SIMULATION OF NEUTRON GENERATION FROM LASER-DRIVEN FUSION IN A LIQUID D_2O SHEET USING NOVEL WARPX MODULE

THESIS

Colton R Stoner, Second Lieutenant, USAF

AFIT-ENP-MS-23-M-106

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENP-MS-23-M-106

SIMULATION OF NEUTRON GENERATION FROM LASER-DRIVEN FUSION IN A LIQUID D_2O SHEET USING NOVEL WARPX MODULE

THESIS

Presented to the Faculty

Department of Engineering Physics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Nuclear Physics

Colton R Stoner, B.S. in Physics

Second Lieutenant, USAF

March 19, 2023

AFIT-ENP-MS-23-M-106

SIMULATION OF NEUTRON GENERATION FROM LASER-DRIVEN FUSION IN A LIQUID D_2O SHEET USING NOVEL WARPX MODULE

THESIS

Colton R Stoner, B.S. in Physics
Second Lieutenant, USAF

Committee Membership:

Anil K Patnaik, Ph.D
Chair

Michael L Dexter, Ph.D
Member

Christopher M Orban, Ph.D
Member

AFIT-ENP-MS-23-M-106

Abstract

This thesis provides an early look at a new particle simulation module. Using WarpX's new nuclear fusion module, this work attempts to model an experimental system at the Extreme Light Laboratory (ELL) at the Air Force Institute of Technology (AFIT) with WarpX and draw conclusions about fusion products from resulting simulations. Recently, a table-top, high repetition rate, mixed radiation source was demonstrated at the ELL employing a high intensity laser (HIL) to fuse the deuterium nuclei present in a unique liquid target of heavy water. Analysis of the simulations predicted an isotropic output of neutrons from deuterium-deuterium fusion. These simulated neutrons were created in tens of femtoseconds from dense bodies of deuterons that were perturbed by the laser. However, it was found that the simulation used to draw these conclusions does not properly resolve a key stability condition in plasma simulation. More stability could be brought to the simulation by increasing spatial resolution, reducing the density of the simulated heavy water, and creating a method to determine the presence and severity of numerical heating in a fusion-enabled particle simulation, amongst many possible options. Therefore, more work is needed to confirm these findings, and there are many options available to do so.

Acknowledgements

I'd like to profusely thank Chris Orban for his guidance and assistance. His ready attitude and ever-presence were an immense boon to the direction and outright completion of this thesis. I would also like to thank my advisor, Anil Patnaik, for his guidance; his wisdom was and is always appreciated and his patience with me through the entirety of this master's program was impressive. There are many more members of the Extreme Light Group (ELG) I'd like to thank, including Michael Dexter for introducing this project to me and getting me on the team, Joe Smith for his assistance with WarpX, simulation, and plasma physics, and so many more people, like Enam Chowdhury, Kyle Frische, Benjamin Knight, Juan Manfredi, and Connor Gautam for their help with both this work and with my graduate school experience as a whole.

I am grateful for my family and friends, without whose support this endeavor would surely have failed. I look forward to returning my gratitude and supporting all of you whenever you may need it.

Colton R Stoner

Table of Contents

Page

Page

List of Figures

List of Tables

SIMULATION OF NEUTRON GENERATION FROM LASER-DRIVEN FUSION IN A LIQUID D_2O SHEET USING NOVEL WARPX MODULE

I. Introduction

WarpX is an advanced electromagnetic and electrostatic particle-in-cell (PIC) code. It has been used to model a chain of plasma accelerator stages, to investigate the three-dimensional structure of pulsar magnetospheres, to develop a virtual electro-magnetic (EM) detector for use in particle accelerator simulations, and many more applications [1] [2] [3]. An exciting, novel application of WarpX is simulation of neutron generation with its new nuclear fusion module. With the National Ignition Facility (NIF) achieving ignition on the fifth of December 2022, simulation of fusion events can be extremely relevant in the future. Given the time, effort, and expenses it takes to plan and execute a shot at NIF, it would come as no surprise if experts turned to simulation to make sure that a shot makes sense and is worthwhile.

This thesis computationally investigates a mixed x-ray, electron, ion, and neutron radiation source using WarpX, with a focus on neutrons. The genesis of this work begins at the Air Force Institute of Technology (AFIT) with the Extreme Light Laboratory (ELL). The ELL is at the forefront of high repetition rate x-ray and neutron radiation generation and recently demonstrated a mixed radiation source using a 1 kHz repetition rate, 10^{19} Wcm^{-2} intensity laser and a heavy water target [4]. This work uses WarpX to model the neutron generation of this high intensity laser (HIL) system of the Extreme Light Group (ELG). Previously, members of the ELG confirmed the neutron generation of the system. The next step was to increase neutron generation and to induce anisotropy in outgoing neutron flux. Pursuant

with this effort, the group added a heavy water ice catcher to the setup, hoping to increase neutron flux in the direction of laser propagation. Up until this point, neutron bubble detectors had been used to characterize the neutron generation of the system, and naturally they were recruited for this anisotropy study as well. However, bubble detectors generally fall short for ascertaining the nature of neutron flux, and are primarily used for dosimetry. That is, bubble detectors only give a rough idea of how much radiation has impinged an area over a large amount of time. They are also known to degrade over time, which can drive down accuracy.

It was therefore determined that a different, more reliable detector should be used to determine isotropy or anisotropy of the neutron generation for the mixed source. With a reliable lifespan and pulse shape discrimination (PSD) capabilities to resolve individual neutron events on the detector, Eljen's EJ-309 organic scintillator detectors were a good choice for what the group needed to help answer their anisotropy question and give the results from the bubble detectors a bit more legitimacy. So, with the help of Juan Manfredi, Connor Gautam, Kyle Frische, Ben Knight, and others, an experiment lead by the author was conducted using three EJ-309 detectors placed at different viewing angles of the mixed source generated by the HIL impacting a heavy water target. The experiment was a success, but its results seemed less conclusive than was hoped. It was determined that the source was isotropic or more precisely that the source was not anisotropic to any degree of certainty.

Because the results from the experiment were inconclusive, WarpX was recruited as a means to investigate in simulation what had been attempted in experiment. Fortunately, WarpX contributors Neïl Zaïm and Remi Lehe had recently expanded WarpX's collision capabilities by adding a nuclear fusion module. This new development allowed probing of the nature of the HIL while the system was getting upgrades and while it was in use for other endeavors.

1.1 Motivation

The repetition rate of the HIL is able to reach frequencies of up to 1 kHz, while comparable tabletop sources generally operate in the 1-10 Hz range [5] [6]. The tabletop neutron source is novel because of this high repetition rate. This allows for more data to be taken in novel experiments and in future applications. For example, neutron spectroscopy has applications in studying minerals [7], while neutron radiography and tomography have been used to study the fabrication techniques of Renaissance bronzes [8], additive manufacturing processes [9], and the condition of irradiated nuclear fuel [10].

Having demonstrated an energetic source of electrons and x-rays, the ELL at AFIT continues to use its HIL for high repetition rate laser-driven fusion (LDF) and characterize the generated mixed source of radiation [5] [4]. Mirroring this experimental characterization effort, a simulation that captures the dynamics of the system would be beneficial in the effort to characterize the neutron source for closer study.

Additionally, WarpX's new nuclear fusion module was still in testing stages at the beginning of this project. This new module added to the goals set for this work; one goal was to emulate experimental conditions as much as was reasonable. Adequate emulation would allow better comparison with experiment. The other goal was to test the new nuclear fusion model to see if its capabilities were useful to the LDF characterization effort.

1.2 Objectives

As previously outlined, the goals of this project are to test the new fusion module of WarpX, corroborate results from the isotropy experiment through simulation, and advance understanding of neutron generation in a high-intensity laser environment. A basic understanding of the experiment and its results will be provided, but the

focus of this work remains in simulation.

To support the results from the isotropy experiment and to increase understanding of neutron generation from fusion with WarpX, the following three questions will be addressed with simulation: One, how does the number of neutrons evolve with time? Two, where do neutrons generate? Three, where do neutrons propagate? Or, to rephrase in the lens of the related experimental portion of this work, is the neutron generation isotropic?

II. Experimental Study of the Isotropic Nature of Neutron Generation

Although a primary focus of this thesis is the computational challenges of modeling laser-driven fusion (LDF) with a deuterated liquid target, an experimental background is provided to set the stage for the computational aspects of this work. The specifications of the laser and target informed how they were modeled in the simulation, and the method by which the neutron output was previously characterized sets up well the motivation for using computational methods.

2.1 D-D Fusion and the Heavy Water Target

The 780 nm wavelength high intensity laser (HIL) in the Extreme Light Laboratory (ELL) has the capability of operating at an energy of 7-11 mJ, a pulse duration of 40 fs, and an intensity of $10^{19} - 10^{20} W cm^{-2}$. This laser was used to create a plasma from a sheet of liquid heavy water, D_2O. To create such a plasma at kilohertz repetition rate, the target was continuously replenished by two water pumps that were arranged as described in K.M. George et al. [11].

In such plasma interactions, the two most common deuterium-deuterium fusion reactions are nearly equally likely [12]:

$$\begin{aligned} {}^2H +^2 H \rightarrow^3 He + n + 3.3\ MeV \\ {}^2H +^2 H \rightarrow^3 H + p + 4.0\ MeV \end{aligned} \tag{1}$$

This work will only be concerned with the neutron reaction. In this reaction, the neutron carries off 2.45 MeV out of the total 3.3 MeV of energy assuming no excess energy is brought into the reaction.

2.2 The Ice Catcher

The plasma created by the laser is thought to emit neutrons isotropically because the energy of the deuterons in the plasma is assumed to be dominated by the thermal energy imparted to the target by the laser. Assuming this thermal component dominates the plasma, the deuterons should have no directional preference on average.

Preliminary fusion experiments that generated this plasma revealed that neutrons were being emitted at such a low rate that it was difficult to characterize in detail. In an attempt to increase neutron flux, an ice catcher was added in the forward direction of the laser downstream of the target; see Figure 1 for an experimental setup diagram. This ice catcher, made from heavy water ice, was thought to add anisotropy by increasing the neutron flux in the forward direction of the laser. In theory, the extra deuterons provided by the ice catcher should function as secondary targets for the deuterons accelerated by the laser. If the ice catcher was working as intended, then the neutron source should show anisotropy since the forward traveling deuterons were more likely to interact with the ice catcher and produce neutrons.

However, initial tests of the neutron flux indicated that it was not anisotropic, and thus the ice catcher was not affecting the neutron flux considerably. One of the goals of this work was to test whether or not a heavy water ice catcher can work to induce anisotropy through simulation.

2.3 Detectors and Pulse Shape Discrimination

The detectors used to characterize the neutron output of the experimental system in the most recent experiment were Eljen EJ-309 organic scintillator detectors. Three of these detectors were used in an isotropy study led by the author in an attempt to determine whether or not the catcher was performing as intended; an experimental setup is pictured in Figure 1. The isotropy study gathered data effectively using the

tools it had available but was limited in scope. With such an experimental setup, it was possible to gather data at only three points, and due to the nature of the detection system, raw data regarding neutrons could not be analyzed directly. Instead, each instance of a neutron impinging on the detector had to be inferred from pulse shape discrimination (PSD).

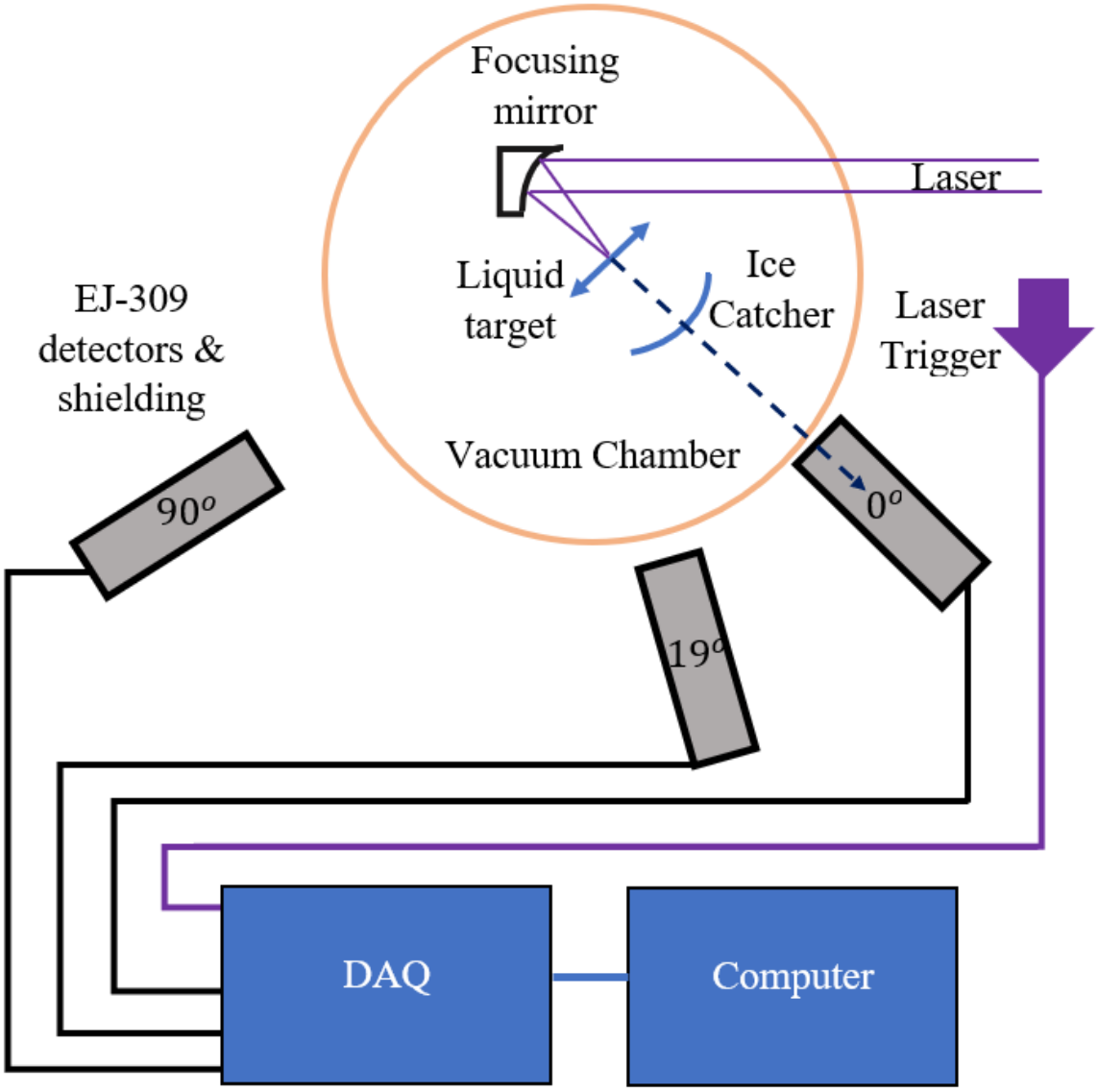

Figure 1: The experimental setup in which three EJ-309 detectors were used to determine if the mixed radiation source from the HIL system was isotropic. Processed and raw data from the detector labelled 0^o are pictured in Figures 2 and 3, respectively.

EJ-309 detectors have PSD capability to allow separation of x-ray or gamma events from neutron events. PSD is defined as [13]

$$PSD = \frac{Q_{long} - Q_{short}}{Q_{long}}, \tag{2}$$

where Q_{long} is the integral of the light intensity vs. time curve under the long time gate, and Q_{short} is the same integral under the short time gate. The duration of the short and long gates are chosen by the experimenter while processing raw data from the detector. In general, the short and long gates are adjusted until a separation can be seen between particle events in a PSD plot, as in Figure 2. What makes a separation "good enough" is a subject for a different paper [14]. These light intensity curves are provided by the detector when it detects a quantum of radiation. A given curve has a certain shape determined by the type of radiation that is incident on the detector, illustrated by Figure 3. For photons, the light curve is characterized by a sharp peak followed by a quick falloff. For neutrons, however, the light curve is characterized by a sharp peak followed by a slower falloff. The difference in falloff is due to a higher concentration of triplet states generated in the scintillation material by the neutron, which leads to more delayed radiation as the triplet states interact [15].

It was determined that the neutron output from the mixed radiation source was not anisotropic to any degree of certainty. More experimental work was needed to confirm or deny the hypothesis that the catcher was not introducing anisotropy. This result was not surprising given that the data came from just three spatially distinct regions.

