Documentation for hybrid.F90
A proposed extension to EPOCH

Stuart Morris
York Plasma Institute

August 8, 2022

Overview

This document describes the implementation of an electron-transport mode to EPOCH. We give a brief overview to the background physics and applications, how these have been implemented and benchmarked in the EPOCH framework, and include a user-manual section to explain how to access these modules from the input deck. Originally this was intended to be sent to Warwick to be added to the main branch of EPOCH, but this extension may be deemed too large to add. In this case, the report will serve as documentation for the code when requested by interested parties. I reserve the rights to copy large parts of this report into my thesis.

Contents

1 Introduction

Particle-in-cell (PIC) simulations offer a computationally feasible framework for studying the interaction of high intensity lasers with matter, and have seen widespread use in modelling laser-wakefield [1], laser-electron-beam [2], and laser-solid systems [3–5]. In order to model solid materials in PIC codes, the user must represent them as a cold, dense plasma, which requires a very high resolution spatial grid and many macroparticles per cell to suppress the effects of numerical self-heating. These computational demands place heavy restrictions on the size and timescales of these systems, and most authors limit themselves to only 2D simulations.

Hybrid-PIC codes (also called electron transport codes) present an alternative way to model solids in the PIC framework, treating the solid as a background fluid. This offers a significant speed boost as we no longer have huge numbers of solid macroparticles to track, which opens the possibility for 3D simulations of laser-solid systems. Additional hybrid routines are added to PIC codes which describe how electron temperatures and currents in the solid evolve over time, which influence the electric and magnetic fields within the solid. The laser is replaced by a hot electron injector which roughly characterises the expected hot electron distribution from the laser pulse, and further hybrid routines describe the interaction of hot electrons with the solid by modifying their trajectories and energies.

Several groups have already developed hybrid-PIC codes to describe laser-solid physics, but existing hybrid codes like `ZEPHYROS` (RAL) and `THOR` (AWE) are designed for low energy electrons, and `LSP` (Northrop Grunmman) is a commercial code which isn't open-source. As multi-petawatt lasers start to become readily available, hot electron energies could exceed 100 MeV in these targets, which would require high energy physics packages like bremsstrahlung radiation and Møller scatter to accurately model their propagation.

We have developed a hybrid extension to `EPOCH`, which follows the traditional hybrid-PIC framework of codes like `ZEPHYROS`, with additional high-energy subroutines taken from `Geant4`. In Section 2 we detail the equations and models which underpin the extension, while Section 3 describes the implementation of these models into the `EPOCH` framework. This code has been rigorously benchmarked against experiments where available, and other simulation codes otherwise, and a summary of these tests has been presented in Section 4. Finally, we include a user-manual style chapter in Section 5 which describes how to run a hybrid simulation in `EPOCH`.

2 Theory

In high intensity laser-solid interactions, there is sufficient energy in the laser pulse to ionise the front surface of the target, creating a pre-plasma of ions and electrons. While both electrons and ions can absorb laser energy in the pre-plasma, the lighter electrons reach higher speeds, and a current of high energy (hot) electrons is injected into the target.

The currents, fields, ionisation state, resistivity and temperature of the solid vary both spatially and temporally in these systems. The injected hot electron current can be very high, exceeding the Alfvén limit and generating a return current response in the electrons of the solid [6]. This return current and its corresponding fields can be solved in terms of the hot electron current and the temperature-dependent resistivity of the solid. The hot electron current dissipates power as it passes through the resistive medium, which rises the electron-temperature of the solid locally, altering the local resistivity and ionisation state of the solid atoms. The thermal energy gained by the solid electrons is then slowly transferred to the solid ions through collisions.

The solid also influences the propagation of the hot electrons. Elastic scatter between hot electrons and solid atoms affects the electron trajectories without reducing their speed, leading to a broadening of the hot current. Hot electrons also undergo ionisation energy loss, where energy is lost to exciting background electrons to higher energy states, again raising the electron temperature of the solid.

At higher energies, hot electron collisions with electrons from the background solid atoms are capable of transferring large amounts of energy and fully ionising electrons in the solid, leading to the creation of secondary hot electrons, or "δ-rays". Bremsstrahlung radiation also becomes an important energy loss mechanism for high energy electrons in solids.

Once hot electrons break through the rear of the target, the highest energy electrons escape, but the rest cannot overcome the attraction to the positive ions left behind. These trapped electrons establish a negative charge sheath on the surface which acts to help reflect electrons back into the target, and also to pull ions out in a process termed target normal sheath acceleration (TNSA). Some electron energy is lost to the ions accelerating out of the target, and electrons typically reflect back into the target with lower energy than they left with. These electrons will then continue to reflect between target boundaries, losing energy to bremsstrahlung, resistive fields, ionisation loss, and TNSA until they run out of energy or escape.

The remainder of this section will look at each of these processes in more detail (except for bremsstrahlung, which already exists in `EPOCH` and was documented in my previous report).

2.1 Electron injection

The hot electron injection is one of the greatest uncertainties in these systems. In general, the nature of hot electrons in a laser-solid interaction will depend on the pre-plasma density profile, laser intensity, and the temporal and spatial structure of the laser-pulse. Electron transport codes often approximate this distribution by considering a small number of easily customisable parameters, including an efficiency parameter $\eta_{l\rightarrow e}$ describing the fraction of laser energy absorbed by the electrons in the pre-plasma.

