

1995

## A Delta-f Monte Carlo method to calculate parameters in plasmas

Maciek Sasinowski

*College of William & Mary - Arts & Sciences*

Follow this and additional works at: <https://scholarworks.wm.edu/etd>



Part of the [Computer Sciences Commons](#), and the [Plasma and Beam Physics Commons](#)

---

### Recommended Citation

Sasinowski, Maciek, "A Delta-f Monte Carlo method to calculate parameters in plasmas" (1995).

*Dissertations, Theses, and Masters Projects*. Paper 1539623873.

<https://dx.doi.org/doi:10.21220/s2-j9t0-g494>

This Dissertation is brought to you for free and open access by the Theses, Dissertations, & Master Projects at W&M ScholarWorks. It has been accepted for inclusion in Dissertations, Theses, and Masters Projects by an authorized administrator of W&M ScholarWorks. For more information, please contact [scholarworks@wm.edu](mailto:scholarworks@wm.edu).

## INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

**The quality of this reproduction is dependent upon the quality of the copy submitted.** Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

# UMI

A Bell & Howell Information Company  
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA  
313/761-4700 800/521-0600



A  $\delta f$  Monte Carlo Method to Calculate Parameters in Plasmas

---

A Dissertation

Presented to

The Faculty of the Physics Department

The College of William and Mary in Virginia

in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy

---

by

Maciek Sasinowski

May 12, 1995

**UMI Number: 9605790**

---

**UMI Microform 9605790**

**Copyright 1995, by UMI Company. All rights reserved.**

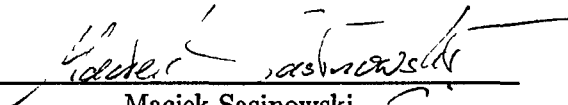
**This microform edition is protected against unauthorized  
copying under Title 17, United States Code.**

---


**UMI**

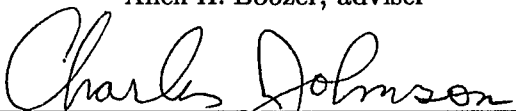
**300 North Zeeb Road  
Ann Arbor, MI 48103**

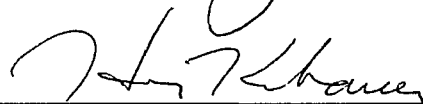
This dissertation is submitted in partial fulfillment of  
the requirements for the degree of  
Doctor of Philosophy

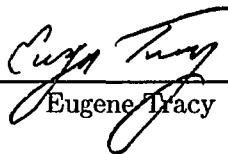
  
\_\_\_\_\_  
Maciek Sasinowski


Approved, May 12, 1995

  
\_\_\_\_\_  
Allen H. Boozer, adviser

  
\_\_\_\_\_  
Charles Johnson

  
\_\_\_\_\_  
Henry Krakauer

  
\_\_\_\_\_  
Eugene Tracy

  
\_\_\_\_\_  
George Vahala

# Table of Contents

Acknowledgements . . . . .	v
List of Tables . . . . .	vii
List of Figures . . . . .	viii
Abstract . . . . .	xiii
1 Introduction . . . . .	2
1.1 Motivation . . . . .	2
1.2 What is a Plasma? . . . . .	3
1.3 Thermonuclear Fusion . . . . .	5
1.3.1 Plasma Heating . . . . .	7
1.3.2 Plasma Confinement . . . . .	8
1.4 Numerical Simulations of Plasmas . . . . .	12
2 Basic Plasma Physics . . . . .	16
2.1 Individual particle motion . . . . .	16
2.1.1 Gyromotion, Magnetic Moment and Pitch . . . . .	16
2.1.2 Particle Drifts . . . . .	21
2.2 General Coordinates . . . . .	23
2.3 The Tokamak . . . . .	27
2.3.1 Tokamak Equilibrium . . . . .	28
2.3.2 Magnetic Coordinates . . . . .	31
2.3.3 Currents and Fluxes . . . . .	34

2.3.4	Particle Trajectories in a Tokamak . . . . .	37
2.4	The Particle Equations of Motion . . . . .	39
2.5	Plasma Kinetic Theory . . . . .	47
2.5.1	The Phase Space Distribution Function . . . . .	48
3	The $\delta f$ Method . . . . .	53
3.1	Introduction . . . . .	54
3.2	Derivation of the $\delta f$ Method . . . . .	56
3.3	The Monte Carlo Code . . . . .	58
3.4	Theory of the Bootstrap Current . . . . .	64
3.5	Theory of the Pfirsch–Schlüter Current . . . . .	68
3.6	Relating Monte Carlo Results to Theoretical Results . . . . .	70
3.7	Results . . . . .	72
3.8	Momentum Conservation . . . . .	82
3.9	Electron Contribution . . . . .	88
3.10	Non-Axisymmetric Configuration . . . . .	92
3.11	New Set of Variables . . . . .	96
3.12	Enforcement of Quasi-Neutrality . . . . .	98
4	Conclusion . . . . .	110
	Appendix: The $\delta f$ Method . . . . .	112
	References . . . . .	141
	VITA . . . . .	143



# Acknowledgements

This dissertation would not have been possible without the support of countless people. Though it is not feasible to adequately express my gratitude to all who deserve it, I shall attempt my best. My thanks to...

Allen Boozer, for his guidance, advice and patience (and tolerance towards my frequent out-of-state trips...) Allen's knowledge and insight allow him to see right through a problem to its core — they have never ceased to amaze me. Allen, it has been a privilege to work with you. Thank you for everything.

My Mom and Dad, for raising me the way I am. Their selfless act of leaving Poland and everything we owned behind to create a better future for my sister and me seems to have paid off...

Gene Tracy, for his friendship and advice. Gene, I'll never forget our first conversation. Thank you for believing in me.

George Vahala, for hundreds of hours of one-on-one plasma physics lectures and all of our interesting discussions.

Henry Krakauer and Charlie Johnson, for squeezing my defense into their busy schedules. Thank you for your time, effort and comments on my dissertation.

Dirk Walecka and all other members of the faculty and staff (Paula, thanks for everything!) at our physics department, for their attitude, respect and support towards the students. I consider myself very lucky to have had you at my side.

The physics students at William and Mary, for creating an environment that is very

difficult to leave behind. Special thanks to Xianzhu Tang, for our usefull discussions and for being a great apartment-mate in NYC; Qun Yao, for his help during the beginning stages of my research; Sam Varner and Andy Wilson, for the countless hours of “blowing off” time in our office; Alastair Neil, Chree Haas and Matt Coffey, for their patience and help with the work stations; Gary Rutledge, for taking one worry off my head; and finally to my friends from the Fall '91 first-year class: Daniele Gaetano (I'm looking forward to working with you!), Rob Martin, Justin McIntyre, Chris Nichols, Jaehwan Oh and Phil Spickler (Thanks for being a great roomie!), for the late night problem-solving sessions. Good luck to all of you!

Mike West, Don Berry and all other members of the faculty, students and staff at the Institute of Statistics and Decision Sciences at Duke University, for their hospitality and friendship. Thanks for the great office space and our Saturday morning soccer games.

Everybody at the applied mathematics and applied physics department at Columbia University, for hosting me for a semester.

Stephen Hanzely and Ed Mooney at Youngstown State University, for their mentorship.

Bill and Sandy Black, Lynn and Judy Weekley and Louis and Frances DeSimone (and Steph'!) for giving me a home away from home.

Oh, yeah ... almost forgot ... the people at Caltech, for making me decide not to go there...

Finally, the most important reason why my experience at William and Mary has been so positive: my soon to be wife, Heather. The last four years would have been a lot less fun without you. Thank you for putting up with me.

I love you.

# List of Tables

1-1	Approximate densities and temperatures for typical plasmas . . . . .	4
1-2	Several of the new, large toroidal confinement devices. Devices marked with stars have not yet been built, all others have been built in the 1980's. The devices with italicized device names are stellarators, the others are tokamaks. . . . .	11

# List of Figures

1-1	The product of the density and energy confinement time versus temperature. The minimum value of $n\tau_E$ gives the so-called Lawson's criterion.	6
1-2	A geometrical torus and the terminology associated with toroidal geometry. . . . .	9
2-3	Electron guiding-center motion $\mathbf{v}_{\text{guiding}}$ along the magnetic field line $\mathbf{B}$ compared with its exact motion $\mathbf{v}_{\text{gyro}}$ . . . . .	18
2-4	Magnetic trapping occurs only for particles with total energy $E < \mu B_{\text{max}}$ . Particles with higher energy circulate freely. . . . .	20
2-5	The $\mathbf{E} \times \mathbf{B}$ drift arises from the acceleration of the electric field, leading to a difference in the magnitude of the gyroradius. . . . .	22
2-6	Particle drift in an inhomogeneous magnetic field. . . . .	23
2-7	A picture of the <i>Joint European Torus</i> (JET) located in the Culham Laboratories in Abingdon, U.K. JET came on line in June of 1983 as a collaboration of several European countries operating under the auspices of EURATOM. (This image was scanned from J. Wesson's book <i>Tokamaks</i> [3]). . . . .	28
2-8	The curvature and gradient $B$ drifts produce up-down charge separation inducing an electric field which produces the outward $\mathbf{E} \times \mathbf{B}$ drift. . . .	30
2-9	The magnetic flux surfaces form nested toroids. . . . .	31
2-10	Definitions of canonical coordinates used to describe a tokamak. The poloidal and toroidal angles $\theta$ and $\phi$ and the poloidal and toroidal magnetic fluxes $-\chi$ and $\psi$ are defined. The definitions of the toroidal and poloidal currents $G$ and $I$ are also given. . . . .	32

2-11	Projection of the guiding center motion for a deeply trapped particle in a tokamak magnetic field. . . . .	38
3-12	The bootstrap current $j_b \approx e\delta(T/m)^{1/2} dn/d\psi$ is due to the radial density gradient of the plasma ( $n_1 > n_2$ ). $C_L$ marks the center line of the torus.	64
3-13	Comparison of aspect ratio dependence of $\Delta_0(\epsilon)$ obtained from Eq. (3-180) (dashed line) and from numerical integration of Eq. (3-182) (solid line). . . . .	66
3-14	Simulated bootstrap and Pfirsch–Schlüter currents (solid lines) and their asymptotic fits (dashed lines). The bootstrap current is the greater of the two. . . . .	72
3-15	Normalized bootstrap current vs. collisionality at $\epsilon = 0.1$ . The circles represent values obtained from Monte Carlo simulations. The dotted line is the theoretical prediction given by Hinton and Rosenbluth ( $j_b \propto [1 + \nu_*^{1/2} + 0.54\nu_*]^{-1}$ ) and the solid curve represents a fit where $j_b \propto [1 + \nu_*^{1/2} + 1.44\nu_*]^{-1}$ . . . . .	73
3-16	Normalized components of the bootstrap current vs. inverse aspect ratio at $\nu_* = 0.1$ . The squares and circles represent contributions due to trapped and passing particles, respectively. The total bootstrap current is represented by triangles. The solid line is an analytic prediction for the passing particle contribution. . . . .	74
3-17	Log-log plot of the ratio of the fraction of trapped and passing particles vs. the ratio of the contribution to the bootstrap current due to the trapped and passing particles. The line represents the analytical prediction with slope 3/2. . . . .	75

3-18	Normalized components of the bootstrap current divided by the square root of epsilon vs. epsilon at $\nu_* = 0.1$ . The squares, circles and triangles represent the trapped particle contribution, passing particle contribution and the sum, respectively. The solid line is an analytic prediction for the passing particle contribution. . . . .	76
3-19	$(j_b)_{\text{Monte Carlo}}$ vs. $(j_b)_{\text{theoretical}}$ . The solid line marks the unit slope. . .	77
3-20	$\text{Cos}(\theta)$ and $\text{cos}(2\theta)$ components of the Pfirsch–Schlüter current vs. collisionality at $\epsilon = 0.1$ . . . . .	78
3-21	Normalized $\text{cos}(\theta)$ (circles) and $\text{cos}(2\theta)$ (squares) components of the Pfirsch–Schlüter current vs. inverse aspect ratio at $\nu_* = 0.1$ . . . . .	79
3-22	Normalized $\text{cos}(\theta)$ (circles) and $\text{cos}(2\theta)$ (squares) components of the Pfirsch–Schlüter current vs. inverse aspect ratio obtained from simulations of the Vlasov equation ( $\nu_* = 0.0$ ). . . . .	80
3-23	$(j_{P.S.})_{\text{Monte Carlo}}$ vs. $(j_{P.S.})_{\text{theoretical}}$ . The stars and squares represent the $\text{cos}(\theta)$ component of the Pfirsch–Schlüter current with $\nu_* \neq 0$ and $\nu_* = 0$ , respectively. The triangles and circles represent the $\text{cos}(2\theta)$ component with $\nu_* \neq 0$ and $\nu_* = 0$ , respectively. The solid line marks the unit slope. . . . .	81
3-24	The parameter $k$ vs. the inverse aspect ratio $\epsilon$ . . . . .	86
3-25	Results for the bootstrap current obtained with momentum conservation normalized to the theoretically predicted values as a function of the inverse aspect ratio. . . . .	87
3-26	The ion and electron bootstrap currents obtained by executing the code concurrently for both species (solid line) and first for the ions and then electrons using the ion results in the electron code (dashed line). . . . .	90

3-27	A comparison of the results for the bootstrap current obtained from a single-species, non-momentum conserving simulation and the sum of the momentum conserving ion and electron contributions to the bootstrap current for several values of collision frequency and aspect ratio. The solid line marks the unit slope. . . . .	91
3-28	The rotational transform $\iota$ as a function of the normalized radial distance. $\psi_a$ gives the toroidal flux at the edge of the plasma. . . . .	93
3-29	The destruction of the banana orbits by the toroidal ripple $\delta_r$ . . . . .	94
3-30	The bootstrap current vs. time in an axisymmetric configuration and a configuration with a small toroidal field ripple. The solid line represents $\delta_r = 0.0$ , the dashed line represents $\delta_r = 0.02$ . . . . .	95
3-31	The bootstrap current vs. time for two values of the toroidal field ripple. The solid line represents $\delta_r = 0.07$ and the dashed line represents $\delta_r = 0.15$ . . . . .	96
3-32	The $S_{01}$ Fourier coefficient of the density for the ions and electrons. The solid lines represent the coefficients without enforcement of quasi-neutrality. The dashed lines show the density coefficients with quasi-neutrality enforcement. . . . .	102
3-33	The diffusion coefficient vs. time for the ions (solid line) and electrons (dashed line). . . . .	106
3-34	Ion transport as a function of collisionality. The solid line represents the diffusion coefficients for the ions in an axisymmetric configuration. The squares represent the diffusion coefficients for $\delta_r = 0.01$ and the triangles represent the diffusion coefficients for $\delta_r = 0.1$ . . . . .	107

3-35 Electron transport as a function of collisionality. The solid line represents the diffusion coefficients for the electrons in an axisymmetric configuration. The squares represent the diffusion coefficients for  $\delta_r = 0.01$  and the triangles represent the diffusion coefficients for  $\delta_r = 0.1$ . . . . . 108



# ABSTRACT

A Monte Carlo code has been developed which very efficiently calculates plasma parameters, such as currents, potentials and transport coefficients for a fully three dimensional magnetic field configuration. The code computes the deviation,  $\hat{f}$ , of the exact distribution function,  $f$ , from the Maxwellian,  $F_M$ , with  $\psi$  the toroidal magnetic flux enclosed by a pressure surface and  $H$  the Hamiltonian. The particles in the simulation are followed with a traditional Monte Carlo scheme consisting of an orbit step in which new values for the positions and momenta are obtained and a collision step in which a Monte Carlo equivalent of the Lorentz operator is applied to change the pitch of each particle. Since the  $\delta f$  code calculates only the deviations from the Maxwellian rather than the full distribution function, it is about  $10^4$  times as efficient as other Monte Carlo techniques used to calculate currents in plasmas.

The  $\delta f$  code was used to study the aspect ratio and collisionality dependence of the bootstrap current and two Fourier components of the Pfirsch-Schlüter current. It was also used to calculate electric potentials within magnetic surfaces due to the explicit enforcement of the quasi-neutrality condition. The code also calculated transport coefficients for the ions and electrons under various conditions. The agreement between the values predicted by the code for the plasma currents and analytic theory is excellent. The transport parameters calculated for the ions and electrons are in qualitative agreement with values predicted from neoclassical transport theory, including transport induced by a toroidal ripple. The in-surface electric potentials induced by explicitly enforcing the quasi-neutrality condition are too small to significantly enhance transport across the magnetic surfaces.

*To my mom,  
for all her love*

A  $\delta f$  MONTE CARLO METHOD TO CALCULATE PARAMETERS IN PLASMAS

# Chapter 1

## Introduction

### 1.1 Motivation

Monte Carlo methods have been an important tool in plasma simulations. Despite their usefulness, traditional Monte Carlo methods are very inefficient when applied to systems which are described by distribution functions which deviate only slightly from a Maxwellian, such as thermonuclear plasmas used in fusion research. Thermonuclear plasmas are confined over times much longer than a collision time which means that the deviations,  $\delta f$ , from a Maxwellian,  $F_M$ , are very small. Typically, for plasmas of fusion interest,  $\delta f/F_M \sim 1\%$ . In Monte Carlo simulations, statistical fluctuations of the results scale inversely with the square-root of the number of particles in the simulation. In other words, if the number of particles is increased by a factor of a hundred, the statistical fluctuations decrease by a factor of ten. Using a traditional Monte Carlo simulation which calculates the full distribution function, at least 10,000 particles are needed to detect fluctuations of order 1%, since  $1/\sqrt{10,000} = 0.01$ . If, on the other hand, we can devise a method which calculates only those deviations, rather than the full distribution function, we could see those deviations using a single particle. Since the time spent on a simulation is proportional to the number of particles involved in the simulation, the efficiency increase of such a method over traditional Monte Carlo methods would be of order  $10^4$ . The  $\delta f$  Monte Carlo method described

in this dissertation is such a method. The method is derived and described in detail in Chapter 3. It is first applied to simple systems for which there exist results from analytic theory and numerical simulations (section 3.7) before it is applied to more complicated systems (sections 3.8–3.12). Before the  $\delta f$  method is described, Chapter 1 gives an introduction to plasma confinement, thermonuclear fusion and numerical simulations of plasmas. In Chapter 2 we discuss the basic theory of plasma physics, including particle motion in electromagnetic fields, kinetic theory and magnetic coordinates. The majority of Chapter 2 consists of a compilation and the author's interpretation of parts of several excellent introductory plasma physics books [1, 2, 3, 4, 5, 6, 7, 8]. Chapter 4 gives concluding remarks about this work.

## 1.2 What is a Plasma?

Consider heating a solid—a block of ice, say. At normal pressure, if it is heated past  $0^\circ$  Celsius, the individual molecules acquire enough thermal energy to break out of the lattice structure associated with solids. The ice goes through a phase transition and becomes water—a liquid. Let us continue the heating process. At around  $100^\circ$  Celsius, or 372 Kelvin, the individual molecules will have enough thermal energy to completely break their bonding with neighboring molecules and another phase transition occurs. The water becomes water vapor—a gas. If the vapor is heated to even higher temperatures, above 10,000 Kelvin, the thermal energy of the constituents becomes high enough to break the electrostatic bonds which hold the electrons and ions together. The vapor enters the fourth state of matter—a plasma. The most notable difference between a plasma and a gas is that a plasma is a good conductor of electricity because it consists of charged particles, the electrons and the ionized nuclei. A gas, on the other hand, consists of neutral molecules which cannot carry currents. The degree of ionization of the plasma increases as the temperature is increased. For hydrogen at 1 Atmosphere, the

ionization becomes almost complete at around 20,000–30,000 Kelvin. A vast majority of the matter in the universe is in the plasma state. On earth, however, the conditions for a plasma to exist have to be created artificially. The most widely used applications of plasmas are probably fluorescent lights and plasma etching of electronic components. Outside of the surface of the earth, the examples of known plasma phenomena are countless: the Aurora Borealis and lightning in the Earth atmosphere, solar wind, flares in the solar corona, stellar interior, interstellar gas, gaseous nebulae and, in science fiction, plasma guns on alien spaceships.

Table 1-1 shows approximate densities and temperatures for some typical plasmas [9]. The plasma discussed in this dissertation is constrained to the category of thermonuclear plasma.

Table 1-1: Approximate densities and temperatures for typical plasmas

Plasma Type	Density (cm) <sup>-3</sup>	Temperature (K)
Interstellar gas	1	10 <sup>4</sup>
Gaseous nebula	10 <sup>3</sup>	10 <sup>4</sup>
Ionosphere (F layer)	10 <sup>6</sup>	10 <sup>3</sup>
Solar corona	10 <sup>6</sup> –10 <sup>9</sup>	10 <sup>6</sup>
Solar atmosphere	10 <sup>14</sup>	10 <sup>4</sup>
Thermonuclear plasma	10 <sup>14</sup>	10 <sup>8</sup>
Laser plasma	10 <sup>20</sup>	10 <sup>6</sup>
Metal plasma	10 <sup>23</sup>	10 <sup>2</sup>
Stellar interior	10 <sup>27</sup>	10 <sup>7</sup>

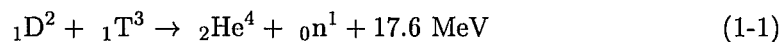
The word *plasma* comes from the Greek *plásma* which means “something formed or molded” [8]. The first use of the word related to its modern use in physics occurred in 1929 by Tonks and Langmuir [10] in conjunction with oscillations observed in a gaseous discharge. They referred to those oscillation as *plasma oscillations*. For several decades afterwards, plasma physics research was conducted by only a few individuals. It was

not until the early 1950's when the prospect of thermonuclear fusion became possible that the field of plasma physics began growing rapidly.

### 1.3 Thermonuclear Fusion

The goal of thermonuclear fusion research is to create a plasma that will generate more energy through fusion reactions than is required to create and sustain the plasma. To achieve that goal, the so-called ignition condition of the plasma must be reached. The ignition condition is the point at which the temperature of the plasma is high enough for the thermonuclear power within the plasma to balance all power losses which will then allow external heating to be discontinued. The ignition condition depends on three factors. In simple terms, the plasma has to be hot enough, dense enough and remain so for a sufficiently long period of time. We will discuss the ignition condition in more detail below, but first let us consider the fuel used in thermonuclear fusion.

The most promising fusion reaction is that in which a nucleus of deuterium and tritium fuse to produce an alpha particle and a neutron. The nuclear rearrangement results in a reduction of total mass which leads to a release of energy in the form of kinetic energy of the alpha particle and neutron. The energy released per reaction is 17.6 MeV.



The fuel reserves are practically inexhaustible, as deuterium occurs in water with an abundance of about one part per 10,000. Tritium, even though it does not occur naturally, can be bred using the neutrons from the nuclear reaction. The element used to breed tritium is lithium, of which there exist large reserves.

To achieve fusion between two positively charged particles, the ions must have enough kinetic energy to overcome the electrostatic repulsion due to the Coulomb force between them. The cross-section for the deuterium-tritium reaction increases as the

temperature is increased, until it peaks at around 100 keV (about one billion degrees Celsius). It is fortunate, however, that the plasma does not have to be heated that high for fusion to occur because the reactions occur for particles in the high-energy tails of the Maxwellian energy distribution. The necessary temperature is “only” about 10 keV. As mentioned above, to achieve ignition the plasma energy has to be confined long enough at a sufficiently high density to allow an adequate fraction of the fuel to react. The condition for ignition is calculated by balancing the total power produced by the reactions with the power losses occurring in the plasma. Figure (1-1) shows the so-called Lawson diagram for the D-T reaction. It shows that at the necessary temperature of about 10 keV, the product of the plasma density and confinement time must be greater than about  $10^{20} \text{m}^{-3} \text{sec}$ .

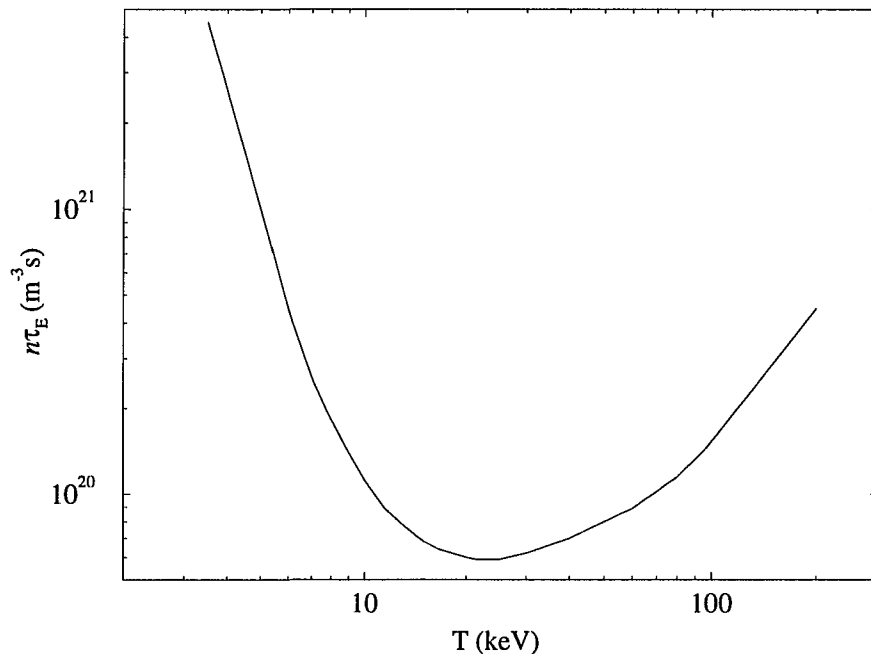


Figure 1-1: The product of the density and energy confinement time versus temperature. The minimum value of  $n\tau_E$  gives the so-called Lawson's criterion.



Two questions arise naturally: How do we heat the plasma efficiently to such high temperatures? And... How do we confine this hot mixture of ions and electrons?

### 1.3.1 Plasma Heating

Let us very briefly consider plasma heating. The initial heating in a plasma comes from the Ohmic, or resistive, heating due to collisions between plasma particles. However, for plasma temperatures above about 1 keV, the effect of plasma resistance becomes much reduced due to the decrease in collisions between particles (the collision frequency of the plasma particles scales as the inverse of the plasma temperature to the three-halves power:  $\nu \propto T^{-3/2}$ ). The two main methods used to heat a plasma to ignition temperatures are injection of energetic neutral beams and the resonant absorption of radio-frequency (RF) waves. The beams used to heat plasmas must be composed of neutral particles because—as will be discussed below—the magnetic field used to confine the plasma would reflect any charged particles. The particles which eventually end up as neutral particles must first be ionized in order to be accelerated to the required energy. They are then neutralized by charge exchange in a gas target and injected into the plasma. Inside the plasma the particles become ionized again and are slowed down by collisions with the plasma particles, giving up their energy to the plasma and thus heating it.

The three main methods of RF heating involve waves at the ion cyclotron frequency ( $\sim 50\text{MHz}$ ), electron cyclotron frequency ( $\sim 100\text{GHz}$ ) [see subsection 2.1.1] and the lower hybrid frequency ( $\geq 1\text{GHz}$ ). The waves must propagate deep into the plasma, where the most energetic particles are located. This requires calculation of wave propagation in non-uniform densities and non-uniform fields and transfer of wave energy to the plasma particles. The detailed theory of neutral beam and RF heating are beyond the scope of this dissertation.

### 1.3.2 Plasma Confinement

Due to the extremely high temperatures of the thermonuclear plasma, confinement by material walls is impossible. Two alternative methods of confinement were developed: magnetic confinement and inertial confinement. In inertial confinement research, the goal is to compress a plasma to ultra-high density ( $\sim 10^{25} \text{ cm}^{-3}$ ) and let the criterion for ignition be satisfied within the time it takes the plasma to blow apart. To do so, very high power density laser beams are used to deliver energy onto a tiny fuel pellet. The design of energy drivers and high gain pellets are the primary tasks of inertial fusion research.

The other method of plasma confinement—magnetic confinement—takes advantage of the fact that the plasma consists of charged particles and therefore interacts with magnetic fields. In particular, as will be seen in the following chapter, charged particles can be confined by magnetic fields because they gyrate around magnetic field lines. Various magnetic field configurations have been proposed and built to confine a plasma. They can be broadly classified into two categories: open-ended confinement and toroidal confinement. In open-ended confinement magnetic coils are arranged to produce the so called magnetic mirror effect [cf. subsection 2.1.1, page 19] to prevent particles from escaping the vessel they are contained in. The magnetic field lines in open-ended confinement do not close in on themselves and those particles which are not reversed by the magnetic mirror effect escape out of the ends of the device. This particle loss, referred to as end loss, is generally thought to be too large for fusion applications.

End loss can be eliminated by bending the field lines into a torus. As will be discussed in subsection 2.3.1, the magnetic field lines in toroidal confinement form toroidally nested magnetic surfaces on which the particles move about. Figure 1-2 shows a picture of a torus and defines some of the terminology associated with toroidal geometry.

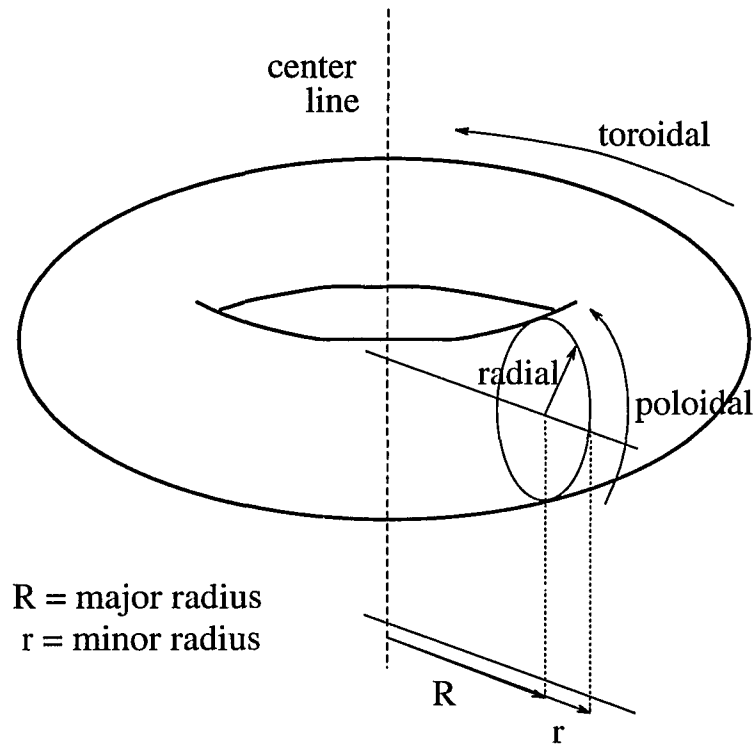


Figure 1-2: A geometrical torus and the terminology associated with toroidal geometry.

The main component of the magnetic field in a toroidal confinement device is the toroidal field produced by external coils. In subsection 2.1.2 it will be shown that a toroidal field alone is not able to confine the plasma because of particle drifts. The magnetic field lines must have a helical twist to confine the particles. If the helical twist is provided by a toroidal plasma current it is called a tokamak. If, on the other hand, the external magnetic coils are arranged in such a way to produce the helical twist of the field lines, the device is called a stellerator. Due to the symmetry of the toroidal magnetic field around the center-line, the tokamak is referred to as an axisymmetric device. The stellerator, on the other hand, possesses non-axisymmetric geometry.

Most of the research in plasma fusion today is carried out in tokamaks. Table (1-2)

shows some parameters for several large toroidal confinement devices built in the 1980's. The italicized devices are stellarators. The last three devices marked with stars have not yet been built and the values for the toroidal field; plasma current; temperature; and the product of density, energy confinement time and temperature ( $n\tau_E T$ ) are predicted values.

Not unlike all other large scale experiments performed today, the time and material resources required to design, build and maintain large scale plasma experiments are tremendous. The maintenance cost for the Princeton Plasma Physics Lab is on the order of a hundred million dollars a year. The predicted cost for ITER is on the order of ten billion dollars. Considering the steady decrease of the budget for fusion since the mid-eighties and the tremendous advances in computing speed, it is not surprising, therefore, that over the past decade the trend in plasma physics has moved toward numerical simulations.

Table 1-2: Several of the new, large toroidal confinement devices. Devices marked with stars have not yet been built, all others have been built in the 1980's. The devices with italicized device names are stellarators, the others are tokamaks.

Device Name, Location	Major radius (m)	Minor radius (m)	Toroidal field (T)	Plasma current (MA)	$n\tau_E T$ $\times 10^{20}$ ( $m^{-3} s keV$ )	T (keV)
JET (Joint European Torus), Culham, U.K.	3.0	1.2	3.5	3.5	10	30
JT60, Naka, Japan	3.0	0.9	4.5	2.3	0.8	3
TFTR (Tokamak Fusion Test Reactor), Princeton, USA	2.4	0.8	5.0	2.5	4	30
<i>Wendelstein VII-AS</i> , Garching, Germany	2.0	0.2	3.0		0.03	0.3
ASDEX Upgrade, Garching, Germany	1.6	0.4	2.7	1.6	0.2	1
* <i>Wendelstein VII-X</i> , Garching, Germany	5.5	0.5	3.0		4	5
* <i>LHD (Large Helical Device)</i> , Nagoya, Japan	3.9	0.5	3.4		??	??
*ITER (International Thermonucl. Reactor) Site not yet chosen	6.0	2.0	4.9	22	??	??

## 1.4 Numerical Simulations of Plasmas

Numerical simulations of plasmas have begun in the 1960's, shortly after the prospect of thermonuclear fusion had become a possibility. They have played an essential role in the development of plasma theory. Numerical simulations have allowed scientists to test new theories used to describe plasma phenomena and even allowed them to predict some, not yet experimentally seen plasma behavior. Simulations have also become an efficient design tool to provide performance predictions of confinement devices before entering the engineering phase. Many numerical codes are used to predict the optimum design of ITER [11, 12, 13, 14, 15]. One step beyond the theoretical experiments and engineering experiments described above is the “numerical tokamak” envisioned by some (very optimistic) computational plasma physicists. For a detailed review of plasma simulations, in particular plasmas for fusion applications, the reader is encouraged to examine a recent review paper by W. Arter [16].

Numerical simulations of any kind require an almost experimental approach to them. Indeed, the peculiar term “numerical experimentalist” is not as inaccurate as may be assumed. Experimentalists may argue that the only physics computational physicists get out of the computer is the physics they put in. To some extent they are correct. However, often the theory used to describe physical systems is so complex that it is beyond hope to try to solve it analytically beyond some asymptotic limits which simplify calculations. The analogy of numerical simulations with real life experiments are quite numerous. A well-engineered simulation will have undergone a designing stage to ensure cost effectiveness (time efficiency). Furthermore, a simulation should have a modular structure with built-in diagnostics which allow for checking it part by part. In fact, wherever possible, every part of the simulation should be checked for correct performance before it is integrated into the main body of the code. Once the code is assembled, it should be tested against known results obtained from other codes and

from analytic theory in asymptotic limits. Only after the testing and calibration are complete is the simulation ready to run, which brings forth the final analogy between real life and numerical experiments, the vast amounts of data to be interpreted.

A plasma consists of a large number of charged particles moving in electric and magnetic fields. We must realize, however, that using slowly varying prescribed fields is a vast simplification of the actual physics because we only deal with the effects of the fields on the particles, but neglect the effects the particles have on the fields. A general solution of this problem, therefore, is extremely complicated. Laboratory plasmas have volumes of 1–100 m<sup>3</sup> with a density of about  $10^{20}$  m<sup>-3</sup>. A simple minded simulation of the trajectory of every particle in the plasma followed by a calculation of the fields due to every particle is, therefore, vastly beyond even the fastest computer (and will remain so for the foreseeable future). Besides the inability of representing the true number of particles in the system of interest, another humbling experience every computational physicist must face is the inability to represent a continuous system on a computer. A suitable scale must therefore be chosen on which the discreteness of the numerical simulation will become negligible and the discrete steps taken in the simulation will represent continuity with sufficient accuracy. We have to realize, however, that the smaller we “coarse grain” the system, the more expensive (time consuming) the simulation becomes. A balance must be found between the two aspects of representing a continuous system with adequate accuracy and not making the simulations unnecessary long.

Numerical simulations of plasma physics can be broadly classified into two categories: kinetic simulations and fluid simulations. In fluid simulations, the goal is to solve the magnetohydrodynamic (MHD) equations of a plasma. Kinetic simulations consider more detailed models of the plasma given by the kinetic equations. If the fluid and kinetic description are combined in a simulation, it is referred to as a hybrid simulation.

Kinetic simulations can be further classified into several categories. The first classification depends on the type of kinetic equations that are being simulated. If only the lowest order terms in the averaged gyro-motion of the particles are being considered, the model is referred to as a drift-kinetic model [cf. subsection 2.1.1, pages 17 and 18]. If higher order corrections to the drifts are used, the model is called a gyro-kinetic model. Kinetic simulations are further classified into collisionless (Vlasov) models in which the particles are treated as uncorrelated, and collisional (Fokker–Planck) models in which the particle interactions are modeled by a collision operator. The collisionless treatment is only valid for times shorter than the particle collision times. For times greater than the collision times of the particles, the diffusion in velocity space must be considered. The diffusion in velocity space can be modeled by either the use of a velocity space mesh to calculate velocity gradients, or the so-called Monte Carlo methods, in which a random number generator is used to simulate the effects of particle collisions by scattering parameters in velocity space.

The concept of simulating particles by integrating their equations of motion forward in time may seem quite intuitive. The mathematical significance of the equations of motion which allows us to use them to solve a partial differential equation such as the Fokker–Planck equation involves the concept of characteristics, which happen to be the equations of motion. Therefore, what physicists look at as simulating the motion of particles along their trajectories, mathematicians may prefer to view as an integration of a PDE along the characteristics of that equation. The Fokker–Planck equation contains a hyperbolic part (representing orbital motion) and an elliptic part (representing collisions). Analytically, as well as in a traditional Monte Carlo simulation, the orbital part of the equation is solved by integrating along the characteristics of the equation given by the particle trajectories. Solving it in this fashion (by moving along particle trajectories) corresponds to the Lagrangian formalism, which is defined by a coordinate



system that moves with the particles. The collisional part is solved analytically by the Eulerian method, which means that a time-independent coordinate system is used to describe the velocity space operations. Numerically, the gradients in velocity space in the collisional part can be solved either using an Eulerian or Lagrangian model. In an Eulerian scheme, a time independent velocity space grid is used to calculate the gradients. In a Lagrangian model, the particles are assigned a parameter called the pitch [cf. subsection 2.1.1] which is scattered by using a Monte Carlo method.

Before we continue discussing numerical simulations of plasmas, it is necessary to form a basic understanding of the theory used to describe plasma behavior. The following chapter gives a brief overview of basic plasma physics theory.

# Chapter 2

## Basic Plasma Physics

### 2.1 Individual particle motion

#### 2.1.1 Gyromotion, Magnetic Moment and Pitch

In this subsection, we will briefly discuss the motion of a single charged particle moving in a prescribed electromagnetic field. The force acting on a particle due to the electric field  $\mathbf{E}$  and the magnetic induction  $\mathbf{B}$  is the Lorentz force. (The magnetic induction is usually—and incorrectly—referred to as the magnetic field. This dissertation will not be an exception. . . ) If relativistic effects are neglected, the particle mass is constant and the equation of motion for a charged particle of mass  $m$  and charge  $q$  is

$$m \frac{d\mathbf{v}}{dt} = q(\mathbf{E} + \mathbf{v} \times \mathbf{B}). \quad (2-2)$$

Let us consider two simple cases:

If  $\mathbf{E} = \text{constant} \neq 0$ , but  $\mathbf{B} = 0$ , the particle velocity changes linearly with time and the particle accelerates freely in the direction of  $\mathbf{E}$ . This situation is not of great interest.

In the other case,  $\mathbf{E} = 0$  and  $\mathbf{B} = \text{constant} \neq 0$ . It is useful to separate the motion into components parallel and perpendicular to the magnetic field lines. Let us introduce the notation

$$\mathbf{b} \equiv \mathbf{B}/|\mathbf{B}|, \quad v_{\parallel} \equiv (\mathbf{v} \cdot \mathbf{b}), \quad \mathbf{v}_{\perp} \equiv \mathbf{v} - v_{\parallel} \mathbf{b}. \quad (2-3)$$

From the parallel and perpendicular components of Eq. (2-2) we find  $\dot{v}_{\parallel} = 0$  and  $\dot{\mathbf{v}}_{\perp} = -\omega_c^2 \mathbf{v}_{\perp}$ ; the parallel velocity component is a constant and the perpendicular component executes a simple harmonic motion with  $\omega_c = |q|B/m$  known as the gyro-frequency, Larmor frequency or cyclotron frequency. Introducing Cartesian coordinates  $(x, y, z)$  and choosing  $\mathbf{B} = (0, 0, B)$  we have

$$\begin{aligned} v_x &= v_{\perp} \cos(\omega_c t + \phi_0), \\ v_y &= -v_{\perp} \sin(\omega_c t + \phi_0), \\ v_z &= v_{\parallel} = \text{const}, \end{aligned} \tag{2-4}$$

and

$$\begin{aligned} x &= x_0 + \frac{v_{\perp}}{\omega_c} \sin(\omega_c t + \phi_0), \\ y &= y_0 - \frac{v_{\perp}}{\omega_c} \cos(\omega_c t + \phi_0), \\ z &= z_0 + v_z t. \end{aligned} \tag{2-5}$$

The constants of integration  $\phi_0$  and  $x_0, y_0, z_0$  are the initial phase of the gyromotion and the initial coordinates of the particle, respectively. The superposition of the parallel and perpendicular components leads to a right-handed (left-handed) helical motion around the magnetic field line for the electrons (ions). The radius of gyration, called the gyro- or Larmor radius, is

$$\rho_L = \frac{v_{\perp}}{\omega_c}. \tag{2-6}$$

Remembering  $\omega_c = |q|B/m$ , we obtain  $(\rho_L)_i \approx \sqrt{m_i/m_e}(\rho_L)_e$ .

The position about which the particle gyrates is known as the guiding center. The concept of guiding center motion, which is the particle motion averaged over one gyro-period  $\tau_c = 2\pi/\omega_c$ , is extremely useful when studying particle motion in inhomogeneous fields. Figure 2-3 shows that the guiding center motion for a particle in a uniform magnetic field is a linear motion along  $\mathbf{B}$ , free from rapid helical gyration associated with

exact particle motion.