Given enough computational power, analysis of a simulation could gather data from any number of points, and could observe instances of neutrons directly through diagnostic data. This powerful characteristic was a motivator for the use of simulation

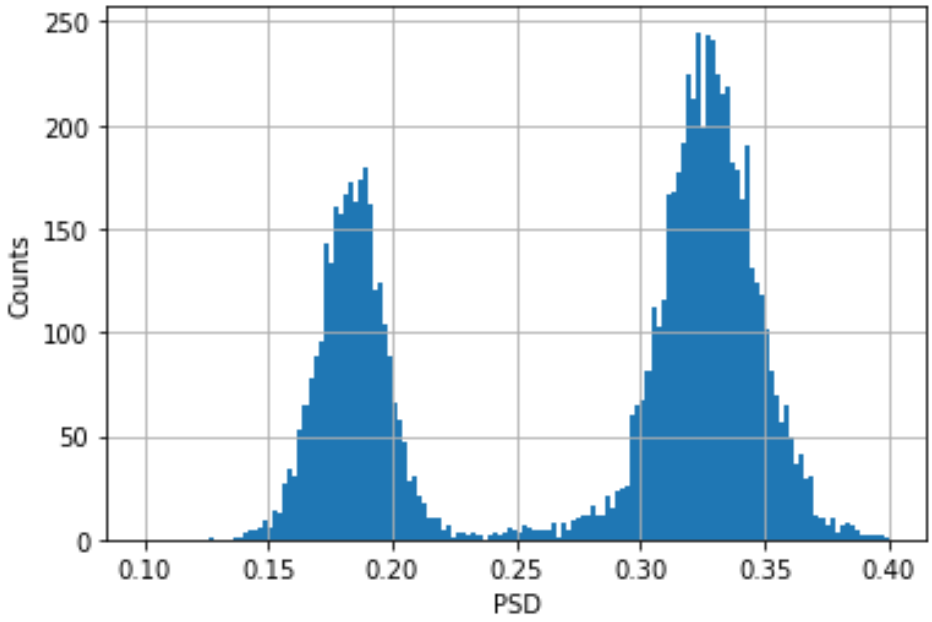

Figure 2: A PSD histogram created from data taken from the EJ-309 detector on-axis to the laser. Two distinct peaks can be seen; the lower PSD-valued peak is representative of x-rays while the higher PSD-valued peak is representative of neutrons.

to model LDF.

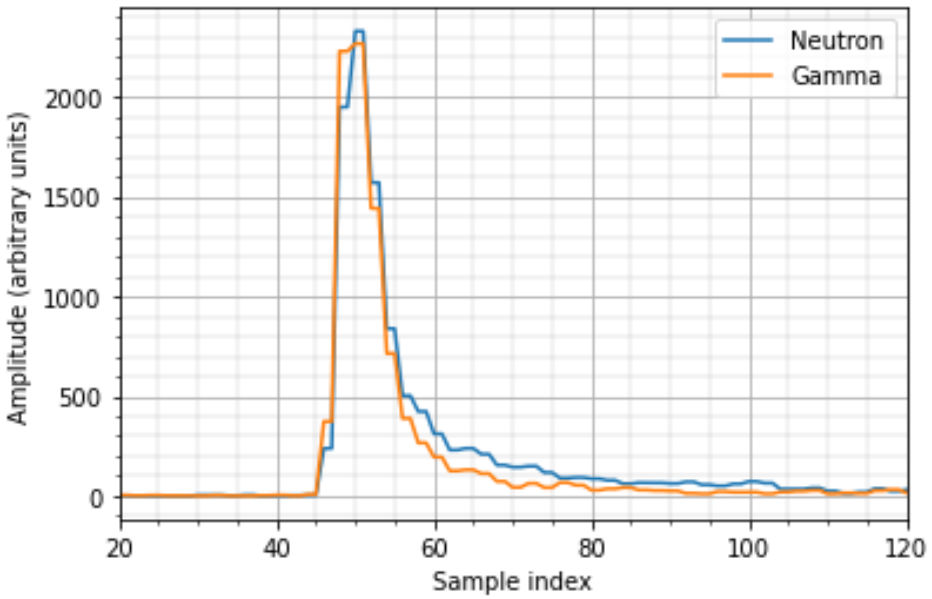

Figure 3: A plot of two light pulses from the EJ-309 detector on-axis to the laser. In this plot, the sample index, pictured on the x axis, is the time dimension determined by the sampling rate of the data acquisition system. The difference in tail height in this light pulse plot, as shown between the top and bottom pulses, allows for discrimination to occur between neutron and x-ray events.

III. Simulation Using WarpX

WarpX was chosen as the simulation code because it was free, open-access, and just released a nuclear fusion module. At the time research for this thesis began, the nuclear fusion module was still unreleased in the main branch of WarpX. This presented a unique opportunity to test out a new capability of WarpX and to simulate the underlying physics mechanisms taking place in the HIL system at once.

As previously stated, the LDF and plasma interactions were simulated using WarpX, a particle-in-cell (PIC) simulation software package. This approach to simulating large groups of charged particles involves tracking each simulated particle on a grid or "mesh". The usual PIC simulation involves a four-step process as follows [16]:

1. Current Deposition: interpolate particle charges onto grid
2. Field Solve: calculate charge and current densities; solve Maxwell's equations
3. Field Gather: interpolate electric and magnetic fields onto grid
4. Particle Push: calculate forces on particles from fields and push to next position

First, the particles' charges must be interpolated onto the grid. This interpolation enables calculation of charge and current density and is therefore the "current deposition" step. Acquisition of charge and current densities allows Maxwell's equations to be solved, hence this step's name of "field solve". This step gives the location and strengths of electric and magnetic fields, and interpolating field strengths at each location on the grid enables calculation of the forces the fields exert on the particles, concluding the "field gather" step. These forces are then used along with the particles' velocities to push the particles to their next location, giving the "particle push" step

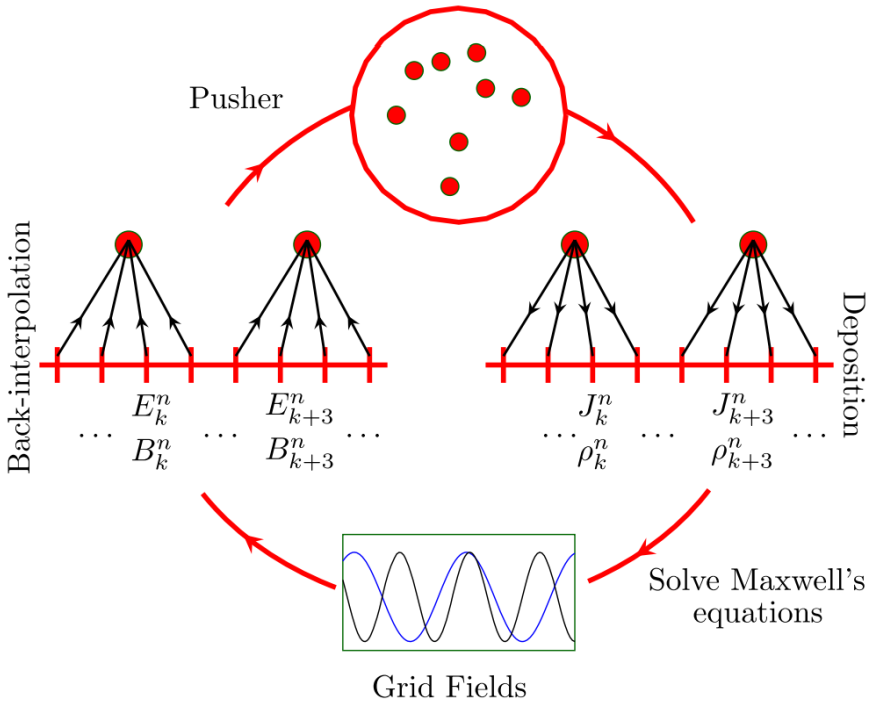

Figure 4: Figure from Shalaby 2017 [16]. In this figure, Deposition corresponds to the Current Deposition step, and the other steps pictured follow in order of the process.

and leading back to the current deposition step. This process is repeated until the simulation completes. Because each time step of a PIC simulation requires four steps, it takes many calculations to complete just one step. Escalating this process to the huge number of particles present in a usual PIC simulation make these simulations extremely computationally expensive.

A technique to relieve some of this computational stress using so-called "macroparticles" (also called super-particles) is implemented in WarpX, as in other PIC simulation packages. Macroparticles are essentially containers for groups of particles; forces that would be applied to many particles in a similar state are instead applied to a singular entity, making computation much cheaper. A particle "weight" is applied to each macroparticle, which represents the number of "real" particles that are being represented by a given macroparticle. This definition changes slightly in two dimensions - in that circumstance, macroparticle weight is the number of "real" particles

that are being represented by a given macroparticle *per unit length*. Because the non-planar coordinate in WarpX's 2D simulations is one meter long, macroparticle weight in two dimensions is equivalent to particles per meter.

3.1 Current Deposition

Charge and current densities are deposited on the grid according to

$$\begin{aligned} \rho &= \frac{1}{\Delta x \Delta y \Delta z} \sum_n q_n S_n, \\ \mathbf{J} &= \frac{1}{\Delta x \Delta y \Delta z} \sum_n q_n \mathbf{v}_n S_n, \end{aligned} \tag{3}$$

respectively. In Equation (3), q_n is the charge of a given particle, $\mathbf{v}_n$ is the velocity of that particle, Δx, Δy, and Δz are the simulation's spatial step sizes, and S_n is a spline of a given order [17]. S_n helps interpolate values onto the grid, while the other terms come from the definitions of ρ and $\mathbf{J}$, namely

$$\begin{aligned} \rho &= \frac{Q}{V}, \\ \mathbf{J} &= \rho \mathbf{v}. \end{aligned} \tag{4}$$

In Equation (4), Q is the total charge, represented discretely as a summation in Equation (3). Similarly, V is the volume, calculated by the cell dimensions $\Delta x \Delta y \Delta z$ shown in Equation (3). Each source of current tabulated by $\mathbf{J}$ may have an independent velocity, so $\mathbf{v}$ must be put in the summation term in Equation (3).

Another piece to this puzzle is preventing buildup of computational errors brought on by the violation of the current density continuity equation [18]:

$$\nabla \cdot \mathbf{J} = -\frac{\partial \rho}{\partial t}. \tag{5}$$

This is accomplished by either using a method that is very precise, or by adding

a corrective measure to the algorithm. WarpX's default setting chooses the former, using Esirkepov's method in combination with the Yee solver to get an exact answer for splines of arbitrary order [19].

3.2 Field Solve

As with all forays into electricity and magnetism, the basis of understanding for this step in the simulation process will be Maxwell's equations:

$$\begin{aligned} \frac{\partial \mathbf{B}}{\partial t} &= -\nabla \times \mathbf{E} \\ \frac{\partial \mathbf{E}}{\partial t} &= \nabla \times \mathbf{B} - \mathbf{J} \\ \nabla \cdot \mathbf{E} &= \rho \\ \nabla \cdot \mathbf{B} &= 0 \end{aligned} \tag{6}$$

where $\mathbf{E}$ and $\mathbf{B}$ are the electric and magnetic fields, $\mathbf{J}$ and ρ are the current and charge densities, and t is time. Equation (6) is given here in natural units, where $\epsilon_0 = \mu_0 = c = 1$.

By default, WarpX uses a certain second order finite-difference time-domain (FDTD) algorithm to complete the calculations for this step. In FDTD electro-magnetic (EM) algorithms, electric and magnetic fields are staggered in space and time such that a finite difference used to calculate either field with Maxwell's equations can be calculated for an exact point in space and time. For example, consider Figure 5, which shows staggering electric and magnetic fields in space with a Yee mesh and shows staggering in time with so-called "Leapfrog" integration [20].

In Figure 5, Faraday's Law from Equation (6) can be numerically calculated. The finite difference approximation of Faraday's Law at the point $\mathbf{B}_x$ is defined at would be

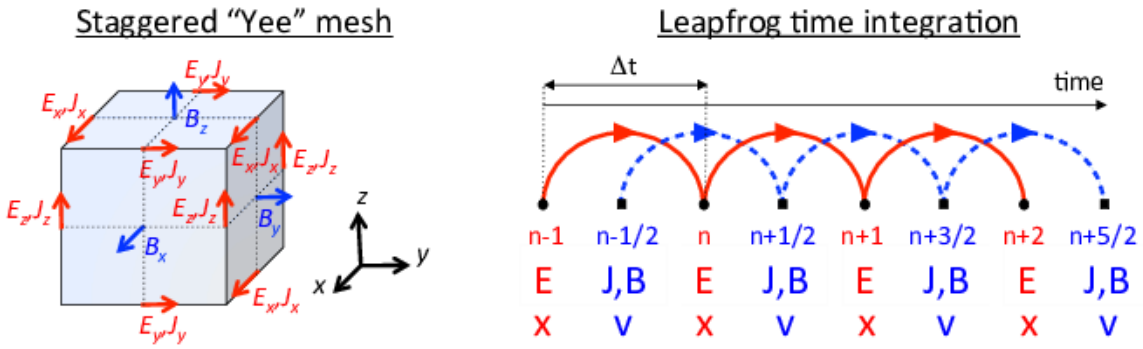

Figure 5: Left, a unit cell with EM fields drawn in colored arrows that represents a Yee mesh. Right, a timeline showing the staggered time integration process and what quantities get calculated at each step. [21]

$$\frac{\mathbf{B}_x|_{(n+1/2)} - \mathbf{B}_x|_{(n-1/2)}}{\Delta t} = \frac{\mathbf{E}_{y+1}|_n \; - \; \mathbf{E}_y|_n}{\Delta z} - \frac{\mathbf{E}_{z+1}|_n \; - \; \mathbf{E}_z|_n}{\Delta y}, \tag{7}$$

where $\mathbf{B}_x|_{(n+1/2)}$ is the magnitude of $\mathbf{B}$ in the x direction at the center of the positive x face of the Yee mesh cube pictured in Figure 5 at time $n + 1/2$. $\mathbf{B}_x|_{(n-1/2)}$ is the same for the time $n - 1/2$, $\mathbf{E}_{y+1}|_t$ is the magnitude of $\mathbf{E}$ in the y direction at the top of the Yee mesh cube at time n, $\mathbf{E}_y|_n$ is the same for the bottom of the cube, $\mathbf{E}_{z+1}|_n$ is the magnitude of $\mathbf{E}$ in the z direction at the positive y side of the Yee mesh cube at time n, $\mathbf{E}_z|_n$ is the same for the negative y side of the cube, and Δt, Δz, and Δy are the finite changes in time, z coordinate, and y coordinate, respectively.

Taking the difference of $\mathbf{B}$ between times $n + 1/2$ and $n - 1/2$ centers its measurement to time n, when the electric fields are being sampled. Similarly, taking the difference of $\mathbf{E}$ between y coordinates $y + 1$ and y and $z + 1$ and z centers its measurement to the center of the face where the magnetic field is being sampled. It is through this method that Maxwell's equations are solved at the same time and space, even though they are staggered.

Note that more equations than this one are necessary for solving any meaningful EM system or for use in a PIC sim. Practically, one would have to solve this equation

for $\mathbf{B}_x|_{(n+1/2)}$, then solve the other curl equation in Maxwell's equations, then solve that for the electric field at time $n+1$. At that point it would then be possible to update this equation again and repeat the process. Equation (7) is not shown here to explain the explicit mechanics of a PIC algorithm, but instead to explain that solving for a desired quantity can still be possible even though $\mathbf{E}$ and $\mathbf{B}$ are staggered through time and space. For the details of this process, see Yee 1966 [20].

For simulations in this work, perfectly matched layer (PML) boundary conditions were used for electricity and magnetism (E&M) field calculation. These boundary conditions work to absorb fields at the boundaries to approximate the laser leaving the target after an interaction. The basic PML mechanism is that it creates bounds outside the simulation window that match the impedance of an incoming wave and absorb it so it stops propagating and does not reflect or refract. This behavior mimics or approximates a wave propagating out of the simulation window.

3.3 Field Gather

There are three options for the field gather step that WarpX considers. The default option and the option used in the simulations in this work is the energy conserving variation. In this variation, fields are interpolated from the staggered Yee grid to the macroparticles using the spline interpretation setup shown by Table 1. If field quantities are known at nodes and not staggered positions, they are interpolated to the staggered grid before they are gathered for the particles.

The other options WarpX considers are the momentum conserving and the uniform options. In the momentum conserving variation, field quantities are interpolated from the grid nodes to the macroparticles; if field quantities are known at staggered positions instead of nodes, they are interpolated to the nodes first. In the uniform variation, field quantities are interpolated from the Yee grid to the macroparticles

Table 1: Field Gather Interpolation Splines

Field Component	Spline Interpolation
E_x	$(S_{nx-1}\ ,\ S_{ny}\ ,\ S_{nz}\ )$
E_y	$(S_{nx}\ ,\ S_{ny-1}\ ,\ S_{nz}\ )$
E_z	$(S_{nx}\ ,\ S_{ny}\ ,\ S_{nz-1})$
B_x	$(S_{nx}\ ,\ S_{ny-1}\ ,\ S_{nz-1})$
B_y	$(S_{nx-1}\ ,\ S_{ny}\ ,\ S_{nz-1})$
B_x	$(S_{nx-1}\ ,\ S_{ny-1}\ ,\ S_{nz}\ )$

directly; if field quantities are known at nodes instead of staggered positions, they are interpolated to staggered positions first.

3.4 Particle Push

To advance the particles forward in time, their velocity and acceleration are found from their previous acceleration and from the forces affecting them. WarpX uses the Newton-Lorentz equations of motion:

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{v} \\ \frac{d(\gamma\mathbf{v})}{dt} &= \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \end{aligned} \tag{8}$$

where $\mathbf{x}$ is the position of the particle, $\mathbf{v}$ is its velocity, q is its charge, m is its mass, and γ is the relativistic factor $\gamma = 1/\sqrt{1-v^2}$. Here, γ is expressed in natural units where c=1. In SI units, γ is expressed as $\gamma = 1/\sqrt{1-v^2/c^2}$.

This formulation is fairly straightforward: $\frac{d\mathbf{x}}{dt} = v$ is the definition of velocity, and the other part of Equation (8) can be derived from the Lorentz force and the relativistic equation for momentum [22]:

$$\begin{aligned} \mathbf{F} &= q\mathbf{E} + q\mathbf{v} \times \mathbf{B} \\ \mathbf{p} &= \gamma m\mathbf{v} \end{aligned} \tag{9}$$

where $\mathbf{p}$ represents momentum. Taking a derivative of the relativistic equation for

momentum and substituting $\mathbf{F}$ for it will make a quick arrival at Equation (8), bearing in mind that we assume the mass of a given particle does not change.