According to pondermotive scaling, the mean energy of the electron distribution is approximately $a_0 m_e c^2$, where the normalised vector potential a_0 is given by

$$a_0 = \frac{eE_0}{m_e \omega c} \tag{2.1}$$

with E_0 and ω denoting the peak electric field and angular frequency of the laser cycle, $-e$ and m_e are the charge and mass of the electron, and c is the speed of light. The instantaneous intensity of a laser pulse I_{inst} is given by

$$I_{\text{inst}} = \epsilon_0 c E(t)^2 \tag{2.2}$$

where ϵ_0 is the permittivity of free space. As the electric field of the laser pulse oscillates over a much shorter timescale than the laser pulse envelope varies, we instead use the cycle-averaged intensity, I when referring to laser intensity,

$$I = \frac{1}{2}\epsilon_0 c E_0^2, \tag{2.3}$$

which allows us to rewrite (2.1) into the more convenient form

$$a_0 = \sqrt{\frac{e^2}{2\pi^2 c^5 m_e^2 \epsilon_0} I\lambda^2}, \tag{2.4}$$

or $a_0 \approx 8.5 \times 10^{-10}\sqrt{I_{\text{Wcm}^{-2}}\lambda_{\mu\text{m}}{}^2}$, where intensity and wavelength are expressed in the more common units of Wcm^{-2} and μm respectively.

To determine how many electrons must be injected at a given point and time on the solid surface, $(\mathbf{r}, t)$, we have to characterise the spatial $f(\mathbf{r})$ and temporal $g(t)$ distributions of the laser respectively. If these represent fractions of the peak laser intensity I_0, then we have the laser intensity $I(\mathbf{r}, t) = I_0 f(\mathbf{r}) g(t)$. The laser energy passing through a small area dA at position $\mathbf{r}$ over times t to $t + dt$ is $I(\mathbf{r}, t)dAdt$, and if the fraction $\eta_{l\to e}$ is transferred to hot electrons of average energy $\langle\epsilon\rangle = a_0(\mathbf{r}, t)m_e c^2$, then the number of electrons to inject, $N(\mathbf{r}, t)$ is

$$N(\mathbf{r}, t) = \frac{I_0 f(\mathbf{r}) g(t) dA dt \eta_{l\to e}}{\langle\epsilon\rangle}, \tag{2.5}$$

where $a_0(\mathbf{r}, t)$ is evaluated using $I(\mathbf{r}, t)$.

Typically hot electrons follow an exponential energy distribution, but the angular distribution is less well understood. Two approaches to describing angular distributions were developed by Moore *et al* [7] and Sheng *et al* [8]. In Moore scaling, the electrons are injected into a cone with an energy dependent cone angle, θ_c

$$\tan(\theta_c) = \sqrt{\frac{2}{\gamma - 1}} \tag{2.6}$$

where the cone angle is relative to the laser axis, and γ denotes the electron Lorentz factor. The Sheng model describes oblique laser pulses which make an angle α to the target normal, and shows the expected angle of electron injection, θ_i is

$$\frac{1}{\tan(\theta_i)} = \sqrt{\frac{2}{\gamma - 1} + \frac{\gamma + 1}{(\gamma - 1)\tan^2(\alpha)}} \tag{2.7}$$

if the scalar potential field is roughly uniform throughout the interaction.

2.2 Hybrid field solver

The hybrid field solver follows the method described by Davies *et al* [9, 10]. We assume the propagation of hot electrons is neutralised by a return current of background electrons, which establishes an Ohmic electric field, $\mathbf{E}$ due to the resistivity, η of the solid

$$\mathbf{E} = \eta \mathbf{j}_b \tag{2.8}$$

where $\mathbf{j}_b$ is the current density of the background electrons. Assuming total current density $\mathbf{j}$ is the sum of $\mathbf{j}_b$ and the hot electron current density $\mathbf{j}_h$, the Ampère-Maxwell law can be written as

$$\mathbf{E} = \eta \left(\frac{1}{\mu_0} \nabla \times \mathbf{B} - \mathbf{j}_h \right) \tag{2.9}$$

where μ_0 is the permeability of free space. We have neglected the displacement current using the arguments of Davies *et al* [9], where they assert this is only important while the return current establishes itself. They calculate this timescale to be of order $\epsilon_0 \eta$ (about 0.02 fs for aluminium targets), which is negligible over picosecond pulses.

The magnetic field evolves according to the Faraday-Lenz law

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E} \tag{2.10}$$

as in regular PIC codes.

2.3 Resistivity

The resistivity of the solid determines the magnitude of the electric field created by the return current, and also determines the power dissipation of electrons in the solid for Ohmic heating. The resistivity of a solid is a function of the electron temperature, and is typically evaluated using the Lee-More model in hybrid PIC codes [11].

Lee and More quote the electrical conductivity ($1/\eta$) in (23a [11]), which corresponds to a resistivity of

$$\eta = \frac{m_e}{Z^* n_i e^2 \tau A^\alpha (\mu / k_B T_e)} \tag{2.11}$$

where Z^* and n_i are the ion charge state and ion number density of the background solid respectively, τ is the average electron relaxation time, and A^α is a function of the chemical potential μ, the Boltzmann constant k_B and the electron temperature T_e. We have opted for a reduced model which has no dependence on the chemical potential in our code, but it helps to discuss the full model to explain the approximations we have made. The remainder of this section will describe how to evaluate the terms appearing in (2.11).

2.3.1 Ionisation state

The ion charge state, Z^* can be calculated using a fit to the Thomas-Fermi ionisation states given by More [12] in their Table IV, reproduced in our Table 1 for completeness.