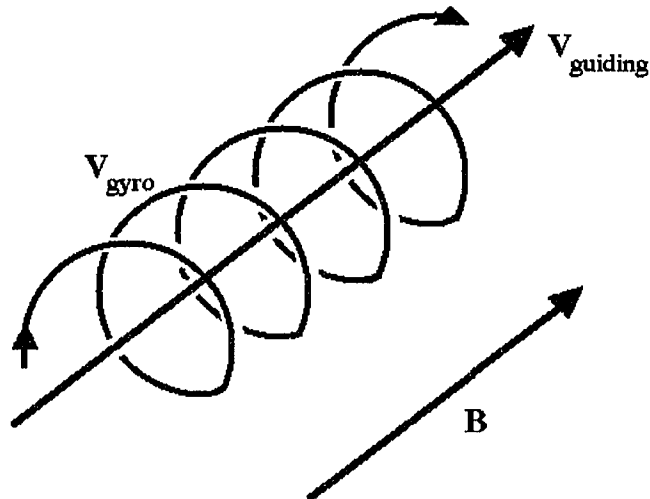


Figure 2-3: Electron guiding-center motion  $\mathbf{v}_{\text{guiding}}$  along the magnetic field line  $\mathbf{B}$  compared with its exact motion  $\mathbf{v}_{\text{gyro}}$ .

The particle gyration composes an electric current loop which leads to the very useful concept of the magnetic moment. The magnetic moment of a current loop with current  $I$  and area  $A$  is  $\mu = IA$ . In the case of a charged particle gyrating in a magnetic field,  $I = q\omega_c/2\pi$ ,  $A = \pi\rho_L^2 = \pi v_\perp^2/\omega_c^2$  and the magnetic moment becomes

$$\mu = \frac{q\omega_c}{2\pi} \frac{\pi v_\perp^2}{\omega_c^2} = \frac{mv_\perp^2}{2B}. \quad (2-7)$$

The magnetic moment is an adiabatic invariant under a slowly varying magnetic field. An important but little known principle from classical mechanics is that in a system under periodic motion with a coordinate  $q$  and conjugate momentum  $p$ , the action, defined by

$$J = \oint p \frac{dq}{dt} dt = \oint p dq \quad (2-8)$$

is a constant of motion under slowly varying external parameters. For a gyrating particle,  $p = mv_{\perp}$ ,  $dq = \rho_L d\phi$ , the action becomes

$$J = \int_0^{2\pi} mv_{\perp} \rho_L d\phi = 2\pi m \frac{v_{\perp}^2}{\omega_c} = \frac{4\pi m}{q} \mu \quad (2-9)$$

which is the magnetic moment to within a constant.

The invariance of the magnetic moment in a slowly varying magnetic field can be applied to confine particles in a system called a magnetic mirror confinement system or a magnetic trap. In the absence of an electric potential, the kinetic energy of the particle is constant

$$H = E_K = \frac{1}{2}mv_{\parallel}^2 + \mu B = \text{constant}. \quad (2-10)$$

Consider a simple mirror machine configuration. If a particle starts out at the minimum field strength (with some initial values of  $\mu$  and  $E_K$ ) and moves along the field line, the value of the local field strength  $B$  increases. To conserve  $\mu = mv_{\perp}^2/2B$ ,  $v_{\perp}$  must increase as well. When the particle reaches a point along the trajectory where  $\mu B = E_K$ ,  $v_{\parallel}$  vanishes and the particle is reflected. The Aurora Borealis seen in the earth's upper atmosphere is a well known example of particle trapping in a magnetic field. In a magnetic mirror, only particles with parallel energy  $mv_{\parallel}^2/2 < \mu(B_{\max} - B)$  are reflected. ( $B_{\max}$  is the maximum value of the field strength.) Particles which satisfy this condition are called trapped particles and particles with  $mv_{\parallel}^2/2 > \mu(B_{\max} - B)$  are referred to as passing particles (Fig. 2-4).

Another parameter used to identify a particle as trapped or passing is called the pitch. The pitch  $\lambda$  of a particle is the projection of the particle velocity onto the magnetic field line

$$\lambda = \frac{\mathbf{v} \cdot \mathbf{B}}{|\mathbf{v}| |\mathbf{B}|} = \frac{v_{\parallel}}{v} = \cos \theta_v, \quad (2-11)$$

with  $\theta_v$ , the angle between  $\mathbf{v}$  and  $\mathbf{B}$ , the pitch angle. The pitch can take on values between  $-1$  and  $1$  with the two extrema being the pitch of a particle moving parallel

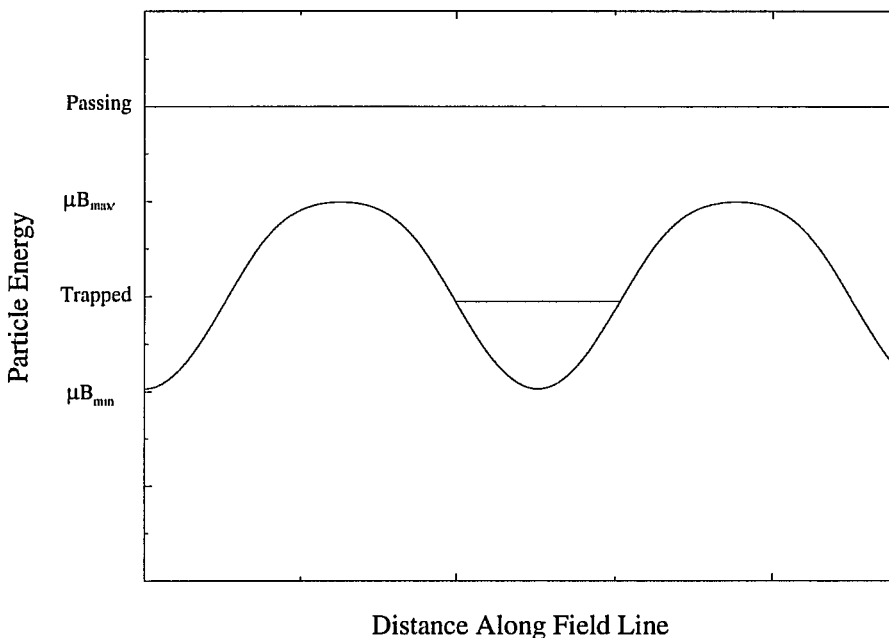


Figure 2-4: Magnetic trapping occurs only for particles with total energy  $E < \mu B_{\max}$ . Particles with higher energy circulate freely.

to the field line ( $\lambda = 1$ ) and the pitch of a particle moving anti-parallel to the field line ( $\lambda = -1$ ). The critical pitch is  $\lambda_c = \sqrt{1 - B/B_{\max}}$ . Particles are trapped or passing depending on whether their pitch is less than or greater than  $\lambda_c$ :

$$\begin{aligned}
 |\lambda| < \sqrt{1 - B/B_{\max}} & \quad \text{trapped} \\
 |\lambda| > \sqrt{1 - B/B_{\max}} & \quad \text{passing.}
 \end{aligned}
 \tag{2-12}$$

In addition to the magnetic moment  $\mu$ , there are two other adiabatic invariants used in plasma physics, the longitudinal and the flux invariant. Since those two invariants played no role in our analysis, we will not discuss them. For more information on the longitudinal and the flux invariant, the reader may refer to any introductory plasma

physics book, eg. Nicholson, *Introduction to Plasma Theory* [8].

### 2.1.2 Particle Drifts

In this subsection, a heuristic development of the  $\mathbf{E} \times \mathbf{B}$ , curvature and gradient  $B$  drifts and the expressions for the corresponding drift velocities will be presented.

#### $\mathbf{E} \times \mathbf{B}$ Drift

Consider a particle gyrating in nonzero  $\mathbf{E}$  and  $\mathbf{B}$  fields. Let us assume that  $\mathbf{E}$  is perpendicular to  $\mathbf{B}$ . In this case, the particle is subjected to a drift velocity in the direction perpendicular to both  $\mathbf{E}$  and  $\mathbf{B}$ . This drift, called the  $\mathbf{E} \times \mathbf{B}$  drift, arises from the acceleration of the particle due to the electric field while it is gyrating around the magnetic field line. In Fig. 2-5, the electric field accelerates the ion on the way up, and decelerates it on the way down. The local gyroradius is therefore larger at the top of the orbit than at the bottom. The resulting motion is a drift motion to the right superimposed on the helical gyromotion along the magnetic field line.

$$\mathbf{v}_{\mathbf{E} \times \mathbf{B}} = \frac{\mathbf{E} \times \mathbf{B}}{B^2} \quad (2-13)$$

Note that the  $\mathbf{E} \times \mathbf{B}$  drift velocity is independent of mass and charge, hence it is equal for ions and electrons. In general, any force  $\mathbf{F} \times \mathbf{B}$  creates a drift of the form

$$\mathbf{v}_{\mathbf{F} \times \mathbf{B}} = \frac{\mathbf{F} \times \mathbf{B}}{qB^2}. \quad (2-14)$$

#### Curvature Drift

The second drift we will consider is due to the inertial force acting on a particle when it moves along a curved path, such as the magnetic field line in a tokamak. The inertial, or centrifugal, force acting on the particle is perpendicular to its path and is given by

$$\mathbf{F}_c = \frac{mv_{\parallel}^2}{R_c} \frac{\mathbf{R}_c}{R_c}, \quad (2-15)$$

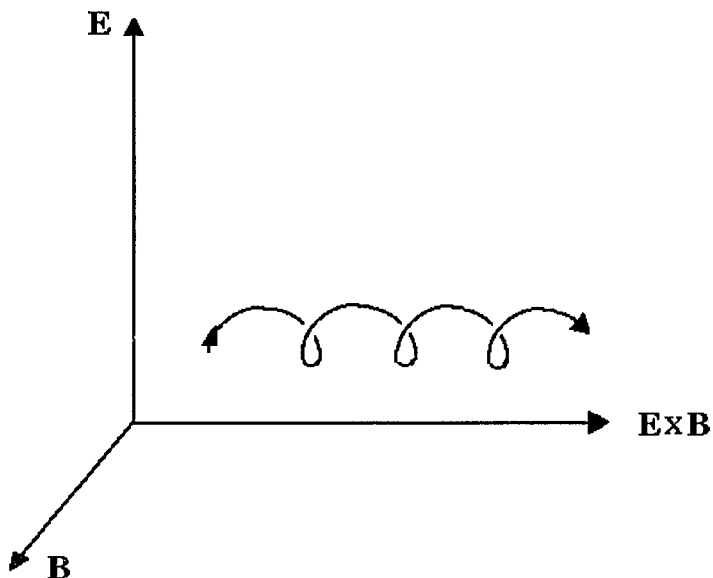


Figure 2-5: The  $\mathbf{E} \times \mathbf{B}$  drift arises from the acceleration of the electric field, leading to a difference in the magnitude of the gyroradius.

with  $R_c$  the local radius of curvature. Substituting Eq. (2-15) into Eq. (2-14) yields the expression for the curvature drift

$$\mathbf{v}_c = \frac{v_{\parallel}^2}{\omega_c} \frac{\mathbf{R}_c \times \mathbf{B}}{R_c^2 B}, \quad (2-16)$$

where  $\mathbf{R}_c$  is in the direction of the centrifugal force. We note that the curvature drift depends on the charge through  $\omega_c$  and the ions and electrons drift in opposite directions.

### Gradient $B$ Drift

Consider a particle gyrating around a magnetic field line in the  $z$  direction and a magnetic field strength variation  $dB/dx \neq 0$ . The local Larmor radius  $\rho_L \propto 1/B$  of the particle will consistently be smaller on the side of the stronger field. The result is a drift in the direction of  $\mathbf{B} \times \nabla B$  called the gradient  $B$  drift (see Fig. 2-6). The expression



for the drift velocity is

$$\mathbf{v}_{\nabla B} = \frac{v_{\perp}^2}{2\omega_c} \frac{\mathbf{B} \times \nabla B}{B^2} \quad (2-17)$$

Let us also note that similar to the curvature drift, the gradient  $B$  drift is in opposite directions for the electrons and ions.

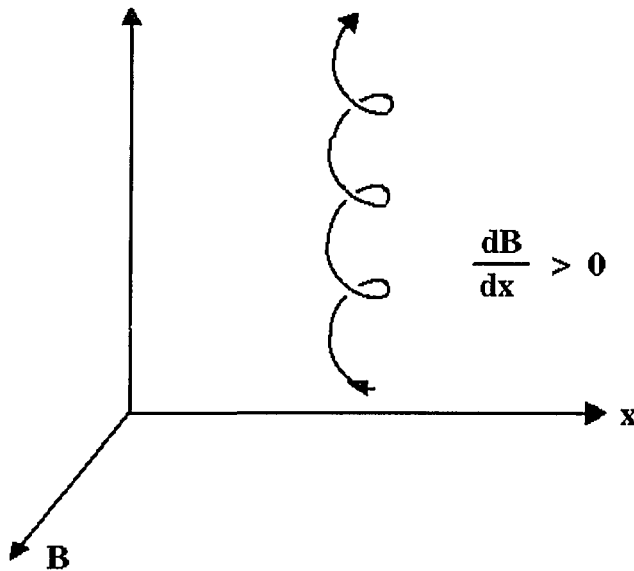


Figure 2-6: Particle drift in an inhomogeneous magnetic field.

The  $\mathbf{E} \times \mathbf{B}$ , curvature and gradient  $B$  drifts form a complete set of the drifts for a particle in a magnetic field to lowest order in gyroradius.

## 2.2 General Coordinates

The study of plasma physics, especially plasma confinement, often deals with geometry for which the Cartesian coordinates used in the previous section are inadequate. One of

the critical issues in toroidal confinement is the drift of the particles across the toroidal pressure surfaces. It is advantageous, therefore, to construct a coordinate system which includes the pressure surfaces as one of the coordinates. The so-called magnetic coordinates exhibit that property. To be able to use magnetic coordinates, we must develop a theory for dealing with coordinates other than the well-known Cartesian, cylindrical and spherical coordinates. This section will describe the theory of general coordinates which will be applied extensively towards the magnetic coordinates in the description of the tokamak in the next section. (The following description of general coordinates as well as subsection 2.3.2 is based on Allen Boozer's *Plasma Physics* [4].)

To specify a position in three dimensions we need three coordinates  $(\xi^1, \xi^2, \xi^3)$ . Using these coordinates, a vector can be represented either using the so-called covariant representation

$$\mathbf{B} = B_1 \nabla \xi^1 + B_2 \nabla \xi^2 + B_3 \nabla \xi^3 \quad (2-18)$$

or the contravariant representation

$$\mathbf{B} = B^1 \frac{\partial \mathbf{x}}{\partial \xi^1} + B^2 \frac{\partial \mathbf{x}}{\partial \xi^2} + B^3 \frac{\partial \mathbf{x}}{\partial \xi^3} \quad (2-19)$$

In the special case of Cartesian coordinates,  $x = \xi^1$ ,  $y = \xi^2$  and  $z = \xi^3$  and the covariant and contravariant representations are identical, with

$$\hat{\mathbf{x}} = \frac{\partial \mathbf{x}}{\partial \xi^1} = \nabla \xi^1, \quad \hat{\mathbf{y}} = \frac{\partial \mathbf{x}}{\partial \xi^2} = \nabla \xi^2, \quad \hat{\mathbf{z}} = \frac{\partial \mathbf{x}}{\partial \xi^3} = \nabla \xi^3 \quad (2-20)$$

and

$$B_1 = B^1, \quad B_2 = B^2, \quad B_3 = B^3. \quad (2-21)$$

(It is important not to confuse  $\frac{\partial \mathbf{x}}{\partial \xi^i} = \nabla \xi^i$ , which is only true for the Cartesian coordinates with  $\frac{\partial \xi^i}{\partial \mathbf{x}} \equiv \nabla \xi^i$ , which is the definition of the derivative of each of the coordinates with respect to position.)

The covariant and contravariant representations are in general not identical. In general, the two representations are related to each other through the Jacobian and the dual relations. The Jacobian for a three dimensional space is defined as

$$\begin{aligned} J &\equiv \frac{\partial \mathbf{x}}{\partial \xi^1} \cdot \left( \frac{\partial \mathbf{x}}{\partial \xi^2} \times \frac{\partial \mathbf{x}}{\partial \xi^3} \right) \\ &= \frac{1}{\nabla \xi^1 \cdot (\nabla \xi^2 \times \nabla \xi^3)}. \end{aligned} \quad (2-22)$$

The orthogonality relation and the the dual relations in ordinary 3-d space are

$$\nabla \xi^i \cdot \frac{\partial \mathbf{x}}{\partial \xi^j} = \delta_j^i \quad (2-23)$$

and

$$\frac{\partial \mathbf{x}}{\partial \xi^i} = \frac{1}{2} \epsilon^{ijk} J (\nabla \xi^j \times \nabla \xi^k) \quad (2-24)$$

$$\nabla \xi^i = \frac{1}{2} \epsilon^{ijk} \frac{1}{J} \frac{\partial \mathbf{x}}{\partial \xi^j} \times \frac{\partial \mathbf{x}}{\partial \xi^k}, \quad (2-25)$$

with  $\epsilon^{ijk}$  the fully anti-symmetric tensor. The orthogonality relation is obtained using the chain rule, and the dual relations are obtained with the help of Eqs. (2-18) and (2-19) and the orthogonality relation. The dual relations greatly simplify the calculations of the gradients of the coordinates because they eliminate the need to construct the inverse functions  $\xi^i(\mathbf{x})$ . In general, given  $\mathbf{x}(\xi^1, \xi^2, \xi^3)$  as the definition of a coordinate system, the inverse functions  $\xi^1(\mathbf{x})$ ,  $\xi^2(\mathbf{x})$ ,  $\xi^3(\mathbf{x})$  are very difficult to obtain.

Using the chain rule, some vector properties and the dual relations, the gradient, divergence and curl in general coordinates are shown to be

$$\nabla f = \frac{\partial f}{\partial \xi^1} \nabla \xi^1 + \frac{\partial f}{\partial \xi^2} \nabla \xi^2 + \frac{\partial f}{\partial \xi^3} \nabla \xi^3, \quad (2-26)$$

$$\nabla \cdot \mathbf{B} = \frac{1}{J} \sum_i \frac{\partial (JB^i)}{\partial \xi^i}, \quad (2-27)$$

and

$$\nabla \times \mathbf{B} = \sum_{i,j} \nabla \xi^i \times \nabla \xi^j \left( \frac{\partial B_j}{\partial \xi^i} \right). \quad (2-28)$$

The expressions for the line, area and volume integrals of a function  $\mathbf{B}(\xi^1, \xi^2, \xi^3)$  in general coordinates are:

Line integral over  $\xi^1$ , with  $\xi^2$  and  $\xi^3$  treated as constants

$$\int \mathbf{B} \cdot d\mathbf{x} = \int \mathbf{B} \cdot \left( \frac{\partial \mathbf{x}}{\partial \xi^1} \right) d\xi^1, \quad (2-29)$$

surface integral over  $\xi^2$  and  $\xi^3$  ( $\xi^1 = \text{const}$ )

$$\int \mathbf{B} \cdot d\mathbf{a} = \int \mathbf{B} \cdot \left( \frac{\partial \mathbf{x}}{\partial \xi^2} \times \frac{\partial \mathbf{x}}{\partial \xi^3} \right) d\xi^2 d\xi^3 = \int (\mathbf{B} \cdot \nabla \xi^1) J d\xi^2 d\xi^3, \quad (2-30)$$

and the volume integral over  $\xi^1$ ,  $\xi^2$  and  $\xi^3$

$$\int \mathbf{B} d\mathbf{v} = \int \mathbf{B} \frac{\partial \mathbf{x}}{\partial \xi^1} \cdot \left( \frac{\partial \mathbf{x}}{\partial \xi^2} \times \frac{\partial \mathbf{x}}{\partial \xi^3} \right) = \int \mathbf{B} J d\xi^1 d\xi^2 d\xi^3. \quad (2-31)$$

The Jacobian between the Cartesian coordinates  $x^1, x^2, x^3$  (where  $x^1 = x, x^2 = y, x^3 = z$ ) and the general coordinates  $\xi^1, \xi^2, \xi^3$  is defined as the determinant of the matrix

$$\begin{bmatrix} \partial x^1 / \partial \xi^1 & \partial x^1 / \partial \xi^2 & \partial x^1 / \partial \xi^3 \\ \partial x^2 / \partial \xi^1 & \partial x^2 / \partial \xi^2 & \partial x^2 / \partial \xi^3 \\ \partial x^3 / \partial \xi^1 & \partial x^3 / \partial \xi^2 & \partial x^3 / \partial \xi^3 \end{bmatrix} \quad (2-32)$$

Introducing the Jacobian notation, the determinant of the matrix can be written as

$$J = \frac{\partial(x^1, x^2, x^3)}{\partial(\xi^1, \xi^2, \xi^3)}. \quad (2-33)$$

The Jacobian notation can also be used to write partial derivatives. The partial derivative of  $f$  with respect to  $\xi^1$ , holding  $\xi^2$  and  $\xi^3$  constant, can be written in Jacobian notation as

$$\left( \frac{\partial f}{\partial \xi^1} \right)_{\xi^2, \xi^3} = \frac{\partial(f, \xi^2, \xi^3)}{\partial(\xi^1, \xi^2, \xi^3)}. \quad (2-34)$$

Some useful properties of the Jacobian notation are

$$\frac{\partial(x^1, x^2, x^3)}{\partial(\eta^1, \eta^2, \eta^3)} = \frac{\partial(x^1, x^2, x^3)}{\partial(\xi^1, \xi^2, \xi^3)} \frac{\partial(\xi^1, \xi^2, \xi^3)}{\partial(\eta^1, \eta^2, \eta^3)} \quad (2-35)$$

and

$$\frac{\partial(f, \eta^2, \eta^3)}{\partial(\eta^1, \eta^2, \eta^3)} = 0 \quad \text{and} \quad \frac{\partial(f, \eta^2, \eta^3)}{\partial(\xi^1, \xi^2, \eta^3)} = \left[ \frac{\partial(f, \eta^2)}{\partial(\xi^1, \xi^2)} \right]_{\eta^3}. \quad (2-36)$$

We will use the Jacobian notation and the above described properties extensively in section 2.4

## 2.3 The Tokamak

As seen in subsection 2.1.1, straight (open-ended) geometry does not allow satisfactory confinement of particles because of the loss of passing particles. Another disadvantage of open-ended confinement is the susceptibility to the so-called micro-instabilities [cf. section 3.12]. Those disadvantages convinced scientists that the only devices which would lead to reasonable confinement of the plasma must contain geometry in which the field lines form bounded surfaces. The only geometrical arrangement exhibiting this property is a torus. The most widely used toroidal confinement device is the tokamak.. Figure 2-7 shows a picture of the world's largest tokamak, JET. The word "tokamak" is a Russian acronym for *Toroidalna Kamera Magnitnaya*, or "Toroidal magnetic chamber". Despite the tokamaks' advantages over open-ended confinement devices, there are disadvantages associated with tokamaks due to the more complicated geometry. We will discuss some of those disadvantages below.

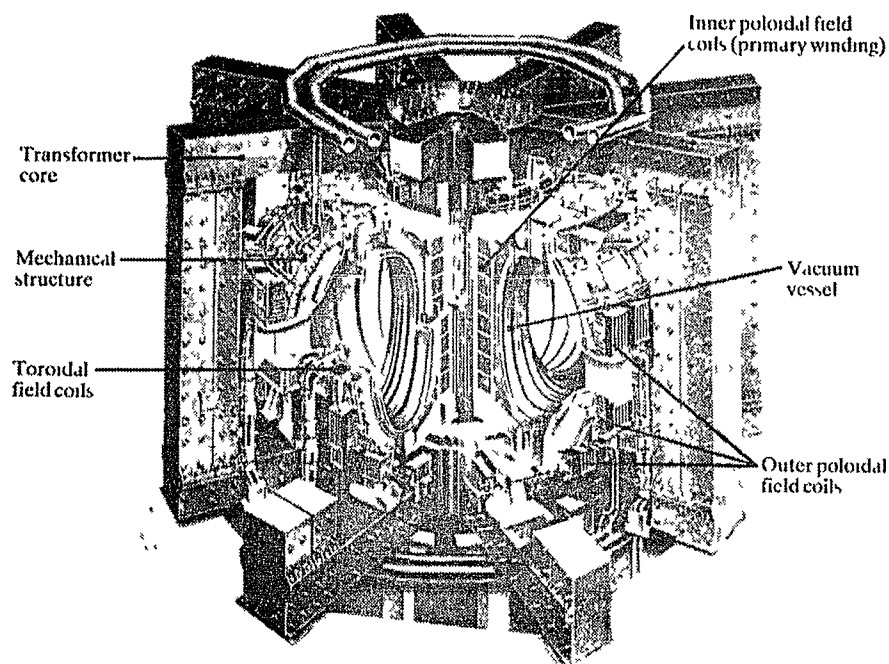


Figure 2-7: A picture of the *Joint European Torus* (JET) located in the Culham Laboratories in Abingdon, U.K. JET came on line in June of 1983 as a collaboration of several European countries operating under the auspices of EURATOM. (This image was scanned from J. Wesson's book *Tokamaks* [3]).

### 2.3.1 Tokamak Equilibrium

The condition for plasma equilibrium requires the force due to the plasma pressure  $p$  to be balanced by the force due to the interaction of the plasma current with the magnetic field  $\mathbf{B}$

$$\nabla p = \mathbf{j} \times \mathbf{B}. \quad (2-37)$$

This expression is referred to as the force balance equation, but we notice that the quantities that are balanced are the forces per unit volume, with  $j$  the current density. Equation (2-37) together with Ampère's Law, which relates the current density and the

magnetic field,

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{j} \quad (2-38)$$

and the condition that the magnetic field be divergence free

$$\nabla \cdot \mathbf{B} = 0 \quad (2-39)$$

form the basis of the physics of plasma equilibrium.

The magnetic field in a tokamak is composed of two, axisymmetric, components. The toroidal field (the field along the long, or  $\phi$ , direction) is produced by external magnetic coils wound around the torus. From subsection 2.1.2 we see that the toroidal field only is not able to confine the plasma because of the charge separation produced by the curvature and gradient  $B$  drifts. In addition to the unfavorable up-down drifts of the ions and electrons away from each other, the confinement is further weakened by the outward  $\mathbf{E} \times \mathbf{B}$  drift due to charge separation (Fig. 2-8). The solution to this problem is the poloidal magnetic field (along the short, or  $\theta$ , direction) induced by the toroidal plasma current. The magnetic field lines due to the superimposition of the toroidal and the much weaker poloidal field spiral slowly around the torus. The equation of a field line is given by

$$\frac{r d\theta}{R d\phi} = \frac{B_\theta}{B_\phi}. \quad (2-40)$$

The rate of precession (angle of rotation on a poloidal cross section during one toroidal orbit divided by  $2\pi$ ) of the field lines is given by the rotational transform  $\iota$ .

$$\iota = \frac{d\theta}{d\phi} = \frac{R B_\theta}{r B_\phi} \approx \frac{R_0 B_\theta}{r_a B_0}, \quad (2-41)$$

with  $R_0$  the major radius,  $r_a$  the minor radius, and  $B_0$  the surface averaged value of the toroidal field strength. The inverse of  $\iota$  is known as the safety factor (or winding number)  $q$ . If  $\iota$  is an irrational number, the magnetic field lines form nested surfaces of constant  $r$  (Fig. 2-9).

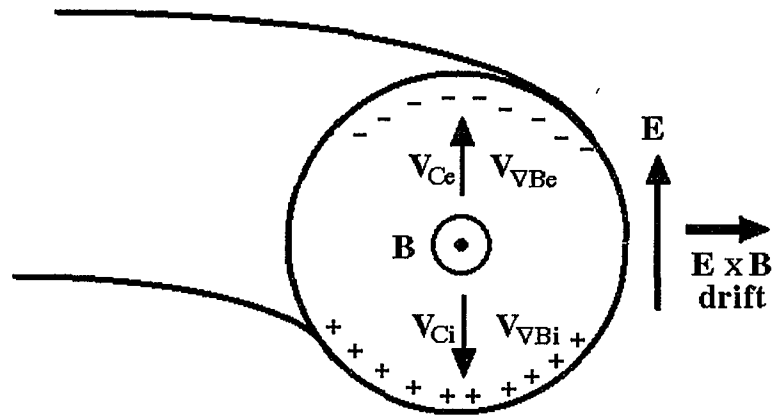


Figure 2-8: The curvature and gradient  $B$  drifts produce up-down charge separation inducing an electric field which produces the outward  $\mathbf{E} \times \mathbf{B}$  drift.

The speed of sound in fusion plasmas is typically  $\sim 10^6$  m/sec therefore there is rapid pressure equalization along the magnetic field lines and the nested surfaces are constant pressure surfaces. This important result is obtained quantitatively from the scalar product of  $\mathbf{B}$  with Eq. (2-37)

$$\mathbf{B} \cdot \nabla p = 0. \quad (2-42)$$

which states that there is no pressure gradient along the magnetic surfaces, hence the magnetic surfaces are surfaces of constant pressure. Equation (2-37) also gives

$$\mathbf{B} \cdot \nabla j = 0 \quad (2-43)$$

which means that the current lines also lie in the magnetic surfaces.



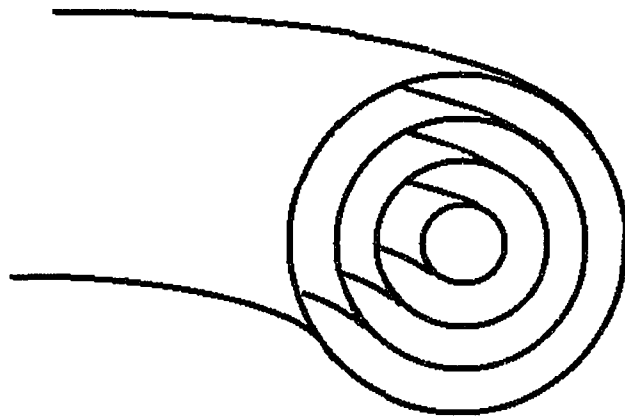


Figure 2-9: The magnetic flux surfaces form nested toroids.

### 2.3.2 Magnetic Coordinates

Magnetic coordinates can be found for a plasma in which there exist magnetic surfaces. Hamiltonian theory guarantees the existence of magnetic surfaces in devices with a symmetry direction such as an ideal, axisymmetric tokamak or a helically symmetric stellarator. Magnetic surfaces do not exist everywhere in non-symmetric devices because of the existence of ergodic regions of non-zero volume. If the perturbations from symmetry are small, however, those devices can be studied by considering approximate magnetic surfaces. The precise condition for the existence of magnetic coordinates is the existence of a function  $f(\mathbf{x})$ , such that

$$\mathbf{B} \cdot \nabla f(\mathbf{x}) = 0 \quad (2-44)$$

with  $|\nabla f| > 0$  satisfied everywhere except along isolated curves. Comparing Eq. (2-44) with Eq. (2-42) we see that magnetic coordinates exist if an equilibrium plasma

has a pressure gradient. Figure 2-10 defines the magnetic coordinates consisting of the toroidal and poloidal angles ( $\phi$  and  $\theta$ ) and the toroidal and poloidal fluxes ( $\psi$  and  $-\chi$ ). The expressions for the toroidal and poloidal currents ( $I$  and  $G$ ) are also given. The derivations of the fluxes and currents will be presented following the discussion of the contravariant and covariant representations of the magnetic field.

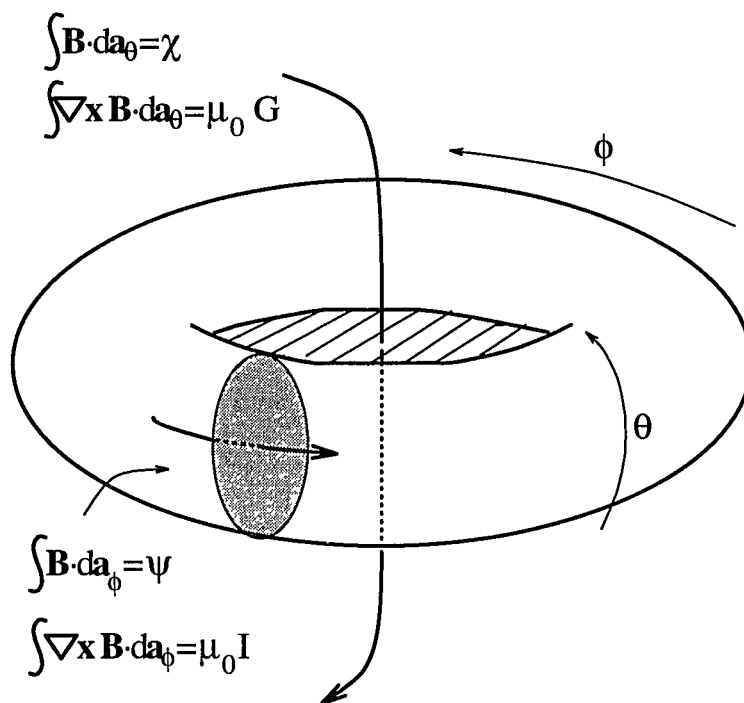


Figure 2-10: Definitions of canonical coordinates used to describe a tokamak. The poloidal and toroidal angles  $\theta$  and  $\phi$  and the poloidal and toroidal magnetic fluxes  $-\chi$  and  $\psi$  are defined. The definitions of the toroidal and poloidal currents  $G$  and  $I$  are also given.

### Contravariant Representation

The contravariant representation for the magnetic field is based on the fact that any

divergence-free vector can be represented by four well behaved functions of position. This is done by choosing an appropriate gauge function in the form of the vector potential associated with the divergence-free vector. Two of the four functions are arbitrarily specified functions of position. We shall take these functions to be the toroidal and poloidal angles ( $\phi$  and  $\theta$ ). The vector potential in toroidal coordinates (with  $r$  the radial coordinate) is

$$\mathbf{A}(r, \theta, \phi) = A_r(r, \theta, \phi)\nabla r + A_\theta(r, \theta, \phi)\nabla\theta + A_\phi(r, \theta, \phi)\nabla\phi. \quad (2-45)$$

Introducing a gauge function  $G(r, \theta, \phi)$  which satisfies  $\partial G/\partial r = A_r$ , the vector potential can be written as

$$\mathbf{A}(r, \theta, \phi) = \psi\nabla\theta - \chi\nabla\phi + \nabla G \quad (2-46)$$

with  $2\pi\psi \equiv A_\theta - \partial G/\partial\theta$  and  $-2\pi\chi \equiv A_\phi - \partial G/\partial\phi$ . The contravariant expression for the magnetic field is then

$$\mathbf{B} = \frac{1}{2\pi} \left[ \nabla\psi(r, \theta, \phi) \times \nabla\theta + \nabla\phi \times \nabla\chi(r, \theta, \phi) \right]. \quad (2-47)$$

Defining  $\psi$  as the radial variable, we arrive at the contravariant representation of the magnetic field

$$\mathbf{B} = \frac{1}{2\pi} \left[ \nabla\psi \times \nabla\theta + \nabla\phi \times \nabla\chi(\psi, \theta, \phi) \right]. \quad (2-48)$$

The contravariant representation for the magnetic field can be further simplified by giving up the freedom of one of the angles to write  $\chi = \chi(\psi)$  as required in a scalar pressure equilibrium [cf. Eq. (2-42)],

$$\mathbf{B} \cdot \nabla p(\psi) = \frac{1}{2\pi} \left[ \nabla\psi \times \nabla\theta + \nabla\phi \times \nabla\chi(\psi, \theta, \phi) \right] \cdot \nabla\psi \frac{\partial p}{\partial\psi} = 0. \quad (2-49)$$

The poloidal flux  $-\chi$  must be a function of  $\psi$  alone and the contravariant expression for the magnetic field becomes

$$\mathbf{B} = \frac{1}{2\pi} \left[ \nabla\psi \times \nabla\theta + \nabla\phi \times \nabla\chi(\psi) \right]. \quad (2-50)$$

### Covariant Representation

The covariant, or Boozer, representation of the magnetic field is based on the fact that the pressure is a function of  $\psi$  alone and the toroidal flux  $\psi$  satisfies the equations  $\mathbf{B} \cdot \nabla\psi = 0$  and  $\mathbf{j} \cdot \nabla\psi = 0$  [cf. Eqs.(2-42) and (2-50)]. In general, a vector can be represented using the magnetic coordinates  $(\psi, \theta, \phi)$  in the covariant form

$$\mathbf{B} = a_1 \nabla\psi + a_2 \nabla\theta + a_3 \nabla\phi \quad (2-51)$$

Using  $(\nabla \times \mathbf{B}) \cdot \psi = 0$  and writing

$$\frac{a_3}{\mu_0} = G(\psi') + (G(\psi') + \iota I(\psi')) \frac{\partial \nu(\psi', \theta, \phi)}{\partial \phi} \quad (2-52)$$

$$\frac{a_2}{\mu_0} = I(\psi') + (G(\psi') + \iota I(\psi')) \frac{\partial \nu(\psi', \theta, \phi)}{\partial \theta} \quad (2-53)$$

the covariant form of the magnetic field becomes

$$\mathbf{B} = \frac{\mu_0}{2\pi} [G(\psi') \nabla\phi' + I(\psi') \nabla\theta' + \beta_*(\psi', \theta', \phi') \nabla\psi'], \quad (2-54)$$

with  $(\psi', \theta', \phi')$  the Boozer coordinates related to the original magnetic coordinates through

$$\psi' = \psi \quad (2-55)$$

$$\theta' = \theta + \iota \nu(\psi', \theta, \phi) \quad (2-56)$$

$$\phi' = \phi + \nu(\psi', \theta, \phi). \quad (2-57)$$

The identifications of  $G$  and  $I$  with the toroidal and poloidal currents will be derived in the next subsection.

### 2.3.3 Currents and Fluxes

#### Poloidal Current

The poloidal current  $G(\psi)$  can be obtained from the following short calculation:

$$\frac{1}{\mu_0} \int (\nabla \times \mathbf{B}) \cdot d\mathbf{a}_\theta = \frac{1}{\mu_0} \oint \mathbf{B} \cdot d\mathbf{l}_\phi, \quad (2-58)$$

which is the Stoke's theorem applied to the integration over the poloidal surface. Using the covariant representation of the magnetic field (2-54) and writing  $d\mathbf{l}_\phi = \frac{\partial \mathbf{x}}{\partial \phi} d\phi$ , we arrive at

$$\frac{1}{\mu_0} \int (\nabla \times \mathbf{B}) \cdot d\mathbf{a}_\theta = G(\psi) \quad (2-59)$$

which corresponds to the expression for the poloidal current given in Fig. 2-10.

We can obtain an approximate relationship between the poloidal current and the toroidal magnetic field in terms of a characteristic length of the system. To do so, we approximate the toroidal field strength by  $B_0$ , the surface-averaged toroidal field strength. Substituting into the left hand side of Eq. 2-58 yields

$$\oint \mathbf{B} \cdot d\mathbf{l}_\phi = \oint B_\phi d\phi \approx \oint B_0 d\phi \quad (2-60)$$

with  $B_0 = \text{const}$  over the range of integration. Together with Eq. 2-59 this gives

$$B_0(2\pi R_0) = \mu_0 G_0 \quad (2-61)$$

with  $R_0$  the major radius. The characteristic length of the system is then

$$2\pi R_0 = \frac{\mu_0 G_0}{B_0}. \quad (2-62)$$

### Toroidal Current

The expression for the toroidal current  $I(\psi)$  can be obtained in an analogous manner to the calculation for the poloidal current,

$$\frac{1}{\mu_0} \int (\nabla \times \mathbf{B}) \cdot d\mathbf{a}_\phi = I(\psi). \quad (2-63)$$

This is the expression for the toroidal current given in Fig. 2-10.

We can obtain an approximate relationship between the toroidal and poloidal currents by applying Stoke's theorem to the left hand side of Eq. (2-63) and approximating the poloidal field  $B_\theta$  by  $\iota(r/R_0)B_0$ , which yields

$$2\pi R_0 \epsilon^2 \iota = \frac{\mu_0 I(\psi)}{B_0}, \quad (2-64)$$

with  $\epsilon = r/R$ , the inverse aspect ratio. Using Eq. (2-62), we find that

$$I \approx \epsilon^2 \iota G \quad (2-65)$$

which means that for large aspect ratios, the toroidal current  $I$  is much smaller than the poloidal current  $G$  and can be neglected.

### Toroidal Flux

A similar calculation to the ones performed to calculate the currents can be performed to obtain the toroidal flux.

$$\int \mathbf{B} \cdot d\mathbf{a}_\phi = \oint d\theta \int_0^\psi d\psi' J \mathbf{B} \cdot \nabla \phi \quad (2-66)$$

Using (2-22) for the Jacobian and the contravariant representation for the magnetic field [Eq. (2-50)], we obtain

$$\int \mathbf{B} \cdot d\mathbf{a}_\phi = \frac{1}{2\pi} \oint d\theta \int_0^\psi d\psi' \frac{(\nabla\psi \times \nabla\theta + \nabla\phi \times \nabla\chi) \cdot \nabla\phi}{(\nabla\psi \times \nabla\theta) \cdot \nabla\phi} \quad (2-67)$$

$$\int \mathbf{B} \cdot d\mathbf{a}_\phi = \frac{1}{2\pi} \int_0^{2\pi} d\theta \int_0^\psi d\psi' \quad (2-68)$$

$$\int \mathbf{B} \cdot d\mathbf{a}_\phi = \psi. \quad (2-69)$$

In particular, the toroidal flux at the edge of the plasma,  $\psi_a$ , in terms of the surface averaged toroidal field strength,  $B_0$ , can be obtained from

$$\int \mathbf{B} \cdot d\mathbf{a}_\phi \approx B_0(\pi r^2) = \psi_a. \quad (2-70)$$

### Poloidal Flux

Through an analogous calculation to the one performed for the toroidal flux, the poloidal flux  $-\chi(\psi)$  is shown to be

$$\int \mathbf{B} \cdot d\mathbf{a}_\theta = \chi(\psi). \quad (2-71)$$

The toroidal and poloidal fluxes are related through the rotational transform  $\iota$  by

$$\frac{d\chi(\psi)}{d\psi} \equiv \iota(\psi). \quad (2-72)$$

To prove Eq. (2-72) let us consider a small radial annulus of width  $dr$ . The toroidal flux through the annulus is given by (2-69) as

$$d\psi = B_\phi(2\pi r)dr \quad (2-73)$$

and the poloidal flux through the annulus can be written as

$$d\chi(\psi) = B_\theta(2\pi R)dr. \quad (2-74)$$

Dividing Eq. (2-74) by Eq. (2-73) and remembering the equation for the rotational transform [Eq. (2-41)],

$$t = \frac{RB_\theta}{rB_\phi}, \quad (2-75)$$

gives the relationship between the fluxes.

