      d1=-alpha*dden*dden+betai*dden*gdent*ar2
      dent=dden+d1*dtc(1)
c
      return
      end
c
c
c
      subroutine newengy(ar2)
      COMMON /VECTOR/ DQ(501,3),RHOA(501),RHOUA(501),EA(501),
     >                U(501),P(501),EI(501),C(501),RM(501),
     >                DTC(501),source(501)
      common /exp/ e0,ef,ssde,sdevs(501),a0(501),npr
      common /datac/ temp(4),yl(4),bea(3),beb(3),evk
      common /recom1/ dent,betai,aro1,gdent
      common /cros/ selsmax,elsengy0,elsengy1,elsengy2,
```

```
     1               sextmax,extengy0,extengy1,extengy2,
     1               sionmax,ionengy0,ionengy1,ionengy2,
     1               achrgx,bchrgx,ascat,bscat
          common /cs1/ sels(500),sext(500),sion(500)
          common /chp/ dx,xnc,evscale,ivscale,q1(2),m1(2),xnm
          common /ds/ dens(9001,2),sde(9001),sdev(9001),cm2,cm2c
          dimension aro(501)
          real m,ionengy0,nu1,nu2,nu3
c
c
          vel=sqrt(e0/evscale)
          ien=int(e0*5.+.5)
          if(ien.lt.1) ien=1
          gden1=gdent*ar2
          nu1=gden1*vel*sels(ien)
          nu3=gden1*betai
          ef=e0-dtc(1)*(nu1*cm2*e0+nu3*ionengy0)
c
          return
          end
c
c
c
          subroutine icoef(TT)
          common /datac/ temp(4),yl(4),bea(3),beb(3),evk
          common /recom1/ dent,betai,aro1,gdent
c
          if(TT.lt.temp(2)) betai=bea(1)*TT+beb(1)
          if(TT.gt.temp(2).and.TT.le.temp(3)) betai=bea(2)*TT+beb(2)
          if(TT.gt.temp(3)) betai=bea(3)*TT+beb(3)
          betai=10.**betai*1.e-12
c
          return
          end
```

APPENDIX B

INPUT AND OUTPUT DATA

Input Data

```
50  200  2  10   5  10
2   1000   1    5.55e-05    2.52e-07     8.85e-12
2.e-13  1.0  1.38e-23  300  2.45e+25  -1.6e-19  1.6e-19  9.1e-31  6.68e-26
3.14159   6.28319     20000   4   201    500   1.e-300
2.0e-19    0.3     12.0    15.0
7.0e-21    11.55     30.0      100.0
3.0e-20    15.76     30.0      100.0
2.0e-19    5.5e-19    1.8e-19    4.0e-19
1.29e+13   1.29e+13   5.07e+11   2.07e-14
5.175e+02
46000   1.16e+04
8000.   16000.  32000.  64000.
-22.52  -18.   -16.   -14.699
```

Output Data

```
 pulse number          1

    time(ps)        computational particles    real particles

   1.000000000              1          1   1.000000000
   2.000000238              1          1   1.000000000
   3.000000477              1          1   1.000000000
   4.000000477              1          1   1.000000000
   4.999999523              1          1   1.000000000
   5.999998569              1          1   1.000000000
   6.999997616              1          1   1.000000000
   7.999996662              2          2   2.000000000
   8.999996185              3          3   3.000000000
   9.999995232              3          3   3.000000000
   10.99999428              3          3   3.000000000
   11.99999332              3          3   3.000000000
   12.99999237              3          3   3.000000000
   13.99999142              3          3   3.000000000
   14.99999046              3          3   3.000000000
   15.99998951              3          3   3.000000000
   16.99999237              3          3   3.000000000
   17.99999619              3          3   3.000000000
   19.00000000              3          3   3.000000000
   20.00000381              3          3   3.000000000
   21.00000763              3          3   3.000000000
   22.00001144              3          3   3.000000000
   23.00001526              3          3   3.000000000
   24.00001907              3          3   3.000000000
   25.00002289              3          3   3.000000000
   26.00002670              3          3   3.000000000
   27.00003052              3          3   3.000000000
   28.00003433              3          3   3.000000000
   29.00003815              3          3   3.000000000
   30.00004196              3          3   3.000000000
   31.00004578              3          3   3.000000000
   32.00004959              3          3   3.000000000
   33.00005341              3          3   3.000000000
   34.00005722              3          3   3.000000000
   35.00006104              3          3   3.000000000
   36.00006485              3          3   3.000000000
   37.00006866              3          3   3.000000000
   38.00007248              3          3   3.000000000
   39.00007629              3          3   3.000000000
   40.00008011              3          3   3.000000000
```

average electron energy (eV) 4.795877934
electron density (before,after recom.) (m-3)
 0.1415094416E+14 0.1415094416E+14
electron energy (before,after recom.) (eV)
  4.795877934   1.589057565

pulse number 2

time(ps)	computational particles		real particles
1.000000000	3	3	3.000000000
2.000000238	3	3	3.000000000
3.000000477	4	4	4.000000000
4.000000477	5	5	5.000000000
4.999999523	5	5	5.000000000
5.999998569	6	6	6.000000000
6.999997616	6	6	6.000000000
7.999996662	7	7	7.000000000
8.999996185	8	8	8.000000000
9.999995232	9	9	9.000000000
10.99999428	10	10	10.00000000
11.99999332	10	10	10.00000000
12.99999237	10	10	10.00000000
13.99999142	10	10	10.00000000
14.99999046	11	11	11.00000000
15.99998951	11	11	11.00000000
16.99999237	11	11	11.00000000
17.99999619	11	11	11.00000000
19.00000000	11	11	11.00000000
20.00000381	11	11	11.00000000
21.00000763	11	11	11.00000000
22.00001144	11	11	11.00000000
23.00001526	11	11	11.00000000
24.00001907	11	11	11.00000000
25.00002289	11	11	11.00000000
26.00002670	11	11	11.00000000
27.00003052	11	11	11.00000000
28.00003433	11	11	11.00000000
29.00003815	11	11	11.00000000
30.00004196	11	11	11.00000000
31.00004578	11	11	11.00000000
32.00004959	11	11	11.00000000
33.00005341	11	11	11.00000000
34.00005722	11	11	11.00000000
35.00006104	11	11	11.00000000
36.00006485	11	11	11.00000000
37.00006866	11	11	11.00000000
38.00007248	11	11	11.00000000
39.00007629	11	11	11.00000000
40.00008011	11	11	11.00000000

average electron energy (eV) 2.103952646
electron density (before,after recom.) (m-3)
 0.5188679526E+14 0.5188679526E+14
electron energy (before,after recom.) (eV)
  2.103952646   1.264349937

VITA

Quan Zhang was born in Beijing, China on October 27, 1958. She attended elementary and high schools in Beijing, China. In July 1982 she graduated from The Tsinghua University, Beijing, China, where she received a Bachelor of Science degree in Nuclear Engineering. In August 1982, she entered the graduate school of the Tsinghua University ,China, where she received the degree of Master of Science in Nuclear Engineering in 1984. From 1984 to 1986 she hold a research position in the Engineering Physics department at the Tsinghua University.

She came to The University of Tennessee Space Institute as a graduate research assistant in January 1987. This dissertation work was carried out in pursuit of a Doctor of Philosophy degree in Engineering Science and Mechanics, and was completed in April 1992.

She was married to Xiaohang Peng in August, 1985, and has a four year old daughter, Barbara Mong-Mong, and a one year old son, Aaron Yuan.