

UNIVERSITY OF CALIFORNIA
Los Angeles

**Particle-In-Cell Modeling
of Plasma-Based Accelerators
in Two and Three Dimensions**

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy
in Physics

by

Roy Gerrit Hemker

2000

© Copyright by
Roy Gerrit Hemker
2000

The dissertation of Roy Gerrit Hemker is approved.

Steve Cowley

Chandrasekhar J. Joshi

John M. Dawson, Committee Co-Chair

Warren B. Mori, Committee Co-Chair

University of California, Los Angeles

2000

DEDICATION

To my family and my friends

Contents

List of Figures	vi
Acknowledgements	xii
Vita	xiii
Abstract of the Dissertation	xv
1 Introduction	1
1.1 Introduction	1
1.2 Plasma-Based Accelerator Concepts	2
1.3 Research Areas Relevant to Plasma-Based Accelerators	6
1.4 The Role of Simulations in Plasma and Accelerator Research	8
1.5 The Case for the Use of Object-Oriented Simulation Codes	10
1.6 Overview	13
2 Review of Plasma-Based Accelerator Physics	14
2.1 Single Particle Dynamics in a Wakefield	14
2.2 Laser Beams	16
2.3 Charged Particle Beams	19
2.4 Wakefield Generation	21
3 Review of Basic Particle-In-Cell Algorithms	25
3.1 The PIC-Method	25
3.2 PIC Algorithms for 2D-Cartesian Simulations	28
3.3 3D-Cartesian and 2D-Cylindrically-Symmetric Algorithms	35
4 The Implementation of the Object-Oriented Code OSIRIS	43
4.1 Development Strategy and Code Design	43
4.2 High Level Description	47
4.3 Variable Dimension Field Objects	49
4.4 Global and Local Objects	51
4.5 Dynamic Simulation Spaces	55

4.6	The Motion of Internal Boundaries	60
4.7	Multi-Dimensional Issues	64
4.8	Conclusion	66
5	Electron Beam Production Using Multiple Laser Beams in Plasmas	68
5.1	Introduction	68
5.2	Acceptance of a Plasma Wave	70
5.3	Simulation Parameters	72
5.4	2D Simulation Results	74
5.5	3D Simulation Results	83
5.6	1D Models	90
5.7	Conclusion	94
6	Long Wavelength Hosing of Laser Beams	96
6.1	Introduction	96
6.2	Motivation	97
6.3	Theoretical Approach	98
6.4	Simulation Results	103
6.5	Conclusion	107
7	LWFA in a Parabolic Channel	108
7.1	Introduction	108
7.2	Simulation Setup	109
7.3	Simulation Results	111
7.4	Conclusion	120
8	Plasma Wakefield Acceleration in the Blowout Regime	121
8.1	Introduction	121
8.2	2D Cylindrically-Symmetric Simulations	123
8.3	3D Cartesian Simulations	142
8.4	An Analytical Model	150
8.5	Conclusion	161
9	Summary	163
9.1	Important Results	163
9.2	Future Challenges	165
A	OSIRIS - A Brief User's Guide	168
A.1	General Information	168
A.2	An Input File Example	171
	Bibliography	194

List of Figures

1.1	Plasma-based accelerator concepts: a) Plasma Wakefield Accelerator (PWFA) b) Laser Wakefield Accelerator (LWFA) c) Plasma Beatwave Accelerator (PBWA) d) Self-modulated Laser Wakefield Accelerator (SMLWFA)	3
1.2	Comparison of the increase of energy in plasma-based accelerations against the Livingston curve (Courtesy of T. Katsouleas)	5
1.3	Multidisciplinary areas which have contributed to plasma-based accelerator research	7
1.4	Increase in computing speed and memory for different supercomputer architectures	11
2.1	The figure illustrates the evolution of the spot size of a Gaussian beam during its propagation through a vacuum. After propagating away from the focal plane at $z = 0$, where the beam has its minimum spot size w_0 , to a distance of one Rayleigh length z_R the spot size has increased to $\sqrt{2}w_0$	18
3.1	The basic loop for PIC simulations. Time is increased in steps of Δt so that $t = t_o + n \times \Delta t$	27
3.2	The grid for a 2D PIC simulation. The staggered spacing of the E , B , and j components, and of ρ allows for a higher precision of the calculations	28
3.3	The figure shows an example of the paths over which the ISIS method averages using the particle position before and after the particle push. It also shows where current is deposited on the grid for the case of this example.	33
3.4	The figure shows an example of the paths over which the TRISTAN method averages using the particle position before and after the particle push. It also shows where current is deposited on the grid for the case of this example.	34
3.5	The grid for a 3D PIC simulation. The staggered placing of the E , B , and j components, and of ρ allows for a higher precision of the calculations	36

3.6	The grid for a 2D cylindrically-symmetric PIC simulation. The axis has been placed through the middle of the first grid cell in order to avoid having to calculate j_z on for $r = 0$	37
3.7	The charge represented by a simulation particle in the r - z -plane is a ring	39
3.8	The position update for a particle in 2D cylindrically-symmetric coordinates. The use of a “pseudo” 3D push followed by a rotation also requires an update of the momentum.	41
4.1	The flow control diagram of OSIRIS. It follows the general structure of Fig. 3.1 but shows differences that arise from the specific details of the implementation of OSIRIS.	48
4.2	The class hierarchy of OSIRIS. The figure shows most of the classes and modules used but omits some for the goal of clarity.	49
4.3	The definition of the VDF type in OSIRIS	50
4.4	The code running on each node has several instances of a grid object. The global grid object describes the the grid of the whole simulation (black) and how it has moved. The local grid objects that are part of each VDF object contain the same information for the domain assigned to a specific node (red).	53
4.5	The possible decompositions in a 3D simulation.	54
4.6	Two different concepts for objects on a parallel computer. In a parallel object communication is done by the methods of a class whenever information from another node is needed within that method. The concept of one object-per-node treats boundaries with other nodes as one particular kind of boundary condition.	55
4.7	For a dynamic space the boundaries can move inward or outward from there current position. A moving window is the special case of the front boundary moving outward and the back boundary moving inward.	57
4.8	The code running on each node has two instances of a space object. The global space object describes the space of the whole simulation and its motion. The local space object contains the same information for the domain assigned to the specific node. The figure shows the update of the space boundaries explicitly only for the global object but for the local space the same updating of X_{\min} and X_{\max} is done.	59
4.9	The figure shows how the motion of the lower boundary of the global and local grid is followed. The upper boundaries are tracked in the same way.	61

4.10 If a boundary between nodes moves then the boundary condition handling this case has to move the necessary data. This figure shows the motion of particles between nodes. The red, blue, and yellow colors for the particles in the figure are used to distinguish the different groups of particles. The red and blue particles are originally on the left and right node respectively. The yellow particles are newly initialized particles.	63
4.11 The communication pattern of OSIRIS for 2D domain decomposition. The communications and boundary handling takes place for one direction at a time	65
5.1 Geometry of the cathodeless injector concept. The injection phase of the injection pulse is defined by the distance between the trailing edge of the drive pulse and the center of the injection pulse when it crosses the drive pulse.	69
5.2 The number of trapped electrons, the normalized emittance, and the energy spread of the trapped particles as a function of the injection phase. The injection amplitude b is 2.0 and the drive amplitude a is 1.0. The connecting lines between the data points have been added to make it easier to distinguish the different data. The inset shows the raw data for the transverse phase space of the trapped particles that is used to calculate the emittance for the simulation at $\psi = 1.8\pi$. . .	75
5.3 The number of trapped electrons, the normalized emittance, and the energy spread of the trapped particles as a function of the injection amplitude. The injection phase ψ is 1.3π . All other parameters are the same as the ones used in the simulations of Fig. 5.2. The connecting lines between the data points have been added to make it easier to distinguish the different data.	76
5.4 The figure shows the initial position of trapped particles for two different simulations. The red particles come from a simulation with $\psi = 1.8\pi$ and $b = 2.0$. The blue particles come from a simulation with $\psi = 1.3\pi$ and $b = 1.8$. The position of the drive pulse in the figure is illustrative and does not match $\psi = 1.3\pi$ or $\psi = 1.8\pi$	82
5.5 p_1 of a test particle as a function of time. The two curves are the results from simulations with (solid) and without (dashed) an injection pulse. $\psi = 1.3\pi$ and $b = 1.8$ for the simulation with an injection pulse. The initial position of the test particle is given by offsets of -2.3 in x_1 and -0.1 in x_2 relative to the intersection of the pulses (see Fig. 5.4). The vertical line in the figure indicates the time $t = 42.0$ at which the electric fields are given in Fig. 5.6.	84
5.6 The E_1 field at the time $t = 42.0$ for $\psi = 1.3\pi$ and $b = 1.8$. The cross indicates the position of the test particle shown in Fig. 5.5	85

5.7	The longitudinal phase space x_1-p_1 of the injected particles at the end of the 3D simulation.	88
5.8	The transverse phase space data at the end of the 3D simulation. (a) shows the x_1-x_2 and the x_1-x_3 distribution of the injected particles. (b) shows the x_1-p_2 and the x_1-p_3 distribution of the injected particles particles.	89
5.9	p_1 vs. time for a particle in a 2D non-self-consistent simulation (solid) and for a 1D numerical calculation(dashed). The 1D calculation had the starting parameters $x_0 = -0.5$, $w_0 = -1.5$ and $\varphi_0 = \frac{4}{3} \cdot \pi$	91
6.1	A sequence of color contours of the laser's electric field in units of $eE / (m\omega_0) \simeq a$. The results are the same from the same simulation.	99
6.2	The growth rate for hosing vs. wavenumber for $\tilde{x}_R = 256$	102
6.3	Color contours of the laser's electric field in units of $eE / (mc\omega_0) \simeq a$ to show further evidence for long wavelength hosing. The results are from three different simulations.	105
6.4	Color contour of electron density showing self-trapped electrons exiting the plasma. The results are from the same simulation as Fig. 6.3a).	107
7.1	The density profile of the plasma channel modeled in the simulation.	110
7.2	The envelope of the matched laser beam after about half a Rayleigh length and after about 27 Rayleigh lengths of laser propagation	112
7.3	The electric field of the plasma wake after about half a Rayleigh length and after about 27 Rayleigh lengths of laser propagation	114
7.4	The longitudinal, $x_1 p_1$, phase space of the test particles with a momentum $p_1 \geq 15m_e c$ after 27 Rayleigh lengths of propagation. Note that the x_1 axis of the plots only extends over about the last 1/6-th of the simulation window.	115
7.5	The spatial distribution of the test particles with a momentum $p_1 \geq 15m_e c$ after 27 Rayleigh length of propagation. Note that the x_1 axis of the plots only extends over about the last 1/6-th of the simulation window.	116
7.6	The $x_1 p_2$ -phase space of the test particles with a momentum $p_1 \geq 15m_e c$ after 27 Rayleigh length of propagation. Note that the x_1 axis of the plots only extends over about the last 1/6-th of the simulation window.	117
8.1	The setup of the E-157 experiment at SLAC.	122
8.2	The figure shows plots of several quantities at the beginning of the simulations just after the beam fully entered the plasma. See the main text for a detailed explanation of the plotted quantities.	125

8.3	The figure shows plots of several quantities at the first minimum of the betatron oscillation of the beam after $\sim 191\text{mm}$ of propagation through the plasma. See the main text for a detailed explanation of the plotted quantities.	126
8.4	The figure shows plots of several quantities at the first maximum of the betatron oscillation of the beam after $\sim 396\text{mm}$ of propagation through the plasma. See the main text for a detailed explanation of the plotted quantities.	127
8.5	The figure shows plots of several quantities at the end of the simulations after $\sim 1.4\text{m}$ of propagation through the plasma. See the main text for a detailed explanation of the plotted quantities.	128
8.6	A radial lineout of the plasma charge density at the center of the beam after 1.4 meters of propagation.	134
8.7	The beam current, the plasma current, and the total current for each 0.12 pico-second bin at the beginning of the simulation just after the beam fully entered the plasma.	136
8.8	The mean, maximum, and minimum momentum in the propagation direction, p_z , as well as the number of electrons for each 0.12 pico-seconds bin after 1.4 meters of propagation using the full PIC simulation to propagate the beam.	138
8.9	The mean, maximum, and minimum momentum in the propagation direction, p_z , as well as the number of electrons for each 0.12 pico-seconds bin after 1.4 meters of propagation using the initial fields at the initial positions of the particles to propagate the beam.	139
8.10	The initial response of plasma in simulations with (a) $N_1 = 3.7 \times 10^{10}$ beam electrons and (b) $N_2 = 1.85 \times 10^{10}$ beam electrons.	140
8.11	The lineout of the accelerating field along the axis for a simulation with $N_1 = 1.85 \times 10^{10}$	141
8.12	The mean, maximum, and minimum momentum in the propagation direction, p_z , as well as the width of the distribution of p_z , σ_{p_z} , for each 0.12 pico-seconds bin after 1.3 meters of propagation using the full 2D cylindrically-symmetric PIC simulation to propagate the beam.	143
8.13	The axial lineout of the accelerating field of simulations of the PWFA using three different grid resolutions. The number of particles per cell is the same for all three simulations.	144
8.14	The lineouts of the accelerating field along the axis of the beam for a 2D cylindrically-symmetric and a 3D Cartesian simulation with the same beam and plasma parameters.	146
8.15	The lineouts of the accelerating electric field along the axis of the beam in the propagation direction z for different aspect ratios of the transverse beam spotsizes: (a) 1:1 (b) $2:\frac{1}{2}$ (c) $1:\frac{1}{4}$ (d) 0.9722:0.4348	147

8.16 Selected isosurfaces of the accelerating field. The dark blue, light blue, green, and yellow surfaces corresponds to an acceleration gradients of 0.5 , 0.4, 0.2, and 0.1 GeV/m while the red surfaces correspond to a decelerating gradient of 0.1 GeV/m. The fields shown in the left column and right column are from the simulation with an aspect ratio of 1:1 and 2: $\frac{1}{2}$ respectively. This figure has been made possible by the help of the Office of Academic Computing at UCLA.	149
8.17 Solutions for the electron trajectories for 13 different initial radii starting at $r_i = \sigma_r$ and then increasing in equal steps of $\sigma_r/3$. The beam has $N_b = 3.7 \times 10^{10}$. The upper part of the picture shows the results using the same scaling for both axes. The lower part of the figure shows the same results but with a blown up radial axis in order to show more details. The vertical line in the center of the figure indicates the center of the electron beam.	156
8.18 Solutions for the electron trajectories for 60 different initial radii starting at $r_i = \sigma_r/12$ and then increasing in equal steps of $\sigma_r/12$. The beam has $N_b = 3.7 \times 10^{10}$. The upper part of the picture shows the results using the same scaling for both axes. The lower part of the figure shows the same results but with a blown up radial axis in order to show more details. The vertical line in the center of the figure indicates the center of the electron beam.	158
8.19 Solutions for the electron trajectories for 60 different initial radii starting at $r_i = \sigma_r/12$ and then increasing in equal steps of $\sigma_r/12$. The beam has $N_b = 1.85 \times 10^{10}$. The upper part of the picture shows the results using the same scaling for both axes. The lower part of the figure shows the same results but with a blown up radial axis in order to show more details. The vertical line in the center of the figure indicates the center of the electron beam.	159
8.20 The forces acting on an electron starting at an initial radius $r_i = \sigma_r$	160

ACKNOWLEDGEMENTS

First of all, I want to express my thanks and appreciation to Prof. Warren Mori. His strong support and his guidance throughout my research were essential to the success of this work. I would not have been able to complete this project without it.

I would like to thank Prof. John Dawson and Prof. Chan Joshi for their support and active interest in my research. In their conversations with me they have inspired me with their broad view of the field of plasma physics which helped me greatly to see my own research in perspective. My special thanks goes to Dr. Viktor Decyk. Much of the work in this dissertation builds on the foundation of his ideas. His suggestions and advice have been invaluable.

I am grateful to Seung Lee for her help with the cylindrically-symmetric algorityhms, Dr. Frank Tsung for his help with the 3D current deposition scheme, and Brian Duda for his collaboration on the long wavelength hosing. My discussions with Dr. Kuo-Cheng Tzeng and Prof. Tom Katsoules helped me with many parts of my research and contributed significantly to the success of this work.

Last but not least I would like to thank the staff, researchers, and students who I had the pleasure of working with for making UCLA such a friendly and welcoming place.

VITA

1993	M.S., Physics California State University, Northridge
2000	Ph.D. in Physics University of California, Los Angeles

PUBLICATIONS

- R.G. Hemker, W.B. Mori, S. Lee, T. Katsouleas, “Three-dimensional wake structures from asymmetric drive beams”, in preparation.
- F.S. Tsung, R.G. Hemker, C. Ren, L.O. Silva, W.B. Mori, T. Katsouleas, “Generation of a single-cycle laser pulse via photon deceleration”, submitted to Phys. Rev. Lett.
- R.G. Hemker, W.B. Mori, S. Lee, T. Katsouleas, “Dynamic Effects in Plasma Wakefield Excitation”, submitted to Phys. Rev. Spec. Top. - Acc.&Beams.
- S. Lee, T. Katsouleas, R.G. Hemker, W.B. Mori, E. Esarey, C. Schroeder, “Simulations of a meter long plasma wakefield accelerator”, accepted by in Phys. Rev. E.
- B.J. Duda, R.G. Hemker, K.C. Tzeng, W.B. Mori, “A long-wavelength hosing instability in laser-plasma interactions”, Phys. Rev. Lett., vol. 83, 1978(1999).
- R.G. Hemker, F.S. Tsung, V.K. Decyk, W.B. Mori, S. Lee, T. Katsouleas, “Development of a Parallel Code for Modeling Plasma Based Accelerators”, Proceedings of the 1999 Particle Accelerator Conference.
- R.G. Hemker, K.-C. Tzeng, W.B. Mori, C.E. Clayton, T. Katsouleas, “Computer simulations of cathodeless, high-brightness electron-beam production by multiple laser beams in plasmas”, Phys. Rev. E, vol. 57, 5920(1998).

R.G. Hemker, K.C. Tzeng, W.B. Mori, C.E. Clayton, T. Katsouleas, "Cathodeless, high-brightness electron-beam production by multiple laser beams in plasmas", Proceedings of the 1997 Particle Accelerator Conference, vol. 3, p.2870-2 (1998).

N. Kioussis, H. Watanabe, R.G. Hemker, W. Gourdin, A. Gonis, P.E. Johnson, "Effect of boron and hydrogen impurities on the electronic structure of Ni₃Al", Mater. Res. Soc. Proc., vol. 319, 363(1994).

ABSTRACT OF THE DISSERTATION

Particle-In-Cell Modeling of Plasma-Based Accelerators in Two and Three

Dimensions

by

Roy Gerrit Hemker

Doctor of Philosophy in Physics

University of California, Los Angeles, 2000

Professor Warren B. Mori, Co-Chair

Professor John M. Dawson, Co-Chair

In this dissertation, a fully object-oriented, fully relativistic, multidimensional Particle-In-Cell code was developed and applied to answer key questions in plasma-based accelerator research. The simulations increase the understanding of the processes in laser plasma and beam-plasma interaction, allow for comparison with experiments, and motivate the development of theoretical models.

The simulations support the idea that the injection of electrons in a plasma wave by using a transversely propagating laser pulse is possible. The beam parameters of the injected electrons found in the simulations compare reasonably with beams produced by conventional methods and therefore laser injection is an interesting concept for future plasma-based accelerators. Simulations of long laser pulses, such as the ones used in self-modulated laser wakefield acceleration, predict the existence of a

hosing instability with a wavelength longer than the plasma wavelength. It is found that this effect might increase the emittance of electron beams produced by this acceleration method.

Simulations of the optical guiding of a laser wakefield driver in a parabolic plasma channel support the idea that electrons can be accelerated over distances much longer than the Rayleigh length in a channel. Simulations of plasma wakefield acceleration in the nonlinear blowout regime give a detailed picture of the highly nonlinear processes involved. Using OSIRIS, we have also been able to perform full scale simulations of the E-157 experiment at the Stanford Linear Accelerator Center. These simulations have aided the experimentalists and they have assisted in the development of a theoretical model that is able to reproduce some important aspects of the full PIC simulations.

Chapter 1

Introduction

1.1 Introduction

One of the most important practical problems in high energy physics has always been how to increase the energy of the beams that are used in lepton collision experiments. This is also true today but the problem is more fundamental than it was in the past because the conventional design of accelerators is constrained by some fundamental physical and practical limits. The maximum acceleration gradient that can be achieved using RF-waveguides in existing facilities, e.g., SLAC, is of the order of about 25MeV/m. This technology is limited because the material used to build the waveguides will be destroyed by tunneling ionization at field strength near 100MeV/m using existing RF frequencies. This means that in order to get an energy gain of 50GeV an electron would have to be accelerated over 2000m with a gradient of 25MeV/m. The two conventional accelerator designs to achieve this are linear accelerators or accelerator rings which accelerate a particle by sending it repeatedly through the same RF-waveguide for acceleration. For both these designs achieving

higher energies means that they have to get bigger in size.

Current linear accelerators for electrons and positrons are a few miles long and current e^+e^- rings accelerators require diameters of the order of 10 miles. Their energies are a couple of 10GeV. It seems unlikely that accelerators of significantly higher energy and therefore size are going to be build using this conventional technology. What is required is a significant increase in the magnitude of the accelerating field and plasma-based acceleration does offer an answer to this problem.

The remainder of this chapter will first introduce the different basic ideas for plasma-based acceleration and experiments using them. It will then briefly examine the different fields that are of importance to this research and then consider the role that computer simulations can play in physics research and in particular in the research on plasma-based accelerators. Finally, it will briefly explain the usefulness of advanced programming concepts for computer simulations.

1.2 Plasma-Based Accelerator Concepts

Several ways of using a plasma for particle acceleration have been suggested [1, 2, 3]. Fig. 1.1 shows four different concepts. Fig. 1.1 a) shows a plasma wakefield accelerator (PWFA) [3]. This concept uses an electron bunch moving through a plasma to create a wake. The charge of the bunch electrons will push the plasma electrons out of the path of the bunch. These displaced electrons will then oscillate back after the bunch has passed through setting up a plasma oscillation. The phase velocity of the plasma oscillation is the velocity of the electron bunch that created the wake. For highly relativistic electrons this is a velocity very close to speed of light. Since plasma space-charge waves have an electric field component parallel to their propagation

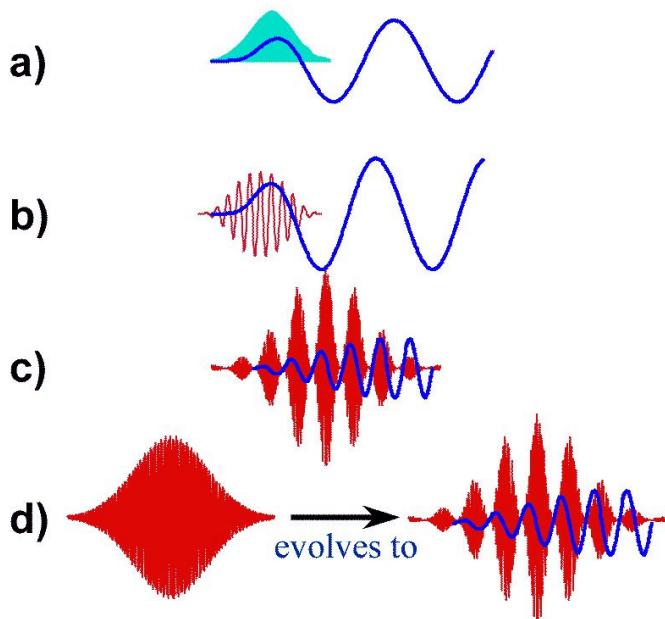


Figure 1.1: Plasma-based accelerator concepts: a) Plasma Wakefield Accelerator (PWFA) b) Laser Wakefield Accelerator (LWFA) c) Plasma Beatwave Accelerator (PBWA) d) Self-modulated Laser Wakefield Accelerator (SMLWFA)

direction a particle placed in this wake with an energy above a certain threshold energy [4, 5, 6, 7, 8] can stay in phase with this accelerating field for long distances and gain significant amounts of energy. In order for the plasma oscillation to have a large amplitude and therefore a large accelerating field the length of the driving bunch has to be of the order of the plasma wavelength.

The other concepts shown in Fig. 1.1 share the idea of generating a wakefield in a plasma and only differ in the driver used to generate this wakefield. In Fig. 1.1 b), the laser wakefield accelerator (LWFA) concept is shown[1]. This concept relies on the ponderomotive force of a short laser pulse to set up a plasma oscillation. In Fig. 1.1 c), the plasma beatwave accelerator (PBWA) concept is shown[1, 2]. In this concept two laser pulses which are long by comparison with the plasma wavelength and which have frequencies that differ by the plasma frequency are used to excite the wake. The beatwave resulting from these two laser pulses can be viewed as a series of successive pulses each of which has a length of one plasma wavelength. Each of these pulses then contributes to setting up a plasma wave in the same way as in concept b). In Fig. 1.1 d) the so called self-modulated laser wakefield accelerator (SMLFWA) concept is shown [9, 10, 11]. In this concept a long single frequency pulse first generates frequency components shifted by the plasma frequency due to Raman forward scattering. The Raman-scattered light beats with the original frequency generating a beatwave and therefore a plasma wave results similarly as in the PBWA concept.

Using plasma wakes generated by particle beam or laser pulse drivers for particle acceleration has been a topic of research since the idea was first published [1]. However, it has only been during the last 5 to 10 years that significant experimental progress has been made [12, 13]. Fig. 1.2 shows the energy gains measured in

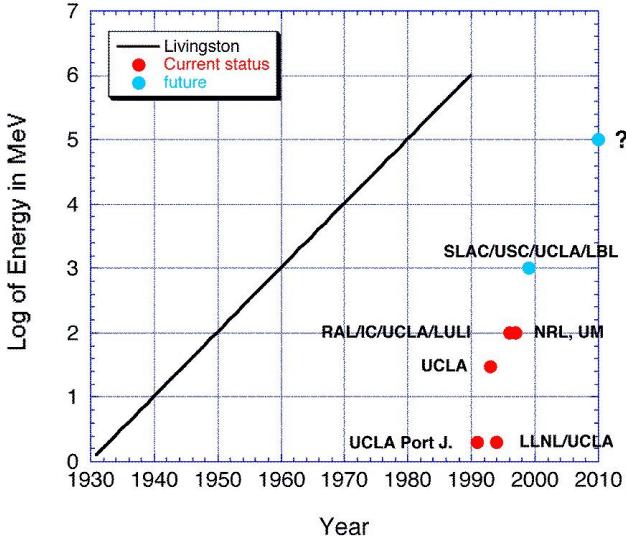


Figure 1.2: Comparison of the increase of energy in plasma-based accelerations against the Livingston curve (Courtesy of T. Katsouleas)

plasma-based acceleration experiments as a function of time. The solid black line indicates the past and projected increase of conventional accelerator technology (the Livingston curve) [14], the red data points show results of past plasma-based accelerators [13], and the blue data points are the expected results of current and future experiments with plasma-based accelerators [15]. The data points for plasma-based acceleration experiments do indeed suggest a faster increase of the output energy than is to be expected for conventional accelerators. Most of the past experiments were based on the PBWA and SMLWFA concept since the laser and plasma parameters for these concepts were easier to realize experimentally. The rapid increase in a laser power and the simultaneous shortening of laser pulse length [16, 17] now make LWFA experiments possible. Several such experiments are being conducted at laboratories around the world.

In contrast to this the most recent experiment indicated in Fig. 1.2 is based on the PWFA concept. The data point labeled with “SLAC/USC/UCLA/LBL” refers to the E-157 experiment conducted at the Stanford Linear Accelerator Center (SLAC) as a collaboration of research groups between SLAC, the University of Southern California (USC), the University of California at Los Angeles (UCLA), and the Lawrence Berkeley Laboratory (LBL)[15]. The goal of this experiment is to use the high quality electron beam generated by the Stanford linear accelerator as a driver for the generation of a wakefield in a plasma. The expected accelerating field for this experiment is up to about $1\text{GeV}/\text{m}$, which is about one order of magnitude larger than what can be achieved with conventional technology. It is worth noting that a significant breakthrough that has made this experiment possible was the construction of a plasma source that is able to generate a uniform plasma over 1m distances [18]. All previous experiments for plasma-based acceleration were limited to a couple of mm for acceleration. Because of the increased length of the acceleration distance in this experiment, energy gains of up to 1GeV are expected.

Modeling the E-157 experiment has served as a motivation for much of the research and code development presented in this dissertation and it will be described in more detail in chapter 8.

1.3 Research Areas Relevant to Plasma-Based Accelerators

Computer simulations play a role in many areas of physics and the topic of this dissertation is at the intersection of several of these research areas. Fig. 1.3 shows schematically the different relevant fields in physics that are of importance to plasma-

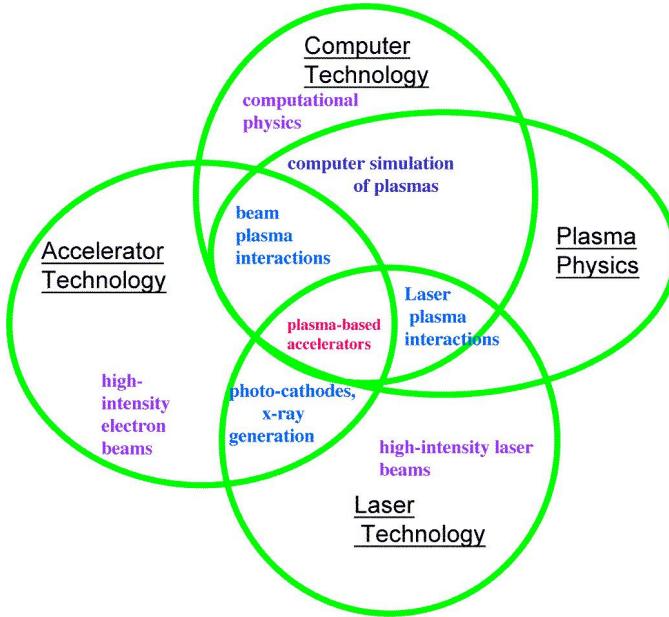


Figure 1.3: Multidisciplinary areas which have contributed to plasma-based accelerator research

based accelerator concepts and how they are connected to each other and computer technology. The importance of computer technology to any research involving simulations is obvious and will be examined further below. The other three fields of importance are laser technology, accelerator physics, which includes charged-beam dynamics, and plasma physics.

There are three key research topics that need to be studied and understood for the successful development of plasma-based accelerators. The first one is the evolution of the drive-beam as it generates the plasma wakefield . The second is the excitation of the wakefield by a given drive beam. The third is evolution of a trailing bunch of particles which is loaded into the wakefield. Each of these topics has been investigated separately, but to get an integrated understanding of an eventual accelerator they have to be answered in a self-consistent manner since the drive beam as well as the

accelerated particle beam are interacting with the plasma wake, i.e., are modifying it and are being modified by it. So far this kind of self-consistent understanding could only be gained by using computer simulations. The importance of the different research areas indicated in Fig. 1.3 is straight forward to understand from these questions. The initial qualities of a laser drive-beam are a problem of laser technology. The initial qualities of a particle drive-beam and the evolution of an accelerated particle bunch are problems also dealt with in accelerator physics and the evolution of either type of drive-beam is a question of laser-plasma or beam-plasma physics.

1.4 The Role of Simulations in Plasma and Accelerator Research

The process of science is mainly a process of comparing the predictions of theories and hypotheses with the results of actual experiments. If a prediction does not agree with a measured result then either the theory that gave rise to the prediction, or the manner in which the prediction was inferred from the theory has to be modified. In this way scientific theories become better and better models of the facts they are trying to explain. Computer simulations like the ones presented in this dissertation come into this picture as a method of bridging the gap between theory and experiment. They are a way to derive predictions from complex and integrated theoretical models which can be compared to experimental results or they are a means to test theories directly.

Problems in many areas of physics today are systems with many degrees of freedom, as for example in plasmas physics. Often the equations that determine the evolution of each degree of freedom over time are very well established, but to track

and understand the simultaneous evolution of more than a few variables is beyond the capacity of the human mind. Computer simulations allow us to do these things. First, they make it possible to derive results from basic theories that can be compared to experimental results. This is particularly important for areas where no simplified analytical model exists at all. Secondly, the insight into the physical processes gained by evaluating the simulation results can in some cases lead to the development of simplified analytical models that are tractable. All these considerations apply directly to the case of plasma physics where research involves a very large number of particles or degrees of freedom.

The view of simulations outlined above is one that follows directly from the standard methodology of science and it should always be kept in mind when using computer simulations. For the case of Particle-In-Cell simulations a different viewpoint, but one which is not in contradiction but is an extension of the one above, is useful as well. Particle-in-cell (PIC) codes used for the research presented in this dissertation make no assumptions in physics except the validity of classical physics. That is, the full set of Maxwell's equations and the relativistic equations of motion for individual particles is self-consistently evolved. Within the validity of classical physics (quantum effects are ignored) and within the limits of numerical accuracy, PIC-simulations of plasmas give exact and very detailed information on the processes within a plasma. They could therefore be considered as numerical experiments, that provide a third kind of methodology to the scientific method on an equal footing with experiment and theory. As such the value of simulations lies in providing us with extremely detailed and accurate information about a simulated problem to an extent that far exceeds the possibilities of either theory or experiment.

1.5 The Case for the Use of Object-Oriented Simulation Codes

The topic of this dissertation is not only the physics learned by conducting simulations but also the development of a completely new kind of PIC code that uses modern state-of-the-art software methods. Developing a new code like this is the equivalent of the development of a new kind of sophisticated experimental laboratory (apparatus and diagnostic techniques) or the development of a new kind of analytical approach to a theoretical problem. In order to be reproducible not only the results of scientific work but also the methods that were applied need to be well documented. In case of a new method this is particularly important in order to make it possible for others to apply the same method to other problems. The development of a new simulation code requires therefore that the new algorithms used in the code as well as sufficient instructions on how to actually use the code should be documented. This kind of documentation for the newly developed code OSIRIS (Object-oriented Simulation Rapid Implementation System) will be part of this dissertation.

In order to see the necessity of a new approach to PIC-simulations, the possibilities opened up by the rapid increase in available computing power have to be understood. Fig. 1.4 shows the advances made in computing speed over time and it indicates an exponential increase in the computing speed as well as in the available memory. These advances in computational speed and memory now make it possible to do full scale 2D and 3D PIC simulations of laser and beam plasma interactions. However, the increased complexity of these codes and interactions make it necessary to apply modern programming approaches like an object oriented programming style to the development of codes.

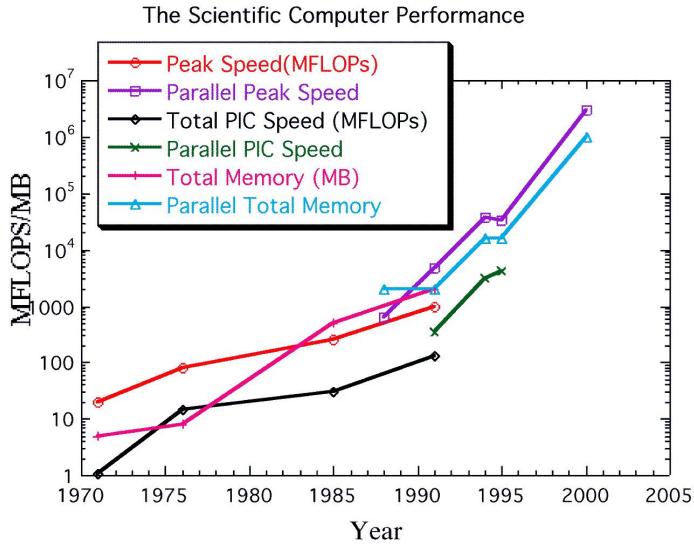


Figure 1.4: Increase in computing speed and memory for different supercomputer architectures

An important fact to note is that the growth in computing power over the last ten years has largely been due to the use of massively parallel computers which now have hundreds of processors. In order to take full advantage of this development it has become necessary to use more complex simulation codes. The increased complexity of codes arises for two reasons. One reason is that the realistic simulation of a problem requires a larger number of more complex algorithms interacting with each other than the simulation of a rather simple model system. For example, initializing an arbitrary laser or particle beam in 3D is a much more difficult problem than doing the same in 1D or 2D. The other reason that simulation codes are becoming more complex is that the computer systems are more complex. Questions a code developer has to consider include things like memory management, operating systems calls, threads, and message passing. As a result the performance obtained from a system

can dramatically differ depending on the code strategy. Parallelized codes that have to handle the problems of parallel communication and parallel I/O are an example of this. A way to deal with this increased complexity is to use an object oriented programming style, which divides the code and data structures into independent classes of objects. This programming style maximizes code reusability and reliability.

The goal of the code development program, that is part of the research presented in this dissertation, was to create a code that breaks up the large problem of a simulation into a set of essentially independent smaller problems that can be solved separately from each other. Object oriented programming achieves this by handling different aspects of the problem in different modules (classes) that communicate through well-defined interfaces. The programming language we chose for this purpose was Fortran 90, mainly because it allowed us to more easily integrate already available Fortran algorithms into the new OSIRIS-framework. As a result of the intensive code development effort OSIRIS now contains algorithms for 1D, 2D, and 3D simulations in Cartesian coordinates and for 2D simulations in cylindrically symmetric coordinates. For all of these algorithms the code is fully relativistic and presently uses a charge-conserving current deposition algorithm. It allows for a moving simulation window and arbitrary domain decomposition for any number of dimensions. This large number of algorithms in one code was only possible due to the object-oriented style of the code. It makes the code a useful tool for many different research problems with the possibility to be extended much further by adding new modules.

There are past and ongoing efforts by other plasma simulation research groups to take advantage of object-oriented programming. Forslund [19] introduced a parallel object-oriented PIC code written in C++ that ran in parallel on a network of workstations. Haney [20] used a hybrid C++/Fortran77 code for tokamak modeling.

Reynders [21] seems to have continued Forslund's work and developed an object-oriented particle simulation library. An ongoing effort to develop an object-oriented library for scientific programming including plasma simulations is the C++ based POOMA project [22]. Verboncoeur [23] developed the object-oriented 2D parallel code OOPIC using C++.

The use of Fortran90 for object-oriented codes in plasma physics has been investigated by Norton [24] and Decyk[25, 26]. Many of the code development results presented in this dissertation build on the results of their research. Qiang [27] has used Fortran90 to develop an object-oriented code for electrostatic simulations of beam dynamics in linear accelerators. The contribution made in this dissertation is a code that combines already existing and new algorithms in a way that leads to significantly improved qualities with regard to operating and extending the code.

1.6 Overview

This chapter has tried to explain the motivations that led to the research results presented in this dissertation. The remainder of this dissertation is structured as follows. First a brief review of basic physics and computer algorithms will be given. This will be followed by a presentation of the implementation details of the new code OSIRIS with particular emphasis on newly developed algorithms. The remaining chapters will then present research results for laser injection of electrons into a plasma based accelerator, long wave length hosing of lasers in plasmas, laser wakefield acceleration in a parabolic plasma channel, and plasma wakefield excitation and acceleration in the blowout regime.

Chapter 2

Review of Plasma-Based Accelerator Physics

This chapter will review the basic physics that governs the behavior of plasma-based wakefield accelerators in general and their drivers, lasers and particle beams. The equations and symbols introduced in this chapter will be used throughout this dissertation.

2.1 Single Particle Dynamics in a Wakefield

We will start with reviewing the behavior of a single particle in a given wakefield. Consider an electron being accelerated in a plasma wave of the form

$$\phi = \phi_0 \left(1 - x_2^2/w_p^2\right) \sin [k_p (x_1 - v_\phi t)] \quad (2.1)$$

where v_ϕ is the phase velocity of the wave and w_p is a parameter describing the width of the plasma wave. This potential describes the behavior of particles close to the

center of a typical plasma wave. We assume $v_\phi \cong c$, i.e., relativistic plasma waves. The subscripts 1 and 2 refer to directions parallel and perpendicular, respectively, to the plasma wave's direction of propagation. The equations of motion for an individual electron are

$$\frac{d}{dt}p_1 = -eE_1 = e\phi_0 k_p \left(1 - x_2^2/w_p^2\right) \cos [k_p (x_1 - v_\phi t)] \quad (2.2)$$

$$\frac{d}{dt}p_2 = -eE_2 = -2e\phi_0 \frac{x_2}{w_p^2} \sin [k_p (x_1 - v_\phi t)] \quad (2.3)$$

The acceleration of single electrons in these fields has been studied extensively [4, 6, 7, 28]. An injected electron accelerated along the axis, $x_2 = 0$, will be trapped if its injection energy (the initial kinetic energy) exceeds the trapping threshold[4, 6, 7, 8].

$$W_i \approx mc^2 \left(\gamma_\phi^2 \left\{ \bar{\phi}_0 + 1/\gamma_\phi - \beta_\phi \left[(\bar{\phi}_0 + 2/\gamma_0) \bar{\phi}_0 \right]^{1/2} \right\} - 1 \right) \quad (2.4)$$

$$\text{with } \bar{\phi}_0 = e\phi_0 / (mc^2)$$

which reduces to $\frac{1}{2} [\bar{\phi}_0 + (1/\bar{\phi}_0)] - 1$ as $\gamma_\phi \rightarrow \infty$. Here $\beta_\Phi = v_\Phi/c$ and $\gamma_\Phi = 1/\sqrt{1 - (v_\Phi/c)^2}$.

Once trapped an electron is accelerated and its speed eventually exceeds the phase velocity of the wave. The acceleration process ceases after the electron outruns the wave and encounters decelerating forces. If $x_2 = 0$, then the maximum energy gain is [1, 4, 6, 7, 8].

$$W_f - W_i \equiv \Delta W \cong 2\gamma_\phi \left[1 + \eta \bar{\phi}_0 \gamma_\phi \right] mc^2 \quad (2.5)$$

where η is 2 if the particle slips through a full π phase of the accelerating bucket. η

usually has a value smaller than 2 depending on certain conditions explained below. ΔW is approximately $2\eta\bar{\phi}_0\gamma_\phi^2mc^2$ if $\bar{\phi}_0\gamma_\phi \gg 1$. The dephasing distance can be estimated by calculating the distance it takes for the electron moving at the speed of light, c , to move forward a half wavelength in a wave moving at $v_\phi \cong c$. This gives [4, 6, 7, 8]

$$L_{dp} = \frac{1}{2}\eta\gamma_\phi^2\lambda_p = \eta\pi\gamma_\phi^2c/\omega_p \quad (2.6)$$

An electron which is not on the axis, $x_2 \neq 0$, will also feel transverse, or so called defocusing/focusing fields, as given by Eq. (2.3). Electrons in the defocusing phase of the wave accelerate away from the axis and are eventually lost [4, 6, 7, 28]. Electrons in the focusing phase execute betatron oscillations (in x_2) as they accelerate along x_1 so only electrons which reside in both focusing and accelerating fields are accelerated to the dephasing limit [4, 6, 7, 28]. These fields are $\pi/2$ out of phase and therefore only a quarter of a plasma wave wavelength can be used for acceleration. This reduces the maximum energy gain and the dephasing length given above by roughly a factor of 2 [i.e., $\eta=1$ in Eq. (2.5)]. In finite-width plasma waves additional second order focusing terms may extend the range of phases which have both focusing and accelerating forces[29, 30]¹. In this case we have $1 < \eta < 2$.

2.2 Laser Beams

A laser beam in vacuum can be described as a Hermite Gaussian beam. This is an solution to the paraxial wave equation which is an approximation to Maxwell's

¹The total dc focusing force is 3/2 times larger than given in Ref. [29], because of an additional electrostatic field.

equations[31]. The lowest order Hermite Gaussian beam propagating in z is given by:

$$E(x, y, z, t) = A \times \frac{e^{-i\Phi(z)}}{\sqrt{1 + \left(\frac{z}{z_R}\right)^2}} \times e^{i\frac{k(x^2+y^2)}{2R(z)}} \times e^{-\frac{(x^2+y^2)}{w(z)^2}} \times e^{i(kz-\omega t)} \quad (2.7)$$

Here we use $\Phi(z) = \arctan \frac{z}{z_R}$ and $R(z) = z + \frac{z_R^2}{z}$. The spotsize w of the laser beam is given by

$$w(z) = w_0 \sqrt{1 + \left(\frac{z}{z_R}\right)^2} \quad (2.8)$$

where the Rayleigh length z_R is defined as $z_R = \frac{\pi}{\lambda} w_0^2 = \frac{1}{2} k w_0^2$. This solution shows that the evolution of the spotsize of a laser beam is characterized by two parameters; it's wavenumber k and the spotsize w_0 in the focal plane where the beam is narrowest. Eq. (2.8) shows that the Rayleigh length is the distance from the focal plane at which the spotsize is $\sqrt{2}$ times the spotsize w_0 in the focal plane and Fig. 2.1 illustrates the physical meaning of z_R and w_0 .

The evolution of the spotsize of an approximately Gaussian beam in either a uniform plasma or a plasma channel is given by the envelope equation for the evolution of the laser spotsize. This equation can be derived by a variety of methods, e.g., the source dependent expansion[32] or the variational principle techniques[33]. For a plasma with a parabolic density profile

$$n(r) = n_0 + \Delta n \frac{r^2}{r_0^2}. \quad (2.9)$$

the envelope equation for the normalized laser spotsize $W(z) = w(z)/w_0$ of a laser beam is [12, 34]

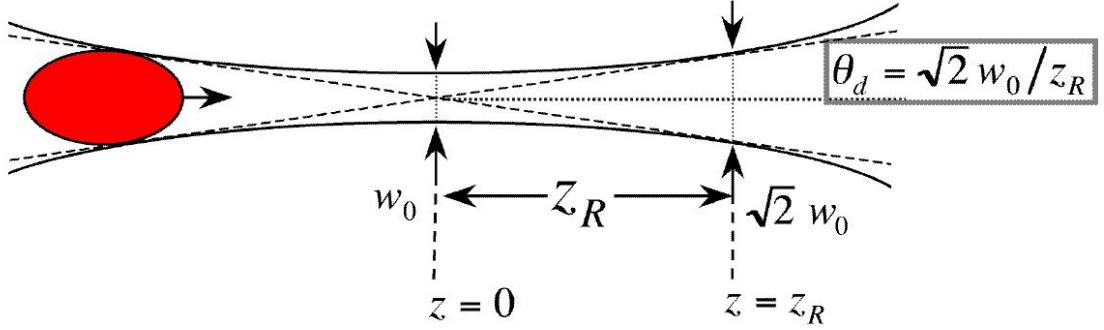


Figure 2.1: The figure illustrates the evolution of the spot size of a Gaussian beam during its propagation through a vacuum. After propagating away from the focal plane at $z = 0$, where the beam has its minimum spot size w_0 , to a distance of one Rayleigh length z_R the spot size has increased to $\sqrt{2}w_0$

$$\frac{d^2W}{dz^2} = \frac{I}{z_R^2 W^3} \left[1 - \frac{P}{P_c} - \frac{\Delta n}{\Delta n_c} W^4 \right] \quad (2.10)$$

where Δn and r_0 characterize the channel and we set $r_0 = w_0$ without loss of generality. The three terms in the bracket are due to I) diffraction, II) relativistic self-focusing, and III) the external focusing forces (e.g., the plasma channel), respectively. $P/P_c = a^2 w_0^2 / 32$ is the laser power normalized to the critical power for relativistic self-focusing, $P_c \cong 17GW \times \left(\frac{\omega}{\omega_p}\right)^2$ [35, 36, 37], and $\Delta n_c = (\pi r_e w_0^2)^{-1}$ with $r_e = e^2 / (m_e c^2)$ (the classical electron radius).

The evolution of the spotsize given by Eq. (2.8) can be recovered as the solution of Eq. (2.10) if only the diffraction term on the right side of the equation is kept. For small laser power, Eq. (2.10) has focusing solutions for normalized spotsizes $W < (\Delta n / \Delta n_c)^{1/4}$ if $\Delta n \geq \Delta n_c$. It also has a stable stationary solution with $W = (\Delta n / \Delta n_c)^{1/4}$ if $\Delta n \geq \Delta n_c$. This stationary solution is the matched beam solution for a parabolic density channel. In the absence of a density channel, there are focusing solutions if $P/P_c > 1$ and a stable stationary solution for $P/P_c = 1$. However, it is

now well known, that a laser pulse with a pulse length $\lesssim \pi c/\omega_p$, as is the case for LWFA drivers, does not relativistically self-focus[38, 39]. Therefore to optically guide a LWFA driver it is necessary to use a density channel [34, 40].

2.3 Charged Particle Beams

The evolution of the spotsize of an accelerated particle beam is determined by its energy, number of particles, spotsize, and normalized emittance ε_n where ε_n is a measure of the area of the beam in transverse phase space. For a relativistic beam (i.e., $\gamma \gg 1$), this area is given by the product of the beam's transverse spot size, σ , angular divergence, $\theta = \Delta p_2/p_1$, and energy, $\gamma \simeq p_1/mc$; therefore $\varepsilon_n = \pi\gamma\theta\sigma \simeq \frac{\Delta p_2}{mc}\sigma$, and it is conserved under ideal conditions. This can be derived by showing that Eq. (2.3) has the adiabatic invariant p_2x_2 for each individual particle.