To implement Equations 8 in a discrete system like a simulation, WarpX uses a centered finite-difference discretized form of the Newton-Lorentz equations:

$$\begin{aligned} \frac{\mathbf{x}^{i+1} - \mathbf{x}^{i}}{\Delta t} &= \mathbf{v}^{i+1/2} \\ \frac{\gamma^{i+1/2}\mathbf{v}^{i+1/2} - \gamma^{i-1/2}\mathbf{v}^{i-1/2}}{\Delta t} &= \frac{q}{m}(\mathbf{E}^{i} + \mathbf{v}^{i} \times \mathbf{B}^{i}) \end{aligned} \tag{10}$$

where i in an exponent position marks the index of a given variable in time. The half-indices stated in Equation (10) mark times in the "leapfrog" solving process that are between consecutive solves for a certain variable. By default, the Boris relativistic velocity rotation is used by WarpX to solve Equation (10) and push particles to their next position [23]:

$$\bar{\mathbf{v}}^{i} = \frac{\gamma^{i+1/2}\mathbf{v}^{i+1/2} + \gamma^{i-1/2}\mathbf{v}^{i-1/2}}{2\bar{\gamma}^{i}} \tag{11}$$

where all variables used are defined as in Equations 9 and 10. This solution is then implemented by Boris's method:

$$\begin{aligned} \mathbf{u}^{-} &= \mathbf{u}^{i-1/2} + (q\Delta\mathbf{t}/2m)\mathbf{E}^{i} \\ \mathbf{u}' &= \mathbf{u}^{-} + \mathbf{u}^{-} \times \mathbf{t} \\ \mathbf{u}^{+} &= \mathbf{u}^{-} + \mathbf{u}' \times 2\mathbf{t}/(1 + \mathbf{t}^{2}) \\ \mathbf{u}^{i+1/2} &= \mathbf{u}^{+} + (q\Delta\mathbf{t}/2m)\mathbf{E}^{i} \end{aligned} \tag{12}$$

where $\mathbf{t}$ is the time, $\mathbf{u} = \gamma\mathbf{v}$ and other quantities are as expressed in Equation (10). This solution is accurate to second order and is time-reversible. A drawback to the Boris method is that it is not Lorentz invariant, which can create errors when calculating relativistic dynamics.

3.5 Simulation Stability Limits

The discretized nature of PIC simulations gives rise to several potential points of instability. Numerical heating of the plasma can occur if the parameters of the simulation are unstable. To be stable, the simulation must resolve the Debye length and the plasma frequency and satisfy the Courant-Friedrichs-Lewy (CFL) limit.

The spatial step of the simulation, Δx, must be less than or equal to the Debye length

$$\lambda_D = \sqrt{\frac{\epsilon_0 kT}{nq^2}}. \tag{13}$$

In Equation (13), ϵ_0 is the permittivity, k is Boltzmann's constant, T is the temperature, q is the charge of the chosen particles, and n is the particle density. If this constraint is not met, the plasma could heat up by a factor of $(\Delta x/\lambda_D)^2$ before reaching a steady state [24].

This so-called "numerical heating" may also occur if Δt, the temporal step of the simulation, is too large; Δt must be less than or equal to two divided by the plasma frequency

$$\omega_p = \sqrt{\frac{nq^2}{m\epsilon_0}}. \tag{14}$$

In Equation (14), n, q, and ϵ_0 are as defined in Equation (13) and m is the particle's mass.

In order to properly resolve the movement of particles and fields, the CFL limit must be met:

$$C \geq c\frac{\Delta t}{\Delta x}, \tag{15}$$

where C is the Courant number, c is the speed of light, and Δx and Δt are the spatial

and temporal step of the simulation, respectively. For this work, C will be taken to be one. In an arbitrary simulation, c would be replaced by the velocity of a subject or particle of interest whose velocity needs to be resolved correctly. It is taken to be the speed of light in this case because light will be simulated. If the CFL limit is not satisfied, the motion of particles or EM waves may not be resolved by the mesh. As an intuitive description, the time step and spatial step of a simulation must be set such that a particle or EM wave will not travel more than one cell in one time step.

Looking through WarpX's code base and landing on CartesianYeeAlgorithm.H, it was found that WarpX automatically computes a maximum time step given a cell size via the CFL limit by

$$\Delta t_{max} = \left(c\sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}} \right)^{-1}, \tag{16}$$

where Δx, Δy, and Δz are the spatial steps for each axis and c is the speed of light. In the case of two-dimensional simulations, Δy is equal to one meter because WarpX assigns the y-axis a distance of one meter in two-dimensional sims. In this work, Δx and Δz will be equal because square cell sizes were set. These restrictions simplify Equation (16) to

$$\Delta t_{max} = \left(c\sqrt{\frac{2}{\Delta x^2} + 1} \right)^{-1}, \tag{17}$$

where $\Delta x = \Delta z$. Without further input, however, WarpX will use its automatically calculated time step value. Luckily, changing the time step is as simple as using the input parameter warpx.cfl and setting it equal to the fraction of the target time step to the calculated time step.

To summarize, a stable simulation's parameters meet the following conditions:

$$\begin{aligned} \Delta x &\leq \lambda_D \\ \Delta t &\leq 2\omega_p^{-1} \\ \Delta t &\leq c^{-1}\Delta x \end{aligned} \tag{18}$$

3.6 Nuclear Fusion Implementation

Up until this section, everything in Chapter III has covered PIC simulations in general. What differentiates this use of WarpX from others is the use of its new nuclear fusion module. The module was written by Remi Lehe and Neïl Zaïm based off a paper by Higginson [25]. Using WarpX code, Higginson's paper, and another work by Takizuka and Abe that Higginson mentions in their paper, a rough outline of a fusion event can be understood as follows [25] [26]:

1. Colocation: two or more macroparticles arrive or exist in the same cell.
2. Particle Pairing: macroparticle pairs are chosen at random to collide.
3. Probability Calculation: fusion probability is calculated.
4. Particle Splitting: macroparticles are split into fusing and non-fusing groups.
5. Fusion: fusing particles are changed to the fusion product particles.
6. State Assignment: momentum, charge, etc. are assigned to fusion products.

Once macroparticles are colocated, or located in the same cell, those particles that undergo a collision are determined randomly. This particle pairing is necessary because the separation of particles within a cell is neglected; the closest two particles can get to each other in a PIC simulation is to be colocated within one cell. Once particles have been paired off for collision, the probability of fusion is calculated. The probability of fusion is given by Higginson Equation 8 [25]:

$$P_{fusion} = N_{ratio} \times F_{mult} \times W_{max} \sigma_{ab} v_{ab} \Delta t / V, \tag{19}$$

where $N_{ratio} = N - 1$ for identical particles, N is the number of macroparticles in a cell, F_{mult} is the fusion multiplier, W_{max} is the maximum weight of the fusing macroparticles, σ_{ab} is the cross section of the fusion reaction, v_{ab} is the relative velocity between the two fusing particles, Δt is the time over which the particles interact, and V is the volume of the space in which the collision will take place.

The fusion multiplier, F_{mult}, is a tunable variable in the simulation. It is introduced in order to increase the number of macroparticle products generated. Increasing the fusion multiplier will increase the probability of fusion events but decrease the weight of the resulting macroparticles. An appropriate F_{mult} can be found by [25]

$$F_{mult} = N_p W_r / Y_p, \tag{20}$$

where N_p is the number of fusion product macroparticles needed for diagnostic purposes, W_r is the weight of the reactant macroparticles, and Y_p is an estimate of the macroparticle yield of the reaction. For comparison, the yield of an example fusion reaction $a + b \rightarrow 3 + 4$ is

$$Y_{ab} = W_a W_b \sigma_{ab} v_{ab} \Delta t / V, \tag{21}$$

where W_a and W_b are the weights of the macroparticle reactants. The "weight" of a macroparticle is the number of "real" particles contained in that macroparticle.[1] The cross section of the fusion reaction, σ_{ab}, is a measure of how likely that reaction is to occur.

After P_{fusion} is calculated, a random uniformly distributed number between zero

[1]For 2D simulations, like those studied in this work, "weight" is the number of particles **per meter** in a macroparticle.

and one is generated. If the generated number is less than P_{fusion}, then fusion occurs. If a fusion event is triggered, the fusing macroparticles are split into pairs of non-fusing macroparticles and fusing macroparticles. The pairs are chosen such that the fusing pair of macroparticles have the same weight. This is to ensure numerical conservation of charge, total energy, and momentum in the case of unevenly weighted macroparticles. Figure 1 of Higginson, shown in Figure 6, shows the process visually.

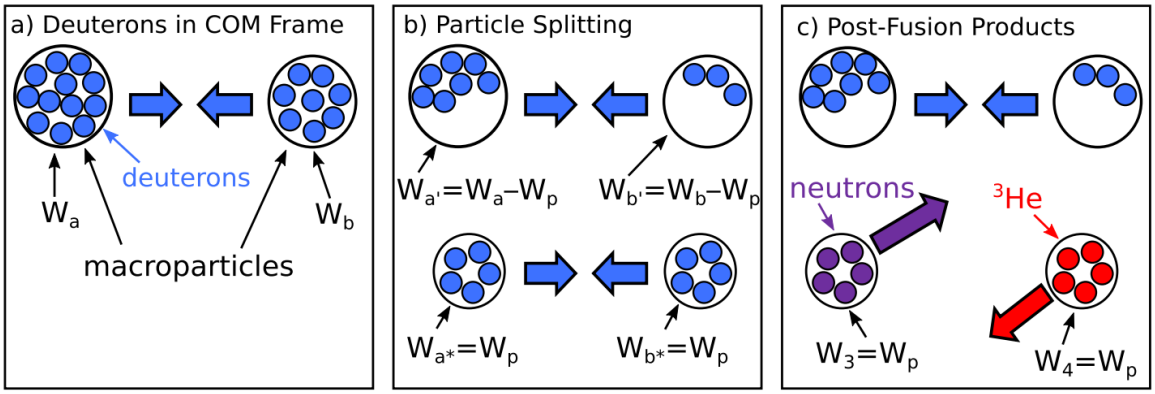

Figure 6: An example of the fusion process for DD fusion. Figure reproduced from Higginson [25]. a) Deuteron macroparticles start in the center of mass frame. b) The particle splitting process occurs by splitting the macroparticles into fusing and non-fusing pairs. c) The fusing pair of macroparticles is converted into fusion products while the non-fusing pair is left untouched.

Following this particle splitting and fusion processes, the outgoing fusion products must be given appropriate energies. Because fusion processes transform rest mass energy into kinetic energy, kinematics must be combined with nuclear physics to determine the energies of the product particles. Momentum and charge of the fusion products must also be calculated and assigned to the resulting macroparticles. In order to retain some sense of brevity and scope, these calculations will be left to other works [12] [25].

IV. Methodology of Modeling Laser-Plasma Interactions

Much of the difficulty of setting up simulations comes with choosing the correct parameters. This chapter is largely devoted to setting the stage of the simulation by providing insight into how the simulation's parameters are set. The simulation is not the only part of this work decided by input parameters, however; data analysis is also dictated by what information the program is told to output. Rounding out this chapter, the methods by which the questions of neutron time dependence, generation, and propagation are answered following the discussion of parameters.

4.1 Simulation Parameters

4.1.1 Dimensions and Timing

Two-dimensional runs of WarpX were preferred over three-dimensional runs; although executing runs in 3D may provide a better analog to the real world, porting 2D simulations into 3D proved too time-consuming to be practical. Issues with memory from the addition of another dimension proved to be too time-consuming to troubleshoot in the long run. Future work on the subject of this thesis should include simulations in 3D if time allows, as it may provide more fruitful insights.

The simulation was set to run until it reached a time of 2000 femtoseconds. This time frame gives the 40 fs duration of the laser plenty of time to propagate to the target and back out of the frame, which takes about 175 fs. It also allows higher-energy particles to travel out of the simulation window if their energy and trajectory allows.

4.1.2 Laser

The laser used in simulations was assumed to be Gaussian in space and have a sine squared relationship in time. The exact form is as follows:

$$E(X,Y,t) = -E_0\sqrt{\frac{w_0}{w(z)}}\sin(\omega_0 t + \frac{kX^2}{2R(z)} - \frac{\psi}{2})\exp(-\frac{X^2}{w(z)^2})\sin(\frac{\pi t}{2d})(t < 2d) \quad (22)$$

where E_0 is the maximum amplitude of the electric field in Volts per meter, w_0 is the minimum beam waist radius in meters, $w(z)$ is the beam waist radius at the target in meters, ω_0 is the temporal frequency of the laser in radians per second, t is the current time of the simulation in seconds, k is the spatial frequency or wave number in radians per meter, X and Y are coordinate positions in the simulation in meters, $R(z)$ is the radius of curvature in meters, ψ is the Gouy phase in radians, and d is the pulse duration in seconds.

In an effort to computationally mirror the efforts of the Extreme Light Group (ELG)'s experimental system, the simulation laser's parameters were picked such that the experimental laser was mimicked as much as possible. Table 2 lists the experimental parameters of the laser measured in the lab and Table 3 lists the parameters used in the simulation representative of those quantities.

Table 2: Experimental Laser Parameters

Parameter	Value	Notes
Energy	7.7 mJ	Describes value on target
Intensity	10^{19} Wcm^{-2}	Describes value on target
Pulse duration	40 fs	Describes FWHM value
Repetition rate	1 kHz	
Spot size	1.8 μm	Describes FWHM value
Wavelength	780 nm	

Table 3: Computational Laser Parameters: these parameters were used in Equation (22) to emulate the Red Dragon laser at the ELL.

Parameter	Value	Symbol
Maximum electric field	6.3×10^{12} Vm^{-1}	E_0
Waist radius	1.5288×10^{-6} m	w_0
1/e waist radius	2.8747×10^{-6} m	$w(z)$
Frequency	2.4166×10^{15} s^{-1}	ω_0
Wave number	8.0554×10^{6} m^{-1}	k
Radius of curvature	2.0901×10^{-5} m	$R(z)$
Gouy Phase	1.010 $radians$	ψ
Pulse Duration	40×10^{-15} $seconds$	d

4.1.2.1 Parameter Calculation

This section contains an itemized list of how each parameter in Table 3 was calculated.

The waist radius, w_0, was found by [27]

$$w_0 = \frac{S_{FWHM}}{\sqrt{2\ln 2}}, \tag{23}$$

where S_{FWHM} is the spot size at full width half max. The 1/e waist radius, $w(z)$, was calculated by [28]

$$w(z) = w_0\sqrt{1 + \left(\frac{z}{z_R}\right)^2}, \tag{24}$$

where z is the position along the beam and z_R is the Rayleigh range [28]:

$$z_R = \frac{\pi n w_0^2}{\lambda}. \tag{25}$$

The frequency ω_0 was found by $\omega_0 = kc$ and the wave number k was found by $k = 2\pi/\lambda$. The pulse duration d was taken directly from the experimental parameters of the laser. The radius of curvature $R(z)$ was calculated by

$$R(z) = z\left(1 + \left(\frac{z_R}{z}\right)^2\right), \tag{26}$$

and the Gouy phase, ψ, was calculated by [28]

$$\psi = \arctan\left(\frac{z}{z_R}\right). \tag{27}$$

The maximum electric field, E_0, was found by error minimization using a Python script found in Appendix A. The script uses the following process: first, guess a value for E_0 and calculate the waist radius from the FWHM spot size value. Calculate the intensity using the guessed E_0 value with

$$I_0 = \frac{c}{2}\epsilon_0 E_0^2, \tag{28}$$

where ϵ_0 is the permittivity of free space and c is the speed of light. Finally, calculate the energy of the beam in 2D and 3D with

$$\begin{aligned} E_{2D} &= \frac{c\epsilon_0}{2}\sqrt{\frac{\pi}{2}} w_0 E_0^2 d_{FWHM} \\ E_{3D} &= \frac{\pi}{2} w_0^2 I_0 d_{FWHM} \end{aligned} \tag{29}$$

and compare E_{3D} to the energy target value. See Table 2 for the target energy value of the laser operated by the ELG. Repeat this process until E_{3D} reaches desired

precision. In Equation (29), d_{FWHM} is the full width half max duration of the pulse. E_{2D} is not used in the E_0 finding process, but can be used as a useful diagnostic for 2D simulations.[1]

4.1.3 Plasma

The collection of particles used in a WarpX simulation will be referred to as the "plasma" in this work. For the purposes of the simulations relevant to this work, each instance of a particle type was assigned a species type, an injection style, a number of macroparticles per simulation cell, a density and density profile, an initial momentum distribution, and a location. An example particle initialization block for a WarpX input deck may look something like the following.

```
deuterium.species_type = deuterium
deuterium.injection_style = "NRandomPerCell"
deuterium.num_particles_per_cell = 100
deuterium.profile = constant
deuterium.density = 6.66e28
deuterium.momentum_distribution_type = "at_rest"
deuterium.xmin =  -0.25e-6
deuterium.xmax =   0.25e-6
deuterium.zmin = -10.e-6
deuterium.zmax =  10.e-6
```

In this example, the WarpX parameter particles.species_names has already been set to include the particle name "deuterium" in its list.

The species chosen for simulations in this work are electrons, deuterium, and oxygen, the constituents of heavy water. Each of these species were initiated twice

[1] See Section 5.1 for the use of E_{2D}.

so that two separate plasma areas were formed: one for the heavy water target and one for the heavy water ice catcher. Initialization needed to be done twice for the heavy water particles so that densities and locations could be independent, as WarpX does not allow two locations or two densities for the same instance of a particle. All instances of particles were given the injection style "NRandomPerCell". Instances of deuterium were given 100 macroparticles per cell so that collision and therefore fusion could be efficiently modeled, while instances of the other constituents were given 25 macroparticles per cell to drive down simulation time.