Algorithm	Parameters	
$T_0 = T_{e,eV} Z^{-4/3}$	a_1	3.323×10^{-3}
$R = \rho_{g/cm^3}/AZ$	a_2	0.9718
$T_F = T_0/(1+T_0)$	a_3	9.26148×10^{-5}
$A = a_1 T_0^{a_2} + a_3 T_0^{a_4}$	a_4	3.10165
$B = -\exp(b_0 + b_1 T_F + b_2 T_F^7)$	b_0	-1.7630
$C = c_1 T_F + c_2$	b_1	1.43175
$Q_1 = AR^B$	b_2	0.31546
$Q = (R^C + Q_1^C)^{1/C}$	c_1	-0.366667
$x = \alpha Q^\beta$	c_2	0.983333
$Z^* = Zx/(1 + x + \sqrt{1+2x})$	α	14.3139
	β	0.6624

Table 1: Algorithm for determining the ionisation state of the solid background [12]. Here we have used the solid mass density (in g/cm^3), ρ_{g/cm^3}, with A and Z representing the mass number and atomic number of the solid atoms respectively, and $T_{e,eV}$ denoting the background electron temperature in eV.

2.3.2 Electron relaxation time

In the plasma regime, τ is given by (24 [11]), which after conversion to SI units reads

$$\tau = 24\pi\epsilon_0^2 \sqrt{\frac{m_e}{2}} \frac{(k_B T_e)^{3/2}}{(Z^*)^2 n_i e^4 \ln\Lambda} \left(1 + e^{-\mu/k_B T_e}\right) F_{1/2}\left(\frac{\mu}{k_B T_e}\right) \tag{2.12}$$

where $\ln\Lambda$ is the Coulomb logarithm, $F_{1/2}$ is a Fermi-Dirac integral of the form

$$F_{1/2}(x) = \int_0^\infty \frac{t^{1/2} dt}{1 + e^{t-x}} = -\frac{\sqrt{\pi}}{2} Li_{3/2}(-e^x) \tag{2.13}$$

and $Li_{3/2}$ is a polylogarithm function of order 3/2. Lee and More's Section III.E [11] presents two models for calculating the electron relaxation time at temperatures below the plasma regime, with one requiring the melting temperature of the metal and a fitting parameter. This model breaks down above the melting temperature, and is swapped out for

$$\tau = R_0/\bar{v} \tag{2.14}$$

where the ion sphere radius R_0 is given by

$$R_0 = (3/4\pi n_i)^{1/3} \tag{2.15}$$

and the mean electron speed $\bar{v}$ is calculated using the mean thermal velocity

$$\bar{v} = \sqrt{3 k_B T_e / m_e}. \tag{2.16}$$

2.3.3 Chemical potential

The chemical potential can be evaluated using the electron distribution function. The Lee-More model uses the Fermi-Dirac distribution, so the total number of electrons in a small region of the solid N, is given by

$$N = \int_0^\infty \frac{g(\epsilon)d\epsilon}{1+\exp\left(\frac{\epsilon-\mu}{k_B T_e}\right)} \tag{2.17}$$

where $g(\epsilon)$ is the density of states, which takes the form

$$g(\epsilon) = \frac{3N}{2\epsilon_F^{3/2}}\sqrt{\epsilon} \tag{2.18}$$

in the Sommerfield free electron gas model, for electrons of energy ϵ. Here, ϵ_F represents the electron Fermi energy, and is given by, $\epsilon_F = \hbar^2(3\pi^2 Z^* n_i)^{2/3}/2m_e$. By substituting (2.18) into (2.17) and solving the resultant Fermi integral using (2.13), we find

$$N = -\frac{3N}{4}\left(\frac{k_B T_e}{\epsilon_F}\right)^{3/2}\sqrt{\pi}Li_{3/2}\left(-e^{\mu/k_B T_e}\right) \tag{2.19}$$

which rearranges to give an expression for the chemical potential as

$$\mu = k_B T_e \ln\left(-Li_{3/2}^{-1}\left(-\frac{4}{3\sqrt{\pi}}\left(\frac{\epsilon_F}{k_B T_e}\right)^{3/2}\right)\right) \tag{2.20}$$

where $Li_{3/2}^{-1}$ denotes an inverse polylogarithm function of order 3/2.

2.3.4 Coulomb logarithm

The equations used in calculating the Coulomb logarithm are presented in their Section III.B [11], and are summarised in SI units in this section. Lee and More use a Coulomb logarithm of the form

$$\ln\Lambda = \max\left(\frac{1}{2}\ln\left(1+\left(\frac{b_{\max}}{b_{\min}}\right)^2\right), 2\right) \tag{2.21}$$

where $b_{\max}$ and $b_{\min}$ denote the upper and lower cut-offs of the Coulomb interaction respectively.

The upper limit, $b_{\max}$ is taken to be the largest of the ion sphere radius R_0 (2.15) and the Debye-Hückel screening length λ_{DH}, such that

$$b_{\max} = \max(\lambda_{DH}, R_0) \tag{2.22}$$

where

$$\frac{1}{\lambda_{DH}^2} = \frac{Z^* e^2 n_i}{\epsilon_0}\left(\frac{1}{\sqrt{(k_B T_e)^2 + \epsilon_F^2}} + \frac{Z^*}{k_B T_i}\right) \tag{2.23}$$

and T_i is the temperature of the background ions.

The lower limit, $b_{\min}$ is taken to be the largest of the de Broglie wavelength, λ_{dB} and the classical distance of closest approach b_{impact}, where

$$b_{\text{impact}} = \frac{Z^* e^2}{4\pi\epsilon_0 m_e v_e^2} \tag{2.24}$$

$$\lambda_{dB} = \frac{h}{2 m_e v_e} \tag{2.25}$$

and so

$$b_{\min} = \max(b_{\text{impact}}, \lambda_{dB}) \tag{2.26}$$

where we have evaluated the background electron velocity again using the mean thermal speed (2.16).