The envelope equation [41] describes the evolution of the beam's spotsize.

$$\frac{d^2}{dx_1^2} \sigma + \frac{1}{\gamma} \frac{d\gamma}{dx_1} \frac{d\sigma}{dx_1} - \left(\frac{\varepsilon_n}{\pi}\right)^2 \frac{1}{\gamma^2\sigma^3} \left[1 + \frac{2\pi^2}{\gamma} \left(\frac{\sigma}{\varepsilon_n}\right)^2 \frac{I}{I_A} - \frac{\gamma\omega_B^2\sigma^4}{c^2} \left(\frac{\pi}{\varepsilon_n}\right)^2 \right] = 0 \quad (2.11)$$

Here I is the beam's current, $I_a \equiv mc^3/e$, is the Alfvén current, and $\omega_B^2 = 2|\bar{\phi}_0|c^2/w_p^2$ is the betatron frequency for the potential given by Eq. (2.1). The three terms in the bracket are due to I) diffraction, II) self space charge, and III) the external focusing forces (i.e., of the plasma wave), respectively.

The parameter characterizing the ratio of the space charge term to the diffraction term in the beam envelope is given by:

$$\rho = \frac{2\pi^2}{\gamma} \left(\frac{\sigma^2}{\varepsilon_n^2} \right) \frac{I}{I_A} \quad (2.12)$$

If the effects of space charge can be neglected, then the equilibrium state of a matched beam (σ doesn't change during the acceleration) can be obtained by balancing the two remaining force terms. These two terms are the one arising from the diffraction and the transverse external force term. The external force term can be related to the amplitude E_{10} of the accelerating electric field of the plasma wave, which is a quantity we observe in our simulations, i.e., $\phi_0 = -E_{10}/k_p$. The resulting condition for a matched beam is:

$$\frac{1}{4\pi^2\gamma} \frac{mc\omega_p}{eE_{10}} \left(\frac{\varepsilon_n}{\sigma} \right)^2 \left(\frac{w_L}{\sigma} \right)^2 = 1 \quad (2.13)$$

Here, we also replace w_p with $w_L/\sqrt{2}$, where w_L is the laser spot size because the transverse profile of the longitudinal field of the plasma wave is proportional to the transverse profile of the laser intensity $E_{10} \propto E_L^2$, since the ponderomotive force of the laser pulses causes the plasma wake [28, 42]. If the expression on the left side of the equation is larger than unity, the focusing forces dominate diffraction.

The evolution of the particle beam spotsize is the same as the evolution of the laser spotsize given by Eq. (2.8) if all terms except the diffraction term on the right side of Eq. (2.11) can be neglected and the assumptions $\frac{1}{\gamma}, \frac{d\gamma}{dt} \ll 1$ hold. In this case Eq. (2.11) can be rewritten as

$$\frac{d^2}{dx_1^2} \bar{\sigma} = \frac{1}{\left(\frac{\pi}{\varepsilon} \sigma_0^2 \right)^2} \frac{1}{\bar{\sigma}^3} \quad (2.14)$$

where σ_0 is the minimum spotsize, $\bar{\sigma} = \sigma/\sigma_0$ the normalized spotsize, and $\varepsilon = \varepsilon_n/\gamma$

the emittance. If we define the quantity $\beta^* \equiv \frac{\pi}{\varepsilon} \sigma_0^2$ and compare Eq. (2.14) with Eq. (2.10) then it is clear that β^* of a particle beam corresponds to the Rayleigh length z_R of a laser and that the emittance ε of a particle beam corresponds to the wavelength of a laser. Since Eq. (2.8) is the solution of Eq. (2.10) if only the diffraction term is kept we can use Eq. (2.8) also to describe the evolution of a particle beam as long as the approximations mentioned above hold.

2.4 Wakefield Generation

The plasma wave wakes can be generated via beatwave, laser wakefield, Raman forward scattering or plasma wakefield excitation. For the purposes of this dissertation we only review wakefield excitation by short laser or particle beams. We begin with laser wakefield excitation. Two quantities that are helpful to define first when discussing the generation of wakefields are the normalized scalar potential

$$\bar{\Phi} = e\Phi / (m_e c^2) \quad (2.15)$$

and the normalized vector potential

$$\vec{a} = e\vec{A} / (m_e c^2) \quad (2.16)$$

The maximum electric field that a plasma wave in a cold plasma can support is determined by the amplitude at which the wave breaks [43, 44]. It is given by:

$$E_{WB} = \sqrt{2} (\gamma_p - 1)^{1/2} E_0 [V/cm] \quad (2.17)$$

with $\gamma_p = 1/\sqrt{1 - (v_\Phi/c)^2}$ where v_Φ is the phase velocity of the plasma wave. E_0 is

here given by

$$E_0 = m_e c \omega_p / e \simeq 0.96 \sqrt{\frac{n_0}{[cm^{-3}]}} [V/cm] \quad (2.18)$$

The excitation of a plasma wake by a non-evolving circularly polarized laser pulse with cylindrically-symmetric envelope can be solved analytically [42]. For a laser with a Gaussian profile of width L in propagation direction and with a peak normalized vector potential a_0 in a plasma with plasma wavenumber k_p the resulting amplitude of the excited wake is [12]

$$E_{max} = E_0 \left(\sqrt{\pi} a_0^2 / 2 \right) k_p L e^{-\frac{k_p^2 L^2}{4}} \quad (2.19)$$

For the optimal length $L = \lambda_p / (\pi \sqrt{2})$ this becomes

$$E_{max} = E_0 a_0^2 \left(\frac{\pi}{2e} \right)^{1/2} \simeq E_0 0.76 a_0^2 \quad (2.20)$$

The normalized vector potential a_0 is related to the laser intensity I by

$$a_0 = \left(2 e^2 \lambda_L^2 I / (\pi m_e^2 c^5) \right)^{1/2} \quad (2.21)$$

λ_L is here the laser wavelength. The total laser power of a Gaussian beam with the spotsize w_0 is related to the intensity by

$$I = 2P / (\pi w_0^2) \quad (2.22)$$

This can be combined to give a direct relationship between power and normalized vector potential.

$$P \simeq 21.5 [GW] (a_0 w_0 / \lambda_L)^2 \quad (2.23)$$

Substituting this into Eq. (2.20) and assuming vacuum diffraction gives an estimate for the maximum diffraction limited energy gain of

$$\Delta W_{max, diff} \cong \int_{-z_R}^{z_R} e E_{max}(z) dz \cong e E_{max}(0) 2z_R \cong 1.4 \cdot 10^3 mc^2 \frac{\omega_p}{\omega_L} \frac{P}{[TW]} \quad (2.24)$$

Next we review plasma wakefield excitation. An expression for the wakefield amplitude of a symmetric Gaussian electron bunch can be obtained from 2D linear theory[45].

$$e E_{wake} = \sqrt{\frac{n_p}{cm^{-3}}} \frac{eV}{cm} \times \frac{n_b}{n_p} \times \frac{k_p \sigma_z e^{-k_z^2 \sigma_p^2 / 2}}{1 + \frac{1}{k_p^2 \sigma_r^2}} \quad (2.25)$$

Here n_p is the plasma density, n_b is the beam density, k_p the plasma wave vector, σ_z the width of the Gaussian in propagation direction, and σ_r the width of the Gaussian perpendicular to the propagation direction.

In the blowout regime of a PWFA, most of the electron driver bunch will propagate through the positively charged ion column created by the blowout at the head of the beam[46]. For highly relativistic beams, i.e., those for which $\frac{1}{\gamma}, \frac{d\gamma}{dt} \ll 1$, Eq. (2.11) can be reduced to

$$\frac{d^2}{dx_1^2} \sigma + k_B^2 \sigma = 0 \quad (2.26)$$

where k_B^2 , is now due to the ion column. We can find the correct k_B from Gauss' law applied to a uniformly charged ion column using a cylindrical surface around the

axis. The radial electric field due to the ions is determined by

$$2\pi r E_r = 4\pi e n_p \pi r^2 \quad (2.27)$$

For this case the force on the beam electrons is

$$F_r = (-e) E_r \simeq \gamma m \frac{d^2}{dt^2} r = -\frac{4\pi e^2 n_p}{2} r^2 = -m \omega_B^2 r^2 \quad (2.28)$$

with $\omega_B = \omega_p/\sqrt{2}$ and therefore

$$k_B^2 = \frac{1}{c^2} \frac{\omega_B^2}{\gamma} = \frac{4\pi e^2 n_p}{2\gamma mc^2} \quad (2.29)$$

gives the k_B due to an ion column in Eq. (2.26).

Chapter 3

Review of Basic Particle-In-Cell Algorithms

There are many variations of the the Particle-In-Cell or PIC method[47]. This chapter will review the general idea of the PIC method and then look at the specific algorithms implemented in the simulations codes used for this dissertation, PEGASUS [48] and OSIRIS. Wherever possible a short and concise review of the actual equations implemented is given; otherwise a short description without the actual equations of the implemented method is given. The 2D Cartesian algorithms described in this chapter are common to PEGASUS and OSIRIS with the exception of one of the current deposition schemes. The 3D Cartesian and 2D cylindrically-symmetric algorithms are only part of OSIRIS.

3.1 The PIC-Method

The basic equations governing the behavior of a plasma are well known. Each particle moves according to the Lorentz force exerted on it by the electromagnetic field at its

position.

$$\vec{F} = q \left(\vec{E} + \frac{\vec{v}}{c} \times \vec{B} \right) \quad (3.1)$$

The field in turn evolves according to Maxwell's Equations with the sources given by the particles of the plasma.

$$\vec{\nabla} \cdot \vec{E} = 4\pi\rho \quad (3.2)$$

$$\vec{\nabla} \times \vec{B} = \frac{1}{c} \frac{\partial \vec{E}}{\partial t} + \frac{4\pi}{c} \vec{j} \quad (3.3)$$

$$-\vec{\nabla} \times \vec{E} = \frac{1}{c} \frac{\partial \vec{B}}{\partial t} \quad (3.4)$$

$$\vec{\nabla} \cdot \vec{B} = 0 \quad (3.5)$$

$$\vec{j}(\vec{x}) = \sum_{i=1}^n q_i v_i \delta(\vec{x} - \vec{x}_i) \quad (3.6)$$

$$\rho(\vec{x}) = \sum_{i=1}^n q_i \delta(\vec{x} - \vec{x}_i) \quad (3.7)$$

Together these equations perfectly describe a plasma and in principle completely predict its behavior within the limits of classical physics; but actually solving these equations for a large collection of particles is computationally challenging.

One way to solve these equations is the PIC method [47, 49]. It breaks up the problem into four distinct steps. Fig. 3.1 shows these steps. Given an initial con-

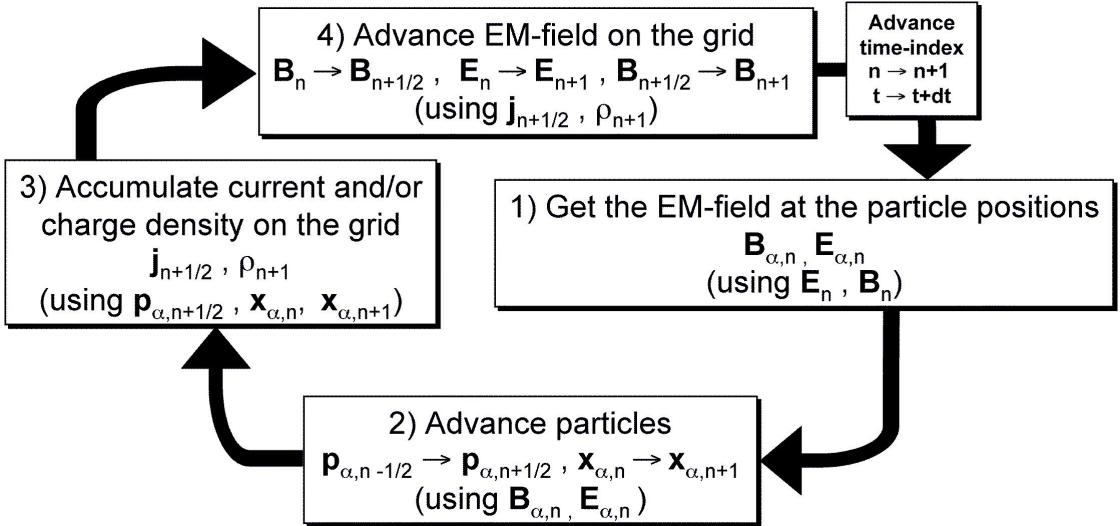


Figure 3.1: The basic loop for PIC simulations. Time is increased in steps of Δt so that $t = t_o + n \times \Delta t$

figuration of particles with certain positions and momenta, and electromagnetic field values known on a staggered grid that is defined throughout the simulation space, a PIC-code first calculates the fields at the particle positions by interpolating the fields on the grid to the particle positions. The dimensions of the grid cells are chosen to resolve the minimum wavelength of interest for the simulated problem. The code then uses these fields and the particle information to calculate the new positions and new momenta of the particles after a suitably chosen timestep, dt . The updated position and momentum data are then used to find the sources of the electromagnetic field, i.e., the current and the charge density are deposited onto the grid. In the final step of the loop, the sources are used to advance the electromagnetic fields in time by a timestep, dt , via Maxwell's equations.

In the following sections of this chapter we will review some of the details for the numerical algorithms of this loop for the 2D cartesian, 3D cartesian, and 2D

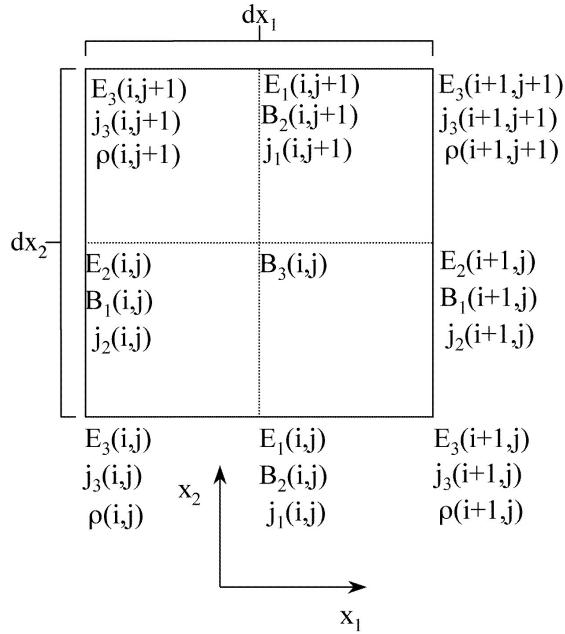


Figure 3.2: The grid for a 2D PIC simulation. The staggered spacing of the E , B , and j components, and of ρ allows for a higher precision of the calculations

cylindrically-symmetric simulations that are possible with OSIRIS.

3.2 PIC Algorithms for 2D-Cartesian Simulations

Fig. 3.2 shows where the different quantities are located on staggered grids in 2D simulations. The staggered grids are used since they increase the numerical accuracy [47]. The field solve in OSIRIS works in three different steps to advance the fields at a grid point with the indices i_1 and i_2 by a timestep, dt , from a time index n to a time index $n + 1$. It starts with the \vec{E} and \vec{B} fields fields at time n and the current density \vec{j} at the centered time $n + \frac{1}{2}$.

The first step is to advance \vec{B}^n by only half a timestep, $dt/2$, to $\vec{B}^{n+\frac{1}{2}}$, using \vec{E}^n , through Faraday's law:

$$\begin{aligned}
B_{1,i1,i2}^{n+\frac{1}{2}} &= B_{1,i1,i2}^n - c \frac{dt}{2} \times \frac{E_{3,i1,i2+1}^n - E_{3,i1,i2}^n}{dx_2} \\
B_{2,i1,i2}^{n+\frac{1}{2}} &= B_{2,i1,i2}^n + c \frac{dt}{2} \times \frac{E_{3,i1+1,i2}^n - E_{3,i1,i2}^n}{dx_1} \\
B_{3,i1,i2}^{n+\frac{1}{2}} &= B_{3,i1,i2}^n - c \frac{dt}{2} \times \frac{E_{2,i1+1,i2}^n - E_{2,i1,i2}^n}{dx_1} \\
&\quad + c \frac{dt}{2} \times \frac{E_{1,i1,i2+1}^n - E_{1,i1,i2}^n}{dx_2}
\end{aligned} \tag{3.8}$$

The next step is to advance \vec{E}^n by a full timestep, dt , to \vec{E}^{n+1} , using $\vec{B}^{n+\frac{1}{2}}$ and $\vec{j}^{n+\frac{1}{2}}$, through Ampere's law:

$$\begin{aligned}
E_{1,i1,i2}^{n+1} &= E_{1,i1,i2}^n - 4\pi dt \times j_{1,i1,i2}^{n+\frac{1}{2}} \\
&\quad + c dt \times \frac{B_{3,i1,i2}^{n+\frac{1}{2}} - B_{3,i1,i2-1}^{n+\frac{1}{2}}}{dx_2} \\
E_{2,i1,i2}^{n+1} &= E_{2,i1,i2}^n - 4\pi dt \times j_{2,i1,i2}^{n+\frac{1}{2}} \\
&\quad - c dt \times \frac{B_{3,i1,i2}^{n+\frac{1}{2}} - B_{3,i1-1,i2}^{n+\frac{1}{2}}}{dx_1} \\
E_{3,i1,i2}^{n+1} &= E_{3,i1,i2}^n - 4\pi dt \times j_{1,i1,i2}^{n+\frac{1}{2}} \\
&\quad + c dt \times \frac{B_{2,i1,i2}^{n+\frac{1}{2}} - B_{2,i1-1,i2}^{n+\frac{1}{2}}}{dx_1} \\
&\quad - c dt \times \frac{B_{1,i1,i2}^{n+\frac{1}{2}} - B_{1,i1,i2-1}^{n+\frac{1}{2}}}{dx_2}
\end{aligned} \tag{3.9}$$

The final step is to again advance \vec{B} by another half a timestep, $dt/2$, from $\vec{B}^{n+\frac{1}{2}}$ to \vec{B}^{n+1} , using \vec{E}^{n+1} , through Faraday's law:

$$\begin{aligned}
B_{1,i1,i2}^{n+1} &= B_{1,i1,i2}^{n+\frac{1}{2}} - c \frac{dt}{2} \times \frac{E_{3,i1,i2+1}^{n+1} - E_{3,i1,i2}^{n+1}}{dx_2} \\
B_{2,i1,i2}^{n+1} &= B_{2,i1,i2}^{n+\frac{1}{2}} + c \frac{dt}{2} \times \frac{E_{3,i1+1,i2}^{n+1} - E_{3,i1,i2}^{n+1}}{dx_1} \\
B_{3,i1,i2}^{n+1} &= B_{3,i1,i2}^{n+\frac{1}{2}} - c \frac{dt}{2} \times \frac{E_{2,i1+1,i2}^{n+1} - E_{2,i1,i2}^{n+1}}{dx_1} \\
&\quad + c \frac{dt}{2} \times \frac{E_{1,i1,i2+1}^{n+1} - E_{1,i1,i2}^{n+1}}{dx_2}
\end{aligned} \tag{3.10}$$

These equations above can be derived in a straight forward manner from the differenced form of Maxwell's equation if the translational invariance in x_3 is used to remove the dx_3^{-1} terms in the full equations. The first and the second part of advancing \vec{B} have the same form and only the arguments differ. This is therefore implemented in the codes by calling the same subroutine once with \vec{B}^n and \vec{E}^n as arguments and then again later with $\vec{B}^{n+\frac{1}{2}}$ and \vec{E}^{n+1} . The benefit of splitting up the advancement of \vec{B} is that \vec{E} and \vec{B} after the advancement are both known at the same time index n . This keeps the particle push and the field solve time centered. An equivalent implementation would be to set $\vec{B}^{n+\frac{1}{2}} = \frac{1}{2} (\vec{B}^{n+1} + \vec{B}^n)$, but the above implementation requires less memory since \vec{B}^n and \vec{B}^{n+1} are not needed simultaneously.

The fields are interpolated to a particle position by weighting each field component linearly from the particle's nearest four grid points for which each component of \vec{E} or \vec{B} is known. For example, if a particle is in the grid cell $i1$ in x_1 and $i2$ in x_2 at a position $(\varepsilon_1 dx_1, \varepsilon_2 dx_2)$ within that grid cell then the field component E_3 at the particle position is given by:

$$E_3 = (1 - \varepsilon_1) ((1 - \varepsilon_2) E_{3,i1,i2} + \varepsilon_2 E_{3,i1,i2+1}) \\ + \varepsilon_1 ((1 - \varepsilon_2) E_{3,i1+1,i2} + \varepsilon_2 E_{3,i1+1,i2+1}) \quad (3.11)$$

The fields at the particle's position are then used to update the momentum of the particle. This happens in several steps in order to increase the accuracy of the momentum push. A derivation of this method can be found in the literature [47]. This so called Boris push can be summarized as,

$$\begin{aligned} \vec{p}' &= \vec{p}^{n-\frac{1}{2}} + q \frac{dt}{2} \vec{E}^n \\ \vec{p}'' &= \vec{p}' + q \frac{dt}{2} \vec{p}' \times \vec{B}^n \frac{1}{\sqrt{1 + \vec{p}'^2}} \\ \vec{p}''' &= \vec{p}' + q dt \vec{p}'' \times \vec{B}^n \frac{1}{\sqrt{1 + \vec{p}''^2}} \frac{1}{1 + (\vec{B}^n)^2} \\ \vec{p}^{n+\frac{1}{2}} &= \vec{p}''' + q \frac{dt}{2} \vec{E}^n \end{aligned} \quad (3.12)$$

where $\vec{p}^{n-\frac{1}{2}}$ and $\vec{p}^{n+\frac{1}{2}}$ are the momenta of the particle before and after the push. The updated momentum, $\vec{p}^{n+\frac{1}{2}}$, is then used to update the particle position according to:

$$\vec{x}^{n+1} = \vec{x}^n + q dt \frac{\vec{p}^{n+\frac{1}{2}}}{\sqrt{1 + (\vec{p}^{n+\frac{1}{2}})^2}} \quad (3.13)$$

where only the components in the simulation plane are updated.

For 2D simulations OSIRIS provides two different current deposition algorithms[50, 51, 52] both of which use the old position \vec{x}^n and the new position \vec{x}^{n+1} of each par-

ticle to calculate the current on the grid. Both of these current deposition schemes have in common that they rigorously obey the continuity equation:

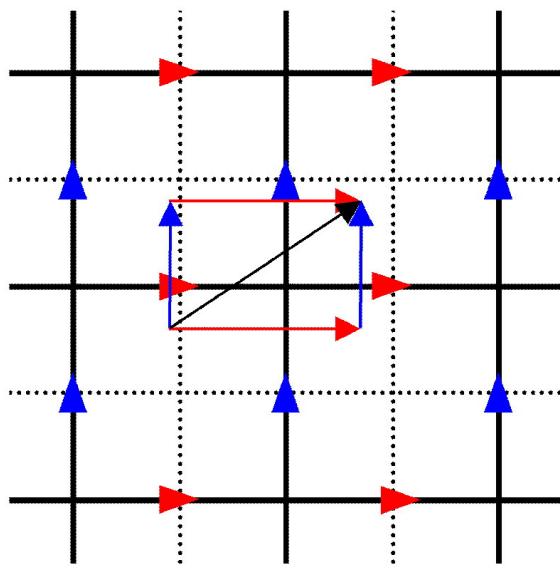
$$\vec{\nabla} \cdot \vec{j} + \frac{\partial \rho}{\partial t} = 0 \quad (3.14)$$

However, this leaves the possibility of adding an arbitrary curl to the current. Therefore, both of these charge conserving algorithms give the same value for $\vec{\nabla} \cdot \vec{j}$ but sometimes give different values for $\vec{\nabla} \times \vec{j}$.

In both methods a particle is viewed as a finite size particle which contributes a charge density ρ to the nearest grids using a weighting function. In OSIRIS, the weighting functions differ between the methods but this is not the fundamental difference between them. Both methods are based on the idea that the contribution of a particle to the charge density on the grid before and after the push can be used to infer the current that has to be assigned to the grid. Due to these common ideas both methods satisfy Eq. (3.14). If a particle stays within a cell during a timestep then both methods give the same answer. The difference between the methods lies in the assumed paths for a particle when it crosses a cell boundary.

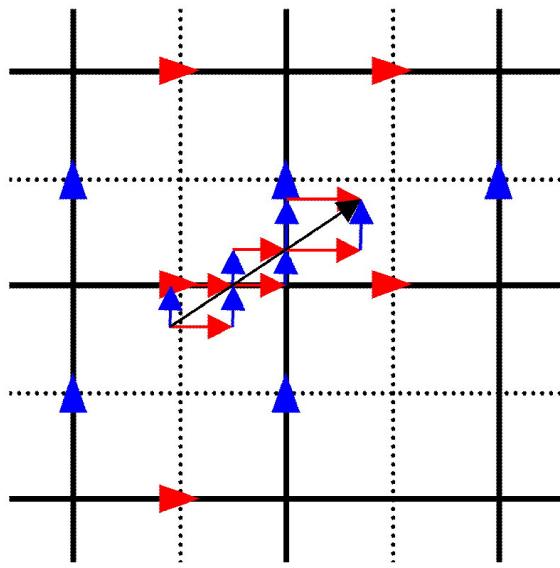
An example for the paths that the first method, which is also implemented in PEGASUS, assumes is illustrated in Fig. 3.3. This scheme was taken directly from ISIS and it is similar in spirit to appendix A of Ref.[50]. Fig. 3.4 on the other hand gives an example for the paths that the virtual particle method[51, 52] assumes for the same actual particle motion as shown in Fig. 3.3. This method is identical to that of Villasenor and Buneman and it is the one used in the well known TRISTAN code[51, 53].

The differences between the two methods, which we will refer to as ISIS or TRIS-



- ▶ **j_1 grid positions with deposition**
- ▲ **j_2 grid position with deposition**

Figure 3.3: The figure shows an example of the paths over which the ISIS method averages using the particle position before and after the particle push. It also shows where current is deposited on the grid for the case of this example.



- ▶ **j_1 grid positions with deposition**
- ▲ **j_2 grid position with deposition**

Figure 3.4: The figure shows an example of the paths over which the TRISTAN method averages using the particle position before and after the particle push. It also shows where current is deposited on the grid for the case of this example.

TAN methods, can be understood by noting that in two dimensions any straight line trajectory can be decomposed into two orthogonal moves. If we define $\Delta x_1 \equiv x_1^{n+1} - x_1^n$ and $\Delta x_2 \equiv x_2^{n+1} - x_2^n$ as the changes in each 2D coordinate during a push then the trajectory can be viewed as $\hat{x}_1 \Delta x_1 + \hat{x}_2 \Delta x_2$. However, if this motion is used to determine the current, there is an ambiguity between letting the particle first move in \hat{x}_1 and then in \hat{x}_2 or vice versa. The difference in the paths is an overall current loop, so there is an ambiguity in the curl. In the ISIS algorithm the current deposited is that from the average of the two moves. In the TRISTAN (or virtual particle) method the same procedure is used as long as the particle stays within a cell. However, when it doesn't, as in the case shown in Fig. 3.3 and Fig. 3.4, the complete straight line trajectory is broken up into two straight line paths which connect at the cell face boundaries, and each separate straight line is then broken up as the average of the two types of orthogonal moves as described earlier. The TRISTAN method is more accurate since it approximates the path of a particle more closely, but it is also computationally more expensive.

3.3 3D-Cartesian and 2D-Cylindrically-Symmetric Algorithms

The field solve for 3D Cartesian simulations is a simple extension to the one used for 2D Cartesian simulations. In Fig. 3.5 we show how the different quantities are staggered on a 3D grid. The main difference between the 2D and 3D setup is that there are additional terms in the calculation of E_1 , E_2 , B_1 , and B_2 , because derivatives with respect to x_3 are not assumed to be zero. Some other straightforward changes arise from the additional staggering of some grid quantities in x_3 . This can also be

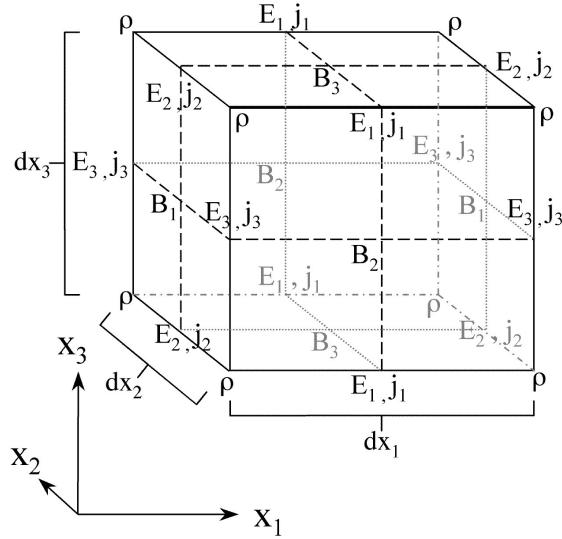


Figure 3.5: The grid for a 3D PIC simulation. The staggered placing of the E , B , and j components, and of ρ allows for a higher precision of the calculations

seen in Fig. 3.5.

On the other hand the 2D cylindrically symmetric field solve algorithm is not a straightforward extension of the 2D Cartesian case. In Fig. 3.6 the grid for the 2D cylindrically symmetric case is shown. This grid can be derived from the 2D Cartesian grid by making the substitutions:

$$\begin{array}{llllllll}
 dx_1 & \rightarrow & dz, & dx_2 & \rightarrow & dr \\
 B_1 & \rightarrow & B_z, & B_2 & \rightarrow & B_r, & B_3 & \rightarrow & B_\Theta \\
 E_1 & \rightarrow & E_z, & E_2 & \rightarrow & E_r, & E_3 & \rightarrow & E_\Theta \\
 j_1 & \rightarrow & j_z, & j_2 & \rightarrow & j_r, & j_3 & \rightarrow & j_\Theta
 \end{array}$$

In the field solve for 2D cylindrically-symmetric simulations the way E_z and B_z are calculated differs significantly from the way E_1 and B_1 are calculated for 2D Cartesian simulations. For the other components of the fields it is possible to derive the

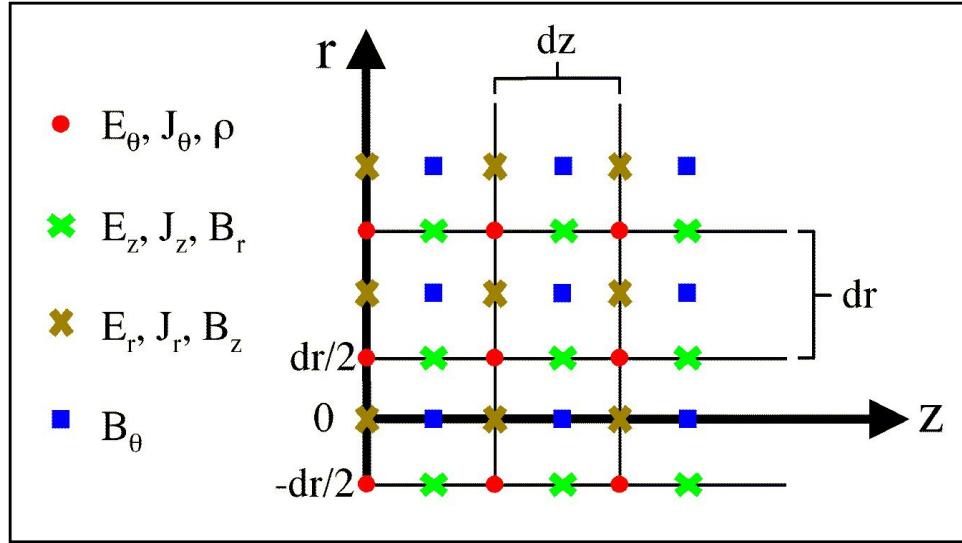


Figure 3.6: The grid for a 2D cylindrically-symmetric PIC simulation. The axis has been placed through the middle of the first grid cell in order to avoid having to calculate j_z on for $r = 0$.

correct equations for the 2D cylindrically-symmetric case by applying the substitutions above. The equations for z -components have an explicit dependency on r in cylindrical coordinates and the modified equations for the x_1 -components of Eq. (3.8), Eq. (3.9), and Eq. (3.10) are:

$$\begin{aligned}
 B_{z,i1,i2}^{n+\frac{1}{2}} &= B_{z,i1,i2}^n - c \frac{dt}{2} \times \frac{1}{r_{i2+\frac{1}{2}}} \frac{r_{i2+1}E_{\Theta,i1,i2+1}^n - r_{i2}E_{\Theta,i1,i2}^n}{dr} \\
 E_{z,i1,i2}^{n+1} &= E_{z,i1,i2}^n - 4\pi dt \times j_{z,i1,i2}^{n+\frac{1}{2}} \\
 &\quad + c dt \times \frac{1}{r_{i2}} \frac{r_{i2+\frac{1}{2}}B_{\Theta,i1,i2}^{n+\frac{1}{2}} - r_{i2-\frac{1}{2}}B_{\Theta,i1-1,i2}^{n+\frac{1}{2}}}{dr} \\
 B_{z,i1,i2}^{n+1} &= B_{z,i1,i2}^{n+\frac{1}{2}} - c \frac{dt}{2} \times \frac{1}{r_{i2+\frac{1}{2}}} \frac{r_{i2+1}E_{\Theta,i1,i2+1}^{n+1} - r_{i2}E_{\Theta,i1,i2}^{n+1}}{dr}
 \end{aligned} \tag{3.15}$$

Here the following conventions are used: $i1$ is the grid index for z and $i2$ is the grid

index for r . The grid cells on axis in Fig. 3.6 have the index $i2 = 1$ and $r_{i2} = dr(i2 - \frac{3}{2})$.

The calculation of the field right on the z -axis is also not straightforward. In OSIRIS the $r = 0$ axis is not the lower boundary of the simulations. The simulation space extends to $r = -dr/2$ as shown in Fig. 3.6. This is done in order to avoid having to calculate E_z on axis. It is easier to calculate B_z since it does not require interpolating the current component j_z to the $r = 0$ axis. It is interesting to note that we initially used the $r = 0$ as the lower boundary. This led to substantial short-wavelength ($\lambda \sim dz$) noise near the $r = 0$ axis. Based on work by Seung Lee the present algorithm for the cylindrically-symmetric field solve was implemented into OSIRIS.

For the chosen staggered grid the axial boundary conditions also differ for the vector field components parallel and perpendicular to the axis. In particular for the perpendicular components we have:

if $f(z, r) = B_r, B_\Theta, E_r, E_\Theta, j_r, j_\Theta$ then $f(z, r) = -f(z, -r) \rightarrow f(z, 0) = 0$

While for the parallel components (and scalars) we have:

if $f(z, r) = B_z, E_z, j_z, \rho$ then $f(z, r) = f(z, -r) \rightarrow$ no restrictions for $f(z, 0)$

Here $f(z, r)$ stands for a generic field component used in the simulation. With these boundary conditions and the standard difference equations for the fields, we can find all the required quantities in the grid cell on axis except B_z . We use the integral form of Faraday's law to find B_z on axis by integrating $\oint \vec{E} \cdot d\vec{l}$ around a loop half a grid cell off axis. The change of B_z on axis is then given by:

$$B_{z,i1,1}^{n+\frac{1}{2}} = B_{z,i1,1}^n - 4c \frac{dt}{2} \times \frac{E_{\Theta,i1,2}^n}{dr} \quad (3.16)$$

Note that again this advance of B_z has to be applied a second time - after \vec{E} has been

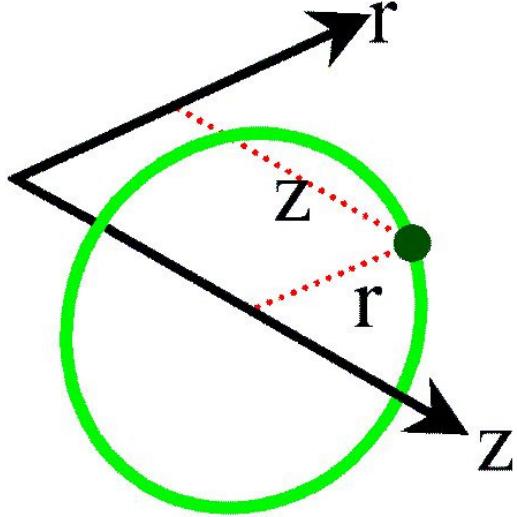


Figure 3.7: The charge represented by a simulation particle in the r - z -plane is a ring

advanced - in order to obtain $B_{z,i1,1}^{n+1}$.

The weighting of the \vec{E} and \vec{B} -fields to the particle positions works the same in 2D cylindrically-symmetric simulation as in 2D Cartesian and the method is straightforward to extend to a 3D algorithm. Extending the push is even simpler because the 2D and 3D Cartesian, and 2D cylindrically symmetric simulations in OSIRIS all use the same subroutine for the momentum update. The algorithm was described above, but it is worth commenting on the 2D cylindrically-symmetric algorithm. The reason why the same algorithm can still be used in the 2D cylindrically-symmetric coordinates is that the 2D cylindrically-symmetric algorithms keep track of $\vec{p} = (p_z, p_r, p_\theta)$. This is the momentum vector represented in the local Cartesian coordinate system at any given point $[\vec{e}_z, \vec{e}_r, \vec{e}_\Theta]$ and therefore it can be updated by a momentum update in Cartesian coordinates.

The position update for 3D Cartesian Coordinates is a straightforward extension from the 2D Cartesian algorithm and does not need to be described any further. However, the position update for 2D cylindrically symmetric is more complicated and requires an understanding of what is actually represented by a simulation particle in a 2D cylindrically-symmetric simulation. As shown in Fig. 3.7, a simulation particle in this type of simulation represents a ring of charge. Combining this with the fact that the momentum for the particle is known in the local Cartesian coordinates of the particle, leads to a method of position update. The particle position is first updated in a 3D Cartesian space in a way that is identical to the one in 3D Cartesian simulations. Afterwards all \vec{x} and \vec{p} quantities are transformed into a rotated frame of reference that eliminates the change in position in the third dimension [47]. This is illustrated in Fig. 3.8 which shows the Cartesian position update in the x - y -plane with successive rotation from r to r' . The x and y coordinates in this figure are used to describe the plane transverse to the z axis. For the momentum, the substitutions $p_r \rightarrow p_x$, and $p_\Theta \rightarrow p_y$ are used. It should be noted that there is an additional change of the momentum vector caused by the rotation of the coordinate system. The next paragraph will give a more detailed explanation of the 2D cylindrically-symmetric position update.

Once the new momenta are calculated each particle is pushed to its new positions as follows. First, the a “pseudo” 3D push is done to get the new temporary position vector in a 3D Cartesian representation,

$$\vec{x}_{3D,new}^{n+1} = (\vec{x}_{2D}^n, 0) + q \cdot dt \frac{\vec{p}^{n+\frac{1}{2}}}{\sqrt{1 + (\vec{p}^{n+\frac{1}{2}})^2}} \quad (3.17)$$

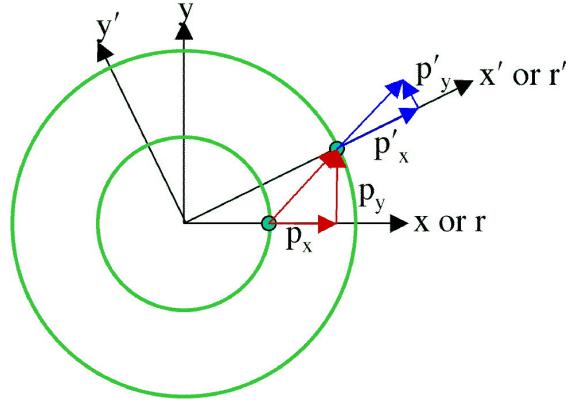


Figure 3.8: The position update for a particle in 2D cylindrically-symmetric coordinates. The use of a “pseudo” 3D push followed by a rotation also requires an update of the momentum.

Secondly, the new position vector for the particle in 2D cylindrical coordinates is obtained in the rotated coordinate system:

$$x_{1,2D}^{n+1} = x_{1,3D,new} \quad (3.18)$$

$$x_{2,2D}^{n+1} = \sqrt{x_{2,3D,new}^2 + x_{3,3D,new}^2} = r^{n+1}$$

Finally the momentum vector in the new rotated coordinate system is calculated:
(using $r^n = x_2^n$)

$$\begin{aligned} p_{2,new}^{n+\frac{1}{2}} &= \left(p_2^{n+\frac{1}{2}} x_{2,3D,new} + p_3^{n+\frac{1}{2}} x_{3,3D,new} \right) / r^{n+1} \\ p_{3,new}^{n+\frac{1}{2}} &= p_3^{n+\frac{1}{2}} r^n / r^{n+1} \end{aligned} \quad (3.19)$$

There is a subtle detail about the calculation of $p_{3,new}^{n+\frac{1}{2}}$ that is easy to miss. The

transformation of the components of \vec{p} suggested in Fig. 3.8 is correct for a point particle. However, the code assumes “ring particles”, so for a uniform ring any contribution that the component p_2 in the old coordinate system will make to the new $p_{3,new}$ after the rotation has to cancel out for reasons of symmetry. The equation for $p_{3,new}^{n+\frac{1}{2}}$ above takes this fact into account, because it is derived from the conservation of angular momentum instead of using the straight forward transformation of the momentum of a point particle.

The current deposition scheme used for 3D simulations in OSIRIS is a simple extension of the 2D TRISTAN deposition method to three dimensions and it is well described in the literature[51, 52]. The deposition algorithms and subroutines used for the 2D cylindrically-symmetric simulations are the same as for the 2D cartesian simulations. This is true for ISIS method as well as for the TRISTAN method and it is possible because the quantity which is deposited is not the current density but the current due to the particles. The difference between the current deposition schemes in the 2D Cartesian and 2D cylindrically-symmetric operating modes of OSIRIS is simply in calculating the current density, \vec{j} from the current, \vec{I} . In the cylindrical geometry the volume of a grid cell depends on its radial position. For a grid cell with a distance r from the axis we get $j_r = I_r/V(r)$, $j_\Phi = I_\Phi/V(r)$, and $j_z = I_z/V(r)$ with $V(r) = dr r d\Phi dz$.

Chapter 4

The Implementation of the Object-Oriented Code OSIRIS

In this chapter, we describe the strategy used to develop OSIRIS and the important features in object oriented structure of code.

4.1 Development Strategy and Code Design

The code OSIRIS is written in the programing language Fortran90[54] and is implemented using an object-oriented style of problem solving [55]. It is the first fully object oriented, multi-dimensional, electromagnetic PIC code written in Fortran90 that is also being used to undertake large scale production runs. As a result, it has been used to gain valuable new insights into physics problems. Several of these will be discussed in the subsequent chapters.

The central goal of developing OSIRIS was to get a code that would support multiple algorithms for multi-dimensional PIC simulations in a distributed computing environment by using multi-dimensional domain decomposition. An additional, es-

sential requirement for OSIRIS was the implementation of the dynamic simulation space concept, which is explained below. In order to achieve these goals the development of OSIRIS followed a step by step process that allowed each step to be verified by comparing the code's results to previous results. This was done as follows:

1. An Object oriented single-node 2D PIC code based on the numerical algorithms of ISIS/Pegasus [48] was developed.
2. The dynamic simulation space algorithm was implemented.
3. Parallelization was implemented.
4. OSIRIS was ported to several different architectures.
5. A 2D cylindrically-symmetric algorithm was implemented.
6. A fully 3D algorithm was implemented
7. New algorithms and physics packages are continually incorporated.

Even though the development of OSIRIS had these distinct stages there were a number of general principles that were used in designing the objects and algorithms of the code throughout the development. These principles were motivated by the eventual goal of the code development which was explained above. The general principles we used were:

- All real physical quantities should have a corresponding object in the code and distinct physical processes should have a corresponding application of a method in the main loop of the code. Following this principle makes the physics being modeled in the code clear and therefore easier to modify and extend.

- There should be, as far as possible, a distinction between the physical objects of the code, e.g., the particle object, and the numerical objects of the code, e.g., the grid object. Physical objects encapsulate information about physical aspects of the simulation. Numerical objects encapsulate information about the numerical algorithms being used. This isolation of numerical and physical aspects allows one to change one of them without having to change the other one.
- All information required frequently throughout the simulation should be declared as a variable in the main program. This gives clarity about which information is available at any point in time, which means that unintended changes of variable values are less likely to happen. This increases the safety and reliability of the code.
- The input file of the code should define as far as possible only the global physical problem to be simulated. Node specific information should be avoided as far as possible. In this way the user of the code can focus on defining the physical problem and does not have to be concerned with parallelization issues.
- As far as possible all classes and objects should refer to a single node and should not be affected by parallelization issues. This is realized by treating all communication of physical objects between nodes as boundary conditions for the physical object on each node. This strategy simplifies incorporation of new algorithms into the code.
- The code should be written in such a way that is largely independent from the dimensionality or the coordinate system used in order to allow for polymorphism[26].

This way much of the code can be reused when incorporating a new algorithm with a different dimensionality or with a different coordinate system.

- Objects and methods should be designed to allow for easy incorporation of old but fast Fortran77 style algorithms as subroutines. This should make it easy to incorporate algorithms from one of the many Fortran77 legacy codes.
- All system dependent parts of the code should be encapsulated in as few modules as possible. This ensures easy portability of the code to other systems.

In addition to these general principles there are a couple of conventions used in OSIRIS. These conventions are listed here for the benefit of people who are interested in understanding or modifying OSIRIS for future research. With few exceptions these conventions are applied throughout most of the code.

- Subroutines/Methods modifying specific data are in the same module/class as the type-/object-definition of those data. This is a general principle of object-oriented programming. It is not maintained in this code for certain utility modules which do not contain any type/object definition but supporting subroutines that are used by more than one other module/class. Another exception is the VDF-class described below. It provides direct access to the arrays of different dimensionality it contains in order to allow for polymorphism in the code.
- With the exceptions of dummy arguments with the pointer attribute all dummy arguments in subroutines are declared with an intent. (It is not possible to declare an intent for dummy arguments with the pointer attribute.)
- In the argument list of any subroutine first the arguments with the “intent(out)” attribute, then the arguments with the “intent(inout)” attribute, and finally the

arguments with the “intent(in)” attribute are listed.

- In the declaration part of any subroutine the order in which dummy arguments are declared corresponds to their order in the subroutine call argument list. Exceptions are commented on in the code.
- The names of modules have the structure m_<rest of the name>. The names of types have the structure t_<rest of the name>. If a type is defined in a module then <rest of the name> is the same for the type and the module.
- The names of all compile-time parameters have the form p_<rest of the name>.

4.2 High Level Description

Fig. 4.1 and Fig. 4.2 show the flow chart and the class hierarchy of OSIRIS. Together, these two figures give a high level description of the code. It is worth noting, that in the main loop of OSIRIS the distinct physical and other (diagnostic, restart, etc.) operations correspond to a specific step in the loop. It differs from the loop shown in Fig. 3.1 by the fact that step one to three of Fig. 3.1 are all part of step six of Fig. 4.1. For computational efficiency all these steps are best taken care of in a combined algorithm on this level of the code. The different subroutines that are called for these different steps are called on a lower level of the code.

Fig. 4.2 is using the Object Modeling Technique Notation (OMT)[55] to describe the class hierarchy of OSIRIS. The top level of the class hierarchy shows four different classes, particles, electromagnetic fields, source fields, and the laser pulse sequence. The first three correspond to distinctly different physical quantities. The laser pulse sequence actually does not belong on this level and should in future versions of the

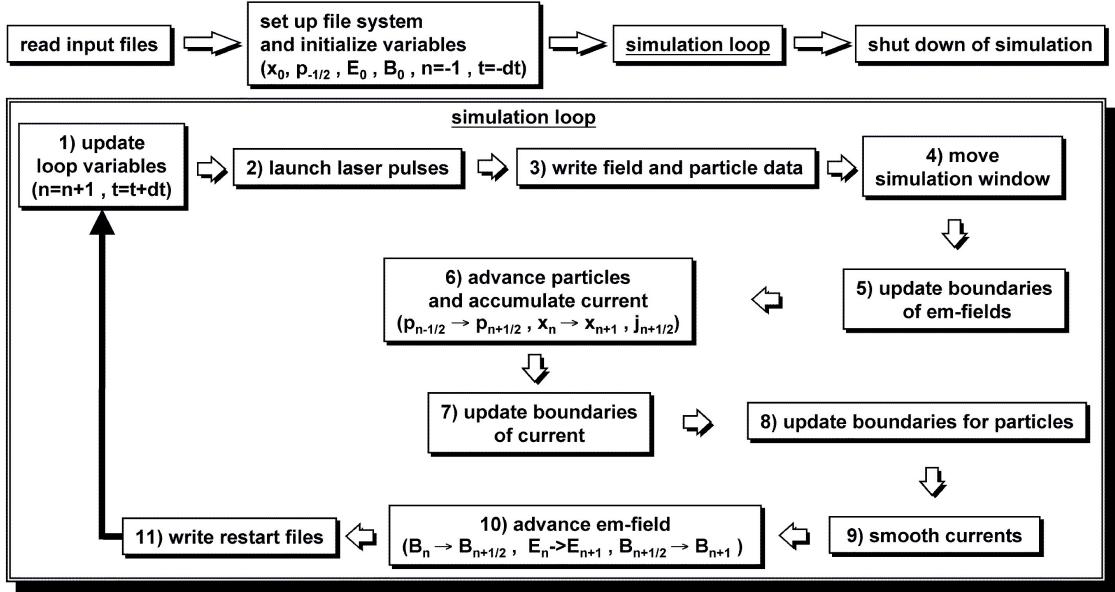


Figure 4.1: The flow control diagram of OSIRIS. It follows the general structure of Fig. 3.1 but shows differences that arise from the specific details of the implementation of OSIRIS.

code become a sub-object of the electromagnetic field.