The toroidal flux  $\psi$  is defined to be zero on the magnetic axis, which means that  $\psi$  is the magnetic flux *enclosed* by a magnetic pressure surface. The poloidal flux,  $-\chi$ , is defined to be the flux *outside* of a pressure surface, which means it is the flux through the hole of the torus.

### 2.3.4 Particle Trajectories in a Tokamak

To lowest order in  $r_a/R_0$ , the minor radius divided by the major radius, the magnetic field in a tokamak, is given by

$$B = B_0 \left( 1 - \frac{r}{R_0} \cos \theta \right). \quad (2-76)$$

Let us recall that the radial coordinate  $r$  represents a toroidal flux surface  $\psi$ . This form of the magnetic field strength and the expression for the particle energy [Eq. (2-10)] leads to magnetic mirroring of particles that satisfy the trapped particle criterion of Eq. (2-12). The motion of the trapped particles can be projected onto a constant  $\phi$  plane (Fig. 2-11) and because of the axisymmetry of the  $B$ -field, the toroidal motion does not affect this projection. The magnetic field is weaker on the outside of the tokamak ( $\theta = 0$  side)

than on the inside, leading the particles to execute the characteristic “banana orbits” (named for obvious reasons). Note that Fig. 2-11 is a projection of the guiding center motion of the particle rather than a projection of the gyromotion.

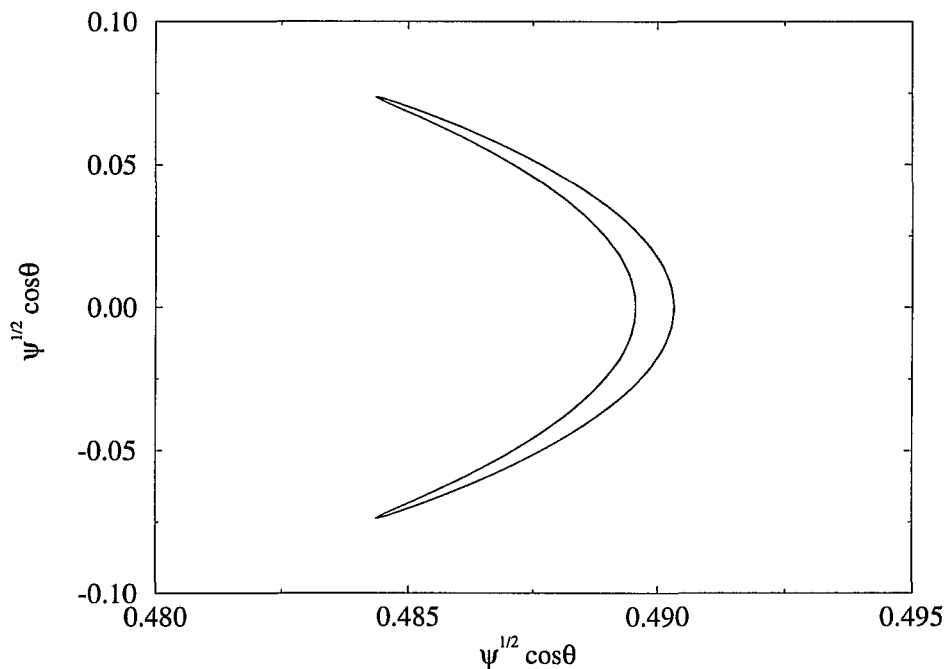


Figure 2-11: Projection of the guiding center motion for a deeply trapped particle in a tokamak magnetic field.

The passing particles circulate around the torus. The parallel velocity of the passing particles is modified by local magnetic field strength variations, but it does not change sign. A characteristic width of the banana can be obtained from the expression for the canonical toroidal momentum [Eq. 2-89]. Remembering that in an axisymmetric configuration  $p_\phi$  is a constant of motion, we have

$$\Delta\psi \sim \frac{\Delta v_{\parallel}}{\nu\omega_c}. \quad (2-77)$$

The banana width is greater for the trapped particles because the parallel velocity



vanishes at the bounce point of the orbit. For the passing particles, the parallel velocity fluctuates, but does not vanish anywhere on the orbit. We can write,  $\Delta v_{\parallel} \sim \epsilon v_{\parallel}$ , for the passing particles, which gives

$$(\Delta\psi)_{\text{passing}} \sim \frac{\rho_L}{\iota}. \quad (2-78)$$

For trapped particles, we have  $\Delta v_{\parallel} \sim \epsilon^{1/2} v$ , which gives

$$(\Delta\psi)_{\text{trapped}} \sim \frac{\rho_L}{\iota\epsilon^{1/2}}. \quad (2-79)$$

The final feature of the banana orbits we will mention is the toroidal precession of the banana orbits. The reflection of the trapped particles occurs at the same critical poloidal angle, but the trapped particles experience a cumulative cross-field drift as well as an oscillatory radial drift. This causes an advance of the mirror reflections around the torus leading to a toroidal precession of the banana orbit.

## 2.4 The Particle Equations of Motion

Let us now turn to obtaining the Hamilton equations of motion for a charged particle in a magnetic field. We will first find the poloidal and toroidal canonical momenta from the drift Lagrangian and then obtain the drift Hamiltonian. The analysis will be done in two sets of variables,  $(\theta, \phi, p_{\theta}, p_{\phi})$  and  $(\theta, \phi, \psi, \rho_{\parallel})$ .

### First Set of Variables — $(\theta, \phi, p_{\theta}, p_{\phi})$

The drift Lagrangian for a charged particle in a magnetic field is

$$L_d = \frac{1}{2} m v_{\parallel}^2 + q \mathbf{v} \cdot \mathbf{A}(\mathbf{x}) - \mu B(\psi, \theta, \phi) - q \Phi(\psi, \theta, \phi), \quad (2-80)$$

where  $\mu$  is the magnetic moment,  $q$  is the charge of the particle and  $\mathbf{A}(\mathbf{x})$  is the vector potential of the magnetic field  $\mathbf{B}$ . The magnetic field associated with plasma equilibrium,  $\nabla p = \mathbf{j} \times \mathbf{B}$ , has a covariant representation in Boozer coordinates  $(\psi, \theta, \phi)$

[Eq. (2-54)]

$$\mathbf{B} = \frac{\mu_0}{2\pi} [G(\psi)\nabla\phi + I(\psi)\nabla\theta + \beta_*(\psi, \theta, \phi)\nabla\psi], \quad (2-81)$$

where  $\mu_0$  is the permeability of free space,  $G(\psi)$  and  $I(\psi)$  are the poloidal and toroidal currents and  $\beta_*$  is closely related to the Pfirsch–Schlüter current. The angles  $\theta$  and  $\phi$  are assumed to have period  $2\pi$ . In a vacuum field, also called a curl-free field,  $(\nabla \times \mathbf{B})_\phi = 0$  and the toroidal plasma current vanishes. The covariant representation simplifies to

$$\mathbf{B} = \frac{\mu_0}{2\pi} [G_0\nabla\phi]. \quad (2-82)$$

The covariant representation for the vector potential is

$$\mathbf{A}(\mathbf{x}) = \frac{1}{2\pi} [\psi\nabla\theta - \chi(\psi)\nabla\phi]. \quad (2-83)$$

The velocity,  $\mathbf{v}$ , can be written as

$$\mathbf{v} = \dot{\psi} \frac{\partial \mathbf{x}}{\partial \psi} + \dot{\theta} \frac{\partial \mathbf{x}}{\partial \theta} + \dot{\phi} \frac{\partial \mathbf{x}}{\partial \phi}. \quad (2-84)$$

Then, using the orthogonality relations of general coordinates,

$$v_{\parallel} = \frac{\mathbf{v} \cdot \mathbf{B}}{B} = \frac{1}{B} \left[ \dot{\psi} \frac{\partial \mathbf{x}}{\partial \psi} + \dot{\theta} \frac{\partial \mathbf{x}}{\partial \theta} + \dot{\phi} \frac{\partial \mathbf{x}}{\partial \phi} \right] \cdot \frac{\mu_0}{2\pi} [G_0(\psi)\nabla\phi] = \frac{1}{2\pi} \frac{\mu_0 G}{B} \dot{\phi} \quad (2-85)$$

$$\mathbf{v} \cdot \mathbf{A}(\mathbf{x}) = \frac{1}{2\pi} \left[ \dot{\psi} \frac{\partial \mathbf{x}}{\partial \psi} + \dot{\theta} \frac{\partial \mathbf{x}}{\partial \theta} + \dot{\phi} \frac{\partial \mathbf{x}}{\partial \phi} \right] \cdot [\psi\nabla\theta - \chi(\psi)\nabla\phi] = \frac{1}{2\pi} [\psi\dot{\theta} - \chi(\psi)\dot{\phi}]. \quad (2-86)$$

The drift Lagrangian then becomes

$$L_d = \frac{1}{2} m \left( \frac{1}{2\pi} \frac{\mu_0 G_0}{B} \dot{\phi} \right)^2 + \frac{q}{2\pi} (\psi\dot{\theta} - \chi(\psi)\dot{\phi}) - \mu B - q\Phi \quad (2-87)$$

and the canonical momenta are

$$p_\theta = \frac{\partial L_d}{\partial \dot{\theta}} = \frac{q\psi}{2\pi} \quad (2-88)$$

$$p_\phi = \frac{\partial L_d}{\partial \dot{\phi}} = m \left( \frac{1}{2\pi} \frac{\mu_0 G_0}{B} \right)^2 \dot{\phi} - \frac{q\chi}{2\pi}. \quad (2-89)$$

To obtain the drift Hamiltonian,  $H$ , from the drift Lagrangian,  $L_d$ , we use

$$H = \dot{\theta}p_\theta + \dot{\phi}p_\phi - L_d. \quad (2-90)$$

Then

$$\begin{aligned} H &= \dot{\theta} \frac{q\psi}{2\pi} + \dot{\phi} \left[ m \left( \frac{1}{2\pi} \frac{\mu_0 G_0}{B} \right)^2 \dot{\phi} - \frac{q\chi}{2\pi} \right] \\ &\quad - \left[ \frac{1}{2} m \left( \frac{1}{2\pi} \frac{\mu_0 G_0}{B} \dot{\phi} \right)^2 + \frac{q}{2\pi} (\psi \dot{\theta} - \chi \dot{\phi}) - \mu B - q\Phi \right] \end{aligned} \quad (2-91)$$

$$H = \frac{(2\pi)^2}{2m} (p_\phi + \iota p_\theta)^2 \left( \frac{B}{\mu_0 G_0} \right)^2 + \mu B + q\Phi \quad (2-92)$$

with  $\iota = \partial\chi/\partial\psi \approx \chi/\psi$  the rotational transform. We can now use the Hamilton equations of motion to find the orbit equations.

$$\begin{aligned} \dot{\theta} &= \frac{\partial H}{\partial p_\theta} = \iota \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta) \left( \frac{B}{\mu_0 G_0} \right)^2 \\ &\quad + \left[ \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta)^2 \left( \frac{B}{\mu_0 G_0} \right)^2 + \mu B \right] \frac{1}{B} \left( \frac{2\pi}{q} \right) \frac{\partial B}{\partial \psi} \end{aligned} \quad (2-93)$$

$$\dot{\phi} = \frac{\partial H}{\partial p_\phi} = \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta) \left( \frac{B}{\mu_0 G_0} \right)^2 \quad (2-94)$$

$$\dot{p}_\theta = -\frac{\partial H}{\partial \theta} = -\left[ \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta)^2 \left( \frac{B}{\mu_0 G_0} \right)^2 + \mu B \right] \frac{1}{B} \frac{\partial B}{\partial \theta} \quad (2-95)$$

$$\dot{p}_\phi = -\frac{\partial H}{\partial \phi} = -\left[ \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta)^2 \left( \frac{B}{\mu_0 G_0} \right)^2 + \mu B \right] \frac{1}{B} \frac{\partial B}{\partial \phi}. \quad (2-96)$$

The approach of canonical variables and momenta is the most intuitive and straight forward approach to particle motion. We must not fail to notice, however, that the reason we were able to write the equations of motion in a simple form was the simple relationship between the toroidal flux and the canonical poloidal momentum [Eq. (2-88)] due to the curl free-field. In general, the relationship between  $\psi$  and  $p_\theta$  is more complicated [cf. Eq. (2-100)].

### Second Set of Variables — $(\theta, \phi, \psi, \rho_{\parallel})$

We will now discuss another set of variables  $(\theta, \phi, \psi, \rho_{\parallel})$  which can be used to describe particle motion. Let us go back to Eq. (2-85). Without the assumption of a curl-free field, the parallel velocity is

$$v_{\parallel} = \frac{\mu_0}{2\pi B} [G(\psi)\dot{\phi} + I(\psi)\dot{\theta} + \beta_*(\psi, \theta, \phi)\dot{\psi}]. \quad (2-97)$$

It can be shown that when this expression for  $v_{\parallel}$  is used in the guiding center Lagrangian,  $p_{\psi}$  and  $\psi$  can be determined once  $\theta, \phi, p_{\theta}, p_{\phi}$  are given. The Hamiltonian, therefore, depends only on four variables  $(\theta, \phi, p_{\theta}, p_{\phi})$ , but the Lagrangian depends on six  $(\theta, \phi, \psi, \dot{\theta}, \dot{\phi}, \dot{\psi})$ . The Hamilton equations of motion can only be obtained when the Lagrangian depends on the same number of variables as the associated Hamiltonian. This is usually ensured by choosing an appropriate gauge function. In our case, the problem can be solved by realizing that  $\dot{\psi}$  is of order gyroradius to system size  $\rho/R$  smaller than  $\dot{\theta}$  and  $\dot{\phi}$ . We can redefine the parallel velocity as

$$v_{\parallel}' = \frac{\mu_0}{2\pi B} [G(\psi)\dot{\phi} + I(\psi)\dot{\theta}]. \quad (2-98)$$

with the difference between  $v_{\parallel}$  and  $v_{\parallel}'$  only of order  $\rho/R \ll 1$ . The drift Lagrangian now becomes

$$L_d = \frac{1}{2} m \frac{\mu_0}{2\pi B} [G(\psi)\dot{\phi} + I(\psi)\dot{\theta}]^2 + \frac{q}{2\pi} (\psi\dot{\theta} - \chi(\psi)\dot{\phi}) - \mu B - q\Phi \quad (2-99)$$

and the canonical momenta are

$$p_{\theta} = \frac{mv_{\parallel}'}{2\pi B} \mu_0 I + \frac{q\psi}{2\pi} = \frac{q}{2\pi} (\rho_{\parallel} i + \psi) \quad (2-100)$$

$$p_{\phi} = \frac{mv_{\parallel}'}{2\pi B} \mu_0 G - \frac{q\chi}{2\pi} = \frac{q}{2\pi} (\rho_{\parallel} g - \chi), \quad (2-101)$$

with  $\rho_{\parallel} = v_{\parallel}'/\omega_c = mv_{\parallel}'/qB$  and  $g(\psi) = \mu_0 G(\psi)$ ,  $i(\psi) = \mu_0 I(\psi)$ . The drift Hamiltonian in terms of the new variables is

$$H = \frac{1}{2} m v_{\parallel}'^2 + \mu B + q\Phi = \frac{q^2 B^2}{2m} \rho_{\parallel}^2 + \mu B + q\Phi. \quad (2-102)$$

To obtain the time evolution equations for  $\theta$ ,  $\phi$ ,  $\rho_{\parallel}$ ,  $\psi$  we perform a coordinate transformation from  $(\theta, \phi, p_{\theta}, p_{\phi})$  to the new coordinate system  $(\theta, \phi, \rho_{\parallel}, \psi)$ . The Jacobian of this coordinate transformation is given by

$$J = \frac{\partial(\theta, \phi, p_{\phi}, p_{\theta})}{\partial(\theta, \phi, \rho_{\parallel}, \psi)} = \left[ \frac{\partial(p_{\phi}, p_{\theta})}{\partial(\rho_{\parallel}, \psi)} \right]_{\theta, \phi} = \left[ \left( \frac{\partial p_{\phi}}{\partial \rho_{\parallel}} \right)_{\psi} \left( \frac{\partial p_{\theta}}{\partial \psi} \right)_{\rho_{\parallel}} - \left( \frac{\partial p_{\theta}}{\partial \rho_{\parallel}} \right)_{\psi} \left( \frac{\partial p_{\phi}}{\partial \psi} \right)_{\rho_{\parallel}} \right]_{\theta, \phi} \quad (2-103)$$

Using Eqs. (2-100), (2-101) and

$$\left( \frac{\partial p_{\phi}}{\partial \psi} \right)_{\theta, \phi, \rho_{\parallel}} = \left( \frac{\partial p_{\phi}}{\partial \chi} \right)_{\theta, \phi, \rho_{\parallel}} \left( \frac{\partial p_{\chi}}{\partial \psi} \right)_{\theta, \phi, \rho_{\parallel}} = \iota \left( \frac{\partial p_{\phi}}{\partial \chi} \right)_{\theta, \phi, \rho_{\parallel}} \quad (2-104)$$

the Jacobian becomes

$$J = \left( \frac{q}{2\pi} g \right) \left( \frac{q}{2\pi} \right) - \left( \frac{q}{2\pi} i \right) \left( -\frac{q}{2\pi} \iota \right) = \left( \frac{q}{2\pi} \right)^2 (g + \iota i). \quad (2-105)$$

The time derivatives of the new variables  $(\theta, \phi, \rho_{\parallel}, \psi)$  in terms of  $(\theta, \phi, \rho_{\parallel}, \psi)$  can be obtained from the Hamilton equations of motion for the canonical variables and momenta with the help of some Jacobian algebra [cf. section 2.2]. The expressions for  $\dot{\theta}$  and  $\dot{\phi}$  are obtained quite trivially,

$$\begin{aligned} \dot{\theta} &= \left( \frac{\partial H}{\partial p_{\theta}} \right)_{p_{\phi}, \theta, \phi} = \frac{\partial(H, p_{\phi}, \theta, \phi)}{\partial(p_{\theta}, p_{\phi}, \theta, \phi)} = \frac{\partial(H, p_{\phi}, \theta, \phi)}{\partial(\rho_{\parallel}, \psi, \theta, \phi)} \frac{\partial(\rho_{\parallel}, \psi, \theta, \phi)}{\partial(p_{\theta}, p_{\phi}, \theta, \phi)} \\ &= -\frac{1}{J} \frac{\partial(H, p_{\phi}, \theta, \phi)}{\partial(\rho_{\parallel}, \psi, \theta, \phi)} = -\frac{1}{J} \left[ \frac{\partial(H, p_{\phi})}{\partial(\rho_{\parallel}, \psi)} \right]_{\theta, \phi} \\ &= -\frac{1}{J} \left[ \left( \frac{\partial H}{\partial \rho_{\parallel}} \right)_{\psi} \left( \frac{\partial p_{\phi}}{\partial \psi} \right)_{\rho_{\parallel}} - \left( \frac{\partial p_{\phi}}{\partial \rho_{\parallel}} \right)_{\psi} \left( \frac{\partial H}{\partial \psi} \right)_{\rho_{\parallel}} \right]_{\theta, \phi} \quad (2-106) \end{aligned}$$

$$\dot{\phi} = \left( \frac{\partial H}{\partial p_{\phi}} \right)_{p_{\theta}, \theta, \phi} = \frac{\partial(H, p_{\theta}, \theta, \phi)}{\partial(p_{\phi}, p_{\theta}, \theta, \phi)} = \frac{\partial(H, p_{\theta}, \theta, \phi)}{\partial(\psi, \rho_{\parallel}, \theta, \phi)} \frac{\partial(\psi, \rho_{\parallel}, \theta, \phi)}{\partial(p_{\phi}, p_{\theta}, \theta, \phi)}$$

$$\begin{aligned}
&= -\frac{1}{J} \frac{\partial(H, p_\theta, \theta, \phi)}{\partial(\psi, \rho_{\parallel}, \theta, \phi)} = -\frac{1}{J} \left[ \frac{\partial(H, p_\theta)}{\partial(\psi, \rho_{\parallel})} \right]_{\theta, \phi} \\
&= -\frac{1}{J} \left[ \left( \frac{\partial H}{\partial \psi} \right)_{\rho_{\parallel}} \left( \frac{\partial p_\theta}{\partial \rho_{\parallel}} \right)_\psi - \left( \frac{\partial p_\theta}{\partial \psi} \right)_{\rho_{\parallel}} \left( \frac{\partial H}{\partial \rho_{\parallel}} \right)_{\psi} \right]_{\theta, \phi} \quad (2-107)
\end{aligned}$$

To obtain the expressions for  $\dot{\rho}_{\parallel}$  and  $\dot{\psi}$  in terms of  $(\theta, \phi, \rho_{\parallel}, \psi)$  requires a little more work. We begin by writing

$$\begin{aligned}
\dot{\rho}_{\parallel} &= \left( \frac{\partial \rho_{\parallel}}{\partial \theta} \right)_{p_\theta, p_\phi, \theta} \dot{\theta} + \left( \frac{\partial \rho_{\parallel}}{\partial \phi} \right)_{p_\theta, p_\phi, \phi} \dot{\phi} \\
&\quad + \left( \frac{\partial \rho_{\parallel}}{\partial p_\theta} \right)_{p_\phi, \theta, \phi} \dot{p}_\theta + \left( \frac{\partial \rho_{\parallel}}{\partial p_\phi} \right)_{p_\theta, \theta, \phi} \dot{p}_\phi + \left( \frac{\partial \rho_{\parallel}}{\partial t} \right)_{p_\theta, p_\phi, \theta, \phi} \\
&= \left( \frac{\partial \rho_{\parallel}}{\partial p_\theta} \right)_{p_\phi, \theta, \phi} \left( -\frac{\partial H}{\partial \theta} \right)_{p_\theta, p_\phi, \phi} + \left( \frac{\partial \rho_{\parallel}}{\partial p_\phi} \right)_{p_\theta, \theta, \phi} \left( -\frac{\partial H}{\partial \phi} \right)_{p_\theta, p_\phi, \theta} + \left( \frac{\partial \rho_{\parallel}}{\partial t} \right)_{p_\theta, p_\phi, \theta, \phi} \quad (2-108)
\end{aligned}$$

The variables held constant in the partial derivatives must be  $(\theta, \phi, \rho_{\parallel}, \psi)$ .

$$\begin{aligned}
\left( \frac{\partial \rho_{\parallel}}{\partial p_\theta} \right)_{p_\phi, \theta, \phi} &= \frac{\partial(\rho_{\parallel}, p_\phi, \theta, \phi)}{\partial(p_\theta, p_\phi, \theta, \phi)} = -\frac{\partial(p_\phi, \rho_{\parallel}, \theta, \phi)}{\partial(\psi, \rho_{\parallel}, \theta, \phi)} \frac{\partial(\psi, \rho_{\parallel}, \theta, \phi)}{\partial(p_\theta, p_\phi, \theta, \phi)} \\
&= -\frac{1}{J} \left( \frac{\partial p_\phi}{\partial \psi} \right)_{\rho_{\parallel}, \theta, \phi} \quad (2-109)
\end{aligned}$$

$$\begin{aligned}
\left( \frac{\partial H}{\partial \theta} \right)_{p_\theta, p_\phi, \phi} &= \frac{\partial(H, p_\theta, p_\phi, \phi)}{\partial(\theta, p_\theta, p_\phi, \phi)} = \frac{\partial(H, p_\theta, p_\phi, \phi)}{\partial(\theta, \rho_{\parallel}, \psi, \phi)} \frac{\partial(H, \rho_{\parallel}, \psi, \phi)}{\partial(\theta, p_\theta, p_\phi, \phi)} \\
&= -\frac{1}{J} \frac{\partial(H, \rho_{\parallel}, \psi, \phi)}{\partial(\theta, \rho_{\parallel}, \psi, \phi)} \frac{\partial(H, p_\theta, p_\phi, \phi)}{\partial(H, \rho_{\parallel}, \psi, \phi)} = \left( \frac{\partial H}{\partial \theta} \right)_{\rho_{\parallel}, \psi, \phi} \quad (2-110)
\end{aligned}$$

$$\begin{aligned}
\left( \frac{\partial \rho_{\parallel}}{\partial p_\phi} \right)_{p_\theta, \theta, \phi} &= \frac{\partial(\rho_{\parallel}, p_\theta, \theta, \phi)}{\partial(p_\phi, p_\theta, \theta, \phi)} = -\frac{\partial(p_\theta, \rho_{\parallel}, \theta, \phi)}{\partial(\psi, \rho_{\parallel}, \theta, \phi)} \frac{\partial(\psi, \rho_{\parallel}, \theta, \phi)}{\partial(p_\phi, p_\theta, \theta, \phi)} \\
&= \frac{1}{J} \left( \frac{\partial p_\theta}{\partial \psi} \right)_{\rho_{\parallel}, \theta, \phi} \quad (2-111)
\end{aligned}$$

$$\begin{aligned}
\left( \frac{\partial H}{\partial \phi} \right)_{p_\theta, p_\phi, \theta} &= \frac{\partial(H, p_\theta, p_\phi, \theta)}{\partial(\phi, p_\theta, p_\phi, \theta)} = \frac{\partial(H, p_\theta, p_\phi, \theta)}{\partial(\phi, \rho_{\parallel}, \psi, \theta)} \frac{\partial(H, \rho_{\parallel}, \psi, \theta)}{\partial(\phi, p_\theta, p_\phi, \theta)} \\
&= -\frac{1}{J} \frac{\partial(H, \rho_{\parallel}, \psi, \theta)}{\partial(\phi, \rho_{\parallel}, \psi, \theta)} \frac{\partial(H, p_\theta, p_\phi, \theta)}{\partial(H, \rho_{\parallel}, \psi, \theta)} = \left( \frac{\partial H}{\partial \phi} \right)_{\rho_{\parallel}, \psi, \theta} \quad (2-112)
\end{aligned}$$

The last expression we have to cast in terms of the new variables is

$$\begin{aligned}
\left(\frac{\partial \rho_{\parallel}}{\partial t}\right)_{p_{\theta}, p_{\phi}, \theta, \phi} &= \frac{\partial(\rho_{\parallel}, p_{\theta}, p_{\phi}, \theta, \phi)}{\partial(t, p_{\theta}, p_{\phi}, \theta, \phi)} = \frac{\partial(\rho_{\parallel}, p_{\theta}, p_{\phi}, \theta, \phi)}{\partial(t, \rho_{\parallel}, \psi, \theta, \phi)} \frac{\partial(t, \rho_{\parallel}, \psi, \theta, \phi)}{\partial(t, p_{\theta}, p_{\phi}, \theta, \phi)} \\
&= -\frac{1}{J} \frac{\partial(p_{\theta}, p_{\phi}, \rho_{\parallel}, \theta, \phi)}{\partial(t, \psi, \rho_{\parallel}, \theta, \phi)} = -\frac{1}{J} \left[ \frac{\partial(p_{\theta}, p_{\phi})}{\partial(t, \psi)} \right]_{\rho_{\parallel}, \theta, \phi} \\
&= -\frac{1}{J} \left[ \left( \frac{\partial p_{\theta}}{\partial t} \right)_{\psi} \left( \frac{\partial p_{\phi}}{\partial \psi} \right)_{\psi} - \left( \frac{\partial p_{\phi}}{\partial t} \right)_{\psi} \left( \frac{\partial p_{\theta}}{\partial \psi} \right)_{\psi} \right]_{\rho_{\parallel}, \theta, \phi} \quad (2-113)
\end{aligned}$$

The only time dependence in the canonical momentum  $p_{\phi}$  comes from the loop voltage

$$V(t) = \left(\frac{\partial \chi}{\partial t}\right)_{p_{\theta}, p_{\phi}, \theta, \phi} = \left(\frac{\partial \chi}{\partial t}\right)_{\rho_{\parallel}, \psi, \theta, \phi}. \quad (2-114)$$

The poloidal canonical momentum  $p_{\theta}$  does not have a time dependence. Using Eq. (2-114) along with the equations for the canonical momenta (2-100) and (2-101), we have

$$\left(\frac{\partial \rho_{\parallel}}{\partial t}\right)_{p_{\theta}, p_{\phi}, \theta, \phi} = \frac{1}{J} \left[ -\frac{q}{2\pi} V(t) \right] \left[ \rho_{\parallel} \left( \frac{\partial i(\psi)}{\partial \psi} \right)_{t, \rho_{\parallel}, \theta, \phi} + \frac{q}{2\pi} \right] \quad (2-115)$$

Substituting Eqs. (2-109), (2-110), (2-111), (2-112) and (2-115) into Eq. (2-108) yields the desired expression for  $\rho_{\parallel}$  in terms of the new set of variables

$$\begin{aligned}
\dot{\rho}_{\parallel} &= \frac{1}{J} \left( \frac{\partial p_{\phi}}{\partial \psi} \right)_{\rho_{\parallel}, \theta, \phi} \left( \frac{\partial H}{\partial \theta} \right)_{\rho_{\parallel}, \psi, \phi} - \frac{1}{J} \left( \frac{\partial p_{\theta}}{\partial \psi} \right)_{\rho_{\parallel}, \theta, \phi} \left( \frac{\partial H}{\partial \phi} \right)_{\rho_{\parallel}, \psi, \theta} \\
&\quad - \frac{1}{J} \left[ \frac{q}{2\pi} V(t) \right] \left[ \rho_{\parallel} \left( \frac{\partial i(\psi)}{\partial \psi} \right)_{t, \rho_{\parallel}, \theta, \phi} + \frac{q}{2\pi} \right] \quad (2-116)
\end{aligned}$$

A similar calculation is required to obtain the time derivative of  $\psi$ .

$$\begin{aligned}
\dot{\psi} &= \left( \frac{\partial \psi}{\partial \theta} \right)_{p_{\theta}, p_{\phi}, \theta} \dot{\theta} + \left( \frac{\partial \psi}{\partial \phi} \right)_{p_{\theta}, p_{\phi}, \theta} \dot{\phi} \\
&\quad + \left( \frac{\partial \psi}{\partial p_{\theta}} \right)_{p_{\phi}, \theta, \phi} \dot{p}_{\theta} + \left( \frac{\partial \psi}{\partial p_{\phi}} \right)_{p_{\theta}, \theta, \phi} \dot{p}_{\phi} + \left( \frac{\partial \psi}{\partial t} \right)_{p_{\theta}, p_{\phi}, \theta, \phi} \\
&= \left( \frac{\partial \psi}{\partial p_{\theta}} \right)_{p_{\phi}, \theta, \phi} \left( -\frac{\partial H}{\partial \theta} \right)_{p_{\theta}, p_{\phi}, \theta} + \left( \frac{\partial \psi}{\partial p_{\phi}} \right)_{p_{\theta}, \theta, \phi} \left( -\frac{\partial H}{\partial \phi} \right)_{p_{\theta}, p_{\phi}, \theta} + \left( \frac{\partial \psi}{\partial t} \right)_{p_{\theta}, p_{\phi}, \theta, \phi} \quad (2-117)
\end{aligned}$$

Again, the variables held constant in the partial derivatives must be  $(\theta, \phi, \rho_{\parallel}, \psi)$ .

$$\begin{aligned} \left(\frac{\partial\psi}{\partial p_{\theta}}\right)_{p_{\phi},\theta,\phi} &= \frac{\partial(\psi, p_{\phi}, \theta, \phi)}{\partial(p_{\theta}, p_{\phi}, \theta, \phi)} = -\frac{\partial(p_{\phi}, \psi, \theta, \phi)}{\partial(\rho_{\parallel}, \psi, \theta, \phi)} \frac{\partial(\rho_{\parallel}, \psi, \theta, \phi)}{\partial(p_{\theta}, p_{\phi}, \theta, \phi)} \\ &= \frac{1}{J} \left(\frac{\partial p_{\phi}}{\partial \rho_{\parallel}}\right)_{\psi,\theta,\phi} \end{aligned} \quad (2-118)$$

$$\begin{aligned} \left(\frac{\partial H}{\partial \theta}\right)_{p_{\theta},p_{\phi},\phi} &= \frac{\partial(H, p_{\theta}, p_{\phi}, \phi)}{\partial(\theta, p_{\theta}, p_{\phi}, \phi)} = \frac{\partial(H, p_{\theta}, p_{\phi}, \phi)}{\partial(\theta, \psi, \rho_{\parallel}, \phi)} \frac{\partial(H, \psi, \rho_{\parallel}, \phi)}{\partial(\theta, p_{\theta}, p_{\phi}, \phi)} \\ &= -\frac{1}{J} \frac{\partial(H, \psi, \rho_{\parallel}, \phi)}{\partial(\theta, \psi, \rho_{\parallel}, \phi)} \frac{\partial(H, p_{\theta}, p_{\phi}, \phi)}{\partial(H, \psi, \rho_{\parallel}, \phi)} = \left(\frac{\partial H}{\partial \theta}\right)_{\psi,\rho_{\parallel},\phi} \end{aligned} \quad (2-119)$$

$$\begin{aligned} \left(\frac{\partial\psi}{\partial p_{\phi}}\right)_{p_{\theta},\theta,\phi} &= \frac{\partial(\psi, p_{\theta}, \theta, \phi)}{\partial(p_{\phi}, p_{\theta}, \theta, \phi)} = -\frac{\partial(p_{\theta}, \psi, \theta, \phi)}{\partial(\rho_{\parallel}, \psi, \theta, \phi)} \frac{\partial(\rho_{\parallel}, \psi, \theta, \phi)}{\partial(p_{\phi}, p_{\theta}, \theta, \phi)} \\ &= -\frac{1}{J} \left(\frac{\partial p_{\theta}}{\partial \rho_{\parallel}}\right)_{\psi,\theta,\phi} \end{aligned} \quad (2-120)$$

$$\begin{aligned} \left(\frac{\partial H}{\partial \phi}\right)_{p_{\theta},p_{\phi},\theta} &= \frac{\partial(H, p_{\theta}, p_{\phi}, \theta)}{\partial(\phi, p_{\theta}, p_{\phi}, \theta)} = \frac{\partial(H, p_{\theta}, p_{\phi}, \theta)}{\partial(\phi, \psi, \rho_{\parallel}, \theta)} \frac{\partial(H, \psi, \rho_{\parallel}, \theta)}{\partial(\phi, p_{\theta}, p_{\phi}, \theta)} \\ &= -\frac{1}{J} \frac{\partial(H, \psi, \rho_{\parallel}, \theta)}{\partial(\phi, \psi, \rho_{\parallel}, \theta)} \frac{\partial(H, p_{\theta}, p_{\phi}, \theta)}{\partial(H, \psi, \rho_{\parallel}, \theta)} = \left(\frac{\partial H}{\partial \phi}\right)_{\psi,\rho_{\parallel},\theta} \end{aligned} \quad (2-121)$$

$$\begin{aligned} \left(\frac{\partial\psi}{\partial t}\right)_{p_{\theta},p_{\phi},\theta,\phi} &= \frac{\partial(\psi, p_{\theta}, p_{\phi}, \theta, \phi)}{\partial(t, p_{\theta}, p_{\phi}, \theta, \phi)} = \frac{\partial(\psi, p_{\theta}, p_{\phi}, \theta, \phi)}{\partial(t, \rho_{\parallel}, \psi, \theta, \phi)} \frac{\partial(t, \rho_{\parallel}, \psi, \theta, \phi)}{\partial(t, p_{\theta}, p_{\phi}, \theta, \phi)} \\ &= -\frac{1}{J} \frac{\partial(p_{\phi}, p_{\theta}, \psi, \theta, \phi)}{\partial(t, \rho_{\parallel}, \psi, \theta, \phi)} = -\frac{1}{J} \left[\frac{\partial(p_{\phi}, p_{\theta})}{\partial(t, \rho_{\parallel})}\right]_{\psi,\theta,\phi} \\ &= -\frac{1}{J} \left[ \left(\frac{\partial p_{\phi}}{\partial t}\right)_{\rho_{\parallel}} \left(\frac{\partial p_{\theta}}{\partial \rho_{\parallel}}\right)_t - \left(\frac{\partial p_{\theta}}{\partial t}\right)_{\rho_{\parallel}} \left(\frac{\partial p_{\phi}}{\partial \rho_{\parallel}}\right)_{\rho_{\parallel}} \right]_{\psi,\theta,\phi} \end{aligned} \quad (2-122)$$

Again, using Eq. (2-114) and the equations for the canonical momenta (2-100) and (2-101), we have

$$\left(\frac{\partial\psi}{\partial t}\right)_{p_{\theta},p_{\phi},\theta,\phi} = -\frac{1}{J} \left[ -\frac{q}{2\pi} V(t) i(\psi) \right] \quad (2-123)$$

Finally, substituting Eqs. (2-118), (2-119), (2-120), (2-121) and (2-123) into Eq. (2-117) yields

$$\dot{\psi} = -\frac{1}{J} \left(\frac{\partial p_{\phi}}{\partial \rho_{\parallel}}\right)_{\psi,\theta,\phi} \left(\frac{\partial H}{\partial \theta}\right)_{\rho_{\parallel},\psi,\phi} + \frac{1}{J} \left(\frac{\partial p_{\theta}}{\partial \rho_{\parallel}}\right)_{\psi,\theta,\phi} \left(\frac{\partial H}{\partial \phi}\right)_{\rho_{\parallel},\psi,\theta}$$



$$+\frac{1}{J}\left[\frac{q}{2\pi}V(t)i(\psi)\right] \quad (2-124)$$

It should be noted that for non-zero values of the loop voltage the particle energy is no longer conserved. The particles gain (or loose) the equivalent of one loop voltage worth of energy after every toroidal circuit. The energy change is given by

$$\begin{aligned} \frac{dH}{dt} &= \left(\frac{\partial H}{\partial t}\right)_{p_\theta, \theta, \phi} = \frac{\partial(H, p_\theta, \theta, \phi)}{\partial(t, p_\theta, \theta, \phi)} \\ &= \frac{\partial(H, p_\theta, \theta, \phi)}{\partial(p_\phi, p_\theta, \theta, \phi)} \frac{\partial(p_\phi, p_\theta, \theta, \phi)}{\partial(\chi, p_\theta, \theta, \phi)} \frac{\partial(\chi, p_\theta, \theta, \phi)}{\partial(t, p_\theta, \theta, \phi)} \\ &= \dot{\phi} \left(\frac{q}{2\pi}\right) V(t), \end{aligned} \quad (2-125)$$

where Eqs. (2-94), (2-101) and (2-114) were used in the final step.

## 2.5 Plasma Kinetic Theory

The study of individual particle motion is very valuable because it leads to an intuitive understanding of particle trajectories, drifts and the difficulties associated with confining particles in a device with complicated geometry. In practice, however, we have no way of measuring single particle behavior. It is not possible, therefore, to relate the theory of individual particle motion to actual experiments. An alternative approach is to look at collective particle behavior in terms of distribution functions. In this section we discuss how measurable quantities such as particle density, fluid velocity, pressure and temperature can be represented using distribution functions. This section is concluded with the introduction of the Fokker-Planck equation.

### 2.5.1 The Phase Space Distribution Function

If we wanted to describe the state of a plasma containing  $N$  particles, each with position  $\mathbf{x}_i$  and velocity  $\mathbf{v}_i$ , in detail, we could do it using a  $6N$ -dimensional phase space. Each state of the plasma would then occupy a single point  $(\mathbf{x}_1, \dots, \mathbf{x}_N, \mathbf{v}_1, \dots, \mathbf{v}_N)$  in that space. Obviously, such a concept is very difficult to work with. A simpler concept is to reduce the space to six dimensions  $(\mathbf{x}, \mathbf{v})$  and to define a distribution function  $f(\mathbf{x}, \mathbf{v}, t)$  where

$$f(\mathbf{x}, \mathbf{v}, t) d^3\mathbf{x}d^3\mathbf{v} = f(\mathbf{x}, \mathbf{v}, t) dx dy dz dv_x dv_y dv_z \quad (2-126)$$

is the number of particles at position  $\mathbf{x}$ , velocity  $\mathbf{v}$  and time  $t$  in an element of the six dimensional space (called a phase space) of volume  $dx dy dz dv_x dv_y dv_z$ . In other words, Eq. (2-126) describes the number of particles at time  $t$  between  $(x, y, z)$  and  $(x+dx, y+dy, z+dz)$  with velocity between  $(v_x, v_y, v_z)$  and  $(v_x+dv_x, v_y+dv_y, v_z+dv_z)$ . Integrating Eq. (2-126) over all velocities, we obtain the spatial density  $n(\mathbf{x}, t)$

$$n(\mathbf{x}, t) = \int_{-\infty}^{+\infty} f(\mathbf{x}, \mathbf{v}, t) d^3\mathbf{v}. \quad (2-127)$$

The spatial density is often called the zeroth moment of the distribution. The first moment is the local fluid velocity  $\mathbf{u}(\mathbf{x}, t)$  given by averaging the particle velocity

$$\mathbf{u}(\mathbf{x}, t) = \frac{\int_{-\infty}^{+\infty} \mathbf{v} f(\mathbf{x}, \mathbf{v}, t) d^3\mathbf{v}}{\int_{-\infty}^{+\infty} f(\mathbf{x}, \mathbf{v}, t) d^3\mathbf{v}} = \frac{1}{n} \int_{-\infty}^{+\infty} \mathbf{v} f(\mathbf{x}, \mathbf{v}, t) d^3\mathbf{v} \equiv \langle \mathbf{v} \rangle. \quad (2-128)$$

The second moment of the distribution function is the tensor  $\langle \mathbf{v}\mathbf{v} \rangle$  which is proportional to the energy, pressure and temperature of the distribution.

The fundamental equation in plasma kinetic theory which describes the time evolution of the distribution function is the Fokker-Planck equation.

$$\frac{df}{dt} = C_{F.P.}(f) \quad (2-129)$$

The left hand side can be re-written using the convective derivative as

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial \mathbf{x}} \cdot \frac{d\mathbf{x}}{dt} + \frac{\partial f}{\partial \mathbf{v}} \cdot \frac{d\mathbf{v}}{dt}$$

$$\begin{aligned}
&= \frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} + \frac{\mathbf{F}}{m} \cdot \frac{\partial f}{\partial \mathbf{v}} \\
&= \frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} + \frac{q}{m} (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \frac{\partial f}{\partial \mathbf{v}}
\end{aligned} \tag{2-130}$$

The term on the right hand side of Eq. (2-129)

$$C_{F.P.}(f) \equiv \left( \frac{\partial f}{\partial t} \right)_c \tag{2-131}$$

is called the Fokker–Planck collision operator and represents, as may be expected from the name, the effects of collisions between particles on the distribution function. In fully ionized plasmas, the collisionality is the effect of many small Coulomb collisions. Let us derive the form of the Fokker–Planck collision operator.