2.3.5 The A^α factor

The A^α factor is first given in Lee and More's equation (25a [11]), and is of the form

$$A^\alpha\left(\frac{\mu}{k_B T_e}\right) = \frac{4}{3}\frac{F_3(\mu/k_B T_e)}{(1+exp(-\mu/k_B T_e))(F_{1/2}(\mu/k_B T_e))^2} \tag{2.27}$$

where $F_{1/2}(x)$ is defined in (2.13), and $F_3(x)$ is given by

$$F_3(x) = \int_0^\infty \frac{t^3 dt}{1+e^{t-x}}. \tag{2.28}$$

Lee and More calculate that A^α varies between 1 and $32/3\pi$ in the degenerate (low T_e) and non-degenerate (high T_e) limits respectively, as shown in (28a, 30a [11]).

2.4 Ionisation energy loss

As high energy charged particles propagate through solids, they can couple to the electrons in solid atoms and excite them to higher energy levels. This background ionisation acts as an energy loss mechanism for hot electrons, and can be parametrised by the mean excitation energy I_{ex} of the background solid (typically around $11Z$ eV, where Z is the solid atomic number).

The hybrid code outlined in the original Davies paper [9] considers a typical momentum loss rate of the form

$$\frac{dp}{dt} \approx -\frac{Z n_i e^4}{4\pi\epsilon_0^2 m_e v^2}\ln\frac{\epsilon_k}{I_{ex}} \tag{2.29}$$

where v, ϵ_k, and p are the hot electron speed, kinetic energy and momentum respectively. This form is accurate for hot electron kinetic energies between 10 keV and a few MeV, but neglects the density effect correction for higher energy electrons [13].

In the Monte Carlo code `Geant4` [14–16], the nature of ionisation energy loss depends on the energies of the ionised background electrons - which are termed 'δ-rays' to avoid confusion with the incident hot electrons [17]. Hot electrons of kinetic energy ϵ_k can create δ-rays up to a kinetic energy of $\epsilon_k/2$, with low and high energy δ-rays separated by a threshold kinetic energy $\epsilon_{k,cut}$ and treated differently. Low energy δ-rays have a negligible effect on the trajectories of their associated hot electrons, and their effects on the hot electron energy can be treated as a continuous energy loss, while higher energy δ-rays are described by the Møller scatter process. As these δ-rays require a greater transfer of energy from their hot electrons, they must be treated as discrete emissions in order to conserve energy and momentum with the incident hot electron. This approach includes high energy physics which is neglected by the Davies method, and is described more rigorously in the following subsections.

2.4.1 Continuous energy loss

The stopping power $d\epsilon/dx$ acting on incident hot electrons due to the creation of low energy δ-rays is given by the Berger-Seltzer formula [18].

$$\left.\frac{d\epsilon}{dx}\right|_{\epsilon_k^\delta < \epsilon_{k,cut}} = \frac{Zn_i e^4}{8\pi\epsilon_0^2 m_e v^2}\left(\ln\left(\frac{2(\gamma+1)m_e^2c^4}{I_{ex}^2}\right) + F^-(\tau,\tau_{up}) - \delta\right) \tag{2.30}$$

where γ is the Lorentz factor ϵ/m_ec^2, ϵ_k^δ refers to the δ-ray kinetic energy, v is the hot electron speed and δ is a density correction factor. The factor $F^-(\tau,\tau_{up})$ takes the form

$$F^-(\tau,\tau_{up}) = -1 - \left(\frac{v}{c}\right)^2 + \ln\left((\tau-\tau_{up})\tau_{up}\right) + \frac{\tau}{\tau-\tau_{up}} + \frac{1}{\gamma^2}\left(\frac{\tau_{up}^2}{2} + (2\tau+1)\ln\left(1-\frac{\tau_{up}}{\tau}\right)\right) \tag{2.31}$$

with additional parameters

$$\tau = \gamma - 1 \tag{2.32}$$

$$\tau_{up} = \min\left(\frac{\epsilon_{k,cut}}{m_ec^2}, \frac{\tau}{2}\right). \tag{2.33}$$

The `Geant4` Physics Reference Manual provides a method for evaluating the density effect, δ for different types of materials [17]. By defining a parameter, x which is dependent on the incident electron energy

$$x = \frac{\ln(\gamma^2-1)}{4.606} \tag{2.34}$$

we can express the density effect as

$$\delta(x) = \begin{cases} 0 & x < x_0 \\ 4.606x - C + a(x_1 - x)^3 & x_0 < x < x_1 \\ 4.606x - C & x > x_1 \end{cases} \tag{2.35}$$

where the additional parameters have values

$$C = 1 + 2\ln\left(\frac{I_{ex}}{\hbar}\sqrt{\frac{\epsilon_0 m_e}{Z n_i e^2}}\right) \tag{2.36}$$

$$a = 4.606\left(\frac{C}{4.606} - x_0\right)/(x_1 - x_0)^3 \tag{2.37}$$

$$(x_0, x_1) = \begin{cases} (0.2, 2) & I_{ex} < 100eV,\ C \leq 3.681 \\ (0.326C - 1, 2) & I_{ex} < 100eV,\ C > 3.681 \\ (0.2, 3) & I_{ex} \geq 100eV,\ C \leq 5.215 \\ (0.326C - 1.5, 3) & I_{ex} \geq 100eV,\ C > 5.215 \end{cases} \tag{2.38}$$

when considering condensed matter targets like solids.