Each of the physical objects contains a sub-object that describes its boundary conditions and another sub-object that describes the data diagnostic for the object. In the case of the particle object, which is composed of an arbitrary number of species objects, the boundary conditions are actually defined for each single species object separately. A species object contains all information required for one particular particle population, e.g., the actual data for each single particle, the initial density of the species and the temperature.

The electromagnetic field object contains the information for electric and magnetic fields, and the source field objects contain the information for the source terms in Maxwell's equation, the current and the charge density. The distinction between the electromagnetic and source fields is made because particle information is needed to

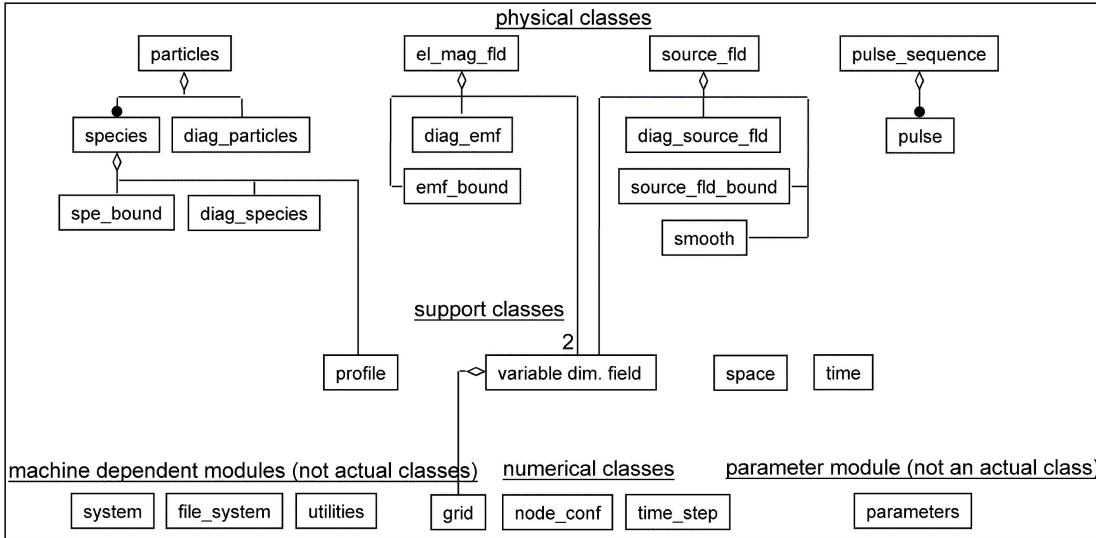


Figure 4.2: The class hierarchy of OSIRIS. The figure shows most of the classes and modules used but omits some for the goal of clarity.

be deposited onto the grid to calculate the source fields. This requires an object with different properties than the electromagnetic field object. In addition to the field data both classes of field objects also contain sub-classes that define the boundary conditions and the diagnostics for the fields.

4.3 Variable Dimension Field Objects

For storage of the actual field information the electromagnetic field objects as well as the source field objects have variable-dimension-field objects (VDF objects) as sub-objects. An object of this type contains the field and grid information for a scalar or vector field in a 1D, 2D, or 3D simulations space. The use of these polymorph objects [26] makes it possible to avoid the explicit use of the dimensionality of a simulation in most of the code. The dimensionality only becomes important when the actual data

```

type :: t_vdf

!      allow access to type components only to module procedures
private

!      variable to indicate status of vdf-object
logical :: associated

!      pointer to field data for field 1D space f(j,i1)
real(p_k_rdoub), dimension(:,:), pointer :: f1

!      pointer to field data for field 2D space f(j,i1,i2)
real(p_k_rdoub), dimension(:,:,:), pointer :: f2

!      pointer to field data for field 3D space f(j,i1,i2,i3)
real(p_k_rdoub), dimension(:,:,:,:), pointer :: f3

!      grid information for this object on the local node
type( t_grid ), pointer :: grid

end type t_vdf

```

Figure 4.3: The definition of the VDF type in OSIRIS

within one of the arrays in a VDF object need to be used. For this case the VDF class provides functions to inquire about the dimensionality and other properties of the VDF-object. It also provides pointer-valued functions that allow direct access to the actual arrays. Fig. 4.3 shows the actual type definition of the vdf-class. Note that the class methods make sure that only one of the pointers is used for a given VDF object. The grid component of the VDF object contains the information about the computational grid that the field is defined on. Polymorph objects like the VDF-objects were suggested by Decyk et al. as a way to allow Fortran90 codes to simulate certain uses of C++ templates [26].

4.4 Global and Local Objects

Data in OSIRIS can be classified into two different categories, global data and local data. Global data are referring to the whole simulation. Local data are referring to the part of the simulation running on one specific node. In OSIRIS most classes refer only to local data. The exceptions are the node-configuration class, the space class, and the grid class. A node-configuration object always contains global as well as local data. The space class and the grid class have global as well as local instances. However, each space or grid object contains either global or local data but not both. The global space and global grid describe the space and the grid of the whole simulation. The local space and the local grids describe the space and the grid on the node that a process is running on. All other classes of OSIRIS contain only local data. This section will describe how the global and local objects of a simulations are initialized at startup and how this relates to the one object per node strategy that is used in OSIRIS.

A simulation starts with initializing all necessary variables of the code with the correct initial values. This happens in OSIRIS by first reading in from the input file for each object separately the information the object requires from that file. The input file contains, with the exception of the node-configuration information, only information about the global physical problem to be simulated. The node configuration object contains the information on how many computing nodes the simulation will run on and on how the whole simulated space is decomposed to the different nodes. Therefore, after reading in the input file information, the code first fully initializes the node-configuration object, then a space object that describes the global simulation space and then a grid object that describes the global simulation grid. A grid object contains

the information about a computational grid that is necessary to define fields in a space. The next step uses the global space object and the node-configuration object to generate a space object that describes the local space. A local grid object is generated in a similar fashion (from the global grid and the node-configuration object). All other objects of the simulation are then initialized as local objects for each specific node using the information from the local space and local grid objects to adapt as much as necessary the information read in from the input file (which is global information) to the local node.

Fig. 4.4 shows the global and the local grid which would be used for a 2D-simulation on 2×2 nodes. The global grid has indices from 1 to $nx_p(1)(global)$ in the x_1 -direction and from 1 to $nx_p(2)(global)$ in the x_2 -direction. The local grid has indices from 1 to $nx_p(1)(local)$ in the x_1 -direction and from 1 to $nx_p(2)(local)$ in the x_2 -direction. For the currently implemented algorithm all local grids have the same size in a given direction if the global grid can be divided up evenly over the number of nodes in this direction. If the global grid can not be divided up evenly over the number of nodes then a certain number of nodes will have one grid cell less than the other nodes.

To make the generation of the local space and grid possible, each node is assigned a certain position in a regular grid of nodes. This node-grid is 3D for 3D-simulation, and of lower dimensionality for 1D- and 2D-simulations. The assignment is done by placing the unique task (or process) IDs for all nodes in a 1D array which has a size given by the total number of nodes used. The task IDs are provided to the program by the message-passing library that is used. The current implementation of OSIRIS uses MPI [56, 57] for message-passing. The position in the array is then used as an ID number (AID) to access the task ID when necessary. If the number of processors

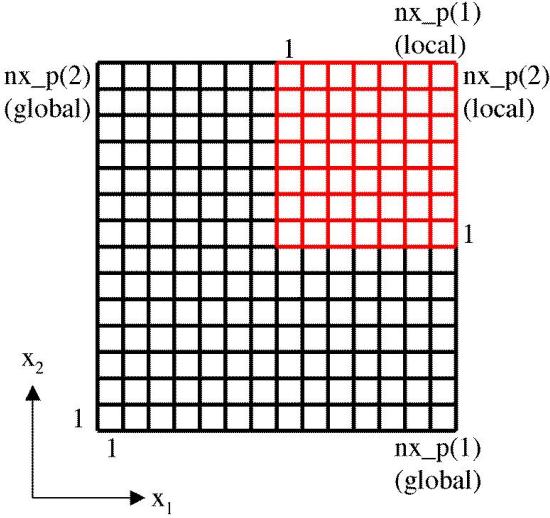


Figure 4.4: The code running on each node has several instances of a grid object. The global grid object describes the the grid of the whole simulation (black) and how it has moved. The local grid objects that are part of each VDF object contain the same information for the domain assigned to a specific node (red).

in each direction is given by $nx(1)$, $nx(2)$, $nx(3)$ then the total number of processors is $nx(1) \times nx(2) \times nx(3)$. The Array ID for the node at the node grid position $n(1)$, $n(2)$, $n(3)$ is then given by:

$$AID = (n(1) - 1) + (n(2) - 1) \times nx(1) + (n(3) - 1) \times nx(1) \times nx(2) + 1 \quad (4.1)$$

This can be inverted to give a unique node-grid position for a given array ID. Fig. 4.5 shows the different type of decompositions that this node-assignment algorithm allows for 3D simulations. The parallel efficiency of these decomposition has been investigated previously by Lyster et al. for an electrostatic PIC code[58].

The separation of global and local data described above together with the node-

1D-Decompositions 2D-Decompositions 3D-Decompositions

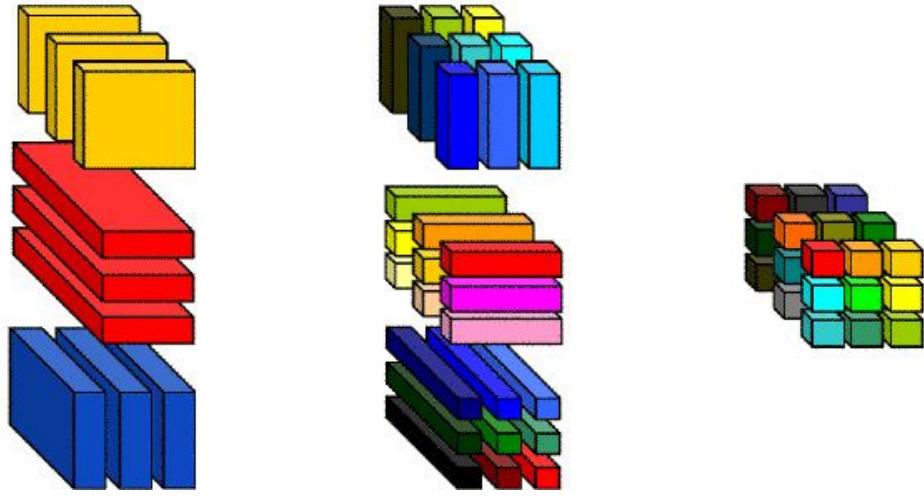


Figure 4.5: The possible decompositions in a 3D simulation.

configuration object make it possible to implement a one-object-per-node strategy, since all information within the physical objects is local and all information required for communication with other nodes is contained in the node-configuration object. When a physical object, which contains only local data, needs to exchange information with a neighboring node at a given boundary it only has to specify the information and the boundary and hand this information to the node-configuration object. This process is implemented as one particular boundary condition. The node-configuration object then manages the details of which nodes send or receive information. The one-object-per-node strategy implemented in OSIRIS is illustrated in Fig. 4.6. Since this strategy makes it much easier to extend OSIRIS with new algorithms it was an important goal of the code development. The design of the space and grid objects and of the setup was done in the way described above because of the one-object-per-node strategy.

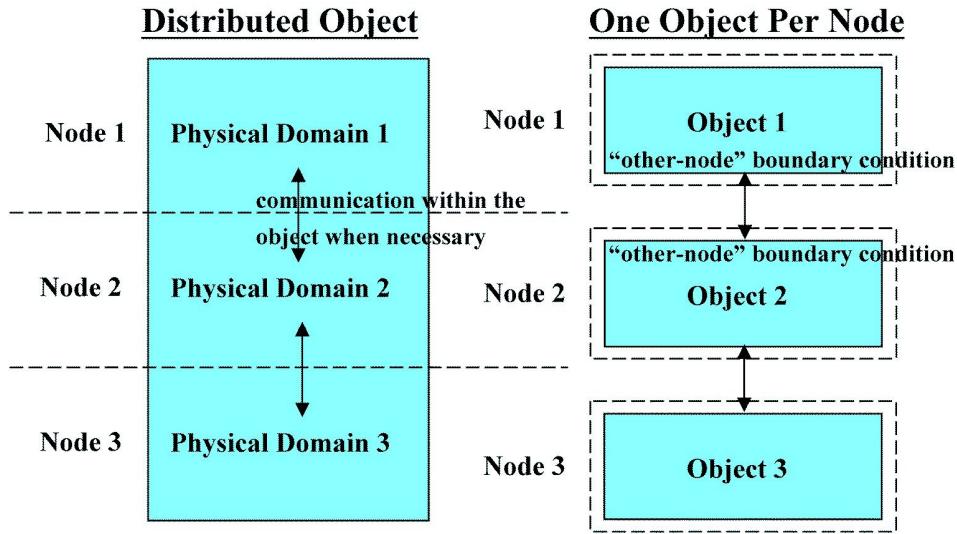


Figure 4.6: Two different concepts for objects on a parallel computer. In a parallel object communication is done by the methods of a class whenever information from another node is needed within that method. The concept of one object-per-node treats boundaries with other nodes as one particular kind of boundary condition.

4.5 Dynamic Simulation Spaces

Once the global and local objects have been defined, they have to be consistently maintained throughout the run. This includes consistency with each other, with the respective global objects, and with the objects on other nodes. The implementation of this is not straightforward since in OSIRIS moving boundaries are possible with a special case being the motion of boundaries required for a moving simulation window[48]. There are two types of moving boundaries in OSIRIS; one that moves outward from the simulation window with the speed of light and one that moves inward with the speed of light. An outward moving boundary requires us to extend the simulation window by one grid cell whenever enough time has passed for the boundary to have moved outward by at least that far. An inward moving boundary

requires us to shorten the simulation window by one grid cell whenever enough time has passed for the boundary to have moved inward by at least that far. If dx is the grid cell size and dt is the timestep size then we know that the Courant condition [47] for electromagnetic PIC codes requires that $c dt < dx$. Therefore a moving boundary can not move by one grid cell at every timestep. It is only moved by dx when the mismatch between the location where the boundary actually is and the location where the boundary should be becomes larger than dx . At timesteps where the boundary is not moved by a grid cell no special boundary algorithms are needed.

All objects, the space objects (global and local), the grid objects (global and local), and all the physical objects, have to be modified according to this change of the space. The fact that a boundary is moving outward with the speed of light makes it possible to simply initialize the plasma in the new space as a thermal plasma with zero electromagnetic field since it is in an area that can not have been affected in any way by the interior of the simulation. The boundary condition for a boundary moving inward with the speed of light can be implemented in a similar way only that simulation space has to be removed and the plasma in that space has to be discarded. Again the fact that the boundary is moving at the speed of light makes this boundary condition simple since none of the discarded space can have any further affect on the remaining simulation space. With these kind of moving boundaries the simulation space becomes a dynamic window that moves in space and can change in size as it follows the physics of interest. A dynamic window is very useful for plasma-based accelerator simulations. In such cases it is useful for the front boundary of the simulation window to move outward with the speed of light and the back boundary to move inward with the speed of light. Fig. 4.7 illustrates the implementation of a moving window for laser-plasma physics using the dynamic space concept. Another

$$\begin{aligned}
 x_{\text{pulse}} &\rightarrow x_{\text{pulse}} + v_{\text{pulse}} \Delta t \\
 x_{\text{min}} &\rightarrow x_{\text{min}} + c \Delta t \\
 x_{\text{max}} &\rightarrow x_{\text{max}} + c \Delta t
 \end{aligned}$$

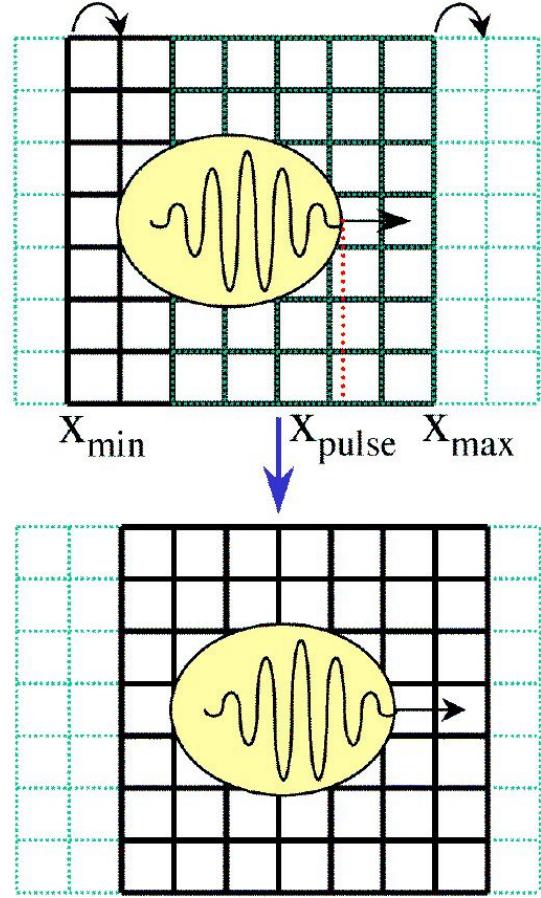


Figure 4.7: For a dynamic space the boundaries can move inward or outward from there current position. A moving window is the special case of the front boundary moving outward and the back boundary moving inward.

application of dynamic space is that of an expanding or collapsing space. This is not fully implemented yet, but planed for a future version of the code.

Fig. 4.8 illustrates the relationship between the global and the local space object for a moving window. The global, as well as the local, space contain the information about the lower and upper boundaries of the simulation window in each direction. In general these boundaries are different for the global and local space on a given node unless the code is running on only one node. In the case of a single-node simulation the

global and the local space describe the same space (the same is true for the global and local grid). If the window of the simulation moves then the boundary information of both spaces has to be updated each time the boundary moves. Actually, the information that describes the motion of the simulation window is stored as part of the global space object. If the global simulation window moves at a certain timestep then the local space object will move accordingly. Note that even though in Fig. 4.8 only the update of the values X_{\min} and X_{\max} of the global space boundary is explicitly shown, this is done for the local space boundaries as well (as the actual picture in Fig. 4.8 also shows). The next section of this chapter will give a more detailed description of the implementation of moving boundaries between nodes.

After the space objects have been moved, the motion of the global grid is determined according to the motion of the global space. All other objects use information from the local space, which has been updated using the global space object, to decide whether and how to move each boundary. This is motivated by the strategy to use local-node information wherever possible.

There is a second data structure in the code that keeps track of the motion of objects and boundaries in space. Each grid object keeps track of the motion of its boundaries as it moves with respect to the initial global grid (note that the global grids on all nodes are identical to each other at all times). In this way the grid objects on each node can keep track of their size and motion in space. The global grid keeps track of its size and motion with regard to its own initial state. This information is in a certain sense redundant with some of the information stored in the space objects but the redundancy is justified by providing easy access to this information without having to translate from the continuous space boundary description to the discrete grid cell description of it every single time it is needed. It therefore simplifies

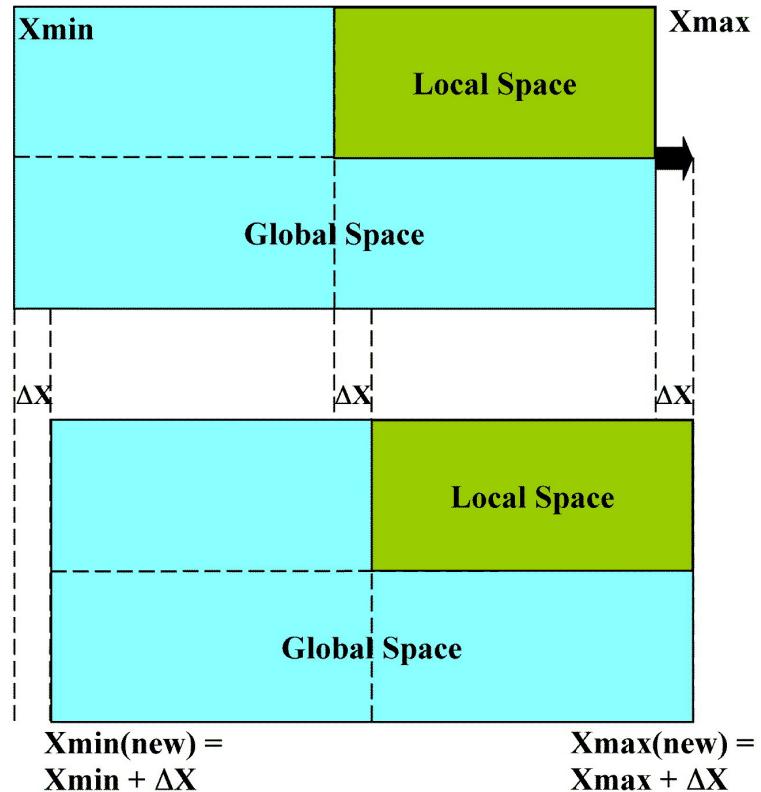


Figure 4.8: The code running on each node has two instances of a space object. The global space object describes the space of the whole simulation and its motion. The local space object contains the same information for the domain assigned to the specific node. The figure shows the update of the space boundaries explicitly only for the global object but for the local space the same updating of X_{min} and X_{max} is done.

a lot of algorithms. Fig. 4.9 illustrates the way the global and local grid objects keep track of the boundary motion for their lower boundaries. They do so by simply adjusting the variable that describes the boundary position with respect to the initial grid each time the boundary is moved. The upper boundary is followed in the same way and together these two numbers also describe the size of the grid. Please note that following the positions of the grid boundaries is completely independent from the grid indices provided by grid objects to the algorithms of the code. The grid indices start for a given grid always with 1 at the lower boundary and go up to the maximum number of grid points in a given direction for this grid.

4.6 The Motion of Internal Boundaries

In the previous section we explained that the information in the local space object is used to steer the boundary movement of all objects except of the global space and the global grid. This decouples the motion of the boundary of a node from the motion of the global boundary. In the case where a node has a boundary that is also a global, external boundary of the whole simulation the result has to be the same, but in the case where the boundary is an internal boundary to another node the signal to move the boundary will have to make use of different algorithms than the ones described above which are just applicable to the external boundaries of the simulation.

The additional complications that need to be addressed when moving an internal boundary between nodes, and that are outlined below, are independent from the reasons why the boundary is moved. Before describing details of how internal boundary motion is handled it is therefore worth noting that although we have implemented moving boundaries as part of the motion of the whole simulation space, this ability

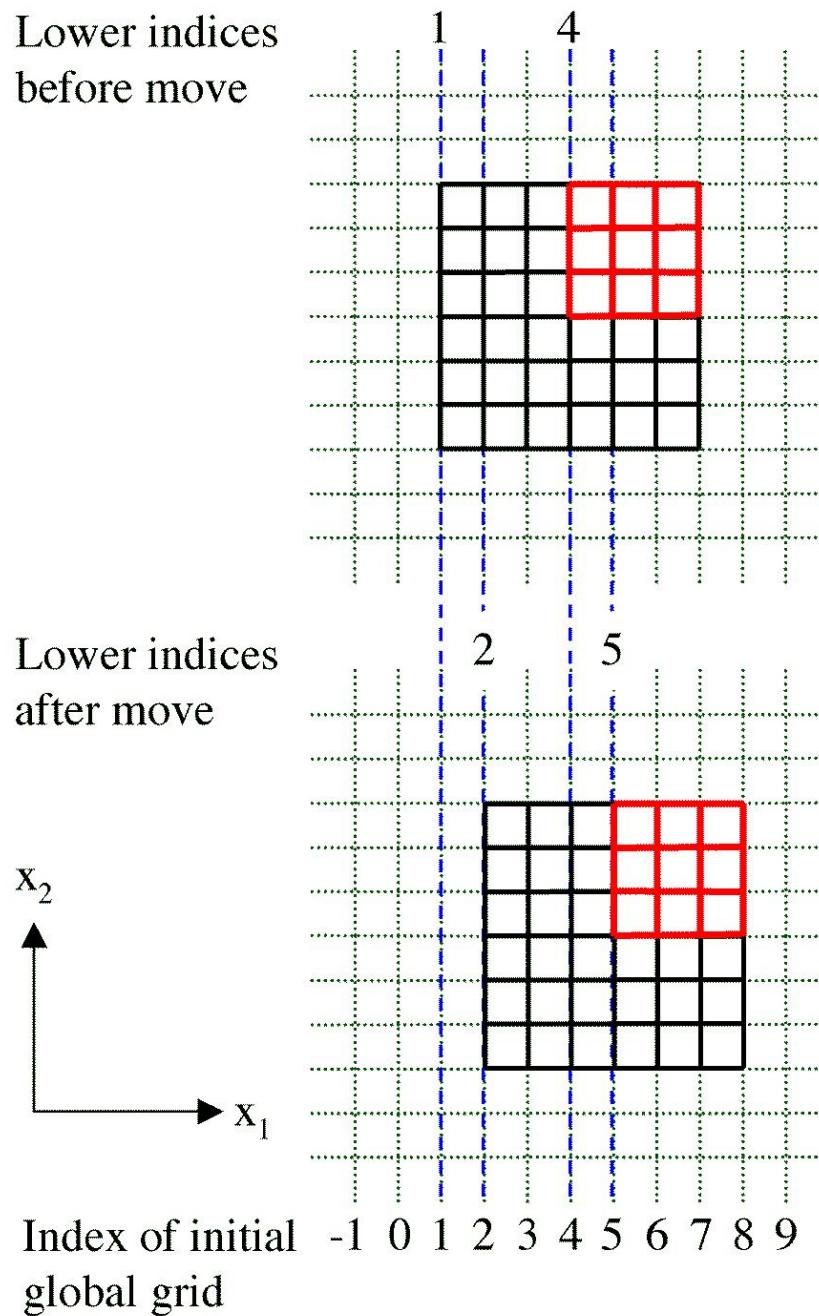


Figure 4.9: The figure shows how the motion of the lower boundary of the global and local grid is followed. The upper boundaries are tracked in the same way.

makes it possible to reassign a piece of simulation space from one node to its neighboring node for the purpose of dynamic load balancing. While dynamic load balancing is currently not implemented in OSIRIS, it could easily be done by adding some sort of algorithm, probably as part of the node-configuration object, that modifies the boundary motion of the local space object after the boundary motion of this object due to the global window motion has been done, but before it is used to steer the motion of other objects.

We next describe as an example to illustrate a moving internal boundary how the particle boundary conditions are handled in the moving window frame. Fig. 4.10 shows some of the details of the process of moving an internal boundary in a moving window simulation for a (particle-)species object. The first row of the figure shows the space and the grid of a 2-node simulation moving to the right. The black cells indicate the physical space of the simulations while the green cells correspond to the guard cells. If the simulation were not in a moving window then the following would have to be done to take care of the particle boundary conditions. The left boundary of the left node (node 1) and the right boundary of the right node (node 2) would have to be treated according to some other global, external boundary condition defined there. The particles in the right guard cells of the node 1 would have to be sent to the node 2 and placed at the corresponding positions within the physical space of the node 2. The left side of node 2 would be treated similarly.

For a moving window this process is more complicated. First, consider what happens at the external, global boundaries of the simulation window, .i.e., the left boundary of node 1 and the right boundary of node 2 in Fig. 4.10. The particles in the guard cells as well as in the first column of cells on the left side of the node 1 have to be discarded. The particles in the guard cells on the right side of node 2 are kept

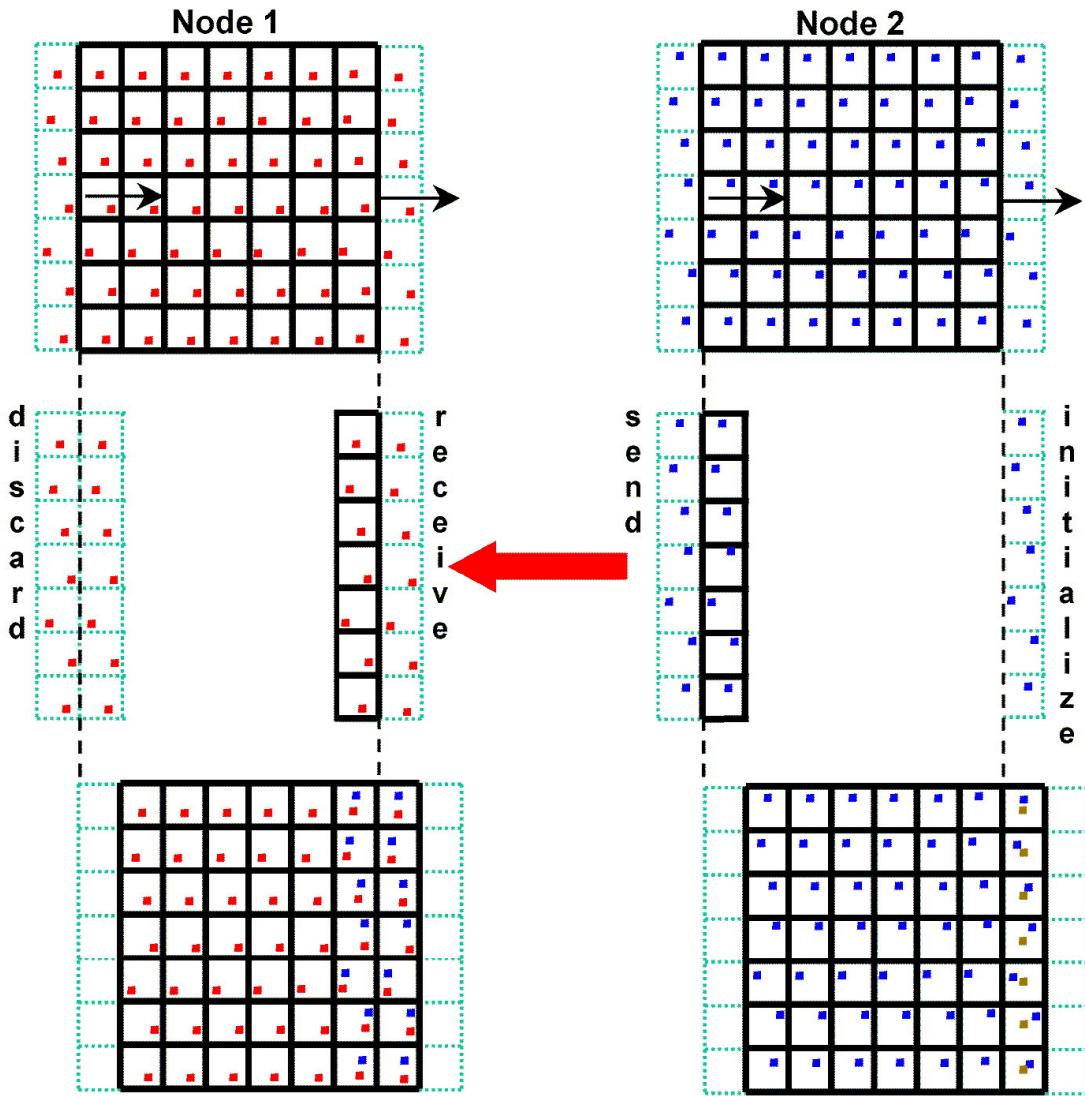


Figure 4.10: If a boundary between nodes moves then the boundary condition handling this case has to move the necessary data. This figure shows the motion of particles between nodes. The red, blue, and yellow colors for the particles in the figure are used to distinguish the different groups of particles. The red and blue particles are originally on the left and right node respectively. The yellow particles are newly initialized particles.

and new particles are initialized according to the plasma that needs to be simulated in the new area. At the internal boundary, a boundary that is physically moving to the right, the right node is sending particle information and the left node is receiving particle information due to the motion of the boundary. No particles are send from the left node to the right node. This asymmetry in the message passing will also be true for the electromagnetic fields. In the current version of the code the source fields are recalculated from scratch at every timestep after the whole system has been moved already. Therefore, the source fields do not need to be explicitly moved since they are calculated from the particles, which were already moved. For this reason the source field message passing is always symmetric. The third row of Fig. 4.10 shows the final particle distribution after the motion of the boundary and the passing of particles. Note that all the guard cells are now free of particles as they need to be.

4.7 Multi-Dimensional Issues

So far all the discussions of algorithms and the figures illustrating them ignored the problems arising from simulations with more than one spatial dimension. This is justified because extending all algorithms described above is straightforward when all boundaries of a given node, whether they are external or internal, are handled one dimension after another. The suggestion to exchange messages between nodes one direction at a time and therefore have multiple exchanges of messages for a multi-dimensional domain decomposition has been made before[58].

The idea is illustrated for the case of four nodes with internal boundaries between them in Fig. 4.11. The figure shows the communication patterns between the nodes. First each node takes care of its boundaries in x_1 (by first sending and then receiving

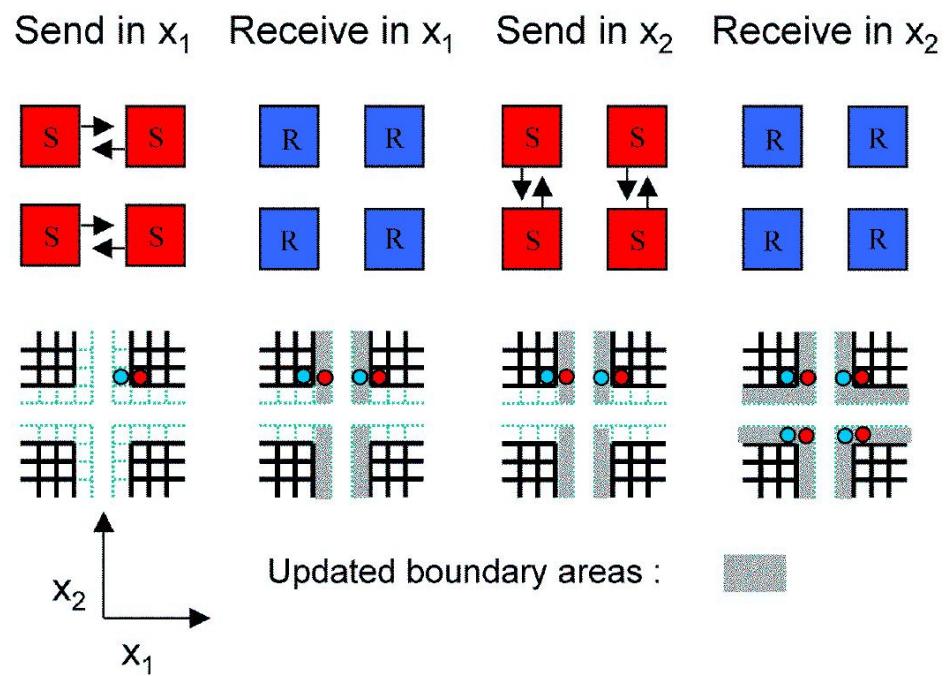


Figure 4.11: The communication pattern of OSIRIS for 2D domain decomposition.
The communications and boundary handling takes place for one direction at a time

information) and then it does the same thing in x_2 . This approach can obviously be extended to a third dimension as well.

To our knowledge the novel aspect of this idea as it is implemented in OSIRIS is that message passing at internal boundaries and boundary conditions at external boundaries are all handled in the same way since message passing is treated as one particular kind of boundary condition. The approach of taking care of all boundary conditions in a given direction first before taking care of the boundary conditions in another direction also works for cases where internal node-to-node boundaries in one direction are followed by external boundary conditions in another. For example, a node in a 3D simulation could have internal boundaries in x_1 which it takes care of by sending and receiving messages and then have external, conducting boundaries in x_2 which can be applied after the x_1 boundary conditions have been taken care of. Finally if the x_3 boundaries are internal boundaries again it will exchange information but this time with its neighbors in the x_3 direction.

4.8 Conclusion

This chapter described the most important object-oriented strategies and parallel algorithms of OSIRIS. Using object-oriented programming in Fortran 90 made it possible to combine several features and algorithms, some of which are novel. The most important ones are:

- The implementation of multi-dimensional domain decomposition.
- The implementation of the one-object-per node strategy.
- The implementation of a dynamic space algorithm using moving boundaries.

- The implementation of information exchanges between nodes as a boundary condition.
- The handling of boundaries one direction at a time including node-to-node boundaries.
- The encapsulation of the dimensionality and the coordinate system of a simulation by using polymorphic objects.

This code is operational and has been used to model a variety of problems for the first time. In addition the structure of its objects and the codes modularity make it easily extendable so that in the future new algorithms for increasing the efficiency of the code (e.g., dynamic load balancing, ponderomotive guiding center description for lasers) or for including new physics (e.g., ionization) can be integrated.

Chapter 5

Electron Beam Production Using Multiple Laser Beams in Plasmas

5.1 Introduction

D. Umstadter et al. [59] proposed the use of two orthogonal laser pulses in a plasma to trap and accelerate an ultra-short bunch of electrons. As envisioned, the first (or drive) pulse creates a plasma wave which is below its self-trapping or wavebreaking threshold. The transverse ponderomotive force of the second (or injection) pulse was argued to give electrons an extra kick forward in the wake direction, enabling them to be trapped and accelerated in the wake of the drive pulse. This geometry is illustrated in Fig. 5.1. Such a cathodeless injector (or perhaps more correctly, a plasma cathode) is of interest for a wide variety of applications including an injector for future linear accelerator technologies with short wavelength accelerating structures, a source of short pulses of light or x-rays, or a source of electron bursts for pulsed radiology and ultra fast pump-probe chemistry [60].

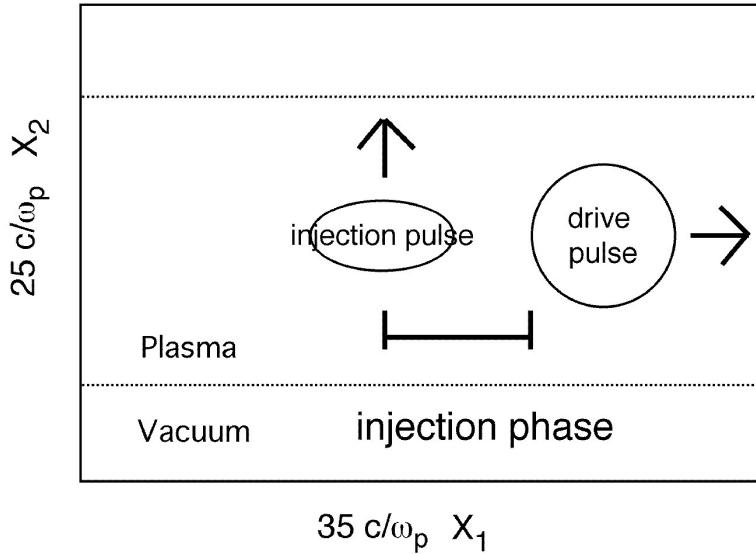


Figure 5.1: Geometry of the cathodeless injector concept. The injection phase of the injection pulse is defined by the distance between the trailing edge of the drive pulse and the center of the injection pulse when it crosses the drive pulse.

For plasma accelerator applications in particular, the scheme naturally overcomes problems of synchronizing the injector with the accelerator. Moreover, the rapid acceleration of the bunch in the plasma (order of 10-100 GeV/m) [1, 2, 3, 12] minimizes the effect of space charge that would be severe for such dense beams ($10^{14} - 10^{18} \text{ cm}^{-3}$) produced from a conventional thermionic photocathode [61].

The original analysis of Ref. [59] used single particle theory and estimates based on one-dimensional (1D) PIC simulations. This chapter contains the results from a detailed 2D and 3D PIC simulation analysis of this concept. We find that our results support the feasibility of such a cathodeless injection scheme, but that in the regime studied here the physical mechanism for the trapping is different from the one originally suggested. Furthermore, we show that the number of particles, emittance, and energy spread can all depend sensitively on the laser parameters and

the injection phase. Depending on the applications, these results place constraints on the allowable shot to shot jitter of the injection laser. Last, based on the new insight into the trapping mechanism, we put forth additional geometries, e.g., co- and counterpropagating pulses, as well as related injection schemes.

5.2 Acceptance of a Plasma Wave

Before considering the simulation results we present here a calculation of the acceptance[62] of a plasma wave. The acceptance of a plasma wave is an estimate for the upper limit of the emittance of a beam in a plasma wave since it is defined as the maximum transverse phase space volume that can be accelerated by an accelerating system. For a plasma wave the acceptance can be approximately calculated by assuming a transverse potential profile that is responsible for the focusing forces of the plasma wave. For a given transverse potential, $\phi_2 \equiv \phi_0(1 - x_2^2/w_p^2)$, we can find the maximum transverse momentum p_2 that a particle can have at a given transverse position x_2 before the particle can escape the potential well. Since the plasma wave as well as the particle both move with almost the same velocity, c , the potential function ϕ_2 will change slowly and we will neglect that change here.

We start with the condition that an electron is trapped transversely in the plasma wave's potential well, i.e., that the transverse kinetic energy has to be smaller than the energy needed to escape the transverse potential $|E_{k,2}| < |E_{p,2}|$.

$$\sqrt{p_2^2 c^2 + p_1^2 c^2 + m^2 c^4} - \sqrt{p_1^2 c^2 + m^2 c^4} < -e\phi_2 \quad (\phi_2 \leq 0)$$

This can be solved, giving an inequality for the p_2 of a trapped electron.

$$|p_2| c < \sqrt{\left(-e\phi_2 + \sqrt{p_1^2 c^2 + m^2 c^4}\right)^2 - m^2 c^4 - p_1^2 c^2}$$

Rearranging terms gives the following result:

$$|p_2| < mc \sqrt{-\frac{2e\phi_2\gamma_1}{mc^2}} \sqrt{1 + \frac{-e\phi_2}{2mc^2} \frac{1}{\gamma_1}} \equiv p_{2,max}(x_2) \quad (5.1)$$

where $\gamma_1^2 = 1 + (p_1/mc)^2$.

For linear waves $\bar{\phi}_2 = e\phi_2/(mc^2) \leq 1/2$; so to lowest order the second square root term can be approximated as unity. We use Eq. (5.1) to calculate the normalized acceptance [62].

$$A_n = 2 \int_{-\infty}^{\infty} \frac{p_{2,max}}{mc} dx_2 = 2 \int_{-\infty}^{\infty} \frac{\sqrt{-2me\phi_2\gamma_1}}{mc} dx_2 \quad (5.2)$$

Assuming the potential given in Eq. (2.1), and replacing w_p with $w_L/\sqrt{2}$, we get an approximate result for A_n by replacing the integration limits with $w_L/\sqrt{2}$ and $-w_L/\sqrt{2}$:

$$\begin{aligned} A_n &= 2\sqrt{2m e\phi_0 \cos(\alpha)} \gamma_1 \frac{1}{mc} \int_{-w_L/\sqrt{2}}^{w_L/\sqrt{2}} \left(1 - 2\frac{x_2^2}{w_L^2}\right)^{1/2} dx_2 \\ &= 2\pi w_L \sqrt{\gamma_1 \bar{\phi}_0} \sqrt{\cos(\alpha)} \end{aligned} \quad (5.3)$$

where α is the phase of the electron in the wave with respect to the potential maximum. If we assume γ_1 is of the order of the trapping threshold then $\bar{\phi}_0\gamma_1 = O(1)$, so

the ε_n for any cathodeless injection scheme is bounded by $\varepsilon < 2w_L\pi$. If the trapping of a particle bunch by a plasma wave doesn't take place at the maximum of the potential then $\cos(\alpha)$ is smaller than 1 and the emittance of the beam can be expected to be smaller than this upper bound. Note that if Eq. (2.13) is solved for ε_n then it results in $\varepsilon_n = 2\pi\sqrt{\gamma e E_{10}/(mc\omega_p)}(\sigma/w_L)^2\sigma$. Using $\gamma \approx \gamma_1$, $\bar{\phi}_0 = k_p^{-1}eE_{10}/(mc^2)$, and $\sigma = w_L$ leads to $\varepsilon_n = 2\pi w_L\sqrt{\gamma_1\bar{\phi}_0}$. This means that the acceptance is the emittance for a matched beam.

5.3 Simulation Parameters

The simulations were done with Pegasus [48] on a single node with the moving window to follow the laser pulse for extended periods of time. Fig. 5.1 shows the basic set up of the simulations. The following parameters are valid for most of the simulations results presented below unless stated differently. The simulation box has a size of $35 c/\omega_p$ in the x_1 direction and $25 c/\omega_p$ in the x_2 direction and the simulations run for a time of $105 \omega_p^{-1}$. The simulations use a 700×500 grid, a timestep, $dt=0.035 \omega_p^{-1}$, and four particles per cell.

In the beginning of the simulation, as the drive pulse enters into the cold plasma in the x_1 direction, it creates a plasma wave in its wake. At a later time the injection pulse is launched in a vacuum region at the side of the box and propagates in the x_2 direction crossing the path of the drive pulse. The frequency ratio ω_0/ω_p between the laser frequency and the plasma frequency is 5 for both pulses, and both have their polarization in the plane of the simulation. (This means the drive pulse has mainly an E_2 component and the injection pulse mainly an E_1 component). We adopt the notation of Ref. [59], where the normalized vector potential for the drive pulse is

$a \equiv eA_y/mc^2 = 1$ and for the injection pulse is $b \equiv eA_x/mc^2 = 2$, unless stated otherwise. We observed in the simulation that the plasma wave amplitude caused by $a = 1$ is about $\bar{\phi}_0 = 0.45$. The transverse profile for each laser is given by a Gaussian with a spot size of $3c/\omega_p$. The temporal profile has a symmetric rise and fall of the form $f(x) = 10 \cdot x^3 - 15 \cdot x^4 + 6 \cdot x^5$ with $0 \leq x = \tau/\tau_L \leq 1$. The value of τ_L is $\pi c/\omega_p$ for the drive pulse and $\frac{1}{2}\pi c/\omega_p$ for the injection pulse; thus the simulations have fewer laser cycles than in typical experiments. We define the injection phase ψ to be the distance between the back of the drive pulse and the center of the injection pulse as it crosses the axis. This is shown in Fig. 5.1.

In order to convert the simulation results to physical units, we assume a plasma density of $10^{16} cm^{-3}$. If not stated differently all quantities are given in normalized Gaussian units with the plasma frequency equal to 1. The number of accelerated electrons is estimated from the simulations as follows:

$$N = \frac{\# \text{ of trapped simulation particles}}{\# \text{ of particles per cell}} \cdot n \cdot dx_1 \cdot dx_2 \cdot \Delta x_3 \times ((mc^2)/(4\pi e^2 n))^{3/2} \quad (5.4)$$

Here n is the electron density in cm^{-3} , dx_1 and dx_2 are the cell sizes in the x_1 and x_2 direction, and Δx_3 is an assumed extension in the x_3 direction. dx_1 , dx_2 and Δx_3 are in normalized units. We assume Δx_3 to be equal to Δx_2 , the width of the group of accelerated particles in x_2 . The normalized emittance is calculated as:

$$\varepsilon_n = \gamma \times \frac{\Delta p_2}{p_1} \cdot \Delta x_2 \times \left((mc^2) / (4\pi e^2 n) \right)^{1/2} \quad (5.5)$$

with $\gamma = \sqrt{1 + \vec{p}^2} \approx p_1$. Here, p_1 is the average longitudinal momentum and Δp_2 and Δx_2 are the width of the distributions of p_2 and x_2 for the group of accelerated

particles. It should be noted that the number of electrons as well as the normalized emittance both scale with $n^{-1/2}$. All quantities including the energy spread are calculated after the final timestep of the calculation, i.e., after a propagation distance of $105c/\omega_p$ (particles are trapped, $\gamma > \gamma_\phi$, between 50 and 60 c/ω_p -see Fig. 5.5). The values of Δx_2 and Δp_2 are defined to be the standard deviations of the particles bunches for these quantities. The energies of the trapped particles are around 10 MeV, which is of the order of the theoretical energy gain $\Delta W = 16MeV$ (Eq. (2.5) with $\eta = 1$, $\bar{\phi}_0 = 0.45$, and $\gamma_\phi = 5$) that would be obtained over the dephasing distance of $80c/\omega_p$ [see Eq. (2.6)]. The trapping threshold for these simulations is 0.05 MeV [see Eq. (2.4)]. The simulation show that trapped particles close to the maximum accelerating gradient, which is consistent with the result above.

5.4 2D Simulation Results

The engineering results of the simulations can be summarized in Fig. 5.2 and Fig. 5.3. In Fig. 5.2 we plot the number of trapped electrons, the emittance, and the energy spread as a function of the injection phase for a fixed value of the injection amplitude, $b=2.0$. In Fig. 5.3, we plot the same quantities as in Fig. 5.2 but as a function of the injection amplitude for a fixed value of the injection phase, $\psi = 1.3\pi$. All other parameters have the values given before. Note that negative values for ψ mean that the center of the injection pulse crosses the x_2 axis before the end of the drive pulse.