In order to establish some sort of basis of comparison to experiment, the plasma in the simulations of this work were given the properties of heavy water and heavy water ice, the target of the laser featured in the isotropy experiment and the catcher of said experiment, respectively. A density of $1,106\ kgm^{-3}$ was used for heavy water, while a density of $1,017\ kgm^{-3}$ was used for heavy water ice [29]. This resulted in a molecular density of $3.33 \times 10^{28} m^{-3}$ for heavy water and $3.06 \times 10^{28} m^{-3}$ for heavy water ice. Because there are two deuterium atoms per heavy water molecule, deuterium was assigned a particle density of $6.66 \times 10^{28} m^{-3}$ for heavy water and $6.12 \times 10^{28} m^{-3}$ for heavy water ice.[2] There is one oxygen atom and ten electrons per heavy water molecule; those particle groups were assigned their corresponding multiple of the molecular density.

The type of momentum distribution chosen for the simulations in this work was "at-rest". Starting the plasma at rest was chosen because it seemed to produce neutrons, and because giving one particle species a nonzero initial momentum and

[2]It was found after this work was written that user error by the author resulted in the wrong densities being copy/pasted into the input file: the particle densities for the particle species constituting heavy water ice in the catcher were identical to those for heavy water. However, because the densities of heavy water and heavy water ice are of similar magnitude, it is unlikely that the outcome of the simulation would change. This seems especially true given that the Debye length is unresolved by several orders of magnitude, numerical heating is present, and that deuteron macroparticles of the catcher do not seem to interact with deuteron macroparticles from the target. See Appendix B for the input file used for the simulation featured in this work.

starting the other species at rest resulted in the production of neutrons before the laser impacted the target.[3]

The target plasma that the laser impacts was set to be half a micron thick and 20 microns in length. The thickness was chosen to match that of the experimental conditions, while the length was chosen to be longer than the spot size of the laser to explore the dynamics of the system. The heavy water ice catcher used in the physical isotropy experiment was $4mm$ thick. Unfortunately, this thickness alone would expand the simulation window by two orders of magnitude, not including the typical physical distance between target and catcher. Therefore, it was decided that the catcher would be reduced in size such that it would fit in the established simulation window of 30 microns by 30 microns. The catcher was chosen to be five microns thick and 20 microns in length. This gives the catcher a greater volume than the target while leaving some space unfilled by the catcher to observe the dynamics of the system.

The laser, target, and catcher can be seen in Figure 7. The rainbow spectrum of color represents the electric field from the 780 nm laser, while the two purple rectangles represent the target and catcher. In Figure 7, only the density of deuterium is represented; other particles like electrons and oxygen are also present but are not shown for conciseness. The target is the smaller rectangle in the middle of the visualization window, and the catcher is the larger, off-center rectangle. Figure 7 shows the state of the simulation at 40 fs; at this time, no neutrons have been generated. If neutrons were present they would be represented by a blue/green color scale.

[3]It is suspected that the neutrons produced from starting the plasma at rest was anomalously generating these neutrons by numerical heating. See Section 5.1 for further discussion on this topic.

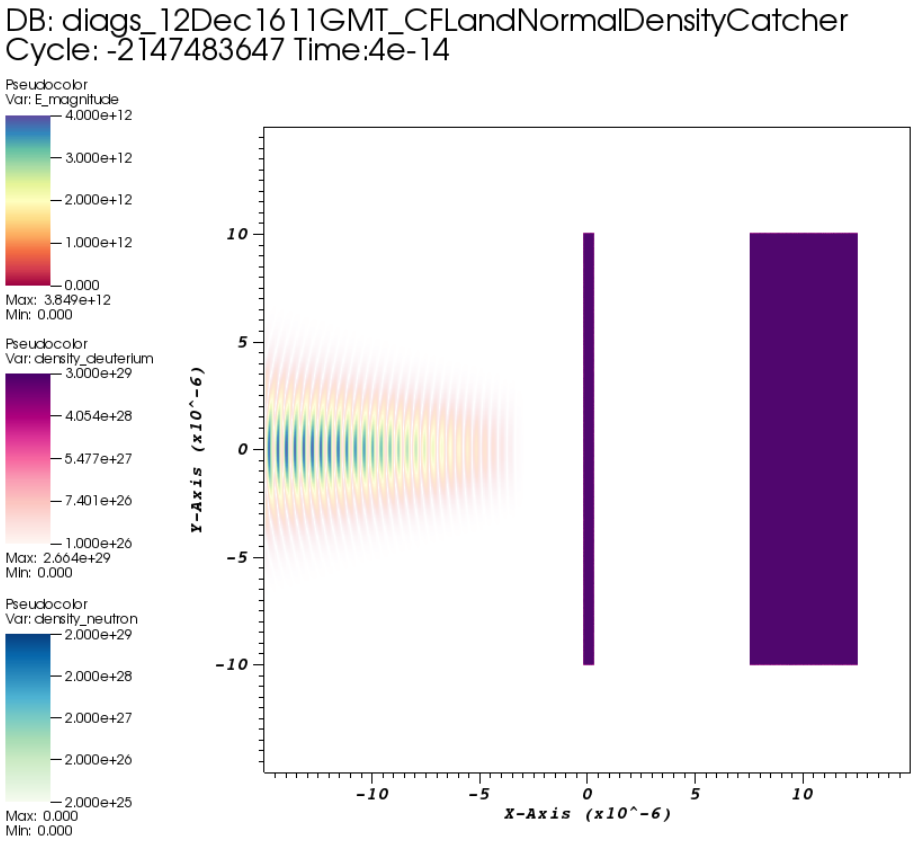

Figure 7: The setup of the simulation as shown by a frame at a time of 40 fs. The magnitude of the electric field is represented by a spectrum of colors. These electric field fluctuations represent the 780 nm laser. The density of deuterium is represented by a purple to white spectrum. The purple rectangles representing the target (the thinner rectangle) and the catcher (the thicker rectangle) are both composed of deuterium, and as such are a deep purple at their initial density.

4.2 Data Analysis

This work uses two main methods of analyzing data from WarpX runs: full diagnostics files and reduced diagnostics files. Full diagnostics files paint an all-inclusive picture of the state of the simulation by dumping field and particle information, while reduced diagnotics files hone in on a specific quantity specified by the user. Full diagnostics files were used to visualize aspects of the simulation with VisIt, a mass-data visualization software. Because full diagnostics files are saved in binary, some sort of data extraction or visualization software or code language module must be used to unpack the data.[4] Although full diagnostics files are saved in binary, their file sizes can get massive when very frequent samples of data are needed. For this work, a full diagnostics file was saved every femtosecond, or every 25 time steps.

Reduced files on the other hand were used to create simple plots to track single quantities, like particle number or particle energy. These files output as readable text files, and an output from one time step of a simulation gets output as one line in the file. This means that while reduced diagnostics files are limited in scope, they tend to be much more portable because of their comparatively small file size. For this work, reduced diagnostics files were saved every time step. This allows analysis of reduced quantities to take place on personal machines and user-based analysis algorithms. To this end, MATLAB was used to plot reduced quantities for analysis. Its ability to automatically create import functions at a click of a button was unerringly useful for this work.

An example of how full and reduced diagnostics are set up in an input deck is shown below.

[4] Along with VisIt, the Python module yt could also be used for data analysis and visualization. This module was experimented with by the author, but no figures from that program have been used in this work.

```
#################################
########## DIAGNOSTICS ##########
#################################

########## FULL DIAGS ###########
my_constants.intVar = 25
my_constants.RedintVar = 1
diagnostics.diags_names = diag1
diag1.intervals = intVar
diag1.diag_type = Full
diag1.fields_to_plot = Ex Ey Ez part_per_cell

diag1.particle_fields_to_plot = density
diag1.particle_fields.density(x,y,z,ux,uy,uz) = 1./cell_volume
diag1.particle_fields.density.do_average = 0

######### REDUCED DIAGS ##########
warpx.reduced_diags_names = particle_num particle_ene
particle_num.intervals = RedintVar
particle_num.type = ParticleNumber
particle_num.separator = ","

particle_ene.intervals = RedintVar
particle_ene.type = ParticleEnergy
particle_ene.separator = ","
```

Thanks to reduced diagnostics files, analysis of the number of neutrons generated as a function of time was straightforward, at least on a surface level. By asking WarpX to record the amount of particles currently present in the simulation at all time steps, the number of neutrons present at any given time can be determined outright.

The questions of where neutrons generate and where neutrons propagate, however, were not able to be answered this way. Instead, the simulation window was visualized with VisIt, including the magnitude of the electric field, the density of deuterium, and the density of neutrons. An example of the simulation window being plotted can be found in Figure 7. The location that neutrons were generated from and where they propagate to were determined from visual inspection of these VisIt plots.

V. Results and Analysis

Results from the most current WarpX simulation will be discussed here. The input file used for this simulation can be found in Appendix B. The three main questions of neutron time dependence, generation, and propagation and what implications their answers have for the system at large will be analyzed. The effect of numerical heating and the effect it has on the data and on potential conclusions will be discussed.

There is still much work to be done, but the novel nature of this endeavor should not be ignored. WarpX is free, computationally efficient, and the addition of its new nuclear fusion module allows the Extreme Light Group (ELG) to examine laser-driven fusion (LDF) in a novel way. There are certainly issues that need to be resolved to acquire meaningful information from the simulations, but because of how new the nuclear fusion module is, it was still a useful practice to test it out and find if it had any glaring bugs or inaccuracies. Fortunately, this did not appear to be the case.

5.1 Stability Limits

In Section 3.5, three stability conditions were discussed. These are summarized in Equation (18), and include one limit relating the temporal step to the plasma frequency, one limit that relates the spatial and temporal steps to each other, and one limit relating the spatial step to the Debye length.

The temporal step was most limited by electrons, whose parameters in the simulation resulted in a plasma frequency of $3.26 \times 10^{16}\ Hz$. This set a boundary of $6.14 \times 10^{-17}\ s$ for the time step. The time step was set at $4 \times 10^{-17}\ s$, resolving that stability condition.

The chosen spatial step (or cell size) was $2.93 \times 10^{-8}\ m$ for both axes. Using this value and Equation (17), a Δt_{max} of $6.91 \times 10^{-17} s$ was found. The chosen temporal

step is under this value, satisfying the Courant-Friedrichs-Lewy (CFL) limit. To get $4 \times 10^{-17}s$ from $6.91 \times 10^{-17}s$, a warpx.cfl value of about 0.579 was used.

Unfortunately, the realistic densities that were the target of the simulation made the Debye length and therefore the step size prohibitively small to simulate on a reasonable time scale. Oxygen is the most restrictive species present in the simulation, with a resulting Debye length of $8.1 \times 10^{-13}m$. The chosen cell size of $2.93 \times 10^{-8}m$ clearly cannot resolve this value by several orders of magnitude.

It is therefore theorized that the unresolved Debye length is causing anomalous heating of the plasma when hit by the laser. It was mentioned in Section 4.1.3 that giving one particle species a nonzero initial momentum and starting the other species at rest resulted in the production of neutrons before the laser impacted the target. This was tested later and helped determine the presence of numerical heating. It is theorized that when only one species was given momentum and the remaining species started at rest, numerical heating started occurring sooner in the simulation due to the presence of moving particles, thus causing formation of neutrons. In a later run, all species were given a Maxwell-Boltzmann distribution at 100 eV and no neutrons ever formed in the simulation, even after being struck by the laser. Unfortunately, this fascinating behavior could not be pursued because of time constraints.

If it is true that the unresolved Debye length is causing numerical heating, the plasma could heat up by a factor of

$$(\Delta x/\lambda_D)^2 = \left(\frac{2.93 \times 10^{-8}m}{8.1 \times 10^{-13}m}\right)^2 = 1.3 \times 10^9$$

before reaching a steady state [24]. In a simulation without fusion, there would be a straightforward way to test for this heating: by comparing the energy of particles to the energy of the input laser, one could determine if the energy of particles in the simulation ever exceeds that of the laser. Any initial energy must be accounted for,

but regardless this would still be a decent litmus test. However, due to the cultivation of energy from the rest mass of deuterium particles, comparitively smaller changes in energy from numerical heating become extremely difficult to spot with precision. Take the following example: Figure 8 shows the total energy of all particles in the simulation as a function of time.

With the laser parameters detailed in Table 3 and using Equation (29), the calculated beam energy in two dimensions should be about 4000 Joules, while the total particle energy in Figure 8 seems to arrive at its maximum at about 1,750,000 Joules. This discrepancy of several orders of magnitude would normally prove that some sort of numerical instability or heating is present in the simulation.

However, an estimate of the energy yield from fusion agrees with this total energy. As seen in Equation (1), each reaction that creates a neutron imparts 3.3 MeV to the product particles. The maximum number of neutrons present in the sim is about 3.34×10^{18}. Multiplying these quantities and converting to Joules, this results in a total fusion energy output of $1.8 \times 10^{6}\ J$. The maximum particle energy present in the simulation, visualized in Figure 8, can be seen around this value and was found to match it to the same precision. Therefore, it is difficult to distinguish any evidence of numerical heating from energy transfer to particles from fusion, so this method of energy analysis cannot be used in this case.

Although numerical heating is suspected to take place in this simulation, gathering enough evidence to accurately diagnose it is difficult. This difficulty, combined with strict time constraints, requires the author to leave finding a method to accurately diagnose numerical heating in fusion simulations to a future endeavor. Data from this run will still be used in this work to talk about the potential behavior of the system, but it will be used with the knowledge that the sim is imperfect and that some conclusions can change as more accurate simulations take place.

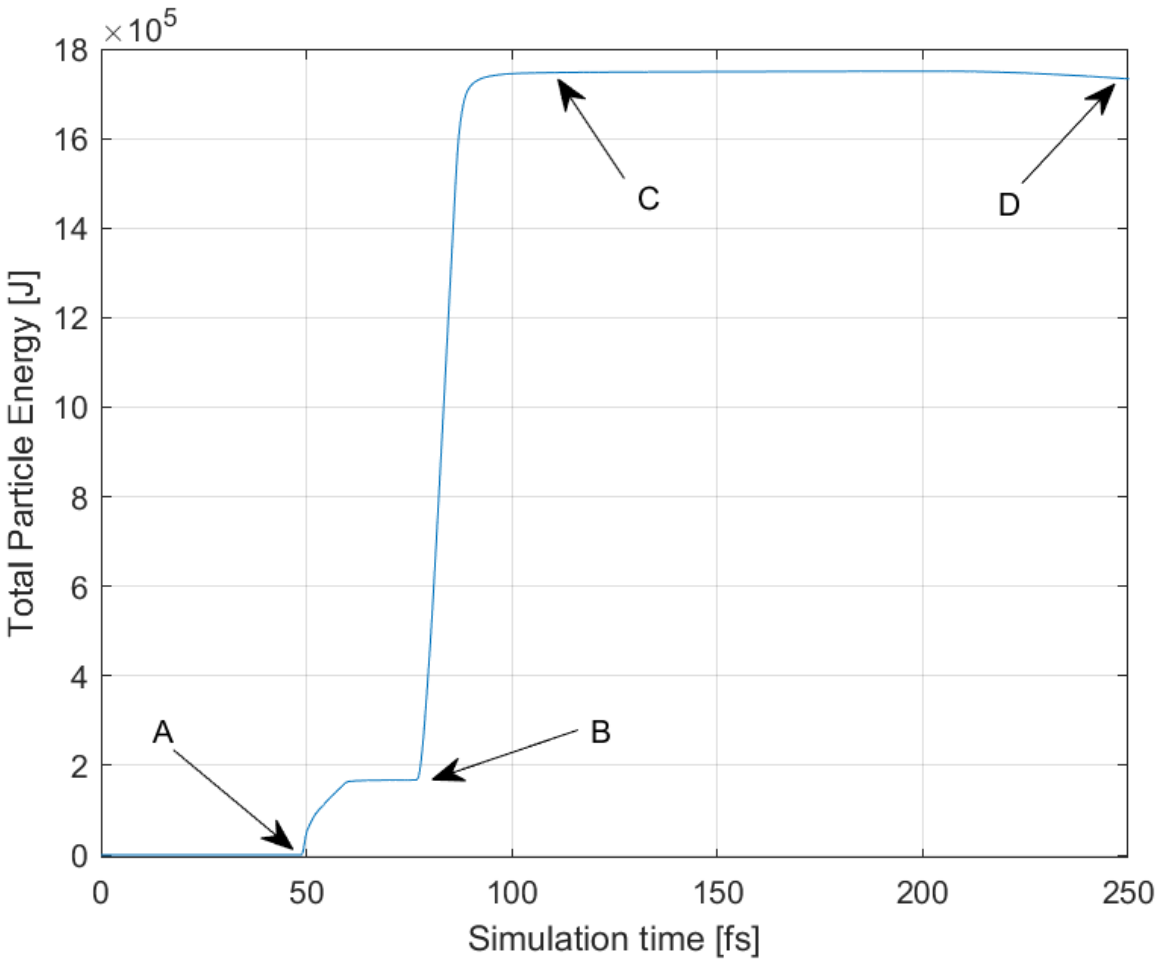

Figure 8: The total energy of all particles in the simulation as a function of time. Points A and B mark times where rest mass energy from deuterons is released as kinetic energy to fusion products, while points C and D mark times when particles are travelling in a steady state and when particles begin leaving the simulation window, respectively.