Let  $g(\mathbf{v}, \Delta\mathbf{v})$  be the probability that a particle with initial velocity  $\mathbf{v}$  acquires an increment of velocity  $\Delta\mathbf{v}$  in a time  $\Delta t$  due to the Coulomb collisions. We can then write the distribution function  $f(\mathbf{x}, \mathbf{v}, t)$  as a product of the distribution at time  $t - \Delta t$  multiplied by the probability of change in the time  $\Delta t$  integrated over all possible  $\Delta\mathbf{v}$

$$f(\mathbf{x}, \mathbf{v}, t) = \int f(\mathbf{x}, \mathbf{v} - \Delta\mathbf{v}, t - \Delta t) g(\mathbf{v} - \Delta\mathbf{v}, \Delta\mathbf{v}) d^3 \Delta\mathbf{v}. \tag{2-132}$$

Taylor expanding the product  $fg$  to second order

$$f(\mathbf{x}, \mathbf{v}, t) = \int \left[ f(\mathbf{x}, \mathbf{v}, t - \Delta t) g(\mathbf{v}, \Delta\mathbf{v}) - \Delta\mathbf{v} \cdot \left( \frac{\partial}{\partial \mathbf{v}} (fg) \right) \right. \tag{2-133}$$

$$\left. + \frac{1}{2} \Delta\mathbf{v} \Delta\mathbf{v} : \left( \frac{\partial^2}{\partial \mathbf{v} \partial \mathbf{v}} (fg) \right) \right] d^3 \Delta\mathbf{v} \tag{2-134}$$

we obtain

$$f(\mathbf{x}, \mathbf{v}, t) = f(\mathbf{x}, \mathbf{v}, t - \Delta t) - \frac{\partial}{\partial \mathbf{v}} \cdot (f \langle \Delta\mathbf{v} \rangle) + \frac{1}{2} \frac{\partial^2}{\partial \mathbf{v} \partial \mathbf{v}} : (f \langle \Delta\mathbf{v} \Delta\mathbf{v} \rangle), \tag{2-135}$$

where

$$\int \psi d^3(\Delta\mathbf{v}) \equiv 1, \tag{2-136}$$

$$\int \psi \Delta\mathbf{v} d^3(\Delta\mathbf{v}) \equiv \langle \Delta\mathbf{v} \rangle, \tag{2-137}$$

and

$$\int \psi \Delta\mathbf{v} \Delta\mathbf{v} d^3(\Delta\mathbf{v}) \equiv \langle \Delta\mathbf{v} \Delta\mathbf{v} \rangle. \tag{2-138}$$

Using Eq. (2-131) we obtain the Fokker-Planck collision term

$$C_{F.P.}(f)\Delta t = -\frac{\partial}{\partial \mathbf{v}} \cdot (f\langle \Delta \mathbf{v} \rangle) + \frac{1}{2} \frac{\partial^2}{\partial \mathbf{v} \partial \mathbf{v}} : (f\langle \Delta \mathbf{v} \Delta \mathbf{v} \rangle). \quad (2-139)$$

The collision operator conserves particles, momentum and energy. The three conservation laws are expressed as

$$\int C(f) d^3 \mathbf{v} = 0 \quad (2-140)$$

$$\int m \mathbf{v} C(f) d^3 \mathbf{v} = 0 \quad (2-141)$$

$$\int H C(f) d^3 \mathbf{v} = 0. \quad (2-142)$$

We can see that the full Fokker-Planck equation is an extremely complicated integro-differential equation and can not be solved analytically. Let us look at what the two terms on the right hand side represent physically. The first term describes the frictional force slowing down fast particles and accelerating slow ones. The negative divergence in velocity space means a narrowing of the the distribution. The second term describes the diffusion in velocity space. It describes the broadening of a narrow velocity distribution as a result of collisions. In an equilibrium, the two terms balance each other and the distribution function is a constant. An equilibrium distribution function is called a Maxwellian. Let us derive the form of the Maxwellian.

From the H-theorem of thermodynamics, we know that the entropy,  $S$ , of a system increases until the system reaches an equilibrium. The entropy in kinetic theory is given by

$$S(t) = - \int f \ln(f) d^3 \mathbf{x} d^3 \mathbf{v}. \quad (2-143)$$

An equilibrium distribution function must therefore satisfy

$$\frac{dS}{dt} = - \int \ln(F_M) C(F_M) d^3 \mathbf{x} d^3 \mathbf{v} = 0 \quad (2-144)$$

where we used Eq. (2-140). Using the conservation properties of the collision operator and Eq. (2-144), the Maxwellian can be written as

$$F_M(\mathbf{v}) = \exp(1 + \gamma \cdot \mathbf{v} + \zeta - \beta H). \quad (2-145)$$

In a stationary reference frame,  $\gamma = 0$  and the Maxwellian is

$$F_M(\mathbf{v}) = \exp(1 + \zeta - \beta H). \quad (2-146)$$

Substituting  $F_M$  into Eq. (2-143) and using the expression for the Gibbs free energy  $G \equiv U - TS + pV = N\mu$  with  $T$  the temperature,  $N$  the number of particles,  $U$  the energy and  $\mu$  the chemical potential, we can identify  $\beta$ ,  $\zeta$  and  $H$  in Eq. (2-146) as  $1/T$ ,  $\mu/T$  and  $U$ , respectively. A equivalent, but more common expression for the Maxwellian is

$$F_M = \left( \frac{n}{(2\pi mT)^{3/2}} \right) \exp\left(-\frac{mv^2}{2T}\right), \quad (2-147)$$

where now

$$\zeta = \frac{\mu}{T} = \ln \left[ n \left( \frac{m}{2\pi T} \right)^{3/2} \right] + 1,$$

$$\beta = \frac{1}{T}, \quad \text{and} \quad H = \frac{1}{2}mv^2.$$

It is clear that for any system in which the distribution is a Maxwellian everywhere (global Maxwellian) the plasma is uniform in space and the system is free from any interesting physics. In tokamaks, plasmas are considered to be close to thermodynamic equilibrium on a local scale, even though there exist large radial temperature and density gradients on a global scale. The concept of a local Maxwellian—a function of a spatial coordinate (in this case the radial coordinate) as well as the velocity—is introduced. For the equations above, it means that both  $\zeta$  and  $\beta$ , which relate to the density and temperature, are functions of  $\psi$ , the radial coordinate. A system is considered to be in local thermodynamic equilibrium if it can be described by a local Maxwellian  $F_M(\psi, \mathbf{v})$ , which means that the system is considered to be a Maxwellian in velocity distribution over a region that is small compared to its size.

Let us make a final remark on the velocity distributions in plasmas of fusion interest. Fusion plasmas are generally confined for many collision times, which means that the

deviations from a Maxwellian must be very small. In fact, the deviations from equilibrium on a local scale are only of order 1%. The  $\delta f$  Monte Carlo method described in the next chapter takes advantage of the smallness of the deviations to calculate various parameters of the plasma much more efficiently than traditional Monte Carlo methods.

## Chapter 3

### The $\delta f$ Method

In this chapter, the  $\delta f$  method is introduced and applied as a scheme for a Monte Carlo simulation. We will discuss the logic leading to the approach of this method and derive it. This chapter will also serve as a link between the plasma theory described in the previous chapter and the actual numerical simulation. We will describe in detail how the equations used in the  $\delta f$  code were obtained from the Fokker–Planck equation discussed in the previous section. The  $\delta f$  code will be used to calculate two plasma currents, the bootstrap current and the Pfirsch–Schlüter current. To test the  $\delta f$  method, we first consider a very simplified system for which there exist well established analytical theory and results from previous simulations. Some preliminary results for a single species (ions), non-momentum conserving, axisymmetric system are presented and compared to analytic theory and other numerical simulations. (That portion of the chapter follows very closely the paper [A  \$\delta f\$  Monte Carlo Method to calculate plasma currents](#) [17].) After establishing reliability of the  $\delta f$  method for the simple case, we apply it to more complicated systems. We expand the simulation to include momentum-conservation, a second species (electrons), non-axisymmetric fields, a loop voltage (time varying Hamiltonian), radial dependence in the toroidal and poloidal currents and finally, quasi-neutrality.

### 3.1 Introduction

Monte Carlo codes have proven to be an important tool for studying the confinement properties of plasma devices. Despite their usefulness, traditional Monte Carlo techniques are extremely inefficient when calculating plasma currents because they involve not only calculations of the part of the distribution function directly responsible for the currents, but the entire distribution function. The method presented here calculates only the deviation  $\hat{f}$  of the distribution function from a Maxwellian rather than the Maxwellian itself. For most flows or currents in plasmas of fusion interest  $\hat{f}$  is small, of order  $10^{-2}$ . The statistical errors in Monte Carlo techniques scale as  $N^{-1/2}$  with  $N$  the number of particles in the simulation. A standard Monte Carlo calculation, which simulates the full distribution function  $f$  would take more than  $10^4$  particles just to detect the presence of such a current; the  $\delta f$  Monte Carlo method is of order  $10^4$  times as efficient.

We want to test the new method in areas where there exist well established analytic predictions as well as results from other numerical simulations before applying it to more complicated systems (following section 3.7). We used the  $\delta f$  method to obtain predictions for the bootstrap current and the  $\cos(\theta)$  and  $\cos(2\theta)$  components of the Pfirsch–Schlüter current in an axisymmetric configuration.

The simulated values for the bootstrap and Pfirsch–Schlüter currents have been compared with values obtained from analytic theory. Over the ranges  $0.001 < \nu_* < 100.0$  and  $0.005 < \epsilon < 0.3$ , the simulated values for the bootstrap current accurately fit the formula

$$j_b \equiv \left\langle \frac{j_{\parallel}}{B} \right\rangle B_0 = \frac{(j_b)_0 \sqrt{\epsilon} (1 + C_t \epsilon)}{1 + \sqrt{\nu_*} + 1.44 \nu_*} \quad (3-148)$$

where  $(j_b)_0$  is the analytic value for the bootstrap current divided by  $\sqrt{\epsilon}$  in the limit as  $\epsilon \rightarrow 0$ ,  $\nu_* \rightarrow 0$  and  $C_t$  is a constant representing the contribution to the bootstrap current due to trapped particles. The collisionality dependence of the simulated bootstrap



current is very similar to the analytic expression obtained by Hinton and Rosenbluth [18],

$$j_b \propto \frac{1}{1 + \sqrt{\nu_*} + 0.54\nu_*}. \quad (3-149)$$

The analytic dependence of the bootstrap current on the aspect ratio and  $(j_b)_0$  were obtained by numerically integrating an expression given by Boozer and Gardner [19]. The expression represents only the passing particle contribution to the bootstrap current. The trapped particle contribution to the bootstrap current is smaller than the passing particle contribution by a factor of the three-halves power of the fraction of trapped particles. The fraction of the trapped particles approaches zero at high aspect ratios, hence the analytic value for  $(j_b)_0$  calculated for passing particles represents  $(j_b)_0$  for all particles. The simulated values for the bootstrap current do not exhibit a linear dependence in  $\epsilon$  in contradiction to results recently obtained by Wu and White [20] using a traditional Monte Carlo method. Wu and White found the dependence of the bootstrap current to be  $j_b \propto 1.6\sqrt{\epsilon} - 0.6\epsilon$ . Their result is in agreement with the analytic result obtained by Rutherford [21], who also obtained  $j_b \propto 1.6\sqrt{\epsilon}$ . Rutherford's calculation involved an analytic expansion of an elliptic integral, which he carried out to lowest order only. If this elliptic integral is evaluated numerically, his result becomes  $j_b \propto 1.46\sqrt{\epsilon}$  which is in agreement with our results and results obtained previously by Pytte and Boozer [22].

The simulated Pfirsch–Schlüter current does not exhibit a collisionality dependence in agreement with analytic theory. The aspect ratio dependences of both the  $\cos(\theta)$  and  $\cos(2\theta)$  components agree very well with their analytic predictions. The analytic predictions for the Pfirsch–Schlüter current were obtained by solving the Vlasov equation as well as the fluid equations.

Numerical simulations were carried out using a fourth order Runge–Kutta integrator

in the orbit step and a Monte Carlo equivalent of the Lorentz collision operator developed by Boozer and Kuo-Petravic [23] in the collision step. We have ignored energy scattering in the simulation, therefore the collision operator is a pitch angle scattering operator only. For simplicity the initial distribution function of the particles was taken to be a delta function in energy. The values for the bootstrap current and the Pfirsch-Schlüter current were calculated 100 times and recorded ten times per collision time. There are subtleties associated with the sampling rate and the conversion of the Monte Carlo summation over particles into the integration over the magnetic coordinates  $\theta, \phi$  and  $\psi$  found in the analytic calculation for the currents. Both subtleties will be discussed in subsequent sections.

### 3.2 Derivation of the $\delta f$ Method

The starting point in this simulation is the Fokker-Planck equation.

$$\frac{df}{dt} = C(f) \quad (3-150)$$

As stated in the previous chapter, fusion plasmas are in near equilibrium on a local scale, which means the local distribution function  $f$  is almost a Maxwellian. The deviation from a Maxwellian is about one part in a hundred, hence we can write  $f$  as

$$f = F(\psi, H)_M \exp(\hat{f}), \quad (3-151)$$

where  $F(\psi, H)_M$  is a local Maxwellian with  $\psi$  the toroidal magnetic flux enclosed by a constant pressure surface and  $H = \frac{1}{2}mv^2 = \frac{1}{2}mv_{\parallel}^2 + \mu B$  the particle kinetic energy. The deviation from the Maxwellian distribution is represented by  $\hat{f} \ll 1$ . The Fokker-Planck equation can then be written as

$$f \frac{de^{\hat{f}}}{dt} + e^{\hat{f}} \frac{dF}{dt} = C(f) \quad (3-152)$$

Introducing a modified collision operator  $C_m(f) \equiv C(f)/f$  and using

$$\frac{1}{F_M} \frac{dF_M}{dt} = \frac{d(\ln F_M)}{dt}$$

the Fokker–Planck equation becomes

$$\frac{d(\ln F_M)}{dt} + \frac{d\hat{f}}{dt} = C_m(f). \quad (3-153)$$

The local Maxwellian  $F_M$  has the following form

$$F(\psi, H)_M = \exp(\zeta(\psi) - 1 - H\beta(\psi)) \quad (3-154)$$

where  $\zeta = \mu/T$ , the chemical potential divided by the temperature and  $\beta = 1/T$ . The time derivative of  $\ln F_M$  can now be written as

$$\frac{d(\ln F_M)}{dt} = \frac{d\zeta}{dt} - H \frac{d\beta}{dt} - \beta \frac{dH}{dt}. \quad (3-155)$$

For a time independent problem the only time dependence in a drift Hamiltonian  $H$  is in the poloidal flux  $\chi(\psi, t)$  with

$$\frac{dH}{dt} = \frac{\partial H}{\partial t} = \left(\frac{e}{c}\right) v_{\parallel} \frac{\partial \chi}{\partial t} \quad (3-156)$$

The time derivative of the poloidal flux is the loop voltage,  $V = \partial\chi/\partial t$ , which increases the energy of particles. For simplicity we set  $V = 0$ . The Fokker-Planck equation, after applying the chain rule, then becomes

$$\frac{d\hat{f}}{dt} + \left(\frac{d\zeta}{d\psi} - H \frac{d\beta}{d\psi}\right) \frac{d\psi}{dt} = C_m(f). \quad (3-157)$$

Eq. (3-157) is a linear equation in  $d\psi/dt$ . It can be separated into two equations, one involving  $\zeta$  and the other involving  $\beta$ .

$$\frac{d\delta_{\zeta}}{dt} + \frac{d\psi}{dt} = C(\delta_{\zeta}), \quad \text{where } \delta_{\zeta} \equiv \frac{\hat{f}}{\frac{d\zeta}{d\psi}} \quad (3-158)$$

$$\frac{d\delta_{\beta}}{dt} - H \frac{d\psi}{dt} = C(\delta_{\beta}), \quad \text{where } \delta_{\beta} \equiv \frac{\hat{f}}{\frac{d\beta}{d\psi}}. \quad (3-159)$$

To find the deviation  $\delta$  it appears we must solve both equations, but without energy scattering,  $H$  remains constant and equations (3-158) and (3-159) are trivially related. It is therefore sufficient to solve only Eq. (3-158).

### 3.3 The Monte Carlo Code

To solve the equation

$$\frac{d\delta}{dt} + \frac{d\psi}{dt} = C(\delta) \quad (3-160)$$

we monitor a large number of particles distributed over a narrow annulus in  $\psi$ . Following Eq. (3-160), we can identify  $\delta$  as the displacement of a particle from its original  $\psi$  surface, which we shall call the particle's home flux surface. The displacements  $\delta$  at the start of the simulation are set to zero, which is equivalent to assuming the initial distribution is Maxwellian. The initial toroidal and poloidal positions as well as the initial pitches of the particles are irrelevant because after about one collision time the particles will distribute themselves evenly throughout the  $\theta$ ,  $\phi$  and  $\lambda$  regions with equal number of particles in equal volumes. The initial energies for all particles are equal and do not change during the simulation. We choose a monoenergetic distribution as our background "Maxwellian" and a collision operator that does not include energy scattering rather than a true Maxwellian and an energy scattering operator because it greatly increases the efficiency of the code. The computing time needed for a Monte Carlo simulation with a true Maxwellian velocity distribution which includes pitch angle and energy scattering is about an order of magnitude larger than the time needed for a simulation with a monoenergetic distribution and pitch angle scattering only. The reason for the inefficiency of the simulation with a Maxwellian distribution are the highly energetic particles which require much smaller step sizes than the average particles to ensure the same accuracy of the integration. The number of timesteps needed to follow

the highly energetic particles over one collision time is therefore much greater than the number required for the average particle. In addition, the collision time  $\tau_c$  scales as  $E^{3/2}$  which means that the collision time is very long for highly energetic particles. Results obtained with a monoenergetic distribution can be convoluted with a Maxwellian energy distribution to obtain results that agree with those obtained from a Monte Carlo simulation with Maxwellian energy distribution [24].

Following the initialization procedure, we employ the usual Monte Carlo scheme consisting of an orbit step followed by a collision step to update the particles' positions, momenta and pitches. During the orbit step, in which a fourth order Runge-Kutta integrator is applied, the particles' toroidal and poloidal positions and their canonical momenta are updated using the Hamilton equations of motion. If a particle leaves the narrow  $\psi$  annulus, the event is recorded and the particle is put back into the middle of the annulus. To bring the Hamiltonian  $H$  into the desired form, we use Eqs. (2-88) and (2-89) for the canonical momenta. The Hamiltonian then becomes

$$H = \frac{(2\pi)^2}{2m} \left( \frac{B}{\mu_0 G_0} \right)^2 \left( p_\phi + \frac{q\chi}{2\pi} \right)^2 + \mu B \quad (3-161)$$

where  $\mu_0$  is the permeability of free space,  $G_0$  the total poloidal current outside a constant-pressure surface and  $-\chi$  the poloidal flux. The magnetic moment

$$\mu = \frac{mv^2(1 - \lambda^2)}{2B} \quad (3-162)$$

is an adiabatic invariant. It is useful to write the magnetic field strength  $B(\theta, \phi, \psi)$  as a Fourier sum for  $1/B^2(\theta, \phi, \psi)$  because this form simplifies calculations of the currents:

$$\frac{1}{B^2(\theta, \phi, \psi)} = \frac{1}{B_0^2(\psi)} \left[ \sum_{n,m} \delta_{nm}(\psi) e^{i(n\phi - m\theta)} \right], \quad (3-163)$$

with  $B_0(\psi)$  the average of the magnetic field on the surface. In an axisymmetric configuration,  $B(\theta, \psi)$  can be written as

$$B = B_0 \left( 1 + c_1 \epsilon \sqrt{\frac{\psi}{\psi_a}} \cos(\theta) + c_2 \epsilon^2 \left( \frac{\psi}{\psi_a} \right) \cos(2\theta) \right)^{-\frac{1}{2}} \quad (3-164)$$

with  $\epsilon$  the inverse aspect ratio  $r_a/R$  and  $\psi_a$  the toroidal flux enclosed by the pressure surface at the edge of the plasma. The constants  $c_1$  and  $c_2$  were chosen to be 2 and 4, respectively, to simplify the analytic calculations and to make our expression for the magnetic field strength consistent with magnetic field models described in previous papers [19, 20]. Using Eq. (3-164) as the model for the axisymmetric magnetic field and Eqs. (2-93)–(2-96) we arrive at the explicit forms of the equations to be used in the Runge–Kutta integrator routine to update  $\theta$ ,  $\phi$ ,  $p_\theta$  and  $p_\phi$

$$\begin{aligned} \dot{\theta} = & \iota \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta) \left( \frac{B}{\mu_0 G_0} \right)^2 \\ & - \left( \frac{2\pi}{q} \right) \left[ \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta)^2 \left( \frac{B}{\mu_0 G_0} \right)^2 + \mu B \right] \times \\ & \left( \frac{B}{B_0} \right)^2 \left( \frac{c_1 \epsilon}{4\sqrt{\psi} \psi_a} \cos \theta + \frac{c_2 \epsilon^2}{2\psi_a} \cos 2\theta \right) \end{aligned} \quad (3-165)$$

$$\dot{\phi} = \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta) \left( \frac{B}{\mu_0 G_0} \right)^2 \quad (3-166)$$

$$\begin{aligned} \dot{p}_\theta = & - \left[ \frac{(2\pi)^2}{m} (p_\phi + \iota p_\theta)^2 \left( \frac{B}{\mu_0 G_0} \right)^2 + \mu B \right] \times \\ & \left( \frac{B}{B_0} \right)^2 \left( \frac{c_1 \epsilon}{2} \sqrt{\frac{\psi}{\psi_a}} \sin \theta + c_2 \epsilon^2 \left( \frac{\psi}{\psi_a} \right) \sin 2\theta \right) \end{aligned} \quad (3-167)$$

$$\dot{p}_\phi = 0. \quad (3-168)$$

The magnetic field is independent of the toroidal position, hence according to the Hamilton equations of motion, the canonical toroidal momentum  $p_\phi$  is time invariant.

In the collision step, we wish to simulate the Lorentz collision operator

$$C(f) = \frac{\nu}{2} \frac{\partial}{\partial \lambda} \left[ (1 - \lambda^2) \frac{\partial f}{\partial \lambda} \right]. \quad (3-169)$$

Following Boozer and Kuo-Petravic [23], the Lorentz collision operator is simulated by applying the Monte Carlo equivalent of the Lorentz operator to change each particle's

pitch after every timestep with step size  $\tau$  according to

$$\lambda_{\text{new}} = \lambda_{\text{old}}(1 - \nu\tau) \pm [(1 - \lambda_{\text{old}}^2)\nu\tau]^{\frac{1}{2}}. \quad (3-170)$$

The symbol  $\pm$  means the sign is chosen randomly with the probability of obtaining a plus or minus the same. This random walk in pitch space is the reason this method is referred to as a Monte Carlo method.

After the orbit and collision steps, a new value of the displacement  $\delta$  is obtained by solving the left hand side of Eq. (3-160) for  $\delta$ ,

$$\delta = - \int \frac{d\psi}{dt} dt. \quad (3-171)$$

This is equivalent to the change in  $\delta$  during a time step being the change in  $\psi$  during that step. The values of  $\delta$  are used 100 times per collision time to calculate the bootstrap and Pfirsch-Schlüter currents. The change in energy between successive time steps is calculated after the orbit step and is kept between one part in  $10^{11}$  and six parts in  $10^{11}$  by varying the size of the timestep.

It is important to distinguish between the orbital or simulated time of the particles and the number of timesteps through which a given particle has been simulated. Even though all particles are initialized with equal step sizes, the step sizes are adjusted for each particle based on the change in energy between successive timesteps. During an orbit step a highly passing particle moves farther along the field line than a particle with smaller parallel velocity and equal step size. The farther advance along the trajectory for the highly passing particle results in greater change in energy and therefore a smaller step size. In the collision step immediately following the described orbit step, the pitch of the highly passing particle will be changed very slightly because of the small step size [cf. Eq. (3-170)]. After a large number of timesteps the distribution in pitch will become uneven with particles favoring the high lambda regions. To avoid this pressure anisotropy during the sampling time, the highly passing particles must be allowed to

redistribute themselves evenly throughout pitch space before the currents are calculated. This is accomplished by sampling the particles at equal orbital times rather than after equal number of timesteps. To achieve that, the sampling method was designed in the following manner. The total time for each particle is monitored by summing the values of the step sizes for each particle after every iteration. After a particle has reached the value of the total time corresponding to the sampling time (in our case 1/100 of a collision time), it is omitted from the simulation until all particles have reached that time. At that point the currents are calculated and the simulation is continued with all particles. The efficiency of the code is not affected by omitting particles during certain periods of the simulation, unless it is executed on a massively parallel machine which simulates each particle on a separate processor.

The other subtlety associated with this Monte Carlo simulation is the conversion of the summation over particles in the code to the integration over the magnetic coordinates and velocity space found in analytic theory. The summation over all particles can be identified with a phase space average. We can define the average of the quantity  $F(\mathbf{x}, \mathbf{v})$  as

$$[F] \equiv \frac{\sum F_i(\mathbf{x}_i, \mathbf{v}_i)}{N} = \frac{\int_P F(\mathbf{x}, \mathbf{v}) d^3\mathbf{x} d^3\mathbf{v}}{\int_P d^3\mathbf{x} d^3\mathbf{v}}, \quad (3-172)$$

with  $N$  the number of particles in the simulation and the subscript  $P$  denoting a phase space integral. The volume element  $d^3\mathbf{x}$  can be written in terms of the magnetic coordinates as

$$d^3\mathbf{x} = J d\theta d\phi d\psi = \frac{\mu_0 G_0}{(2\pi B)^2} d\theta d\phi d\psi \quad (3-173)$$

with the Jacobian  $J$  obtained from the scalar product of the covariant [Eq. (2-51)] and contravariant [Eq. (2-47)] representations of the magnetic field  $\mathbf{B}$ . The velocity space volume element can be written as

$$d^3\mathbf{v} = J_v dv d\theta_v d\phi_v = v^2 \sin\theta_v dv d\theta_v d\phi_v = -2\pi v^2 dv d\lambda, \quad (3-174)$$

where we used  $\lambda = \cos\theta_v$ ,  $d\lambda = -\sin\theta_v d\theta$  (with  $\theta_v$  the angle between  $\mathbf{v}$  and  $\mathbf{B}$ )



and the symmetry due to the rapid gyromotion in the  $\phi_v$  direction. In the case of a monoenergetic distribution,  $F(\mathbf{x}, v, \lambda) = F'(\mathbf{x}, \lambda)\delta(v - v_0)$  and the velocity space integration can be further simplified to

$$\int_{-\infty}^{+\infty} v^2 dv \int_{-1}^{+1} d\lambda F'(\mathbf{x}, \lambda)\delta(v - v_0) = v_0^2 \int_{-1}^{+1} d\lambda F'(\mathbf{x}, \lambda). \quad (3-175)$$

Equation (3-172) then becomes

$$[F] = \frac{2\pi v_0^2 \int_0^{2\pi} d\theta \int_0^{2\pi} d\phi \int_{\psi_{\min}}^{\psi_{\max}} d\psi \int_{-1}^{+1} d\lambda F J}{2\pi v_0^2 \int_0^{2\pi} d\theta \int_0^{2\pi} d\phi \int_{\psi_{\min}}^{\psi_{\max}} d\psi \int_{-1}^{+1} d\lambda J} \quad (3-176)$$

The integration over  $\psi$  is performed over the narrow region of the  $\psi$  annulus. In the limit  $\Delta\psi \rightarrow 0$ , the integration over  $\psi$  can be neglected and  $[F]$  becomes a pressure surface average of the quantity  $F$  rather than a volume average. Also, in an axisymmetric configuration,  $F$  is independent of the toroidal angle  $\phi$  and  $\int d\phi$  can be neglected as well. Finally, we have

$$\begin{aligned} [F] &\equiv \frac{\sum F_i(\mathbf{x}_i, \mathbf{v}_i)}{N} = \frac{2\pi v_0^2 \int_0^{2\pi} d\theta \int_{-1}^{+1} d\lambda F J}{2\pi v_0^2 \int_0^{2\pi} d\theta \int_{-1}^{+1} d\lambda J} \\ &= \frac{\int_0^{2\pi} d\theta \int_{-1}^{+1} d\lambda F J}{2 \int_0^{2\pi} d\theta J}. \end{aligned} \quad (3-177)$$

We were able to perform the final simplification in the denominator because the Jacobian  $J$  is independent of the pitch  $\lambda$ .

The integration over the magnetic coordinates is required for the Fourier decomposition of the parallel current where the  $m = 0$ ,  $n = 0$  term defines the bootstrap current and the nonzero  $m$  and  $n$  terms define the various components of the Pfirsch–Schlüter current. We will come back to the conversion of the Monte Carlo summation to integrals found in analytic theory in section 3.6.

### 3.4 Theory of the Bootstrap Current

The bootstrap current is due to the radial, or  $\psi$ , gradient in the density of the particles. Consider the orbits of two trapped particles moving in adjacent “banana orbits” around their poloidal home flux surfaces. Because of the density gradient  $dn/d\psi$ , the relative displacement of the parallel and anti-parallel moving particles is different. Let us say the particle moving parallel is traversing the inside banana orbit and the particle moving anti-parallel is traversing the outside banana. In this case there are more particles moving parallel to the magnetic field lines than there are particles moving anti-parallel. The resulting current is called the bootstrap current (Fig. 3-12).

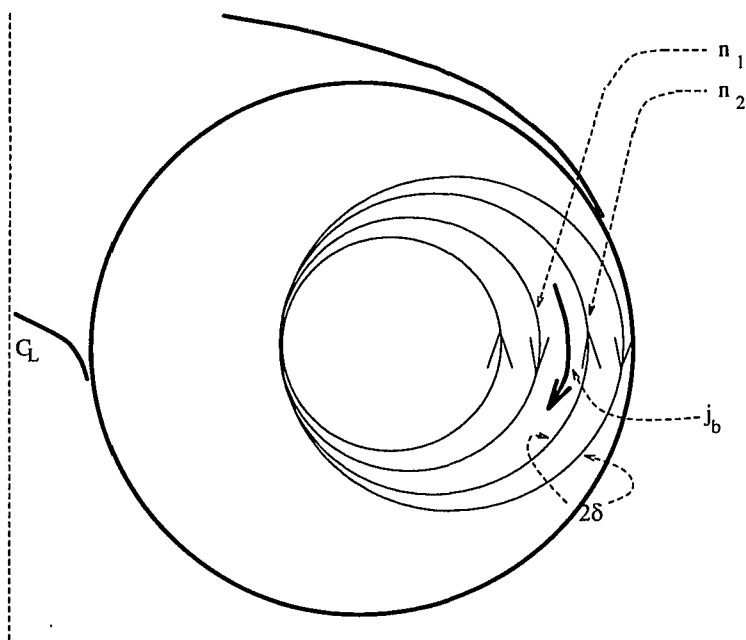


Figure 3-12: The bootstrap current  $j_b \approx e\delta(T/m)^{1/2} dn/d\psi$  is due to the radial density gradient of the plasma ( $n_1 > n_2$ ).  $C_L$  marks the center line of the torus.

The bootstrap current is approximately  $j_b \approx e\delta\sqrt{T/m} dn/d\psi$  with  $\delta$  the half-width of the banana orbit and  $\sqrt{T/m}$  the typical speed of the particles along the field lines. A better expression for the bootstrap current is given by Boozer and Gardner [19] as

$$j_b = -\Delta_0 \left( \frac{\mu_0 G_0}{B_0} \right) T \frac{dn}{d\psi}, \quad (3-178)$$

where

$$j_b = \left\langle \frac{j_{\parallel}}{B} \right\rangle B_0 \equiv \frac{\int_0^{2\pi} \left( \frac{j_{\parallel}}{B} \right) d\theta}{\int_0^{2\pi} d\theta} B_0 \quad (3-179)$$

is the average of the parallel current over  $\theta, \phi$  and  $\psi$  (in our case only the  $\theta$  averaging is relevant) and  $\Delta_0$ , the contribution to the bootstrap current due to passing particles, has the form

$$\Delta_0 = \int_0^1 d\xi' \int_0^{\xi'} d\xi \frac{3}{4} \frac{1}{\langle v_{\parallel}/B^2 \rangle} \left( \left\langle \frac{v_{\parallel}}{B^2} \frac{\partial}{\partial \alpha} \int^{\zeta} d\zeta \frac{v}{v_{\parallel}} \right\rangle - \left\langle \frac{v_{\parallel}}{B^2} \right\rangle \left\langle \frac{\partial}{\partial \alpha} \int^{\zeta} d\zeta \frac{v}{v_{\parallel}} \right\rangle \right) \quad (3-180)$$

with  $\alpha = \theta - \iota\phi, \zeta = \phi$  and  $\xi = \mu B_{\max}/E$  a nondimensional variable for the pitch angle. ( $B_{\max}$  is the maximum value of the magnetic field on the surface.) The contribution to the bootstrap current due to trapped particles is smaller than the contribution due to passing particles by a factor of the three-halves power of the fraction of trapped particles and will be neglected. In the limit  $\nu_* \rightarrow 0$  and  $\epsilon \rightarrow 0$ ,  $\Delta_0$  reduces to

$$\Delta_0 = 1.46 \frac{\sqrt{\epsilon}}{\iota}, \quad (3-181)$$

with

$$\iota = \frac{d\chi}{d\psi} \quad (3-182)$$

the rotational transform. Boozer and Gardner [19] also showed that in order to obtain  $\Delta_0$  at non-zero inverse aspect ratios and collisionalities it is not necessary to evaluate the expression given in (3-180), because  $\Delta_0(\epsilon)$  is well approximated by  $\Delta_0(\epsilon) = 1.08\Delta'_b(\epsilon)$ , where  $\Delta'_b(\epsilon)$  can be obtained from the boundary condition at the trapped-passing boundary,  $\mu = E/B_{\max}$  or  $\xi = 1$ , where

$$\Delta'_b(\epsilon) \int_{\zeta_1}^{\zeta_2} d\zeta = \frac{3B_{\max}}{2v} \int_{\zeta_1}^{\zeta_2} d\zeta \int_{\zeta_1}^{\zeta} d\zeta' \frac{\partial(v_{\parallel}/B)}{\partial \alpha}. \quad (3-183)$$

Fig. 3-13 shows a plot of  $\Delta_0$  as a function of inverse aspect ratio obtained from Eq. (3-181) and from Eq. (3-183).

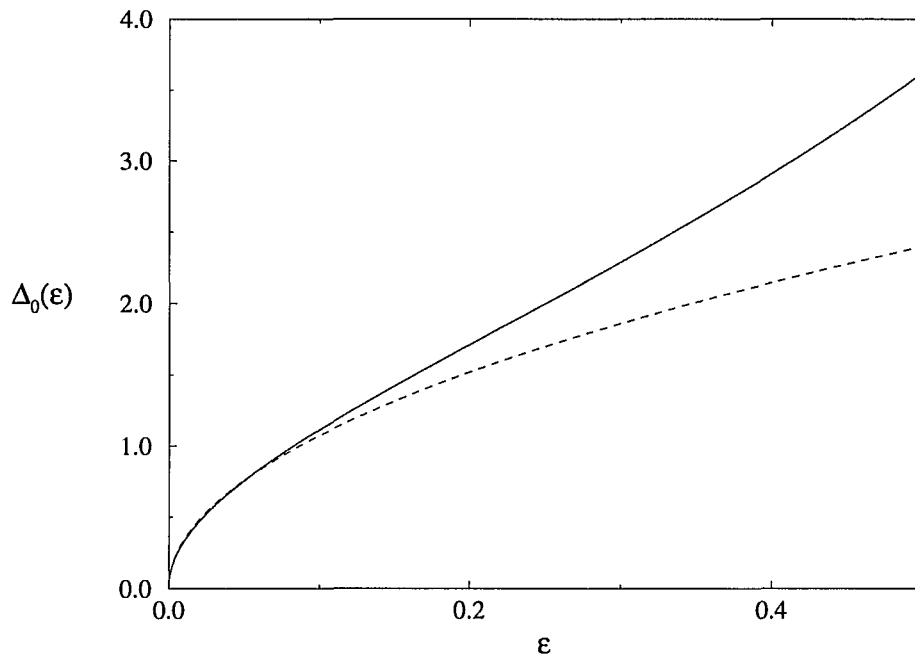


Figure 3-13: Comparison of aspect ratio dependence of  $\Delta_0(\epsilon)$  obtained from Eq. (3-181) (dashed line) and from numerical integration of Eq. (3-183) (solid line).

The temperature  $T$  in Eq. (3-178) can be expressed in terms of the energy from

$$T = \frac{1}{3} \frac{\int v^2 f d^3v}{\int f d^3v}. \quad (3-184)$$

The velocity space integral  $d^3v$  should be carried out in energy and pitch space,

$$T = \frac{1}{3} \frac{\int_{-1}^1 d\lambda \int_0^\infty \left(\frac{2E}{m}\right)^{3/2} f dE}{\int_{-1}^1 d\lambda \int_0^\infty \left(\frac{2E}{m}\right)^{1/2} f dE}. \quad (3-185)$$

Substituting in the expression for the distribution function  $f \approx c_0 \delta(E - E_0)$  with  $c_0$  a normalization constant and  $E_0$  the energy of the particles at the beginning of the simulation, the expression for the temperature reduces to

$$T = \frac{2}{3} E_0. \quad (3-186)$$

The factor  $(\mu_0 G_0/B_0)$  in Eq. (3-178) can be related to parameters used in the simulation through Eq. (2-61)

$$\frac{\mu_0 G_0}{B_0} = 2\pi R_0. \quad (3-187)$$

The expression for the bootstrap current due to the passing particles is

$$(j_b)_{passing} = -\Delta_0(\epsilon) \left(2\pi R_0\right) \frac{2E_0}{3} \frac{dn}{d\psi}. \quad (3-188)$$

To find an approximate analytic expression for the total bootstrap current, we will use the result that the contribution due to trapped particles is proportional to the three-halves power of epsilon [19]. The analytic behavior of the bootstrap current at constant collisionality can then be approximated by

$$j_b = (j_b)_0 \sqrt{\epsilon} (1 + C_t \epsilon), \quad (3-189)$$

where

$$(j_b)_0 = -\frac{\Delta_0(\epsilon)}{\sqrt{\epsilon}} (2\pi R_0) \left(\frac{2}{3} E_0\right) \frac{dn}{d\psi}. \quad (3-190)$$

and  $C_t$  is a constant representing the contribution due to trapped particles. To find an expression for the analytic collisionality dependence of the bootstrap current, we will use the result obtained by Hinton and Rosenbluth [18], where

$$j_b \propto \frac{1}{1 + \sqrt{\nu_*} + K_\nu \nu_*}, \quad (3-191)$$

so that the analytic expression for the bootstrap current as a function of aspect ratio and collisionality is

$$j_b = \frac{(j_b)_0 \sqrt{\epsilon} (1 + C_t \epsilon)}{1 + \sqrt{\nu_*} + K_\nu \nu_*}. \quad (3-192)$$

### 3.5 Theory of the Pfirsch–Schlüter Current

The Pfirsch–Schlüter current is the current required to cancel out the  $\frac{1}{B}$  dependence of the perpendicular current. Its analytic value can be obtained from

$$\mathbf{B} \cdot \nabla \left( \frac{\mathbf{j}_{\parallel}}{B} \right) = -\nabla \cdot \mathbf{j}_{\perp}. \quad (3-193)$$

The Pfirsch–Schlüter current is the special solution of this inhomogeneous differential equation. Its surface averaged value for the  $m$ -th component is

$$(j_{P.S.})_m \equiv \left\langle \frac{j_{\parallel} \cos(m\theta)}{B} \right\rangle B_0 = -\frac{c_m \epsilon^m}{2\iota} \left( \frac{\mu_0 G_0}{B_0} \right) T \frac{dn}{d\psi}. \quad (3-194)$$

The Pfirsch–Schlüter current can be obtained from the fluid equations [25] or from the Vlasov equation. The fluid equations approach is presented first.