The most notable feature of Fig. 5.2 is the large variation of the three beam quantities as a function of ψ and especially the strong difference in the number of particles and their emittance between positive injection phases larger and smaller than π . The direct overlap of the injection pulse with the drive pulse (i.e. injection

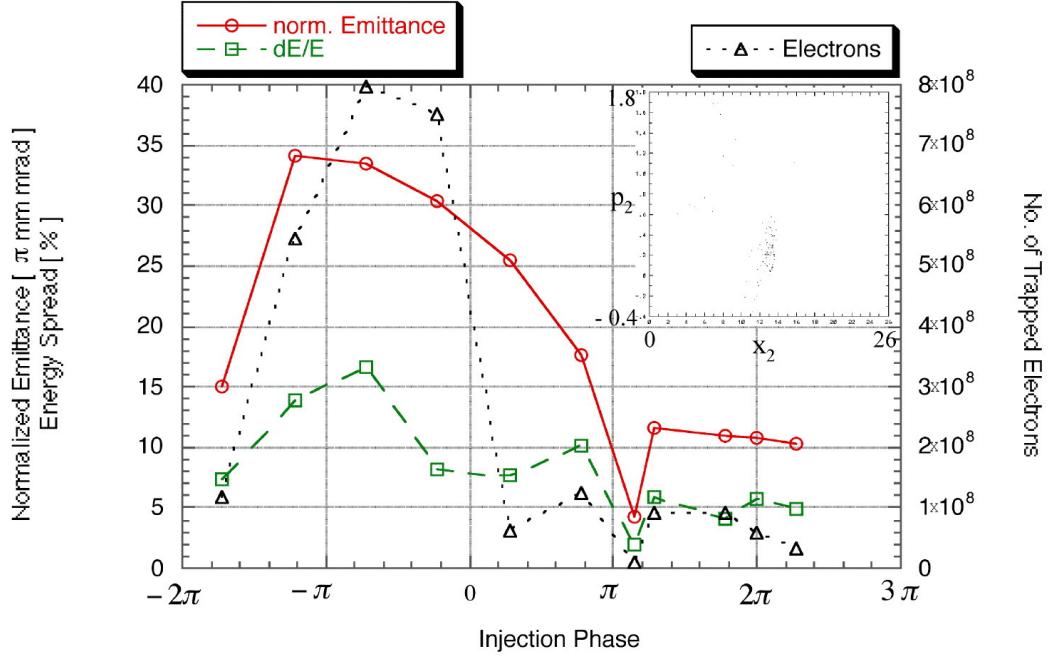


Figure 5.2: The number of trapped electrons, the normalized emittance, and the energy spread of the trapped particles as a function of the injection phase. The injection amplitude b is 2.0 and the drive amplitude a is 1.0. The connecting lines between the data points have been added to make it easier to distinguish the different data. The inset shows the raw data for the transverse phase space of the trapped particles that is used to calculate the emittance for the simulation at $\psi = 1.8\pi$.

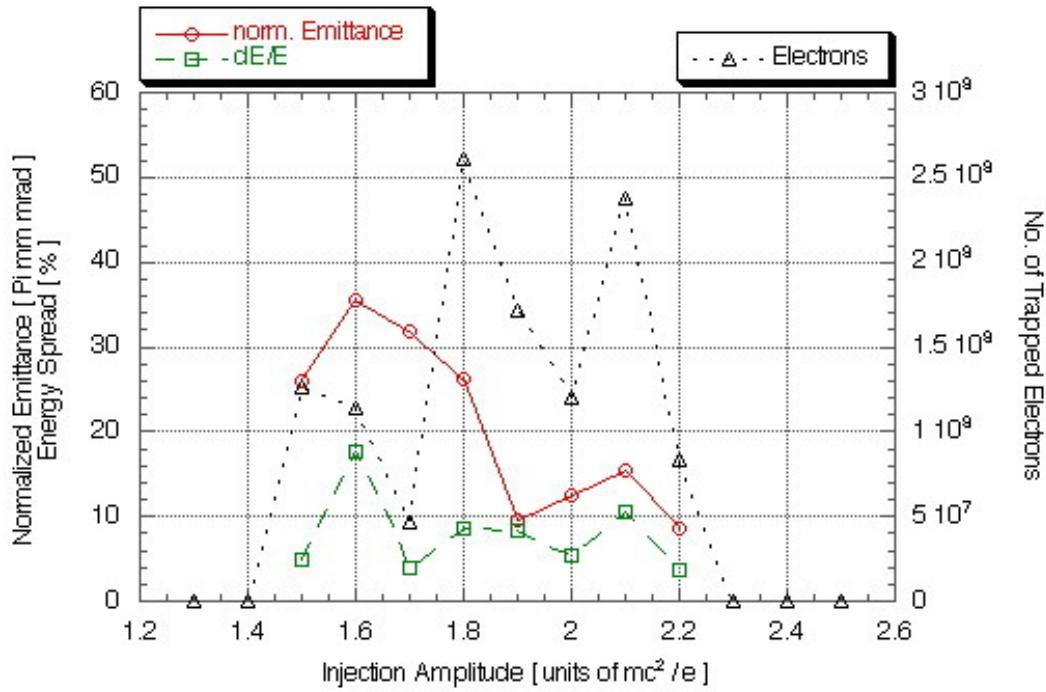


Figure 5.3: The number of trapped electrons, the normalized emittance, and the energy spread of the trapped particles as a function of the injection amplitude. The injection phase ψ is 1.3π . All other parameters are the same as the ones used in the simulations of Fig. 5.2. The connecting lines between the data points have been added to make it easier to distinguish the different data.

phase smaller than π) clearly yields the largest number of trapped particles. The maximum number of trapped electrons corresponds to 8×10^8 at a plasma density of $10^{16} cm^{-3}$ (or to 6×10^7 at a density of $10^{19} cm^{-3}$). Note that 100% beamloading [63] corresponds to $N = 5 \times 10^5 \bar{\phi}_0 \sqrt{n_0 cm^3} A cm^{-2} \approx 8 \times 10^9$ for $n_0 = 10^{16} cm^{-3}$ where we use a laser beam cross section of $A = \pi w_p^2$ with $w_p = w_L/\sqrt{2} = \frac{3}{\sqrt{2}} c/\omega_p$. Therefore, there is $\leq 10\%$ beamloading for negative and $\leq 1\%$ positive injection phases.

The number of particles decreases by an order of magnitude for injection phases larger than π . The normalized emittance on the other hand is better for injection phases larger than π , with the smallest normalized value of 3π mm mrad in a $10^{16} cm^{-3}$ density plasma (or 0.1π mm mrad at $10^{19} cm^{-3}$). Note from Eq. (5.3) that the acceptance for the plasma wave places an upper bound on the emittance of $2w_L\pi = 300\pi$ mm-mrad for $n_0 = 10^{16} cm^{-3}$. However since the particles are getting trapped at a phase close to the maximum accelerating phase of the plasma wave (i.e., close to a zero for the focusing field), the cos term in Eq. (5.3) is small. We therefore expect the emittance to be smaller than this upper limit. For injection phases smaller than π the emittance increases by a factor of five. The emittance therefore seems to grow with the number of particles. Although this is suggestive of some sort of space charge degradation, we will show later that space charge is not important. Instead, we believe that the relatively larger emittance and number of particles at smaller ψ are both due to a stochastic interaction between the plasma and the overlapping laser fields.

The energy spread of the accelerated bunch also varies widely; it is between 2% and 17% at a beam energy of 10 MeV and we expect the energy spread $\Delta E/E$ to scale as $1/\gamma$ for simulations with larger dephasing energies (i.e., larger values of ω_0/ω_p), since $\Delta E/E \propto \Delta E/\gamma$ and ΔE is not expected to change significantly. There is an

interesting difference between the behavior of the energy spread and the number of particles on the one hand and of the emittance on the other hand for the larger injection phases in Fig. 5.2. The energy spread, and to some extent the number of particles fluctuate as a function of ψ . The emittance remains almost constant which suggests that it is determined by qualities of the accelerating plasma wave and not by details of the injection process like the injection phase.

Although the simulations with $b=2.0$ produce similar numbers of particles at $\psi = 1.3\pi$ or 1.8π as can be seen from Fig. 5.2, for $b=1.8$ the number of particles changes from a 10^8 at $\psi = 1.3\pi$ (see Fig. 5.3) to nearly zero at $\psi = 1.8\pi$ (data not shown in figures). This indicates that the results of the simulations are quite sensitive to b and ψ so that the curve found in Fig. 5.2 for the injection phase dependence at injection amplitudes of 2.0 is not readily applicable to other values of this parameter but merely indicates the magnitudes of various quantities that can be obtained.

The value of $\psi = 1.3\pi$ is used for the simulations of Fig. 5.3 since it seems to have close to an optimal injection phase judging from the data of Fig. 5.2. As a function of the injection amplitude the normalized emittance and the energy spread do not seem to show any systematic behavior on the scale that is resolved by the simulations. The values of the energy spread vary between 4% and 18% while the values for the emittance are between $10 \pi \text{ mm mrad}$ and $40 \pi \text{ mm mrad}$. Depending on the application, these variations will place a limit on the tolerable shot to shot laser jitter.

The number of trapped electrons on the other hand seems to show a systematic behavior. What should be expected is that the number of trapped particles first rises with increasing injection amplitude and then falls off. This is recognizable in the figure even though the curve is quite noisy. The decrease with an increased amplitude causes

an increase in transverse momentum, P_2 , that is transferred to the particles by the injection pulse. At a certain value, the transverse momentum becomes large enough to prevent the trapping of the particles.

We may use Fig. 5.2 and Eq. (5.4) and Eq. (5.5) to obtain an interesting scaling law for the brightness of the plasma cathode injector. The normalized brightness can be defined by $B_n = I/\varepsilon_n^2$ [62] for axially symmetric beams. For the average current of a bunch we find $I \propto N/T_p = N/\omega_p$, where T_p is the plasma wave period. As noted earlier N scales with $n^{-1/2}$ while ω_p scales as $n^{1/2}$. Therefore, the product $N\omega_p$ does not depend on the density as the simulation results are scaled to different densities for fixed values of ω_0/ω_p , a , and b . For example at $\psi = 1.8\pi$, we get $I_{max}=220$ Amps and $\varepsilon_n = 11\pi mm\ mrad [(10^{16}cm^{-3})/n]^{1/2}$ which scales as $n^{-1/2}$. Combining these results predicts a brightness of $B_n = 1.8 \cdot 10^7 \times n / (10^{16}cm^{-3}) \times Amps/cm^2$ which scales linearly with density.

The insensitivity of the beam current to the plasma density should also hold if ω_p , a , and b are changed. This can be argued as follows. The beam current can be written as $I = en_b c \times \sigma^2 \pi$ where n_b is the beam density. If we normalize n_b with respect to the plasma density n_0 and the spot size σ with respect to c/ω_p we find that

$$I = en_0 c \left(\pi c^2 / \omega_p^2 \right) \frac{n_b}{n_0} (\sigma c / \omega_p)^2 = \frac{I_A}{4} \frac{n_b}{n_0} (\sigma c / \omega_p)^2 \quad (5.6)$$

This expression for I is insensitive to the plasma density for various laser parameters, if the normalized beam density and the spot size are relatively insensitive to the plasma density. We expect that the ratio n_b/n_0 is not a strongly varying function of $\gamma_\phi = \omega_0 \omega_p$, since the trapping threshold asymptotes for large γ_ϕ [see Eq. (2.4)]. Note also that since n_b/n_0 is typically less than 1 and σ is typically c/ω_p or less, this shows

that the current is typically some fraction of the Alfvén current.

It is interesting that despite their high brightness and density, the bunches are not space charge dominated. From the discussion above I/I_A is of the order of 10^{-2} , while $(\sigma/\varepsilon_n)^2$ is typically of order unity. Thus, using Eq. (2.12) we find that $\rho \ll 1$ at all times in the plasma and the beam is emittance dominated. We note that once the bunch leaves the plasma and expands in free space it can rapidly become space charge dominated. For beams generated by the cathodeless injection scheme this typically occurs in a distance of the order of $1 \text{ cm} \times [(10^{16} \text{ cm}^{-3})/n]^{1/2}$. Since the effects of space charge can be neglected, it is possible to apply Eq. (2.13), the condition for matched beams. For the simulation parameters the left side of Eq. (2.13) has values between 2 and 3, which means the external force term is larger than the diffraction term. For the beam emittances in the simulations, we also note that ε_n is between 0.01 and 0.12 times the plasma wave acceptance that was calculated above. The numbers for the matched beam condition and the emittance to acceptance ratio indicate that once the electrons are “injected” they are well within the parameters of stable acceleration for the plasma wave.

To achieve high energies in the LWFA the laser pulse must propagate through many diffraction or Rayleigh lengths of plasma. One way to guide a pulse is to use a parabolic density channel [64, 65]. Therefore the cathodeless injection scheme may need to work in plasma channels. We have carried out a simulation in which the drive pulse propagated down a channel and the injection pulse propagated across the channel. The channel had a width of $3.25c/\omega_p$ and the density was decreased by 40% in the middle of the channel. In the simulation the number of trapped particles as well as the emittance of the particle bunch are reduced to about 20% from their values in the uniform plasma case. We also note that for all results presented in this chapter

so far the initial plasma is cold. We have done simulations with a 1 KeV plasma and the number of electrons as well as the emittance decrease to about 40% of the cold plasma values.

Insight into the mechanism of trapping can be gained by studying the original location and the trajectories of the trapped particles. In Fig. 5.4, we plot the original (x_1, x_2) positions for all the trapped particles from two simulations. The red points are for $\psi = 1.8\pi$ and $b=2.0$, while the blue points are for a $\psi = 1.3\pi$ and $b=1.8$. There are several important points to be noticed. The first is that for both cases the particles are to the left of the injection pulse. Therefore, these particles feel a transverse ponderomotive force to the left not to the right as was presumed in Ref. [59]. We have verified this by rerunning the simulations without the drive pulse to see only the effect of the injection pulse.

To gain a deeper understanding of the process, we follow the momentum of a single, typical, trapped particle as function of time in the 2D simulation. We consider a particle for the case of $\psi = 1.3\pi$ and $b=1.8$. The data are shown in Fig. 5.5. The initial momentum is zero since the simulation uses a cold plasma. We show here the results for only one particle, but the curves are very similar for other trapped particle phase space trajectories in this simulation.

The solid curve in Fig. 5.5 shows the longitudinal momentum of the particle; the dotted curve shows p_1 for the same particle in a simulation where the injection pulse is not launched. As expected, the same particle simply oscillates in the wake of the drive pulse. In the full simulation we can see that the injection pulse has completely passed by the test particle at about the time $t=31.7$. Although the injection pulse has an impact on the particle, the really large changes occur at a time when the injection pulse has already left the area of the test particle. This indicates that the

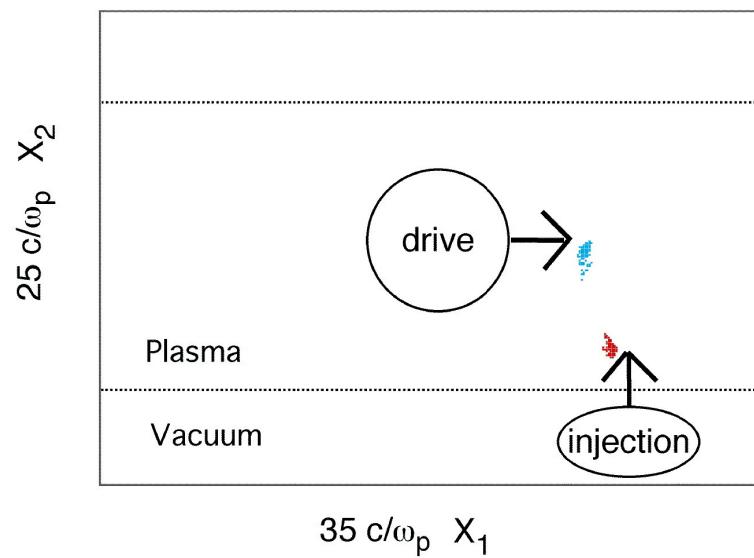


Figure 5.4: The figure shows the initial position of trapped particles for two different simulations. The red particles come from a simulation with $\psi = 1.8\pi$ and $b = 2.0$. The blue particles come from a simulation with $\psi = 1.3\pi$ and $b = 1.8$. The position of the drive pulse in the figure is illustrative and does not match $\psi = 1.3\pi$ or $\psi = 1.8\pi$.

trapped particle gets the extra momentum needed for getting trapped from the interaction of the two plasma wake fields created by these pulses rather than from any effect directly related to the laser pulses (since those have already left the area of the particle). Note that the trapped particle goes through one full oscillation (accelerating, decelerating, and accelerating again) before it is trapped. This feature that the particles get trapped in a multi-step process (acceleration-deceleration-acceleration) caused by the interaction of the wake fields is not unique to this particular simulation. Other simulations with different values for ψ and b showed the same process.

Fig. 5.6 shows the E_1 field at $t=42.0$. The blue areas accelerate, while the red areas decelerate electrons with respect to the x_1 direction. The green cross marks the position of the test particle shown in Fig. 5.5 at that time. The position of the particle in this picture is consistent with the development of p_1 in Fig. 5.5. The particle is at the edge of the accelerating area and will slip back into the decelerating area. The field magnitude of the decelerating area is clearly smaller than that of the accelerating area. The spatial structure of the E_1 field seen in this figure can qualitatively be understood as mainly a superposition of the longitudinal field of the drive pulse wake and the transverse field of the injection pulse wake.

5.5 3D Simulation Results

A key motivation for developing OSIRIS was to be able to routinely carry out 3D simulations. In this section we give one example of the usefulness of 3D simulations. In particular, we use them to check the validity of the 2D simulation results presented earlier in this chapter. The 3D simulation results presented below are for the same parameters as the 2D simulation with an injection phase of $\psi = 1.3\pi$ and a normalized

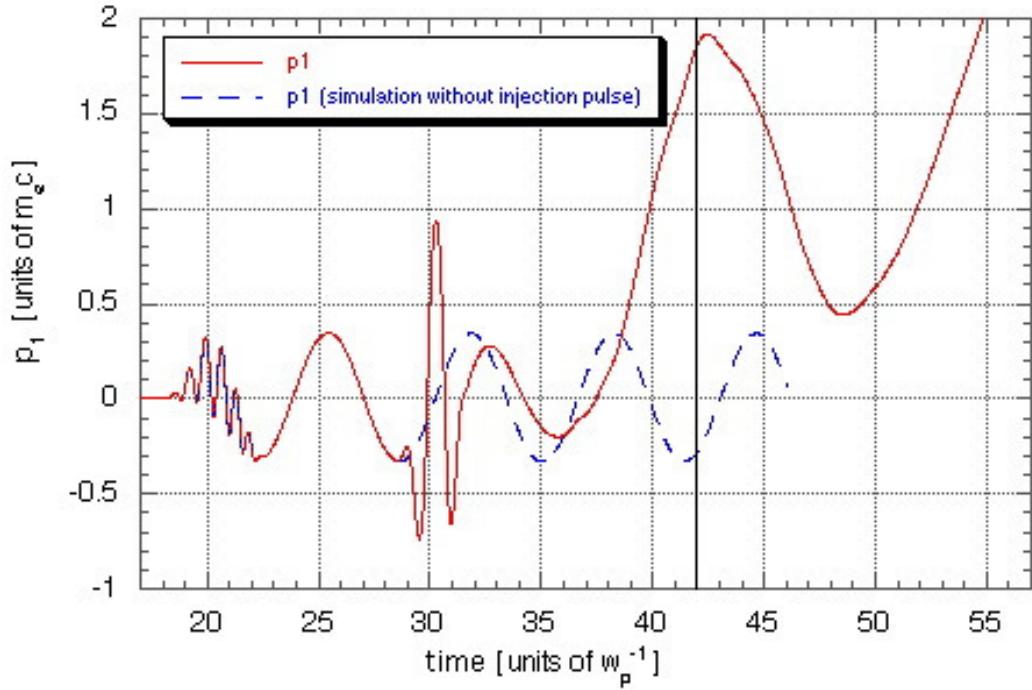


Figure 5.5: p_1 of a test particle as a function of time. The two curves are the results from simulations with (solid) and without (dashed) an injection pulse. $\psi = 1.3\pi$ and $b = 1.8$ for the simulation with an injection pulse. The initial position of the test particle is given by offsets of -2.3 in x_1 and -0.1 in x_2 relative to the intersection of the pulses (see Fig. 5.4). The vertical line in the figure indicates the time $t = 42.0$ at which the electric fields are given in Fig. 5.6.

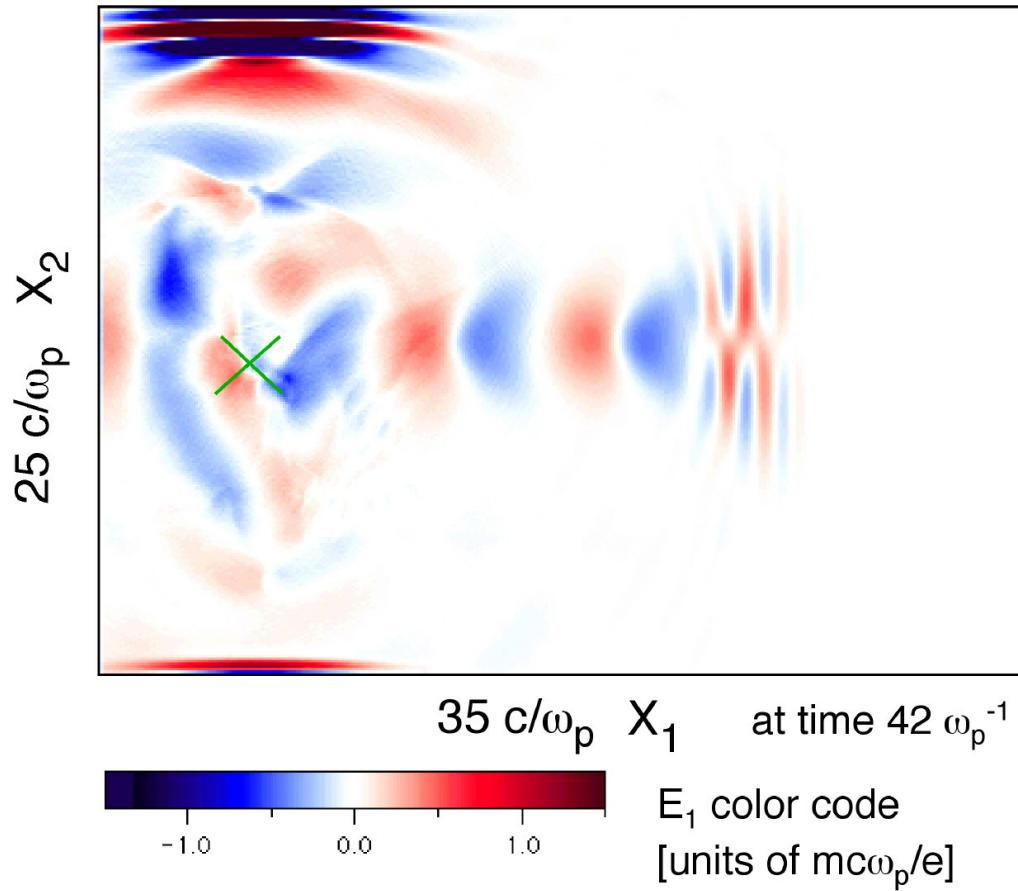


Figure 5.6: The E_1 field at the time $t = 42.0$ for $\psi = 1.3\pi$ and $b = 1.8$. The cross indicates the position of the test particle shown in Fig. 5.5

vector potential of $b = 1.8$ for the injection pulse. The 2D result for the electron number, the emittance, and the energy spread were summarized in Fig. 5.3. In order to decrease the computational cost of the simulation the numerical parameters are slightly modified in the 3D simulation. For the 3D simulation the simulation box has a size of $35.09 c/\omega_p$ in the x_1 direction and $25.13 c/\omega_p$ in the x_2 and x_3 directions. The simulation uses a $400 \times 280 \times 280 \simeq 31 \times 10^6$ grid, a timestep $\text{dt}=0.0513 \omega_p^{-1}$, and four particles per cell. The total number of particles in the simulation is ~ 45 million. The injection laser for the 3D simulation propagated in the x_2 -direction as it did in the 2D simulations.

The results of the 3D simulation regarding the injected electrons are summarized in Tab. 5.1. Tab. 5.1 gives the mean values and widths \bar{x}_i , \bar{p}_i , σ_{x_i} , and σ_{p_i} as well as the number of the injected electrons after the final timestep for the two distinct particle bunches that got injected during the simulation. Various phase space plots for the particle data are plotted in Fig. 5.7 and Fig. 5.8. (Note that in these phase space plots the axes that are not shown are axes that have been integrated over, e.g., in Fig. 5.7 x_2 , x_3 , p_2 , and p_3 have been integrated over.)

Fig. 5.7 shows the longitudinal phase space x_1-p_1 of the injected particles at the end of the simulation. The figure shows clearly that electrons are injected into two distinct buckets of the plasma wake but most of the injection takes place in the later bucket. This later bucket is the same that accelerated electrons in the 2D simulation. Using Eq. (5.4) to calculate the number of trapped electrons gives 1.7×10^7 electrons for the first electron bunch and 2.6×10^8 electrons for the second electron bunch. A comparison with the 2D simulation result shows that the number of trapped electrons in the second bunch for the 2D and 3D simulations is the same within less than%5. That the first bunch is not seen in the 2D simulation can be understood when consid-

	3D - Bucket 1	3D - Bucket 2	2D - Bucket 2
\bar{x}_1	111.84	105.88	-
σ_{x_1}	0.66	0.27	0.4
\bar{x}_2	1.79	-0.59	-
σ_{x_2}	2.72	1.45	1.0
\bar{x}_3	-0.02	-0.11	-
σ_{x_3}	1.85	0.97	-
\bar{p}_1	13.40	13.71	16.5
σ_{p_1}	2.98	1.45	0.7
\bar{p}_2	0.39	-0.11	-
σ_{p_2}	0.60	0.35	0.8
\bar{p}_3	-0.02	-0.03	-
σ_{p_3}	0.38	0.25	-
N	1.7×10^7	2.6×10^8	2.6×10^8

Table 5.1: The mean values and widths \bar{x}_i , \bar{p}_i , σ_{x_i} , and σ_{p_i} as well as the number of the injected electrons after the final timestep for the two distinct particle bunches that got injected in the 3D simulation as well as for the bunch in the second bucket of the corresponding 2D simulation. Some values for the 2D simulation were not available.

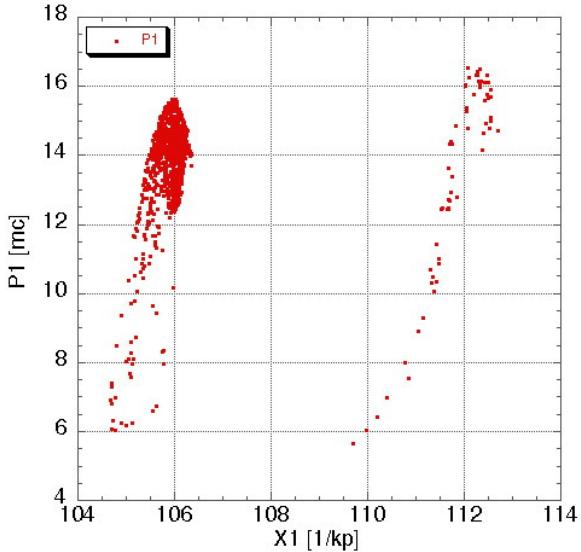


Figure 5.7: The longitudinal phase space x_1 - p_1 of the injected particles at the end of the 3D simulation.

ering the number of simulation particles involved in the 2D and 3D simulations. The first and second groups of electrons are represented by ~ 50 and ~ 1000 , simulation particles in the 3D simulation. The number of particles used in the 2D simulation to represent the bunch which corresponds to the second bunch in the 3D simulation is ~ 100 . This means that the smaller bunch in the 3D simulation is probably too small to be correctly resolved in the 2D simulation. For this reason only the results for the second, larger group of accelerated electrons will be considered when comparing the 2D and 3D simulations.

The final energy in the 2D simulation is $16.5mc^2$. This is more than the final energy in the 3D simulation which is $13.7mc^2$. The difference is probably due to the fact that the laser pulse is diffracting more quickly in 3D than in 2D. As a result the laser intensity decreases more quickly and therefore the wakefield is smaller in the 3D simulation. [see Eq. (2.19)]. The relative energy spread is larger in the 3D simulation

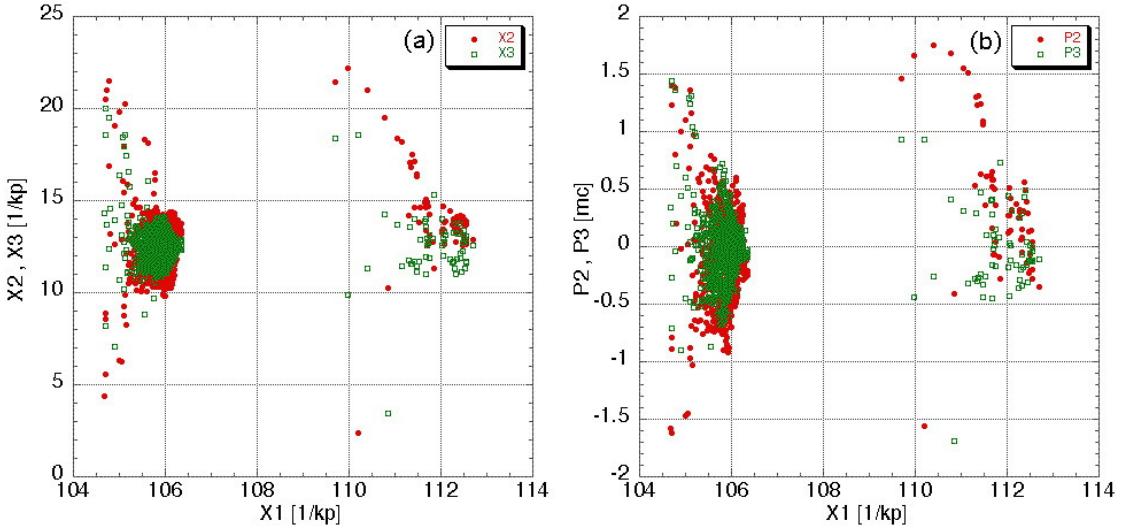


Figure 5.8: The transverse phase space data at the end of the 3D simulation. (a) shows the x_1 - x_2 and the x_1 - x_3 distribution of the injected particles. (b) shows the x_1 - p_2 and the x_1 - p_3 distribution of the injected particles particles.

than in the 2D simulation. The 3D simulation shows a $\Delta E/E = 21\%$ while the 2D simulation has an energy spread of about 9%.

In Fig. 5.8 the various transverse phase spaces of the injected electrons are shown. As before there are two bunches but only the properties of the larger group (i.e. later bunch) will be discussed here. In Fig. 5.8a) the x_1 - x_2 and the x_1 - x_3 phase spaces are shown. As expected there are differences between the two distributions. In particular the width of the distribution in x_2 is 1.5 times the width of the distribution in x_3 .

Fig. 5.8b) shows the x_1 - p_2 and the x_1 - p_3 phase spaces. Again the width the distribution in p_2 is larger than the width in p_3 . The ratio of the two values is 1.4. Using the values for the width of the distributions to calculate the normalized emittance of the particle bunch in the x_2 and x_3 directions gives $\varepsilon_{n,2} = 27\pi mmmrad$ and $\varepsilon_{n,3} = 13\pi mmmrad$. The value of $\varepsilon_{n,2}$ is the same as the emittance in the 2D simulation. Another piece of information in the table above to note is that the width

of the transverse distributions of the particles is considerably larger than the distance of the mean values from zero (which means for x_2 and x_3 the distance from the center of the wakefield).

There are two main conclusions to be drawn from the results of the 3D simulation results. The first is that 2D simulations are an excellent tool for studying the transverse laser injection process since they show the same general behavior and many of the injected beam parameters are quantitatively the same in the 2D and 3D simulations. The second result is that there is a 50% asymmetry for the spotsize and the transverse momentum spread in the two transverse directions. These results are preliminary and more work is needed.

5.6 1D Models

The simulation results raise the question whether the injection is mostly a linear effect that arises from the superposition of the two plasma waves or whether it is essentially a non-linear effect arising from the interaction of the two plasma waves mediated by the plasma. To address this question, we show the results of non-self-consistent 2D simulations and 1D numerical calculations (Fig. 5.9). The 2D non-self-consistent simulations are done by turning off the field solver of the Pegasus code and instead calculating the fields of the lasers and their wakes analytically from linear theory at each timestep [42]. As a result it is possible to follow test particles in the fields caused by the linear superposition of the two laser pulses and their wakes. In a second non-self-consistent 2D simulation the injection pulse is neglected while the linear wake it produces is not.

The 1D numerical calculations use the following electric fields for the wakes to

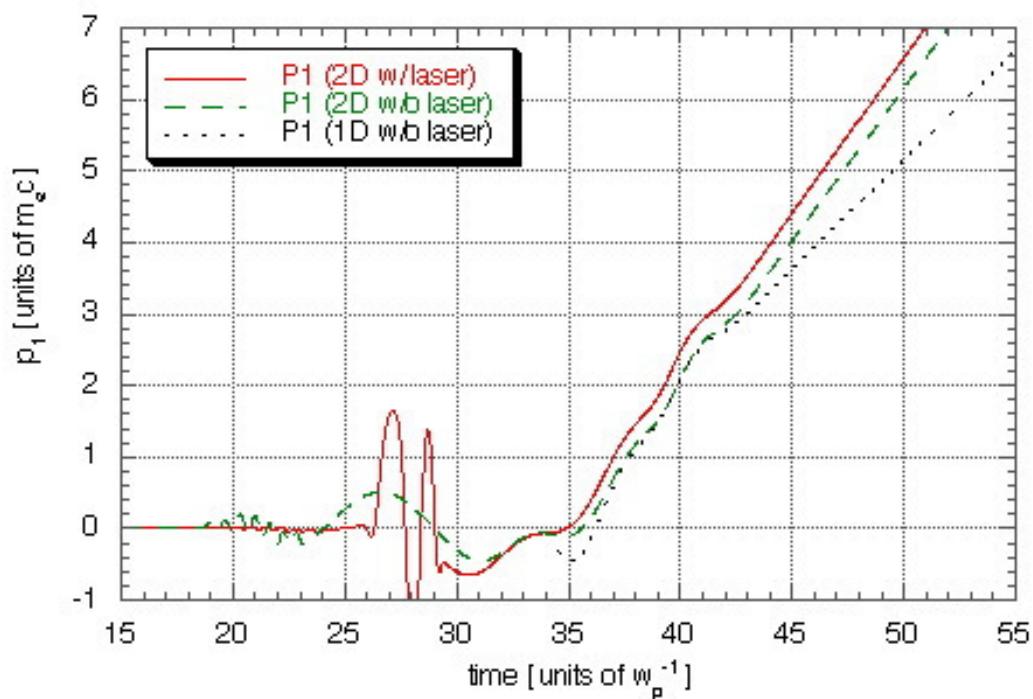


Figure 5.9: p_1 vs. time for a particle in a 2D non-self-consistent simulation (solid) and for a 1D numerical calculation(dashed). The 1D calculation had the starting parameters $x_0 = -0.5$, $w_0 = -1.5$ and $\varphi_0 = \frac{4}{3} \cdot \pi$.

calculate the trajectory of a particle.

Wake from the drive pulse:

$$E_D = E_{D,max} \cdot \sin(k_p x - \omega_p t - \varphi_0) \quad (5.7)$$

Wake from the injection pulse:

$$E_I = E_{I,max} \cdot 2 \cdot e^{1/2} \cdot \sin(\omega_p t) \cdot x/w_0 \cdot e^{-2(x/w_0)^2} \quad (5.8)$$

These equations follow from the ones used for the non-self-consistent 2D simulations. The laser fields are also omitted in this 1D calculation. The initial conditions of the particle are given by its position in the plasma wave described by Eq. (5.7). $E_{D,max}$ and $E_{I,max}$ are taken as 0.45 and 0.35 since these are the values seen in the self-consistent calculation with $\psi = 1.3\pi$ and $b=1.8$. For the 2D non-self-consistent simulations the laser amplitudes are slightly adjusted to yield those values, too. All other parameters of the non-self-consistent 2D simulations and the 1D calculations are the same as in the self-consistent simulation with $\psi = 1.3\pi$ and $b=1.8$, unless stated otherwise.

The results of these idealized models can be seen in Fig. 5.9. The linear superposition of two crossed plasma waves (solid line) creates conditions under which particles get trapped. On the other hand the actual development of p_1 after the injection pulse has passed the particle looks different from the self-consistent results, which suggests that the trapping process is modified by the nonlinear interaction between the two plasma waves. The multi-step trapping discussed above seems to be a result of this modification.

The result of the 2D simulation without the injection laser (dashed line) differ

strongly from the one with the laser up to time the injection pulse has passed. After that time the two curves are rather similar and they differ mainly due to a small displacement of one compared to the other along the time axis. Noting that the two curves belong to particles with a different original position in the simulation, suggests that the effect of the ponderomotive force is to change which particles are trapped. Direct comparison of the temporal evolution between the 1D and 2D results is complicated because the same particles are not trapped. We place the 1D curve in such a way that it is easy to compare the trajectories once a particle is trapped. The similarity of this curve with the curves from the non-self-consistent 2D simulations indicates that the basic physics of the trapping can be studied by Eq. (5.7) and Eq. (5.8).

We close this section by commenting that determining whether the trapping results from a ponderomotive kick or from interfering wakes is important to developing simplified models to explain and extend the scheme investigated here. Our results suggest that the trapping is due to the interaction of two plasma waves rather than a plasma wave and a ponderomotive kick (impulse). However, this does not rule out the possibility that a different choice of parameters for the injection pulse will result in trapping due to a direct kick by the transverse ponderomotive force [66]. A possible advantage of the mechanism found in our results here relates to Eq. (5.6). If the trapping of particles is caused by dephasing them with respect to the accelerating wake, as we find it here, rather than from directly increasing their momentum then n_b/n_0 could be a much weaker function of $\gamma_\phi = \omega_0/\omega_p$ indicating that this injection method might also be useful for larger γ_ϕ .

Understanding the trapping mechanism allows one to propose and understand other possible geometries. A co-propagating geometry is the easiest to visualize [66].

The second pulse should be tightly focused to interact with a single (or perhaps a few) bucket and it should be phased to enhance the original wake to amplitudes above wavebreaking. In this geometry the ponderomotive force and the wake are intimately connected for the first oscillation. However, in subsequent oscillations the interaction of the wakes could lead to injection.

A counter-propagating geometry is more complicated (This scheme differs from a recent idea of Esarey et al. [67] which considered a co-linear geometry with an intense pump pulse and two counter-streaming injection pulses). Once again a second pulse is focused tightly to interact with only a single bucket. In this case the injection pulse is phased to reinforce the electrons motion as they move backwards. Therefore, the wake is unequivocally essential in order for the electrons to be trapped as they oscillate forward. In simulations of this scheme we have observed an additional trapping mechanism at the plasma boundary. This mechanism might be of interest for experiments in which the plasma boundaries are sharp.

In another possible scenario, a plasma wave moving across the first wake (other geometries are also possible) could be gradually built up over time until a trapping threshold is reached. This scheme also clearly would rely only on the interfering wakes.

5.7 Conclusion

In this chapter, the injection scheme proposed in Ref. [59] was studied using the results of 2D and 3D PIC computer simulations. We find that the beam brightness and quality compares reasonably with that of electron bunches produced using conventional technologies. However, we find that the mechanism for the trapping of particles

is not the transverse ponderomotive force of the injection pulse, but rather the interaction of the particles with the two plasma wakes. The 2D and 3D simulations are in good agreement with each other and give therefore confidence in the results obtained from 2D simulations.

These results open up a number of possibilities for future investigations, both to obtain analytical models and to consider other injection schemes and geometries. One important goal of future research would be to find an analytical model of the process that is able to predict the results seen in the simulations. This could then be used to determine fundamental limits on beam number and emittance, as well as to optimize parameters to achieve these limits. Another research direction would be the use of more realistic laser parameters in 2D and 3D simulations.

Chapter 6

Long Wavelength Hosing of Laser Beams

6.1 Introduction

Understanding the evolution of short-pulse high-intensity lasers as they propagate through underdense plasmas is essential for the successful development of some plasma accelerator [1] and radiation schemes [68], as well as for the fast ignitor fusion concept [69]. As a result, there has been much research during the past few years on short-pulse laser-plasma interactions. This work has resulted in the identification of numerous self-modulated processes, e.g., relativistic self-focusing [36], ponderomotive blowout/cavitation, and Raman forward scattering (RFS) instabilities [10, 40, 70], including envelope self-modulation [71] and hosing [72, 73]. While the work of the last few years has led to the determination of the spatial-temporal growth behavior of the above processes [10, 40, 70, 71, 72, 73], it is not clear which, if any, of these processes dominate the evolution of the laser after these processes have saturated.

In this chapter of this dissertation, we use the particle-in-cell (PIC) model PEGASUS [48, 17] to investigate the final nonlinear state of short-pulse lasers after they have propagated through a few Rayleigh lengths of plasma. We find over a wide parameter space that the laser's evolution follows a common sequence of events. Furthermore, we find that the final state of the laser is dominated by a new long wavelength hosing instability. We present a variational principle analysis which provides the growth rate for the well known Raman type hosing instability [72, 73], but which clearly identifies a long wavelength hosing (LWH) regime. At higher densities, we find ion motion to be important. Last, we illustrate through PIC simulations that a consequence of LWH is for the self-trapped electrons [74] to be displaced sideways.

6.2 Motivation

We begin by presenting results from a PEGASUS simulation in which a $600\text{fs}/\mu\text{m}$ laser is focused with a peak intensity of $5 \times 10^{18} \text{ W/cm}^2$ and a spot size of $20\mu\text{m}$ onto the edge of a 1.4×10^{19} plasma slab. For these parameters, $\omega_0/\omega_p = 8.5$, $c/\omega_p=1.36\mu\text{m}$, the Rayleigh length is $x_R=1.2 \text{ mm}$, the peak normalized vector potential $a_0 = eA_0/mc^2 = 2$, and the ratio of laser power to the critical power for relativistic focusing is [36] $P/P_c = a^2 (k_p w)^2 / 32 = 27$. In the simulation 1.2×10^7 electrons are followed on a 8192×256 x-y cartesian grid, while the ions are modeled as a smooth neutralizing background.

In Fig. 6.1, we show a sequence of four color contour plots of the laser's electric field with a common color map. The four snapshots correspond to when the laser initially impinges on the plasma and when the head of the laser has penetrated .57 mm, .83 mm, and 1.82 mm into the plasma respectively. After only .57 mm, i.e., .5

x_R , the head of the pulse has been depleted from Raman scattering while the back of the pulse has strongly self-focused. Details of this have been reported elsewhere [48, 17].

Eventually , as seen in Fig. 6.1c, the middle of the pulse is modulated from Raman forward scattering, while the back of the pulse expands and breaks up into two major filaments in which both Raman forward scattering and conventional hosing are occurring. However, later in time the pulse reaches a “final” nonlinear state where the back of the pulse has refocused into a major filament (with two weaker filaments surrounding it) whose average position in the y direction, y_a oscillates about the original laser axis. The intensity contours closest to the front of the pulse alternate above and below the original laser axis at a wavelength of roughly twice the plasma wavelength, $\lambda_p = 2\pi c/\omega_p$. At positions further back, y_a is modulated at a longer wavelength - between 5-10 λ_p . This hosing behavior at wavelengths longer than λ_p , i.e., LWH, was not discussed in the earlier theoretical analysis [72, 73]. We emphasize that the nonlinear evolution of the pulse is also influenced by wavebreaking and intense plasma heating [75, 76].

6.3 Theoretical Approach

In order to present a possible explanation for LWH, we present here a variational principle approach, developed by B.J. Duda and W.B. Mori [77], to describe the evolution of short-pulse laser interacting with their self-consistent wakes. The standard equations for describing short-pulse lasers, which include the lowest order relativistic corrections and assume a cold plasma, are now well established to be:

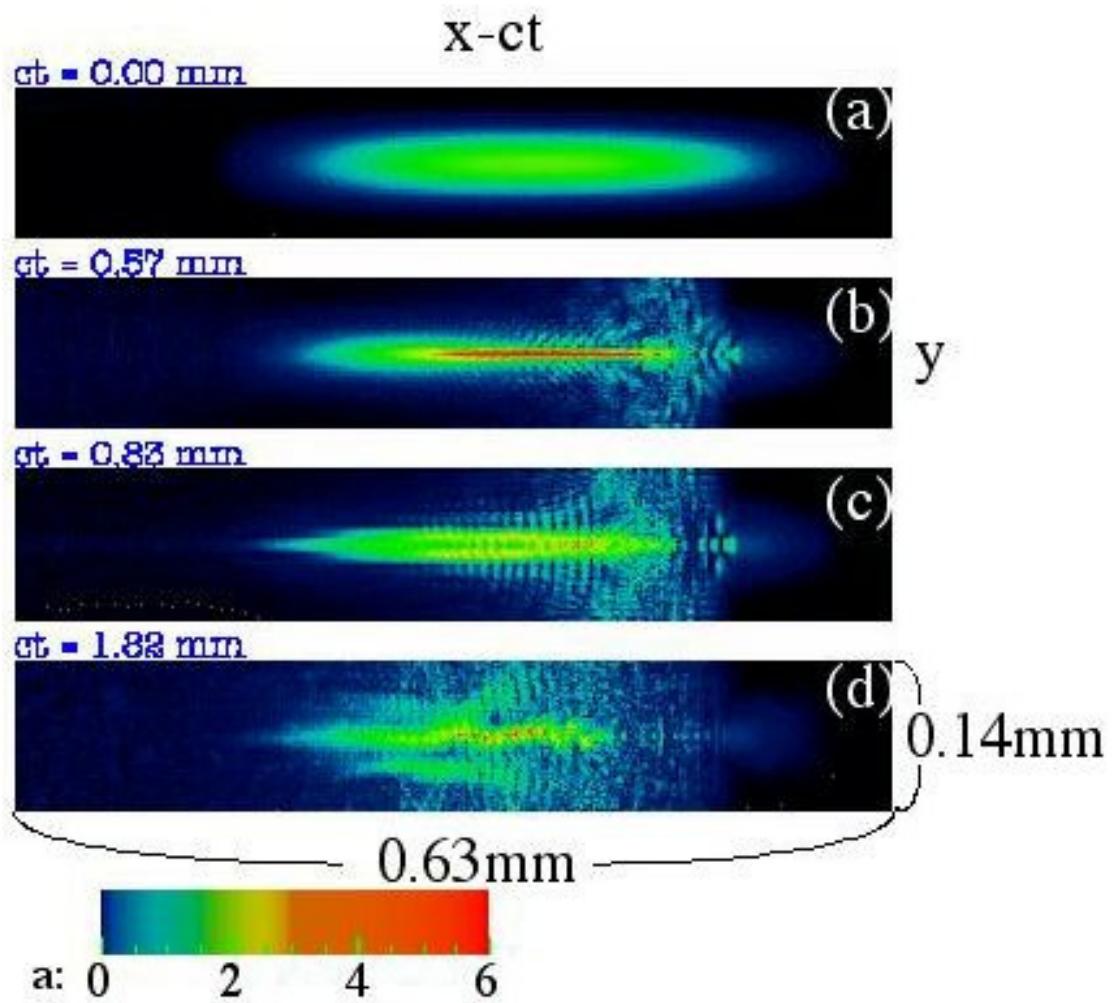


Figure 6.1: A sequence of color contours of the laser's electric field in units of $eE/(m\omega_0) \simeq a$. The results are the same from the same simulation.

$$\left(\nabla_{\perp}^2 - \frac{2}{c} \frac{\partial^2}{\partial \psi \partial \tau} - 2ik_0 \frac{\partial}{\partial \tau} \right) a = \frac{\omega_p^2}{c^2} (1 - \phi) a \quad (6.1)$$

$$\left(\frac{\partial^2}{\partial \psi^2} + \omega_p^2 \right) \phi = \omega_p^2 \frac{|a|^2}{4}. \quad (6.2)$$

where a is the normalized envelope for the complex vector potential of the laser, $eA/mc^2 = (a/2) \exp[-i\omega_0 \psi] + c.c.$, ϕ is the scalar potential of the plasma, and $\psi = t - x/c$, $\tau = x/c$ are convenient variables for describing short-pulse lasers.

In the variational method a Lagrangian density, \mathcal{L} , needs to be found for which the Euler-Lagrange equations, obtained by varying the action, $S = \int dx_{\perp} d\psi d\tau \mathcal{L}$, recover Eq. (6.1) and Eq. (6.2). We find such an \mathcal{L} to be:

$$\begin{aligned} \mathcal{L}(a, a^*, \phi) &= \vec{\nabla}_{\perp} a \cdot \vec{\nabla}_{\perp} a^* - ik_0 (a \partial_{\tau} a^* - a^* \partial_{\tau} a) \\ &\quad - \frac{2}{c} (\partial_{\psi} \phi)^2 + 2 \frac{\omega_p^2}{c^2} \phi^2 - \frac{\omega_p^2}{c^2} (\phi - 1) |a|^2 \end{aligned}$$

where we have dropped the so-called dispersive terms, i.e., those which give the mixed derivative term on the left-hand side of Eq. (6.1) [77]. Dropping the dispersive terms leads to conservation of power, i.e., $\partial \tau \int dx_{\perp} |a|^2 = 0$. Anderson and Bonnedal [78] used the variational approach to study only self-focusing, which precludes any coupling to the plasma wave wake and hence their \mathcal{L} depends upon a and a^* only.