5.2 Simulation Road Map

Although it may not work for proving numerical heating, Figure 8 acts as a good road map of the simulation as a whole. The state of the simulation at various times will be shown and discussed using this plot as a road map.

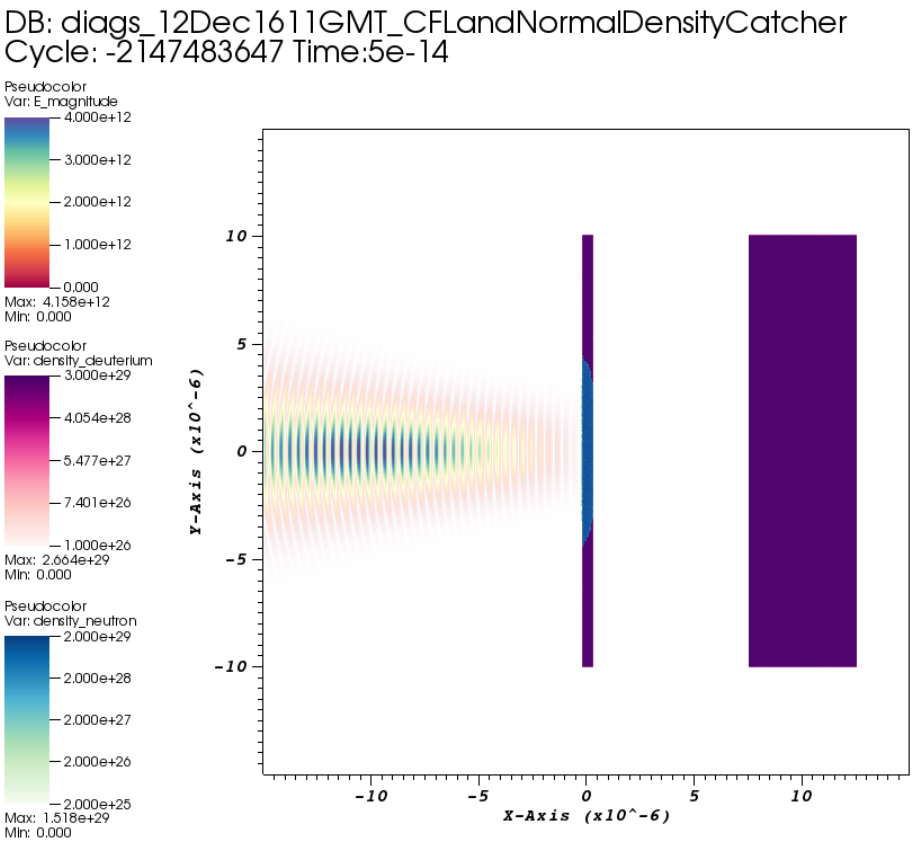

Figure 9: Simulation Frame 50 fs.

Point A occurs around 50 fs, shown by Figure 9. At this time, the laser has just started significantly interacting with the particles in the target, and a significant portion of deuterons in the target have undergone fusion, creating a tight group of neutrons.

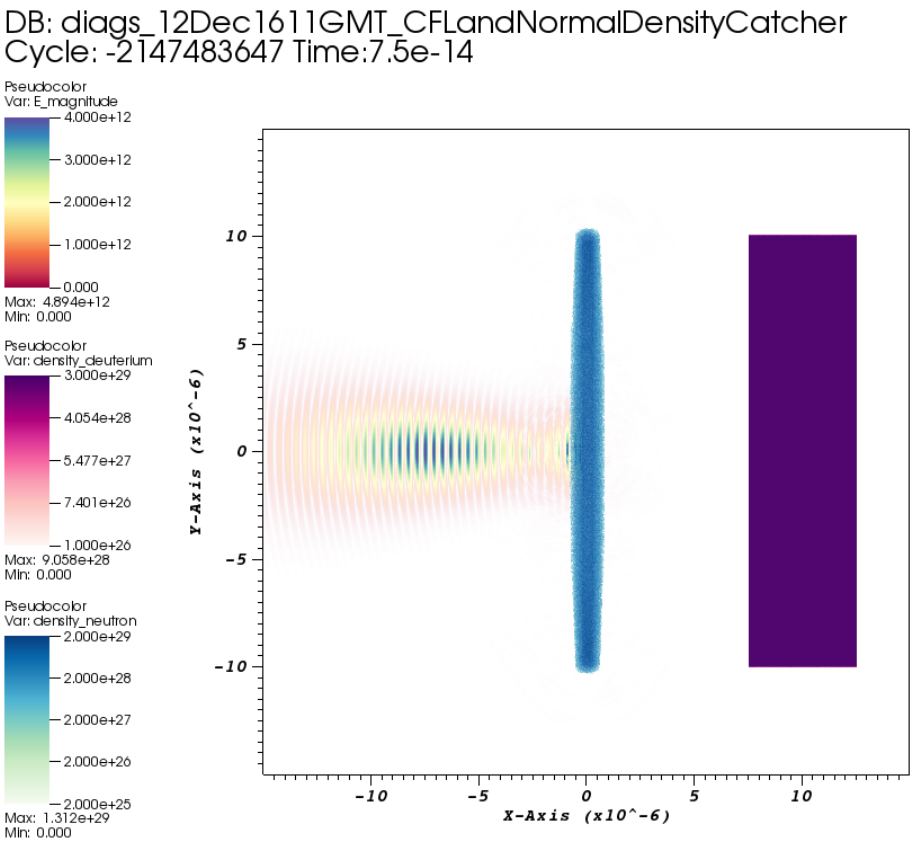

Figure 10: Simulation Frame 75 fs.

Point B occurs around 75 fs, shown by Figure 10. Neutrons have at this point fully enveloped the target and are beginning to travel away in an isotropic manner.

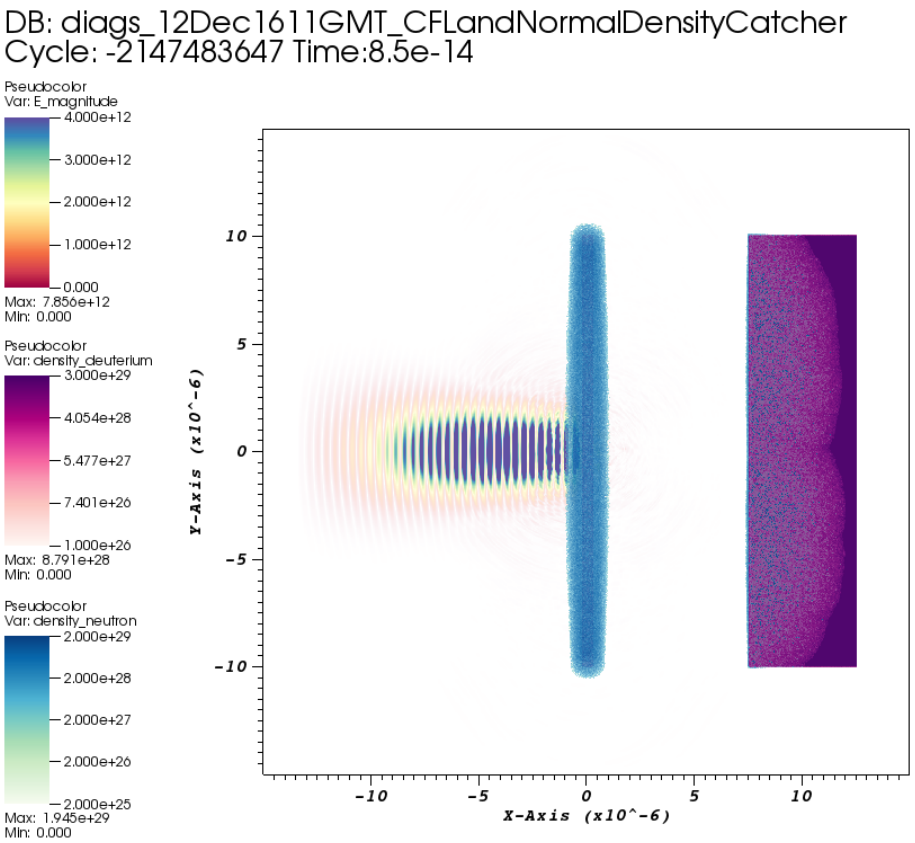

Figure 11: Simulation Frame 85 fs.

Between points B and C, energy is transmitted to the catcher and it begins fusion of its deuterons in earnest. Figure 11 shows the catcher in progress of fusing its fusible material. It is thought that the ignition of the catcher is also caused or aided by numerical heating from the energy from the laser pulse that is transmitted through the target, though other causes of this mysterious auto-ignition are possible.

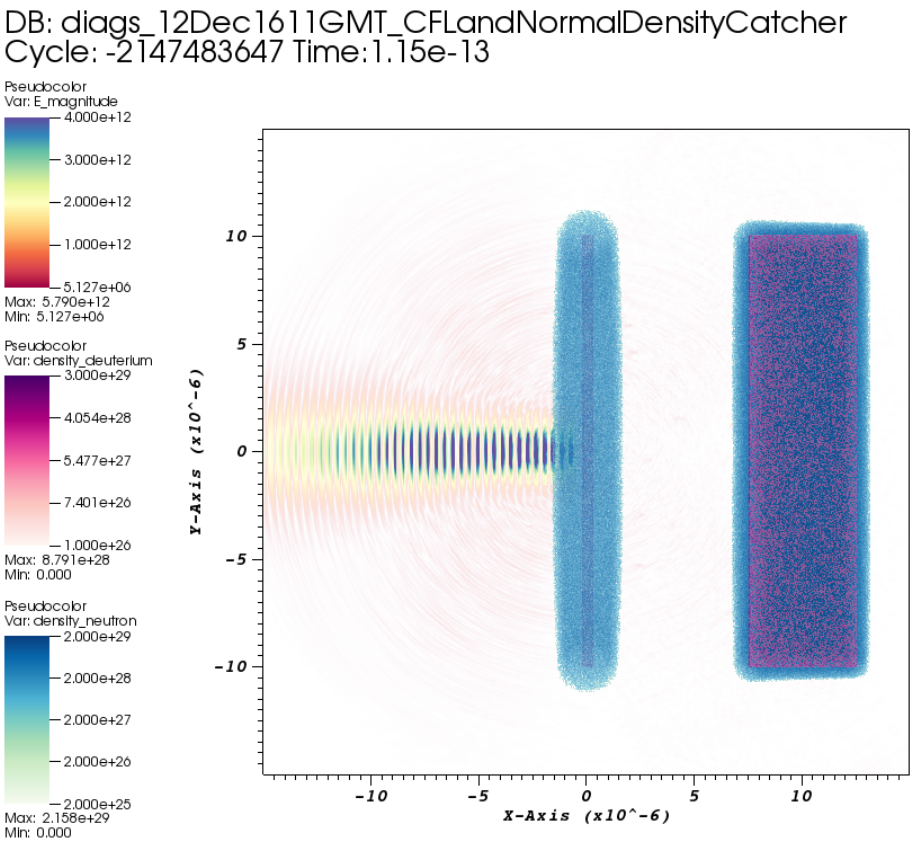

Figure 12: Simulation Frame 115 fs.

Point C occurs around 115 fs, shown by Figure 12. Here, the catcher has been encased in neutrons from fusion. A weaker-magnitude electric field can be seen emanating from the center of the target, likely from transmission of a small amount of energy from the laser pulse.

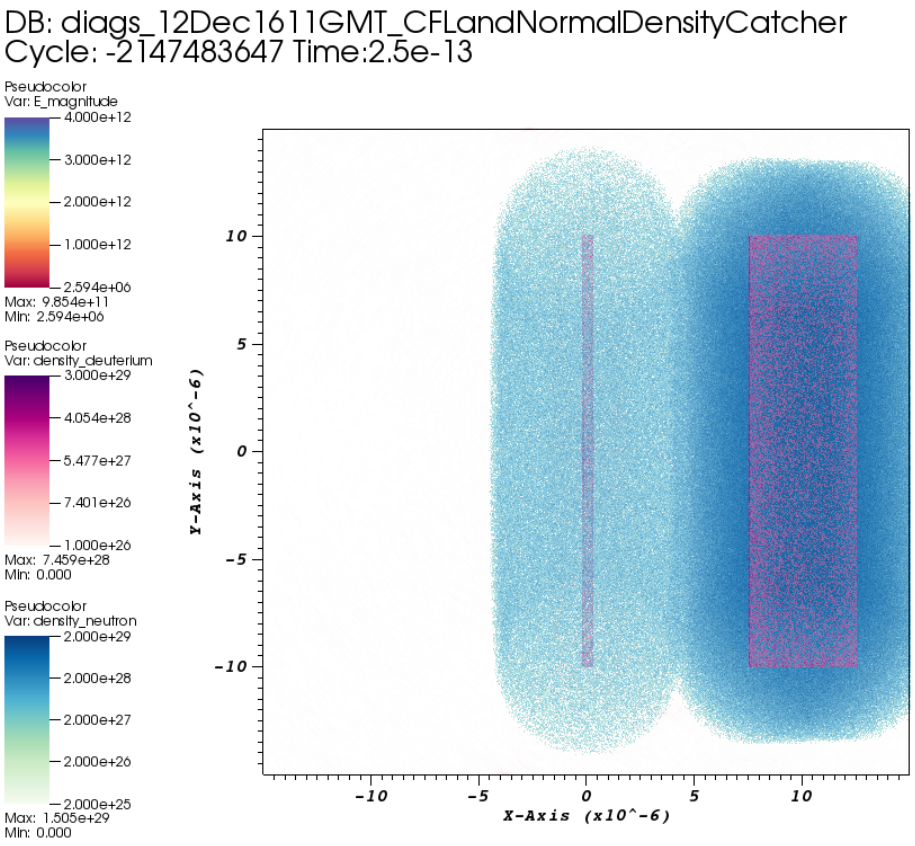

Figure 13: Simulation Frame 250 fs.

Point D occurs around 250 fs, shown by Figure 13. The energy curve slopes downwards as particles begin leaving the simulation window, as is shown by the high density of neutrons intersecting the border on the positive x-axis.

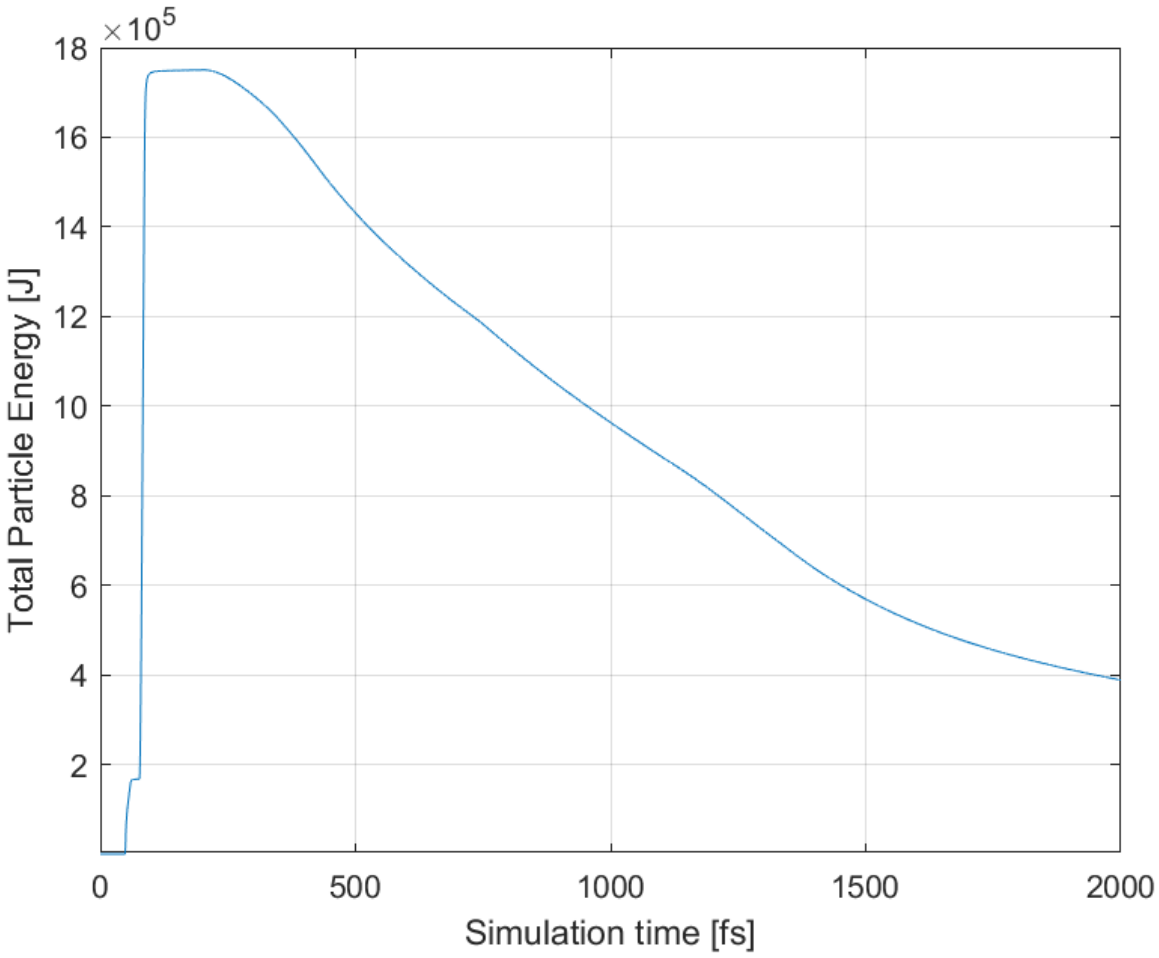

Figure 14: The total energy of all particles present in the simulation as a function of time, extended out to the full time of the simulation, 2000 fs. The first 250 fs can be seen in more detail in Figure 8.