#### Fluid Equations Approach

We start with the equilibrium equation

$$\nabla p = \mathbf{j} \times \mathbf{B} \quad (3-195)$$

Then

$$\mathbf{B} \times \nabla p = \mathbf{B} \times (\mathbf{j} \times \mathbf{B}) = \mathbf{j}(\mathbf{B} \cdot \mathbf{B}) - \mathbf{B}(\mathbf{B} \cdot \mathbf{j}) = B^2 \mathbf{j} - (j_{\parallel} B) \mathbf{B} \quad (3-196)$$

$$\frac{\mathbf{B} \times \nabla p}{B^2} = \mathbf{j} - \left( \frac{j_{\parallel}}{B} \right) \mathbf{B} = \mathbf{j}_{\perp} \quad (3-197)$$

For  $p = p(\psi)$ , we have  $\nabla p = \nabla \psi dp/d\psi$ . Then using Eq. (2-54)

$$\mathbf{j}_{\perp} = \frac{\mathbf{B} \times \nabla p}{B^2} = \frac{\mu_0}{2\pi B^2} \left[ G(\psi)(\nabla \phi \times \nabla \psi) + I(\psi)(\nabla \theta \times \nabla \psi) \right] \frac{dp}{d\psi} \quad (3-198)$$

and

$$\nabla \cdot \mathbf{j}_{\perp} = \left[ \nabla \theta \frac{\partial}{\partial \theta} + \nabla \phi \frac{\partial}{\partial \phi} + \nabla \psi \frac{\partial}{\partial \psi} \right] \cdot \frac{\mathbf{B} \times \nabla p}{B^2} \quad (3-199)$$

For  $B = B(\theta)$ ,

$$\nabla \cdot \mathbf{j}_{\perp} = \nabla \theta \cdot (\nabla \phi \times \nabla \psi) \left( \frac{\mu_0 G_0}{2\pi} \frac{dp}{d\psi} \right) \frac{\partial}{\partial \theta} \left( \frac{1}{B^2} \right). \quad (3-200)$$

From  $\nabla \cdot \mathbf{j} = 0$  and using  $\nabla \cdot \mathbf{B} = 0$ , we obtain

$$\nabla \cdot \mathbf{j}_\perp = -\mathbf{B} \cdot \nabla \left( \frac{j_\parallel}{B} \right). \quad (3-201)$$

Using the contravariant representation for the magnetic field (2-47) and

$$\nabla \left( \frac{j_\parallel}{B} \right) = \nabla \theta \frac{\partial}{\partial \theta} \left( \frac{j_\parallel}{B} \right) = \iota \nabla \phi \frac{\partial}{\partial \theta} \left( \frac{j_\parallel}{B} \right) \quad (3-202)$$

we get

$$\mathbf{B} \cdot \nabla \left( \frac{j_\parallel}{B} \right) = \left[ \frac{1}{2\pi} (\nabla \psi \times \nabla \theta + \nabla \phi \times \nabla \chi) \right] \cdot \left[ \iota \nabla \phi \frac{\partial}{\partial \theta} \left( \frac{j_\parallel}{B} \right) \right] \quad (3-203)$$

$$\mathbf{B} \cdot \nabla \left( \frac{j_\parallel}{B} \right) = \frac{\iota}{2\pi} (\nabla \psi \times \nabla \theta) \cdot \nabla \phi \frac{\partial}{\partial \theta} \left( \frac{j_\parallel}{B} \right). \quad (3-204)$$

Combining Eqs. (3-200), (3-201) and (3-204) we find

$$\frac{j_\parallel}{B} = -\frac{\mu_0 G_0}{\iota B^2} \frac{dp}{d\psi}. \quad (3-205)$$

To find the surface average of the  $m$ -th component of the Pfirsch–Schlüter current, we must calculate

$$\left\langle \frac{j_\parallel \cos(m\theta)}{B} \right\rangle = -\frac{\mu_0 G_0}{\iota B_0^2} \frac{dp}{d\psi} \frac{\int_0^{2\pi} (1 + c_1 \epsilon \cos(\theta) + c_2 \epsilon^2 \cos(2\theta)) \cos(m\theta) d\theta}{\int_0^{2\pi} d\theta} \quad (3-206)$$

which yields

$$(j_{P.S.})_m \equiv \left\langle \frac{j_\parallel \cos(m\theta)}{B} \right\rangle B_0 = -\frac{c_m}{2\iota} \epsilon^m \frac{\mu_0 G_0}{B_0} \frac{dp}{d\psi}. \quad (3-207)$$

### Vlasov Equation Approach

The Pfirsch–Schlüter current can also be obtained from the Vlasov equation which we will use in the form of Eq. (3-160) with  $C(\delta) = 0$ .

$$\frac{d\delta}{dt} + \mathbf{v} \cdot \nabla \psi = 0 \quad (3-208)$$

Using

$$\frac{d\delta}{dt} = \frac{\partial \delta}{\partial \theta} \frac{\partial H}{\partial p_\theta} \quad (3-209)$$

and

$$\mathbf{v} \cdot \nabla \psi = \frac{d\psi}{dt} = -\frac{2\pi}{q} \frac{\partial H}{\partial \theta} \quad (3-210)$$

we obtain

$$\delta = \int \frac{2\pi}{q} \left( \frac{\partial H}{\partial \theta} \right) d\theta. \quad (3-211)$$

Using Eqs. (2-89), (2-93), (2-95) and neglecting the magnetic field strength variation in the radial direction, we get

$$\delta = \int \frac{1}{2q\iota} \frac{\left( m(v^2 - \frac{2\mu B}{m}) + \mu B \right) \left( \frac{B}{B_0} \right)^2 \left( c_1 \epsilon \sin(\theta) + 2c_2 \epsilon^2 \sin(2\theta) \right)}{\sqrt{v^2 - \frac{2\mu B}{m} \left( \frac{B}{\mu_0 G_0} \right)}} d\theta. \quad (3-212)$$

We can use Eq. (3-164) to change the variable of integration from  $\theta$  to  $B$ . The integral then takes the form

$$\delta = \int \frac{1}{q\iota} \frac{\left( m(v^2 - \frac{2\mu B}{m}) + \mu B \right)}{\sqrt{v^2 - \frac{2\mu B}{m} \left( \frac{B^2}{\mu_0 G_0} \right)}} dB. \quad (3-213)$$

Performing the integral yields

$$\delta = -\frac{1}{q\iota} \frac{\mu_0 G_0}{B} m v \lambda. \quad (3-214)$$

The expression for the surface averaged  $m$ -th component of the Pfirsch–Schlüter current is

$$\left\langle \frac{j_{\parallel} \cos(m\theta)}{B} \right\rangle_{B_0} = \frac{\int_{-1}^1 d\lambda \int_0^{2\pi} d\theta \frac{\delta v_{\parallel}}{B} \cos m\theta}{\int_{-1}^1 d\lambda \int_0^{2\pi} d\theta} B_0. \quad (3-215)$$

A few steps of algebra reduce the above equation to the result obtained in Eq. (3-207).

### 3.6 Relating Monte Carlo Results to Theoretical Results

In section 3.3 we discussed the conversion of the summation over particles in the Monte Carlo code to integrals found in analytic theory. Let us now apply those results to obtain expressions for the bootstrap and Pfirsch–Schlüter currents in terms of parameters



associated with the Monte Carlo code.

The bootstrap current can be written as

$$j_b = \left\langle \frac{j_{\parallel}}{B} \right\rangle B_0 = \frac{\int (\frac{qv_{\parallel}}{B} f) d^3\mathbf{v} d\theta d\phi d\psi}{\int d\theta d\phi d\psi} B_0 \quad (3-216)$$

The distribution function  $f = F_M(1 + \hat{f})$ , with  $F_M = c_0 \delta(v - v_0)$ . Using Eq. (3-177) we have

$$j_b = \left\langle \frac{j_{\parallel}}{B} \right\rangle B_0 = \frac{2\pi v_0^3 q c_0 \int \frac{\lambda}{B} \hat{f} d\lambda d\theta}{\int d\theta} B_0 \quad (3-217)$$

The constant  $c_0$  is found from the normalization condition

$$\int f d^3\mathbf{v} = \int F_M(1 + \hat{f}) d^3\mathbf{v} = 1 \quad (3-218)$$

which yields

$$c_0 = \frac{1}{2\pi v_0^2}. \quad (3-219)$$

Substituting Eq. (3-219) into Eq. (3-217) gives

$$j_b = \frac{qv_0 \int \frac{\lambda \hat{f}}{B} d\lambda d\theta}{\int d\theta} B_0 = qv_0 \frac{2 \int d\theta J \sum_i \frac{\lambda_i \hat{f}_i}{B_i} J_i^{-1}}{\int d\theta} B_0 \quad (3-220)$$

Using the form of the Jacobian given in Eq. (3-173),

$$\frac{\int J d\theta}{\int d\theta} = \frac{\int \frac{\mu_0 G_0}{(2\pi B)^2} d\theta}{\int d\theta} = \frac{\mu_0 G_0}{(2\pi B_0)^2} \quad (3-221)$$

the bootstrap current in terms of the sum over all particles becomes

$$j_b = 2qv_0 \frac{\sum_i \lambda_i \hat{f}_i B_i}{N} B_0^{-1}. \quad (3-222)$$

The expression for the various components of the Pfirsch–Schlüter current can be obtained in an analogous manner with

$$j_{P.S.}^m = qv_0 \frac{\sum_i \lambda_i \hat{f}_i B_i \cos(m\theta)}{N} B_0^{-1}. \quad (3-223)$$

The factor of 1/2 with respect to the bootstrap current comes from the additional averaging of the  $\cos(m\theta)$  terms.

### 3.7 Results

The simulations were run with a wide range of values for the collisionality and the aspect ratio. The ranges were:  $0.001 < \nu_* < 100.0$  and  $0.005 < \epsilon < 0.5$ . The other parameters in the simulations were:  $E_0 = 5 \times 10^{-11}$ ,  $B_0 = 1$ ,  $m = 1$ ,  $q = 1$ ,  $R_0 = 3.0$ ,  $\iota = 0.43$ . All simulations were continued for several collision times after the asymptotic value of the currents has been reached to ensure small statistical deviations. Figure 3-14 shows the bootstrap and Pfirsch-Schlüter currents as functions of time in a typical simulation. The dashed lines in Fig. 3-14 mark the fits which enable us to obtain the asymptotic values for the currents.

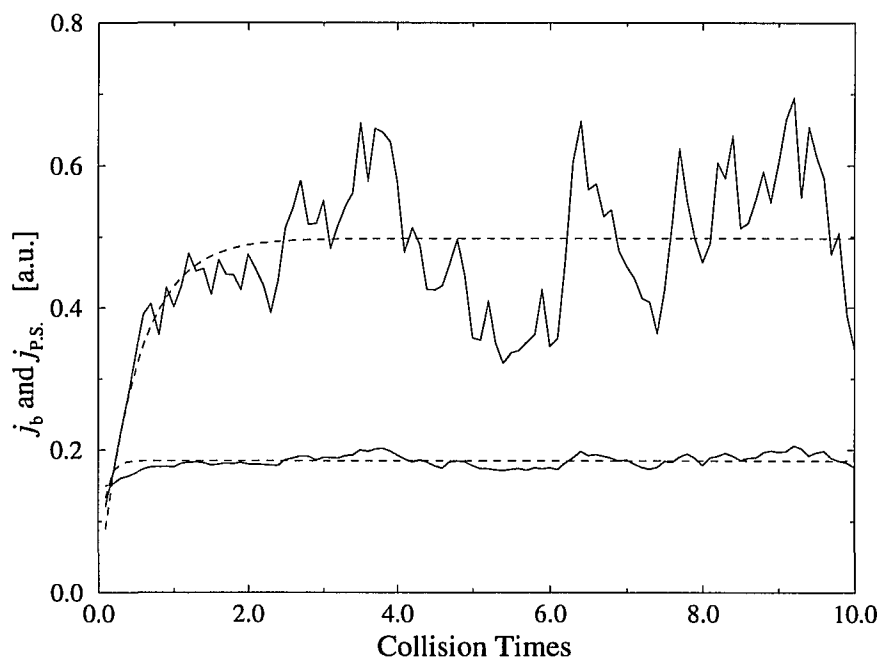


Figure 3-14: Simulated bootstrap and Pfirsch-Schlüter currents (solid lines) and their asymptotic fits (dashed lines). The bootstrap current is the greater of the two.

The fitting formula  $y = A(1 - \exp(-x/T))$  involves two free parameters.  $A$  represents

the asymptotic value of the currents and  $T$  represents the correlation time, which for the bootstrap current is about one collision time and for the Pfirsch-Schlüter current about one tenth of a collision time.

The collisionality dependence of the simulated bootstrap current was compared to the theoretical prediction given by Hinton and Rosenbluth [18] (Fig. 3-15). Hinton and Rosenbluth found the dependence to be

$$j_b \propto \frac{1}{1 + \sqrt{\nu_*} + 0.54\nu_*}. \quad (3-224)$$

Our results agree with this prediction (dashed line) well, but a slightly better fit can be obtained if the 0.54 is replaced by 1.44 (solid line).

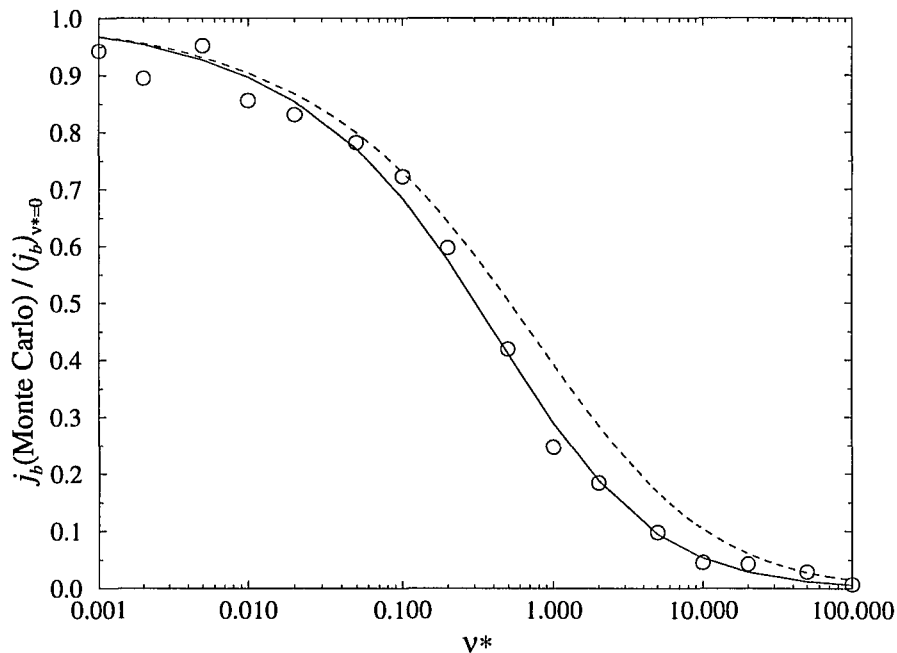


Figure 3-15: Normalized bootstrap current vs. collisionality at  $\epsilon = 0.1$ . The circles represent values obtained from Monte Carlo simulations. The dotted line is the theoretical prediction given by Hinton and Rosenbluth ( $j_b \propto [1 + \nu_*^{1/2} + 0.54\nu_*]^{-1}$ ) and the solid curve represents a fit where  $j_b \propto [1 + \nu_*^{1/2} + 1.44\nu_*]^{-1}$ .

To study the aspect ratio dependence of the bootstrap current, we numerically integrated Eq. (3-183). Fig. 3-16 shows a comparison of the analytical prediction and our results. The prediction is for passing particles represented by circles. The trapped particle contribution exhibits an  $\epsilon^{3/2}$  dependence in agreement with previously obtained results and analytic theory.

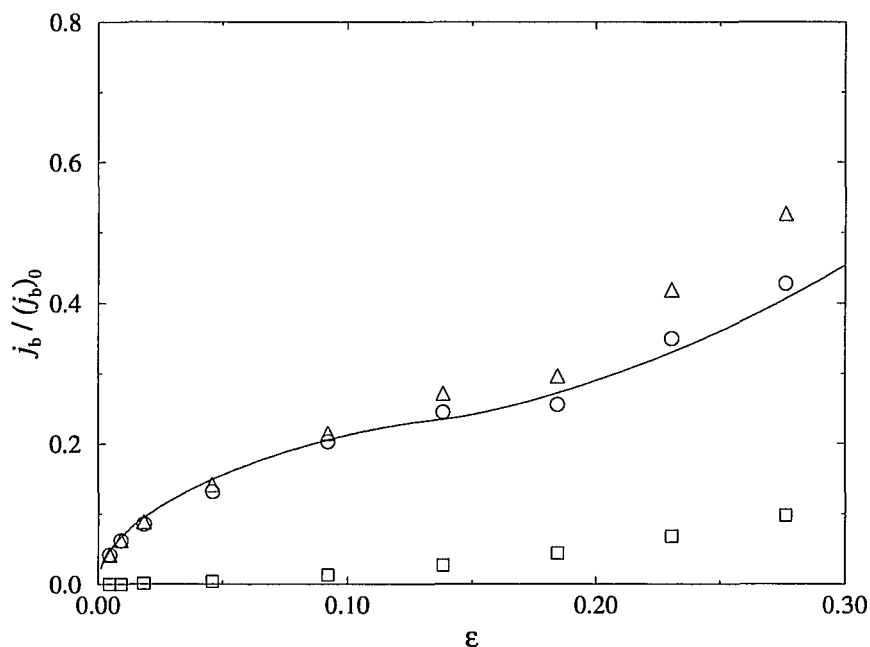


Figure 3-16: Normalized components of the bootstrap current vs. inverse aspect ratio at  $\nu_* = 0.1$ . The squares and circles represent contributions due to trapped and passing particles, respectively. The total bootstrap current is represented by triangles. The solid line is an analytic prediction for the passing particle contribution.

We also checked the prediction that

$$\frac{(j_b)_{trapped}}{(j_b)_{passing}} \propto \left( \frac{\#trapped}{\#passing} \right)^{3/2} \quad (3-225)$$

Figure 3-17 is a log-log plot of the trapped particle contribution to the bootstrap current to the trapped particle ratio. The line marks the slope 1.5, hence our results support

the prediction strongly.

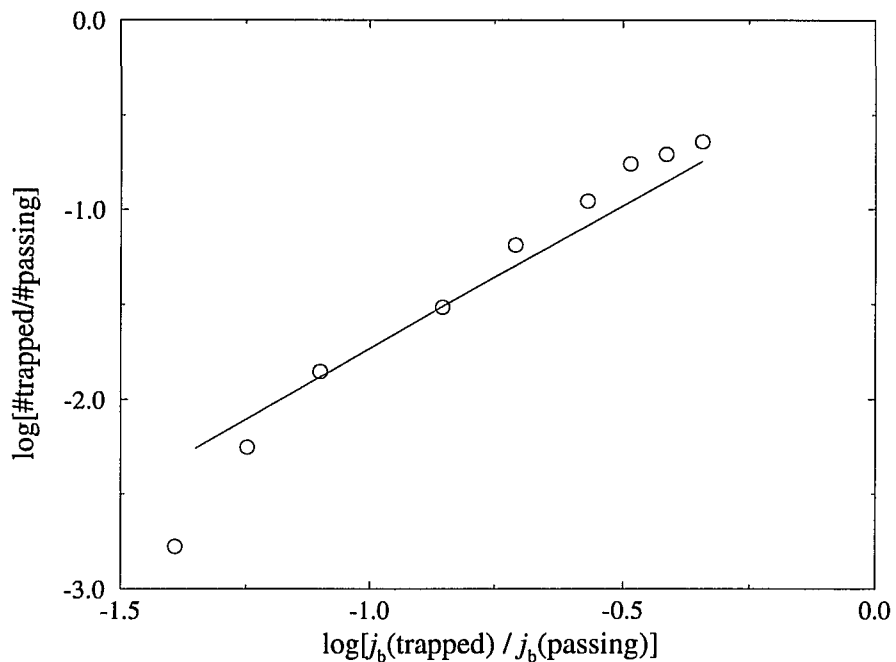


Figure 3-17: Log-log plot of the ratio of the fraction of trapped and passing particles vs. the ratio of the contribution to the bootstrap current due to the trapped and passing particles. The line represents the analytical prediction with slope  $3/2$ .

In Fig. 3-18 we plotted the various contributions to the bootstrap current divided by the square root of  $\epsilon$  versus  $\epsilon$  to better study higher order dependence of the bootstrap current on aspect ratio. The triangles, representing the total bootstrap current divided by the square root of epsilon, exhibit a linear dependence on epsilon with a slope of 2.74.

Combining our results for the collisionality and aspect ratio dependences of the total bootstrap current, we find

$$j_b = \frac{(j_b)_0 \sqrt{\epsilon} (1 + 2.74\epsilon)}{1 + \sqrt{\nu_*} + 1.44\nu_*}. \quad (3-226)$$

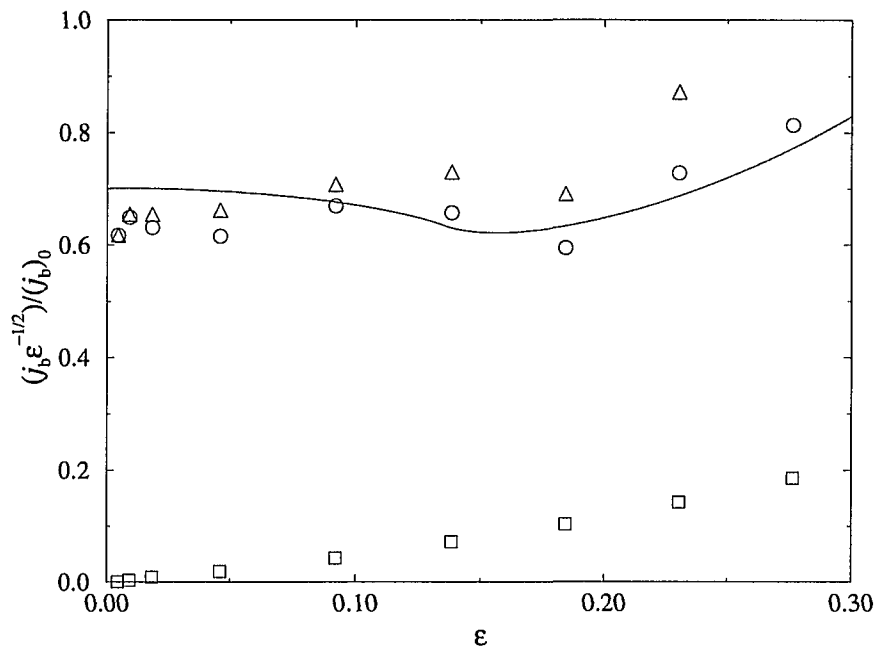


Figure 3-18: Normalized components of the bootstrap current divided by the square root of epsilon vs. epsilon at  $\nu_* = 0.1$ . The squares, circles and triangles represent the trapped particle contribution, passing particle contribution and the sum, respectively. The solid line is an analytic prediction for the passing particle contribution.

Figure 3-19 shows a comparison of the simulated bootstrap current to the bootstrap current obtained from Eq. (3-226). The solid line marks the unit slope. The average slope of the circles is 1.02 with a standard deviation of 1%.

The Monte Carlo predictions for the Pfirsch-Schlüter current were studied by simulating the collisionless Vlasov equation as well as the Fokker-Planck equation. In the simulations involving the Fokker-Planck equation, the Pfirsch-Schlüter current did not exhibit a collisionality dependence over the entire range of  $\nu_*$  (Fig. 3-20).

At constant collisionality both components of the simulated Pfirsch-Schlüter current agreed very well with their analytic predictions over the entire range of epsilon. Figures

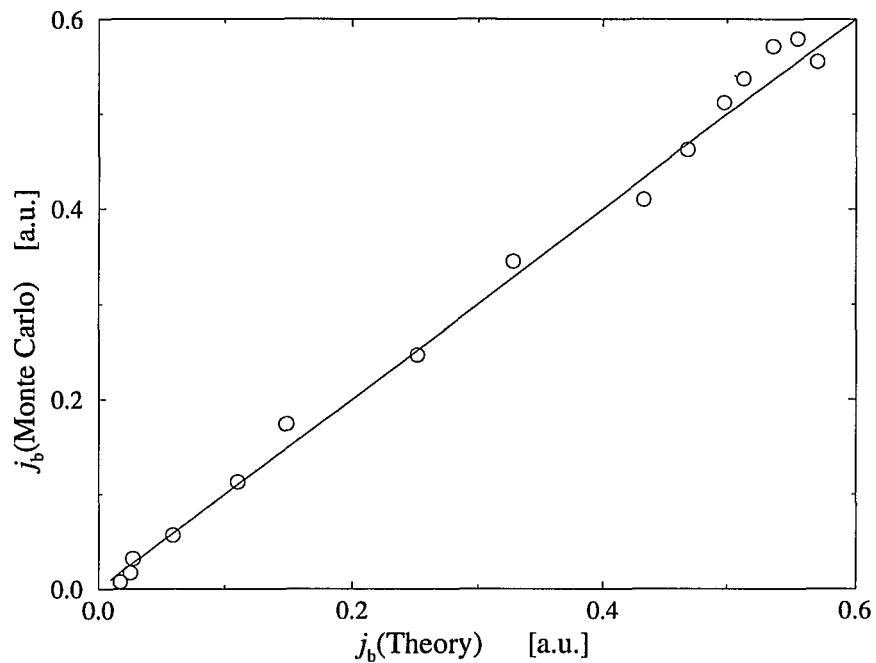


Figure 3-19:  $(j_b)_{\text{Monte Carlo}}$  vs.  $(j_b)_{\text{theoretical}}$ . The solid line marks the unit slope.

3-21 and 3-22 show the  $\cos(\theta)$  and  $\cos(2\theta)$  components of the Pfirsch–Schlüter currents normalized to their values obtained from analytic theory. Figure 3-21 shows the results for simulations of the Fokker–Planck equation and Fig. 3-22 shows results obtained by simulating the collisionless Vlasov equation.

Figure 3-23 is a comparison of simulated and analytic results for both components of the Pfirsch–Schlüter current in the  $\nu_* = 0$  and the  $\nu_* \neq 0$  cases. The line marks the unit slope. The average slope of the circles is 1.04 with a standard deviation of 1%.

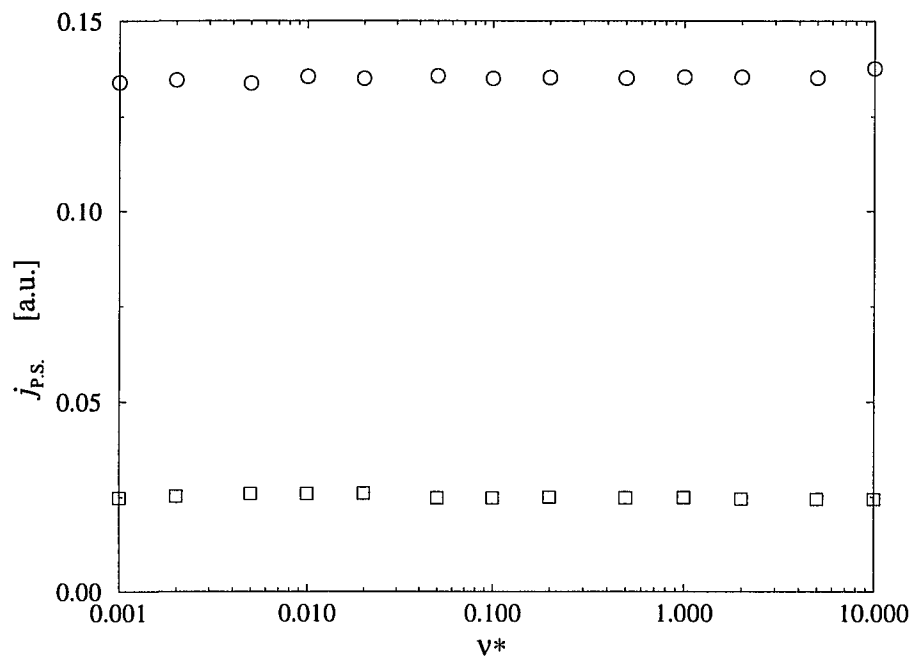


Figure 3-20:  $\cos(\theta)$  and  $\cos(2\theta)$  components of the Pfirsh-Schlüter current vs. collisionality at  $\epsilon = 0.1$ .



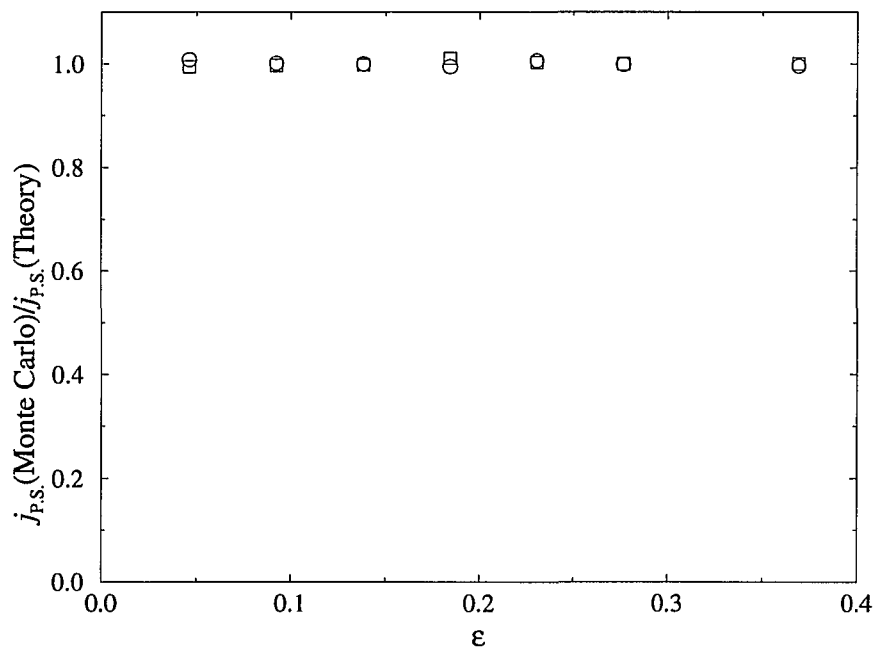


Figure 3-21: Normalized  $\cos(\theta)$  (circles) and  $\cos(2\theta)$  (squares) components of the Pfirsch-Schlüter current vs. inverse aspect ratio at  $\nu_* = 0.1$ .

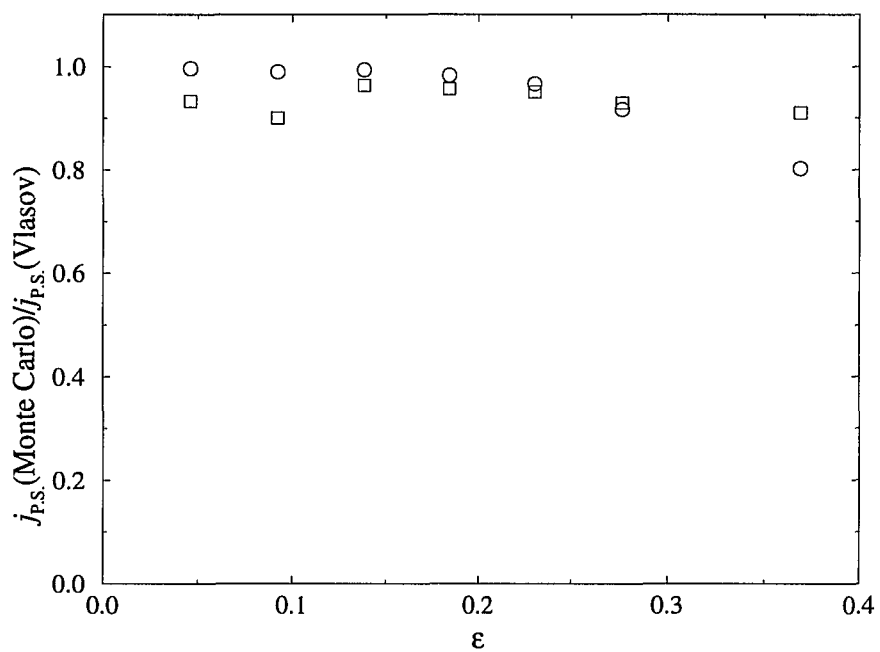


Figure 3-22: Normalized  $\cos(\theta)$  (circles) and  $\cos(2\theta)$  (squares) components of the Pfirsch-Schlüter current vs. inverse aspect ratio obtained from simulations of the Vlasov equation ( $\nu_* = 0.0$ ).

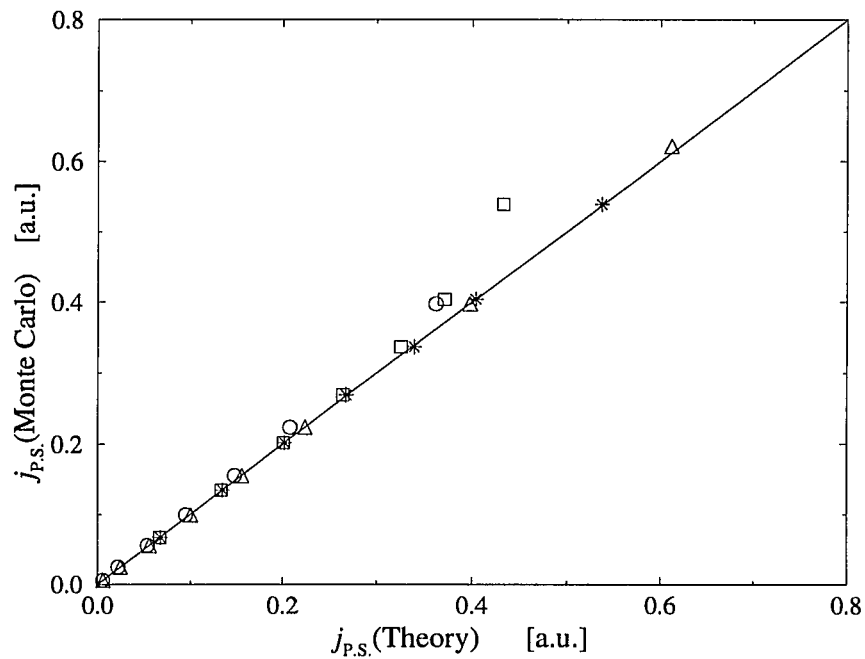


Figure 3-23:  $(j_{P.S.})_{\text{Monte Carlo}}$  vs.  $(j_{P.S.})_{\text{theoretical}}$ . The stars and squares represent the  $\cos(\theta)$  component of the Pirsch-Schlüter current with  $\nu_* \neq 0$  and  $\nu_* = 0$ , respectively. The triangles and circles represent the  $\cos(2\theta)$  component with  $\nu_* \neq 0$  and  $\nu_* = 0$ , respectively. The solid line marks the unit slope.

### 3.8 Momentum Conservation

The Lorentz collision operator  $C_L$  is used to simulate collisions due to stationary particles. If one wishes to simulate effects of collisions due to moving particles, the collision operator must include a term representing the flow velocity  $u$  of the fluid. One form of a momentum conserving operator was given by Rosenbluth, Hazeltine and Hinton [26]

$$C_{m.c.}(f) = \nu \frac{m}{B} v_{\parallel} \frac{\partial}{\partial \mu} \left[ \mu \left( v_{\parallel} \frac{\partial f}{\partial \mu} + \frac{uB}{T} f \right) \right]. \quad (3-227)$$

The collision operator can be written in terms of the pitch, rather than the magnetic moment

$$C_{m.c.}(f) = \frac{\nu}{2} \frac{\partial}{\partial \lambda} \left[ (1 - \lambda^2) \left( \frac{\partial f}{\partial \lambda} - 3 \frac{u}{v} f \right) \right]. \quad (3-228)$$

where we used

$$\mu = \frac{mv^2(1 - \lambda^2)}{2B}, \quad (3-229)$$

$$\frac{\partial}{\partial \mu} = -\frac{B}{mv^2\lambda} \frac{\partial}{\partial \lambda} \quad (3-230)$$

and Eq. (3-186). For a near Maxwellian distribution,  $f = F_M e^{\hat{f}}$ , and

$$\begin{aligned} C_{m.c.}(f) &= \frac{\nu}{2} \frac{\partial}{\partial \lambda} \left[ (1 - \lambda^2) \left( f \frac{\partial \hat{f}}{\partial \lambda} + e^{\hat{f}} \frac{\partial F_M}{\partial \lambda} - 3 \frac{u}{v} f \right) \right] \\ &= \frac{\nu}{2} \frac{\partial}{\partial \lambda} \left[ (1 - \lambda^2) \left( f \frac{\partial \hat{f}}{\partial \lambda} - 3 \frac{u}{v} f \right) \right] \end{aligned} \quad (3-231)$$

because  $F_M$  is independent of  $\lambda$ . Working through the partial derivatives,

$$\begin{aligned} C_{m.c.}(f) &= \frac{\nu}{2} \left[ f \frac{\partial}{\partial \lambda} \left( (1 - \lambda^2) \frac{\partial \hat{f}}{\partial \lambda} \right) + \frac{\partial \hat{f}}{\partial \lambda} (1 - \lambda^2) \frac{\partial f}{\partial \lambda} \right. \\ &\quad \left. - 3 \frac{u}{v} (1 - \lambda^2) \frac{\partial f}{\partial \lambda} - 3 \frac{u}{v} f (-2\lambda) \right]. \end{aligned} \quad (3-232)$$

Dividing both sides by  $f$  and neglecting the subdominant terms (the second and third term), yields

$$C_{m.c.}(f) = C_m(f) + 3\nu \frac{u}{v} \lambda, \quad (3-233)$$

with  $C_m(\hat{f})$  the modified Lorentz collision operator defined in section 3.2.

An expression for the local flow velocity  $u$  is obtained from the momentum conservation condition for  $C_{m.c.}(f)$

$$\int mv C_{m.c.}(f) d^3\mathbf{v} = 0. \quad (3-234)$$

Due to the rapid gyro-motion perpendicular to the field, the distribution function is isotropic in that direction. Therefore, only the momentum conserving properties along the field line are important. The momentum conservation condition for  $C_{m.c.}(f)$  becomes

$$\int mv_{\parallel} C_{m.c.}(f) d^3\mathbf{v} = 0. \quad (3-235)$$

Inserting the momentum conserving operator and integrating twice by parts while remembering that

$$\int d^3\mathbf{v} = 2\pi v_0^2 \int_{-1}^{+1} d\lambda, \quad (3-236)$$

we find

$$\begin{aligned} \int mv_{\parallel} C_{m.c.}(f) d^3\mathbf{v} &= \int mv\lambda \frac{\nu}{2} \frac{\partial}{\partial \lambda} \left[ (1 - \lambda^2) \left( \frac{\partial f}{\partial \lambda} - 3 \frac{u}{v} f \right) \right] d^3\mathbf{v} \\ &= mv \frac{\nu}{2} \int \frac{\partial}{\partial \lambda} \left[ \lambda(1 - \lambda^2) \left( \frac{\partial f}{\partial \lambda} - 3 \frac{u}{v} f \right) \right] \\ &\quad - \left[ (1 - \lambda^2) \left( \frac{\partial f}{\partial \lambda} - 3 \frac{u}{v} f \right) \right] d^3\mathbf{v} \\ &= -mv \frac{\nu}{2} \int \frac{\partial}{\partial \lambda} \left[ \lambda(1 - \lambda^2) f \right] + f(2\lambda) \\ &\quad - 3(1 - \lambda^2) \frac{u}{v} f d^3\mathbf{v} \\ &= -mv\nu \int f \lambda d^3\mathbf{v} + 3mv \frac{\nu}{2} \int (1 - \lambda^2) \frac{u}{v} f d^3\mathbf{v} \\ &= 0. \end{aligned} \quad (3-237)$$

Substituting for  $f = F_M(1 + \hat{f}) = c_0 \delta(v - v_0)(1 + \hat{f})$  yields

$$mv\nu 2\pi v_0^2 c_0 \int (1 + \hat{f}) \lambda d\lambda = 3mv \frac{\nu}{2} 2\pi v_0^2 c_0 \int (1 - \lambda^2) \frac{u}{v} (1 + \hat{f}) d\lambda \quad (3-238)$$

and finally

$$u = \frac{v}{2} \int \hat{f} \lambda d\lambda = \frac{1}{2} \int \hat{f} v_{\parallel} d\lambda. \quad (3-239)$$

We notice that  $u$  in Eq. (3-239) is just the definition of the local parallel flow velocity  $u_{\parallel}(\mathbf{x})$

$$u = u_{\parallel}(\mathbf{x}) \equiv \frac{\int v_{\parallel} f d^3\mathbf{v}}{\int f d^3\mathbf{v}}. \quad (3-240)$$

In the axisymmetric case  $u_{\parallel}$  depends only on the poloidal variable  $\theta$  and the toroidal flux (radial variable)  $\psi$ . The local parallel flow velocity  $u_{\parallel}(\mathbf{x})$  is related to the local parallel current  $j_{\parallel}(\mathbf{x})$  through the charge  $q$ ,  $j_{\parallel}(\mathbf{x}) = qu_{\parallel}(\mathbf{x})$ . The local parallel current can be expressed in terms of the bootstrap and Pfirsch–Schlüter currents by a Fourier decomposition in  $\theta$ .

$$\begin{aligned} j_{\parallel}(\psi, \theta) &= \left[ \left\langle \frac{j_{\parallel}}{B} \right\rangle + 2 \sum \left\langle \frac{j_{\parallel} \cos(m\theta)}{B} \right\rangle \cos(m\theta) \right] B(\psi, \theta) \\ &= \left[ \frac{j_b}{B_0} + 2 \frac{j_{P.S.}^1}{B_0} + 2 \frac{j_{P.S.}^2}{B_0} + \dots \right] B(\psi, \theta). \end{aligned} \quad (3-241)$$

The subscripts on the Pfirsch–Schlüter current indicate the corresponding poloidal component of the P.S. current. To show that Eq. (3-241) holds true, we divide by  $B/B_0$  and average over  $\theta$  to arrive at our definition of the bootstrap current [Eq. (3-179)]

$$\frac{\int_0^{2\pi} \left( \frac{j_{\parallel}}{B} \right) d\theta}{\int_0^{2\pi} d\theta} B_0 \equiv \left\langle \frac{j_{\parallel}}{B} \right\rangle B_0 = j_b. \quad (3-242)$$

Similarly, for the  $m$ -th component of the Pfirsch–Schlüter current, we divide by  $B/B_0$ , multiply by  $\cos(m\theta)$  and average over  $\theta$  to obtain

$$\frac{\int_0^{2\pi} \left( \frac{j_{\parallel} \cos(m\theta)}{B} \right) d\theta}{\int_0^{2\pi} d\theta} B_0 = \left\langle \frac{j_{\parallel} \cos(m\theta)}{B} \right\rangle B_0 = (j_{P.S.})_m. \quad (3-243)$$

To implement momentum conservation into the Monte Carlo code, the initial idea may be to find a Monte Carlo equivalent of the momentum conserving collision operator in analogy to the procedure applied to the Lorentz operator and update the pitch of every particle after each timestep according to the newly obtained prescription. A more esthetic and clever approach is to go back to the final form of the Fokker–Planck

equation [Eq. (3-160)] with the momentum conserving operator in place of the Lorentz operator, with  $\hat{f}$  replaced by  $\delta$ ,

$$\frac{d\delta}{dt} + \frac{d\psi}{dt} = C_L(\delta) + 3\nu\frac{u}{v}\lambda. \quad (3-244)$$

If we bring the momentum conserving term to the left hand side of the equation,

$$\frac{d\delta}{dt} + \frac{d\psi}{dt} - 3\nu\frac{u}{v}\lambda = C_L(\delta) \quad (3-245)$$

we notice that we can use the non-momentum conserving form of the collision operator (and the already obtained form for the Monte Carlo equivalent) to enforce momentum conservation if  $\delta$  for every particle is updated after each timestep according to

$$\Delta\delta_i = -\Delta\psi_i + 3\nu_i \frac{u(\psi, \theta)_i}{v_0} \lambda_i \Delta\tau_i, \quad (3-246)$$

with  $\Delta\tau_i$  the increment for the particle. The flow velocity  $u$  is obtained from the previous results for the bootstrap and Pfirsch–Schlüter currents [cf. Eq. (3-241)]. To minimize statistical fluctuations, the results for the currents were averaged over the last 100 timesteps using a running average.

Conservation of momentum increases the value of the bootstrap current by the factor  $1/(1-k)$  [19], where  $k$  deviates from unity by approximately the fraction of the trapped particles. We can find  $k$  from

$$k = \frac{3}{4} \int_0^1 d\xi \int_\xi^1 \frac{d\xi'}{\langle v_{\parallel}/v \rangle} \quad (3-247)$$

with  $\xi$  a nondimensional variable for the pitch angle defined previously in section 3.4 as  $\xi = \mu B_{\max}/E$ . Figure 3-24 shows the parameter  $k$  as a function of the inverse aspect ratio  $\epsilon$ .

Figure 3-25 shows the momentum conserving bootstrap current normalized to its theoretical value as a function of the inverse aspect ratio  $\epsilon$ .