2.4.2 Møller scatter

While continuous energy loss deals with the creation of low energy δ-rays below the $\epsilon_{k,cut}$ cut-off, Møller scatter describes the discrete process of high energy δ-ray emission. The `Geant4` Physics Reference Manual [17] quotes the cross section of a Møller scatter event per background solid atom, σ_{atom} to be

$$\sigma_{atom} = \frac{e^4}{8\pi m_e \epsilon_0^2}\frac{Z}{v^2\epsilon_k}\left(\frac{(\gamma-1)^2}{\gamma^2}\left(\frac{1}{2} - y\right) + \frac{1}{y} - \frac{1}{1-y} - \frac{2\gamma-1}{\gamma^2}\ln\left(\frac{1-y}{y}\right)\right) \tag{2.39}$$

where the parameter y represents $\epsilon_{k,cut}/\epsilon_k$. Once a discrete Møller scatter event takes place, the δ-ray energy is sampled from the Møller scattering differential cross section using the following algorithm:

1. Generate two uniformly distributed random numbers between 0 and 1 (r_1 and r_2)

2. Draw a ξ using

$$\xi = \frac{\xi_0}{1 - (1 - 2\xi_0)r_1}$$

3. Calculate $g(\xi)$ using

$$g(\xi) = \frac{4}{9\gamma^2 - 10\gamma + 5}\left((\xi(\gamma-1))^2 - (2\gamma^2 + 2\gamma - 1)\frac{\xi}{1-\xi} + \frac{\gamma^2}{(1-\xi)^2}\right)$$

4. Reject or accept δ-ray kinetic energy

$$\epsilon_k^\delta = \begin{cases} \epsilon_k \xi & r_2 \le g(\xi) \\ \text{Go to 1.} & r_2 > g(\xi) \end{cases}$$

where $\xi_0 = \epsilon_{k,cut}/\epsilon_k$.

The direction of the δ-ray momentum, $\mathbf{p}_\delta$ with respect to the incident hot electron momentum $\mathbf{p}_i$ can be expressed in terms of an azimuthal angle ϕ_δ and a polar angle θ_δ, with an equivalent pair of angles, ϕ_e, θ_e describing the momentum of the scattered hot electron $\mathbf{p}_e$. We firstly assume that emission is isotropic in the azimuthal direction, so we randomly sample ϕ_δ and let $\phi_e = \pi + \phi_\delta$. As we have already sampled the δ-ray kinetic energy, we can solve for the polar angles through conservation of energy and momentum

$$p_i = p_e \cos(\theta_e) + p_\delta \cos(\theta_\delta) \tag{2.40}$$

$$p_e \sin(\theta_e) = p_\delta \sin(\theta_\delta) \tag{2.41}$$

$$p_e = \frac{1}{c}\sqrt{((\epsilon_k + m_e c^2) + m_e c^2 - (\epsilon_k^\delta + m_e c^2))^2 - m_e^2 c^4} \tag{2.42}$$

$$p_\delta = \frac{1}{c}\sqrt{(\epsilon_k^\delta + m_e c^2)^2 - m_e^2 c^4} \tag{2.43}$$

which can be rearranged to give

$$\cos(\theta_e) = \frac{p_e^2 - p_\delta^2 + p_i^2}{2 p_i p_e} \tag{2.44}$$

$$\cos(\theta_\delta) = \frac{p_\delta^2 - p_e^2 + p_i^2}{2 p_i p_\delta} \tag{2.45}$$

where we have assumed our δ-ray electron is initially at rest.

2.5 Elastic scatter

The cross section for the Coulomb-driven elastic scatter of hot electrons in solids is huge, making it infeasible to simulate each individual scatter event like we would for bremsstrahlung or Møller scatter. Instead, we solve for the average scattering effect from electrons having traversed a given distance through a solid material, but Davies-style hybrid codes [9, 10] and the Monte Carlo code `Geant4` adopt different methods to calculate this. The equations relevant to these two approaches are given in the following subsections.

2.5.1 Davies algorithm

Davies solves the Fokker-Planck equation in the limit of low Z targets and neglecting large angle scattering, to obtain a mean change per second for the square of the scattering angle

$$\left\langle \Delta\theta^2 \right\rangle \approx \frac{Z^2 n_i e^4}{2\pi\epsilon_0^2} \frac{\gamma m_e}{p^3} \ln \Lambda_s \tag{2.46}$$

where p and γ represent the hot electron momentum and Lorentz factor respectively, and

$$\Lambda_s = \frac{4\epsilon_0 h}{Z^{1/3} m_e e^2} p. \tag{2.47}$$

These results are combined with stochastic differential equations to produce an expected deflection $d\theta$ over a time-step dt of the form

$$d\theta = \Gamma(t) \sqrt{\frac{Z^2 n_i e^4}{2\pi\epsilon_0^2} \frac{\gamma m_e}{p^3} \ln \Lambda_s dt} \tag{2.48}$$

where $\Gamma(t)$ is a time varying random number sampled from a standard normal distribution.

2.5.2 Urban algorithm

There are multiple elastic scatter methods used in `Geant4`, but the default model for electrons is the Urban multiple scattering (MCS) algorithm (although it can be applied to any particle type) [19]. The method is derived from Lewis theory [20], which makes it challenging to adapt to a PIC framework.

The Urban method relies on the distinction between the geometric path length, z (distance between start and end points of a particle over a simulation step), and the true path length, t which takes into account the total distance travelled between start and end points (including scattering deflections in between). The mean geometric path length is related to the true path length through the first transport mean free path λ_1,

$$\langle z \rangle = \lambda_1 \left(1 - e^{-t/\lambda_1}\right) \tag{2.49}$$

where the k-th transport mean free path is given by

$$\frac{1}{\lambda_k} = 2\pi n_i \int_{-1}^{1} (1 - P_k(\cos\chi)) \frac{d\sigma(\chi)}{d\Omega} d(\cos\chi). \tag{2.50}$$

Here, P_k is the k-th Legendre polynomial, and $d\sigma(\chi)/d\Omega$ is the differential cross section of the scattering angle. These transport mean free paths are used to give the mean and variance of the deflection $\cos\theta$

$$\langle \cos\theta \rangle = e^{-t/\lambda_1} \tag{2.51}$$

$$\left\langle \cos^2\theta \right\rangle - \langle \cos\theta \rangle^2 = \frac{1 + 2e^{-2t/\lambda_2}}{3} - e^{-2t/\lambda_1}. \tag{2.52}$$

In `Geant4`, the difference between geometric and true step lengths can give rise to an additional displacement on the particle position at the end-step point, but we will not consider this for our code. The particle step sizes in hybrid PIC codes are restricted to the cell size, which makes the typical hybrid PIC step significantly smaller than the typical `Geant4` step, and so the differences between z and t are less important. This will be demonstrated in the benchmarking of Section 4.2.