In the variational method, the complexity of the system is reduced by substituting trial functions for a and ϕ into the action and performing the dx_{\perp} integration. To consider lossing, we assume a trial function for a of the form $a = \mathcal{A} e^{i\chi} e^{ik_y(y-y_a)} e^{-2[(y-y_a)^2+z^2]/w^2}$ for ϕ of the form $\phi = \Phi e^{-2[(y-y_{\phi})^2+z^2]/w^2}$ where the

parameters \mathcal{A} , Φ , χ , k_y , y_a , and y_ϕ are treated as functions of (ψ, τ) . The spot size, w , is taken to be a constant which we allow to be the same for both a and ϕ . The “centroid” variables y_a and y_ϕ measure the distance that the center of the laser and its wake are displaced from the original axis. Performing the dx_\perp integration yields a reduced action which is a functional of the variational parameters, i.e., $\bar{S}(\mathcal{A}, \chi, \Phi, \alpha, k_y, y_a, y_\phi) = \int d\psi d\tau \bar{\mathcal{L}}$. Varying \bar{S} with respect to χ yields the power conservation law, $\partial_\tau P = \partial_\tau (\mathcal{A}^2 w^2) = 0$. Variations with respect to the functions α and k_y give the relationships $\alpha = -(k_0/4) \partial_\tau (w^2)$, and $k_y = -k_0^2 \partial_\tau y_a$, which can be substituted back into $\bar{\mathcal{L}}$ to yield the following reduced form of $\bar{\mathcal{L}}$, $\bar{\mathcal{L}}(\Phi, y_a, y_\phi)$:

$$\begin{aligned} \mathcal{L}(\Phi, y_a, y_\phi) = & -\frac{k_0^2}{4} P (\partial_\tau y_a)^2 \\ & + \frac{k_p^2}{2} \left(w^2 \Phi^2 - \frac{P\Phi}{2} e^{-\frac{(y_a-y_\phi)^2}{w^2}} \right) - \frac{1}{c^2} \left[\frac{w^2}{2} (\partial_\psi \Phi)^2 + \Phi^2 (\partial_\psi y_\phi)^2 \right]. \end{aligned}$$

Next we linearize the Euler-Lagrange equations of $\bar{\mathcal{L}}$ about a solution in which $y_{a0} = y_{\phi0} = 0$, and $\Phi_0 = a_0^2/4$, giving the coupled equations for y_a and y_ϕ :

$$\partial_\tau^2 y_a + c^2 g \frac{P}{P_c} \frac{1}{x_R^2} y_a = c^2 g \frac{P}{P_c} \frac{1}{x_R^2} y_\phi \quad (6.3)$$

$$\partial_\psi^2 y_\phi + \omega_p^2 y_\phi = \omega_p^2 y_a, \quad (6.4)$$

where $P/P_c = \mathcal{A}^2 (k_p w)^2 / 32$, g is a geometric factor which is 1 in cylindrical and $2^{-3/2}$ in slab geometry (used in the simulations), and $x_R = k_0 w^2 / 2$ is the Rayleigh length for the equilibrium laser profile. Note that these equations are identical in form to those which describe hosing of electron beams in the ion focused regime [79],

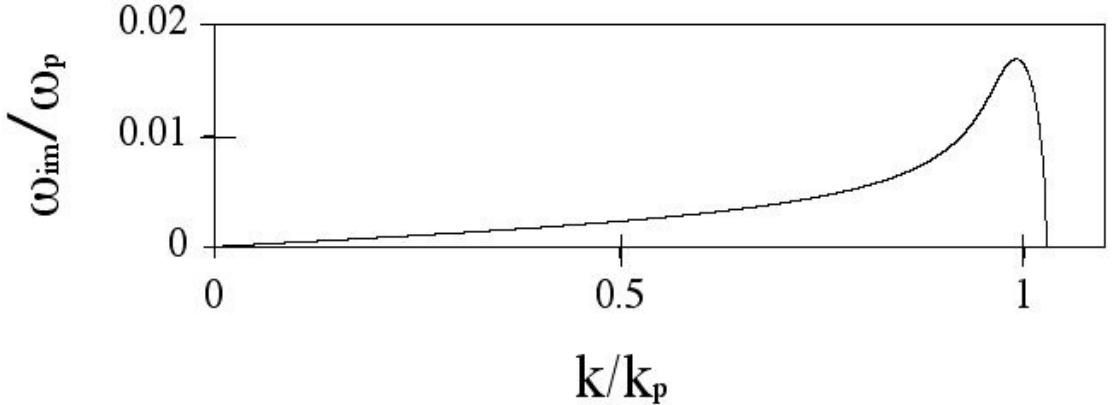


Figure 6.2: The growth rate for hosing vs. wavenumber for $\tilde{x}_R = 256$.

and they reproduce Eq. (5) in Ref. [72].

To discuss the growth rate and range of unstable wavelengths for hosing, we obtain a dispersion relation in the lab frame by using the transformations $\partial_\tau \rightarrow \partial_t + \partial_x$ and $\partial_\psi \rightarrow \partial_t$ and substituting solutions of the form $\exp(i(kx - \omega t))$ into Eq. (6.3)&(6.3), yielding $\tilde{\omega}^2 (\tilde{\omega} - \tilde{k})^2 - (\tilde{\omega}^2 g(P/P_c) / \tilde{x}_R^2) - (\tilde{\omega} - \tilde{k})^2 = 0$, where $\tilde{\omega} \equiv \omega/\omega_p$, $\tilde{k} \equiv k/k_p$, and $\tilde{x}_R \equiv k_p x_R$. In Fig. 6.2 we plot the growth rate, i.e., the imaginary part of the $\tilde{\omega}$, vs. real \tilde{k} for $P/P_c=1$, i.e., a matched beam. This confirms that the peak growth rate occurs for $\tilde{k} \sim 1$, i.e., $k \sim k_p$. This region of unstable growth is related to Raman forward scattering (RFS), since a plasma wave is being excited, and it is the regime discussed in Refs. [72, 73].

However, Fig. 6.2 also makes clear that the range of unstable wave numbers extends continuously down to $\tilde{k} = 0$. This long wavelength regime has heretofore never been discussed. This regime could have been obtained immediately if $y_\phi = y_a$ was assumed in the trial functions, which forces the centroids for ϕ and a to be in phase. In this limit, the plasma response, ϕ is due almost entirely to relativistic mass corrections, i.e., no plasma waves are excited. Therefore, this long wavelength regime

is the whole beam analog to relativistic self-phase modulation (RSPM) [35]. LWH is therefore physically distinct from conventional hosing in the same way that RSPM is distinct from RFS.

6.4 Simulation Results

The spatial-temporal growth for the conventional and LWH regimes also differ. In the RFS regime,¹ for $\tilde{\omega}$ near k_p the asymptotic spatial-temporal growth for hosing is given by [72, 73] $y_{aor\phi} \sim \exp \left[\left(3^{3/2}/4 \right) [g(P/P_c)\omega_p\psi]^{1/3} (\tau/\tau_R)^{2/3} \right]$. In the LWH regime, where the inequality $\partial_\psi^2 \ll \omega_p^2$ holds, Eq. (6.4) leads to $y_\phi \cong y_a/(1 - k^2)$. Substituting this relationship into Eq. (6.3) gives the spatial-temporal growth $y_{aor\phi} \sim \exp \left[(gP/P_c)^{1/2} (k/k_p)^{1/2} (\tau/\tau_R) \right]$. These expressions are only valid under the ideal conditions of cold plasmas, weakly relativistic pumps, and matched beams.

However, for current experimental parameters [80, 81, 82, 83, 84, 85] the conditions are far from ideal. Therefore, to accurately determine the relative importance of the various regimes for hosing with respect to other self-modulation processes, we next present additional results from fully nonlinear PIC simulations. In Fig. 6.3 we show color contour plots of the laser's electric field in units of $eE/(mc\omega_0) \approx a$ to illustrate the “final” nonlinear state of short-pulse laser from four different simulations. In each case 600fs laser pulse is focused to the edge of a uniform, preformed plasma slab and the ions are a fixed neutralizing background. It is clear that for each simulation the “final” state shows strong self-focusing and a dominant LWH component. In Fig. 6.3a, the laser's electric field is shown after a propagation distance of $1.8mm = 6x_R$ from a simulation with parameters identical to those in Fig. 6.1 except $w_0 = 10\mu m$ instead of $20\mu m$. The dominant hosing wavelength is similar to that in Fig. 6.1d but the

amplitude of the centroid seems larger and the instability seems to have saturated. The spatial-temporal theory predicts that the number of e-foldings for LWH scales as $1/w_0$ for otherwise fixed parameters. This scaling is consistent with the observation from Figs. 6.3a) and Fig. 6.1d) that LWH is stronger when $w_0 = 10\mu m$ compared to when $w_0 = 20\mu m$. For the parameters of this simulation, $P/P_c \simeq 6.75$ and $k/k_0 \simeq 10$, the spatial-temporal theory predicts ~ 9 e-foldings of LWH growth, using the focused value of the spot size ($w=5.6\mu m$), and the fact that a^2w is conserved in slab geometry.

The importance of LWH is further illustrated in Fig. 6.3b), which shows results from a simulation which followed 10^8 particles on a 16384×1024 grid. The plasma density was increased to $10^{20} cm^{-3}$, i.e., $\omega_0\omega_p = 3.3$, the laser intensity was lowered to $1.25 \times 10^{18} W/cm^2$, i.e., a_0 , and the spot size was decreased to $6\mu m$, i.e., $k_p w_0 = 11.3$. Once again, after only a few ($480\mu m \simeq 4x_R$) Rayleigh lengths of propagation the laser has strongly self-focused and a LWH mode is dominant. The dominant wavelength is $\sim 15\text{-}30 k_p$ in this case. Using the self-focused spot size, the spatial-temporal theory predicts $\sim 5\text{-}8$ e-foldings of LWH.

In each simulation, there is little or no evidence of the conventional (RFS) type of hosing, except for its presence in the filaments of Fig. 6.1c). However, the spatial-temporal theory predicts many e-foldings of growth. Furthermore, we have independently excited both conventional and long wavelength hosing in smaller test case simulations by adding large fictitious hosing noise sources. Therefore, the lack of RFS hosing is due to nonlinear effects. There are several possible nonlinear explanations. Due to its lower initial noise source, hosing generally occurs after the beam has strongly self-modulated from RFS and self-focusing. The occurrence of RFS divides the beam into beamlets spaced at λ_p (this is seen in Figs. 6.1b) and Fig. 6.1c). When hosing occurs as seen in Fig. 6.1d), it appears to first displace one beamlet upward

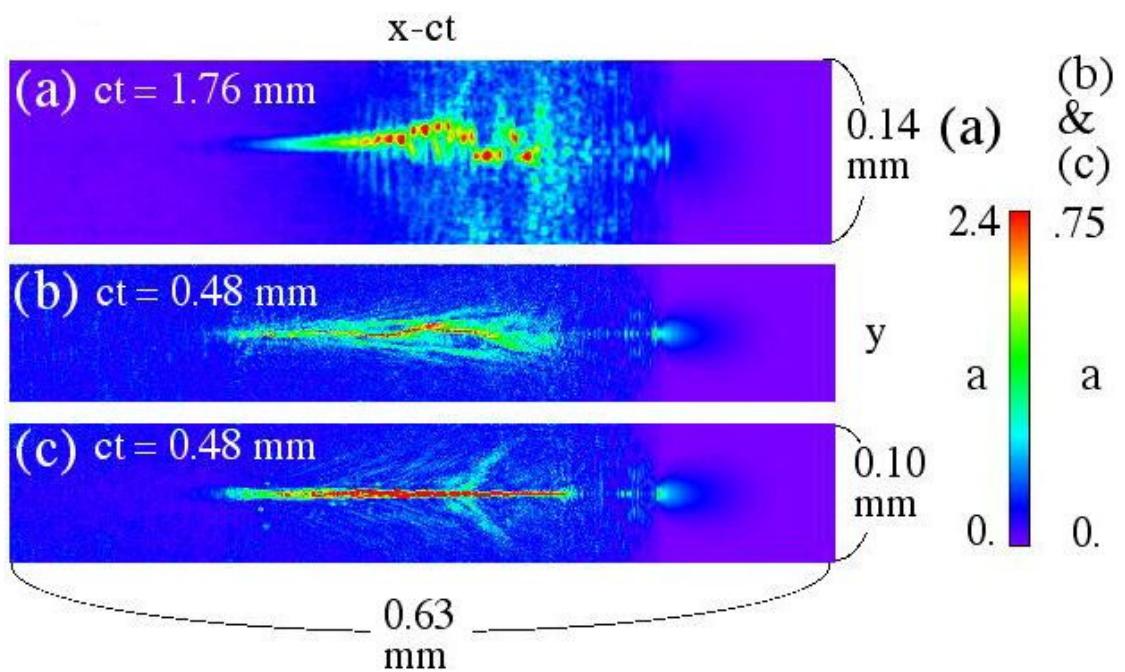


Figure 6.3: Color contours of the laser's electric field in units of $eE / (mc\omega_0) \simeq a$ to show further evidence for long wavelength hosing. The results are from three different simulations.

and the next beamlet downward. This results in a hosing wavelength of $2 \lambda_p$, and as the laser continues to evolve, even longer wavelength modes dominate. Therefore, it appears that hosing behaves differently when other instabilities such as RFS have already grown to saturated levels. Another explanation for the lack of RFS hosing is that the plasma has been strongly heated by the time hosing occurs. RFS hosing involves the excitation of a plasma wave, which can be strongly damped at high temperatures, thereby causing a suppression of RFS hosing.

In regards to the fast ignitor, where longer pulses and higher densities are important (particularly for higher densities), the frequency of the hosing, $\omega = ck$, can be lower or on the same order as the ion plasma period, $\omega_{pi} = 4\pi e^2 n_0 / m_i$. In this case, the ion dynamics cannot be ignored. In Fig. 6.3c), we show results from an identical simulation to that shown in Fig. 6.3b) except mobile hydrogen-like ions were used. The difference between the two cases is dramatic. The ion motion appears to stabilize the hosing (at least for the duration of the simulation). On the other hand, we note that in a simulation with ten times higher intensity, i.e., $a_0 = 3$, ion dynamics did not stabilize hosing. Instead, it appeared to cause the beam to self-focus and filament differently with LWH still occurring in the individual filaments. The wavelength for hosing was shorter than $2\pi c/\omega_{pi}$ in this case. So it appears that ion dynamics can stabilize hosing when $\lambda_{hosing} \gtrsim 2\pi c/\omega_{pi}$. We also note that LWH can occur for densities above quarter critical where RFS cannot, because no plasma wave is excited. Preliminary evidence of a LWH effect has already been observed in simulations for the density regime [86]. Therefore, LWH could be important for the fast ignitor fusion concept.

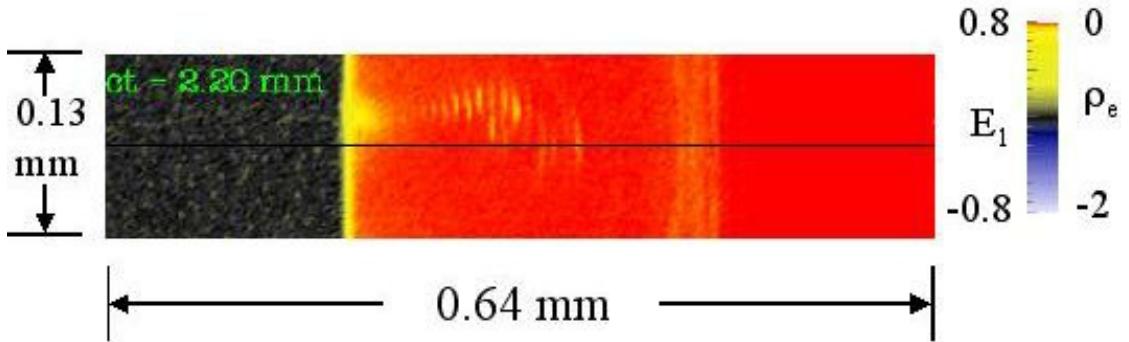


Figure 6.4: Color contour of electron density showing self-trapped electrons exiting the plasma. The results are from the same simulation as Fig. 6.3a).

6.5 Conclusion

We have shown analytically that a LWH regime exists and shown for the first time evidence of any type of hosing in self-consistent PIC simulations. These simulations show that the LWH eventually dominates over a wide parameter regime due to nonlinear effects. Furthermore, we note that LWH might have important consequences for the electron spectra generated in self-trapped acceleration experiments [80, 81, 82, 83, 84, 85]. This is illustrated in Fig. 6.4 where a color contour plot of the plasma density is shown as the self-trapped electrons exit into a vacuum region. The results are from the simulation corresponding to Fig. 6.3a). The black line is drawn in the middle for reference. The electrons are clearly exiting the plasma off axis by a distance $\sim 10\mu m$, and their pattern corresponds to the laser profile in Fig. 6.3a). When the plasma slab was shortened to 1mm, no hosing was seen to occur, and the accelerated electrons were not displaced [74]. In addition we believe that LWH will be important when lasers propagate in higher density plasmas above $n_c/4$. This could have important consequences to the fast ignitor concept.

Chapter 7

LWFA in a Parabolic Channel

7.1 Introduction

The conceptually simplest plasma based acceleration concept that uses a laser pulse as the drive beam is the laser wakefield accelerator(LWFA). The problem of the LWFA concept is that in its original form it requires a very powerful laser to produce particles with a significant energy gain. As an example we can take a laser with a wavelength of $\lambda_L = 1\mu m$ propagating in a plasma with a density of $n_p = 10^{17} cm^{-3}$. If the excited wake has an amplitude of $eE_{max} = mc\omega_p \simeq 31 GeV/m$ then the dephasing limited maximum energy gain for a particle in this plasma according to Eq. (2.24) is $\Delta W_{dephasing} \simeq 11 GeV$. However, in order to get an energy gain of even 1GeV we find that because of the diffraction limit given by Eq. (2.24) we need a laser power of 150TW. While such lasers are technologically feasible, they are still not readily available.

This power required by a LWFA can be reduced if the diffraction of the laser pulse can be avoided by optically guiding the laser. Several possibilities of guiding have

been theoretically investigated [12]. In this chapter, we present simulation results on LWFA acceleration when the laser is guided by a parabolic plasma channel. For a laser guided by a plasma channel the ideal cross section is determined by channel properties. The Rayleigh length does not have to be taken into account anymore. The cross section of the laser can therefore be smaller than for a laser in a homogeneous plasma and the laser will require less power. Since a full self-consistent theory of a LWFA in a parabolic plasma channel has so far not been developed it is a research area where computer simulations are the best way of gaining a better understanding. That is, the phase velocity of the wake and the laser's intensity will evolve as the wake is excited. In this chapter we study the self-consistent acceleration and excitation process for a particular example.

7.2 Simulation Setup

The simulation results presented in this chapter are based on a simulation with parameters close to the matched beam situation explained in section 2.2. The simulation uses a $1\mu m$ laser with a spotsize of $w_0 = 6.4\mu m$ and a length of $\tau_{FWHM} = 19 \times 10^{-15}s$. The laser has a peak intensity of $I_{peak} = 4 \times 10^{18}W/cm^2$, and therefore a power of $P = 2.6TW$ and a total energy of $E = 50mJ$. The Rayleigh length for this laser pulse is $z_R \approx 128\mu m$. The channel as described by Eq. (2.9) has the parameters $n_0 = 2.79 \times 10^{18}cm^{-3}$, $r_0 = w_0$, $\Delta n = 1.32$, and $\Delta n_c = 3.70 \times 10^{18}cm^{-3}$. The simulation actually uses a piecewise linear profile that deviates up to about 10% from the parabolic profile shape. The higher plasma density compared to the example given in the introduction of this chapter was used to reduce the computational requirements for the run. A channel with these parameters should be weakly focusing

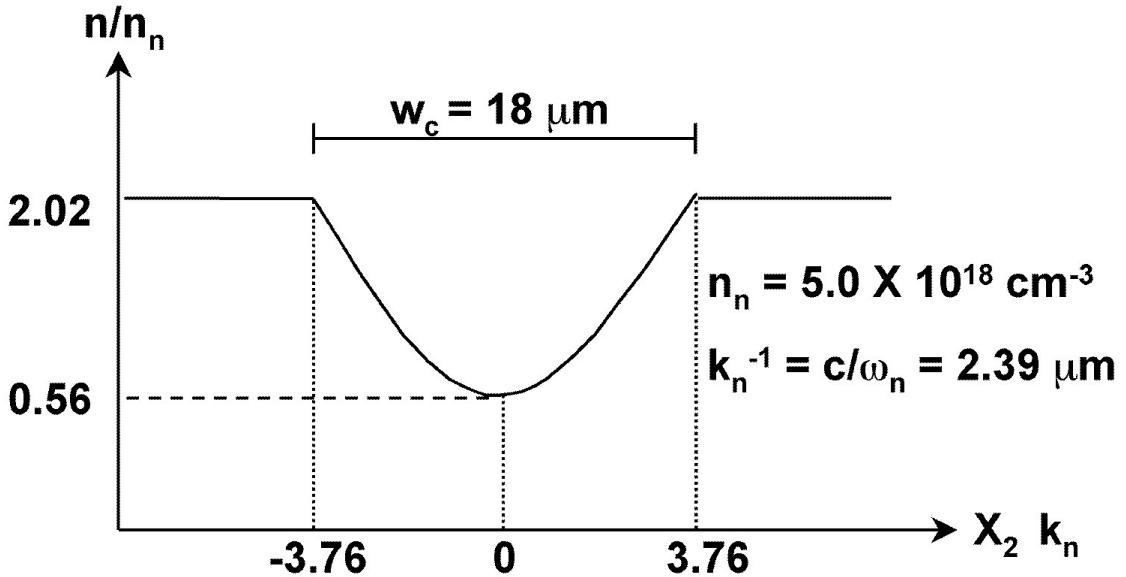


Figure 7.1: The density profile of the plasma channel modeled in the simulation.

for the laser pulse and therefore optically guide it. The geometry of the channel is shown in Fig. 7.1. The total width of the channel is $18\mu m$. In the outer areas of the simulation the parabolic channel density profile is replaced by a constant density.

The simulation was a 2D cartesian simulation using the moving simulation window and was done with the PEGASUS code. The simulation window in normalized units had a size along the propagation direction x_1 of $30.72c/\omega_n$ and a size in the transverse direction x_2 of $15.36c/\omega_n$ with a grid of $N_1 \times N_2 = 1024 \times 128$. Here c is the speed of light and $\omega_n = 1.26 \times 10^{14} s^{-1}$ is the plasma frequency for the normalizing density $n_n = 5 \times 10^{18} cm^{-3}$ used in this simulation. This corresponds to $c/\omega_n = 0.239\mu m$ and therefore the simulation in physical units has a size of $73\mu m \times 37\mu m$. The laser propagated through the plasma for 50000 timesteps of size $dt = 0.0291\omega_n^{-1}$ (corresponding to $3474\mu m$ propagation distance) for a total of $1455\omega_n^{-1}$ ($\approx 3.5mm \approx 27z_R$). Ten particles per cell were used for the plasma. The acceleration of particles

by the wake was tested by uniformly placing 16128 test particles of negligible charge evenly over an area of $15.0\mu m \times 1.15\mu m$ in the center of the generated plasma wake. The plasma was initialized as a cold plasma while the test particles were initialized with a momentum in x_1 of $p_{0,1} = 15m_ec$ and a momentum in x_2 of $p_{0,2} = 0.25m_ec$.

7.3 Simulation Results

The electric field of the laser in the simulation is perpendicular to the plane of the simulation. The envelope of this electric field component, E_3 , and therefore of the laser is shown in Fig. 7.2 at two different times. The first frame is after only about $\frac{1}{2}$ Rayleigh length of propagation into the channel, the second frame is after about 27 Rayleigh length. The most important feature to note is that the channel indeed prevents the diffraction of the laser pulse. Even though the pulse undergoes some evolution during the propagation it is still a Gaussian beam with a spotsize that only decrease slightly during the propagation of $\sim 26\frac{1}{2}z_R$. This is the expected result since the channel is weakly focusing with regard to the initial laser pulse as explained earlier.

In the propagation direction there are several effects on the pulse. The most noticeable one is the falling back of the pulse within the simulation window. Since the simulation window moves with c this can be used to measure the group or energy transport velocity of the laser. The group velocity measured in this way, which is also roughly the phase velocity of the wake seen in Fig. 7.3, is $v_{p,eff} = 0.9965c$ and therefore the pulse and the wake have an effective γ -factor of $\gamma_{eff} = 1/\sqrt{1 - (v_{p,eff}/c)^2} = 11.95$. If we assume that this effective gamma is due to an effective density that the laser pulse experiences while travelling through the channel then this density is $n_{eff} = 1.58 n_n$.

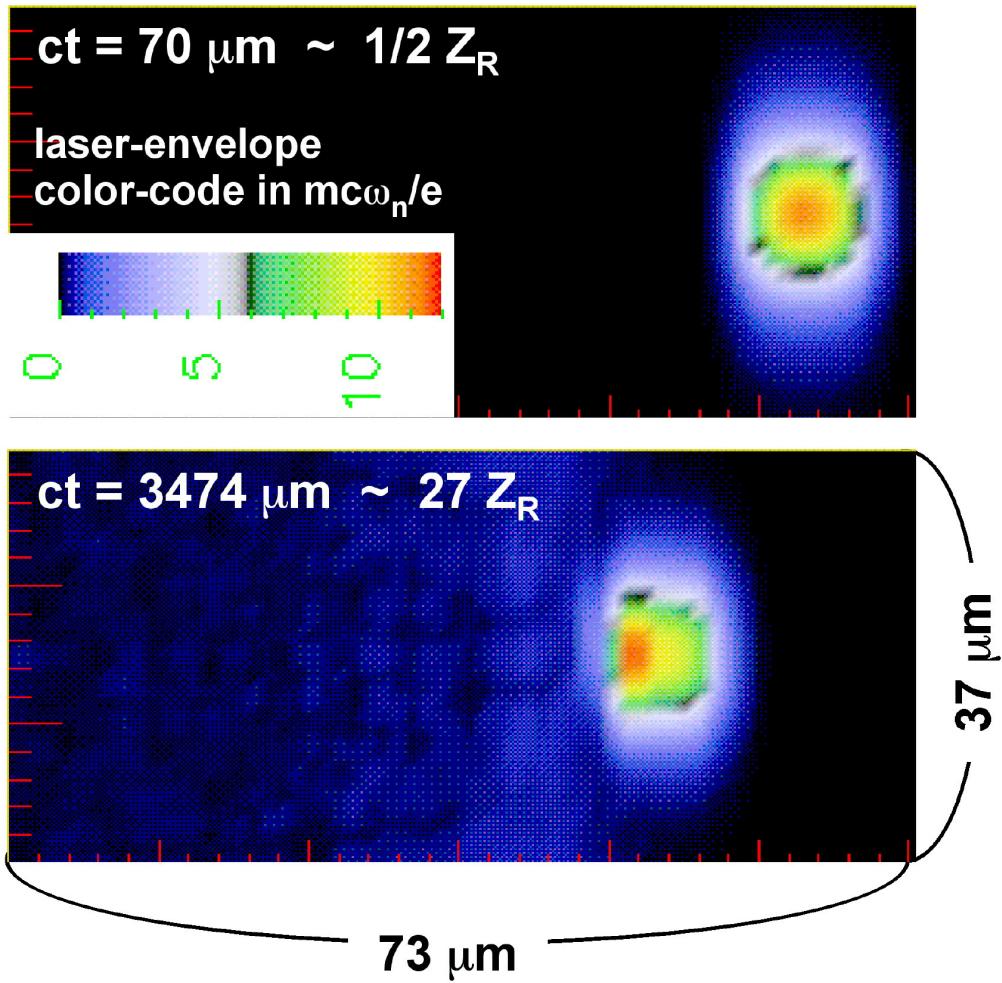
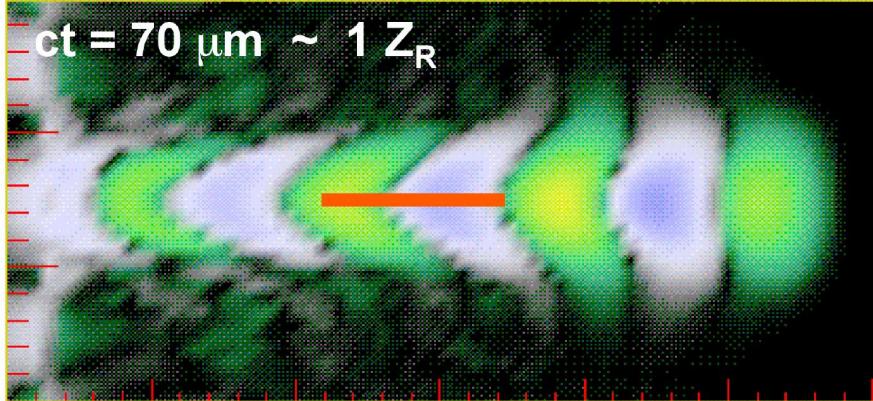


Figure 7.2: The envelope of the matched laser beam after about half a Rayleigh length and after about 27 Rayleigh lengths of laser propagation

Another effect is that the peak intensity of the pulse is moving backwards within the pulse. This is due to the interaction between the laser pulse and the wake generated by the pulse [34, 70]. The laser is focused into a density depression of the generated plasma wave. Over the course of the simulation the peak field of the laser first increases after entering the plasma to about 110% of its original value and then slowly falls off until it has a value of about 106% after propagating $27z_R$.

Fig. 7.3 shows the electric field component parallel to the propagation direction, E_1 , at the same times as Fig. 7.2 shows the laser's envelope. The frames show the plasma wake generated by the laser pulse. The wake is essentially confined to the parabolic channel and shows a slip relative to the simulation window. This slip is equal to the slip seen for the laser. The peak amplitude of the plasma wave increases over time. We concentrate on the third accelerating bucket (measured from the front of the simulation), since this is the eventual location of the test particles. The field evolves from its initial value of $E_{1,peak} = 1.9 \times 10^{10} V/m$ gradually to $E_{1,peak} = 2.6 \times 10^{10} V/m$ (an increase of $\sim 30\%$) by the end of the simulation. This is consistent with the fact that as the laser loses energy by generating the plasma wave it downshifts in frequency [70]. This in turn causes the ponderomotive potential of the laser $\Phi_L = -e \langle \vec{E}^2 \rangle / (2m\omega_L^2)$ and therefore the wakefield amplitude, which is proportional to it, to increase[42]. The wavelength of the plasma wave in the center of the channel can be read from Fig. 7.3 to be $\lambda_p = 17.9 \mu m$. The figure also shows the initial placement of the test particles with regard to the plasma wave and the position of the accelerated test particles after 27 Rayleigh lengths of propagation. The position of the accelerated test particles indicates that the test particles are being accelerated by the third accelerating bucket of the plasma wave instead of the second as it might have been expected from the original position of the test particles. Since all the test

original test particles : —————



accelerated test particles : ●

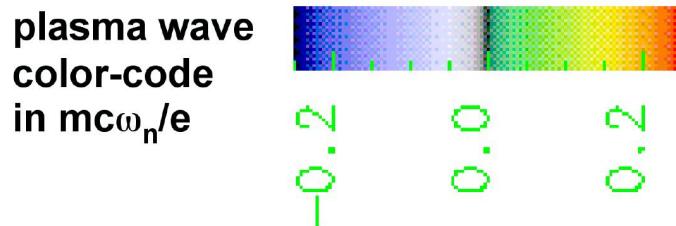
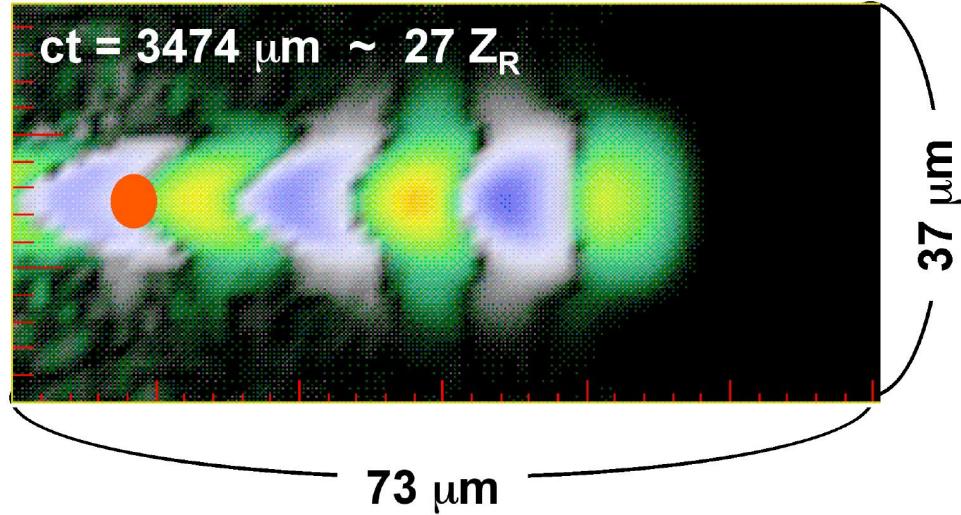


Figure 7.3: The electric field of the plasma wake after about half a Rayleigh length and after about 27 Rayleigh lengths of laser propagation

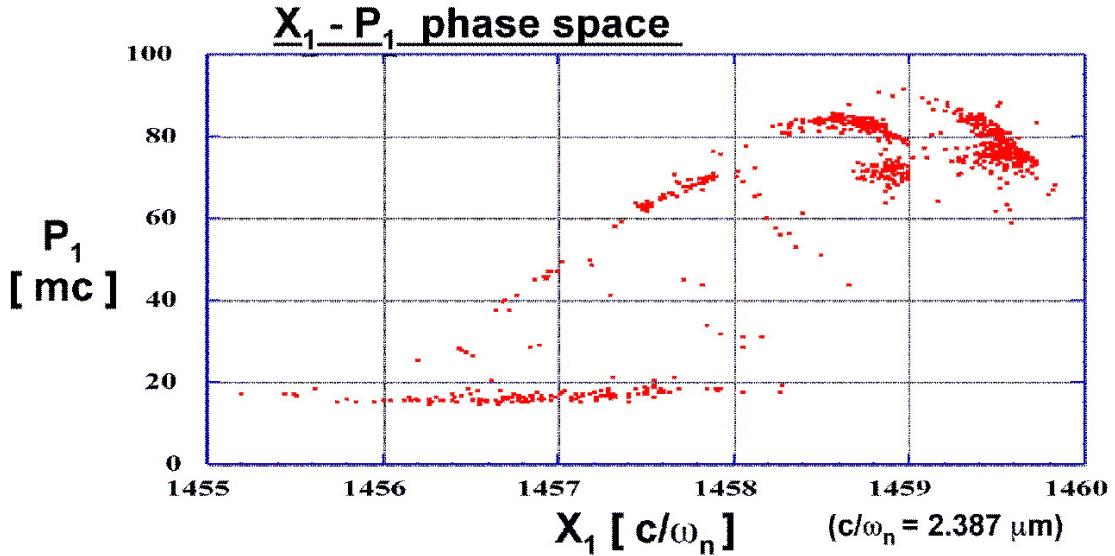


Figure 7.4: The longitudinal, $x_1 p_1$, phase space of the test particles with a momentum $p_1 \geq 15m_e c$ after 27 Rayleigh lengths of propagation. Note that the x_1 axis of the plots only extends over about the last 1/6-th of the simulation window.

particles had an initial γ of ~ 15 those test particles that got eventually accelerated in the third acceleration bucket must have first been decelerated to fall back to the position of third bucket before eventually gaining energy. This initial deceleration is consistent with evolution of test particle data over time.

Fig. 7.4 is the longitudinal phase space of the test particles with a momentum $p_1 \geq 15m_e c$. It shows that some test particles exhibit behavior which is different than expected. A certain number of test particles has essentially the original energy, while a second group has been accelerated to an energy of about $p_1 = 40MeV$, and a third group has energies between these extremes. Since the total number of particles seen in this plot is 821 out of 16128 original test particles, the remaining particles must have been lost due to defocusing fields and deceleration. There are 570 particles with an energy of about $40MeV$. This is about 3% of the total number of particles and

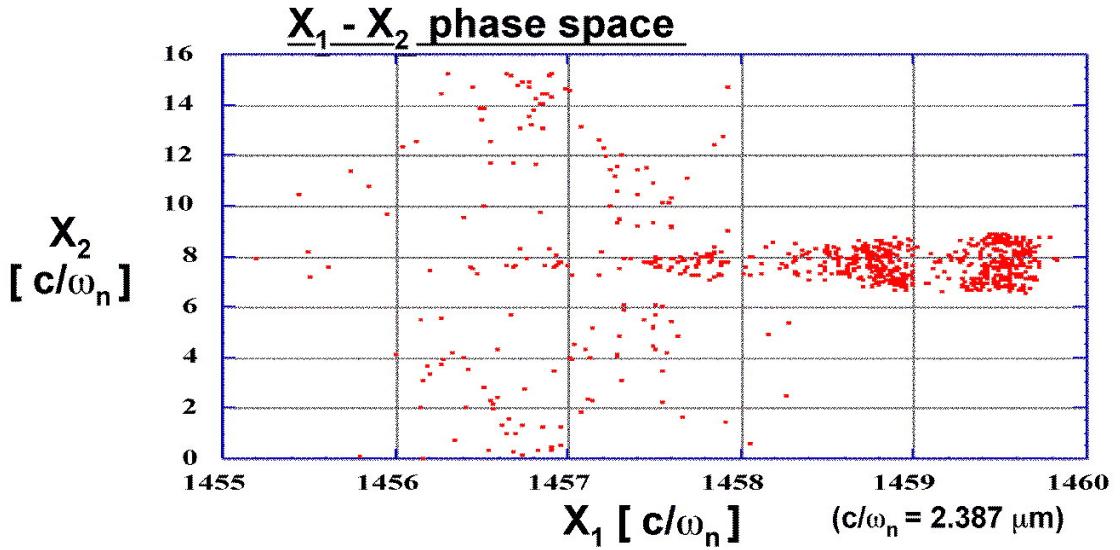


Figure 7.5: The spatial distribution of the test particles with a momentum $p_1 \geq 15m_e c$ after 27 Rayleigh length of propagation. Note that the x_1 axis of the plots only extends over about the last 1/6-th of the simulation window.

gives a first estimate on timing precision required for injecting an external particle bunch into the right phase of this accelerator system. This estimates assumes that all the accelerated particles come from the same area within the original test particle group.

In Fig. 7.5 the distribution of the test particles in real space is shown. The groups of particles with different behavior seen in Fig. 7.4 can be identified with groups of particles in this plot. All the particles with any energy gain are within a radius of about 4μ of the central axis of the laser and the plasma wave while most of the particles without energy gain are further away from the axis out side the area of the plasma channel and the wakefield.

Fig. 7.6 finally shows the $x_1 p_2$ -phase space of the test particles. The most interesting fact to note about this figure is that the particles at higher x_1 which correspond to

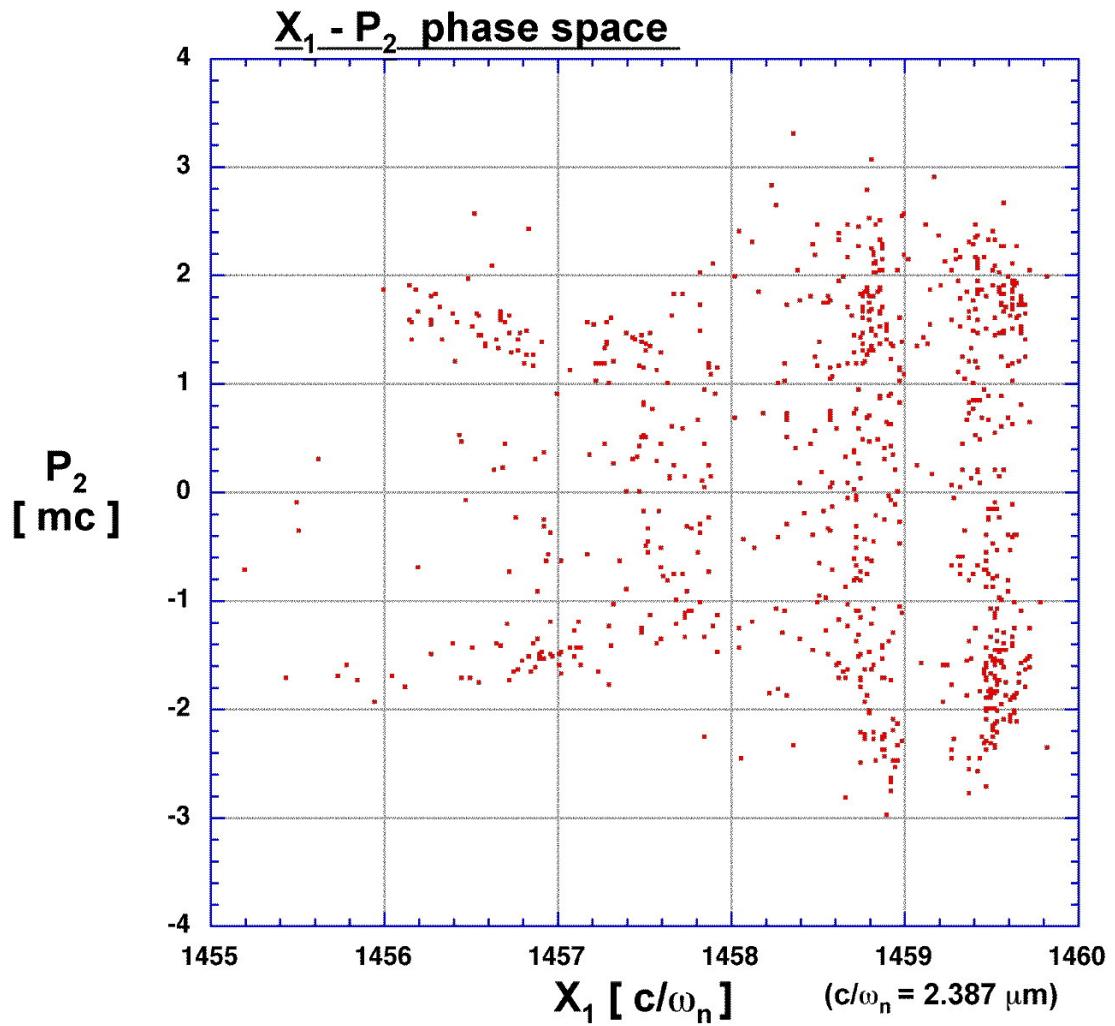


Figure 7.6: The $x_1 p_2$ -phase space of the test particles with a momentum $p_1 \geq 15m_e c$ after 27 Rayleigh length of propagation. Note that the x_1 axis of the plots only extends over about the last 1/6-th of the simulation window.

particles with higher p_1 as can be seen from Fig. 7.4 have on average more transverse momentum than the unaccelerated particles in at lower x_1 . This is consistent with the fact that the accelerated particles are in the focusing region of the plasma wave and are undergoing betatron oscillations. The unaccelerated particles do not undergo any oscillations but have a net transverse momentum that moves them out of the plasma channel as can be seen in Fig. 7.5.

The particle information that was used to generate the Figs. 7.4, 7.5, and 7.6 can be used to directly calculate the parameters that characterize the bunch of 570 particles that were accelerated to about 40MeV . From this we obtain the following results. If not noted differently all widths are calculated as rms-values with respect to the mean value of a quantity. The beam has a momentum of $p_1 = 78.3m_ec$ with a spread of $\Delta p_1 = 5.4m_ec$. The longitudinal spread of the beam is $2 \times \Delta x_1 = 1.94\mu\text{m}$. The total length of the beam from the first to the last particle is $3.59\mu\text{m}$. For the transverse direction the width is $2 \times \Delta x_2 = 2.88\mu\text{m}$ and the momentum spread is $\Delta p_2 = 1.59m_ec$.

These numbers result in an energy spread of $\Delta E/E = 14\%$ and in a normalized emittance $\varepsilon_N = \pi \Delta x_2 \Delta p_2 / m_e c = 2.29 \pi \text{ mm mrad}$. This value can be compared with the original emittance of the test particles and the acceptance of the plasma wake which can be estimated with Eq. (5.3). The initial emittance of the test particles can be calculated by using the total width b of the initial flat distribution profile to calculate the rms-width $\Delta x_{0,2}$ of this profile. We get $\Delta x_{0,2} = b/(2\sqrt{3}) = 1.15\mu/(2\sqrt{3})$. With this the initial normalized emittance becomes $\varepsilon_{0,n} = \pi \Delta x_{0,2} \Delta p_{0,2} / m_e c = 0.08 \pi \text{ mm mrad}$. In order to calculate the acceptance using Eq. (5.3) the normalized peak potential of the plasma wave is required. This can be estimated by assuming a harmonic plasma wave. In this case $\bar{\Phi} = e\Phi/(m_e c^2) = \bar{E}_{1,peak}/\bar{k}_p = 0.107$ with

$\bar{E}_{1,peak} = eE_{1,peak}/(m_e c \omega_n) = 0.105$ and $\bar{k}_p = ck_p/\omega_n = 0.838$. The initial value of $E_{1,peak}$ is used for this estimate. Using these numbers, the estimated value for the acceptance is $A_n = 37\pi mm\ mrad$. The comparison of the initial emittance, the final emittance, and the acceptance indicates that even though there is some emittance growth of the beam it never reaches a matched beam equilibrium. A question that should be investigated is whether this emittance growth is taking place continuously throughout the acceleration or whether it occurs during a specific time period.

The normalized potential calculated above can also be used to calculate the maximum energy gain that the test particles can achieve due to the linear dephasing limit. Using Eq. (2.5) we find (applying the peak potential at the end of the simulation $\bar{\Phi} = 0.14$) that $\Delta W_{max} = 40.9 m_e c^2$. If we include the original energy of the particles then we would expect a final energy of $W_{final} = 28.6 MeV$. This means that the test particles gained more energy than would normally be predicted for a linear 3D plasma wave.

There are two possible reasons for this. One is that the particles are initially not accelerated in the focusing part of the accelerating phase of the plasma wave since acceleration by only the focusing part of the plasma wave was assumed in the calculation of the maximum energy gain above. The possible energy gain over the full accelerating phase of the plasma wave including the defocusing part is $42 MeV$. This is consistent with the final energy of the test particles if we consider the fact that they were first decelerated before being accelerated. The other possibilities is that the wake generated in the plasma channel is nonlinear enough that Eq. (2.5) does not give a good estimate of the maximum energy gain anymore. With the data available from the simulation presented here this can not be decided and is a question for future research.

7.4 Conclusion

This chapter presented the results of a 2D simulation of a LWFA in a parabolic plasma channel. The results suggest that the idea of preventing diffraction of a laser pulse by propagating it through a matched parabolic plasma channel works. Furthermore, the laser in the channel is able to generate a plasma wave useful for acceleration of particles. Test particles accelerated in the plasma wave experience an increase in emittance. Only about 3% of the initial test particles are being accelerated which suggests that for externally injected particles beams the length of the beam and the exact phase will be crucial. The simulations show that the phase velocity of the wake and the energy gain cannot be straightforwardly obtained from the simple 1D theory so this is an area for future research.

Chapter 8

Plasma Wakefield Acceleration in the Blowout Regime

8.1 Introduction

The basic concept of a plasma wakefield accelerator (PWFA) is to accelerate a low current trailing electron bunch by the wakefield generated by a high current driver. If the driving bunch is highly relativistic, then both the accelerating as well as the accelerated bunch are moving with about the speed of light and the accelerated bunch can stay in phase with the accelerating field for distances long enough to gain significant amounts of energy. Motivated by and as part of the preparations for an experiment which is currently being conducted at the Stanford Linear Accelerator Center (SLAC), we have simulated a plasma wakefield accelerator with the expected parameters of this experiment [15].

In this experiment a 30GeV electron beam at SLAC is used to excite a wake of the order 1GeV/m in a $1.4m$ long plasma of density $1 - 2 \times 10^{14}cm^{-3}$. In this

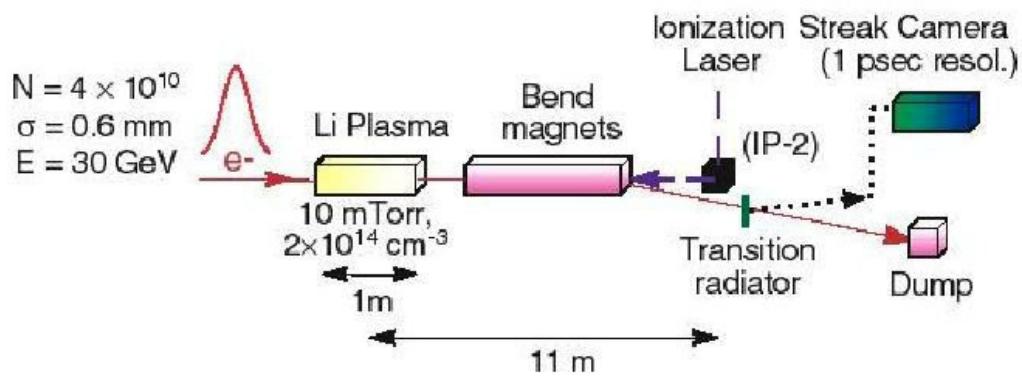


Figure 8.1: The setup of the E-157 experiment at SLAC.

wake the centroid energy of the tail of the beam is expected to increase by several hundred MeV. Since the beam in this experiment is typically much denser than the plasma (e. g., $N = 3.5 - 4 \times 10^{10}$ electrons in a $\sigma_z = 0.6\text{mm}$ bunch length and a spot size of $\sigma_r = 50\mu\text{m}$ corresponding to a beam density $n_b = 1 \times 10^{15}\text{cm}^{-3}$), the PWFA is in the highly non-linear or so-called blowout regime [46]. Fig. 8.1 shows the experimental setup of the experiment. More details about the setup and execution of the experiment can be found in Ref. [87].