After point D, neutrons continue to leave the simulation window, further decreasing the particle energy. Figure 14 shows this behavior of particle energy, which extends beyond what is seen in Figure 8. The simulation runs until 2000 fs have passed; Figure 15 shows this time at the last time step of the simulation.

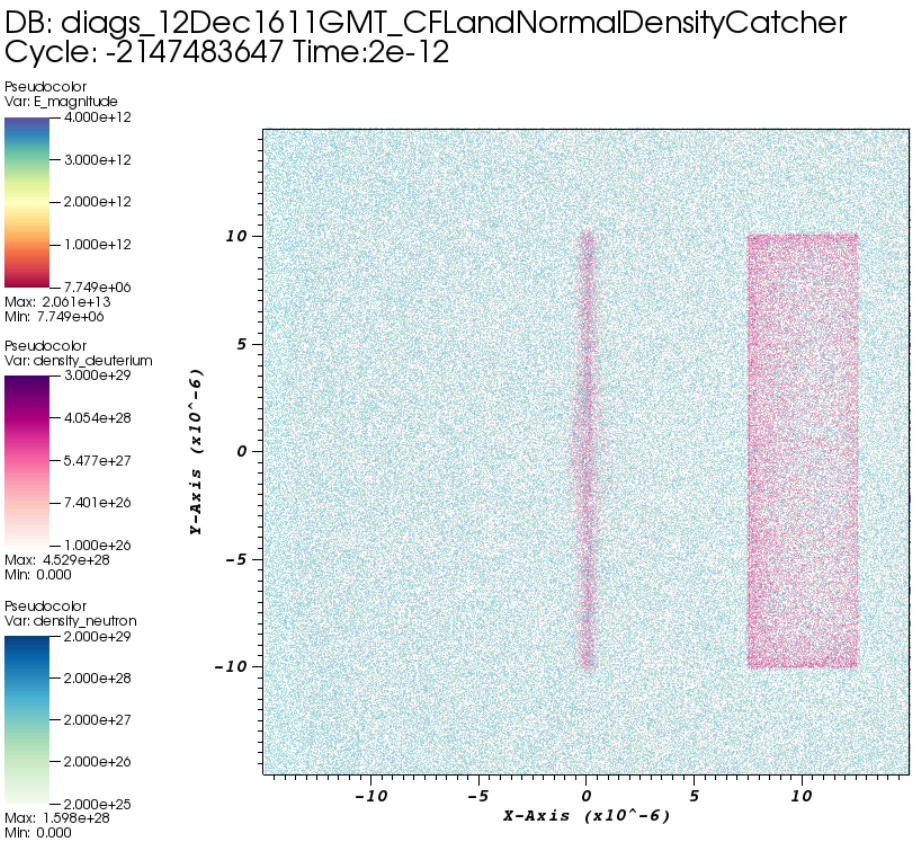

Figure 15: Simulation Frame 2000 fs.

5.3 Where Are Neutrons Generated?

The first time step that neutrons were present is at 48.8 fs. This presence was observed through a reduced diagnostic, which provide quick data collection at the cost of the data being exclusive and specific. Figure 16 shows the state of the system at 49 fs, where the generated neutrons can first be visualized. Similarly, Figure 17 shows the same moment in time with deuterium hidden from view so the generated neutrons can be seen more clearly. It can be seen from these figures that the target generates neutrons starting in the bulk of the material closest to the laser.

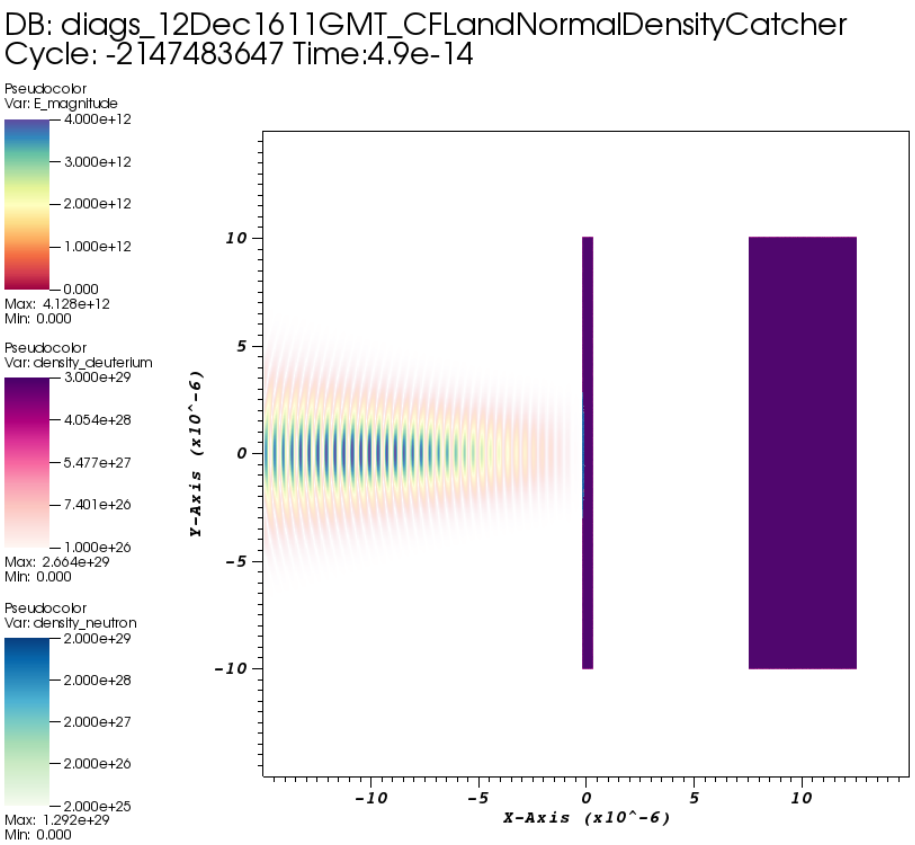

Figure 16: A still frame of the system at 49 fs. A blue streak on the left edge of the target representing newly formed neutrons can be observed. A more clear picture of the neutrons is available in Figure 17.

Simulation frames from Section 5.2 show that neutron generation proceeds from the bulk of the target closest to the laser to the surrounding fusible material until it envelops the target and begins escaping from the bounds of the target. Figure 11 shows the target enveloped in outgoing neutrons and the catcher in the throes of its own fusion process. The purpose of the catcher in this context is to generate more neutrons by adding more deuterons for those accelerated by the laser to hit. However, in the simulation, the catcher's fusion does not seem to be brought on by any deuteron collision from the target. There appears to be no trace of any deuterons transferring from the target area to the catcher at all in the few dozens of femtoseconds that it

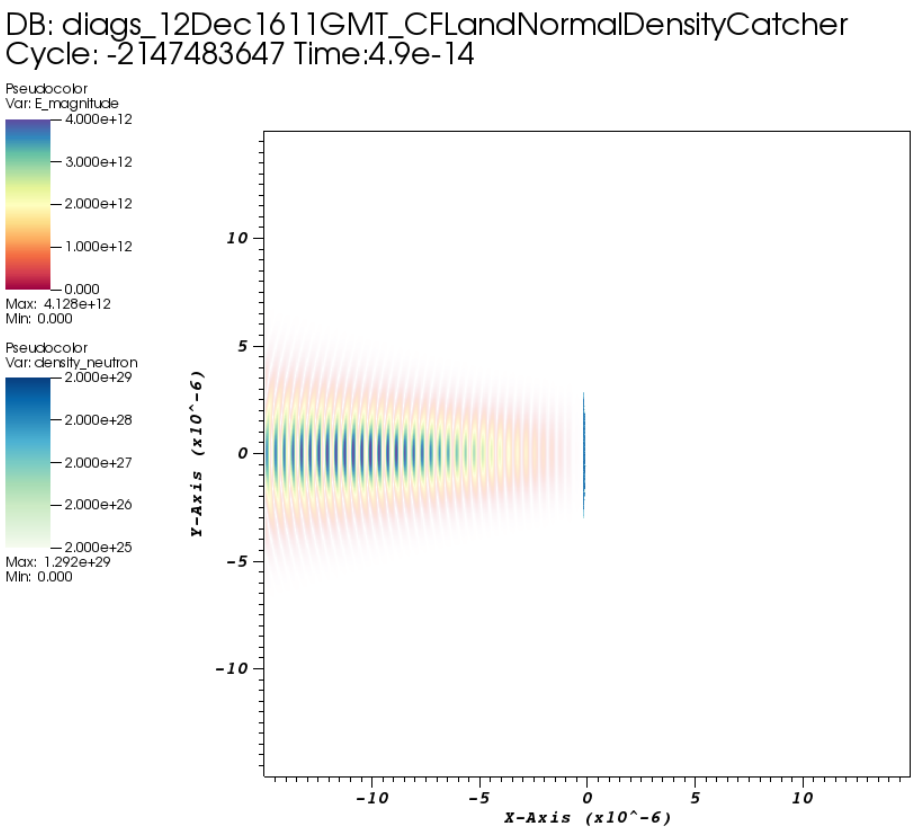

Figure 17: A still frame of the system shown in Figure 16 with all deuterium hidden so that the blue streak signalling a nonzero neutron density can more easily be seen.

takes the catcher to begin fusion. What causes the catcher to rapidly fuse deuterons in the simulation is unknown; it could be that extremely energetic electrons can bridge the gap between target and catcher, or that some energy from the laser manages to transmit through the target and impact the catcher. Numerical heating could also be taking place, as well. Perhaps only a small perturbation is needed to generate enough numerical heating that fusion can start and proliferate throughout a plasma.

Whatever the case, it seems that the catcher is not fulfilling its intended purpose in the simulation. Instead of introducing anisotropy by fusing with deuterons from the target propelled by the laser, it instead ignites on its own and simply seems to

create neutrons that spread out in an isotropic manner. Unintended behavior from the catcher was expected because of its difference from experimental parameters, however. Less than ten microns is magnitudes less distance than experiment, and was only done to constrain the system so that it can be simulated in a reasonable amount of time. If possible, future work should distance the catcher from the target to drive the simulation closer to experiment; more distance from the target could clean up this anomalous behavior as well.

5.4 How Does the Number of Neutrons Evolve with Time?

Figure 18 shows the sum of the neutron macroparticles' weight as a function of time. This can be interpreted as the total number of "actual" particles per length in the simulation. Because WarpX uses a distance of one meter for its y-axis in two-dimensional simulations, A particular point in time in Figure 18 can be interpreted as the total number of neutrons present in a theoretical 30 micron by 30 micron by 1 meter space.

The two plateaus in Figure 18 located around 75 fs and spanning 100-200 fs correlate strongly to the ratio of the area of the target and catcher. The area of the target is $0.5\ \mu m \times 20\ \mu m = 10\ \mu m^2$, while the area of the catcher is $5.0\ \mu m \times 20\ \mu m = 100\ \mu m^2$. This factor of ten is present in the difference in height between the neutron weight plateaus: The small plateau rests at about 3.20×10^{17} neutrons per length, while the taller plateau reaches its maximum value at 3.34×10^{18} neutrons per length. This data shows that when neutrons are created in this system, they are created very quickly and that after this quick creation almost no neutrons are made until another body of deuterons gets enough of a push to begin fusion. For this system, the moment fusion begins for the two separate instances of bodies of deuterium can be seen as the two plateaus in Figure 18.

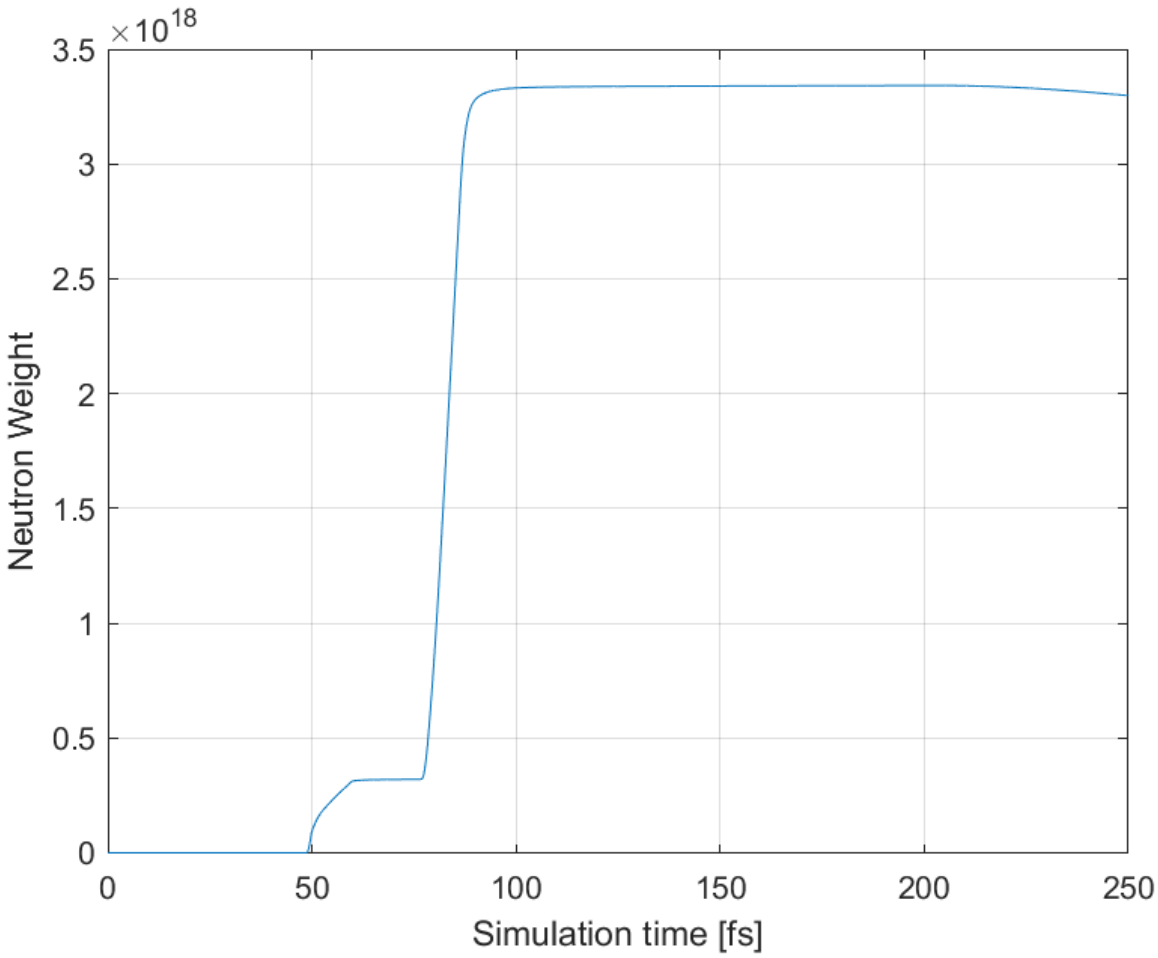

Figure 18: Total weight of neutrons vs time. This represents the total number of neutrons that are present in the simulation's 30 micron by 30 micron by 1 meter space.

Therefore, in this system, the number of neutrons as it evolves with time is dependent on the number of deuterons in the presence of the laser and their proximity to the laser. The neutron population rapidly increases when these bodies of deuterons are perturbed enough to begin fusion then remains relatively constant until another body of deuterons ignites.

Sharp-eyed readers may see that Figure 18 is the spitting image of Figure 8. Figure 19 shows the two plots normalized and combined for a more direct comparison. The two plots seem to perfectly overlap each other until later times in the simulation.

This correlation, along with the energy analysis in Section 5.1 and the neutron number plateau analysis in this section, is indicative that the fusion reactions that create neutrons are directly responsible for the spikes in energy and neutron count. For posterity and completeness, the short and long versions of the same plotted quantities will be shown.

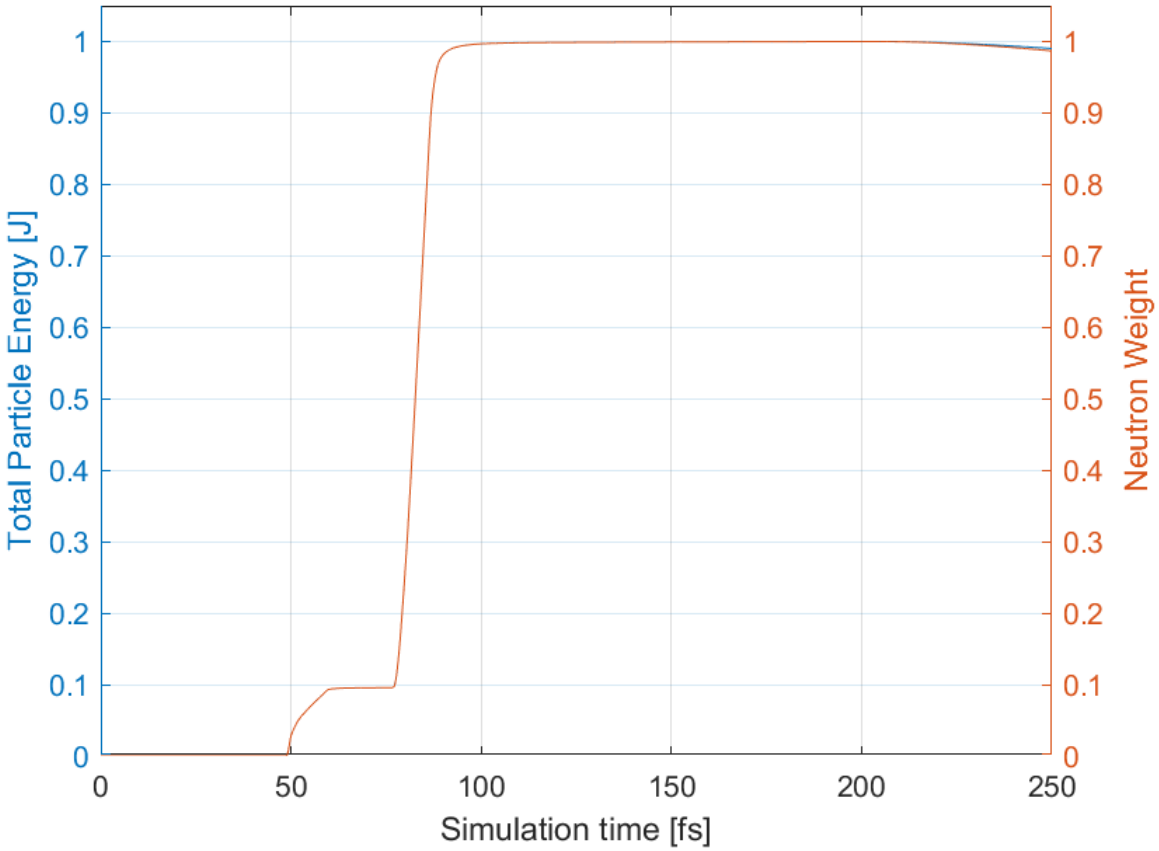

Figure 19: A double plot of neutron weight and particle energy normalized for viewing in the same plot.