This algorithm samples the deflection $u = \cos\theta$ from a distribution $g(u)$ which satisfies the constraints described in (2.51) and (2.52),

$$g(u) = q(pg_1(u) + (1-p)g_2(u)) + (1-q)g_3(u) \tag{2.53}$$

where

$$g_1(u) = C_1 e^{-a(1-u)} \qquad u_0 \leq u \tag{2.54}$$

$$g_2(u) = C_2 \frac{1}{(b-u)^d} \qquad u \leq u_0 \tag{2.55}$$

$$g_3(u) = C_3 \tag{2.56}$$

A detailed description of the sampling method and parameters used is given in the original Urban report [19], and will not be reproduced here. Instead, we will present the Urban algorithm in its most basic form, without the lateral displacement correction, and assuming negligible kinetic energy loss over the small step.

The first transport mean-free-paths for electrons and positrons, λ_1 are tabulated by Liljequist *et al* [21] for kinetic energies between 100 eV and 20 MeV, in materials with atomic numbers ranging from 4 to 82. The Urban algorithm uses this data up to 10 MeV kinetic energy, and then switches to the high energy data presented by Mayol and Salvat [22].

When initialising the Urban algorithm, parameters based on the atomic number, Z of the target are pre-calculated. In the case of compound targets, an average Z value weighted by the number densities of the constituent elements is used. The parameters used in calculating the width of the angular distribution, θ_0 are

$$Z_{fac} = 0.990395 - 0.168386Z^{1/6} + 0.093286Z^{1/3} \tag{2.57}$$

$$\theta_1 = Z_{fac}\left(1 - \frac{0.08778}{Z}\right) \tag{2.58}$$

$$\theta_2 = Z_{fac}(0.04078 + 1.7315 \times 10^{-4}Z) \tag{2.59}$$

and parameters used in calculating the tail of the distribution are of the form

$$d_1 = 2.3785 - 0.41981Z^{1/3} + 0.0631Z^{2/3} \tag{2.60}$$

$$d_2 = 0.47526 + 1.7694Z^{1/3} - 0.33885Z^{2/3} \tag{2.61}$$

$$d_3 = 0.23683 - 1.8111Z^{1/3} + 0.32774Z^{2/3} \tag{2.62}$$

$$d_4 = 0.017888 + 0.019659Z^{1/3} - 0.002664Z^{2/3} \tag{2.63}$$

which come from an empirical fit to muon scattering experiments [23]. The θ_0 variable is taken to be

$$\theta_0 = \frac{13.6\text{MeV}}{v_e p_e}|q_e/e| \begin{cases} \sqrt{\frac{t}{X_0}}\left(\theta_1 + \theta_2 \ln\left(\frac{t}{X_0}\right)\right) & t \geq t_{small} \\ \sqrt{\frac{t_{small}}{X_0}}\left(\theta_1 + \theta_2 \ln\left(\frac{t_{small}}{X_0}\right)\right)\sqrt{\frac{t}{t_{small}}} & t < t_{small} \end{cases} \tag{2.64}$$

where q_e, v_e and p_e are the charge, velocity and momentum of the incident particle respectively, and X_0 is the radiation length of the material. A limiting form of the equation is used once the true path length drops below a critical value t_{small}, where

$$t_{small} = \min\left(\max\left[10^{-5}\lambda_1/\left(\frac{\epsilon_k}{1\text{ MeV}}\left(10+\frac{\epsilon_k}{1\text{ MeV}}\right)\right), 0.01\text{ nm}\right], 1.0\text{ mm}\right) \tag{2.65}$$

and we have expressed λ_1 in metres.

Using $\tau = t/\lambda_1$, we evaluate the following expectation angles,

$$\langle\cos\theta\rangle = \begin{cases} 1-\tau(1-\frac{1}{2}\tau) & \tau < 0.01 \\ e^{-\tau} & \text{otherwise} \end{cases} \tag{2.66}$$

$$\langle\cos^2\theta\rangle = \begin{cases} 1-\frac{1}{3}\tau(5-6.25\tau) & \tau < 0.01 \\ \frac{1}{3}(1+2e^{-2.5\tau}) & \text{otherwise} \end{cases} \tag{2.67}$$

then we calculate the additional parameters

$$u = \max\left(\tau^{1/6}, \left(\frac{0.1\text{nm}}{\lambda_1}\right)^{1/6}\right) \tag{2.68}$$

$$x = \begin{cases} \theta_0^2\left(1-\frac{1}{12}\theta_0^2\right) & \theta_0 < 0.1 \\ 4\sin^2\left(\frac{1}{2}\theta_0\right) & \theta_0 \geq 0.1 \end{cases} \tag{2.69}$$

$$\xi = \max\left(d_1 + d_2 u + d_3 u^2 + d_4 \ln\left(\frac{\lambda_1}{X_0}\right), 1.9\right) \tag{2.70}$$