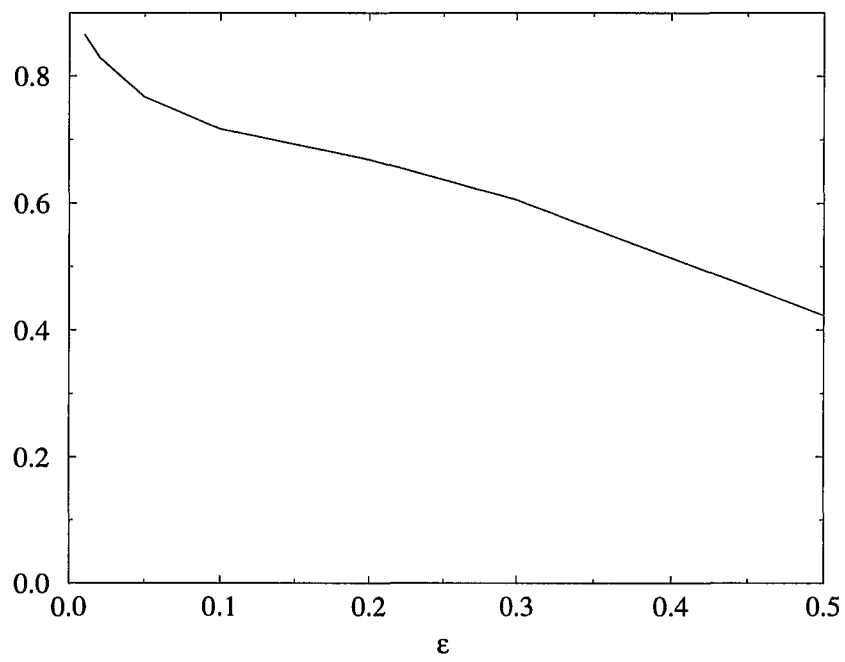


Figure 3-24: The parameter  $k$  vs. the inverse aspect ratio  $\epsilon$ .



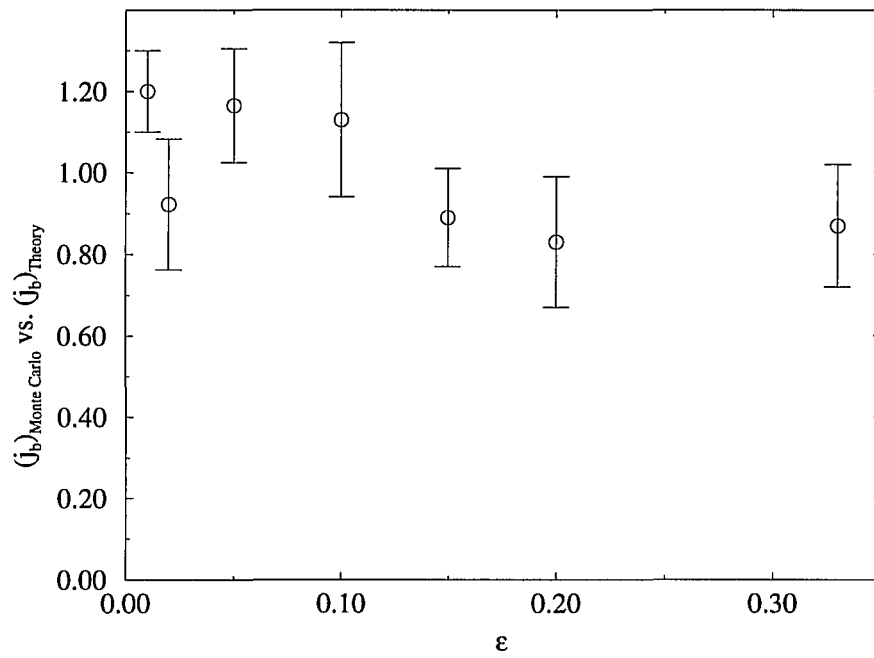


Figure 3-25: Results for the bootstrap current obtained with momentum conservation normalized to the theoretically predicted values as a function of the inverse aspect ratio.

### 3.9 Electron Contribution

So far, in our discussion of the plasma currents, we have included only one species. In principle, we have not explicitly stated whether the treatment was meant for the ions or electrons, but from the form of the collision operator it becomes clear that the species we dealt with were the ions. The reason is that we only dealt with collisions due to like species, and ignored the collisional effects of the other species. This can only be done for the ions because  $m_i \gg m_e$ . Had we wanted to calculate the electron currents, the collision operator would have to have included two collision frequencies,  $\nu_{ee}$ , the frequency of electrons scattering off of electrons, and  $\nu_{ei}$ , the frequency of electrons scattering off of ions. One form of a momentum conserving collision operator for the electrons is [19]

$$C_{m.c.}^e(f) = \frac{m_e v_{\parallel}}{B} \frac{\partial}{\partial \mu} \mu \left[ (\nu_{ee} + \nu_{ei}) v_{\parallel} \frac{\partial f}{\partial \mu} + \left( \nu_{ee} \frac{u_e B}{T_e} + \nu_{ei} \frac{u_i B}{T_i} \right) f \right]. \quad (3-248)$$

Assuming equal ion and electron temperatures, the collision operator can be written in terms of  $\lambda$  as

$$\begin{aligned} C_{m.c.}^e(f) &= \frac{\nu_{ee}}{2} \frac{\partial}{\partial \lambda} \left[ (1 - \lambda^2) \left( \frac{\partial f}{\partial \lambda} - 3 \frac{u_e}{v_e} f \right) \right] \\ &+ \frac{\nu_{ei}}{2} \frac{\partial}{\partial \lambda} \left[ (1 - \lambda^2) \left( \frac{\partial f}{\partial \lambda} - 3 \frac{v_e u_i}{v_i^2} f \right) \right] \\ &= \left( C_{m.c.}^e(f) \right)_{ee} + \left( C_{m.c.}^e(f) \right)_{ei}. \end{aligned} \quad (3-249)$$

For the operator to be momentum conserving, the electron-electron scattering part of the collision operator must obey

$$\int m v_{\parallel} \left( C_{m.c.}^e(f) \right)_{ee} d^3 \mathbf{v} = \int m v_i \lambda \frac{\nu_{ee}}{2} \frac{\partial}{\partial \lambda} \left[ (1 - \lambda^2) \left( \frac{\partial f}{\partial \lambda} - 3 \frac{u_e}{v_e} f \right) \right] d^3 \mathbf{v} = 0 \quad (3-250)$$

with the part of the operator representing electrons scattering on ions vanishing if the electrons are isotropic in a frame moving with the ion parallel velocity  $u_i$ . In other words, the expression for  $u_e$  obtained through Eq. (3-250) in the collision operator

ensures that  $C_{m.c.}^e(f)$  vanishes for electrons moving along the magnetic field lines with the same velocity as the ions. The collision frequencies of the electrons scattering on electrons and electrons scattering on ions are approximately equal,  $\nu_{ee} \approx \nu_{ei} = \nu_e$ .

In analogy to the procedure described in the previous section, the momentum conserving operator for the electrons becomes

$$C_{m.c.}(f) = C_m(f) + 3\nu_e \frac{u_e}{v_e} \lambda + 3\nu_e \frac{v_e u_i}{v_i^2} \lambda \quad (3-251)$$

with  $\nu$  in  $C_m(f)$  replaced by  $(\nu_{ee} + \nu_{ei}) \approx 2\nu_e$  and the local parallel flow velocity for the species  $s$  given by

$$u_s(\psi, \theta) = \frac{1}{q_s} \left[ \left( \frac{j_{b.s}}{B_0} \right)_s + 2 \left( \frac{j_{P.S.}^1}{B_0} \right)_s + 2 \left( \frac{j_{P.S.}^2}{B_0} \right)_s + \dots \right] B(\psi, \theta). \quad (3-252)$$

Again, in analogy with the previous section, we arrive at the prescription for updating  $\delta$  for every electron after each timestep given by

$$\Delta\delta = -\Delta\psi + 3\nu_e \frac{u_e}{(v_0)_e} \lambda_e \Delta\tau + 3\nu_e \frac{v_e u_i}{(v_0)_i^2} \lambda \Delta\tau. \quad (3-253)$$

The Monte Carlo calculation of the electron bootstrap and Pfirsch–Schlüter currents can be done in two ways. The electron collision operator depends on the ion flow, but, as previously mentioned, the ion collision operator does not depend on the flow of the electrons. The ion bootstrap and Pfirsch–Schlüter currents can therefore be calculated without the electrons. This fact enables us to first run a simulation for the ions only and, after the currents have reached their asymptotic values, run the simulation for the electrons using the asymptotic values for the ion currents in Eq. (3-253). This method allows the electron currents to reach their asymptotic values sooner, but it does not allow any feedback from the electrons on the ions because the ion simulation is completed before the electron simulation has begun. That disadvantage prohibits us from using this method for the enforcement of the quasi-neutrality condition through electrostatic potentials [section 3.12]. The other way to calculate the electron currents

is to run the simulation for the ions for some fraction of a collision time and then switch to the electrons until they have reached the same time, etc. This can be done rather easily because the simulation already has an artificial time-boundary built in it which has to be reached by all particles each time the currents are calculated [cf. section 3.2]. The advantage of running the simulation concurrently for both species is the possibility of feedback of electron behavior on ions; the disadvantage is the longer time needed for the electron currents to reach their equilibrium values. Figure (3-26) shows the electron and ion bootstrap currents vs. time as a comparison of the two ways of executing a dual species Monte Carlo code.

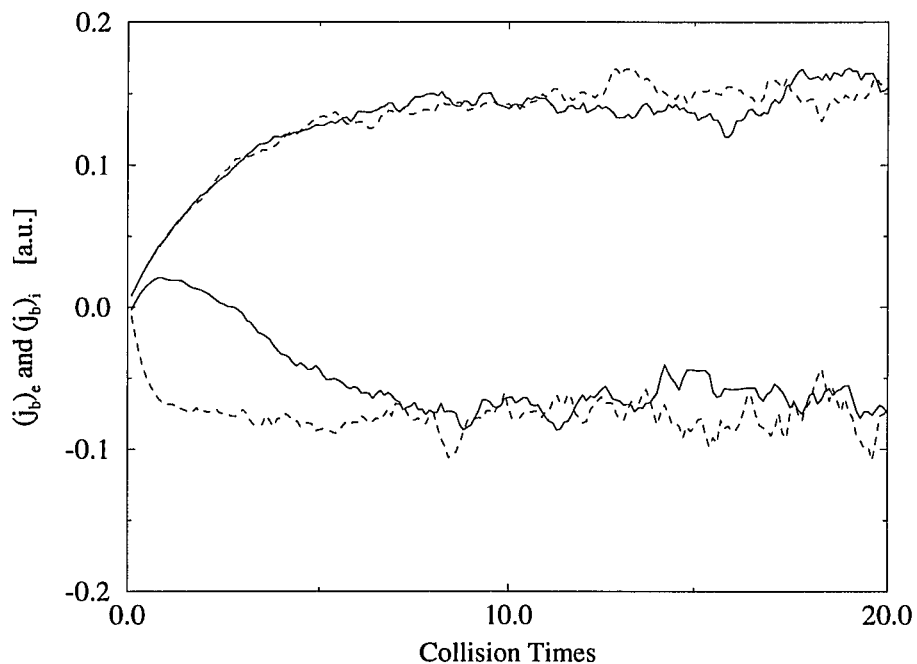


Figure 3-26: The ion and electron bootstrap currents obtained by executing the code concurrently for both species (solid line) and first for the ions and then electrons using the ion results in the electron code (dashed line).

The total bootstrap current is the sum of the ion and electron bootstrap currents

$$j_b = \left\langle \frac{j_{\parallel}}{B} \right\rangle_i B_0 + \left\langle \frac{j_{\parallel}}{B} \right\rangle_e B_0. \quad (3-254)$$

Ironically, the bootstrap current calculated using both ion and electron momentum conservation is almost identical to the bootstrap current calculated using a single species, non-momentum conserving Monte Carlo code. Figure 3-27 is a comparison of the single-species, non-momentum conserving bootstrap current and the sum of the momentum conserving ion and electron currents for several different values of collisionality and aspect ratio.

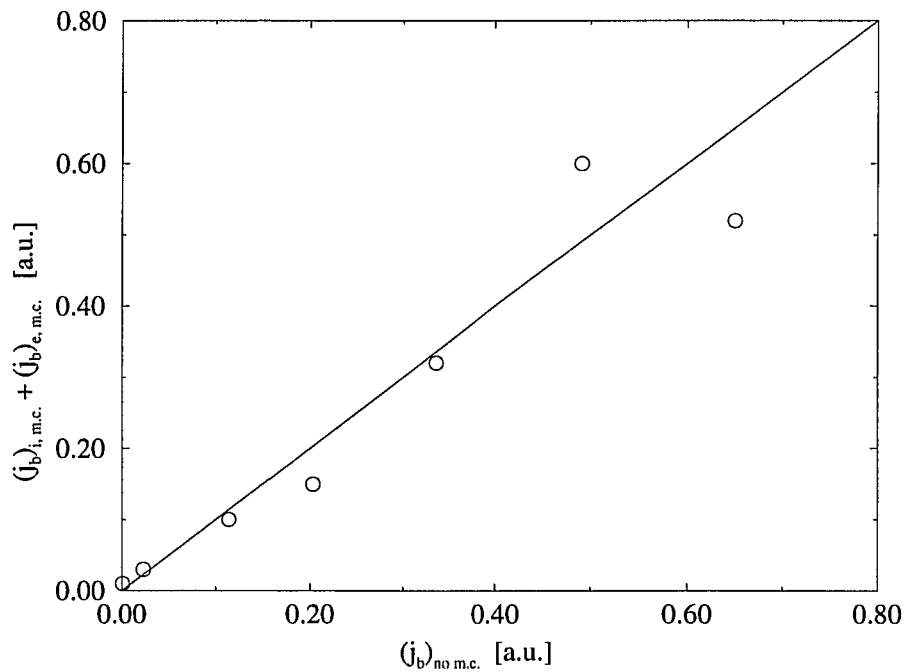


Figure 3-27: A comparison of the results for the bootstrap current obtained from a single-species, non-momentum conserving simulation and the sum of the momentum conserving ion and electron contributions to the bootstrap current for several values of collision frequency and aspect ratio. The solid line marks the unit slope.

### 3.10 Non-Axisymmetric Configuration

So far, our discussion involved magnetic field configurations that did not exhibit a toroidal, or  $\phi$ , dependence. In practice, however, a finite number of toroidal field coils prevents perfect axisymmetry. To model magnetic field dependence in the toroidal direction, we introduce a parameter called the toroidal ripple,  $\delta_r$ . The Fourier expansion for the magnetic field strength [cf. Eqs. (3-163) and (3-164)] now takes on the form

$$B = B_0 \left( 1 + c_{01} \epsilon \sqrt{\frac{\psi}{\psi_a}} \cos(\theta) + c_{02} \epsilon^2 \left( \frac{\psi}{\psi_a} \right) \cos(2\theta) + c_{10} \delta_r \sqrt{\frac{\psi}{\psi_a}} \cos(\phi) + c_{20} \delta_r^2 \left( \frac{\psi}{\psi_a} \right) \cos(2\phi) \right)^{-\frac{1}{2}} \quad (3-255)$$

where we have included the first two  $\cos(n\phi)$  terms, the same number as the  $\cos(m\theta)$  terms.

In non-axisymmetric systems, the rotational transform  $\iota$  cannot be approximated by a constant, but rather must be a function of the radial variable  $\psi$ . In tokamaks, the rotational transform is about one on the magnetic axis ( $\psi = 0$ ) and about one-third at the plasma edge ( $\psi = \psi_a$ ). The expression for the rotational transform that we have used is

$$\iota = \iota_0 + \frac{\psi - \psi_m}{\psi_a} \iota_1, \quad (3-256)$$

where  $\psi_m$  denotes the middle of the annulus we are considering. Figure 3-28 shows  $\iota$  as a function of radial position with  $\iota_0 = 0.43$  and  $\iota_1 = -0.67$ .

The toroidal ripple causes a shift of the banana bounce point across magnetic surfaces. The shift can be obtained by calculating the cross surface drift with and without ripple and taking the difference. It is given by Wu and White [20]

$$\Delta\psi_r = \frac{\sqrt{2\pi N}}{\iota \epsilon^{3/2}} \rho \delta_r, \quad (3-257)$$

with  $\rho$  the Larmor radius,  $\epsilon$  the inverse aspect ratio and  $N$  the toroidal mode number (in our case  $N = 2$ ). Goldston, White and Boozer [27] have shown that if the ripple

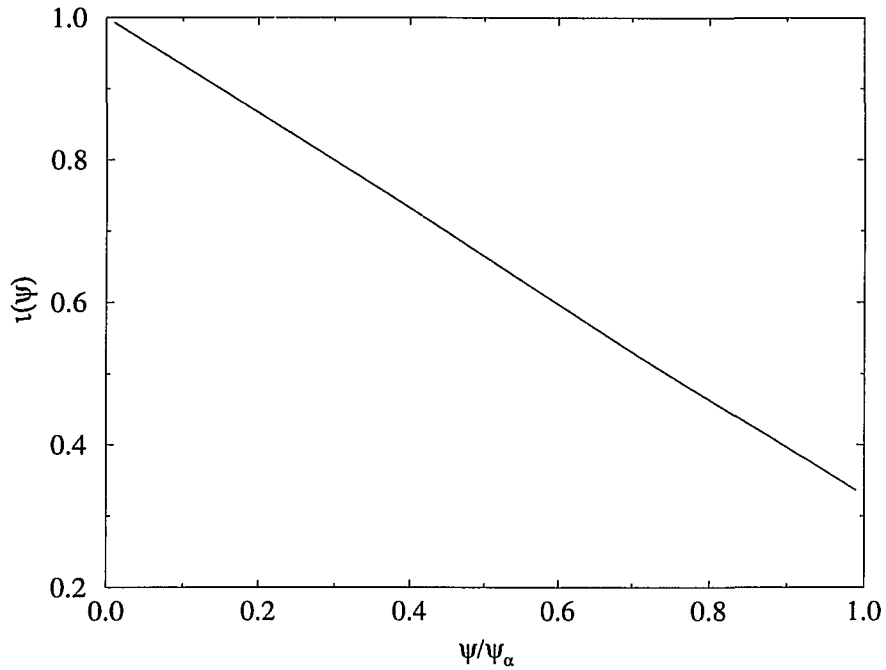


Figure 3-28: The rotational transform  $\nu$  as a function of the normalized radial distance.  $\psi_\alpha$  gives the toroidal flux at the edge of the plasma.

exceeds a certain limit, in particular when the banana tip shift due to the perturbation is of order the banana width, the radial drifts dominate and the banana orbits of the trapped particles become stochastic. Figure 3-29 shows the effect of the toroidal ripple on the banana orbits.

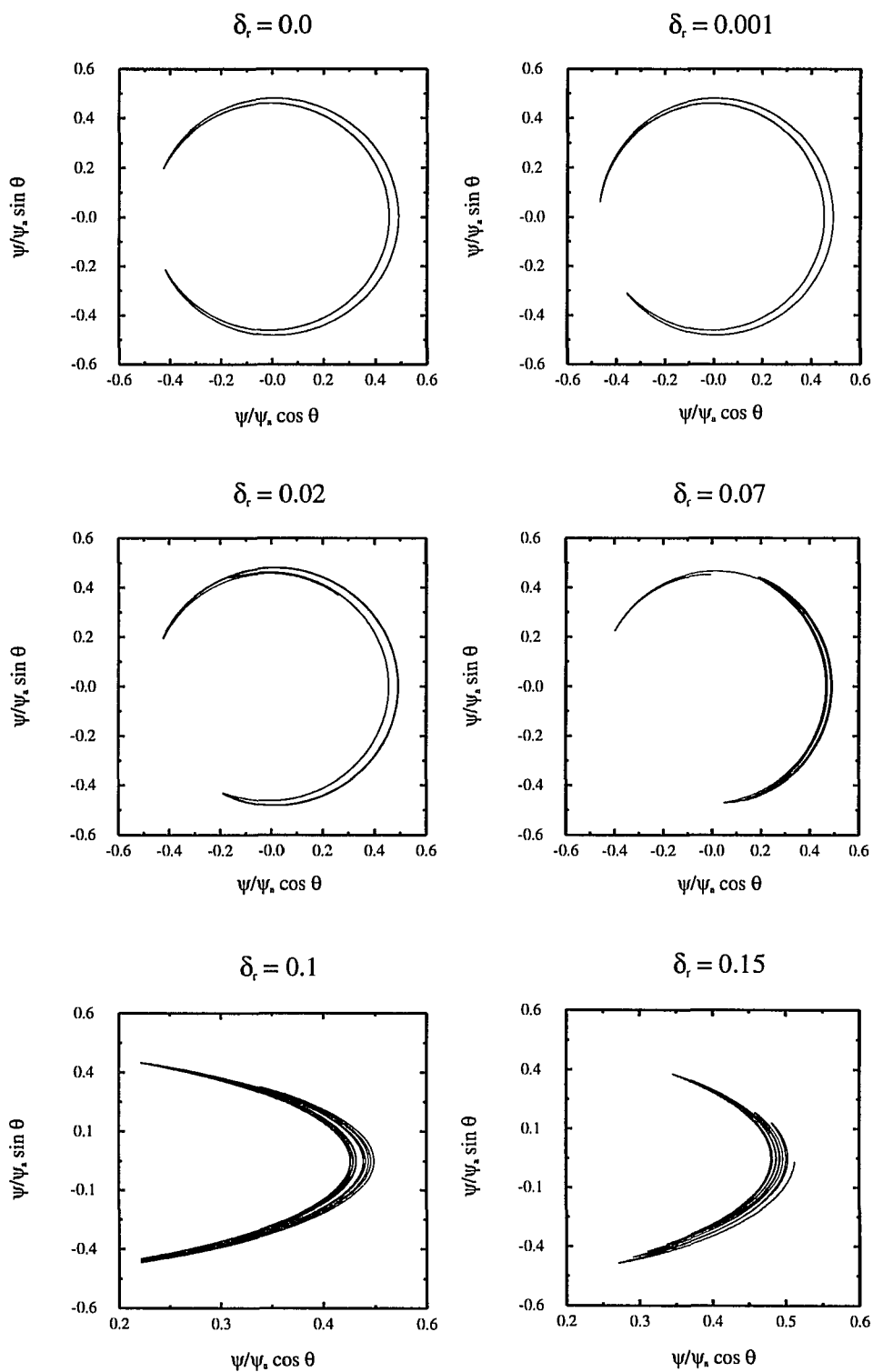


Figure 3-29: The destruction of the banana orbits by the toroidal ripple  $\delta_r$ .



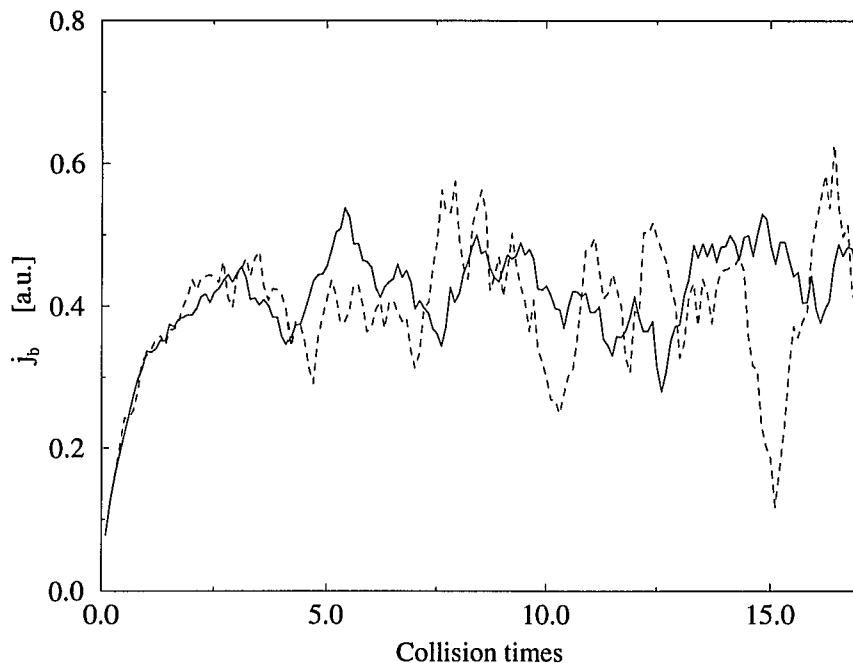


Figure 3-30: The bootstrap current vs. time in an axisymmetric configuration and a configuration with a small toroidal field ripple. The solid line represents  $\delta_r = 0.0$ , the dashed line represents  $\delta_r = 0.02$ .

One of the consequences of the destruction of the banana orbits is that the parallel and anti-parallel moving particles are not forced into orbits which set up the bootstrap current [cf. section 3.4]. We expect, therefore, to see a destructive effect on the bootstrap current due to the toroidal ripple. Figures 3-30 and 3-31 examine the influence of the toroidal ripple on the bootstrap current obtained from the  $\delta f$  simulation.

Figure 3-30 shows the bootstrap current as a function of time for  $\delta_r = 0$  and  $\delta_r = 0.02$ . We can see that the bootstrap current grows until it reaches the asymptotic value and then oscillates around that value. The magnitude of the asymptotic value is not affected by the toroidal ripple. Figure 3-31 shows the bootstrap current as a function of time for  $\delta_r = 0.07$  and  $\delta_r = 0.15$ . The bootstrap current exhibits much stronger fluctuations and at some points even changes sign. It is important to note that

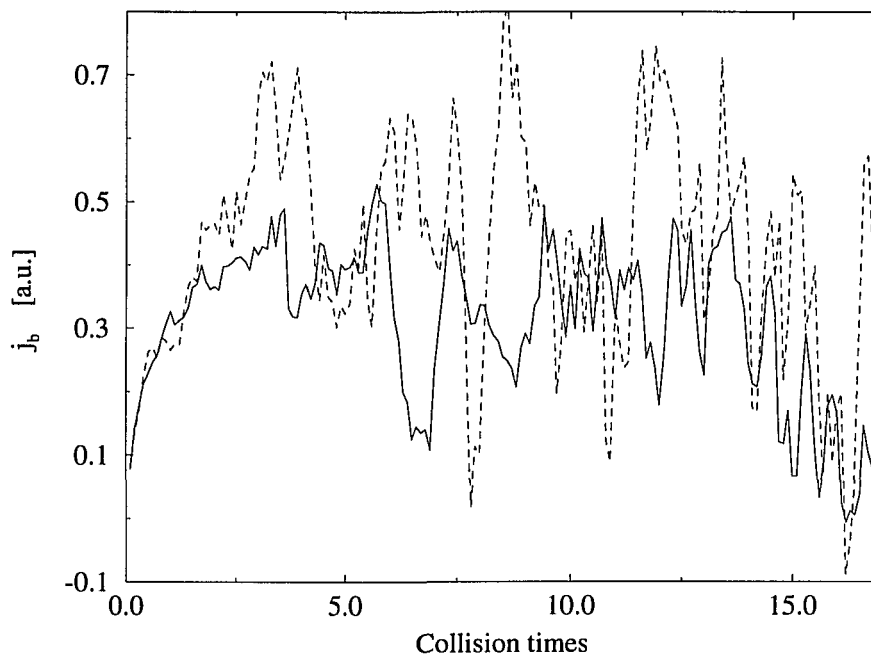


Figure 3-31: The bootstrap current vs. time for two values of the toroidal field ripple. The solid line represents  $\delta_r = 0.07$  and the dashed line represents  $\delta_r = 0.15$ .

all six simulations have been executed with the same number of particles and the large fluctuations do not represent statistical fluctuations.

### 3.11 New Set of Variables

The poloidal current  $G$  and the toroidal current  $I$  (which was previously neglected) are also taken to be functions of the radial position  $\psi$ . The existence of the toroidal current does not allow for the simple relationship between  $\psi$  and the poloidal momentum  $p_\theta$  which is calculated in every time step. To obtain values for the currents and the rotational transform using the canonical toroidal and poloidal positions and momenta as the variables used in the Runge-Kutta algorithm, the code would have to calculate

$\psi$  as a function of the old values of  $(p_\theta, p_\phi, G(\psi), I(\psi), \iota(\psi))$  [cf. Eqs. (2-100) and (2-101)] and then use the newly calculated  $\psi$  to obtain new values for  $G(\psi)$ ,  $I(\psi)$  and  $\iota(\psi)$ . Clearly this approach is inefficient. A much better approach is to define a new set of variables to be used in the Runge–Kutta algorithm which includes  $\psi$  as one of the variables. The new set of variables was introduced in section 2.4 and included the poloidal and toroidal angles, the toroidal flux and the parallel gyro-radius  $(\theta, \phi, \psi, \rho_\parallel)$ . The explicit equations of motion for the new variables to be used in the Runge–Kutta routine can be obtained from Eqs. (2-102), (2-105), (2-106), (2-107), (2-116) and (2-124).

$$\begin{aligned} \dot{\theta} = & -\left(\frac{2\pi}{q}\right)^2 \frac{1}{g + \iota i} \left[ \left( \frac{(2\pi)^2 B^2}{m} \rho_\parallel \right) \left( \rho_\parallel \left( \frac{\partial g(\psi)}{\partial \psi} \right) - \frac{q}{2\pi} \iota(\psi) \right) \right. \\ & \left. + g(\psi) \left( \frac{(2\pi)^2 \rho_\parallel^2 B^2}{m} + \mu B \right) \frac{1}{B} \left( \frac{\partial B}{\partial \psi} \right) \right] \end{aligned} \quad (3-258)$$

$$\begin{aligned} \dot{\phi} = & -\left(\frac{2\pi}{q}\right)^2 \frac{1}{g + \iota i} \left[ i(\psi) \left( \frac{(2\pi)^2 \rho_\parallel^2 B^2}{m} + \mu B \right) \frac{1}{B} \left( \frac{\partial B}{\partial \psi} \right) \right. \\ & \left. - \left( \rho_\parallel \left( \frac{\partial i(\psi)}{\partial \psi} \right) - \frac{q}{2\pi} \iota(\psi) \right) \left( \frac{(2\pi)^2 B^2}{m} \rho_\parallel \right) \right] \end{aligned} \quad (3-259)$$

$$\begin{aligned} \dot{\psi} = & -\left(\frac{2\pi}{q}\right)^2 \frac{1}{g + \iota i} \left[ g(\psi) \left[ \left( \frac{(2\pi)^2 \rho_\parallel^2 B^2}{m} + \mu B \right) \frac{1}{B} \left( \frac{\partial B}{\partial \theta} \right) + q \left( \frac{\partial \Phi}{\partial \theta} \right) \right] \right. \\ & - i(\psi) \left[ \left( \frac{(2\pi)^2 \rho_\parallel^2 B^2}{m} + \mu B \right) \frac{1}{B} \left( \frac{\partial B}{\partial \phi} \right) + q \left( \frac{\partial \Phi}{\partial \phi} \right) \right] \\ & \left. - \frac{q}{2\pi} V(t) i(\psi) \right] \end{aligned} \quad (3-260)$$

$$\begin{aligned} \dot{\rho}_\parallel = & \left(\frac{2\pi}{q}\right)^2 \frac{1}{g + \iota i} \left[ \left[ \rho_\parallel \left( \frac{\partial g(\psi)}{\partial \psi} \right) - \frac{q}{2\pi} \iota(\psi) \right] \left[ \left( \frac{(2\pi)^2 \rho_\parallel^2 B^2}{m} + \mu B \right) \frac{1}{B} \left( \frac{\partial B}{\partial \theta} \right) + q \left( \frac{\partial \Phi}{\partial \theta} \right) \right] \right. \\ & - \left[ \rho_\parallel \left( \frac{\partial i(\psi)}{\partial \psi} \right) + \frac{q}{2\pi} \right] \left[ \left( \frac{(2\pi)^2 \rho_\parallel^2 B^2}{m} + \mu B \right) \frac{1}{B} \left( \frac{\partial B}{\partial \phi} \right) \right] \\ & \left. - \frac{q}{2\pi} V(t) \left[ \left( \frac{\partial i(\psi)}{\partial \psi} \right) + \frac{q}{2\pi} \right] \right] \end{aligned} \quad (3-261)$$

The partial derivatives are taken holding the other variables constant. Using Eq. (3-255) for the magnetic field we have

$$\begin{aligned} \left(\frac{\partial B}{\partial \psi}\right) &= \left(\frac{\partial B}{\partial \psi}\right)_{\theta, \phi, \rho_{\parallel}} \\ &= -\frac{1}{2} \left(\frac{B}{B_0}\right)^2 \left( \frac{c_{01}\epsilon}{2\sqrt{\psi\psi_a}} \cos \theta + \frac{c_{02}\epsilon^2}{\psi_a} \cos 2\theta + \frac{c_{10}\delta_r}{2\sqrt{\psi\psi_a}} \cos \phi + \frac{c_{20}\delta_r^2}{\psi_a} \cos 2\phi \right) \end{aligned} \quad (3-262)$$

$$\begin{aligned} \left(\frac{\partial B}{\partial \theta}\right) &= \left(\frac{\partial B}{\partial \theta}\right)_{\psi, \phi, \rho_{\parallel}} \\ &= \frac{1}{2} \left(\frac{B}{B_0}\right)^2 \left( c_{01}\epsilon \sqrt{\frac{\psi}{\psi_a}} \sin \theta + 2c_{02}\epsilon^2 \frac{\psi}{\psi_a} \sin 2\theta \right) \end{aligned} \quad (3-263)$$

$$\begin{aligned} \left(\frac{\partial B}{\partial \phi}\right) &= \left(\frac{\partial B}{\partial \phi}\right)_{\psi, \theta, \rho_{\parallel}} \\ &= \frac{1}{2} \left(\frac{B}{B_0}\right)^2 \left( c_{10}\delta_r \sqrt{\frac{\psi}{\psi_a}} \sin \phi + 2c_{20}\delta_r^2 \frac{\psi}{\psi_a} \sin 2\phi \right) \end{aligned} \quad (3-264)$$

### 3.12 Enforcement of Quasi-Neutrality

The condition of quasi-neutrality in a plasma means that the ion and electron densities are almost equal everywhere in the plasma. Small deviations from charge neutrality are called micro-instabilities. The fluctuations in densities leading to micro-instabilities in toroidal confinement devices result from the difference in radial excursions of the particles from their original toroidal flux surfaces. The excursions are proportional to the Larmor radius of the particles and are much greater for ions than for electrons [cf. Eqs. (2-78) and (2-79)]. The resulting electric fields associated with charge separation in the magnetic surfaces produce  $\mathbf{E} \times \mathbf{B}$  drifts across the surfaces, thus weakening confinement. The resulting transport is referred to as anomalous transport because it exceeds the theoretically predicted values for transport by several times for the ions and

about two orders of magnitude for the electrons. We will try to reproduce anomalous transport using the  $\delta f$  code by explicitly enforcing quasi-neutrality. We will discuss two different methods which lead to enforcement of the quasi-neutrality condition.

The density  $n$  at each point on the surface can be written as a sum of the average density on the surface  $\bar{n}$  and the deviation from the average density  $\Delta n$ .

$$n = \bar{n} + \Delta n, \quad (3-265)$$

with

$$\bar{n} = \frac{\int f d^3\mathbf{v} d\theta d\phi}{d\theta d\phi} \quad (3-266)$$

and

$$\Delta n = \int (f - F_M) d^3\mathbf{v} = \frac{\bar{n}}{2} \int \delta d\lambda. \quad (3-267)$$

The density fluctuations on the magnetic surface normalized to the average density on the surface can be written in a Fourier decomposed form in  $\theta$  and  $\phi$  for each species

$$\left( \frac{\Delta n}{\bar{n}} \right)_s = \sum_{n,m} (N_{n,m})_s e^{i(n\phi - m\theta)} \quad (3-268)$$

with

$$(N_{m,n})_s = \int \frac{\Delta n}{\bar{n}} e^{-i(n\phi - m\theta)} d\theta d\phi = \frac{1}{2} \int \delta_s e^{-i(n\phi - m\theta)} d\lambda d\theta d\phi. \quad (3-269)$$

The electron density is approximated by the Gibb's distribution

$$n_e = \bar{n}_e e^{|q|\Phi(\psi,\theta,\phi)/T_e} \quad (3-270)$$

which can be written in terms of the deviation from the average density

$$\left( \frac{n}{\bar{n}} \right)_e = e^{|q|\Phi(\psi,\theta,\phi)/T_e} \quad (3-271)$$

$$\left( 1 + \frac{\Delta n}{\bar{n}} \right)_e \approx 1 + q\Phi(\psi, \theta, \phi)/T_e \quad (3-272)$$

Combining Eq. (3-268) with Eq.(3-272) gives a Fourier representation for the so-called ambipolar, or self-consistent, electrostatic potential.

$$\Phi(\psi, \theta, \phi) = -\frac{T}{q} \sum_{m,n} (N_{m,n})_s e^{i(n\phi - m\theta)} \quad (3-273)$$

The quasi-neutrality condition can then be explicitly enforced by inserting  $\Phi(\psi, \theta, \phi)$  with  $(N_{n,m})_i$  into the Hamiltonian for the next iteration and calculating  $(\Delta n/n)$  and  $\Phi(\psi, \theta, \phi)$  again. Repeating this procedure will eventually force  $(N_{m,n})_i = (N_{m,n})_e$  thereby explicitly enforcing the quasi-neutrality condition.

In practice, the deviations from the density were decomposed using the trigonometric functions keeping the first three sine and cosine terms for each direction

$$\begin{aligned}
\frac{\Delta n}{n} &= C_{01} \cos \theta + C_{02} \cos 2\theta + C_{03} \cos 3\theta \\
&+ S_{01} \sin \theta + S_{02} \sin 2\theta + S_{03} \sin 3\theta \\
&+ C_{10} \cos \phi + C_{20} \cos 2\phi + C_{30} \cos 3\phi \\
&+ S_{10} \sin \phi + S_{20} \sin 2\phi + S_{30} \sin 3\phi
\end{aligned} \tag{3-274}$$

with the coefficients calculated from

$$C_{0m} = \frac{1}{N} \sum_i \delta_i \cos(m\theta) \tag{3-275}$$

$$S_{0m} = \frac{1}{N} \sum_i \delta_i \sin(m\theta) \tag{3-276}$$

$$C_{n0} = \frac{1}{N} \sum_i \delta_i \cos(n\phi) \tag{3-277}$$

$$S_{n0} = \frac{1}{N} \sum_i \delta_i \sin(n\phi) \tag{3-278}$$

where the sum is over all particles and  $N$  is the total number of particles in the simulation. The expression for the weight  $\delta_i$  for each particle is obtained by following the derivation given in section 3.2 with  $E_K$  replacing  $H$  as the variable for the particle kinetic energy. We now let  $H$  be the total energy of the particle

$$H = \frac{1}{2}mv^2 + q\Phi(\psi, \theta, \phi). \tag{3-279}$$

Equation (3-155) can now be written as

$$\begin{aligned}
\frac{d \ln F_M}{dt} &= \frac{\partial \ln F_M}{\partial \psi} \frac{d\psi}{dt} + \frac{\partial \ln F_M}{\partial E_K} \frac{dE_K}{dt} \\
&= \left( \frac{\partial \zeta}{\partial \psi} - \frac{mv^2}{2} \frac{\partial \beta}{\partial \psi} \right) \frac{d\psi}{dt} - \beta \frac{dE_K}{dt}.
\end{aligned} \tag{3-280}$$

The time derivative of the kinetic energy is

$$\frac{dE_K}{dt} = \frac{dH}{dt} - q \frac{d\Phi(\psi, \theta, \phi)}{dt} = 0 - \mathbf{v} \cdot \nabla \Phi(\psi, \theta, \phi) \quad (3-281)$$

For the potential within a constant pressure surface ( $\psi = \text{const.}$ )

$$\mathbf{v} \cdot \nabla \Phi(\psi, \theta, \phi) = \dot{\theta} \frac{\partial \Phi}{\partial \theta} + \dot{\phi} \frac{\partial \Phi}{\partial \phi} \quad (3-282)$$

and the Fokker-Planck equation given in Eq. (3-157) now contains the extra terms consisting of the partial derivatives of the electric potential with respect to  $\theta$  and  $\phi$

$$\frac{d\hat{f}}{dt} + \left( \frac{d\zeta}{d\psi} - E_K \frac{d\beta}{d\psi} \right) \dot{\psi} + \beta \left( \frac{\partial \Phi}{\partial \theta} \dot{\theta} + \frac{\partial \Phi}{\partial \phi} \dot{\phi} \right) = C_m(f). \quad (3-283)$$

The extra terms are carried over into the prescription for updating  $\delta$  after every timestep which now reads

$$\Delta \delta = -\Delta \psi + 3\nu_i \frac{u(\psi, \theta)_i}{v_0} \lambda \Delta \tau + \frac{3}{m_i v_{0i}^2} \left( \frac{\partial \Phi}{\partial \theta} \Delta \theta + \frac{\partial \Phi}{\partial \phi} \Delta \phi \right) \quad (3-284)$$

for the ions and

$$\Delta \delta = -\Delta \psi + 3\nu_e \frac{u_e}{(v_0)_e} \lambda_e \Delta \tau + 3\nu_e \frac{v_e u_i}{(v_0)_i^2} \lambda \Delta \tau + \frac{3}{m_e v_{0e}^2} \left( \frac{\partial \Phi}{\partial \theta} \Delta \theta + \frac{\partial \Phi}{\partial \phi} \Delta \phi \right) \quad (3-285)$$

for the electrons. The expression for the ion density fluctuations obtained from Eq. (3-274) is now used to calculate the new potential

$$\Phi(\psi, \theta, \phi) = \frac{T}{q} \left( \frac{\Delta n}{\bar{n}} \right)_i \quad (3-286)$$

for the ions and electrons which will be inserted into the Hamiltonian [Eq. (3-279)] in the next timestep. The potential obtained from the ion fluctuations will force the electrons into orbits with greater radial excursions (on the order of the ion radial excursions), thereby enforcing quasi-neutrality. Figure 3-32 shows the  $S_{01}$  Fourier coefficient of the density for the ions and electrons with (dashed-lines) and without (solid lines) enforcement of quasi-neutrality. The density component for the electrons without enforcement of quasi-neutrality is of order square-root of the mass ratio smaller than the

ion component, as expected. With the quasi-neutrality condition enforced, however, the electrons density component is of order of the ion density component which means that the electrons are performing radial excursions with magnitude comparable to the ions excursions.