The advantages that the blowout regime offers are a high accelerating gradient, a constant accelerating structure with respect to the transverse dimensions, a linear focusing force, and a high transformer ratio. However, in this nonlinear regime neither linear theory nor fluid models are applicable and do not provide an accurate understanding of the physics. Much better insight into the physical processes can be gained by using Particle-In-Cell (PIC) simulations, which allow accurate modeling of highly non-linear processes like the ones occurring here. For these reasons, we conducted PIC simulations to investigate this regime of plasma wakefield acceleration. The simulations were done using both the 2D cylindrically-symmetric and the full 3D system packages in OSIRIS. We have also developed a analytic model which is a bridge between the full particle PIC models and the reduced description PIC models and fluid codes.

8.2 2D Cylindrically-Symmetric Simulations

We carried out simulations for the physical parameters similar to the ones described above, using OSIRIS. The algorithms for the results presented in this section were 2D cylindrically-symmetric and used the moving simulation window to follow the beam

since this limits the simulation domain to the beam and its immediate surroundings rather than the whole propagation distance of the beam. The simulation window in normalized units had a size along the propagation direction, z , of $25c/\omega_p$ and a size in the radial direction, r , of $10c/\omega_p$ with a grid of $N_z \times N_r = 500 \times 200$. Here c is the speed of light and ω_p is the plasma frequency for a given plasma density n_p . We will use a plasma density $n_p = 2.1 \times 10^{14} \text{ cm}^{-3}$, which corresponds to $c/\omega_p = 0.367 \text{ mm}$, throughout this chapter when converting simulation results back into physical units. This means the simulation window corresponds to a size of $9.175 \text{ mm} \times 3.67 \text{ mm}$. The beam propagated through the plasma for 190000 time-steps with $dt = 0.02\omega_p^{-1}$ (corresponding to $18.35 \mu\text{m}$ of propagation distance per timestep) for a total of $3800c/\omega_p$ ($\sim 1.4 \text{ m}$). Nine particles per cell were used for the background plasma and 25 particles per cell for the beam. The beams longitudinal profile was fitted to the experimentally known profile of the SLAC beam [15], which is very close to a longitudinal Gaussian profile of length, $\sigma_z = 0.63 \text{ mm}$, and a transverse Gaussian profile of width, $\sigma_r = 70 \mu\text{m}$. For 3.7×10^{10} electrons this corresponds to a peak density of $7.56 \times 10^{14} \text{ cm}^{-3}$.

Fig. 8.2 to 8.5 give detailed information about the beam and background plasma at several timesteps. Together they present a picture of the development of the beam and plasma over time. The four figure represent data after 0 mm , 191 mm , 396 mm , and 1.4 m . The figures for the times 191 mm , 396 mm are shown because these are the times of the first betatron oscillation minimum and maximum after the beam enters the plasma. The figure is composed of a number of plots that shows seven different aspects of the simulation data. Please note that in these figures the indices 1, 2, and 3 are used instead of z , r , and Θ respectively. The plot in the upper left corner is a colored, rubber-sheet representation of the longitudinal, accelerating electric field. For this visualization the elevation of a surface point as well as its color

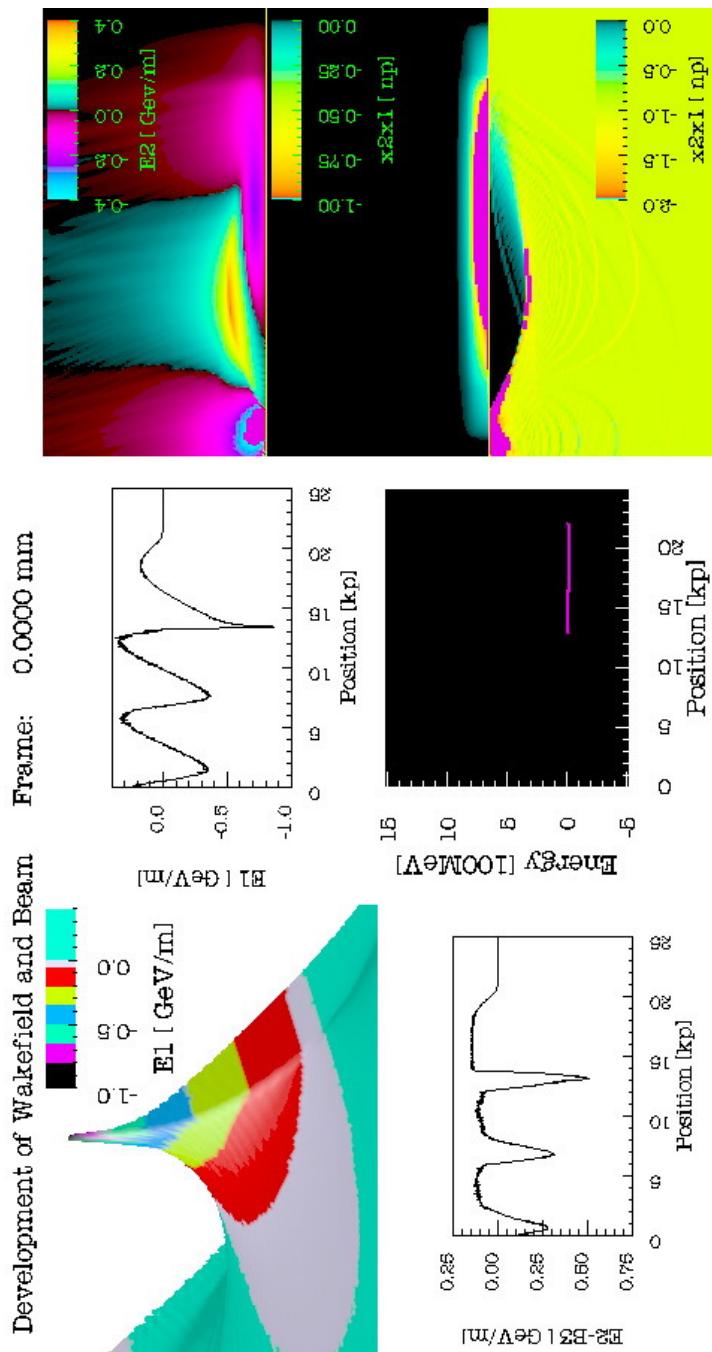


Figure 8.2: The figure shows plots of several quantities at the beginning of the simulations just after the beam fully entered the plasma. See the main text for a detailed explanation of the plotted quantities.

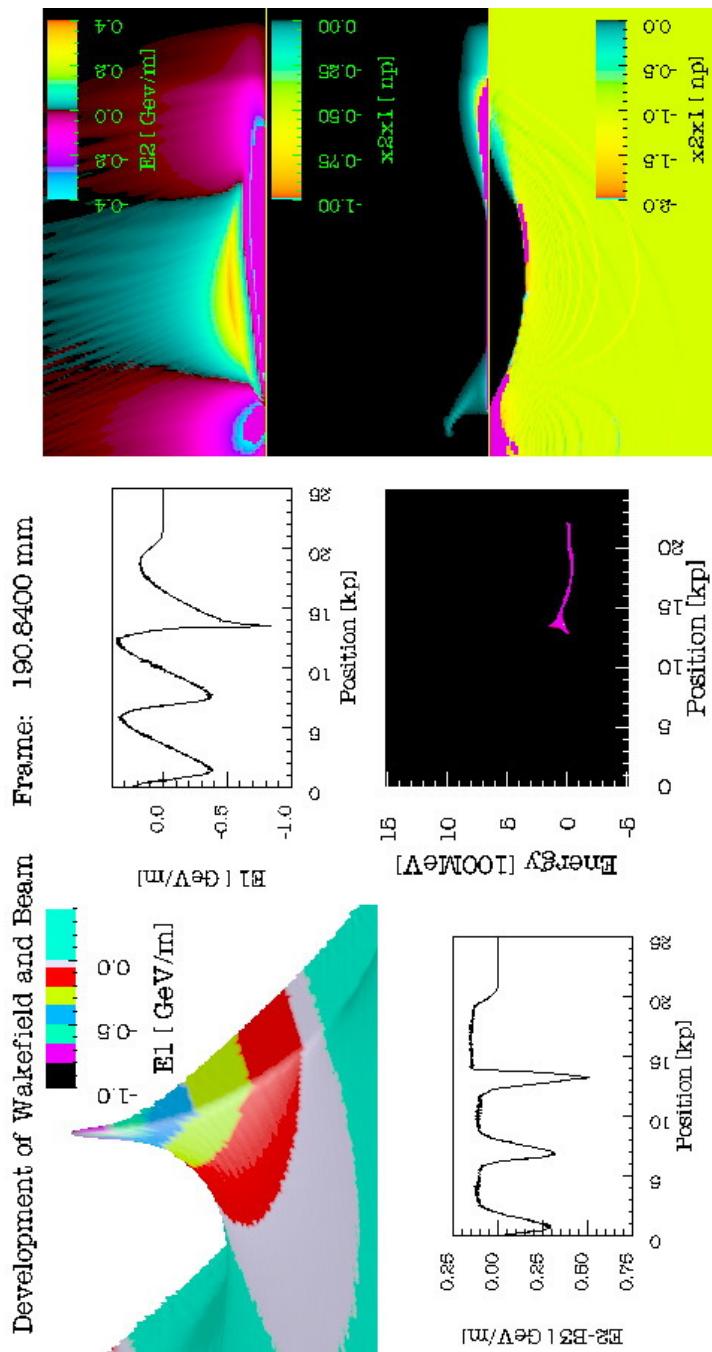


Figure 8.3: The figure shows plots of several quantities at the first minimum of the betatron oscillation of the beam after $\sim 191\text{mm}$ of propagation through the plasma. See the main text for a detailed explanation of the plotted quantities.

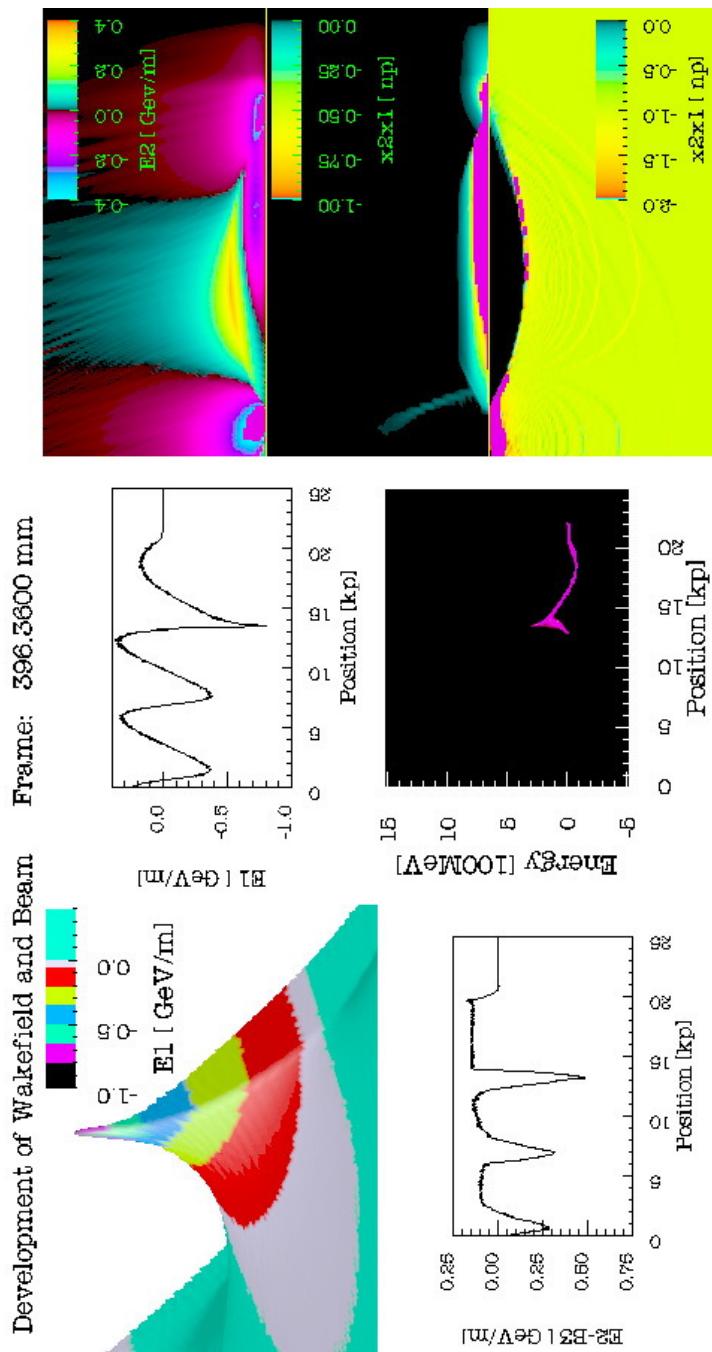


Figure 8.4: The figure shows plots of several quantities at the first maximum of the betatron oscillation of the beam after $\sim 396\text{mm}$ of propagation through the plasma. See the main text for a detailed explanation of the plotted quantities.

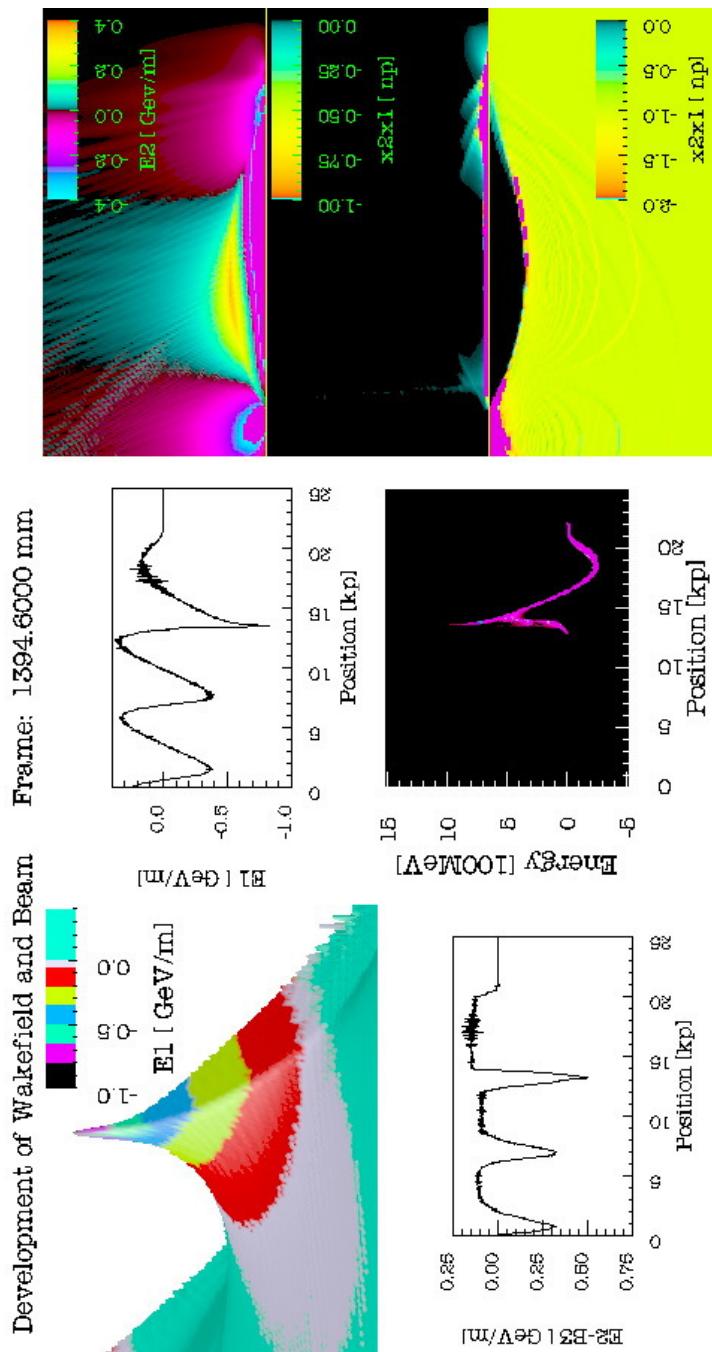


Figure 8.5: The figure shows plots of several quantities at the end of the simulations after $\sim 1.4m$ of propagation through the plasma. See the main text for a detailed explanation of the plotted quantities.

represent the field strengths of the electric field. Note that we chose a perspective for visualizing the rubber-sheet surface so that negative field values of the electric field would be represented by positive values of the surface elevation. This leads to a better visualization of the accelerating region. The sharp edge of the rubber-sheet surface going roughly from the upper left corner to the lower right corner is due to the axial boundary of the 2D cylindrically-symmetric simulation and accordingly r increases starting from this axial boundary towards the lower left corner. Due to the chosen perspective the rubber-sheet does not show the data for the whole simulation. The upper middle plot shows the value of the longitudinal electric field along the axial boundary for its full length of $25c/\omega_p$. The figure in the lower middle below the electric field lineout shows the energy gain and loss of the electron beam as a function of the axial position. The colored areas indicate the parts of this plot where beam electrons are present. Note that the horizontal axis of this plot is precisely aligned with the axis of the field lineout above. The plot in the lower left corner shows the focusing field experienced by the beam electrons, $E_r - B_\Theta$, at a position $73.4\mu m \cong \sigma_r$ off axis. The right column shows three color-plots in the $r - z$ plane. The plots shown (from top to bottom) are the radial electric field E_r , the charge density of the beam, and the charge density of the background plasma. This last plot has been mirrored along the axis to allow for a more direct comparison of the plasma density with the beam density. The horizontal axis for each of the three plots goes from $12.5c/\omega_p$ to $22.5c/\omega_p$ of the simulation window and the vertical axis shows 0 to $5c/\omega_p$ along the radial direction. The field and density values are given by the colorbars in each of the plots. Note that the areas of the plots colored in magenta are areas in which the field or density values are outside the respective color-scales. Since the color-code of the beam charge density plot reaches from 0 to 1, which is the normalized density of the

background plasma, the magenta-colored areas in this plot indicate densities above the background plasma density.

The first fact to note is the lack of change over time in the evolution of the accelerating electric field, and the focusing field. With the exception of the peak accelerating field which fluctuates slightly by about $\pm 0.05\text{GeV/m}$ around a value of about 0.75GeV/m ($\sim \pm 7\%$) and some slight variation in the level of numerical noise, the accelerating electric field essentially does not change over time. This is in strong contrast with the dynamic development of the beam radius (middle plot in the right column) and energy (lower plot in the center column), and the radial electric field (upper plot in the right column). The energy plot shows that every part of the beam except the front part and the very tail gains or loses energy linearly as a function of time. This is consistent with the constant longitudinal field since at an initial energy of about 30GeV the beam electrons experience no significant phase slippage over the time of the simulation.

Two other effects can also be observed. First there is a slight broadening of the front part of the decelerated area of the beam along the energy axis, which means that not all electrons at a given z experience exactly the same decelerating field. Secondly there is a large energy spread of the very back of the beam tail, which splits up into two parts. The first observation can be understood when looking at the background plasma charge density. The plasma charge density plot shows that in the front part of the beam the area of total electron blowout is smaller than in the later parts of the beam, and therefore the radius up to which the focusing force F_r is independent of z is smaller. According to the Panofsky-Wenzel theorem, $\partial F_r/\partial z = \partial F_z/\partial r$, this implies an acceleration gradient that varies along the radial position beyond a small value of r [46]. This can also be noticed for the region of decelerating field that is

visible in the lower right corner of the E_z -rubbersheet plot. The radially flat area increases slightly in width towards the back. The broadening of the front part of the deceleration area of the beam is a result of this non-uniform accelerating field. The energy spread of the tail of the beam can be understood by looking at the narrowing of the accelerating and focusing field profile near the peak-accelerating field. It shows that a part of the tail of the beam, in contrast to the rest of the beam, experiences a strong defocusing force that pushes it radially out of the accelerating field. The blowout of some of the tail-electrons of the beam can also be seen in the development of the beam charge density.

The evolution of the main part of the beam, as seen in the beam charge density plot, is clearly dominated by the betatron oscillation of the beam in the focusing field. The focusing field is mainly due to the ions left in the plasma blowout area, as seen in the plasma charge density plot, since the effects of electric and magnetic fields of the relativistic beam on itself cancel each other almost completely. The linear focusing force in the blowout area results in the same oscillation frequency for all beam electrons in that area. The beam propagates while undergoing betatron oscillations with a wavelength for the spotsize

$$\lambda_{\text{spotsize}} = \lambda_\beta / 2 = \pi \sqrt{\frac{\gamma mc^2}{2\pi e^2 n_0}} \quad (8.1)$$

where λ_β is the betatron wavelength of a single particle. This wavelength follows directly from Eq. (2.29). Measuring this wavelength using the minima of the oscillation of the beam density gives a wavelength $\lambda_{\text{spotsize}} = 40\text{cm}$ as predicted by Eq. (8.1) for the density of the simulation [15].

The dynamics of the front part of the beam is more complex because the blowout

area there is not as wide. This leads to non-harmonic oscillations or so-called aberrations in the focusing force, which leads to phase mixing of the electrons. The oscillation frequency of the beam electrons decreases towards the front. Even though this is not clearly visible from the figures above the full data set of the simulation shows clearly that after the main part of the beam reaches an oscillation minimum this minimum moves forward towards the front of the beam as the electrons there execute betatron oscillations with lower frequencies. This happens while at the same time the main part of the beam starts to expand again. This dynamics at the front of the beam leads to a subtle point. Namely, the focusing field for the beam, $E_r - B_\Theta$, shows an unexpected behavior with time. Initially the focusing force rises slowly over the first one-quarter of the beam, but once the head of the beam begins to pinch the rise becomes steeper. The unexpected behavior results because the transverse profile never relaxes back to the original one. Instead, there is always an axial slice of the beam at the head of the beam that is near a pinch. So on average, the beam density at the front of the beam is always larger than it was at $t = 0$. As a result the occurrence of complete blowout is earlier in the beam and the region of blowout is wider leading to more of the beam undergoing the uniform betatron oscillations than might have been expected. Unlike the beam, the plasma electrons respond predominantly to only E_r . Thus, the blowout of the plasma electrons and their oscillation back onto the axis in the back of the pulse is caused by the total radial electric field that they experience. The figures show that the radial field has two distinct regions. The front, where the plasma electrons are not blown out yet, is dominated by the electric field of the beam; and the back, where the plasma electrons are blown out, is dominated by the radial electric field of the remaining ion charge. The plasma charge density plot shows the effect of this. In the moving window frame, i.e., in the $z - ct$

coordinate, the plasma electrons stream backward past the stationary drive beam. After the radial field force deflects the electrons outward most of them coalesce in a narrow, high density surface layer that lies at the edge of the blowout region. The radius of the blowout region and therefore the radial position of the layer is roughly $0.77c/\omega_p \approx 280\mu m$. This is consistent with the rough theoretical estimate [88].

$$r_{blowout} = 2\sigma_r \sqrt{\frac{n_b}{n_p}} \quad (8.2)$$

where n_b is the peak density of a beam. Note that for a long pulse for which the electrons are blown out adiabatically, $r_{blowout} = \sigma_r \sqrt{n_b/n_p}$ [79].

The electrons stream backward within this narrow surface layer and converge on the axis creating a very dense spike and therefore a sharp peak in the accelerating field. (Note that in the lab frame individual electrons are blown out and then return while remaining near their initial z value, but we will use the moving window point of view for its convenience of description). The insensitivity of the accelerating wake field to the dynamic beam development is a consequence of the beam being narrow when compared to the radius at which the surface layer is located. For most of the plasma oscillation, all of the plasma electrons are outside of the beam so that from Gauss law the electrostatic field affecting them is independent of the radius of the charge inside. Thus the betatron pinching of the beam has little effect on the plasma electrons and hence the wake. The slower evolution in the front of the beam does not have any significant effect either since the slight variations in the initial trajectories of electrons become insignificant after the blown out electrons reach the surface layer. The surface layer is shown in Fig. 8.6, where a radial lineout of the plasma charge density at the center of the beam is plotted after 1.4 meters of propagation. The

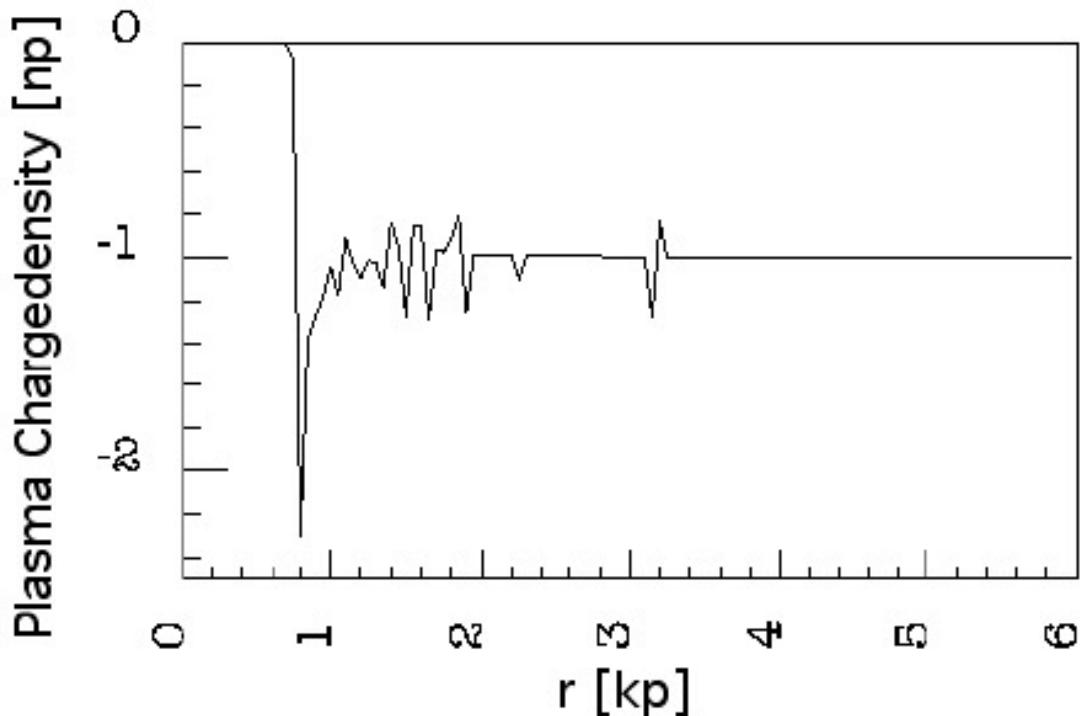


Figure 8.6: A radial lineout of the plasma charge density at the center of the beam after 1.4 meters of propagation.

plasma blowout as well as the surface layer are clearly visible.

Another useful quantity to illustrate the plasma response is the return current carried by the plasma electrons. Fig. 8.7 shows the plasma current, the beam current, and the sum of both. They are calculated from phase space data for each 0.12 pico-second bin of the moving simulation window. In the first half of the beam the plasma current is smaller than the beam current but increases strongly after an initial delay compared to the beam current. At the center of the beam the beam current and the plasma current roughly balance. After this point the plasma current dominates. The main result here is that in the later half of the beam the plasma current completely shields the plasma further away from the beam from the magnetic field generated

by the beam. This is consistent the observation above that the most of the plasma response is confined to a narrow layer outside the blowout region.

Due to the invariance of the accelerating field, the expected energy gain can be predicted with confidence for a specified beam charge and profile. The longitudinal momentum p_z ($\cong \gamma$) vs. ct phase space is shown in Fig. 8.8 and Fig. 8.9 to illustrate the expected acceleration of the beam after $1.4m$ of propagation. The mean, maximum, and minimum energy of the beam are plotted in 0.12 pico-second bins along the length of the beam (Fig. 8.8). This is done in figure Fig. 8.8 for the actual simulation particle data after $1.4m$. Fig. 8.9 by contrast was generated by using the initial particle data propagated for $1.4m$ using the initial fields at the initial positions of the particles. This makes the assumption of a non-evolving field and neglects the betatron oscillation of each particle. The mean, maximum, and minimum energies resulting from these two graphs are very similar for most of the beam. The results only differ at the very end of the beam where Fig. 8.9 shows larger average and maximum energies and lower minimum energies than Fig. 8.9. The similarity between the two figures for the main part of the beam is consistent with our assumption of non-evolving wakefield if the accelerating field has a constant value within the radial range of the betatron oscillation for each particle. The differences in the tail are due to the fact that the particles in the tail at larger radii do not experience a constant accelerating field during their radial motion. For the full simulation this leads to an averaging out of the different accelerations experienced by each particle due to its transverse motion. For the particles accelerated with the initial field this averaging does not happen and the maximum and minimum energies in the beam tail of Fig. 8.9 are therefore a measure of the maximum and minimum accelerating field in that part of the wake. Based on these figures we can say that the maximum field is about $0.85\text{GeV}/\text{m}$ but

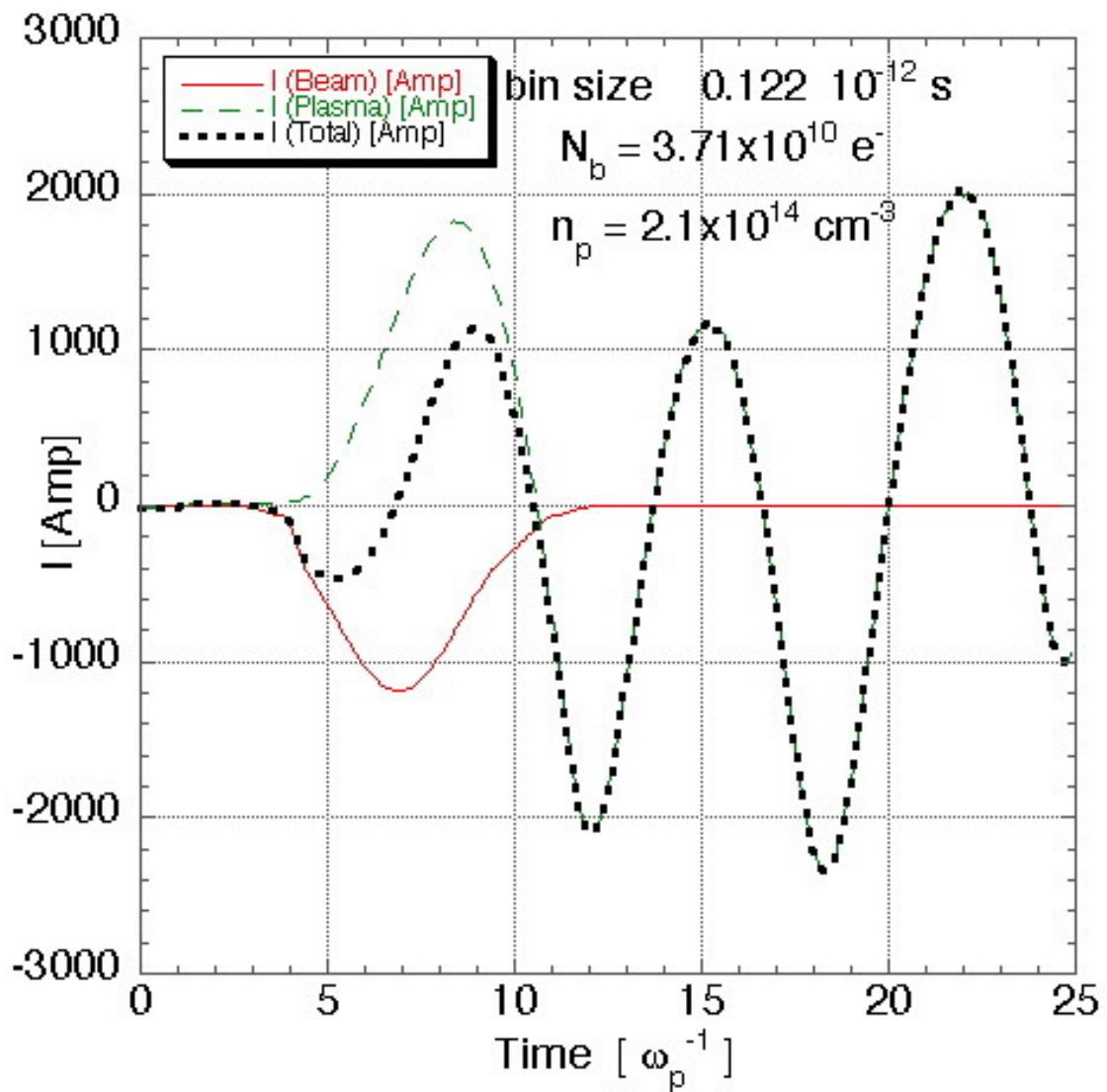


Figure 8.7: The beam current, the plasma current, and the total current for each 0.12 pico-second bin at the beginning of the simulation just after the beam fully entered the plasma.

that the maximum energy gain by a particle after 1.4 m will be about 1GeV. The maximum mean energy for a 0.12 pico-second bin is 550MeV with about 7×10^7 electrons in this maximum energy bin. This is again consistent with the information in Fig. 8.5 for these numbers. The conclusion from Fig. 8.8 and Fig. 8.9 is that the betatron oscillations do not have a significant influence on the acceleration of the beam.

There are a number of parameters in the experiment E-157 that can vary. The number of particles in the electron beam is one of these parameters and in order to investigate the effects of a change in the number of electrons a simulations with only half the number of beam electrons as before was done. All other parameters of the this simulation were kept the same. Fig. 8.10 shows the initial response of plasma for the previous simulation with $N_1 = 3.7 \times 10^{10}$ as well as the plasma response for the simulation with only have the number of beam electrons, $N_2 = 1.85 \times 10^{10}$. The overall structure of plasma response is the same for both beams. The main change is as would be expected from Eq. (8.2) a decrease of the blowout radius by a factor of roughly $\sqrt{2}$. In addition the begin of the total blowout area moves further to the back relative to the center of the electron beam (indicated in Fig. 8.10 by the vertical white line through the figure).

Fig. 8.11 which is a lineout of the accelerating field for the simulation with the smaller number of beam electrons shows an effect of the decreased blowout. The peak accelerating field decreases to $\sim 0.35\text{GeV}/\text{m}$. This is less than half the value seen in the other simulation.

The resulting decrease in the energy gain of the accelerated tail of the beam is shown in Fig. 8.12. The figure shows the mean, maximum, and minimum value of p_z as well as σ_{p_z} for 0.122ps-bins after 1.3 meters of propagation. The maximum mean

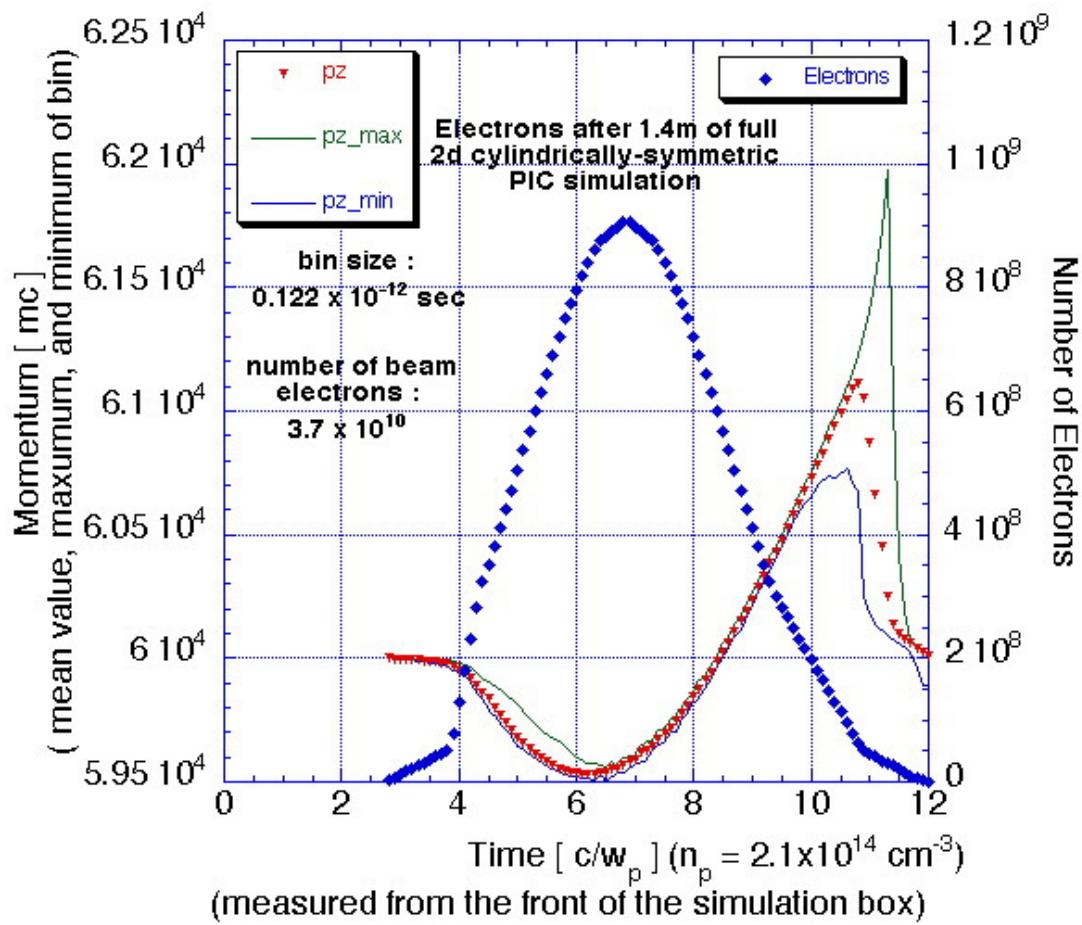


Figure 8.8: The mean, maximum, and minimum momentum in the propagation direction, p_z , as well as the number of electrons for each 0.12 pico-seconds bin after 1.4 meters of propagation using the full PIC simulation to propagate the beam.

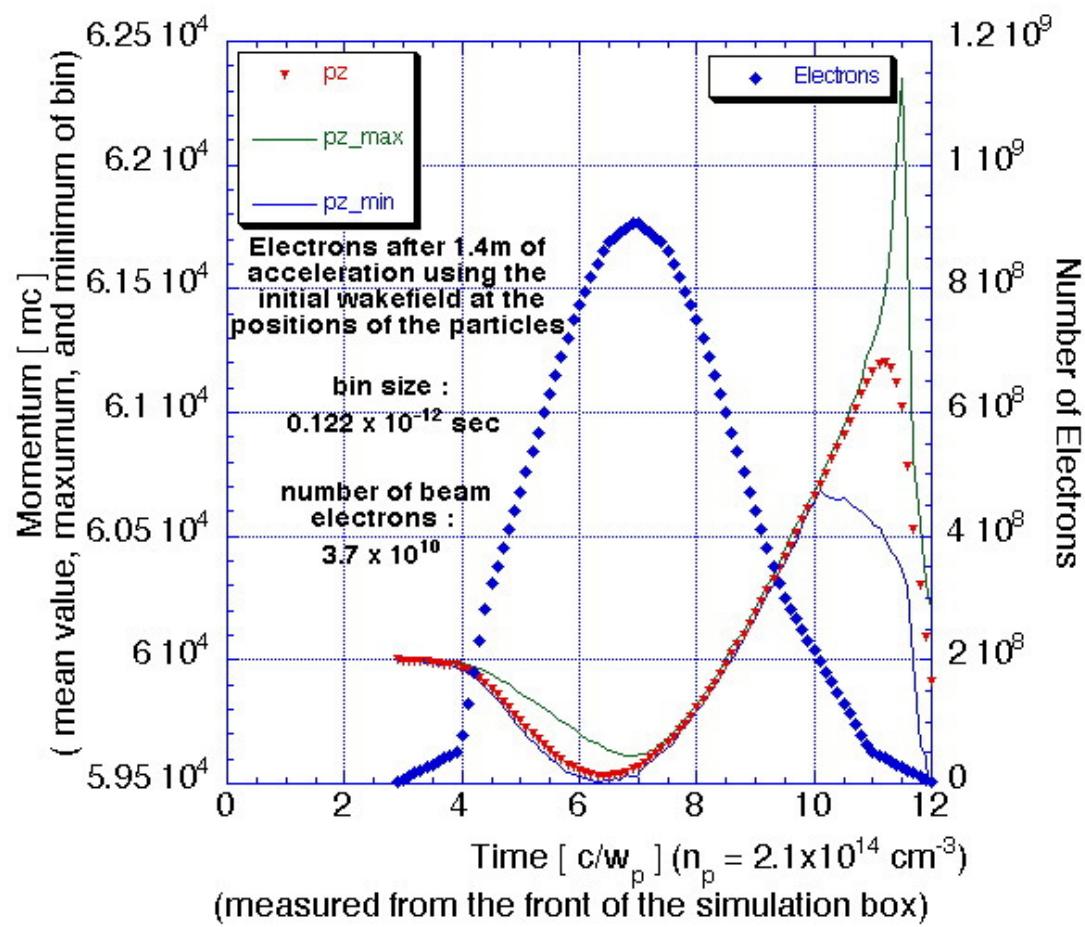


Figure 8.9: The mean, maximum, and minimum momentum in the propagation direction, p_z , as well as the number of electrons for each 0.12 pico-seconds bin after 1.4 meters of propagation using the initial fields at the initial positions of the particles to propagate the beam.

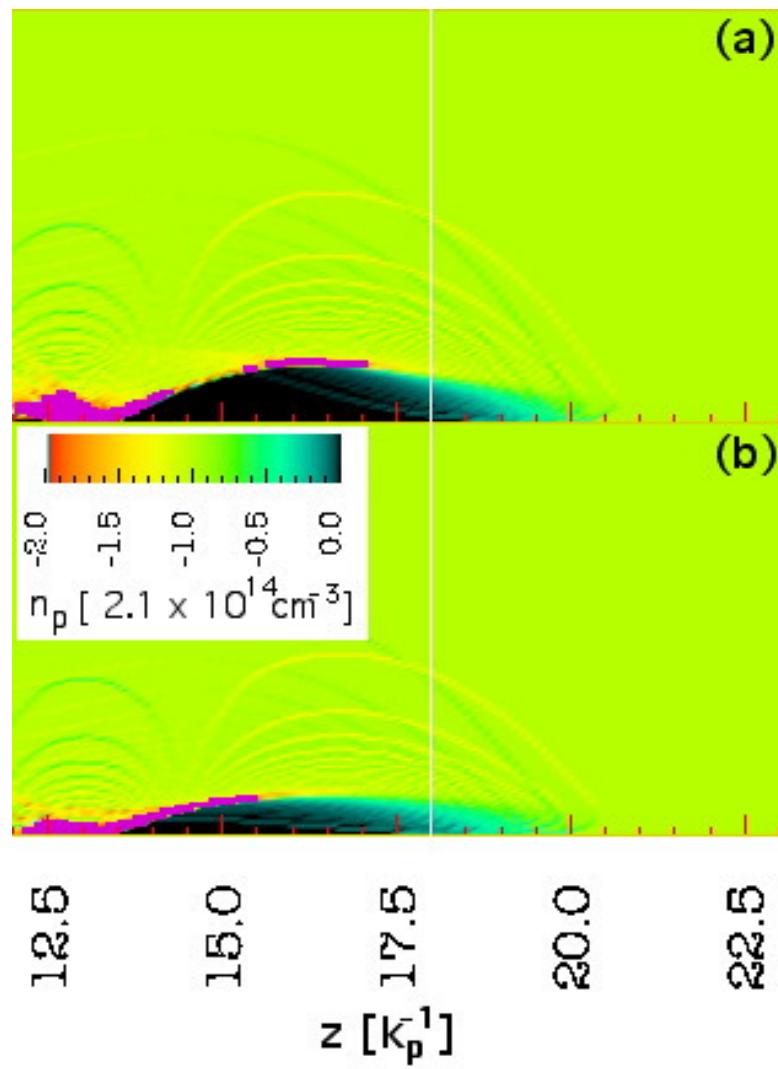


Figure 8.10: The initial response of plasma in simulations with (a) $N_1 = 3.7 \times 10^{10}$ beam electrons and (b) $N_2 = 1.85 \times 10^{10}$ beam electrons.

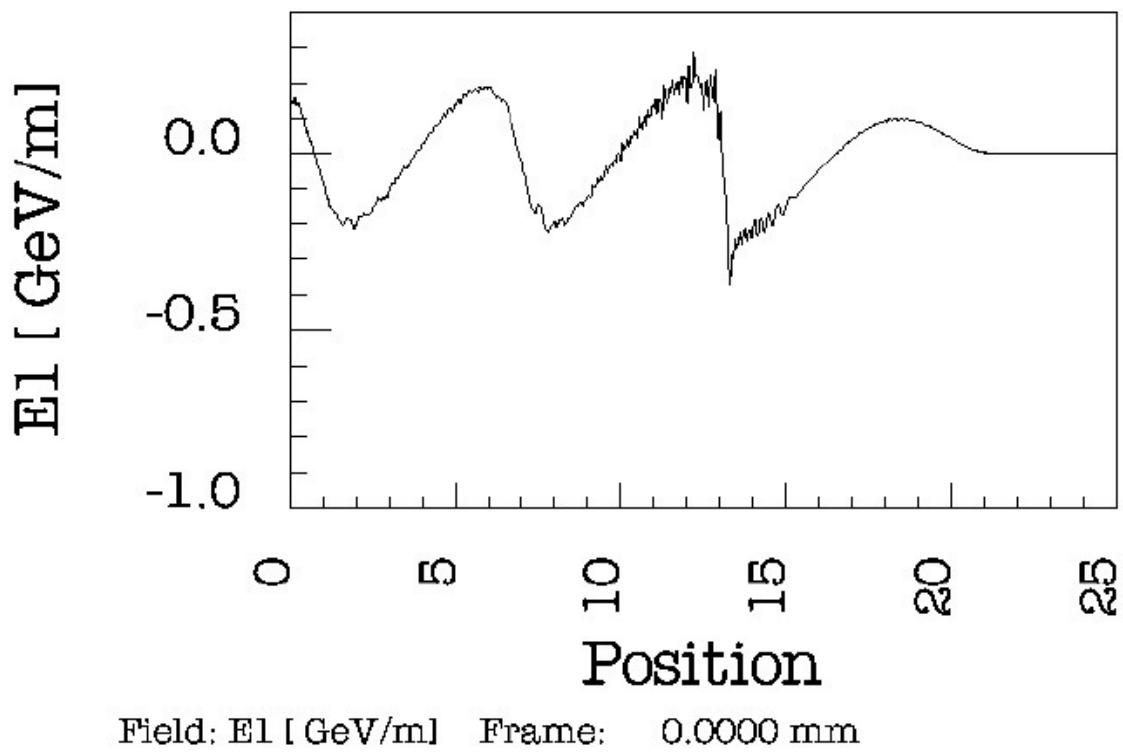


Figure 8.11: The lineout of the accelerating field along the axis for a simulation with $N_1 = 1.85 \times 10^{10}$.

energy gain is only about 250MeV.

8.3 3D Cartesian Simulations

A concern regarding the accuracy of the results presented in section 8.2 is the degree of numerical resolution needed to resolve the spike in the accelerating electric field and the use of 2D cylindrically-symmetric algorithms. In order to address the first point simulations with the beam only propagating for a short distance were done for several cell sizes. The results are summarized in Fig. 8.13. The figure shows the lineouts of the axial accelerating fields for the resolution used above, for twice this resolution, and for five times this resolution, i.e., the number of grid cells in each direction was increased by a factor of two and five, respectively, while the number of particles per cell was kept the same. The most important effect of the increased resolution is that the peak accelerating field becomes larger. Since this affects only a very narrow spike it means that the electric field has a very high value in a very small spatial area. This high field value is not properly resolved by the original simulation, but the rest of the beam plasma interaction is modeled accurately. The question is whether the insufficient resolution in this small area actually matters considering the purpose of the simulations. Since the higher field values will only affect the acceleration of a few simulation particles which will only contribute very little to the mean momentum in a bin of about 1 pico-second (this is the time resolution for measurements of the E-157 experiment[87]) it will not be of particular interest for predicting the energy gain in the experiment or the dynamics of the rest of the beam and the plasma. Furthermore, this large spike may not persist as the beam executes betatron oscillations.