$$C = \begin{cases} 3.001 & |3-\xi| < 0.001 \\ 2.001 & |2-\xi| < 0.001 \\ \xi & \text{otherwise} \end{cases} \tag{2.71}$$

where the conditions in (2.71) are chosen to prevent dividing by zero. These are used to obtain the next set of sampling parameters

$$d = \left(\frac{Cx}{2+(C-\xi)x}\right)^{C-1} \tag{2.72}$$

$$\langle x_1\rangle = 1 + ((1+\xi)e^{-\xi}-1)\frac{x}{1-e^{-\xi}} \tag{2.73}$$

$$\langle x_2\rangle = \frac{1-\xi x + d - \frac{Cx-(2+(C-\xi)x)d}{C-2}}{1-d} \tag{2.74}$$

$$p = \frac{\frac{C-1}{C(1-d)}}{\frac{e^{-\xi}}{1-e^{-\xi}} + \frac{C-1}{C(1-d)}} \tag{2.75}$$

$$q = \frac{\langle\cos\theta\rangle}{p\langle x_1\rangle + (1-p)\langle x_2\rangle} \tag{2.76}$$

Once these have been calculated, we draw up to four uniformly distributed random numbers between 0 and 1: r_1, r_2, r_3, r_4, and use the first to define two final sampling parameters,

$$v_1 = (1-d)r_1 \tag{2.77}$$

$$v_2 = \frac{1-d}{d(C-1)} r_1 \tag{2.78}$$

and finally, we draw a deflection $\cos\theta$ value

$$\cos\theta = \begin{cases} 1 & \theta_0^2 < 10^{-16} \\ -1+2r_2 & \tau \geq 8 \\ S(\langle\cos\theta\rangle, \langle\cos^2\theta\rangle) & \theta_0 > \frac{\pi}{6}, \text{ or } \langle x_1\rangle \leq 0.99\langle\cos\theta\rangle \end{cases} \tag{2.79}$$

otherwise

$$\cos\theta = \begin{cases} 1+\ln\left(e^{-\xi}+r_4\left(1-e^{-\xi}\right)\right)x & r_2 \leq q, \text{ and } r_3 \leq p \\ -1+v_2(1-\frac{1}{2}v_2C)(2+(C-\xi)x) & r_2 \leq q, \text{ and } r_3 > p, \text{ and } v_1 \leq 0.01d \\ 1+x\left(C-\xi-C(v_1+d)^{-1/(C-1)}\right) & r_2 \leq q, \text{ and } r_3 > p, \text{ and } v_1 > 0.01d \\ -1+2r_3 & r_2 > q \end{cases} \tag{2.80}$$

The function $S(\langle\cos\theta\rangle, \langle\cos^2\theta\rangle)$ is a simple scatter function, which is evaluated using

$$a = \frac{2\langle\cos\theta\rangle + 9\langle\cos^2\theta\rangle - 3}{2\langle\cos\theta\rangle - 3\langle\cos^2\theta\rangle + 1} \tag{2.81}$$

$$p = \frac{(a-2)\langle\cos\theta\rangle}{a} \tag{2.82}$$

$$\cos\theta = \begin{cases} -1+2r_2^{a+1} & r_3 < p \\ -1+2r_2 & \text{otherwise} \end{cases} \tag{2.83}$$

The Urban algorithm draws the azimuthal deflection ϕ isotropically.

2.6 Background heating

When energy is deposited in the solid from collisional energy loss or in power dissipation from Ohmic heating, the energy is converted to an increase in the background electron temperature. Calculating the electron temperature rise, dT_e given an energy increase, $d\epsilon$ is done using the heat capacity of the material. For this, Davies [10] cites a value from an unpublished report by Bell [24],

$$C = \frac{d\epsilon}{k_B dT_e} = 0.3 + 0.8\left(\text{DUT3} + 1.5T'\frac{2.2+T'}{(1.1+T')^2}\right) \tag{2.84}$$

$$\text{DUT3} = \frac{(10^{-18}V'^{-0.75} + \ln(10^{22}V'))(2.906 \times 10^8 T'^{-2.0733333} - 7.223 \times 10^6 T'^{-0.853333})}{(3.283 \times 10^7 T'^{0.146666} + 1.805 \times 10^8 T'^{-1.073333})^{2.5}} \tag{2.85}$$

which corresponds to the temperature rise of a single electron having received energy $d\epsilon$. These equations use reduced variables, where $T' = k_B T_e Z^{-4/3}/e$, and $V' = 10^6 ZV$, where V is the volume occupied by a single atom. The DUT3 term has not been included in the Davies paper, but this may be due to its negligible contribution in our systems of interest. For an Al target at about 100 eV (1.2×10^6 K) electron temperature, $V' \approx 5 \times 10^{-23}$, and $T' \approx 3.3$ returns DUT3 $\approx -2 \times 10^{-13}$, and as the second term in the brackets of (2.84) works out to about 1.4, the DUT3 term can be ignored. This yields the Davies equation

$$C = 0.3 + 1.2T' \frac{2.2 + T'}{(1.1 + T')^2}. \tag{2.86}$$

The remainder of this subsection looks at how this heat capacity can be used to deduce the electron temperature increase due to Ohmic heating and collisions, and also how this energy can be transferred to the background ions through collisions.