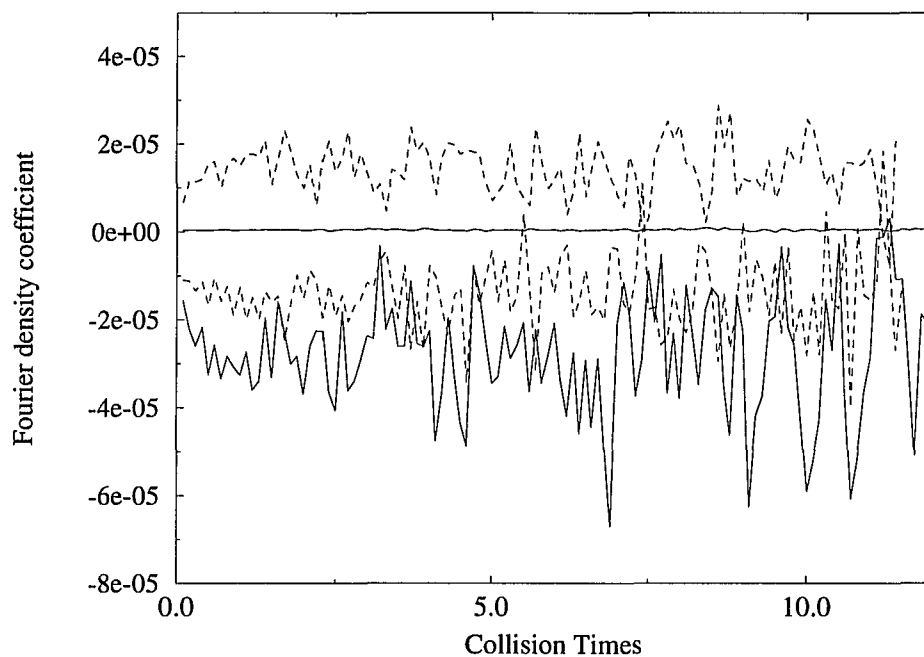


Figure 3-32: The  $S_{01}$  Fourier coefficient of the density for the ions and electrons. The solid lines represent the coefficients without enforcement of quasi-neutrality. The dashed lines show the density coefficients with quasi-neutrality enforcement.

This method of enforcing quasi-neutrality has a drawback. To understand the drawback we will discuss the two parts of the displacement of the particles from their original flux surfaces; the even and odd parts in  $v_{\parallel}$ . The even part of the displacement measures the non-closure of the banana orbits. Trapped particles bounce back and forth between two points on the field line (precessing slowly) with their parallel velocity switching sign at the reflection point, hence their displacement is mostly even in  $v_{\parallel}$ . Passing particles, however, circulate around the torus keeping the same sign in parallel velocity, hence



their displacement is odd in  $v_{\parallel}$ . Only the odd part of the displacement leads to a net parallel current. The magnitude of the parallel current is determined by the odd part of the trapped particles, even though most of the current itself is carried by the passing particles. The part of the displacement which is even in  $v_{\parallel}$  does not contribute to the current. Let us now look at Eqs. (3-284) and (3-285). The last term—associated with the ambipolar potential—in both equations changes the magnitude of  $\delta$ . As previously stated, it should only change the part in  $\delta$  which is even in  $v_{\parallel}$  because the magnitude of the parallel current should not be affected. Due to the small number of particles in the simulation, the contribution to the odd part of  $\delta$  may not be able to cancel itself out completely, as it should. In particular, for the ions, the last term in Eq. (3-284) is of order of the other terms in the equation, so this effect is not great. For the electrons, however, the effect of the ambipolar field is greater than the other terms and the ambipolar field increases the magnitude of the parallel current. This effect is due to the small number of particles and the statistical problems associated with few particle simulations and is inversely proportional to the number of particles included in the simulation. Increasing the number of particles in the simulation is not a very efficient solution, however. There exist another possibility. The method described below allows us to enforce quasi-neutrality without the problems encountered in the previous method.

In the new method, the prescription for updating  $\delta$  will not be changed from the original, non-quasi-neutrality enforcing, prescription since this created the problem in the previous method. To eliminate the unwanted terms in Eqs. (3-284) and (3-285), let us look back to Eq. (3-280). We realize that the last term in that equation must vanish because it is that term which leads to the extra terms in Eqs. (3-284) and (3-285). The term can be made to vanish if the expression for the Maxwellian involves the total energy  $H$  rather than the kinetic energy only because  $dH/dt = 0$ . Let us define the

Maxwellian as

$$F_M(\psi, H) \equiv c_0 \exp \left( -\frac{mv^2}{2T} + \frac{q\Phi(\psi, \theta, \phi)}{T} - \frac{q\bar{\Phi}(\psi)}{T} \right), \quad (3-287)$$

with

$$\bar{\Phi}(\psi) \equiv \frac{\int \Phi(\psi, \theta, \phi) d\theta d\phi}{\int d\theta d\phi} \quad (3-288)$$

the potential averaged over the pressure surface. Let us define the deviation from the surface averaged potential  $\tilde{\Phi}(\psi, \theta, \phi) \equiv \Phi(\psi, \theta, \phi) - \bar{\Phi}(\psi)$ . For simplicity and to be consistent with previous work, let us replace the Maxwellian energy distribution with a monoenergetic distribution.  $F_M(\psi, H)$  can now be written as

$$F_M(\psi, H) = c_0 \delta(v - v_0) \exp \left( \frac{q\tilde{\Phi}(\psi, \theta, \phi)}{T} \right). \quad (3-289)$$

The density  $n(\psi, \theta, \phi)$  at every point of the surface is again written as the sum of the average density on the surface  $\bar{n}(\psi)$  and the deviation from the average density  $\Delta n(\psi, \theta, \phi)$

$$\begin{aligned} n &= \bar{n} + \Delta n \\ &= \int F_m \exp(\hat{f}) d^3 \mathbf{v} = \int c_0 \delta(v - v_0) \exp \left( \frac{q\tilde{\Phi}}{T} + \hat{f} \right) d^3 \mathbf{v} \\ &= n_0 + \frac{n_0}{2} \int \left( \frac{q\tilde{\Phi}}{T} + \hat{f} \right) d\lambda \end{aligned} \quad (3-290)$$

In analogy with Eqs. (3-268) and (3-269) the Fourier decomposed form of  $(\Delta n/n)_s$  is

$$\left( \frac{\Delta n}{\bar{n}} \right)_s = \sum_{n,m} (N_{n,m})_s e^{i(n\phi - m\theta)} \quad (3-291)$$

with

$$(N_{m,n})_s = \int \frac{\Delta n}{\bar{n}} e^{-i(n\phi - m\theta)} d\theta d\phi = \frac{1}{2} \int \left( \delta_s + \frac{q\tilde{\Phi}}{T} \right) e^{-i(n\phi - m\theta)} d\lambda d\theta d\phi. \quad (3-292)$$

In practice this means that the Fourier coefficients are now obtained from

$$C_{0m} = \frac{1}{N} \sum_i \left( \delta_i + \frac{q\tilde{\Phi}}{T} \right) \cos(m\theta) \quad (3-293)$$

$$S_{0m} = \frac{1}{N} \sum_i \left( \delta_i + \frac{q\tilde{\Phi}}{T} \right) \sin(m\phi) \quad (3-294)$$

$$C_{n0} = \frac{1}{N} \sum_i \left( \delta_i + \frac{q\tilde{\Phi}}{T} \right) \cos(n\theta) \quad (3-295)$$

$$S_{n0} = \frac{1}{N} \sum_i \left( \delta_i + \frac{q\tilde{\Phi}}{T} \right) \sin(n\phi). \quad (3-296)$$

This method allows us to keep the old prescription for updating  $\delta$  and instead uses the ambipolar potential in the expressions used to calculate the Fourier coefficients for the density to enforce quasi-neutrality.

To study particle transport across magnetic flux surfaces, a coefficient of diffusion across surfaces is obtained. The coefficient is the sum over the squares of the radial displacement in a time  $\Delta t$  averaged over all particles.

$$D = \frac{\frac{1}{N} \sum (\Delta\psi)_i^2}{2\Delta t}. \quad (3-297)$$

The time increment  $\Delta t$  must be large enough to allow effects due to collisions to be included. A sensible choice is letting  $\Delta t = \tau_c$ .

The code correctly predicted neo-classical transport and the enhancement of transport due to toroidal ripple. Figure 3-33 shows the diffusion coefficient as a function of time for the ions and electrons. The logarithmic scale on the ordinate allows us to see that  $D_i \approx \sqrt{m_i/m_e} D_e$ , as expected [cf. subsection 2.3.4].

Figure 3-34 shows the transport of the ions as a function of the collisionality and toroidal ripple. Figure 3-35 shows the collisionality and toroidal ripple dependence of the electron transport. Both graphs show qualitative agreement with neoclassical and ripple induced transport.

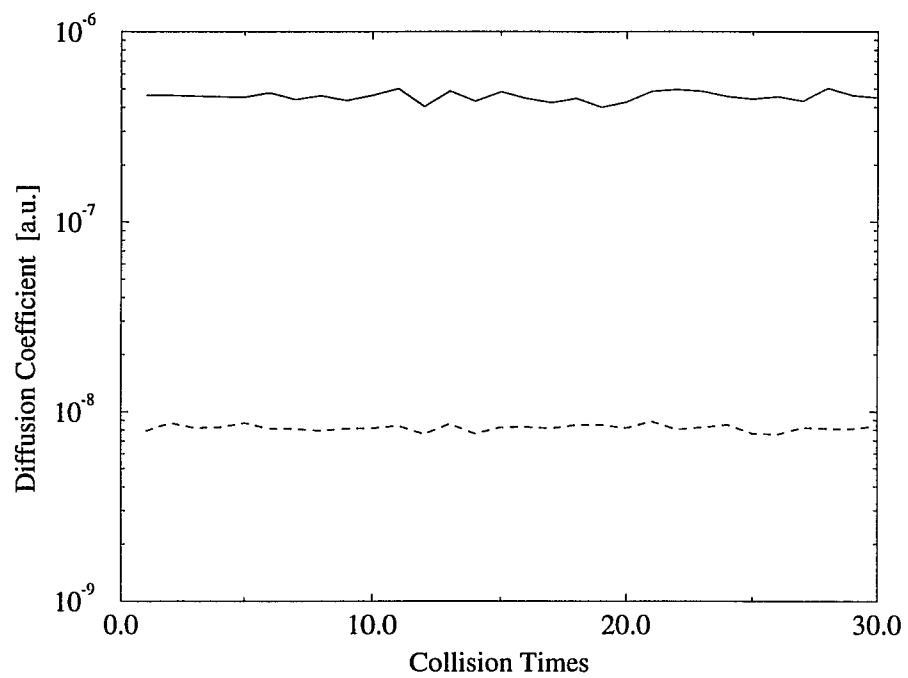


Figure 3-33: The diffusion coefficient vs. time for the ions (solid line) and electrons (dashed line).

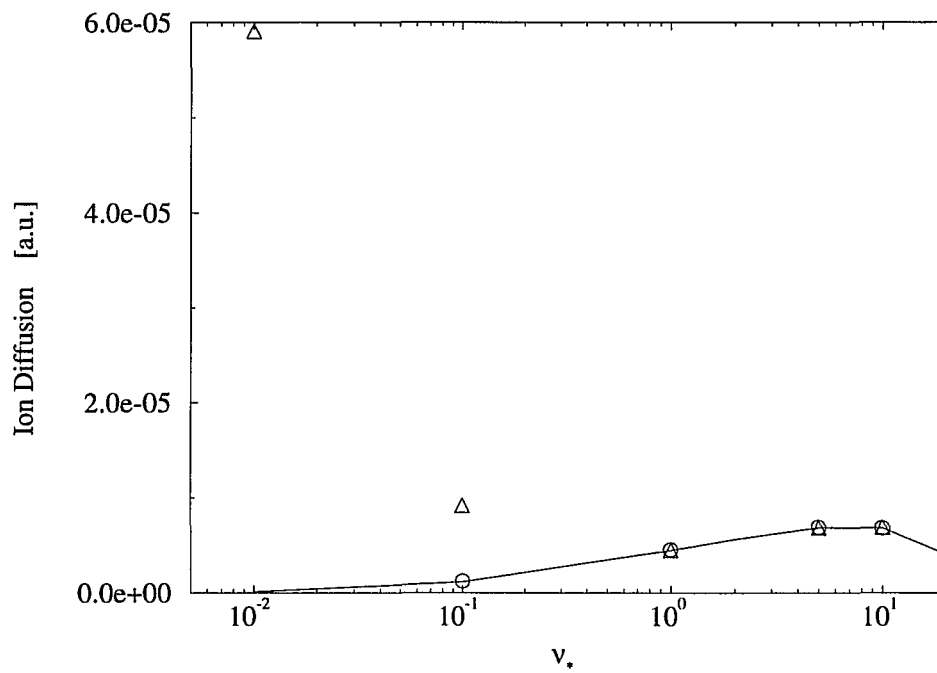


Figure 3-34: Ion transport as a function of collisionality. The solid line represents the diffusion coefficients for the ions in an axisymmetric configuration. The squares represent the diffusion coefficients for  $\delta_r = 0.01$  and the triangles represent the diffusion coefficients for  $\delta_r = 0.1$ .

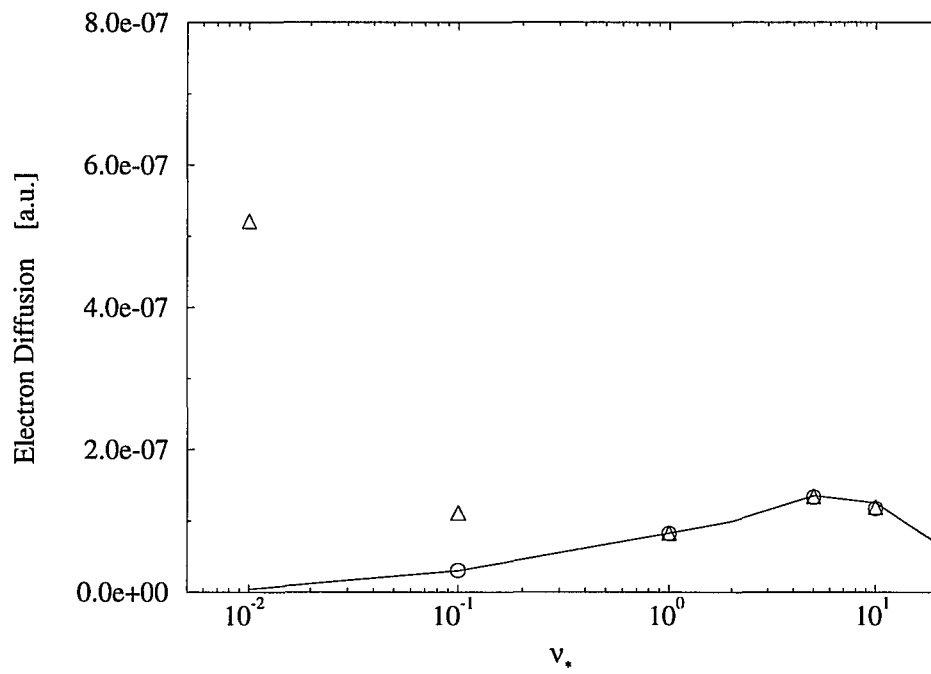


Figure 3-35: Electron transport as a function of collisionality. The solid line represents the diffusion coefficients for the electrons in an axisymmetric configuration. The squares represent the diffusion coefficients for  $\delta_r = 0.01$  and the triangles represent the diffusion coefficients for  $\delta_r = 0.1$ .

An initially surprising result was the non-dependence of the diffusion coefficient on the enforcement of the quasi-neutrality condition. To understand the reason, the Fourier density coefficients were further investigated. It turned out that the value of  $q\bar{\Phi}/T$  was much smaller than anticipated. For the ambipolar field to contribute to transport at least as much as the toroidal ripple does,  $q\bar{\Phi}/T$  must be of order the toroidal ripple  $\delta_r$ . We can obtain this result from the particle Hamiltonian with the expression for  $B$  expanded in a binomial series, keeping only the first terms in  $\epsilon$  and  $\delta_r$ . From Figures 3-34 and 3-35 we see that the toroidal ripple contributes significantly to transport around  $\delta_r \sim 0.1$ . The values of the Fourier components for  $q\bar{\Phi}/T$ , however, are much smaller, of order  $10^{-5}$ - $10^{-6}$ . Our results seem to contradict results obtained earlier by Garabedian and Taylor [28], who claim that transport is driven by explicitly enforcing quasi-neutrality. The apparent disagreement is very interesting and warrants further investigation.

# Chapter 4

## Conclusion

This dissertation introduced a new and efficient approach to calculate parameters in plasmas. The  $\delta f$  method was tested against analytic theory and previous numerical simulations for magnetically confined thermonuclear plasmas. In particular, values for the single species, non-momentum conserving bootstrap current and several components of the Pfirsch–Schlüter current in an axisymmetric magnetic field configuration were calculated and expressions for the currents in the parameter range  $0.001 \leq \nu_* \leq 100.0$  and  $0.005 \leq \epsilon \leq 0.3$ , with  $\nu_*$  the collisionality and  $\epsilon$  the inverse aspect ratio, were obtained [17]. The results agreed very well with results given by analytic theory in asymptotic limits as well as previous simulations which gave results for a smaller parameter range.

Following the preliminary tests, the simulation was upgraded to include momentum conservation, a second species and a non-axisymmetric magnetic field configuration. The currents were calculated again and gave results which agreed with results obtained analytically in asymptotic limits.

The final addition to the  $\delta f$  simulation was the explicit enforcement of the quasi-neutrality condition. This was done by calculating density fluctuations in the magnetic surfaces and creating an in-surface potential to counteract the variations in the ion and electron densities. The parameters of interest in this part were not currents, but rather the cross surface diffusion coefficients for the ions and electrons. The preliminary results seem to exclude in-surface electric potentials as one of the explanations for anomalous



transport.

The  $\delta f$  Monte Carlo code has allowed us to make efficient predictions for currents and transport coefficients. This method eliminates the need for including hundreds of thousands, or even millions of particles in Monte Carlo simulations, but instead uses a few hundred to a thousand. Most of the results presented in this dissertation came from codes executed with 500–2000 particles. The time required to simulate 1000 particles over twenty collision times is about 10 hours of  $\sim 100\%$  CPU on a SUN Sparc5 workstation. The floating point operations speed of a Sparc5 is roughly that of a Pentium90 personal computer. For many of the calculations it is not even necessary to run the simulation over twenty collision times. For example, the diffusion coefficients can be obtained after only one collision time, which, for a simulation with 1000 particles, is equivalent to 30 minutes of CPU time. The same holds true for the Pfirsch–Schlüter current and the density coefficients. The only parameter which takes several collision times to reach its final value is the bootstrap current. This phenomenon is not purely numerical. It represents the time needed by the passing particles to interact with the barely trapped particles executing their banana orbits.

In conclusion, let us remark that the  $\delta f$  method isn't by any means limited to the calculations of currents or diffusion coefficients in thermonuclear plasmas. It can be easily applied to a variety of problems, such as heat transport calculations near divertors or even astrophysical plasma problems. In short, the  $\delta f$  Monte Carlo method is a new and very efficient tool with many potential applications.

# Appendix

## The $\delta f$ Monte Carlo code

```
#include <math.h>
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>
#include <string.h>

# define MOMENTUM_CONSERVATION on      /*****
/* # define QUASI_NEUTRALITY on */     /* Comment out the things not */
/* # define LOOP_VOLTAGE 1.0e-16 */    /* to be included in the      */
/* # define CURRENT_GRADIENTS on */    /* simulation.                */
/* # define TOROIDAL_RIPPLE 0.1 */     /*****

# define PARTICLES 2001
# define COLLISION_TIMES 30
# define CALCULATE_PER_COLTIME 100
# define PRINT_PER_COLTIME 10
# define CALC_TO_PRINT_RATIO (CALCULATE_PER_COLTIME/PRINT_PER_COLTIME)
# define MINOR_RADIUS 0.3
# define MAJOR_RADIUS 3.0
# define ELECTRON_CHARGE -1.0
# define ION_CHARGE 1.0
# define ELECTRON_MASS 2.72e-04
# define ION_MASS 1.0
# define ION_SPEED 0.00001
# define ELECTRON_SPEED (ION_SPEED*sqrt(ION_MASS/ELECTRON_MASS))
# define IOTA0 0.43
# define IOTA1 -0.64
# define C01 2.0
# define C02 4.0
# define C10 2.0
# define C20 4.0
# define NUSTAR 0.1
# define BNAUGHT 1.0
# define ANNULUSWIDTH 0.15

/*****
/*****
/** These variables are constant during the execution of the code**/
/*****
/*****
```

```

double PI;
double twoPI;
double twoPIsq;
double oneovertwoPI;
double length;
double epsilon;
double temperature;
double Psia;
double Psimiddle;
double Psiupper;
double Psilower;
double BMAX;

/*****
/*****
/**      These variables are intrinsic to the code      **/
/*****
/*****
int ende[2];
int t;
int p;
enum SPECIES {IONS, ELECTRONS} species; /* IONS=0; ELECTRONS=1 */
int evaluateflag;
int diffusionflag[2];
int ommitflag[2][PARTICLES];
long ommitcount[2];
int runningavgflag[2];
long N[2][PARTICLES];
int calcbutnoprnt[2];

/*****
/*****
/**      These are control variables (not essential)      **/
/*****
/*****
int totalnumber[2];
double increment[2][PARTICLES];
double totaltime[2][PARTICLES];
double avgincrement[2];
double maxincrement[2];
double minincrement[2];

/*****
/*****
/**      These are the simulated variables and physical parameters      **/
/*****
/*****
double THETA[2][PARTICLES];
double PHI[2][PARTICLES];
double PSI[2][PARTICLES];
double RHO[2][PARTICLES];
double B[2][PARTICLES];

```

```

double LAMBDA [2] [PARTICLES];
double ENERGY [2] [PARTICLES];
double KINETICENERGY [2] [PARTICLES];
double NU [2] [PARTICLES];
double TAU [2] [PARTICLES];
double MU [2] [PARTICLES];
double WEIGHT [2] [PARTICLES];
double THETAOLD [2] [PARTICLES];
double PHIOLD [2] [PARTICLES];
double PSIOLD [2] [PARTICLES];
double RHOOLD [2] [PARTICLES];
double BOLD [2] [PARTICLES];
double ENERGYOLD [2] [PARTICLES];
double bsqavg;
double speed [2];
double m [2];
double e [2];
double VAVERAGE [2];
double ENERGYAVERAGE [2];
double WEIGHTOLDDIFFUSION [2] [PARTICLES];

/*****
/*****
/**      These variables represent the results      **/
/*****
/*****

int trapped [2];          /** The number of trapped and **/
int passing [2];         /**   passing particles   **/
                          /*******

double bootstrap [2];    /** Names for variables rep- **/
double pscostheta [2];   /**   resenting the bootstrap **/
double pscos2theta [2]; /**   and P.S. currents while **/
double pscosphi [2];     /**   they are accumulating in **/
double pscos2phi [2];    /**   the getcurrents()     **/
double boottrapped [2];  /**   subroutine.           **/
double bootpassing [2];  /*******

double bootstrapavg [2]; /** Names for variables rep- **/
double pscosthetaavg [2]; /**   resenting the bootstrap **/
double pscos2thetaavg [2]; /**   and P.S. currents averaged **/
double pscosphiavg [2];   /**   over 1 C.T. divided by   **/
double pscos2phiavg [2];  /**   "CALC_TO_PRINT_RATIO".  **/
                          /*******

double bootstrapavgoverv [2]; /** Variables representing **/
double pscosthetaavgoverv [2]; /** the currents divided by v **/
double pscos2thetaavgoverv [2]; /** used to obtain the local **/
double pscosphiavgoverv [2]; /**   flow velocity in the   **/
double pscos2phiavgoverv [2]; /** adjustweights() subroutine **/

```

```

double diffusioncoefficient[2];
double constant[2];
double mcos1theta[2];
double mcos2theta[2];
double mcos3theta[2];
double msin1theta[2];
double msin2theta[2];
double msin3theta[2];
double ncos1phi[2];
double ncos2phi[2];
double ncos3phi[2];
double nsin1phi[2];
double nsin2phi[2];
double nsin3phi[2];

double constavg[2];
double mcos1avg[2];
double mcos2avg[2];
double mcos3avg[2];
double msin1avg[2];
double msin2avg[2];
double msin3avg[2];
double ncos1avg[2];
double ncos2avg[2];
double ncos3avg[2];
double nsin1avg[2];
double nsin2avg[2];
double nsin3avg[2];

/**** These are the subroutines in the code. ****/
void initransd();
void gaussian();
void init();
double maxfield();
void hamilton();
int errorcontrol();
void getincrements();
void collision();

```

```

void adjustweights();
void psicheck();
void makeold();
void getcurrents();
void averagecurrents();
void resetcurrents();
void printstuff();
void printparameters(float time);
void calculateandprintdiffusion(float time);
void calculateandprintdensity(float time);
float rando();
double drand48();

/*****
/*****
/** These are the inline functions (used with 'gcc') **/
/*****
/*****
inline double Bfield(double theta, double phi, double psi)
{
#ifdef TOROIDAL_RIPPLE
    return(BNAUGHT/sqrt(1.0 + C01*epsilon*sqrt(psi/Psia)*cos(theta) + C02
*epsilon*epsilon*(psi/Psia)*cos(2.0*theta) + C10*TOROIDAL_RIPPLE*sqrt(p
si/Psia)*cos(phi) + C20*TOROIDAL_RIPPLE*TOROIDAL_RIPPLE*(psi/Psia)*cos(
2.0*phi));
#endif
#ifndef TOROIDAL_RIPPLE
    return(BNAUGHT/sqrt(1.0 + C01*epsilon*sqrt(psi/Psia)*cos(theta) + C02
*epsilon*epsilon*(psi/Psia)*cos(2.0*theta)));
#endif
}

inline double eQNPOT(double theta, double phi)
{
    return(-2.0/3.0*m[species]*VAVERAGE[species]*VAVERAGE[species]*(mcos1
avg[0]*cos(theta) + mcos2avg[0]*cos(2.0*theta) + mcos3avg[0]*cos(3.0*th
eta) + msin1avg[0]*sin(theta) + msin2avg[0]*sin(2.0*theta) + msin3avg[0
]*sin(3.0*theta) + ncos1avg[0]*cos(phi) + ncos2avg[0]*cos(2.0*phi) + nc
os3avg[0]*cos(3.0*phi) + nsin1avg[0]*sin(phi) + nsin2avg[0]*sin(2.0*phi
) + nsin3avg[0]*sin(3.0*phi));
}

inline double eQNPOTbyT(double theta, double phi)
{
    return(-2.0*(mcos1avg[0]*cos(theta) + mcos2avg[0]*cos(2.0*theta) + mc
os3avg[0]*cos(3.0*theta) + msin1avg[0]*sin(theta) + msin2avg[0]*sin(2.0
*theta) + msin3avg[0]*sin(3.0*theta) + ncos1avg[0]*cos(phi) + ncos2avg[
0]*cos(2.0*phi) + ncos3avg[0]*cos(3.0*phi) + nsin1avg[0]*sin(phi) + nsi
n2avg[0]*sin(2.0*phi) + nsin3avg[0]*sin(3.0*phi));
}

inline double edQNPOTdtheta(double theta, double sin1t, double sin2t, d

```

```

double cos1t, double cos2t)
{
    return(-2.0/3.0*[species]*VAVERAGE[species]*VAVERAGE[species]*(-mcos
1avg[0]*sin1t - 2.0*mcos2avg[0]*sin2t - 3.0*mcos3avg[0]*sin(3.0*theta)
+ msin1avg[0]*cos1t + 2.0*msin2avg[0]*cos2t + 3.0*msin3avg[0]*cos(3.0*theta));
}

inline double edQNPOTdphi(double phi, double sin1p, double sin2p, double
cos1p, double cos2p)
{
    return(-2.0/3.0*[species]*VAVERAGE[species]*VAVERAGE[species]*(-ncos
1avg[0]*sin1p - 2.0*ncos2avg[0]*sin2p - 3.0*ncos3avg[0]*sin(3.0*phi) +
nsin1avg[0]*cos1p + 2.0*nsin2avg[0]*cos2p + 3.0*nsin3avg[0]*cos(3.0*phi
)));
}

inline double iota(double psi)                /* toroidal transform */
{
    return(IOTA0 + IOTA1*(psi-Psimiddle)/Psia);
}

inline double g(double psi)                   /* poloidal current */
{
    return(length*BNAUGHT);
}

inline double dgdpsi(double psi)
{
    return(0.0);
}

inline double i(double psi)                   /* toroidal current */
{
    return(iota(psi)*epsilon*epsilon*g(psi));
}

inline double didpsi(double psi)
{
    return(0.0);
}

inline double dBdthetaoverB(double sin1t, double sin2t, double psi, double
bno)
{
    return(0.5*bno*bno*(C01*epsilon*sqrt(psi/Psia)*sin1t + 2.0*C02*epsilon
n*epsilon*psi/Psia*sin2t));
}

#ifdef TOROIDAL_RIPPLE
inline double dBdphioverB(double sin1p, double sin2p, double psi, double
bno)

```

```

{
return(0.5*bno*bno*(C10*TOROIDAL_RIPPLE*sqrt(psi/Psia)*sin1p + 2.0*C20*
TOROIDAL_RIPPLE*TOROIDAL_RIPPLE*psi/Psia*sin2p));
}
#endif

inline double dBdpsioverB(double cos1t, double cos2t, double cos1p, dou
ble cos2p, double psi, double bno)
{
#ifdef TOROIDAL_RIPPLE
return(-0.5*bno*bno*(0.5*C01*epsilon/sqrt(psi*Psia)*cos1t + C02*epsil
on*epsilon/Psia*cos2t + 0.5*C10*TOROIDAL_RIPPLE/sqrt(psi*Psia)*cos1p +
C20*TOROIDAL_RIPPLE*TOROIDAL_RIPPLE/Psia*cos2p));
#endif

#ifdef TOROIDAL_RIPPLE
return(-0.5*bno*bno*(0.5*C01*epsilon/sqrt(psi*Psia)*cos1t + C02*epsil
on*epsilon/Psia*cos2t));
#endif
}

#ifdef LOOP_VOLTAGE
inline double loopvoltage()
{
return(LOOP_VOLTAGE);
}
#endif

inline double thetadot(double oneoverJ, double Prho, double rhodgdpsi,
double iota, double g, double Prhosq, double dBdpsioverB)
{
return(-oneoverJ*(Prho*(rhodgdpsi - oneovertwoPI*e[species]*iota) - g
*Prhosq*dBdpsioverB));
}

inline double phidot(double oneoverJ, double Prhosq, double dBdpsioverB
, double i, double rhodidpsi, double Prho)
{
return(-oneoverJ*(Prhosq*dBdpsioverB*i - (rhodidpsi + oneovertwoPI*e[
species])*Prho));
}

inline double psidot(double oneoverJ, double g, double Prhosq, double d
BdthetaoverB, double edQNPOTdtheta, double edQNPOTdphi, double i, doubl
e dBdphioverB, double loopvoltage)
{
return(oneoverJ*(g*(-Prhosq*dBdthetaoverB - edQNPOTdtheta) + i*(Prhos
q*dBdphioverB + edQNPOTdphi) + oneovertwoPI*e[species]*loopvoltage*i));
}

inline double rhodot(double oneoverJ, double rhodgdpsi, double iota, do
uble Prhosq, double dBdthetaoverB, double edQNPOTdtheta, double edQNPOT

```



```

dphi, double rhodidpsi, double dBdphioverB, double loopvoltage)
{
    return(onoeverJ*((rhodgdpsi - onovertwoPI*e[species]*iota)*(Prhosq*d
    BdthetaoverB + edQNPOTdtheta) - (rhodidpsi + onovertwoPI*e[species])*(
    Prhosq*dBdphioverB + edQNPOTdphi) - onovertwoPI*e[species]*loopvoltage
    *(rhodidpsi + onovertwoPI*e[species])));
}

/*****
/*****
/*****      MAIN()      *****/
/*****
/*****
main()
{
    int stepback;                * variable for errorcontrol() */

    initrand();                  /*****/
    for(species = IONS; species<=ELECTRONS; species++){ /**initialize **/
        for(p=1; p<PARTICLES; p++){ /** the random seed and **/
            init();                /** all parameters for the **/
            printparameters(0.0);  /** particles **/
        }                          /*****/
    }                              /*****/

    /*****/
    /*****/
    /*****      MONTE CARLO SCHEME      *****/
    /*****/
    /*****/

    species = 0;                 /*****/ Start with the Ions *****/
    while(!ende[species]){
        t++;                      /*****/ update timestep *****/
        for(p=1; p<PARTICLES; p++){
            if(!ommitflag[species][p]){

                /***** STEP 1. Updating Positions and Momenta *****/
                /*****/
                stepback = 1;      /** Execute hamilton() again **/
                for(;stepback;){  /** if deltaE[p] is above the **/
                    hamilton();  /** upper limit (6E-11) and **/
                    stepback = errorcontrol(); /**adjust increment[species][p]**/
                }                /** depending on deltaE[p]. **/
                /*****/
                getincrements();  /**Calculate min,max,avgincrem.**/
                psicheck();      /**Check if psi is inside ann. **/
                /*****/

                /***** STEP 2. Updating The Pitches *****/

                collision();

```

```

/***** STEP 3. Updating The Weights *****/
adjustweights();

makeold();          /** Adjust variables for next timestep. **/
}
}

/*****
/*****
/**CALCULATION OF CURRENTS, DENSITIES AND AVERAGING PROCEDURES**/
/*****
/*****

if(ommitcount[species] == (PARTICLES-1)){
    /** All particles have reached **/
    /** the time boundary. **/
    getcurrents();          /** Calculate bootstrap and P.S.**/
    averagecurrents();     /** currents and densities and **/
    /** average them over 0.1 C.T. **/
    /*******

    calcbutnoprnt[species]++; /**update variable for calc/print**/

    if(calcbutnoprnt[species] == CALC_TO_PRINT_RATIO){
        /*******
        /**calc/print variable has reached**/
        /** assigned calc/print ratio. **/
        /*******
        runningavgflag[species]++; /** Start running average now. **/
        /*******
        printstuff();          /** Print currents and - if 1 C.T.**/
        /** has passed - also print dens. **/
        /** coeff. and diff. parameter. **/
        /*******
        calcbutnoprnt[species] = 0; /** Reset calc/print variable.**/
        /*******
    }

    resetcurrents();          /** set currents to zero for next step **/
    species = 1 - species;/** switch specie Ions <--> Electrons **/
}
}
}

void initrand()
{
    srand48(2121.21);          /** random seed initialization */
}

```

```

        /* calculations of the constants */
PI          = 4.0*atan(1.0);
twoPI       = 2.0*PI;
twoPIsq     = 4.0*PI*PI;
oneovertwoPI = 1.0/twoPI;
length      = 2.0*PI*MAJOR_RADIUS; /*length = 2*Pi*R = char. lenght*/
epsilon     = MINOR_RADIUS/MAJOR_RADIUS;
Psia        = PI*MINOR_RADIUS*MINOR_RADIUS*BNAUGHT;
Psimiddle   = 0.85*Psia;
Psilower    = Psimiddle - 0.5*ANNULUSWIDTH*Psia;
Psiupper    = Psimiddle + 0.5*ANNULUSWIDTH*Psia;
BMAX        = maxfield();
}

void init()
/*****
/*****
/**          Initialize!          **/
/*****
/*****
{
double v;
double qnpotential = 0.0;
if(species == IONS){
    e[IONS]      = ION_CHARGE;
    m[IONS]      = ION_MASS;
    speed[IONS]  = ION_SPEED;
}
else{
    e[ELECTRONS] = ELECTRON_CHARGE;
    m[ELECTRONS] = ELECTRON_MASS;
    speed[ELECTRONS] = ELECTRON_SPEED;
}

v = speed[species];
N[species][p]      = 1;
increment[species][p] = 20000.0;
WEIGHT[species][p] = 0.0;
THETA[species][p]  = twoPI*rando();
PHI[species][p]    = twoPI*rando();
PSI[species][p]    = Psimiddle;
B[species][p]      = Bfield(THETA[species][p], PHI[species][p], PSI
[species][p]);
LAMBDA[species][p] = -0.99+1.98*rando();
RHO[species][p]    = (oneovertwoPI)*m[species]*LAMBDA[species][p]*v
/B[species][p];
MU[species][p]     = 0.5*m[species]*v*v*(1.0 - LAMBDA[species][p]*L
AMBDA[species][p])/B[species][p];
NU[species][p]     = NUSTAR*(IOTA0 + IOTA1*(PSI[species][p]-Psimidd
le)/Psia)*v*epsilon*sqrt(epsilon)/MAJOR_RADIUS;

```

```

TAU[species][p]      = 1.0/NU[species][p];
ommitflag[species][p] = 0;

#ifdef QUASI_NEUTRALITY
  qnpotential = eQNPOT(THETA[species][p], PHI[species][p]);
#endif
ENERGY[species][p]   = 0.5*(twoPIsq)*RHO[species][p]*RHO[species][p]*
B[species][p]*B[species][p]/m[species] + MU[species][p]*B[species][p] +
  qnpotential;

  printf("Initial Conditions: %i\nEnergy = %e VTOT = %e E2 = %e MU[sp
ecies][p] = %e\nNU=%e LAMBDA= %e\nthetas = %f phi = %f\n\n",p, ENERGY[
species][p], v, 0.5*m[species]*v*v, MU[species][p], NU[species][p], LAM
BDA[species][p], THETA[species][p], PHI[species][p]);
  VAVERAGE[species]   = speed[species];
  evaluateflag = 1;
  makeold();
  WEIGHTOLDDIFFUSION[species][p] = WEIGHT[species][p];
}

void hamilton()
{
/*****
/** These are the local variables used in the Runge - Kutta routine**/
*****/
  double theta1,      theta2,      theta3,      theta4;
  double phi1,        phi2,        phi3,        phi4;
  double psi1,        psi2,        psi3,        psi4;
  double rho1,        rho2,        rho3,        rho4;
  double b1,          b2,          b3,          b4;
  double bno1,        bno2,        bno3,        bno4;
  double deltaH1,     deltaH2,     deltaH3,     deltaH4;
  double sin1t1,      sin1t2,      sin1t3,      sin1t4;
  double sin2t1,      sin2t2,      sin2t3,      sin2t4;
  double cos1t1,      cos1t2,      cos1t3,      cos1t4;
  double cos2t1,      cos2t2,      cos2t3,      cos2t4;
  double sin1p1,      sin1p2,      sin1p3,      sin1p4;
  double sin2p1,      sin2p2,      sin2p3,      sin2p4;
  double cos1p1,      cos1p2,      cos1p3,      cos1p4;
  double cos2p1,      cos2p2,      cos2p3,      cos2p4;
  double tempiota1,    tempiota2,    tempiota3,    tempiota4;
  double g1,          g2,          g3,          g4;
  double i1,          i2,          i3,          i4;
  double oneoverJ1,   oneoverJ2,   oneoverJ3,   oneoverJ4;
  double Prhosq1,     Prhosq2,     Prhosq3,     Prhosq4;
  double Prho1,       Prho2,       Prho3,       Prho4;
  double dBdthetaoverB1, dBdthetaoverB2, dBdthetaoverB3, dBdthetaoverB4;
  double dBdpsioverB1, dBdpsioverB2, dBdpsioverB3, dBdpsioverB4;
  double phidot1,     phidot2,     phidot3,     phidot4;
  double k1theta,     k2theta,     k3theta,     k4theta;
  double k1phi,       k2phi,       k3phi,       k4phi;

```

```

double k1psi,          k2psi,          k3psi,          k4psi;
double kirho,         k2rho,          k3rho,          k4rho;
double kideltaH,     k2deltaH,       k3deltaH,       k4deltaH;
double loopvoltage1 = 0.0,          loopvoltage2 = 0.0,
      loopvoltage3 = 0.0,          loopvoltage4 = 0.0;
double rhodgdpsi1 = 0.0,          rhodgdpsi2 = 0.0,
      rhodgdpsi3 = 0.0,          rhodgdpsi4 = 0.0;
double rhodidpsi1 = 0.0,          rhodidpsi2 = 0.0,
      rhodidpsi3 = 0.0,          rhodidpsi4 = 0.0;
double edQNPOTdphi1 = 0.0,        edQNPOTdphi2 = 0.0,
      edQNPOTdphi3 = 0.0,        edQNPOTdphi4 = 0.0;
double edQNPOTdtheta1= 0.0,       edQNPOTdtheta2 = 0.0,
      edQNPOTdtheta3 = 0.0,       edQNPOTdtheta4= 0.0;
double dBdphioverB1 = 0.0,        dBdphioverB2 = 0.0,
      dBdphioverB3 = 0.0,        dBdphioverB4 = 0.0;
double qnpotential = 0.0;

/*****
/** Check whether the particle p will reach (or surpass) a time-      **/
/** boundary. If so, decrease increment[species][p] such that the    **/
/** boundary is reached precisely after the execution of this        **/
/** timestep and ommit particle [p] from now on until all particles  **/
/** have reached that time-boundary. N[][p] is the number of        **/
/** collision times over which particle p has been simulated.        **/
*****/
if(ommitflag[species][p]){
    ommitcount[species]--;
}
ommitflag[species][p] = 0;
if((increment[species][p]+totaltime[species][p]) >= (((float)(N[speci
es][p]))*TAU[species][p]/((float)(CALCULATE_PER_COLTIME)))){
    increment[species][p] = (((float)(N[species][p]))*TAU[species][p]/(
(float)(CALCULATE_PER_COLTIME))) - totaltime[species][p];
    ommitflag[species][p] = 1;
    N[species][p]++;
    ommitcount[species]++;
}

/*****
/** initialize local variables **/
*****/
theta1 = THETAOLD[species][p];
phi1   = PHIOLD[species][p];
psi1   = PSIOLD[species][p];
rho1   = RHOOLD[species][p];
b1     = BOLD[species][p];

/*****
/** Need this when coefficients in QNPOT change to prevent deltaE **/
/** from becoming too large **/
*****/
if(evaluateflag){