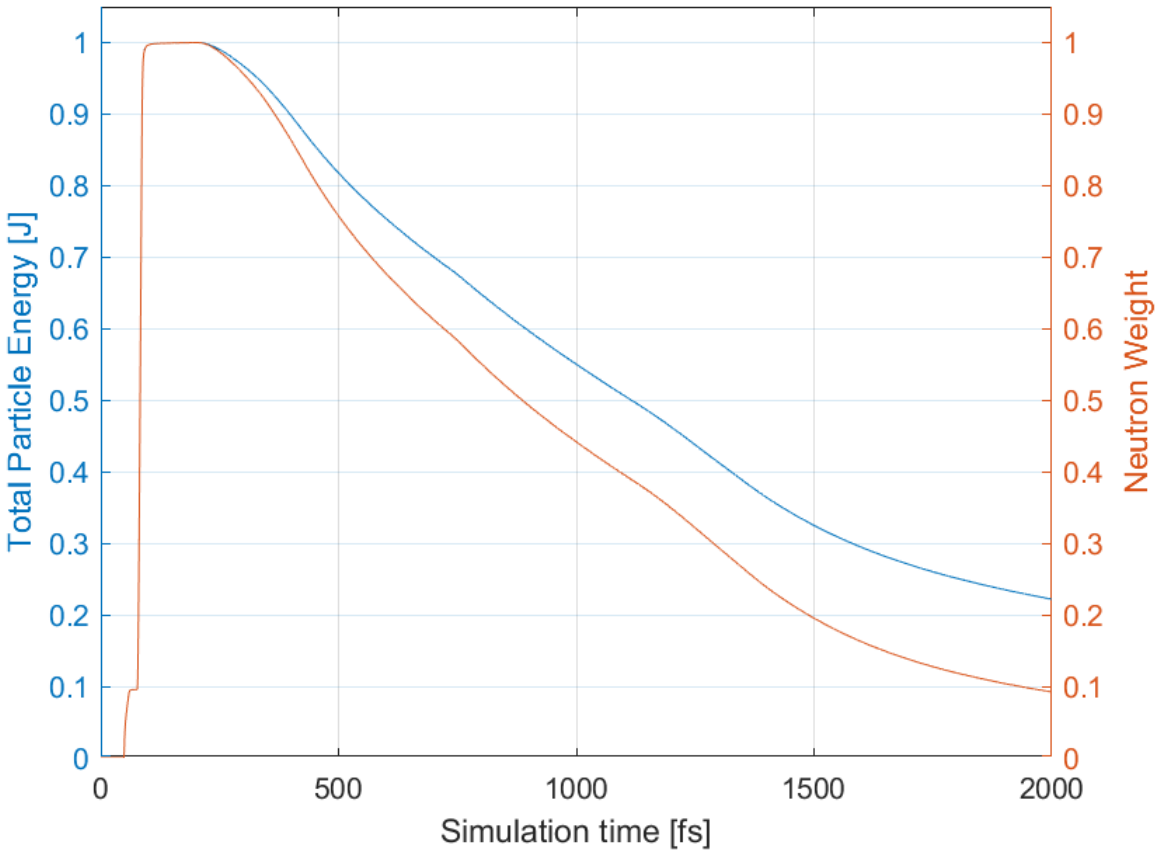

Figure 20: The double plot of neutron weight and particle energy normalized for viewing in the same plot, extended to the full time of the simulation.

5.5 Where Do Neutrons Propagate?

Generally, one would expect the presence of a catcher to induce some sort of anisotropy to the system. However, the mechanism through which anisotropy is to be induced depends on a certain amount of deuteron flux to impinge on the catcher. This flux of deuterons never seems to impinge on the catcher in the simulation, and the catcher instead ignites on its own, as shown in Figure 11 and Figure 12. The self-ignition seems to produce another isotropic shell of neutrons to erupt from the catcher, similar to the target after being struck by the laser, as shown in Figure 10. Because of this behavior, neutrons seem to propagate in an isotropic manner from the perspecive of the center of mass of the system.

However, due to time constraints, a more in-depth study of isotropy or anisotropy was not able to be conducted. Setting up a stable and converged fusion simulation should be the next goal of this continued project; because the simulation covered in this work does not resolve a stability condition by several magnitudes, its isotropic conclusion can be discussed but should be taken with a grain of salt. The author must leave an in-depth study of isotropy or angular dependence to other members of the ELG. Such future work may include acquisition of deuteron flux on the catcher, distancing the catcher from the target to reduce the chance of catcher ignition, and attempting to view the system from a further-away viewpoint to ascertain macro-scale angular dependence.

VI. Conclusions

This work has investigated three avenues of inquiry analyzing the nature of neutron generation using WarpX's new nuclear fusion model. Its novelty and main contribution comes from the use of this nuclear fusion model; the simulation used in this work was compiled with and executed by a beta version of the nuclear fusion model that is now available on the main branch of WarpX. This beta version was previously solely available on Remi Lehe's branch of WarpX on Github.

With this new tool, this work has attempted to analyze three aspects of the neutrons generated from a system modeled after a physical experiment hosted in the Extreme Light Laboratory (ELL). By analyzing simulation frames, the generation and propagation of neutrons was analyzed. It was observed that under the conditions of the simulation, neutrons were generated in the center of the target on the side facing the incoming laser. Following that, fusion reactions continued as neutrons were generated in the catcher starting in the side of the catcher facing the target. After their genesis, neutrons proceeded to travel in an isotropic manner in the center of mass frame. By analyzing diagnostics files, it was revealed that the number of neutrons present in the simulation lines up well with the times that the two bodies of deuterons were seen to undergo fusion. This can be seen in Figure 18. These fusion processes were completed very quickly, in the tens of femtoseconds, and would very quickly cease once enough deuterons were used up. The number of neutrons present in the simulation depended on how fast the bodies of deuterium could be perturbed and spiked whenever such a perturbation occurred.

However, there are issues with the simulation that are likely to affect its results. The most glaring of these issues is that the simulation fails to resolve the Debye length of the plasma present used in the simulation. This could cause numerical heating which could heat the plasma up by a factor of 1.3×10^9 [24]. Therefore,

although the system may appear to have isotropy, that result cannot be taken as the full truth because the simulation does not resolve a necessary stability condition by several magnitudes.

6.1 Future Work

An exciting element of using a new tool is that there are an abundance of ideas to try and refine. First and foremost, the simulation featured in this work should be brought more stability. This could be accomplished by investigating many methods. Increasing the spatial resolution of the simulation by reducing the cell size should help. A global increase in resolution certainly is one answer, but static mesh refinement on a specific area of the simulation could help increase resolution in needed areas while not incurring so much simulation time, which may be worthwhile. Lowering the density of the target is also an option for increasing stability. This may not reflect experiment most accurately, but it may be worthwhile to resolve the state of the simulation. Tied to this effort is a method to diagnose numerical heating in fusion simulations. This may have to include running fusion and non-fusion simulations to compare energy flow and other parameters. The energy deposited into particles by the laser is especially important to account for, so a double-check to confirm if the laser is depositing the energy as expected may also be fruitful.

There are also other potential refinements of the simulation to tweak so that it could emulate reality a bit more. One such refinement is extending this study to a three-dimensional simulation. Such an extension would also aid energy analysis and make comparisons to reality more straightforward. Adding the second D-D fusion reaction (See Equation (1)) into the simulation could also be useful to see if the other fusion products have an effect on the system. Distancing the catcher from the target could reduce the chance that the catcher starts fusing deuterons before deuterons from

the target impact it, and more closely emulate its desired behavior in experiment. An analysis of what distance is required to induce anisotropy could prove useful to future experiments, assuming such a distance exists.

Although an exhausting list, these examples are by no means exhaustive. There are many variables to tweak and many interesting aspects of this project that deserve further study.

6.2 Acknowledgements

This work was supported in part by high-performance computer time and resources from the DoD High Performance Computing Modernization Program.

This research used the open-source particle-in-cell code WarpX `https://github.com/ECP-WarpX/WarpX`, primarily funded by the US DOE Exascale Computing Project. Primary WarpX contributors are with LBNL, LLNL, CEA-LIDYL, SLAC, DESY, CERN, and Modern Electron. We acknowledge all WarpX contributors.

Appendix A. Python script to calculate a laser's maximum electric field

```
# -*- coding: utf-8 -*-
"""
Created on Mon Nov 14 13:29:45 2022

@authors: crsto, JSmith
"""

import numpy as np



def main():
    # Calculate Laser Energy
    c=299792458 #m/s - speed of light
    epsilon_0 = 8.85418782e-12 #F/m = C/(V*m) - permittivity of free space

    # Define input params
    sizeFWHM = 1.8 * 1e-6  # m - Spot Size given in FWHM
    durationFWHM = 40e-15 # s - Pulse duration given in FWHM
    energy_on_target = 7.7e-3    # J - Given energy on target
    # Electric field is numerically solved (guess/check) to match energy in 3D
    E_0 = 6.3e12 # V/m - guessed max E field of the laser
    print("Chosen E0 value: %.2e" %E_0)

    w_0 = sizeFWHM / np.sqrt(2*np.log(2)) # m - Radius of beam
```

```
    intensity = c * epsilon_0 * E_0**2 /2

    print ('Intensity: %.2e' % (intensity/100/100) +' W/cm^2'  )

    # 2D
    Ebeam2D = w_0*np.sqrt(np.pi/2)*E_0**2*epsilon_0*c/2*durationFWHM*1 #1 meter
    print("Calculated Beam energy 2D: " + str(Ebeam2D) + ' J')

    # 3D
    Ebeam3D = (w_0**2)*np.pi/2*intensity*durationFWHM # Power*time = Energy
    print("Calculated Beam energy 3D: " + str(Ebeam3D) + ' J')
    print("Given energy on target:    " + str(energy_on_target) + " J")
    print("Error to given value:      " + str(Ebeam3D-energy_on_target) + " J")

    return # Done


if __name__ == "__main__": main()
```

Appendix B. WarpX Input File Used for Featured Simulation

The input files used for WarpX runs can be input as text files. A separate input file is not included with this work but is instead copied from its original text file here. Ellipses mark spots in the text where a return line was added in this document to force text to fit within the bounds of the pages of this work.

```
# Authors: @crstoner & @RemiLehe
# Initialized 22 Nov 2022


#################################
####### GENERAL PARAMETERS ######
#################################
# Timing
#my_constants.mstep = 100000000
# If both max_step and stop_time, sim stops when first is hit.
#max_step = mstep
# Maximum physical time of sim
stop_time = 2000e-15


# Mesh
my_constants.NcellSide = 1024
amr.n_cell = NcellSide NcellSide
amr.max_grid_size = 128
amr.blocking_factor = 16
amr.max_level = 0 #mesh refinement disabled, 1 requires fine_tags
```

```
# Geometry / "Physical" domain
geometry.dims = 2
my_constants.geoSide = 15.e-6
geometry.prob_lo     = -geoSide  -geoSide
geometry.prob_hi     =  geoSide   geoSide

my_constants.cell_volume = (geoSide/NcellSide)*(geoSide/NcellSide)
#################################
###### Boundary Conditions ######
#################################
boundary.field_lo = pml pml
boundary.field_hi = pml pml

#################################
############ NUMERICS ###########
#################################
warpx.verbose = 1
warpx.cfl = 0.578861137937494000 # use this to resolve timestep

# Order of particle shape factors
algo.particle_shape = 2

#################################
############ PLASMA #############
#################################
```

```
particles.species_names = electron deuterium oxygen ...
electron2 hydrogen2 oxygen2 helium neutron

############ TARGET #############
electron.species_type = electron
electron.injection_style = "NRandomPerCell"
electron.num_particles_per_cell = 25
electron.profile = constant
electron.density = 33.3e28 # matches 10 X molecular density of D2O
electron.xmin =  -0.25e-6  # half micron thick target
electron.xmax =   0.25e-6
electron.zmin = -10.e-6
electron.zmax =  10.e-6
electron.momentum_distribution_type = "at_rest"

deuterium.species_type = deuterium
deuterium.injection_style = "NRandomPerCell"
deuterium.num_particles_per_cell = 100
deuterium.profile = constant
deuterium.density = 6.66e28 # matches 2 X molecular density of D2O
deuterium.xmin =  -0.25e-6
deuterium.xmax =   0.25e-6
deuterium.zmin = -10.e-6
deuterium.zmax =  10.e-6
deuterium.momentum_distribution_type = "at_rest"
```

```
oxygen.species_type = oxygen
oxygen.injection_style = "NRandomPerCell"
oxygen.num_particles_per_cell = 25
oxygen.profile = constant
oxygen.density = 3.33e28 # matches 1 X molecular density of D2O
oxygen.xmin =  -0.25e-6
oxygen.xmax =   0.25e-6
oxygen.zmin = -10.e-6
oxygen.zmax =  10.e-6
oxygen.momentum_distribution_type = "at_rest"

############ CATCHER #############
electron2.species_type = electron
electron2.injection_style = "NRandomPerCell"
electron2.num_particles_per_cell = 25
electron2.profile = constant
electron2.density = 33.3e28 # Mistake: should be 30.6e28
electron2.xmin =   7.5e-6
electron2.xmax =  12.5e-6
electron2.zmin = -10.e-6
electron2.zmax =  10.e-6
electron2.momentum_distribution_type = "at_rest"

hydrogen2.species_type = deuterium # heavy ice of the catcher
hydrogen2.injection_style = "NRandomPerCell"
hydrogen2.num_particles_per_cell = 100
```

```
hydrogen2.profile = constant
hydrogen2.density = 6.66e28 # Mistake: should be 6.124e28
hydrogen2.xmin =   7.5e-6
hydrogen2.xmax =  12.5e-6
hydrogen2.zmin = -10.e-6
hydrogen2.zmax =  10.e-6
hydrogen2.momentum_distribution_type = "at_rest"

oxygen2.species_type = oxygen
oxygen2.injection_style = "NRandomPerCell"
oxygen2.num_particles_per_cell = 25
oxygen2.profile = constant
oxygen2.density = 3.33e28 # Mistake: should be 3.062e28
oxygen2.xmin =   7.5e-6
oxygen2.xmax =  12.5e-6
oxygen2.zmin = -10.e-6
oxygen2.zmax =  10.e-6
oxygen2.momentum_distribution_type = "at_rest"

######## FUSION PRODUCTS ########
neutron.species_type = neutron
helium.species_type = helium3

##################################
############ LASER ##############
##################################
```

```
# Constants used to build laser - see Excel sheet
my_constants.e      = 2.7182818284
my_constants.E0     = 6.3e12        # V/m - Max electric field strength
my_constants.gouy   = 0.5318        # radians - Gouy Phase
my_constants.k      = 7.854e6       # radians/meter - wave number =2pi/800nm
my_constants.omega0 = 2.355e15      # radians/s - frequency = k*clight
my_constants.pi     = 3.14159265359
my_constants.PulseD = 40.0e-15      # s; Pulse duration described in FWHM
my_constants.RC     = 2.0901e-5     # m; Radius of Curvature
my_constants.w0     = 1.5288e-6     # m; beam waist radius minimum
my_constants.wz     = 2.8747e-6     # m; beam waist radius at target
my_constants.wFocus = sqrt(w0/wz)   # unitless; laser focus scaling in 2-D

# Gaussian in space, sin^2 in time

lasers.names        = laser1
laser1.profile      = parse_field_function
laser1.field_function(X,Y,t) = "-E0*wFocus*sin(omega0*t+k*((X**2)/(2*RC))-...
(gouy/2))*e**(-(X/wz)**2)*sin(pi*t/(PulseD*2))*(t<(PulseD*2))"
laser1.position     = -14.99e-6 0. 0.    # This point is on the laser plane
laser1.direction    = 1. 0. 0.           # The plane normal direction
laser1.polarization = 0. 1. 0.           # The main polarization vector
laser1.e_max        = 6.3e12             # Maximum amplitude of laser field [V/m]
laser1.wavelength   = 0.78e-6            # The wavelength of the laser [m]
```

```
##################################
########### COLLISION ############
##################################
collisions.collision_names = DDNHeF1 DDNHeF2 DDNHeF3

DDNHeF1.species = deuterium deuterium
DDNHeF1.product_species = helium neutron
DDNHeF1.type = nuclearfusion
DDNHeF1.fusion_multiplier = 1.e0

DDNHeF2.species = deuterium hydrogen2
DDNHeF2.product_species = helium neutron
DDNHeF2.type = nuclearfusion
DDNHeF2.fusion_multiplier = 1.e0

DDNHeF3.species = hydrogen2 hydrogen2
DDNHeF3.product_species = helium neutron
DDNHeF3.type = nuclearfusion
DDNHeF3.fusion_multiplier = 1.e0

##################################
########## DIAGNOSTICS ##########
##################################
```

```
########## FULL DIAGS ###########
my_constants.intVar = 25 #mstep/50
my_constants.RedintVar = 1
diagnostics.diags_names = diag1
diag1.intervals = intVar
diag1.diag_type = Full
diag1.fields_to_plot = Ex Ey Ez part_per_cell

diag1.particle_fields_to_plot = density
diag1.particle_fields.density(x,y,z,ux,uy,uz) = 1./cell_volume
diag1.particle_fields.density.do_average = 0

######### REDUCED DIAGS ##########
warpx.reduced_diags_names = particle_num particle_ene particle_mom ...
deuteron_KE_Hist hydrogen2_KE_Hist electron_KE_Hist neutron_KE_Hist ...
neut_x neut_z

particle_num.intervals = RedintVar
particle_num.type = ParticleNumber
particle_num.separator = ","
particle_ene.intervals = RedintVar
particle_ene.type = ParticleEnergy
particle_ene.separator = ","
particle_mom.intervals = RedintVar
particle_mom.type = ParticleMomentum
particle_mom.separator = ","
```

```
# Neutron Kinetic Energy
neutron_KE_Hist.intervals = RedintVar
neutron_KE_Hist.type = ParticleHistogram
neutron_KE_Hist.species = neutron
# Relativistic KE = (gamma-1)*mc^2. [ux = gamma (vx/c)].
# gamma = sqrt(1 + ux^2 + uy^2 + uz^2)
neutron_KE_Hist.histogram_function(t,x,y,z,ux,uy,uz) = ...
(sqrt(1 + ux*ux + uy*uy + uz*uz)-1)*1.673e-27*clight*clight
neutron_KE_Hist.bin_number = 100
neutron_KE_Hist.bin_max = 1.602e-12 #J; should be ~= 10 MeV
neutron_KE_Hist.bin_min = 0;

# Deuteron Kinetic Energy;
deuteron_KE_Hist.intervals = RedintVar
deuteron_KE_Hist.type = ParticleHistogram
deuteron_KE_Hist.species = deuterium
deuteron_KE_Hist.histogram_function(t,x,y,z,ux,uy,uz) = ...
(sqrt(1 + ux*ux + uy*uy + uz*uz)-1)*3.34e-27*clight*clight
deuteron_KE_Hist.bin_number = 100
deuteron_KE_Hist.bin_max = 1.602e-12 #J; should be ~= 10 MeV
deuteron_KE_Hist.bin_min = 0;

# hydrogen2 Kinetic Energy; Deuteron KE but for the other instance.
hydrogen2_KE_Hist.intervals = RedintVar
hydrogen2_KE_Hist.type = ParticleHistogram
```

```
hydrogen2_KE_Hist.species = hydrogen2
hydrogen2_KE_Hist.histogram_function(t,x,y,z,ux,uy,uz) = ...
(sqrt(1 + ux*ux + uy*uy + uz*uz)-1)*3.34e-27*clight*clight
hydrogen2_KE_Hist.bin_number = 100
hydrogen2_KE_Hist.bin_max = 1.602e-16 #J; should be ~= 1 keV
hydrogen2_KE_Hist.bin_min = 0;

# Electron Kinetic Energy
electron_KE_Hist.intervals = RedintVar
electron_KE_Hist.type = ParticleHistogram
electron_KE_Hist.species = electron
electron_KE_Hist.histogram_function(t,x,y,z,ux,uy,uz) = ...
(sqrt(1 + ux*ux + uy*uy + uz*uz)-1)*9.11e-31*clight*clight
electron_KE_Hist.bin_number = 100
electron_KE_Hist.bin_max = 1.602e-12 #J; should be ~= 10 MeV
electron_KE_Hist.bin_min = 0;

neut_x.intervals = RedintVar
neut_x.type = ParticleHistogram
neut_x.species = neutron
neut_x.histogram_function(t,x,y,z,ux,uy,uz) = x
neut_x.bin_number = 100
neut_x.bin_max = geoSide #m - matches geometry.prob_lo/hi
neut_x.bin_min = -geoSide
```

```
neut_z.intervals = RedintVar

neut_z.type = ParticleHistogram

neut_z.species = neutron

neut_z.histogram_function(t,x,y,z,ux,uy,uz) = z

neut_z.bin_number = 100

neut_z.bin_max = geoSide #m - matches geometry.prob_lo/hi

neut_z.bin_min = -geoSide
```