In order to avoid the limitations of 2D cylindrically-symmetric simulations full

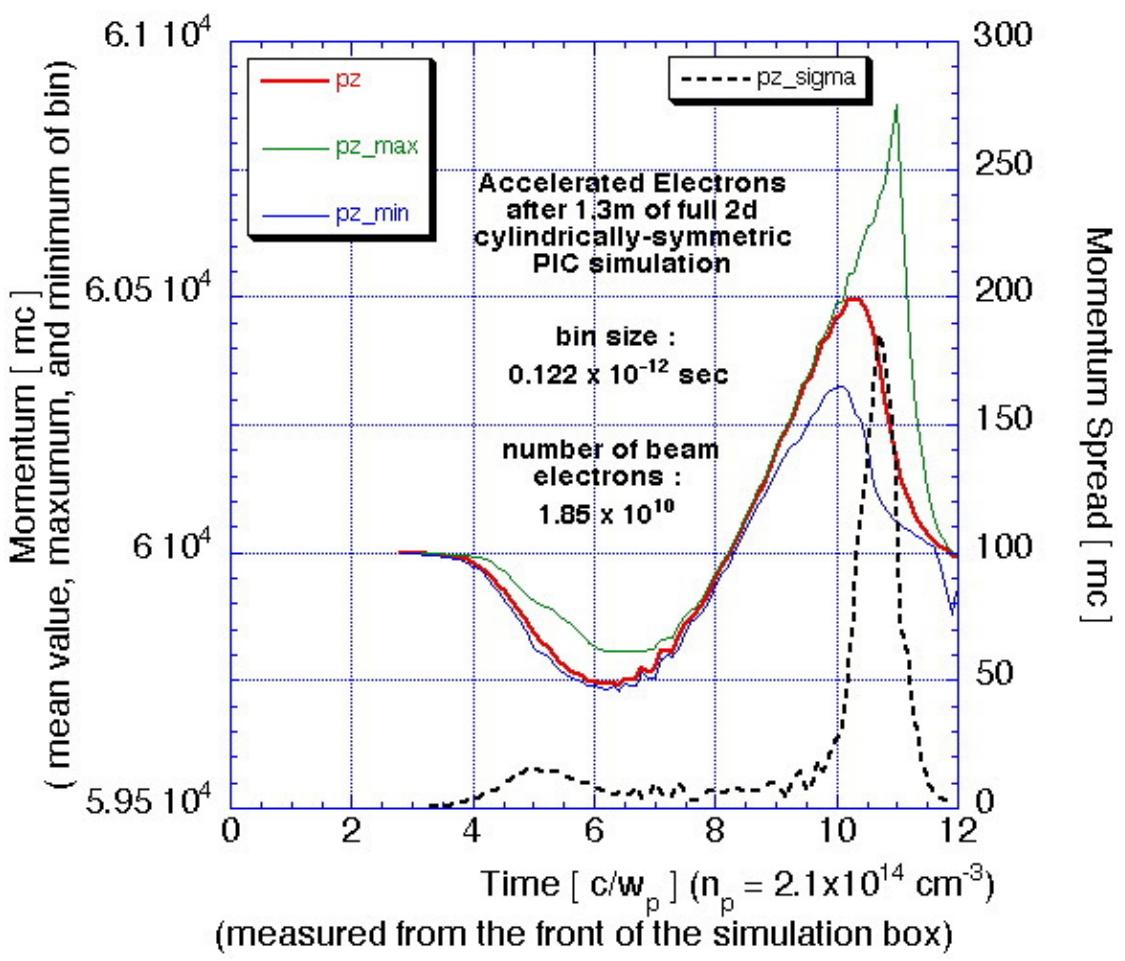


Figure 8.12: The mean, maximum, and minimum momentum in the propagation direction, p_z , as well as the width of the distribution of p_z , σ_{p_z} , for each 0.12 picoseconds bin after 1.3 meters of propagation using the full 2D cylindrically-symmetric PIC simulation to propagate the beam.

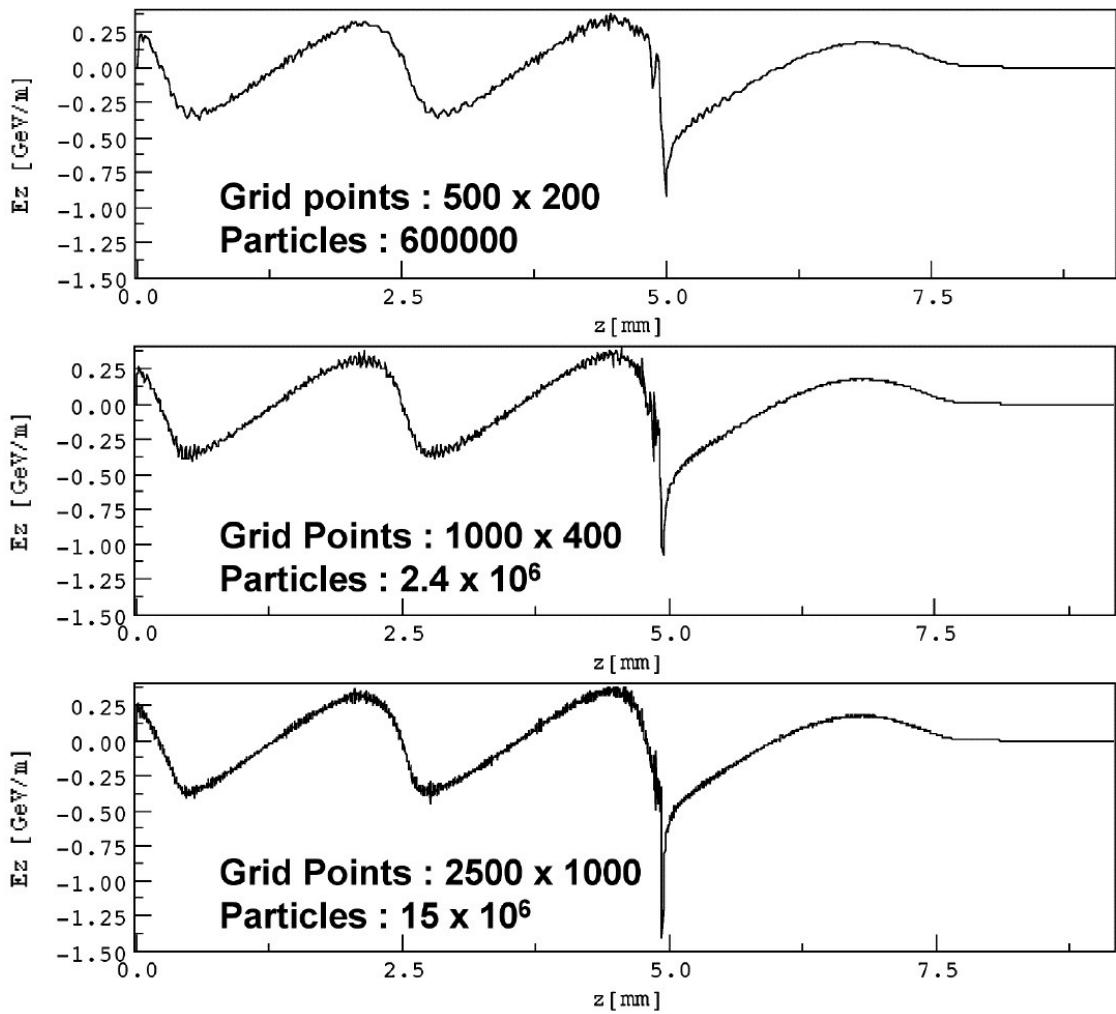


Figure 8.13: The axial lineout of the accelerating field of simulations of the PWFA using three different grid resolutions. The number of particles per cell is the same for all three simulations.

3D simulation were done as well. The simulations were propagated only over short distances that were just long enough for the beam to completely enter in to the plasma. This was done because of the much larger computational cost of full 3D simulations compared to 2D cylindrically-symmetric simulations.

In order to study both the possible numeric inaccuracies near the axis for a cylindrically-symmetric code and the effects of asymmetric drive pulses full 3D simulations were performed. The 3D simulations only modeled the initial excitation of the wake. Presently, it is not computationally feasible to perform full 3D simulations which model \sim meter propagation distances. The moving simulation window was also shortened so that only the first oscillation of the plasma wake fits into the simulations box. The size of the grid cells was kept $dx_1 = dx_2 = dx_3 = 0.05c/\omega_p$. The number of particles per cell for the plasma as well as the beam was four. The total size of the simulation was 14 million grid cells and about 56 million particles.

Fig. 8.14 shows a comparison between the central lineouts of the accelerating fields of the 2D cylindrically-symmetric simulation and of the full 3D simulation. The figure shows that the wake on axis for both simulations has the same structure and the amplitude agrees to within a few percent. Perhaps the most significant agreement is the amplitude and the structure of the spike in the accelerating field. As a result we have confidence that the peak value for the spike does not depend on which algorithms was used in the PIC code. The cell size dependence of the peak accelerating field value in Fig. 8.13 could have been a artifact of the 2D cylindrically-symmetric algorithms on axis but the 3D simulations do not have the same kind of problems on axis. Furthermore, since the 2D and 3D algorithms are very different, we have confidence that both are correct.

Now that we have confidence with the 3D algorithm, we can investigate the effects

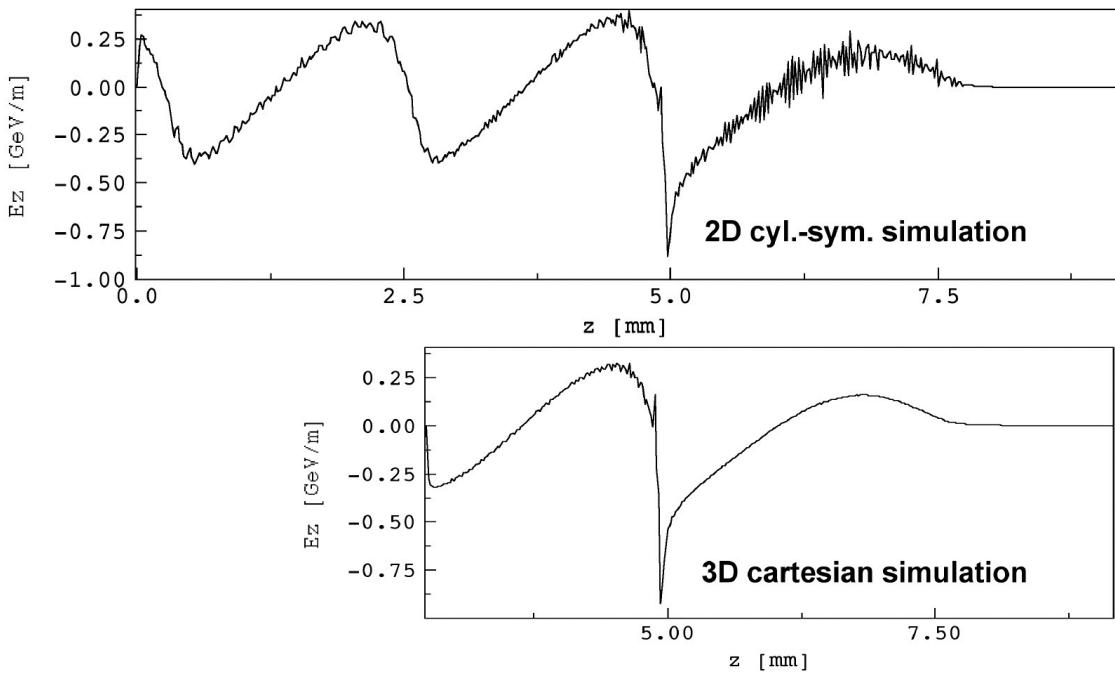


Figure 8.14: The lineouts of the accelerating field along the axis of the beam for a 2D cylindrically-symmetric and a 3D Cartesian simulation with the same beam and plasma parameters.

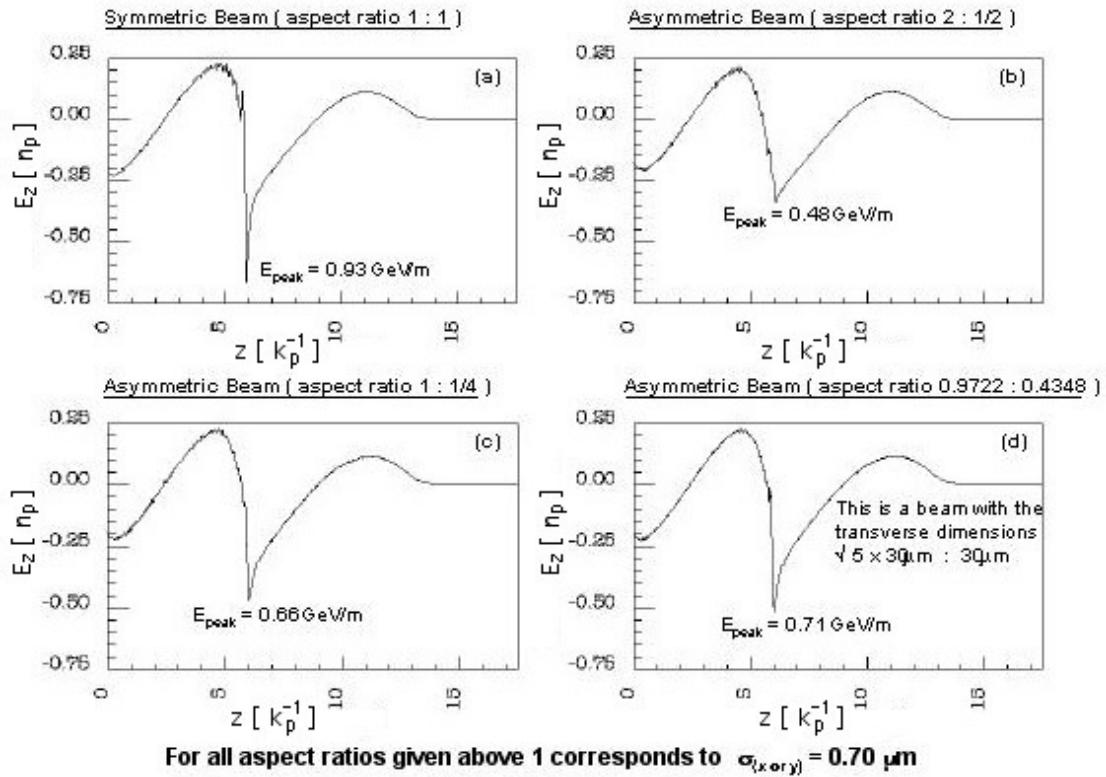


Figure 8.15: The lineouts of the accelerating electric field along the axis of the beam in the propagation direction z for different aspect ratios of the transverse beam spotsizes:
 (a) 1:1 (b) 2: $\frac{1}{2}$ (c) 1: $\frac{1}{4}$ (d) 0.9722:0.4348

of drive beam asymmetries on the wake generation. Fig. 8.15 shows lineouts of the accelerating electric field along the axis of the beam in propagation direction z for different aspect ratios of the transverse beam spotsizes, $\sigma_x : \sigma_y$. Fig. 8.15a) shows again the lineout for the symmetric case. Fig. 8.15b) shows the lineout for a simulation where the beam spotsize was increased by a factor of 2 in one transverse direction and decreased by a factor of 2 in the other transverse direction. For this $2:\frac{1}{2}$ aspect ratio the magnitude of the peak accelerating field decreases to about half of its magnitude in the symmetric case. This decrease of the peak field amplitude is consistent with the idea given earlier that the sharp peak in the accelerating field is due to the return of the blown-out plasma electrons to precisely the same point on axis. If the spotsize of the beam in the different transverse directions is not the same then electrons which are blown out on different transverse trajectories will return to the axis at different points. This means the density spike on axis will be smoothed out over a larger area causing the peak of the accelerating field to be smaller than for a symmetric beam.

Fig. 8.15c) shows the lineout for a simulation where the beam spotsize was kept the same in one transverse direction and decreased by a factor of 4 in the other transverse direction. This $1:\frac{1}{4}$ aspect ratio is different than the $2:\frac{1}{2}$ because in this case the peak density of the beam is larger by a factor of 4. For this aspect ratio the peak accelerating field still decreases, but only by about 70% of the 1:1 case in Fig. 8.15a). This case is important because it shows that any asymmetry in the beam will smear out the spike even if the blowout radius is much larger than either transverse dimension.

Fig. 8.15d) we show the lineout for a simulation where the beam spotsize was decreased only slightly in one transverse direction and decreased by a factor of roughly 2 in the other transverse direction. This 0.9722:0.4348 aspect ratio corresponds to

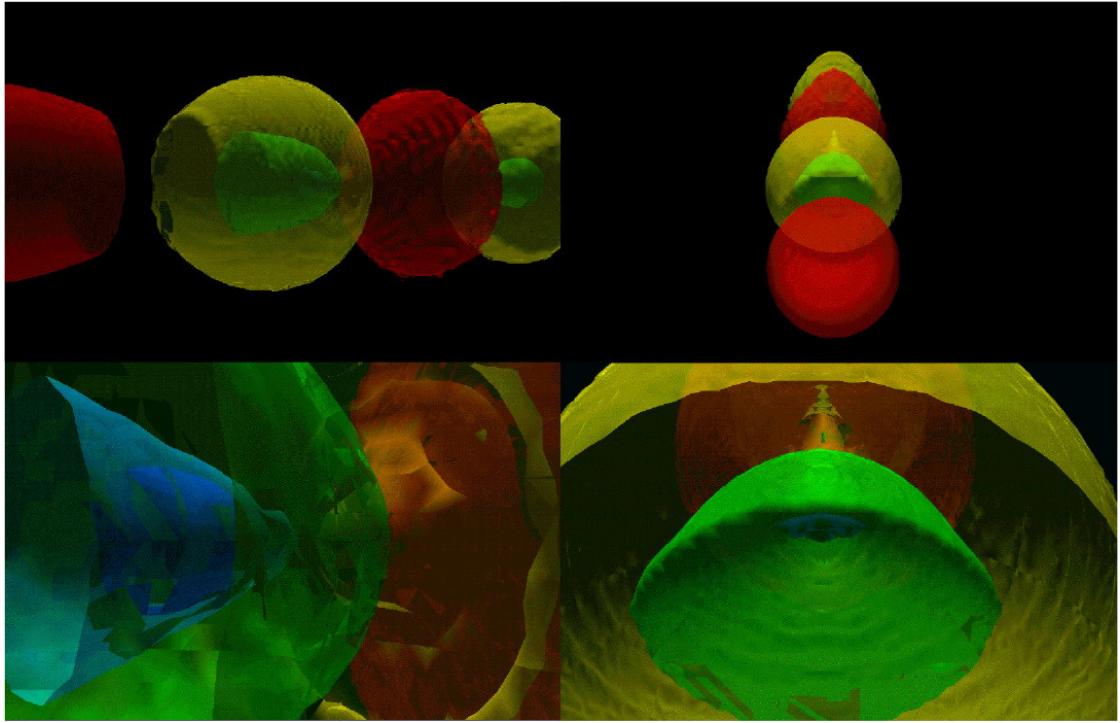


Figure 8.16: Selected isosurfaces of the accelerating field. The dark blue, light blue, green, and yellow surfaces corresponds to an acceleration gradients of 0.5 , 0.4, 0.2, and 0.1 GeV/m while the red surfaces correspond to a decelerating gradient of 0.1 GeV/m. The fields shown in the left column and right column are from the simulation with an aspect ratio of 1:1 and $2:\frac{1}{2}$ respectively. This figure has been made possible by the help of the Office of Academic Computing at UCLA.

that of a typical SLAC beam at energies higher than the 30GeV used in the E-157 experiment[89]. This case looks most similar to the $1:\frac{1}{4}$ case. However, although the peak density is smaller the peak wake amplitude is still higher. This clearly demonstrates that round beams are desirable.

In Fig. 8.16 selected isosurfaces of the accelerating field are shown for two of the 3D simulations. The dark blue, light blue, green, and yellow surfaces corresponds to an acceleration gradients of 0.5 , 0.4, 0.2, and 0.1 GeV/m while the red surfaces correspond to a decelerating gradient of 0.1 GeV/m. The fields shown in the left

column and right column are from the simulation with an aspect ratio of 1:1 and 2: $\frac{1}{2}$ respectively. The viewpoints and perspectives were chosen to best visualize information about the size and shape of the accelerating areas of the wake. The interesting fact to note in this figure is that the peak accelerating volume for the asymmetric beam is small compared to the peak accelerating volume for the symmetric beam. Beam asymmetry therefore does not just decrease amplitude of the peak accelerating field but also causes the accelerating volume to decrease. This is at this time only a qualitative statement and it will require further research and data analysis to be able to quantify this statement more precisely.

8.4 An Analytical Model

The simulation results presented above serve the purpose of giving an understanding of the wake excitation process and of the dependence of blowout on the beam parameters. This can now be used to guide the development of an analytical model that gives a deeper understanding of the physics. In this section we will develop a model that within certain limits predicts the motion of individual plasma electrons, and hence the plasma response to the beam.

The analytical approach combines Whittum's frozen field formalism [79] and a cylindrical sheet model used by Mori et al.[90]. We start with the wave equations for the vector and scalar potential in the Lorentz gauge [91].

$$\left(\frac{1}{c^2} \frac{\partial^2}{\partial t^2} - \vec{\nabla}^2 \right) \vec{A} = \frac{4\pi}{c} \vec{j} \quad (8.3)$$

$$\left(\frac{1}{c^2} \frac{\partial^2}{\partial t^2} - \vec{\nabla}^2 \right) \Phi = 4\pi\rho \quad (8.4)$$

The mathematical transformation to the speed-of-light variables $\xi = z - ct$ and $s = z$ means that

$$\frac{\partial}{\partial z} = \frac{\partial}{\partial \xi} + \frac{\partial}{\partial s} \quad (8.5)$$

and

$$\frac{\partial}{\partial t} = -c \frac{\partial}{\partial \xi} \quad (8.6)$$

From the simulations we know that the plasma response is essentially stationary in the moving frame and we can make the frozen-field approximation [79] $\partial/\partial s \rightarrow 0$. With this Eq. (8.3) and Eq. (8.4) become:

$$-\vec{\nabla}_{\perp}^2 \vec{A} = \frac{4\pi}{c} \vec{j} \quad (8.7)$$

$$-\vec{\nabla}_{\perp}^2 \Phi = 4\pi \rho \quad (8.8)$$

As noted by Whittum, this has essentially reduced this problem to a 2D Poisson problem. We next assume cylindrical symmetry to reduce this even further to a one dimensional problem that can be solved analytically under certain conditions.

Since we know for the plasma electrons that $|v_z| \ll c$ where v_z is the axial velocity of the plasma, we will henceforth assume $v_z = 0$. This approximation can be verified a posteriori. When considering the Lorentz force [Eq. (3.1)] on the plasma electrons this means that the magnetic field can be neglected. The force on the plasma electrons is therefore [91]:

$$\vec{F} = \begin{pmatrix} \vec{F}_z \\ \vec{F}_\perp \end{pmatrix} = q \begin{pmatrix} -\frac{\partial}{\partial z}\Phi - \frac{1}{c}\frac{\partial}{\partial t}A_z \\ -\vec{\nabla}_\perp\Phi - \frac{1}{c}\frac{\partial}{\partial t}\vec{A}_\perp \end{pmatrix} = q \begin{pmatrix} -\frac{\partial}{\partial \xi}(\Phi - A_z) \\ -\vec{\nabla}_\perp\Phi + \frac{\partial}{\partial \xi}\vec{A}_\perp \end{pmatrix} \quad (8.9)$$

Since we are neglecting the axial motion of the plasma electrons we will only consider the perpendicular component of Eq. (8.9). The source term of Φ , i.e., ρ , has three contributions: the charge density due to the beam $\rho_b = -en_b$, the charge density due to the ion background $\rho_i = en_p$, and the charge density due to the plasma electrons $\rho_e = -en_e$. If we neglect the ion motion and the radial motion of the beam electrons due to the betatron oscillations then the source term $\frac{1}{c}\vec{j}_\perp$ of \vec{A}_\perp is given by $-ev_r n_e/c$. Since we know from the simulation and analytical estimates that v_r/c is small compared to 1 we can to lowest order neglect this term and therefore \vec{A}_\perp as well.

In case of a cylindrically-symmetric problem the perpendicular component of Eq. (8.8) reduces to

$$-\frac{1}{r}\frac{\partial}{\partial r}r\frac{\partial}{\partial r}\bar{\Phi} = 4\pi\rho\frac{e}{mc^2} = -k_p^2\frac{n}{n_p} \quad (8.10)$$

where $\bar{\Phi}$ is the normalized potential $e\Phi/(mc^2)$, k_p the plasma wave vector, and n the density of charged particles. Note that n has to be negative in order to make this notation work for positive ions.

The potential due to a given charge distribution can be obtained using the Green's function for Eq. (8.10) which is

$$G(r, r') = \frac{\ln(r/r')}{2\pi} H(r - r') \quad (8.11)$$

Here $H(r - r')$ is the Heaviside step function which is 1 for $r \geq r'$ and 0 otherwise.

For a Gaussian drive beam ρ_b is:

$$\rho_b = -eN_b \times \frac{1}{\sqrt{2\pi}\sigma_z} e^{-\frac{(\xi-\xi_0)^2}{2\sigma_z^2}} \times \frac{1}{2\pi\sigma_r^2} e^{-\frac{r^2}{2\sigma_r^2}} \quad (8.12)$$

where N_b is the number of electrons in the beam, σ_z the width of the Gaussian distribution in propagation direction, σ_r the radial spotsize of the beam, and ξ_0 the position of the beam center using speed-of-light variables. With this we get

$$\begin{aligned} \bar{\Phi}_b(r, \xi) &= 2\pi \int_0^r \frac{\ln(r/r')}{2\pi} \left((2\pi)^{-3/2} \frac{N_b}{\sigma_r^2 \sigma_z} \frac{k_p^2}{n_p} e^{-\frac{(\xi-\xi_0)^2}{2\sigma_z^2}} e^{-\frac{r'^2}{2\sigma_r^2}} \right) r' dr' \\ &= -2 (2\pi)^{-3/2} \frac{N_b}{\sigma_z} \frac{k_p^2}{n_p} e^{-\frac{(\xi-\xi_0)^2}{2\sigma_z^2}} \\ &\quad \times \left(\int_0^\beta \alpha \ln(\alpha) e^{-\alpha^2} d\alpha - \ln(\beta) \int_0^\beta \alpha e^{-\alpha^2} d\alpha \right) \end{aligned} \quad (8.13)$$

with $\beta = r/(\sqrt{2}\sigma_r)$. The two integrals can be solved, leaving:

$$\begin{aligned} \bar{\Phi}_b(r, \xi) &= -2 (2\pi)^{-3/2} \frac{N_b}{\sigma_z} \frac{k_p^2}{n_p} e^{-\frac{(\xi-\xi_0)^2}{2\sigma_z^2}} \\ &\quad \times \left(\frac{1}{4} \left(-\gamma - \Gamma \left(0, \left(\frac{r}{\sqrt{2}\sigma_r} \right)^2 \right) \right) - \frac{1}{2} \ln \left(\frac{r}{\sqrt{2}\sigma_r} \right) \right) \end{aligned} \quad (8.14)$$

where $\gamma \simeq 0.577216$ is Euler's constant and $\Gamma(a, z) = \int_z^\infty t^{a-1} e^{-t} dt$ is the incomplete Euler-gamma function.

For the constant background of fixed ions, we have $\rho_i = -e (-n_p)$. For such a ρ it is straight forward to integrate Eq. (8.11) giving

$$\phi_i(r) = -\frac{k_p^2}{4} r^2 \quad (8.15)$$

Calculating the potential due to the plasma electron density n_e is more difficult since n_e is not explicitly known. However, the simulations show a largely laminar flow of electrons passing the beam (in the beam's moving frame). Based on this observation we can make an approximation that will allow us to calculate a potential that a given electron experiences. We assume that there is no ring crossing of cylindrical charge rings around the axis of the beam. That is, for any given ring of charge the number of electrons inside and outside its radial position at any time does not change. We can therefore obtain the potential and fields experienced by a particular ring by simple considerations. From Gauss' law the field at the position of a particular ring at position r is given by the field due to the amount of electron charge enclosed by the ring. As long as no rings cross each other then the enclosed electron charge is simply that which was enclosed when the electron ring was at its original position r_i . The potential due to the electrons is then given by

$$\Phi_\lambda(r) = -2 \lambda \ln(r/r_0) \quad (8.16)$$

where λ is the charge per unit length for $r < r_i$ and r_0 provides an arbitrary integration constant. An initial uniform charge distribution $\lambda = \pi r_{i^2} n_p$ gives

$$\bar{\Phi}_e(r, \xi) = \bar{\Phi}_e(r(r_i, \xi)) = \frac{k_p^2}{2} r_i^2 \ln\left(\frac{r(r_i, \xi)}{\sqrt{2}\sigma_r}\right) \quad (8.17)$$

where for simplicity r_0 in Eq. (8.16) has been set to $\sqrt{2}\sigma_r$.

The total $\bar{\Phi}$ is therefore given by

$$\begin{aligned}
\bar{\Phi}(r, \xi) &= \bar{\Phi}_b + \bar{\Phi}_i + \bar{\Phi}_e \\
&= -2 (2\pi)^{-3/2} \frac{N_b}{\sigma_z} \frac{k_p^2}{n_p} e^{-\frac{(\xi-\xi_0)^2}{2\sigma_z^2}} \\
&\quad \times \left(\frac{1}{4} \left(-\gamma - \Gamma \left(0, \left(\frac{r}{\sqrt{2}\sigma_r} \right)^2 \right) \right) - \frac{1}{2} \ln \left(\frac{r}{\sqrt{2}\sigma_r} \right) \right) \\
&\quad - \frac{k_p^2}{4} r^2 + \frac{k_p^2}{2} r_i^2 \ln \left(\frac{r}{\sqrt{2}\sigma_r} \right)
\end{aligned} \tag{8.18}$$

The equation of motion for an electron ring is therefore given by

$$\frac{1}{c} \frac{d}{dt} \bar{p}_r = \frac{\partial}{\partial \xi} \gamma \frac{\partial}{\partial \xi} r = \frac{\partial}{\partial r} \bar{\Phi} \tag{8.19}$$

where $\bar{p}_r \equiv p_r/(mc)$. Using the fact that the motion of the plasma is assumed to be purely radial this can be rewritten as

$$\gamma^3 \frac{\partial^2}{\partial \xi^2} r = \left(1 - \left(\frac{\partial}{\partial \xi} r \right)^2 \right)^{-\frac{3}{2}} \frac{\partial^2}{\partial \xi^2} r = \frac{\partial}{\partial r} \bar{\Phi} \tag{8.20}$$

This is a differential equation for the radial motion of a particle with an initial position r_i . The solution will give us r as function of ξ for a given initial r_i . While much analytical work can still be done with Eq. (8.20), we next solve it numerically.

Fig. 8.17 shows the solutions of Eq. (8.20) for 13 different initial radii starting at $r_i = \sigma_r$ and then increasing in equal steps of $\sigma_r/3$. The beam and plasma parameters used are the same as the ones used for the PIC simulation presented above with $N_b = 3.7 \times 10^{10}$. The upper part of the picture shows the results using the same scaling for both axes. The lower part of the figure shows the same results but with a blown up radial axis in order to show more detail. There are two interesting things to

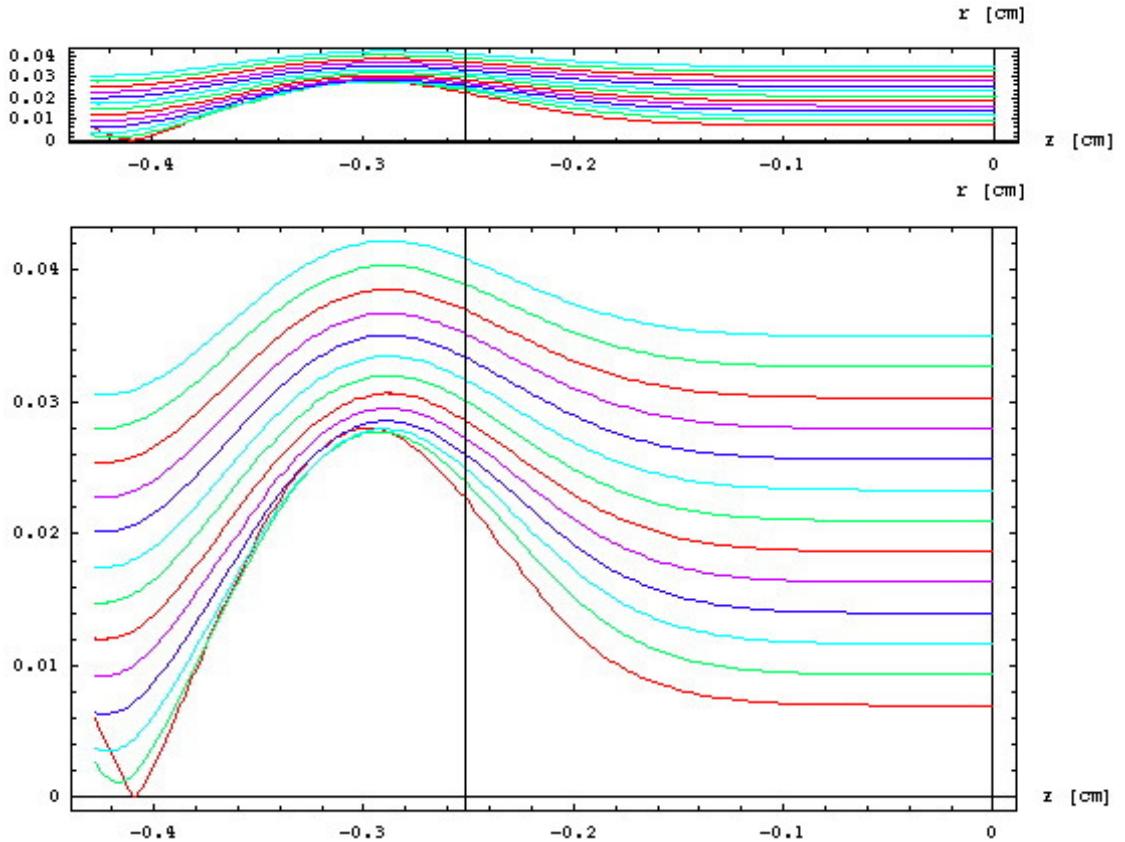


Figure 8.17: Solutions for the electron trajectories for 13 different initial radii starting at $r_i = \sigma_r$ and then increasing in equal steps of $\sigma_r/3$. The beam has $N_b = 3.7 \times 10^{10}$. The upper part of the picture shows the results using the same scaling for both axes. The lower part of the figure shows the same results but with a blown up radial axis in order to show more details. The vertical line in the center of the figure indicates the center of the electron beam.

note in this figure. First, particles with an initial radius of about σ_r have trajectories that cross other trajectories and therefore the question is raised as to how valid the model is for particles with radii of σ_r or smaller. However, the maximum blowout radius seen in this Fig. 8.17 is about $280\mu m$, which is within a few percent of the blowout radius seen in the full PIC simulation. This means that at least up to the point of maximum blowout the model is quite accurate.

Fig. 8.18 shows the solution for 60 different initial radii starting at $\sigma_r/12$ and then increasing in equal steps up to $5\sigma_r$. This “density” of trajectories corresponds to the same transverse density of particles as used in the PIC simulation. The range of ξ over which the trajectories are shown is shorter than in Fig. 8.17 because some of the particles with $r_i < \sigma_r$ will cross the axis, $r = 0$, for $\xi < 0.4cm$ so that the model breaks down. Fig. 8.19 shows trajectories for the same initial radii as in Fig. 8.18 but for a beam with only half the number of electrons. The range of ξ over which the trajectories are shown in Fig. 8.19 is the same as in Fig. 8.17 because no problems with particles crossing the axis appeared. In Fig. 8.18 as well as in Fig. 8.19 the upper part shows the trajectories using the same scaling for both axes while the lower part again blows up the radial axis to make certain details more visible.

Fig. 8.18 and Fig. 8.19 should be compared with Fig. 8.10, which shows the plasma response for the PIC simulations with the same parameters. The similarity of the results is good enough that even certain particle trajectories can be identified which each other. This indicates that the model developed here is quite good for estimating the blowout radius despite the fact that at least some sheet crossing does take place and the longitudinal motion was neglected. For the two different beams shown in Fig. 8.17 to Fig. 8.19 the PIC simulations and the analytical model agree within about 5% for the values of the blowout radii.

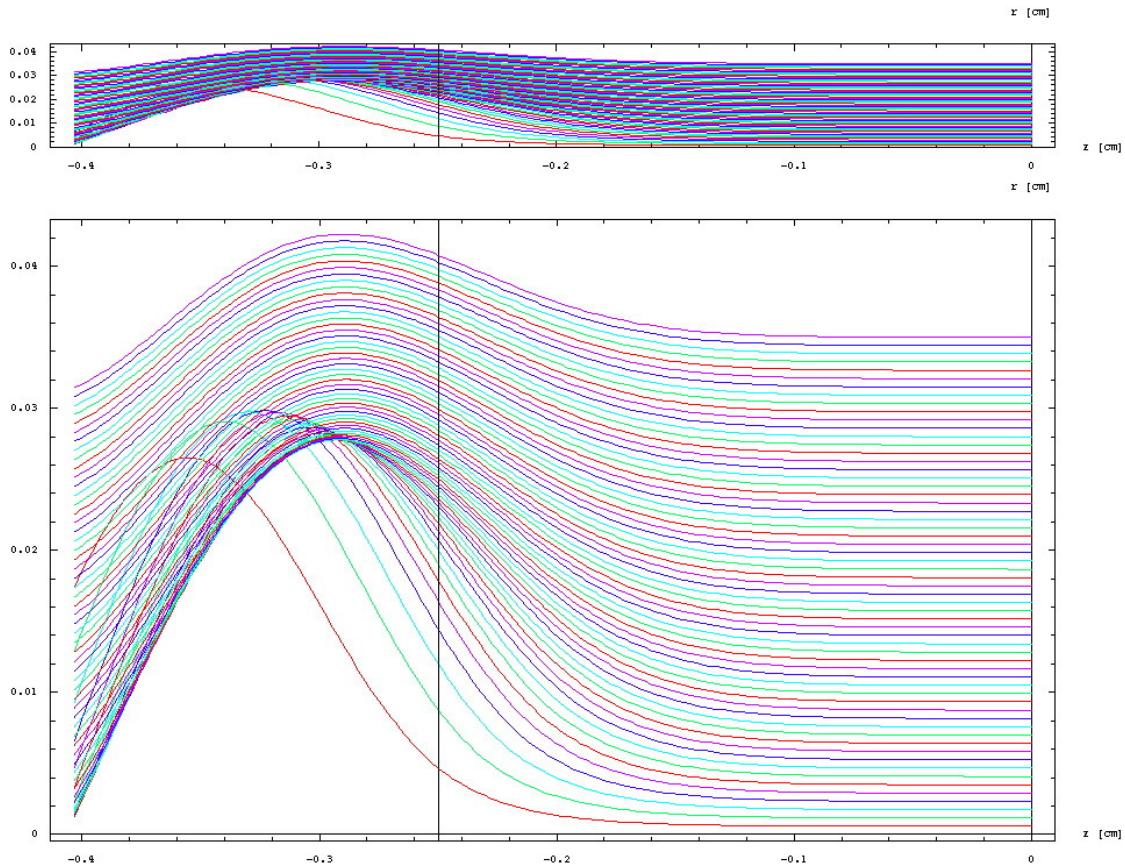


Figure 8.18: Solutions for the electron trajectories for 60 different initial radii starting at $r_i = \sigma_r/12$ and then increasing in equal steps of $\sigma_r/12$. The beam has $N_b = 3.7 \times 10^{10}$. The upper part of the picture shows the results using the same scaling for both axes. The lower part of the figure shows the same results but with a blown up radial axis in order to show more details. The vertical line in the center of the figure indicates the center of the electron beam.

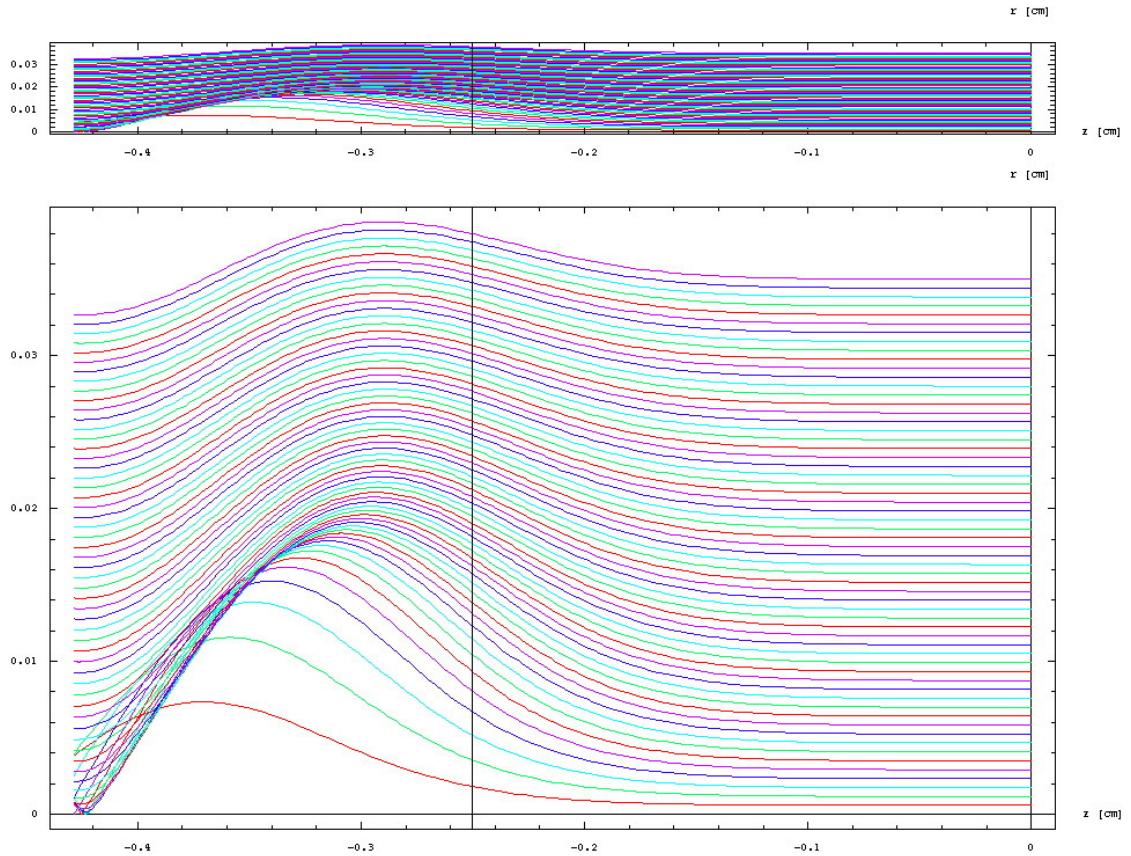


Figure 8.19: Solutions for the electron trajectories for 60 different initial radii starting at $r_i = \sigma_r/12$ and then increasing in equal steps of $\sigma_r/12$. The beam has $N_b = 1.85 \times 10^{10}$. The upper part of the picture shows the results using the same scaling for both axes. The lower part of the figure shows the same results but with a blown up radial axis in order to show more details. The vertical line in the center of the figure indicates the center of the electron beam.

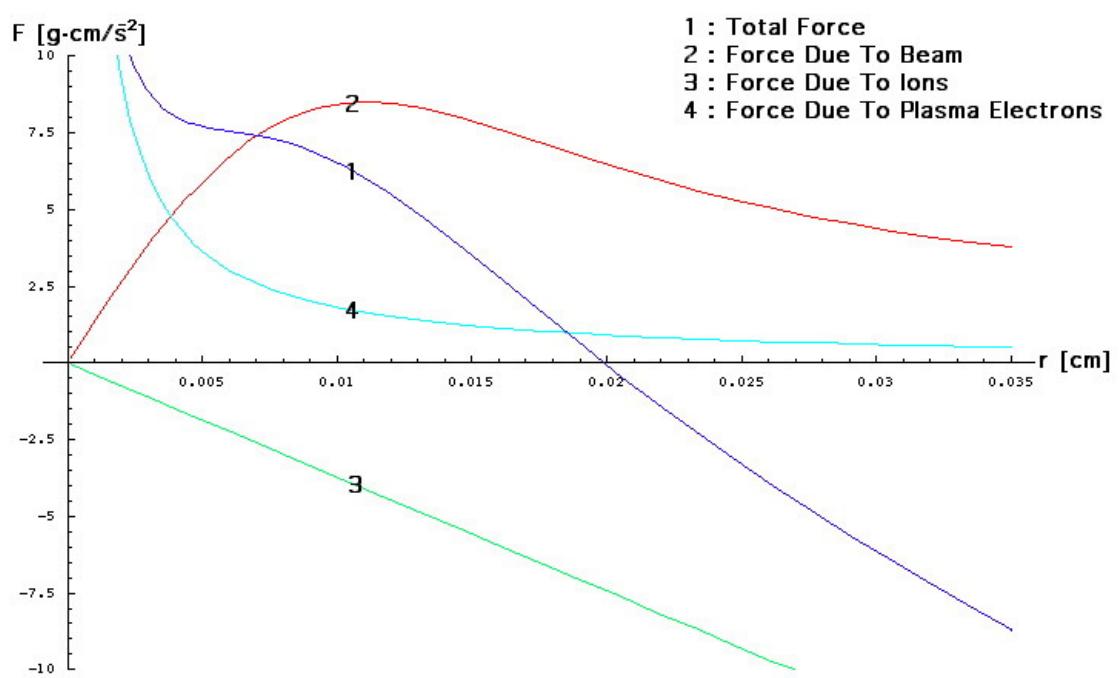


Figure 8.20: The forces acting on an electron starting at an initial radius $r_i = \sigma_r$.

To make the agreement more understandable we plot in Fig. 8.20 the forces acting on an electron with an initial radius $r_i = \sigma_r$. It shows that the contribution of the plasma electrons to the total force is relatively small for a radius of the order of the blowout radius.

8.5 Conclusion

The main result of the analysis of the beam and wakefield dynamics is that the wakefield is rather insensitive to the betatron oscillation dynamics of the beam and therefore essentially constant over time. In addition the acceleration and deceleration of the beam electrons is not affected by the betatron oscillation either. On the other hand the magnitude of the peak accelerating field is decreased significantly by a decrease in the number of beam electrons and by asymmetries in the beam spotsize. These are therefore parameters that have to be controlled carefully in PWFA experiments. The analytical model we developed predicts the blowout radii and trajectories seen in the PIC simulations well. It can therefore be used as a starting point for further analytical work.

The results in this chapter indicate that the blowout regime provides stable and robust plasma wakefield acceleration. Energy gains on the order of a GeV should be achievable in this blowout regime if the physical parameters of the simulations can be realized in an experiment. Far higher gradients and energy gains may be possible with shorter bunches and longer denser plasmas[15]. Such beams would undergo hundreds rather than a few betatron oscillations. Although it maybe desirable to match the beam emittance to the plasma focusing strength to avoid betatron oscillations as discussed in Ref.[15], the analysis here suggests that the presence of the oscillation

is not necessarily detrimental. A major issue for the scaling of plasma wakefield acceleration to the 10s and 100s of GeV, is the possibility of a hosing instability of the beam, which might reduce the achievable energy gain and lead to emittance growth of beam. Hosing is inherently a three-dimensional instability and is therefore absent in the 2D simulations. We are currently carrying out 3D simulations over long propagation distances to study the importance of hosing and other 3D effects and this is an area of future work.

Chapter 9

Summary

9.1 Important Results

The complex nonlinear interactions between the driver, the plasma wake, and the accelerated electron bunch in plasma-based accelerators make PIC modeling the only method of getting a complete, integrated picture of the evolution of such a system. In this dissertation, we have described the development of a new object-oriented code, OSIRIS, which is ideally suited for modeling laser plasma and beam plasma interactions.

The object-oriented design of OSIRIS made it possible to create a currently unique combination of advanced algorithms. OSIRIS now contains algorithms for 1D, 2D, and 3D simulations in Cartesian coordinates and for 2D simulations in cylindrically symmetric coordinates. For all of these algorithms the code is fully relativistic and presently uses a charge-conserving current deposition algorithm. It allows for a moving simulation window and arbitrary domain decomposition for any number of dimensions. This combination of algorithms makes OSIRIS a useful tool for many

different research problems besides plasma-based accelerators, with the possibility to be extended much further by adding new modules.

We applied the PIC modeling technique to investigate a number of problems in plasma-based accelerator research. The main results of our research are:

- Using 2D and 3D PIC simulations we studied the injection scheme proposed in Ref. [59]. We find that the beam brightness and quality compares reasonably with that of electron bunches produced using conventional technologies. We believe that the mechanism for the trapping of particles in our simulation is the interaction of the particles with the two plasma wakes. The 3D simulation is the only one to date on this problem.
- High resolution simulations of long laser pulses indicated the existence of a hosing instability with a wavelength longer than the plasma wavelength. The simulations motivated the development of a theory by Duda and Mori that explains the long wavelength hosing seen in the simulations. It is found that this effect might significantly increase the emittance of electron beams produced in the SMLFWA and it may be important when high-intensity lasers propagate in densities above the quarter critical density as in the fast ignitor concept.
- 2D simulations of a LWFA in a parabolic plasma channel indicate that the idea of preventing diffraction of a laser pulse by propagating it through a matched parabolic plasma channel works. Only about 3% of the length of one plasma wave oscillation accelerates externally injected particles. The test particles in the simulation gain more energy than the linear scaling laws for a uniform plasma would predict.
- 2D and 3D simulations of a PWFA in the blowout regime showed that the wake-

field is rather insensitive to the betatron oscillation dynamics of the beam or the acceleration and deceleration of the beam electrons and is therefore essentially constant over time. On the other hand the magnitude of the peak accelerating field is decreased significantly by a decrease in the number of beam electrons and by asymmetries in the beam spotsize. An analytical model was developed that predicts the blowout radii and trajectories seen in the PIC simulations well.