```

```

#ifdef QUASI_NEUTRALITY
    qnpotential = eQNPOT(theta1, phi1);
#endif
    ENERGYOLD[species][p] = 0.5*(twoPIsq)*rho1*rho1*b1*b1/m[species] +
    MU[species][p]*b1 + qnpotential;

    if(p == (PARTICLES-1)) {evaluateflag = 0;}
}

/***** obtain k1-values *****/
/***** obtain k1-values *****/
/***** obtain k1-values *****/
sin1t1 = sin(theta1);
sin2t1 = sin(2.0*theta1);
cos1t1 = cos(theta1);
cos2t1 = cos(2.0*theta1);
sin1p1 = sin(phi1);
sin2p1 = sin(2.0*phi1);
cos1p1 = cos(phi1);
cos2p1 = cos(2.0*phi1);
bno1    = b1/BNAUGHT;
Prho1   = (twoPIsq)*rho1*b1*b1/m[species];
Prhosq1 = (twoPIsq)*rho1*rho1*b1*b1/m[species] + MU[species][p]*b1;
tempiota1 = iota(psi1);
g1 = g(psi1);
i1 = i(psi1);

#ifdef QUASI_NEUTRALITY
    edQNPOTdtheta1 = edQNPOTdtheta(theta1,sin1t1,sin2t1, cos1t1, cos2t1);
    edQNPOTdphi1   = edQNPOTdphi(phi1, sin1p1, sin2p1, cos1p1, cos2p1);
#endif

#ifdef LOOP_VOLTAGE
    loopvoltage1 = loopvoltage();
#endif

#ifdef CURRENT_GRADIENTS
    rhodgdpsi1 = rho1*dgdpsi(psi1);
    rhodidpsi1 = rho1*didpsi(psi1);
#endif

    oneoverJ1 = 1.0/(g(psi1)*(rho1*rhodidpsi1+oneovertwoPI*e[species]) -
    i(psi1)*(rho1*rhodgdpsi1 - oneovertwoPI*tempiota1*e[species]));
    dBdthetaoverB1 = dBdthetaoverB(sin1t1, sin2t1, psi1, bno1);

#ifdef TOROIDAL_RIPPLE
    dBdphioverB1 = dBdphioverB(sin1p1, sin2p1, psi1, bno1);
#endif

    dBdpsioverB1 = dBdpsioverB(cos1t1,cos2t1,cos1p1,cos2p1,psi1,bno1);
    phidot1 = phidot(oneoverJ1,Prhosq1,dBdpsioverB1,i1,rhodidpsi1,Prho1);

```

```

k1theta = increment[species][p] * thetadot(oneoverJ1, Prho1, rhodgdps
i1, tempiota1, g1, Prhosq1, dBdpsioverB1);
k1phi   = increment[species][p] * phidot1;
k1psi   = increment[species][p] * psidot(oneoverJ1, g1, Prhosq1, dBdt
hetaoverB1, edQNPOTdtheta1, edQNPOTdphi1, i1, dBdphioverB1, loopvoltage1);
k1rho   = increment[species][p] * rhodot(oneoverJ1, rhodgdps1, tempi
ota1, Prhosq1, dBdthetaoverB1, edQNPOTdtheta1, edQNPOTdphi1, rhodidpsi1
, dBdphioverB1, loopvoltage1);
k1deltaH = increment[species][p] * oneovertwoPI*e[species]*loopvoltage1*phidot1;

/*****
/***** obtain k2-values *****/
/*****
theta2 = theta1 + 0.5*k1theta;
phi2   = phi1   + 0.5*k1phi;
psi2   = psi1   + 0.5*k1psi;
rho2   = rho1   + 0.5*k1rho;
sin1t2 = sin(theta2);
sin2t2 = sin(2.0*theta2);
cos1t2 = cos(theta2);
cos2t2 = cos(2.0*theta2);
sin1p2 = sin(phi2);
sin2p2 = sin(2.0*phi2);
cos1p2 = cos(phi2);
cos2p2 = cos(2.0*phi2);
b2      = Bfield(theta2, phi2, psi2);
bno2    = b2/BNAUGHT;
Prho2   = (twoPIsq)*rho2*b2*b2/m[species];
Prhosq2 = (twoPIsq)*rho2*rho2*b2*b2/m[species] + MU[species][p]*b2;
tempiota2 = iota(psi2);
g2 = g(psi2);
i2 = i(psi2);

#ifdef QUASI_NEUTRALITY
edQNPOTdtheta2 = edQNPOTdtheta(theta2, sin1t2, sin2t2, cos1t2, cos2t2);
edQNPOTdphi2   = edQNPOTdphi(phi2, sin1p2, sin2p2, cos1p2, cos2p2);
#endif

#ifdef LOOP_VOLTAGE
loopvoltage2 = loopvoltage();
#endif

#ifdef CURRENT_GRADIENTS
rhodgdpsi2 = rho2*dgdpsi(psi2);
rhodidpsi2 = rho2*didpsi(psi2);
#endif

oneoverJ2 = 1.0/(g(psi2)*(rho2*rhodidpsi2+oneovertwoPI*e[species]) -
i(psi2)*(rho2*rhodgdpsi2 - oneovertwoPI*tempiota2*e[species]));
dBdthetaoverB2 = dBdthetaoverB(sin1t2, sin2t2, psi2, bno2);

```

```

#ifdef TOROIDAL_RIPPLE
    dBdphioverB2 = dBdphioverB(sin1p2, sin2p2, psi2, bno2);
#endif

    dBdpsioverB2 = dBdpsioverB(cos1t2, cos2t2, cos1p2, cos2p2, psi2, bno2);
    phidot2 = phidot(oneoverJ2, Prhosq2, dBdpsioverB2, i2, rhodidpsi2, Prho2);
    k2theta = increment[species][p] * thetadot(oneoverJ2, Prho2, rhodgdps
i2, tempiota2, g2, Prhosq2, dBdpsioverB2);
    k2phi = increment[species][p] * phidot2;
    k2psi = increment[species][p] * psidot(oneoverJ2, g2, Prhosq2, dBdt
hetaoverB2, edQNPOTdtheta2, edQNPOTdphi2, i2, dBdphioverB2, loopvoltage2);
    k2rho = increment[species][p] * rhodot(oneoverJ2, rhodgdpsi2, tempi
ota2, Prhosq2, dBdthetaoverB2, edQNPOTdtheta2, edQNPOTdphi2, rhodidpsi2
, dBdphioverB2, loopvoltage2);
    k2deltaH = increment[species][p] * oneovertwoPI*e[species]*loopvoltage2*phidot2;

/*****
/***** obtain k3-values *****/
/*****
theta3 = theta1 + 0.5*k2theta;
phi3 = phi1 + 0.5*k2phi;
psi3 = psi1 + 0.5*k2psi;
rho3 = rho1 + .5*k2rho;
sin1t3 = sin(theta3);
sin2t3 = sin(2.0*theta3);
cos1t3 = cos(theta3);
cos2t3 = cos(2.0*theta3);
sin1p3 = sin(phi3);
sin2p3 = sin(2.0*phi3);
cos1p3 = cos(phi3);
cos2p3 = cos(2.0*phi3);
b3 = Bfield(theta3, phi3, psi3);
bno3 = b3/BNAUGHT;
Prho3 = (twoPIsq)*rho3*b3*b3/m[species];
Prhosq3 = (twoPIsq)*rho3*rho3*b3*b3/m[species] + MU[species][p]*b3;
tempiota3 = iota(psi3);
g3 = g(psi3);
i3 = i(psi3);

#ifdef QUASI_NEUTRALITY
    edQNPOTdtheta3 = edQNPOTdtheta(theta3, sin1t3, sin2t3, cos1t3, cos2t3);
    edQNPOTdphi3 = edQNPOTdphi(phi3, sin1p3, sin2p3, cos1p3, cos2p3);
#endif

#ifdef LOOP_VOLTAGE
    loopvoltage3 = loopvoltage();
#endif

#ifdef CURRENT_GRADIENTS
    rhodgdpsi3 = rho3*dgdpsi(psi3);
    rhodidpsi3 = rho3*didpsi(psi3);

```



```

#endif

    oneoverJ3 = 1.0/(g(psi3)*(rho3*rhodidpsi3+oneovertwoPI*e[species]) -
i(psi3)*(rho3*rhodgpsi3 - oneovertwoPI*tempiota3*e[species]));
    dBdthetaoverB3 = dBdthetaoverB(sin1t3, sin2t3, psi3, bno3);

#ifdef TOROIDAL_RIPPLE
    dBdphioverB3 = dBdphioverB(sin1p3, sin2p3, psi3, bno3);
#endif

    dBdpsioverB3 = dBdpsioverB(cos1t3,cos2t3,cos1p3,cos2p3,psi3,bno3);
    phidot3 = phidot(oneoverJ3,Prhosq3,dBdpsioverB3,i3,rhodidpsi3,Prho3);
    k3theta = increment[species][p] * thetadot(oneoverJ3, Prho3, rhodgpsi3,
i3, tempiota3, g3, Prhosq3, dBdpsioverB3);
    k3phi = increment[species][p] * phidot3;
    k3psi = increment[species][p] * psidot(oneoverJ3, g3, Prhosq3, dBdthetaoverB3,
edQNPOTdtheta3,edQNPOTdphi3,i3,dBdphioverB3, loopvoltage3);
    k3rho = increment[species][p] * rhodot(oneoverJ3, rhodgpsi3, tempiota3,
Prhosq3, dBdthetaoverB3, edQNPOTdtheta3, edQNPOTdphi3, rhodidpsi3,
dBdphioverB3, loopvoltage3);
    k3deltaH = increment[species][p] * oneovertwoPI*e[species]*loopvoltage3*phidot3;

/***** obtain k4-values *****/
theta4 = theta1 + k3theta;
phi4 = phi1 + k3phi;
psi4 = psi1 + k3psi;
rho4 = rho1 + k3rho;
sin1t4 = sin(theta4);
sin2t4 = sin(2.0*theta4);
cos1t4 = cos(theta4);
cos2t4 = cos(2.0*theta4);
sin1p4 = sin(phi4);
sin2p4 = sin(2.0*phi4);
cos1p4 = cos(phi4);
cos2p4 = cos(2.0*phi4);
b4 = Bfield(theta4, phi4, psi4);
bno4 = b4/BNAUGHT;
Prho4 = (twoPIsq)*rho4*b4*b4/m[species];
Prhosq4 = (twoPIsq)*rho4*rho4*b4*b4/m[species] + MU[species][p]*b4;
tempiota4 = iota(psi4);
g4 = g(psi4);
i4 = i(psi4);

#ifdef QUASI_NEUTRALITY
    edQNPOTdtheta4 = edQNPOTdtheta(theta4,sin1t4,sin2t4, cos1t4, cos2t4);
    edQNPOTdphi4 = edQNPOTdphi(phi4, sin1p4, sin2p4, cos1p4, cos2p4);
#endif

#ifdef LOOP_VOLTAGE

```

```

loopvoltage4 = loopvoltage();
#endif

#ifdef CURRENT_GRADIENTS
  rhodgdpsi4 = rho4*dgdpsi(psi4);
  rhodidpsi4 = rho4*didpsi(psi4);
#endif

  oneoverJ4 = 1.0/(g(psi4)*(rho4*rhodidpsi4+oneovertwoPI*e[species]) -
i(psi4)*(rho4*rhodgdpsi4 - oneovertwoPI*tempiota4*e[species]));
  dBdthetaoverB4 = dBdthetaoverB(sin1t4, sin2t4, psi4, bno4);

#ifdef TOROIDAL_RIPPLE
  dBdphioverB4 = dBdphioverB(sin1p4, sin2p4, psi4, bno4);
#endif

  dBdpsioverB4 = dBdpsioverB(cos1t4,cos2t4,cos1p4,cos2p4,psi4,bno4);
  phidot4 = phidot(oneoverJ4,Prhosq4,dBdpsioverB4,i4,rhodidpsi4,Prho4);
  k4theta = increment[species][p] * thetadot(oneoverJ4, Prho4, rhodgdps
i4, tempiota4, g4, Prhosq4, dBdpsioverB4);
  k4phi = increment[species][p] * phidot4;
  k4psi = increment[species][p] * psidot(oneoverJ4, g4, Prhosq4, dBdt
hetaoverB4, edQNPOTdtheta4,edQNPOTdphi4,i4,dBdphioverB4, loopvoltage4);
  k4rho = increment[species][p] * rhodot(oneoverJ4, rhodgdpsi4, tempi
ota4, Prhosq4, dBdthetaoverB4, edQNPOTdtheta4, edQNPOTdphi4, rhodidpsi4
, dBdphioverB4, loopvoltage4);
  k4deltaH = increment[species][p] * oneovertwoPI*e[species]*loopvoltage4*phidot4;

/*****
/** Obtain new values for global THETA, PHI, RHO, KINETICENERGY, **/
/** ENERGY and B. Calculate edQNPOTdthetaglobal and edPOTdthetaglobal**/
/** to be used in function adjustweights(). **/
/** Also adjust ENERGYOLD to be used in function errorcontrol(). **/
/** Calculate parameter totaltime[species][p] for boundary check in **/
/** the next timestep in function hamilton(). **/
*****/
  THETA[species][p] = THETAOLD[species][p] + 0.16666667*(k1the
ta + 2.0*k2theta + 2.0*k3theta + k4theta);
  PHI[species][p] = PHIOLD[species][p] + 0.16666667*(k1phi
+ 2.0*k2phi + 2.0*k3phi + k4phi);
  PSI[species][p] = PSIOLD[species][p] + 0.16666667*(k1psi
+ 2.0*k2psi + 2.0*k3psi + k4psi);
  RHO[species][p] = RHOOLD[species][p] + 0.16666667*(k1rho
+ 2.0*k2rho + 2.0*k3rho + k4rho);
  ENERGYOLD[species][p] -= 0.16666667*(k1deltaH + 2.0*k2deltaH
+ 2.0*k3deltaH + k4deltaH);
  B[species][p] = Bfield(THETA[species][p], PHI[species][p]
, PSI[species][p]);
  KINETICENERGY[species][p] = 0.5*(twoPIsq)*RHO[species][p]*RHO[species
][p]*B[species][p]*B[species][p]/m[species] + MU[species][p]*B[species
][p];

```

```

#ifdef QUASI_NEUTRALITY
    qnpotential = eQNPOT(THETA[species][p], PHI[species][p]);
#endif
    ENERGY[species][p] = KINETICENERGY[species][p] + qnpotential;

    totaltime[species][p] += increment[species][p];
}

int errorcontrol()
/*****
/*****
/** Check the error in the previous timestep by comparing the new **/
/** Energy of the particles to the old Energy adjusted for the **/
/** expected change in energy due to the loopvoltage and new **/
/** coefficients in QNpotential **/
/*****
/*****
{
    int stepback = 0;
    double deltaE;
    deltaE = (ENERGY[species][p] - ENERGYOLD[species][p])/ENERGYOLD[sp
ecies][p];
    if(fabs(deltaE) < 1.0E-11){
        increment[species][p] += ((2.0E-11)-fabs(deltaE))/(2.0E-10)*increme
nt[species][p];
        if(increment[species][p]*NU[species][p] > 0.01){
            increment[species][p] = 0.01*TAU[species][p];
        }
    }
    else if((3.0E-11 < fabs(deltaE)) && (fabs(deltaE) <= 6.0E-11)){
        increment[species][p] -= (fabs(deltaE)-(2.0E-11))/(2.0E-10)*increme
nt[species][p];
    }
    else if(fabs(deltaE) > 6.0E-11){
        totaltime[species][p] -= increment[species][p]; /* assign old va
lue to totalnumber[species][p] */
        increment[species][p] -= 0.4*increment[species][p];
        /* printf("stepback at time %i for particle %i\n\n", t, p); */
        stepback = 1;
    }
    return(stepback);
}

void getincrements()
/*****
/*****
/** Get the minimum, maximum and average increment for this time **/
/** series to check if everything is running O.K. (Increments that **/
/** are unusually small mean something is wrong. **/

```

```

/*****
/*****
{
  if(p == 1){
    minincrement[species] = increment[species][p];
    maxincrement[species] = increment[species][p];
    avgincrement[species] = 0.0;
  }
  else{
    if(increment[species][p] > maxincrement[species]){maxincrement[spec
ies] = increment[species][p];}
    if(increment[species][p] < minincrement[species]){minincrement[spec
ies] = increment[species][p];}
  }
  avgincrement[species] += increment[species][p] / ((double)(PARTICLES-1));
}

void psicheck()
/*****
/*****
/** Check if particle p is still inside the PSI-annulus. If not, **/
/** re-initialize particle p by putting it in the middle of the **/
/** annulus with arbitrary initial THETA, PHI and LAMBDA. Then, **/
/** execute one timestep forward for that particle. **/
/** Also adjust LAMBDA for all particles because MU is constant until **/
/** collision, but B(THETA, PHI, PSI) changed. **/
/*****
/*****
{
  int stepback;
  double v;
  int tempN;
  if((PSI[species][p] < Psilower) || (PSI[species][p] > Psiupper)){
    totalnumber[species] += 1;
    tempN = N[species][p];
    init();
    N[species][p] = tempN;          /*****
    stepback = 1;                  /* execute hamilton() again */
    for(;stepback;){              /* if deltaE[p] is above */
      hamilton();                  /* the upper limit (6E-12) */
      stepback = errorcontrol(); /* also adjust increment */
    }                               /* depending on deltaE[p] */
                                   /*****

    getincrements();
  }
  LAMBDA[species][p] = (twoPI)*B[species][p]*RHO[species][p]/sqrt(2.0*m
[species]*KINETICENERGY[species][p]);
}

void collision()

```

```

/*****
/*****
/** Obtain new LAMBDA (leading to new MU and RHO) from the Monte      **/
/** Carlo equivalent of the Loretz operator. The NU for the         **/
/** electrons is multiplied by a factor of two because it represents **/
/** the sum of NU(electrons on ions) and NU(electrons on electrons), **/
/** which are equal to each other.                                  **/
/*****
/*****
{
  double lambdacol;
  float sign;

  if(species == IONS){
    sign = rando();
    if (sign < 0.5){
      lambdacol = LAMBDA[0][p]*(1.0 - NU[0][p]*increment[0][p]) + sqrt
((1.0 - LAMBDA[0][p]*LAMBDA[0][p]) * NU[0][p]*increment[0][p]);
    }
    else {
      lambdacol = LAMBDA[0][p]*(1.0 - NU[0][p]*increment[0][p]) - sqrt
((1.0 - LAMBDA[0][p]*LAMBDA[0][p]) * NU[0][p]*increment[0][p]);
    }
  }

  else {      /* species == ELECTRONS */
    sign = rando();
    if (sign < 0.5){
      lambdacol = LAMBDA[1][p]*(1.0 - 2.0*NU[1][p]*increment[1][p]) +
sqrt((1.0 - LAMBDA[1][p]*LAMBDA[1][p]) * 2.0*NU[1][p]*increment[1][p]);
    }
    else {
      lambdacol = LAMBDA[1][p]*(1.0 - 2.0*NU[1][p]*increment[1][p]) -
sqrt((1.0 - LAMBDA[1][p]*LAMBDA[1][p]) * 2.0*NU[1][p]*increment[1][p]);
    }
  }

  LAMBDA[species][p] = lambdacol;
  MU[species][p] = KINETICENERGY[species][p]/(B[species][p])*(1.0 -
LAMBDA[species][p]*LAMBDA[species][p]);
  RHO[species][p] = (oneovertwoPI)*LAMBDA[species][p]*sqrt(2.0*m[spe
cies]*KINETICENERGY[species][p])/B[species][p];
}

void adjustweights()
/*****
/*****
/** Adjust the weights for each particle according to the      **/
/** prescription given by the transformed Fokker-Planck equation. **/
/** Note that the new weight for the ions doesn't depend on the **/
/** background flow of the electrons.                          **/

```

```

/*****
/*****
{
    double momconsstuff = 0.0;
    double deltapsi = PSI[species][p] - PSIOLD[species][p];;
/*   double deltatheta = THETA[species][p] - THETAOLD[species][p];;
    double deltaphi = PHI[species][p] - PHIOLD[species][p];; */
    if(species == IONS){

#ifdef MOMENTUM_CONSERVATION
        momconsstuff = 3.0/e[0]*LAMBDA[0][p]*NU[0][p]*increment[0][p]*(boot
strapavgoverv[0] + 2.0*pscosthetaavgoverv[0]*cos(THETA[0][p]) + 2.0*psc
os2thetaavgoverv[0]*cos(2.0*THETA[0][p]) + 2.0*pscosphiavgoverv[0]*cos(
PHI[0][p]) + 2.0*pscos2phiavgoverv[0]*cos(2.0*PHI[0][p]))*B[0][p];
#endif
        WEIGHT[0][p] += deltapsi + momconsstuff;

/* + 1.0/KINETICENERGY[0][p]*(edQNPOTdtheta(THETA[0][p], sin(THETA[0][p
]), sin(2.0*THETA[0][p]), cos(THETA[0][p]), cos(2.0*THETA[0][p]))*delta
theta + edQNPOTdphi(PHI[0][p], sin(PHI[0][p]), sin(2.0*PHI[0][p]), cos(
PHI[0][p]), cos(2.0*PHI[0][p]))*deltaphi);
*/
    }

    else{ /* species == ELECTRONS */

#ifdef MOMENTUM_CONSERVATION
        momconsstuff = 3.0/e[1]*LAMBDA[1][p]*NU[1][p]*increment[1][p]*(boot
strapavgoverv[1] + 2.0*pscosthetaavgoverv[1]*cos(THETA[1][p]) + 2.0*psc
os2thetaavgoverv[1]*cos(2.0*THETA[1][p]) + 2.0*pscosphiavgoverv[1]*cos(
PHI[1][p]) + 2.0*pscos2phiavgoverv[1]*cos(2.0*PHI[1][p]))*B[1][p] + 3.0
/e[0]*(speed[1]/speed[0])*LAMBDA[1][p]*(NU[1][p])*increment[1][p]*(boot
strapavgoverv[0] + 2.0*pscosthetaavgoverv[0]*cos(THETA[1][p]) + 2.0*psc
os2thetaavgoverv[0]*cos(2.0*THETA[1][p]) + 2.0*pscosphiavgoverv[0]*cos(
PHI[1][p]) + 2.0*pscos2phiavgoverv[0]*cos(2.0*PHI[1][p]))*B[1][p];
#endif
        WEIGHT[1][p] += deltapsi + momconsstuff;

/* + 1.0/KINETICENERGY[1][p]*(edQNPOTdtheta(THETA[1][p], sin(THETA[1][p
]), sin(2.0*THETA[1][p]), cos(THETA[1][p]), cos(2.0*THETA[1][p]))*delta
theta + edQNPOTdphi(PHI[1][p], sin(PHI[1][p]), sin(2.0*PHI[1][p]), cos(
PHI[1][p]), cos(2.0*PHI[1][p]))*deltaphi);
*/
    }
}

void makeold()
{
    THETAOLD[species][p] = THETA[species][p];
    PHIOLD[species][p] = PHI[species][p];
    PSIOLD[species][p] = PSI[species][p];
}

```

```

RHOOLD[species][p]    = RHO[species][p];
ENERGYOLD[species][p] = ENERGY[species][p];
BOLD[species][p]     = B[species][p];
}

void getcurrents()
{
    int pcount;
    double const_multiplier;
    VAVERAGE[species] = 0.0;
    ENERGYAVERAGE[species] = 0.0;
    for(pcount=1; pcount<PARTICLES; pcount++){
        VAVERAGE[species] += sqrt(2.0*KINETICENERGY[species][pcount]/m[species])/((double)(PARTICLES-1));
        ENERGYAVERAGE[species] += ENERGY[species][pcount]/(((double)(PARTICLES-1)));
    }
    /* calculate bootstrap and P.S. currents */

    for(pcount=1; pcount<PARTICLES; pcount++){
        double bno = B[species][pcount]/BNAUGHT;
        const_multiplier = e[species]*WEIGHT[species][pcount]*LAMBDA[species][pcount]*VAVERAGE[species]*bno/(((double)(PARTICLES-1)));
        bootstrap[species] += const_multiplier;
        pscostheta[species] += const_multiplier*cos(THETA[species][pcount]);
    };
    pscos2theta[species] += const_multiplier*cos(2.0*THETA[species][pcount]);
    pscosphi[species] += const_multiplier*cos(PHI[species][pcount]);
    pscos2phi[species] += const_multiplier*cos(2.0*PHI[species][pcount]);

    bsqavg += bno*bno/((double)(PARTICLES-1));

    /* calculate the passing and trapped particle */
    /* contribution to the bootstrap current */
    if(MU[species][pcount] <= ENERGY[species][pcount]/BMAX){
        passing[species]++;
        bootpassing[species] += const_multiplier;
    }
    else {
        trapped[species]++;
        boottrapped[species] += const_multiplier;
    }

    ommitflag[species][pcount] = 0;
}
}

void calculateandprintdiffusion(float time)

```

```

{
    int count;
    int pcount;
    double deltaweightsq[2][PARTICLES];
    static FILE *cp;
    char filename[20];
    for(pcount=1; pcount< PARTICLES; pcount++){
        deltaweightsq[species][pcount] = ((WEIGHT[species][pcount] - WEIGHT
OLDDIFFUSION[species][pcount])*(WEIGHT[species][pcount] - WEIGHTOLDDIFF
USION[species][pcount]));
        diffusioncoefficient[species] += deltaweightsq[species][pcount]/(2.
0*((double)(PARTICLES-1)));
        WEIGHTOLDDIFFUSION[species][pcount] = WEIGHT[species][pcount];
    }
    sprintf(filename, "diffusioncoeff%i", species);
    cp = fopen(filename, "ab");
    fprintf(cp, "%f %e\n", time, (diffusioncoefficient[species]));
    diffusioncoefficient[species] = 0.0;
    fclose(cp);
}

void calculateandprintdensity(float time)
/*****
/*****
/** Calculate the density components. constant[species], mcos[] etc.**/
/** are temporary (until the values accumulate for all particles). **/
/** They are not used in localQNPOTbyT below (avg[species] are) **/
/*****
/*****
{
    double multiplier;
    int count;
    int pcount;
    enum files {CONST, COSTHETA, COS2THETA, COS3THETA, SINTHETA, SIN2THETA,
SIN3THETA, COSPHI, COS2PHI, COS3PHI, SINPHI, SIN2PHI, SIN3PHI, FCOMP};
    static FILE *dp[FCOMP];
    char filename[25];
    char name[FCOMP][25] = {"densityconstavg", "densitycosthetaavg", "den
sitycos2thetaavg", "densitycos3thetaavg", "densitysinthetaavg", "densit
ysin2thetaavg", "densitysin3thetaavg", "densitycosphiavg", "densitycos2
phiavg", "densitycos3phiavg", "densitysinphiavg", "densitysin2phiavg",
"densitysin3phiavg"};
    double localeQNPOTbyT = 0.0;
    for(pcount=1;pcount<PARTICLES;pcount++){
        double bno = B[species][pcount]/BNAUGHT;

#ifdef QUASI_NEUTRALITY
        localeQNPOTbyT = eQNPOTbyT(THETA[species][pcount], PHI[species][pco
unt]);
#endif

        c_multiplier = (WEIGHT[species][pcount] + localeQNPOTbyT)*bno*bno/(

```



```

((double)(PARTICLES-1));
    constant[species] += c_multiplier;
    mcos1theta[species] += c_multiplier*cos(THETA[species][pcount]);
    mcos2theta[species] += c_multiplier*cos(2.0*THETA[species][pcount]);
    mcos3theta[species] += c_multiplier*cos(3.0*THETA[species][pcount]);
    msin1theta[species] += c_multiplier*sin(THETA[species][pcount]);
    msin2theta[species] += c_multiplier*sin(2.0*THETA[species][pcount]);
    msin3theta[species] += c_multiplier*sin(3.0*THETA[species][pcount]);
    ncos1phi[species] += c_multiplier*cos(PHI[species][pcount]);
    ncos2phi[species] += c_multiplier*cos(2.0*PHI[species][pcount]);
    ncos3phi[species] += c_multiplier*cos(3.0*PHI[species][pcount]);
    nsin1phi[species] += c_multiplier*sin(PHI[species][pcount]);
    nsin2phi[species] += c_multiplier*sin(2.0*PHI[species][pcount]);
    nsin3phi[species] += c_multiplier*sin(3.0*PHI[species][pcount]);
}

/* set avg[species] equal to the accumulated value to be used *\
/* in the eQNPOT subroutines */
constavg[species] = constant[species];
mcos1avg[species] = mcos1theta[species];
mcos2avg[species] = mcos2theta[species];
mcos3avg[species] = mcos3theta[species];
msin1avg[species] = msin1theta[species];
msin2avg[species] = msin2theta[species];
msin3avg[species] = msin3theta[species];
ncos1avg[species] = ncos1phi[species];
ncos2avg[species] = ncos2phi[species];
ncos3avg[species] = ncos3phi[species];
nsin1avg[species] = nsin1phi[species];
nsin2avg[species] = nsin2phi[species];
nsin3avg[species] = nsin3phi[species];

/* Set constant[species], mcos1theta[species] etc. to zero for next */
/* time around, so they start accumulating from zero. */
constant[species] = 0.0;
mcos1theta[species] = 0.0;
mcos2theta[species] = 0.0;
mcos3theta[species] = 0.0;
msin1theta[species] = 0.0;
msin2theta[species] = 0.0;
msin3theta[species] = 0.0;
ncos1phi[species] = 0.0;
ncos2phi[species] = 0.0;
ncos3phi[species] = 0.0;
nsin1phi[species] = 0.0;
nsin2phi[species] = 0.0;
nsin3phi[species] = 0.0;

/* print out the density Fourier coefficients */

for(count=0; count<FCOMP; count++){
    sprintf(filename, "%s%i", name[count], species);

```

```

    dp[count] = fopen(filename, "ab");
}

fprintf(dp[CONST], "%f %e\n", time, constavg[species]);
fprintf(dp[COSTHETA], "%f %e\n", time, mcos1avg[species]);
fprintf(dp[COS2THETA], "%f %e\n", time, mcos2avg[species]);
fprintf(dp[COS3THETA], "%f %e\n", time, mcos3avg[species]);
fprintf(dp[SINTHETA], "%f %e\n", time, msin1avg[species]);
fprintf(dp[SIN2THETA], "%f %e\n", time, msin2avg[species]);
fprintf(dp[SIN3THETA], "%f %e\n", time, msin3avg[species]);
fprintf(dp[COSPHI], "%f %e\n", time, ncos1avg[species]);
fprintf(dp[COS2PHI], "%f %e\n", time, ncos2avg[species]);
fprintf(dp[COS3PHI], "%f %e\n", time, ncos3avg[species]);
fprintf(dp[SINPHI], "%f %e\n", time, nsin1avg[species]);
fprintf(dp[SIN2PHI], "%f %e\n", time, nsin2avg[species]);
fprintf(dp[SIN3PHI], "%f %e\n", time, nsin3avg[species]);

for(count=0; count<FCOMP; count++){
    fclose(dp[count]);
}
}

void averagecurrents()
{
    int timeparameter = calcbutnoprnt[species] + 1;

    static double boot[2][CALC_TO_PRINT_RATIO+1];
    static double pscos1t[2][CALC_TO_PRINT_RATIO+1];
    static double pscos2t[2][CALC_TO_PRINT_RATIO+1];
    static double pscos1p[2][CALC_TO_PRINT_RATIO+1];
    static double pscos2p[2][CALC_TO_PRINT_RATIO+1];

    double multiplier = 1.0/((double)(CALC_TO_PRINT_RATIO));
    double deltaboot;
    double deltaps1theta;
    double deltaps2theta;
    double deltaps1phi;
    double deltaps2phi;

    /*****
    /*          Before the first print out          */
    /*          (not all elements of the array are occupied)          */
    /*****
    if(!runningavgflag[species]){
        boot[species][timeparameter] = bootstrap[species];
        pscos1t[species][timeparameter] = pscostheta[species];
        pscos2t[species][timeparameter] = pscos2theta[species];
        pscos1p[species][timeparameter] = pscosphi[species];
        pscos2p[species][timeparameter] = pscos2phi[species];
        bootstrapavg[species] += multiplier*boot[species][timeparameter]/
bsqavg;

```

```

    pscosthetaavg[species] += multiplier*pscos1t[species][timeparameter]/bsqavg;
    pscos2thetaavg[species] += multiplier*pscos2t[species][timeparameter]/bsqavg;
    pscosphiavg[species] += multiplier*pscos1p[species][timeparameter]/bsqavg;
    pscos2phiavg[species] += multiplier*pscos2p[species][timeparameter]/bsqavg;
}

/*****
/* After the first print out the elements[] are changed. */
/* A running average over the fraction of the collision time */
/* corresponding to PRINT_PER_COLTIME/CALCULATE_PER_COLTIME is */
/* kept to minimize statistical fluctuations */
/*****
else{
    deltabout = bootstrap[species] - boot[species][timeparameter];
    bootstrapavg[species] += multiplier*deltabout/bsqavg;
    boot[species][timeparameter] = bootstrap[species];

    deltaps1theta = pscostheta[species] - pscos1t[species][timeparameter];
    pscosthetaavg[species] += multiplier*deltaps1theta/bsqavg;
    pscos1t[species][timeparameter] = pscostheta[species];

    deltaps2theta = pscos2theta[species] - pscos2t[species][timeparameter];
    pscos2thetaavg[species] += multiplier*deltaps2theta/bsqavg;
    pscos2t[species][timeparameter] = pscos2theta[species];

    deltaps1phi = pscosphi[species] - pscos1p[species][timeparameter];
    pscosphiavg[species] += multiplier*deltaps1phi/bsqavg;
    pscos1p[species][timeparameter] = pscosphi[species];

    deltaps2phi = pscos2phi[species] - pscos2p[species][timeparameter];
    pscos2phiavg[species] += multiplier*deltaps2phi/bsqavg;
    pscos2p[species][timeparameter] = pscos2phi[species];
}

    bootstrapavgoverv[species] = bootstrapavg[species]/VAVERAGE[species];
    pscosthetaavgoverv[species] = pscosthetaavg[species]/VAVERAGE[species];
    pscos2thetaavgoverv[species] = pscos2thetaavg[species]/VAVERAGE[species];
    pscosphiavgoverv[species] = pscosphiavg[species]/VAVERAGE[species];
    pscos2phiavgoverv[species] = pscos2phiavg[species]/VAVERAGE[species];
}

void resetcurrents()

```

```

/*****
/*****
/** Reset all currents, min- and maxincrement[species] and      **/
/** ommitcount[species] to zero for next time.                  **/
/*****
/*****
{
  bootstrap[species]   = 0.0;
  pscostheta[species]  = 0.0;
  pscos2theta[species] = 0.0;
  pscosphi[species]    = 0.0;
  pscos2phi[species]   = 0.0;
  bootpassing[species] = 0.0;
  boottrapped[species] = 0.0;
  passing[species]     = 0;
  trapped[species]     = 0;
  bsqavg               = 0.0;
  ommitcount[species]  = 0;
  evaluateflag        = 1;
}

void printparameters(float time)
{
                                /* print important parameters*/
  double qnpotential = 0.0;
  int count;
  enum PARAMETERS {P_THETA, P_PHI, P_LAMBDA, P_WEIGHT, P_PARENERGY, P_NUM};
  static FILE *pp[P_NUM];
  char name[P_NUM][15] = {"theta", "phi", "lambda", "weight", "parenergy"};
  char filename[15];
  for(count = 0; count < P_NUM; count++){
    sprintf(filename, "%f%s%i", time, name[count], species);
    pp[count] = fopen(filename, "ab");
  }

  fprintf(pp[P_THETA],    "%i %e\n", p, THETA[species][p]);
  fprintf(pp[P_PHI],     "%i %e\n", p, PHI[species][p]);
  fprintf(pp[P_LAMBDA],  "%i %e\n", p, LAMBDA[species][p]);
  fprintf(pp[P_WEIGHT],  "%i %e\n", p, WEIGHT[species][p]);

#ifdef QUASI_NEUTRALITY
  qnpotential = eqNPOT(THETA[species][p], PHI[species][p]);
#endif
  fprintf(pp[P_PARENERGY], "%i %e\n", p, (ENERGY[species][p]-qnpotential));

  for(count = 0; count < P_NUM; count++){
    fclose(pp[count]);
  }
}

```

```

void printstuff()
/*****
/*****
/** Print bootstrap and P.S. currents, totalparticles and increments.**/
/*****
/*****
{
    int count;
    int pcount;
    float time = 0.0;
    static float oldtime[2];
    enum FILES {BOOTSTRAP, PSCOSTHETA, PSCOS2THETA, PSCOSPFI, PSCOS2PHI,
TOTALNUMBER, MAXINCREMENT, MININCREMENT, NUMFILES};
    static FILE *fp[NUMFILES];
    char filename[20];
    char name[NUMFILES][20] = {"bootavg","pscsthetaavg","psc2thetaavg"
, "pscospfiavg","psc2phiavg","totalnumber","maxincrement","mininceme
nt"};

    /* calculate the fraction of the C.T. at this point in time */
    for(pcount=1; pcount< PARTICLES; pcount++){
        time += (((float)(totaltime[species][pcount]))/TAU[species][pcount]
)/(((float)(PARTICLES-1))); /* MUST CHANGE IF GAUSSIAN E DIST */
    }
    if(time >= COLLISION_TIMES) ende[species] = 1;

    for(count=0; count<NUMFILES; count++){
        sprintf(filename, "%s%i", name[count], species);
        fp[count] = fopen(filename, "ab");
    }

    fprintf(fp[BOOTSTRAP], "%f %e\n", time, bootstrapavg[species]);
    fprintf(fp[PSCOSTHETA], "%f %e\n", time, pscsthetaavg[species]);
    fprintf(fp[PSCOS2THETA], "%f %e\n", time, psc2thetaavg[species]);
    fprintf(fp[PSCOSPFI], "%f %e\n", time, pscospfiavg[species]);
    fprintf(fp[PSCOS2PHI], "%f %e\n", time, psc2phiavg[species]);
    fprintf(fp[TOTALNUMBER], "%f %i\n", time, totalnumber[species]);
    fprintf(fp[MAXINCREMENT], "%f %e\n", time, maxincrement[species]);
    fprintf(fp[MININCREMENT], "%f %e\n", time, minincrement[species]);

    for(count=0; count<NUMFILES; count++){
        fclose(fp[count]);
    }

    /*****
    /** This happens once per collision time. Diffusion is calculated**/
    /** and printed and density components are calculated and printed.**/
    /*****
    if((time-oldtime[species]) > 0.95){
        calculateandprintdiffusion(time);

```

```

        calculateandprintdensity(time);
        oldtime[species] = time;
    }
    /* for(pcount=1; pcount< PARTICLES; pcount++){
        printparameters(time);
    }
    */
}

double maxfield()
{
    double Bmax;
    double Bold;
    double thetamax = 0.0;
    double inc = 0.01;
    Bmax = 1.0/sqrt(1.0 + C01*epsilon*sqrt(Psimiddle)*cos(thetamax) + C02
*epsilon*epsilon*(Psimiddle)*cos(2.0*thetamax));
    while(fabs((Bmax-Bold)/Bmax) > 1e-15){
        do{
            Bold = Bmax;
            thetamax += inc;
            Bmax = 1.0/sqrt(1.0 + C01*epsilon*sqrt(Psimiddle)*cos(thetamax) +
C02*epsilon*epsilon*(Psimiddle)*cos(2.0*thetamax));
            printf("%f %f %f\n", thetamax, Bmax, inc);
        }
        while((Bmax-Bold) > 0);
        inc = -0.5*inc;
    }
    return(Bmax);
}

float rando()
{
    float rand = drand48();
    return(rand);
}

```

## References

- [1] K. Nishikawa and M. Wakatani. *Plasma Physics*. Springer-Verlag, Berlin, 1990.
- [2] *edit.* R. Dendy. *Plasma Physics: an Introductory Course*. Cambridge University Press, 1993.
- [3] J. Wesson. *Tokamaks*. Clarendon Press, Oxford, 1987.
- [4] A.H. Boozer. *Plasma Physics*. 1991.
- [5] A.H. Boozer. *Statistical Physics*. 1992.
- [6] R.B. White. *Theory of Tokamak Plasmas*. North-Holland Physics, Amsterdam, 1989.
- [7] E. Teller. *Fusion*. Academic, New York, 1981.
- [8] D.R. Nicholson. *Introduction to Plasma Theory*. John Wiley & Sons, New York, 1983.
- [9] J.D. Huba. *NRL Plasma Formulary*. The Office of Naval Research, Washington, 1994.
- [10] L. Tonks and I. Langmuir. *Phys. Rev*, 33:195, 1929.
- [11] L. Schmitz B.J. Lee, B. Merriman and F. Najmabadi. Divertor modeling for reactor. *Sherwood Fusion Conference Proceedings*, page 1C33, 1995.
- [12] S. Ohtsu and B.J. Braams. Modelling of neon puffing and pumping in ITER divertor. *Sherwood Fusion Conference Proceedings*, page 1C38, 1995.
- [13] A.H. Kritz et al. EC heating and current drive for advanced-tokamak operation in ITER. *Sherwood Fusion Conference Proceedings*, page 1D9, 1995.
- [14] B.J. Braams and S. Ohtsu. High power radiative divertor scenarios for ITER and TPX. *Sherwood Fusion Conference Proceedings*, page 3C8, 1995.
- [15] L.D. Pearlstein et al. Robust multivariable shape and stability control for ITER. *Sherwood Fusion Conference Proceedings*, page 3C10, 1995.
- [16] W. Arter. Numerical simulation of magnetic fusion plasmas. *Report on Progress in Physics*, 58:1, 1995.
- [17] M. Sasinowski and A.H. Boozer. *Phys. Plasma*, 2:610, 1995.
- [18] F.L. Hinton and M.N. Rosenbluth. *Phys. Fluid*, 16:836, 1973.
- [19] A.H. Boozer and H.J. Gardner. *Phys. Fluid B*, 2:2408, 1990.

- [20] Y. Wu and R.B. White. *Phys. Fluid B*, 5:3291, 1993.
- [21] P.H. Rutherford. *Phys. Fluid*, 13:482, 1970.
- [22] A. Pytte and A.H. Boozer. *Phys. Fluid*, 24:88, 1981.
- [23] A.H. Boozer and G. Kuo-Petravic. *Phys. Fluid*, 24:851, 1981.
- [24] W. Lotz and J. Nührenberg. *Phys. Fluid*, 31:2984, 1988.
- [25] A.H. Boozer. *Phys. Fluid*, 24:1999, 1981.
- [26] R.D. Hazeltine M.N. Rosenbluth and F.L. Hinton. *Phys. Fluid*, 15:116, 1972.
- [27] R.B. White R.J. Goldston and A.H. Boozer. *Phys. Rev. Lett.*, page 647, 1981.
- [28] P.R. Garabedian and M. Taylor. *Nuclear Fusion*, 32:265, 1992.



# VITA

## Maciek Sasinowski

Born in Gdańsk, Poland on July 3, 1971. Graduated from Meadowbrook High School in May of 1988 as a foreign exchange student from Germany. Entered Youngstown State University in the fall of that year and obtained a B.S. in physics in May of 1991. Began the graduate program at The College of William and Mary in the fall of 1991.