Appendix C. Useful Algorithms for WarpX and VisIt

A roadblock one comes to when working with any new hardware or software is the time it takes to become acquainted with the system. HPC systems use Linux and have little to no GUI, so operations must be executed from the command line. Both HPC systems and WarpX were completely new to the author when he started his research. As such, good note-taking about using the systems was paramount. This section provides step-by-step WarpX guides that the author used time and time again when executing the computational part of this work.

3.1 How to Prepare and Run WarpX on a new system

To install WarpX and get it running on a new system, these steps were followed. Linux command line inputs are provided in case users are completely new to their system. This process, called the GNUmake Build System, is a legacy or outdated process to install, compile, and run WarpX. New users should probably follow WarpX's current install process, found here: `https://warpx.readthedocs.io/en/latest/install/users.html`. In this section, phrases in *italics* are command line inputs.

1. Create a directory with a strong name and clone the WarpX repository of choice. Make directories for each import:
   (a) *mkdir [warpx]*
   (b) *cd [warpx]*
   (c) *git clone https://github.com/ECP-WarpX/WarpX.git ./WarpX*
   (d) *git clone https://github.com/ECP-WarpX/picsar.git ./picsar*
   (e) *git clone https://github.com/ECP-WarpX/warpx-data.git ./warpx-data*
   (f) *git clone https://github.com/AMReX-Codes/amrex.git ./amrex*

2. Modify AMReX make.unknown file:

   (a) *cd [warpx]/amrex/Tools/GNUMake/sites*

   (b) *nano make.unknown*

   (c) change CC/cc/ftn/ftn90 to their corresponding values in the system's user guide – look for Available Compilers and find your system's compiler commands

3. Modify GNUmakefile:

   (a) *cd [warpx]/WarpX*

   (b) *nano GNUmakefile*

   (c) At the top of the document, type NO_MPI_CHECKING = TRUE

   (d) Make sure DIM is set to what you need for your sim (1,2,3)

   (e) Set COMP to your machine's environment (cray, intel, gnu, etc.)

   (f) Set everything that has a boolean input to FALSE except WARN_ALL and TINY_PROFILE.

   (g) Output file appears in "Bin" directory, titled something like "main2d.cray.broadwell.TPROF.MTMPI.ex"

4. Compile WarpX:

   (a) In [warpx]/WarpX: *make clean*

   (b) Run a short test compile to make sure nothing is terribly wrong: *make*

   (c) If everything seems to be running, cancel and run with more cores:

   (d) *ctrl+C* then *make -j 20*

   (e) Look for a SUCCESS message; if errors come up, troubleshooting is needed.

After these preparations are done, WarpX can be compiled and a simulation can be executed using the following:

1. Find or make an inputs file, edit it to the specifications you want, and put it in a new working directory with the executable output file. Example input files can be found in Warpx/Examples/Tests/[subject of interest]. Input parameters can be found in WarpX documentation.

2. Create a file to interface with job scheduling software: *nano run_warpx.sh*. Portable Batch System is an example. Reference the documentation in your system's user guide. Input required parameters, optional parameters, and finally the execution block, where the system will be told what to perform, for example: *aprun -n [match number specified in required parameters] [path to executable] [path to input file] > warpx.out*

3. Submit the job: *qsub run_warpx.sh*. After, the job can be checked with *qstat* or stopped with *qdel* (commands used are for PBS systems).

4. After the job stops, check your working directory. If Backtrace.* files are present, troubleshooting will need to be done. If a diag (diagnostic) directory is present, the run completed successfully. If some diag directories appear to be missing, it may be required to wait a while for the files to be written.

3.2 Visualization of WarpX Output Files with VisIt

Once diag directories are present, the files can now be visualized. This section serves as a step-by-step guide to visualize a simulation's results with VisIt, a mass data visualization software.

Given a diag directory from a successful WarpX run, follow these steps to create a **figure** of your data with VisIt:

1. Give the diag directory a unique name: *mv [diag directory path] [diag unique name]*

2. Copy or move the diag directory to somewhere your installation of VisIt can access: *mv [diag dir] [VisIt access Path]*

3. Open VisIt. Once everything loads in, go to file→ open... and find your diag directory.

4. Find a Header file, click it, then click OK. For a single figure from a specific simulation step, you will have to go into a sub-directory of the original diag directory.

5. On the VisIt toolbar, click add, navigate to a visualization option you would like, and click a quantity you're interested in visualizing. To start out, pseudocolor and some sort of magnitude option are recommended.

6. Click draw. A figure should appear after loading.

7. To save, go to File→save settings, customize your save file, and click Save.

Given a diag directory from a successful WarpX run, follow these steps to create a **movie** of your simulation with VisIt:

1. Give the diag directory a unique name.

2. Create a movie.visit file. This is a file with the path of all Header files from each diag sub-directory within the main diag directory. It can be created for you by AMReX. To do this, navigate to the diag directory and run the following in the command line: *ls -1 diag1*/Header | tee movie.visit*. The diag1* part of the command is meant to search all files that start with diag1; you may need to replace diag1*/Header with [whatever your diag sub-directories start with]*/Header.

3. Copy or move the diag directory to somewhere your installation of VisIt can access: *mv [diag dir] [VisIt access Path]*

4. Open VisIt. Go to File→open... and click on the movie.visit file. Click OK.

5. On the VisIt toolbar, click add, navigate to a visualization option you would like, and click a quantity you're interested in visualizing. To start out, pseudocolor and some sort of magnitude option are recommended.

6. Click Draw and press Play on the video slider to view the movie.

7. To save, go to File→save movie, customize your settings, and save.

Bibliography

1. J.-L. Vay, A. Huebl, A. Almgren, L. D. Amorim, J. Bell, L. Fedeli, L. Ge, K. Gott, D. P. Grote, M. Hogan, R. Jambunathan, R. Lehe, A. Myers, C. Ng, M. Rowan, O. Shapoval, M. Thévenet, H. Vincenti, E. Yang, N. Zaïm, W. Zhang, Y. Zhao, and E. Zoni. Modeling of a chain of three plasma accelerator stages with the warpx electromagnetic pic code on gpus. *Physics of Plasmas*, 28(2):023105, 2021.

2. R. Jambunathan, D. Willcox, A. Myers, and WarpX Team. Fully-kinetic Plasma Simulations of Pulsar Magnetospheres using WarpX. In *American Astronomical Society Meeting Abstracts*, volume 53 of *American Astronomical Society Meeting Abstracts*, page 152.02, January 2021.

3. Elisa Rheaume, Lorenzo Giacomel, Jean-Luc Vay, and Axel Huebl. Development of a virtual em detector for the advanced particle accelerator modeling code warpx, 2022.

4. Anil Patnaik et al. Relativistic laser-plasma interactions with liquid targets at high repetition rate. Technical report, Air Force Institute of Technology, Wright-Patterson AFB, OH, Mar 2022.

5. John T Morrison, Scott Feister, Kyle D Frische, Drake R Austin, Gregory K Ngirmang, Neil R Murphy, Chris Orban, Enam A Chowdhury, and W M Roquemore. MeV proton acceleration at kHz repetition rate from ultra-intense laser liquid interaction. *New Journal of Physics*, 20(2):022001, feb 2018.

6. Charlotte Palmer. Paving the way for a revolution in high repetition rate laser-driven ion acceleration. *New Journal of Physics*, 20(6):061001, jun 2018.

7. Liyuan Liang, Romano Rinaldi, and Helmut Schober. *Neutron applications in earth, energy and environmental sciences.* Springer Science & Business Media, 2008.

8. R. Van Langh, E. Lehmann, S. Hartmann, A. Kaestner, and F. Scholten. The study of bronze statuettes with the help of neutron-imaging techniques. *Anal Bioanal Chem*, 395(7), 2009.

9. Thomas Watkins, Hassina Bilheux, Ke An, Payzant Andrew, Ryan Dehoff, Chad Duty, William Peter, Craig Blue, and Craig Brice. Neutron characterization for additive manufacturing. *Advanced Materials and Processes*, 171(3), 2013.

10. Aaron E. Craft and John P. Barton. Applications of neutron radiography for the nuclear power industry. *Physics Procedia*, 88:73–80, 2017. Neutron Imaging for Applications in Industry and Science Proceedings of the 8th International Topical Meeting on Neutron Radiography (ITMNR-8) Beijing, China, September 4-8, 2016.

11. K. M. George, J. T. Morrison, S. Feister, G. Ngirmang, J. R. Smith, A. J. Klim, J. Snyder, D. Austin, W. Erbsen, K. D. Frische, J. Nees, C. Orban, E. A. Chowdhury, and W. M. Roquemore. High repetition rate (khz) targets and optics from liquid microjets for the study and application of high intensity laser-plasma interactions, 2019.

12. Kenneth Krane. *Introductory Nuclear Physics.* Wiley, 2nd edition, 1987, 529.

13. M P Taggart, C Payne, and P J Sellin. Neutron-gamma discrimination via PSD plastic scintillator and SiPMs. *Journal of Physics: Conference Series*, 763:012007, oct 2016.

14. Willem G. J. Langeveld, Michael J. King, John Kwong, and Daniel T. Wakeford. Pulse shape discrimination algorithms, figures of merit, and gamma-rejection for liquid and solid scintillators. *IEEE Transactions on Nuclear Science*, 64(7):1801–1809, 2017.

15. Glenn F. Knoll. *Radiation detection and measurement*. Wiley, 4th edition, 2010, 223-275.

16. Mohamad Shalaby, Avery E. Broderick, Philip Chang, Christoph Pfrommer, Astrid Lamberts, and Ewald Puchwein. Sharp: A spatially higher-order, relativistic particle-in-cell code. *The Astrophysical Journal*, 841(1):52, 2017.

17. Hirotade Abe, Natsuhiko Sakairi, Ryohei Itatani, and Hideo Okuda. High-order spline interpolations in the particle simulation. *Journal of Computational Physics*, 63(2):247–267, 1986.

18. Barry Marder. A method for incorporating gauss' law into electromagnetic pic codes. *Journal of Computational Physics*, 68(1):48–55, 1987.

19. T.Zh. Esirkepov. Exact charge conservation scheme for particle-in-cell simulation with an arbitrary form-factor. *Computer Physics Communications*, 135(2):144–153, 2001.

20. Kane Yee. Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on Antennas and Propagation*, 14(3):302–307, 1966.

21. WarpX Contributors. Particle-in-cell method — warpx 22.11 documentation.

22. John M. Dawson. Particle simulation of plasmas. *Rev. Mod. Phys.*, 55:403–447, Apr 1983.

23. J. P. Boris. Relativistic plasma simulation-optimization of a hybrid code. *Proceeding of Fourth Conference on Numerical Simulations of Plasmas*, November 1970.

24. David Minot Day. Numerical experiments on unstructured pic stability., 4 2011.

25. Drew Pitney Higginson, Anthony Link, and Andrea Schmidt. A pairwise nuclear fusion algorithm for weighted particle-in-cell plasma simulations. *Journal of Computational Physics*, 388:439–453, 2019.

26. Tomonor Takizuka and Hirotada Abe. A binary collision model for plasma simulation with a particle code. *Journal of Computational Physics*, 25(3):205–219, 1977.

27. Dan Hill. How to convert FWHM measurements to $1/(e^2)$ halfwidths, 3 2021.

28. Orazio Svelto. *Principles of Lasers*. Springer, 5th ed. 2010 edition, 12 2009.

29. www.nuclear power.com. Density of Heavy Water, 11 2021.

Acronyms

AFIT Air Force Institute of Technology. iv, 1, 3

CFL Courant-Friedrichs-Lewy. 19, 20, 36

E&M electricity and magnetism. 16

ELG Extreme Light Group. v, 1, 25, 27, 35, 52

ELL Extreme Light Laboratory. iv, 1, 3, 5, 26, 53

EM electro-magnetic. 1, 14, 15, 20

FDTD finite-difference time-domain. 14

HIL high intensity laser. iv, 1, 2, 3, 5, 7, 11

LDF laser-driven fusion. 3, 5, 9, 11, 35

NIF National Ignition Facility. 1

PIC particle-in-cell. viii, 1, 11, 12, 19, 21

PML perfectly matched layer. 16

PSD pulse shape discrimination. 2, 7, 8

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
19–03–2023	Master's Thesis	Sept 2021 — Mar 2023

4. TITLE AND SUBTITLE

Thesis Title
with a Second Line

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

First M. Last

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-20-M-XXX

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFXX/XXXX
Building XXX
WPAFB OH 45433-7765
DSN XXX-XXXX, COMM 937-XXX-XXXX
Email: first.last@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

XXXX/XXXX

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Short abstract paragraph text here.

15. SUBJECT TERMS

subject terms here

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Captain First M. Last, AFIT/ENG
U	U	U	UU	XXX	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636, ext XXXX; first.last@afit.edu

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18