9.2 Future Challenges

Research on plasma-based acceleration has made significant progress in the areas of experiment, theory, and simulations, but much work remains to be done. Of the research topics in this dissertation the LWFA in a channel and the PWFA in the blowout regime are the ones that will strongly benefit from more simulation research. 3D simulations of the LWFA will require considerable computational resources but would allow a much more detailed understanding of the whole system. One of the current questions for PWFA in the blowout regime is the significance of the tail hosing instability. Hosing is inherently a three-dimensional instability and is therefore absent in 2D simulations. Therefore, 3D simulations over long propagation distances are required and are currently being done to study the importance of hosing and other 3D effects.

An important challenge for the future are simulations of accelerators over the full distance of acceleration. This problem is a motivation for a number of possible extensions to OSIRIS. The challenge lies in the fact that there are in principle three length scales that have been considered, the scale of the driver evolution (laser or particle beam), the scale of plasma wave length which is needed to resolve the evolution of the accelerating wake, and finally the total length of the particle acceleration. The

problem of simulating plasma based acceleration over the distances comparable to real experiments requires very efficient use of the available computational resources. Mechanisms like dynamic load balancing, adaptive mesh refinement [92, 93], and more sophisticated boundary conditions which reduce the required number of simulation cells are ways to strongly improve the code performance while maintaining the accuracy of simulation results. For laser drivers, explicit PIC algorithms lead to the problem of having to resolve the laser wavelength, λ_L , which is typically several orders of magnitude smaller than the plasma wavelength, λ_p , and therefore the required computational effort is the square of λ_p/λ_L times larger than if just the plasma wavelength would be resolved. This increase could be avoided if the laser evolution is not described by a fully explicit PIC algorithm but by other means as for example following a laser envelope equation [94]. For particle beam drivers (and even laser drivers) the timescales for the evolution of the driver are orders of magnitude smaller than the plasma frequency. As a result, algorithms which evolve the drive beam separately would be extremely useful. Such codes already exist. Fully explicit codes will however always be needed to benchmark any reduced description code. Furthermore, OSIRIS with its object-oriented design should make the implementation and use of these feature easier and less time consuming while providing a framework for automatic parallelization and providing sophisticated diagnostic possibilities.

Another problem for the future is the management of a large multi-purpose, multi-user, and multi-author code like OSIRIS. Our current way of management is that there is one code manager who needs to be quite familiar with the details of the code and who manages the “master copy” of the code. This means this code manager has the most updated version of the code. He updates this version with the changes that other authors have made to their versions of the code and then redistributes the

updated version to all users. This model has been workable so far but might become more difficult over time as the code grows. The use of professional code development management software might become necessary in the future to solve this difficulty.

A final point to make is that OSIRIS was designed for distributed computing and more specifically for the currently pervasive type of supercomputers, multiple-instruction-multiple-data parallel computers with local memory. This design can be expected to remain dominant for at least another 5 to 10 years. After that time it is an open question. Depending on how strongly the design for supercomputers will change over time and on how flexible OSIRIS turns out to be the code might either become outdated or it might be able to be adapted to new architectures and remain a helpful tool for a long time to come.

Appendix A

OSIRIS - A Brief User's Guide

A.1 General Information

This appendix will describe how to use OSIRIS and in particular the input file. In addition to the actual input file "os-stdin", which describes the simulation, the code requires five files to be in the same directory as the executable of the code. They contain path-strings which provide the code with direct information on where to find certain files. The path-strings in these files have to conform to the conventions of the computer system the code is running on. The files are:

- path.bin: The path-file that contains the path to the directory with executable.
It is also used to find the "os-stdin" as well as the other path.* files.
- path.mass: The path-file that contains the path to the mass-storage directory.
This mass-storage directory is used for all diagnostic data dumps.
- path.rest: The path-file that contains the path to the restart-file directory. The restart file directory is the directory the restart files are written to or read from.

- path.home: The path in this file is currently not used by the code. This means the file can be empty but it has to exist in order to prevent an accidental crash.
- path.work: The path in this file is also currently not used but the file still needs to exists.

The code also assumes a certain subdirectory structure in the directory given by the path.mass file. The structure that is required is the following one (this list is using Unix notation):

- For the writing of full field data into mass storage: FLD, FLD/B1, FLD/B2, FLD/B3, FLD/E1, FLD/E2, FLD/E3, FLD/J1, FLD/J2, FLD/J3, FLD/RH
- For the writing of averaged data into mas storage: AVE, AVE/b1, AVE/b2, AVE/b3, AVE/e1, AVE/e2, AV/eE3, AVE/j1, AVE/j2, AVE/j3, AVE/rh
- For the writing of particle data: PAR
- For the writing of phase space data for each particle species: PHA, PHA/x2x1, PHA/x3x1, PHA/p1x1, PHA/p2x1, PHA/p3x1, PHA/x3x2, PHA/p1x2, PHA/p2x2, PHA/p3x2, PHA/p1x3, PHA/p2x3, PHA/p3x3, PHA/p2p1, PHA/p3p1, PHA/p3p2. Each of the directories for a specific phase space again needs to have subdirectories for each species. For example: PHA/x2x1/01, PHA/x2x1/02, PHA/x2x1/03, ... PHA/x2x1/[last-species]

Another point that should be mentioned is that even though most parts of OSIRIS have been written for runtime polymorphism some parts have not. Those parts have been written to be compiletime polymorph. This specifically means that the code has to be recompiled when switching from 3D to 2D simulations or when switching

between the different deposition algorithms for 2D simulations. In the first case the parameter "p_x_dim" in the file "os-param.f" has to be set to 2 or 3, depending on which dimensionality is wanted, before recompiling. In the other case an interface in the file "os-spec.f" has to be specified in the following way

```
interface getjr

    module procedure getjr_2d_quadratic ! ISIS algorithm
    !
    module procedure getjr_2d           ! TRISTAN algorithm
    module procedure getjr_3d
end interface
```

in order to use the ISIS method. In order to use the TRISTAN method it has to be changed slightly to

```
interface getjr

    ! module procedure getjr_2d_quadratic ! ISIS algorithm
    module procedure getjr_2d           ! TRISTAN algorithm
    module procedure getjr_3d
end interface
```

This compiletime polymorphism can be easily changed to runtime polymorphism and it will be in a future version of the code.

The actual input file that describes the physics of the simulation, the diagnostic data dumps, and the parallel node-configuration of the simulation is divided into sections corresponding to each object for which data needs to be read in from the input file. Each

of those sections starts and ends with the character “/”. (Note that this is different from the rules for a standard Fortran namelist file.) Between the different sections any kind of comment can be written as long as it doesn’t contain the character “/”.

One of advantages of this type of structured input file is that comments can be scattered throughout the file at the places where they are relevant. The next section is therefore an actual input-file for a 3D run related to the research results presented in chapter 5 with comments on the meaning of the different input variables. The comments are written in such a way that they could also be in an actual input file.

All quantities in the input file are given in dimensionless units which are obtained by normalization with regard to a normalizing frequency, mass, and charge (usually the plasma frequency, the electron mass, and the electron charge), and by normalization with respect to the speed of light. A consistent normalization for all quantities can be derived in this way.

A.2 An Input File Example

----- OSIRIS INPUT DEC -----

This input file is structured into blocks according to the data structures in the program. Each class has its own routine to read in data from this file.

SLASHES are reserved for structuring this file and can not be used for any other purpose.

-----COMMENTS-----

RUN: run655.3d date: 11-01-99

MOTIVATION AND EXPLANATION

This is a 3D version of run651.2d.

It has been suggested that an injection pulse propagating transversely to the pump pulse of a LWFA and passing by behind it should cause a large number of electrons to be trapped in the plasma wave following the pump pulse.

This run is set up to investigate this possibility.

CHANGES COMPARED TO PREVIOUS RUNS

In contrast to run651.2d this run is using a larger grid cell size and timestep size in order to cut down on the computational time required in 3D. The grid size is now such that the laser wavelength corresponds to 14 gridpoints.

$dx(i) = 0.08975$ for all i .

PHYSICS

A laser pump pulse is propagating through a plasma of 4% of critical density. It has a length which is $2\pi c:wp$ in order to create a plasma wake wave.

A second pulse , the injection pulse, is launched in the direction transverse to the first pulse and in such a way as to pass by directly behind the first pulse.

The polarization of both pulses is in the x_1-x_2 plane of the simulation.

The pump pulse propagates 4-5 Rayleigh lengths during the simulation.

RESULTS

----- INPUT DATA -----

-----the node configuration for this simulation-----

```
/ &nl_node_conf  
node_number(1:3) = 16, 2, 2,  
if_periodic(1:3) = .false., .true., .true.,  
/
```

"node_number" is the number of nodes in each direction of the simulation.

The example above has 16 nodes in x1, 2 nodes in x2, and 2 nodes in x3.

The total number of nodes in this simulation is therefore $16 \times 2 \times 2 = 64$.

In this case a full 3D decomposition is used.

The decomposition could be turned into a 2D decomposition by only requesting 1 node in a certain direction

(e.g., `node_number(1:3) = 16, 2, 1,`) or into a 1D decomposition by requesting only 1 node in two directions

(e.g., `node_number(1:3) = 16, 1, 1,`). If all three numbers are 1 the simulation is done on a single node.

"`if_periodic`" is a switch for each of the three directions to turn on periodic boundary conditions. If "`.true.`" is specified here for a certain direction it will override any other boundary condition specified later on in this input file. OSIRIS treats periodic boundary conditions as a aspect of the node configuration because these boundary conditions specify in general that the boundary of one node will communicate with

the boundary of another node. Only under specific circumstances the node will have to "communicate" with its own opposite boundary.

Note that for 2D simulations "node_number" and "if_periodic" have only 2 components each. For example "node_number" would be given by
"node_number(1:2) = 16, 2,".

-----spatial grid-----

```
/ &nl_grid
    nx_p(1:3) = 400, 280, 280,
    coordinates = "cartesian",
/
"nx_p" gives the number of grid cell for the global grid in each direction. The local grid for each node is calculated by nx_p(i) divided node_number(i) for direction i. If for a given direction i nx_p(i) is not a multiple of node_number(i) but nx_p(i) = u x node_number(i) + m then the first m nodes will have u+1 grid cells and the remaining node_number(i) - m nodes will have u grid cells.

"coordinates" is a character string that determines the coordinate system used in a simulation. The currently valid values are "cartesian" and "cylindrical".
```

-----time step and global data dump timestep number-----

```
/ &nl_time_step
    dt      = 0.0513d0,
    ndump  = 136,
/

```

"dt" is the length of a timestep in the simulation.
"ndump" is the number timesteps after which the code checks
for all objects that require writing of data into a file whether
data should be written. In this way it provides a basic measure of
time for diagnostic and restart dumps. The filenames of all dumped
files have a numerical string appended that is based on after how
many multiples of "ndump" timesteps the file was written.

-----restart information-----

```
/ &nl_restart  
  ndump_fac = 1,  file_name  = ' ',  
  if_restart=.false.,  
/  
"ndump_fac" gives the code the information after how  
many multiples of "ndump" to write restart files. In this  
example restart files are written every 136 timesteps. The  
names of the successive files would be "rst-1001", "rst-1002",  
"rst-1003", and so on. If "ndump_fac = 2" then a restart file  
would be written every 272 timesteps and the sequence of  
successive files would be "rst-1002", "rst-1004", "rst-1006",  
and so on. "ndump_fac = 0" turns restart file writing off.  
"if_restart" is a switch that turns on the reading of a restart  
file in beginning of the simulation. if "if_restart=.true.,"  
then the code expects to find restart files in the directory  
given by the file "path.rest" that was discussed in the  
beginning of this chapter.  
"file_name" provides the possibility to attach a prefix to
```

filenames of the restart files.

-----spatial limits of the simulations-----

(note that this includes information about
the motion of the simulation box)

```
/ &nl_space  
  
xmin(1:3) = 0.0000d0 , 0.000d0 , 0.000d0 ,  
xmax(1:3) = 35.9000d0 , 25.130d0 , 25.130d0 ,  
if_move= .true., .false., .false.,  
/
```

"xmin" and "xmax" give the upper and lower boundary
of the global simulation space at the beginning of the
simulation.

"if_move" is a switch for the motion of the space in any of
the three directions. If one of the components is ".true." the
simulation window will move in that direction with the speed
of light. Setting "if_move(i)" to ".true." will override any
other boundary condition specified for these boundaries in
direction i with the exception of the periodic boundary
conditions that are specified in the node-configuration.

For 2D simulation all variables should have one component less.

-----time limits -----

```
/ &nl_time  
  
tmin = 0.0d0, tmax = 104.652d0,  
/  
"tmin" and "tmax" are the initial and final time of the
```

simulation.

-----field solver set up-----

```
/ &nl_el_mag_fld  
b0(1:3)= 0.0d0, 0.0d0, 0.0d0,  
e0(1:3)= 0.0d0, 0.0d0, 0.0d0,  
/
```

"b0" and "e0" are constant external fields that are added to the electromagnetic field.

-----boundary conditions for em-fields -----

```
/ &nl_emf_bound  
type(1:2,1) = 5, 5,  
type(1:2,2) = 5, 5,  
type(1:2,3) = 5, 5,  
/
```

"type" defines here the type of boundary for the electromagnetic field.

The following boundary conditions are currently implemented.

- 1 : boundary moving into the simulation box with c
- 2 : boundary moving outward from the simulation box with c
- 5 : conducting boundary with particle absorption
- 20 : axial b.c. for 2D cylindrically-symmetric coordinates
- 30 : Lindman open-space boundary - limited implementation

Each of the lines above defines boundary conditions for the lower and upper boundaries in one direction. "type(1,3)" for example means the lower boundary in direction 3.

The boundary conditions specified here can be overwritten if

periodic or moving boundaries are specified above. For a 2D simulation the last line "type(1:2,3) = 5, 5," would be removed.

For a 2D cylindrically-symmetric simulation "type(1,2)" needs to be set to 20.

-----diagnostic for electromagnetic fields-----

```
/ &nl_diag_emf

    ndump_fac_all = 1,  file_name_all = ' ',
    ndump_fac_ave = 1,  file_name_ave = ' ',
    n_ave(1:3)      = 16, 10, 10,
    ifdmp_efl(1:3) = .true. , .true. , .true. ,
    ifenv_efl(1:3) = .false. , .false. , .false. ,
    ifdmp_bfl(1:3) = .true. , .true. , .true. ,
    ifenv_bfl(1:3) = .false. , .false. , .true. ,
/

```

This section defines the diagnostic for the electric and magnetic field. There are two different diagnostics. One that writes the full field data of a given field component on each node. The data from this diagnostic has to be merged after the simulation to get the full data set in one file that can then be post-processed further. The second diagnostic averages the data or takes their envelope for a given number of grid cells and then merges the resulting reduced field data at runtime into one field that is written into mass storage.

"ndump_fac_all" determines the times at which the full field data are written and "ndump_fac_ave" determines the times at which the averaged field data are written. If "ndump_fac_all" or "ndump_fac_ave" are 0

then that particular diagnostic is turned off.

"file_name_all" provides the possibility to attach a prefix to filenames of the full data files. "file_name_ave" does the same for the filenames of the averaged data.

"n_ave" gives the number of grid cells that the averaging diagnostic averages over in each direction.

"ifdmp_efl" are switches that turn on both diagnostics for the different components of the electric field if set to ".true.".

"ifdmp_bfl" does the same for the magnetic field components.

"ifenv_efl" decides for each component of the electric field whether the averaging diagnostic really does take the average (.false.) or whether it takes the maximum absolute value (.true.) of the field values in the grid cells determined by "n_ave". "ifenv_bfl" does the same for the magnetic field components.

-----number of particle species-----

```
/ &nl_particles    num_species = 2,    /
```

"num_species" determines the number of species in the simulation.

This section has to be followed by the appropriate kind and number of sections for each species. In this case the data for exactly two species have to be provided below. The code will crash if this is not the case.

-----diagnostics for all particles-----

```
/ &nl_diag_particles  
ndump_fac = 1,  file_name = ' ',
```

```
if_particles_all = .true. ,  
gamma_limit = 6.0d0 ,  
particle_fraction = 1.0d0 ,  
/
```

This section defines a common diagnostic for all particles species.

The particle data are written into a text file. For each particle the number of its species, its position, its momentum, and its simulation charge are written.

"ndump_fac" and "file_name" have the same functions as in the sections where they appeared before but this time they effect the writing of particle data dumps.

"if_particles_all" is a switch that can turn this diagnostic on and off.

"gamma_limit" is a filter for this diagnostic. Only particles with a gamma above "gamma_limit" are written into the dump file.

"particle_fraction" determines which fraction of the particles that are above "gamma_limit" are actually written. The particles that are written are randomly selected from the particles that could be written.

-----information for species 1-----

```
/ &nl_species  
  
num_par_max = 1600000 ,  
rqm=-1.0 ,  
num_par_x(1:3) = 1, 2, 2 ,  
vth(1:3) = 0.0d0 , 0.0d0 , 0.0d0 ,  
vfl(1:3) = 0.0d0 , 0.0d0 , 0.0d0 ,  
den_min = 1.d-5 ,  
if_unneutralized = .false. ,
```

```
num_dgam = 0,
```

```
dgam = 0.0,
```

```
/
```

This section defines the basic variables for the first particle species.

"num_par_max" sets the maximum number of particle on each node for this species.

"rqm" is the mass to charge ratio of the species in normalized units.

For electrons this is -1.

"num_par_x(1:3)" gives the number of particles in each direction in a grid cell. The total number of particles per cell is the product of the numbers in the different directions. The current example has a total of $1 \times 2 \times 2 = 4$ particles per cell. The number of components of "num_par_x" depends on the dimensionality of the run. For example for a 2D simulation a correct setup of 4 particles per cell would be "num_par_x(1:2) = 2, 2,".

"vth(1:3)" defines the thermo-velocity of this particle species in each direction.

"vfl(1:3)" defines a global fluid velocity for this particle species.

This variable is currently not fully implemented.

"den_min" defines an approximate minimum density up to which particles are still initialized in a grid cell. This is necessary in order to avoid the initialization of particle with zero charge. "den_min" should be chosen well below the minimum density of interest for this species.

"if_unneutralized" is currently not fully implemented.

The current algorithms in OSIRIS automatically assume an neutralizing background for newly initialized particles. In future version of OSIRIS "if_unneutralized = .true.," will turn on calculation of the initial

electric and magnetic fields due to a the species that is not neutralized. "dgam" and "num_dgam" are used to specify the acceleration of a beam particle species due to an assumed external field in the x1 direction. "dgam" is here the increase in the gamma of all particles of the species at each timestep of the simulation. "num_dgam" is the number of timesteps for which the gamma of the beam particle species is increased by "dgam". Note that during the timesteps for which this species is being accelerated the transverse momentum of it is not updated.

-----density profile for this species-----

number of points in profile along each direction
/ &nl_num_x num_x = 6, /

"num_x" is the number of points at which the functions that are read in from the next section are defined.

actual profile data

/ &nl_profile

fx(1:6,1) = 1., 1., 0., 0., 1., 1.,
x(1:6,1) = 0., 0.1001, 0.1002, 35.9001, 35.9002, 10000.,
fx(1:6,2) = 0., 0., 1., 1., 0., 0.,
x(1:6,2) = 0., 5.065, 5.0651, 20.13, 20.1301, 25.13,
fx(1:6,3) = 0., 0., 1., 1., 0., 0.,
x(1:6,3) = 0., 5.065, 5.0651, 20.13, 20.1301, 25.13,
/

This section defines the initial density function of this species anywhere in space. The density function is specified as the product of 3 piecewise linear functions. Each of the functions gives the behavior of the density along one of the axes and is given by a number of function values at certain positions. "x(1:6,1) = ..." gives the positions along the x1 direction and "fx(1:6,1) = ..." gives the function values at these positions. The number of points where the function is given for each direction is determined by the variable "num_x" in the previous section. For 2D simulations only functions for 2 directions have to be defined.

-----boundary conditions for this species-----

```
/ &nl_spe_bound
    type(1:2,1) = 5, 5,
    type(1:2,2) = 5, 5,
    type(1:2,3) = 5, 5,
/
"type" defines here the type of boundary for this particle species.
```

The following boundary conditions are currently implemented.

- 1 : boundary moving into the simulation box with c
- 2 : boundary moving outward from the simulation box with c
- 5 : conducting boundary with particle absorption
- 20 : axial b.c. for 2D cylindrically-symmetric coordinates
- 30 : Lindman open-space boundary - limited implementation

The specification of boundary conditions for specific boundaries works in exactly the same way as described above for the boundary conditions of the electromagnetic field.

```

-----diagnostic for this species-----
/ &nl_diag_species

    ndump_fac_pha = 1,  file_name = ' ',
    ps_xmin(1:3) = 0.0,  0.0,  0.0,  ps_pmin(1:3) = -5.0, -10.0, -10.0,
    ps_xmax(1:3) = 0.0, 25.13, 25.13, ps_pmax(1:3) = 25.0, 10.0, 10.0,
    ps_nx(1:3)   = 400, 280, 280,  ps_np(1:3)   = 300, 100, 100,
    if_x2x1     = .true.,
    if_x3x1     = .true.,
    if_p1x1     = .true.,
    if_p2x1     = .true.,
    if_p3x1     = .true.,
    if_x3x2     = .true.,
    if_p1x2     = .true.,
    if_p2x2     = .true.,
    if_p3x2     = .true.,
    if_p1x3     = .true.,
    if_p2x3     = .true.,
    if_p3x3     = .true.,
    if_p2p1     = .true.,
    if_p3p1     = .true.,
    if_p3p2     = .true.,
/

```

This section specifies the diagnostic data dumps for this species. "ndump_fac" and "file_name" work in the same way as described earlier for other diagnostics. The logical variables "if_x2x1", "if_x3x1", ... "if_p3p2" are switches that turn on the writing of a specific

phase space if they are set to ".true.".

"ps_xmin(1:3) = ..." and "ps_max(1:3) = ..." are defining the lower and upper boundary of the ranges of interest for each of the directions in position space. "ps_nx(1:3) = ..." gives the number of points that each direction should be resolved by.

"ps_pmin(1:3) = ...", "ps_pmax(1:3) = ...", and "ps_np(1:3) = ..." specify the same information for the momentum axes of the phase spaces. For example, with the information specified in this input file the simulation will generate phase space data that show the projection of the full phase space of this particle species onto the x2-p1 plane in the area from 0 to 25.13 in x2 and -5 to 25 in p1. This data would be written as a 2D array with a resolution of 280 in x2 and 300 in p1. Note that the projection means that for a phase space plot in a given plane the other position and momentum space directions are integrated over. The numerical values of the array elements are the density of charge with regard to the plane of the given phase space in normalized units.

The upper and lower boundary for x1 are given as zero above. This will not be used directly by the code but triggers the code to use the instantaneous boundaries of the simulation space as the boundaries of the phase space. This can be done for any of the position space axes.

The following information is the information for the second particle species in this simulation. These sections with the information for the second particle species have exactly the same structure as the ones for

the first species and require therefore no additional comments.

-----information for species 2-----

/ &nl_species

```
num_par_max = 600000,  
rqm=-1.0,  
num_par_x(1:3) = 1, 1, 1,  
vth(1:3) = 0.0d0 , 0.0d0 , 0.0d0 ,  
vfl(1:3) = 0.0d0 , 0.0d0 , 0.0d0 ,  
den_min = 1.d-12,  
if_unneutralized = .false.,  
num_dgam = 0,  
dgam = 0.0,
```

/

-----density profile for this species-----

number of points in profile along each direction

/ &nl_num_x num_x = 6, /

actual profile

/ &nl_profile

```
fx(1:6,1) = 0., 0., 1.0d0, 1.0d0, 0., 0.,  
x(1:6,1) = 0., 46.30000, 46.30001, 46.70000, 46.70001, 1000.0,  
fx(1:6,2) = 0., 0., 1.0d-4, 1.0d-4, 0., 0.,  
x(1:6,2) = 0., 12.36500, 12.36501, 12.83000, 12.83001, 25.13,  
fx(1:6,3) = 0., 0., 1.0d-4, 1.0d-4, 0., 0.,  
x(1:6,3) = 0., 12.36500, 12.36501, 12.83000, 12.83001, 25.13,
```

/

-----boundary conditions for this species-----

```

/ &nl_spe_bound

    type(1:2,1) = 5, 5,
    type(1:2,2) = 5, 5,
    type(1:2,3) = 5, 5,
/
-----diagnostic for this species-----
/ &nl_diag_species

    ndump_fac_pha = 1, file_name = ' ',
    ps_xmin(1:3) = 0.0, 0.0, 0.0, ps_pmin(1:3) = -5.0, -10.0, -10.0,
    ps_xmax(1:3) = 0.0, 25.13, 25.13, ps_pmax(1:3) = 25.0, 10.0, 10.0,
    ps_nx(1:3) = 400, 280, 280, ps_np(1:3) = 300, 100, 100,
    if_x2x1 = .true.,
    if_x3x1 = .true.,
    if_p1x1 = .true.,
    if_p2x1 = .true.,
    if_p3x1 = .true.,
    if_x3x2 = .true.,
    if_p1x2 = .true.,
    if_p2x2 = .true.,
    if_p3x2 = .true.,
    if_p1x3 = .true.,
    if_p2x3 = .true.,
    if_p3x3 = .true.,
    if_p2p1 = .true.,
    if_p3p1 = .true.,
    if_p3p2 = .true.,
/

```

```
-----number of pulses-----  
/ &nl_pulse_sequence num_pulses = 2, /  
  
"num_pulses" determines the number of laser pulses launched in the  
simulation. This section has to be followed by the appropriate number of  
of sections which means one for each laser pulse. In this case the data  
for exactly two pulses have to specified below. The code will crash if  
this is not the case.
```

```
-----information for pulse 1-----
```

```
/ &nl_pulse  
  iflaunch = .true.,  
  wavetype=1,  
  w0=3.0d0,      rise=3.14,      fall=3.14,      length=0.0,  
  vosc=1.0d0,    rkfp=5.0,       pol=0.0,       phase = 0.0d0,  
  start=6.29d0,  focus=-30.0,   offset(1:2)=0.0, 0.0, time=0.0,  
 /
```

The following information for a laser pulse is specified in this
section:

"iflaunch" is a switch that can turn the laser pulse on or off.
"wavetype" allows to choose between different kinds of laser pulses.
Currently pulses propagating in x1 (wavetype=1) and a type propagating
in x2 a (wavetype=2) are implemented. In both cases the pulse shape
is approximately Gaussian in the transverse directions.
"w0" specifies the spotsize of the laser in the focal plane.

"rise" gives the distance over which the pulse rises from 0 at the front of the pulse to the peak intensity. The shape of the rise is approximately Gaussian.

"length" defines the length over which the pulse has its peak intensity after reaching it at the end of the rise at the front.

"fall" gives the distance over which the pulse falls off from the peak intensity to 0 at the back of the pulse. The shape of the fall off is approximately Gaussian.

"vosc" specifies the maximum vector potential of the laser pulse at the time it is initialized.

"rkkp" is the wavenumber of the laser pulse in normalized units.

"pol" specifies the plane of polarization of the laser. "pol=90" corresponds to a laser polarized in x3. "pol=0" is laser with a polarization changed by 90 degrees from x3.

"phase" specifies an overall phase change to the laser. This can be used to generate circularly polarized lasers by superposing two laser with different polarization and phase.

"start" gives the position of the front of the laser with respect to the side of the box that the pulse is moving towards. In the example above this means that the front of the pulse is at an x1 position of 29.61 since the simulation window has a length of 35.9 in x1.

"focus" determines the position of the focal plane of the laser in the same way as "start" specifies the position of the front of the laser. Note that in the example above this means that the focal plane is outside the initial box.

"offset(1:2)" specifies an offset of the pulse transverse to the propagation direction from the center of the simulation. If

"offset(1:2)=0.0, 0.0," then the pulse is centered in the simulation box with respect to the transverse coordinates. The components 1 and 2 are referring to x2 and x3 for a pulse propagating in x1 and to x1 and x3 for a pulse propagating in x2.

In a 2D simulation there is only one transverse coordinate and the statement becomes "offset(1:1)=0.0,".

"time" specifies the time at which the pulse is initialized in the simulation.

-----information for pulse 2-----

```
/ &nl_pulse
  iflaunch = .true.,
  wavetype=2,
  w0=3.0d0,      rise=1.57,      fall=1.57,      length=0.0,
  vosc=1.8,       rkfp=5.0,       pol=0.0,       phase = 0.0d0,
  start=21.065,   focus=12.565,   offset(1:2)=10.95,0.0, time=19.90,
/
The section for the second pulse has exactly the same structure as the section for the first one and requires no additional comments.
```

-----smoothing for currents-----

```
/ &nl_smooth
  ifsmooth(1)      = .false.,
  smooth_level(1)  = 3,
  swfj(1:3,1,1)   = 1,2,1,
```

```

swfj(1:3,2,1)    = 1,2,1,
swfj(1:3,3,1)    = 1,2,1,

ifsmooth(2)      = .false.,
smooth_level(2) = 3,
swfj(1:3,1,2)    = 1,2,1,
swfj(1:3,2,2)    = 1,2,1,
swfj(1:3,3,2)    = 1,2,1,

ifsmooth(3)      = .false.,
smooth_level(3) = 3,
swfj(1:3,1,3)    = 1,2,1,
swfj(1:3,2,3)    = 1,2,1,
swfj(1:3,3,3)    = 1,2,1,
/

```

This section specifies the smoothing of the current density. It does this for each direction separately. The smoothing is done in position space using weighting factors.

For each direction i:

"ifsmooth(i) = .true." switches the smoothing on.
 smooth_level(i) = ..., gives the number of times the smoothing is iteratively done over the nearest neighbors. "swfj(1:3,1,i)" gives the weighting factors for the first smoothing iteration,
 "swfj(1:3,2,i)" gives the weighting factors for the second smoothing iteration, and so on. For more details see Ref.

[47].

```

-----diagnostic for currents-----
/ &nl_diag_phy_field

ndump_fac_all = 1, file_name_all = ' ',
ndump_fac_ave = 0, file_name_ave = ' ',
n_ave(1:3) = 1, 1, 1,
ifdmp_phy_field(1:3) = .true. , .true. , .true. ,
/

```

This section specifies the diagnostic data dumps for the current. "ndump_fac_all" and "file_name_all" work in the same way as described earlier for the electromagnetic field diagnostics. "ndump_fac_ave", "file_name_ave", and "n_ave" are currently not implemented but are going to have the same functionality in future versions of the code as described for the electromagnetic field. "ifdmp_phy_field(1:3)" are switches for turning the diagnostic on and off for the different components of the current.

The following two sections define the smoothing and the diagnostic for the charge density in the same way as described above for the current. The only difference is that the variable "ifdmp_phy_field(1:1)" only needs one component to work as a switch for the charge density diagnostic since the charge density is a scalar.

```

-----smoothing for charge-----
/ &nl_smooth
```

```

ifsmooth(1)      = .false. ,
smooth_level(1) = 3,
swfj(1:3,1,1)   = 1,2,1,
swfj(1:3,2,1)   = 1,2,1,
swfj(1:3,3,1)   = 1,2,1,

ifsmooth(2)      = .false. ,
smooth_level(2) = 3,
swfj(1:3,1,2)   = 1,2,1,
swfj(1:3,2,2)   = 1,2,1,
swfj(1:3,3,2)   = 1,2,1,

ifsmooth(3)      = .false. ,
smooth_level(3) = 3,
swfj(1:3,1,3)   = 1,2,1,
swfj(1:3,2,3)   = 1,2,1,
swfj(1:3,3,3)   = 1,2,1,
/
-----diagnostic for charge-----
/ &nl_diag_phy_field
ndump_fac_all = 1,  file_name_all = ' ',
ndump_fac_ave = 0,  file_name_ave = ' ',
n_ave(1:3)      = 1, 1, 1,
ifdmp_phy_field(1:1) = .true.,
/

```

Bibliography

- [1] T. Tajima and J. M. Dawson. Laser electron accelerator. *Phys. Rev. Lett.*, 43:267, 1979.
- [2] C. Joshi, W. B. Mori, T. Katsouleas, J. M. Dawson, J. M. Kindel, and D. W. Forslund. Ultrahigh gradient particle acceleration by intense laser-driven plasma density waves. *Nature*, 311:525, 1984.
- [3] Pisin Chen, J. M. Dawson, Robert W. Huff, and T. Katsouleas. Acceleration of electrons by the interaction of a bunched electron beam with a plasma. *Phys. Rev. Lett.*, 54:693, 1985.
- [4] R. Williams et al. *Laser Part. Beams*, 8:427, 1990.
- [5] T. Katsouleas, C. Joshi, J. M. Dawson, F. F. Chen, C. Clayton, W. B. Mori, C. Darrow, and D. Umstadter. Plasma accelerators. In C. Joshi and T. Katsouleas, editors, *Laser Acceleration of Particles*, Vol. 130 of Proc. AIP Conf., page 63, New York, 1985. Amer. Inst. Phys.
- [6] P. Mora. *Phys. Fluids B*, 4:1630, 1992.
- [7] P. Mora. *J. Appl. Phys.*, 71:2087, 1992.
- [8] E. Esarey and Mark Piloff. Trapping and acceleration in nonlinear plasma waves. *Phys. Plasmas*, 2:1432, 1995.
- [9] E. Esarey, J. Krall, and P. Sprangle. *Phys. Rev. Lett.*, 5:2887, 1990.
- [10] W. B. Mori, C. D. Decker, D. E. Hinkel, and T. Katsouleas. Raman forward scattering of short-pulse high-intensity lasers. *Phys. Rev. Lett.*, 72:1482, 1994.
- [11] N. E. Andreev, L. M. Gorbunov, V. I. Kirsanov, A. A. Pogosova, and R. R. Ramazashvili. The theory of laser self-resonant wake field excitation. *Physica Scripta*, 49:101, 1994.
- [12] E. Esarey, P. Sprangle, J. Krall, and A. Ting. Overview of plasma-based accelerator concepts. *IEEE Trans. Plasma Sci.*, 24:252, 1996 and references therein.
- [13] W. P. Leemans and E. Esarey. Plasma based acceleration concepts. In W. Lawson, C. Bellamy, and D. F. Brosius, editors, *Advanced Accelerator Concepts*, volume 472 of *Proc. AIP Conf.*, pages 174–190, New York, 1999. Amer. Inst. Phys.

- [14] M. Stanley Livingston. *Particle Accelerators: A Brief History*. Havard University Press, Cambridge, Massachusetts, 1969.
- [15] R. Assmann, C. Joshi, and W. Leemans amd S. Siemann. One gev beam acceleration in a one meter long plasma cell, 1997. Proposal to the Stamford Linear Accelerator Center.
- [16] D. Strickland and G. Mourou. Compression of amplified chirped optical pulses. *Opt. Commun.*, 56:219–221, 1985.
- [17] K.-C. Tzeng. *Realistic Modelling of Short-Pulse High-Intensity Lasers in Underdense Plasmas Using Particle-In-Cell Simulations*. PhD thesis, Unversity of California, Los Angeles, 1998.
- [18] P. Muggli, J. R. Hoffman, K. A. Marsh, S. Wang, C. E. Clayton, T. Katsouleas, and C. Joshi. Lithium plasma sources for acceleration and focusing of ultra-relativistic electron beams. In *1999 Proceedings of the Particle Accelerator Conference*, New York, 1999.
- [19] D. W. Forslund, J. S. Junkins, and C. A. Wingate. Wave++: A distributed object-oriented plasma simulation code. In *US-Japan Workshop on Advances in Simulation Techniques Applied to Plasmas ans Fusion*, pages 1–4, Los Angeles, California, September 26-28, 1990.
- [20] S. W. Haney and J. A. Crottinger. C++ proves useful in writing a tokamak systems code. *J. Comp. in Phys.*, 6(5):450–455, 1991.
- [21] J. V. Reynders, D. W. Forslund, P. J. Hinken, M. Tholburn, D. G. Kilman, and W. F. Humphrey. Object-oriented particle simulation on parallel computers. In *Object Oriented Numerics Conference*, pages 266–279, 1994.
- [22] In G. V. Wilson and P. Lu, editors, *Parallel Programming Using C++*, Scientific and Engineering Computation Series, Cambridge, 1996. MIT Press.
- [23] J. P. Verboncoeur, A. B. Langdon, and N. T. Gladd. An object-oriented electromagnetic pic code. *Comp. Phys. Com.*, 87:199–211, 1995.
- [24] C. D. Norton. *Object-Oriented Programming Paradigms In Scientific Computing*. PhD thesis, Rensselaer Polytechnic Institute, 1996.
- [25] V. K. Decyk, C. D. Norton, and B. K. Szymanski. How to express c++ concepts in fortran90. *Scientific Programming*, 6(4):363, 1998.
- [26] V. K. Decyk, C. D. Norton, and B. K. Szymanski. How to support inheritance and run-time polymorphism in fortran 90. *Comp. Phys. Com.*, 115:9–17, 1998.
- [27] J. Qiang, R. D. Ryne, S. Habib, and V. Decyk. An object-oriented parallel particle-in-cell code for beam dynamics simulation in linear accelerators. In *Proceedings of Supercomputing Conference 99*, 1999.

- [28] R. Fedele et al. *Phys. Rev. A*, 33:4412, 1986.
- [29] L. M. Gorbunov et al. *Phys. Rev. Lett.*, 76:2495, 1996.
- [30] W. B. Mori. On beat wave excitation of relativistic plasma waves. *Bull. Am. Phys. Soc.*, 32:1799, 1987.
- [31] Dwight R. Nicholson. *Lasers*. John Wiley and Sons, New York, 1988.
- [32] P. Sprangle et. al. *Phys. Rev. A*, 36:2773, 1987.
- [33] B. J. Duda and W. B. Mori. Variational principle approach to short-pulse laser-plasma interactions in three dimensions. *Phys. Rev. E*, 61, no. 2, 2000.
- [34] W. P. Leemans, C. W. Siders, E. Esarey, N. E. Andreev, G. Shvets, and W. B. Mori. Plasma guiding and wakefield generation for second-generation experiments. *IEEE Trans. Plasma Sci.*, 24:331–342, 1996.
- [35] C. Max, J. Arons, and A. B. Langdon. Self-modulation and self-focusing of electromagnetic waves in plasmas. *Phys. Rev. Lett.*, 33:209, 1974.
- [36] P. Sprangle, C. M. Tang, and E. Esarey. Relativistic self-focusing of short-pulse radiation beams in plasmas. *IEEE Trans. Plasma Sci.*, PS-15:145–153, 1987.
- [37] W. B. Mori, C. Joshi, J. M. Dawson, D. W. Forslund, and J. M. Kindel. Evolution of self-focusing of intense electromagnetic waves in plasma. *Phys. Rev. Lett.*, 60:1298, 1988.
- [38] P. Sprangle, E. Esarey, and A. Ting. Nonlinear interaction of intense laser pulses in plasmas. *Phys. Rev. A*, 41:4463, 1990.
- [39] P. Sprangle, E. Esarey, and A. Ting. Nonlinear theory of intense laser-plasma interactions. *Phys. Rev. Lett.*, 64:2011, 1990.
- [40] T. M. Antonsen, Jr. and P. Mora. Self-focusing and raman scattering of laser pulses in tenuous plasmas. *Phys. Rev. Lett.*, 69:2204, 1992.
- [41] J. D. Lawson. *The Physics of Charged Particle Beams*. Oxford University Press, New York, 1988.
- [42] P. Sprangle, E. Esarey, A. Ting, and G. Joyce. Laser wakefield acceleration and relativistic optical guiding. *Appl. Phys. Lett.*, 53:2146, 1988.
- [43] A. I. Akhiezer and R. V. Polovin. Theory of wave motion of an electron plasma. *Sov. Phys. JETP*, 3:696–705, 1956.
- [44] W. B. Mori and T. Katsouleas. Wavebreaking of longitudinal plasma oscillations. *Physica Scripta*, T30:127–133, 1989.
- [45] T. Katsouleas, J. J. Su, C. Joshi, W. B. Mori, J. M. Dawson, and S. Wilks. SPIE Conf. Proc. OE/LASE '89, Los Angels, CA Jan 16-20, page 428, 1989.

- [46] J. B. Rosenzweig, B. Breizman, and T. Katsouleas J. J. Su. Acceleration and focusing of electrons in two-dimensional non-linear plasma wakefields. *Phys. Rev. A*, 44:R6189–R6192, 1991.
- [47] C. K. Birdsall and A. B. Langdon. *Plasma Physics via Computer Simulation*. Adam Hilger, New York, 1991.
- [48] K.-C. Tzeng, W. B. Mori, and C. D. Decker. Anomalous absorption and scattering of short-pulse high-intensity lasers in underdense plasmas. *Phys. Rev. Lett.*, 76:3332, 1996.
- [49] John Dawson. Particle simulations of plasmas. *Rev. mod. Phys.*, 55:403–447, 1983.
- [50] R.L. Morse and C. W. Nielson. Numerical simulations of the weibel instability in one and two dimensions. *Phys. Fluids*, 14:830, 1970.
- [51] J. Villasenor and O. Buneman. Rigorous charge conservation for local electromagnetic field solvers. *Comp. Phys. Com.*, 69:306, 1992.
- [52] J. W. Eastwood. The virtual particle electromagnetic particle-mesh method. *Comp. Phys. Com.*, 64:252, 1990.
- [53] O. Buneman, T. Neubert, and K. Nishikawa. Solar wind-magnetosphere interaction as simulated by a 3-d em particle code. *IEEE Trans. Plasma Sci.*, 20:810, 1992.
- [54] M. Metcalf and J. Reid. *Fortran 90 Explained*. Oxford University Press, New York, 1994.
- [55] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorenzen. *Object-Oriented Modeling and Design*. Prentice Hall, Englewood Cliffs, NJ, 1991.
- [56] University of Tennessee. The mpi language specification. Special issue of International Journal of Supercomputer Applications.
- [57] W. Gropp, E. Lusk, and Anthony Skjellum. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*. The MIT Press, Cambridge, Massachusetts, 1997.
- [58] P. M. Lyster, P. C. Liewer, and V. K. Decyk R. D. Ferraro. Implementation and characterization of three-dimensional particle-in-cell codes on multiple-instruction-multiple-data massively parallel supercomputers. *Comp. in Phys.*, No.4, 9:420–432, 1999.
- [59] D. Umstadter, J. K. Kim, and E. Dodd. Laser injection of ultrashort electron pulses into wakefield plasma waves. *Phys. Rev. Lett.*, 76:2073, 1996.
- [60] A. Ogata. *IEEE Trans. Plasma Sci.*, 24:453, 1996.
- [61] J. S. Fraser and R. L. Sheffield. High-brightness injectors for rf-driven free electron lasers. *I.E.E.E J. Quant. Elec.*, **QE-23**:1489, 1987.
- [62] M. Reiser. *Theory and Design of Charged Particle Beams*. John Wiley and Sons, New York, 1994.

- [63] T. Katsouleas, S. Wilks, P. Chen, J. M. Dawson, and J. J. Su. Beam loading in plasma accelerators. *Particle Accelerators*, 22:81, 1987.
- [64] P. Sprangle, E. Esarey, J. Krall, and G. Joyce. Propagation and guiding of intense laser pulses in plasmas. *Phys. Rev. Lett.*, 69:2200, 1992.
- [65] C. G. Durfee III and H. M. Milchberg. Light pipe for high intensity laser pulses. *Phys. Rev. Lett.*, 71:2409, 1993.
- [66] D. Umstadter. *Bull. Am. Phys. Soc.*, 42:1184, 1997.
- [67] E. Esarey, R. F. Hubbard, W. P. Leemans, A. Ting, and P. Sprangle. Electron injection into plasma wake fields by colliding laser pulses. *Phys. Rev. Lett.*, 79:2682, 1997.
- [68] edited by W. B. Mori. Special issue, *IEEE Trans. Plasma Sci.*, ps-21 (1), 1993.
- [69] M. Tabek, J. Hammer, M. E. Glinsky, W. L. Kruer, S. C. Wilks, J. Woodworth, E. M. Campbell, M. D. Perry, and R. J. Mason. Ignition and high gain with ultrapowerful lasers. *Phys. Plasmas*, 1:1626, 1994.
- [70] W. B. Mori. The physics of the nonlinear optics of plasma at relativistic intensities for short-pulse lasers. *IEEE J. Quantum Electron.*, 33:1942, 1997.
- [71] E. Esarey, J. Krall, and P. Sprangle. Envelope analysis of intense laser pulse self-modulation in plasmas. *Phys. Rev. Lett.*, 72:2887, 1994.
- [72] P. Sprangle et al. *Phys. Rev. Lett.*, 73:3544, 1994.
- [73] G. Shvets and J. S. Wurtele. *Phys. Rev. Lett.*, 73:3540, 1994.
- [74] K.-C. Tzeng, W. B. Mori, and T. Katsouleas. Electron beam characteristics from laser-driven wave breaking. *Phys. Rev. Lett.*, 79:5258, 1997.
- [75] K.-C. Tzeng and W. B. Mori. Suppression of electron ponderomotive blowout and relativistic self-focusing by the occurrence of raman scattering and plasma heating. *Phys. Rev. Lett.*, 81:104, 1998.
- [76] A. Pukhov and J. Meyer-ter-Vehn. *Phys. Plasmas*, 5:1880, 1998.
- [77] B. J. Duda and W. B. Mori. *Bull. Am. Phys. Soc.*, 43:1658, 1998.
- [78] D. Anderson and M. Bonnedal. *Physics of Fluids*, 22:105, 1979.
- [79] D. H. Whittum, W. M. Sharp, S. S. Yu, M. Lampe, and G. Joyce. Electron-hose instability in the ion-focused regime. *Phys. Rev. Lett.*, 67:991, 1991.
- [80] A. Modena, Z. Najmudin, A. E. Dangor, C. E. Clayton, K. A. Marsh, C. Joshi, V. Malka, C. B. Darrow, C. Danson, D. Neely, and F. N. Walsh. Electron acceleration from the breaking of relativistic plasma waves. *Nature*, 377:606, 1995.

- [81] D. Gordon, K.-C. Tzeng, C. E. Clayton, A. E. Dangor, V. Malka, K. A. Marsh, A. Modena, W. B. Mori, P. Muggli, Z. Najmudin, D. Neely, C. Danson, and C. Joshi. Observation of electron energies beyond the linear dephasing limit from a laser-excited relativistic plasma wave. *Phys. Rev. Lett.*, 80:2133, 1998.
- [82] D. Umstadter, S.-Y. Chen, A. Maksimchuk, G. Mourou, and R. Wagner. Nonlinear optics in relativistic plasmas and laser wake field acceleration of electrons. *Science*, 273:472, 1996.
- [83] R. Wagner, S.-Y. Chen, A. Maksimchuk, and D. Umstadter. Electron acceleration by a laser wakefield in a relativistically self-guided channel. *Phys. Rev. Lett.*, 78:3125, 1997.
- [84] C. A. Coverdale, C. B. Darrow, C. D. Decker, W. B. Mori, K.-C. Tzeng, K. A. Marsh, C. E. Clayton, and C. Joshi. Propagation of intense subpicosecond laser pulses through underdense plasmas. *Phys. Rev. Lett.*, 74:4659, 1995.
- [85] A. Ting, C. I. Moore, K. Krushelnick, C. Manka, E. Esarey, P. Sprangle, R. Hubbard, H. R. Burris, R. Fischer, and M. Baine. Plasma wakefield generation and electron acceleration in a self-modulated laser wakefield accelerator experiment. *Phys. Plasmas*, 4:1889, 1997.
- [86] J. C. Adam. private communication.
- [87] M. J. Hogan, R. Assmann, F. J. Decker, R. Iverson, P. Raimondi, S. Rokni, R. H. Siemann, D. Walz, D. Whittum, B. Blue, C. E. Clayton, E. Dood, R. G. Hemker, C. Joshi, K. A. Marsh, W. B. Mori, S. Wang, T. Katsouleas, S. Lee, P. Muggli, P. Catravas, S. Chattopadhyay, E. Esarey, W. P. Leemans, and P. Wolfbeyn. E-157: A meter-long plasma wakefield acceleration experiment using the 30GeV electron beam from the SLAC lineac.
- [88] S. Lee, T. Katsouleas, R. G. Hemker, W. B. Mori, E. Esarey, and C. Schroeder. Simulations of a meter long plasma wakefield accelerator.
- [89] W. B. Mori. private communication.
- [90] W. B. Mori, J. J. Su, and T. Katsouleas. Plasma physics at the final focus. *Particle accelerators*, 31:1229–1234, 1990.
- [91] J. D. Jackson. *Classical Electrodynamics*. John Wiley and Sons, Inc., New York, 1975.
- [92] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comp. Phys.*, No. 1, 82:64–84, 1989.
- [93] G. L. Bryan. Fluids in the universe: Adapative mesh refinement in cosmology. *Computing in Science and Engineering*, page 46, 1999.
- [94] P. Mora and T. M. Antonsen. Kinetic modeling of intense, short laser pulses propagating in tenuous plasmas. *Phys. Plasmas*, 4(1):217, 1997.