2.6.1 Ohmic heating

The power dissipated, P due to a current I moving through a resistor R is $P = I^2 R$, and using Ohm's law relating current and resistance to potential difference $V = IR$, we may rewrite the power dissipation in the form $P = V^2/R$. Davies [10] defines power density P_ρ as

$$P_\rho = \frac{\mathbf{E} \cdot \mathbf{E}}{\eta} \tag{2.87}$$

where $\mathbf{E}$ is the electric field, and η the resistivity in the solid. However, due to the grid stagger of `EPOCH`, the value of $\mathbf{E}$ depends on neighbouring values of η, which can lead to discontinuities on material boundaries. Instead, we use the equivalent form

$$P_\rho = \eta \mathbf{j} \cdot \mathbf{j} \tag{2.88}$$

using the hot electron current density, $\mathbf{j}$. Over a timestep dt, in a cell of volume V_{cell}, the energy transferred to the electrons is $P_\rho V_{\text{cell}} dt$. If the energy is split evenly between the electrons, each one will receive an energy V_{e^-}/V_{cell}, where the volume taken up by an electron V_e^- is approximately $1/Zn_i$. Hence, the temperature gain due to Ohmic heating can be written

$$\Delta T_e = \frac{\mathbf{j} \cdot \mathbf{j} \eta dt}{Z n_i C k_B} \tag{2.89}$$

2.6.2 Collisional heating

When hot electrons undergo ionisation energy loss, their energy is transferred to either secondary hot electrons (δ-rays), or to exciting lower energy background electrons which raises the background electron temperature. As in Section 2.6.1, the energy deposited by the hot electron is shared between all the background electrons in the cell, and so each background electron receives the fraction $1/Zn_iV_{\text{cell}}$ of the energy loss from each hot electron, $\Delta\epsilon_h$. Hence, the background electron temperature gain due to collisional energy loss is

$$\Delta T_e = \frac{\Sigma_h \Delta\epsilon_h}{Zn_iV_{\text{cell}}Ck_B} \tag{2.90}$$

where we sum the energy losses of each hot electron, h in the cell.

2.6.3 Thermal equilibration of ions and electrons

Sections 2.6.1 and 2.6.2 have explained how the electron temperature increases over the course of the simulation, but the resistivity also depends on the ion temperature, as seen in (2.23). The background electrons can transfer thermal energy to background ions through collisions, as described by Spitzer [25] in their equations (5-30 [25]) and (5-31 [25]). In SI units, these read

$$\frac{dT_1}{dt} = \frac{T_2 - T_1}{t_{eq}} \tag{2.91}$$

$$\frac{1}{t_{eq}} = \frac{2}{3(2\pi k_B)^{3/2}} \frac{q_1^2 q_2^2 n_2 \sqrt{m_1 m_2}\ln\Lambda}{\epsilon_0^2 (T_1 m_2 + T_2 m_1)^{3/2}} \tag{2.92}$$

where we consider the rate of temperature change in species 1 through collisions with species 2, and the charge, mass, temperature and number density of species k are written as q_k, m_k, T_k and n_k respectively. Thus, if the time between steps n and $n+1$ is Δt, the temperature of species k at step $n+1$, T_k^{n+1} can be written

$$T_e^{n+1} = T_e^n + \Delta t \frac{2}{3(2\pi k_B)^{3/2}} \frac{(Z^*)^2 e^4 n_i \sqrt{m_e m_i}\ln\Lambda}{\epsilon_0^2 (T_e^n m_i + T_i^n m_e)^{3/2}} (T_i^n - T_e^n) \tag{2.93}$$

$$T_i^{n+1} = T_e^n + \Delta t \frac{2}{3(2\pi k_B)^{3/2}} \frac{(Z^*)^3 e^4 n_i \sqrt{m_e m_i}\ln\Lambda}{\epsilon_0^2 (T_e^n m_i + T_i^n m_e)^{3/2}} (T_e^n - T_i^n) \tag{2.94}$$

where we have written the electron number density as Z^*n_i, and the Coulomb logarithm $\ln\Lambda$ is evaluated using the same method as in Section 2.3.4 for the reduced Lee-More model.

2.7 K-alpha emission

As hot electrons traverse through solid density targets, they may trigger the production of Kα photons. In Kα emission, a hot electron ionises a bound electron from the 1s shell of a target atom (principle quantum number $n = 1$), and a photon is produced as bound electrons from higher energy shells de-excite to fill the vacancy [26].

The electron-impact ionisation cross section of a bound shell can be described by the relativistic binary-encounter Bethe (RBEB) model derived by Kim *et al* [27]. Through combining the general RBEB cross section (22 [27]) with an inner shell correction factor (37 [27]), the resultant cross section may be written as

$$a_{nl} = \frac{1}{2}\left(1 + \frac{\beta_t^2 + \beta_u^2 + \beta_b^2}{\beta_t^2}\right) \tag{2.95}$$

$$b_{nl} = \frac{2\pi a_B^2 \alpha^4}{(\beta_t^2 + \beta_u^2 + \beta_b^2)b'} \tag{2.96}$$

$$c_{nl} = \frac{1}{2}\left(\ln\left(\frac{\beta_t^2}{1-\beta_t^2}\right) - \beta_t^2 - \ln(2b')\right) \tag{2.97}$$

$$d_{nl} = 1 - \frac{1}{t} - \frac{\ln(t)}{t+1}\frac{1+2t'}{(1+0.5t')^2} + \frac{b'^2(t-1)}{2(1+0.5t')^2} \tag{2.98}$$

$$\sigma_{\mathrm{rbeb},nl} = a_{nl}b_{nl}\left(c_{nl}\left(1 - \frac{1}{t^2}\right) + d_{nl}\right) \tag{2.99}$$

where the β terms represent v/c for electrons with different speeds, v. Specifically, β_t, β_u and β_b are calculated for electrons with kinetic energy ϵ_k, U_{nl}, and B_{nl} respectively, where U_{nl} is the average kinetic energy of bound electrons in the orbital nl, and B_{nl} is the corresponding binding energy. The ϵ_k energy refers to the kinetic energy of the incident hot electron which triggers the ionisation, and t is the ratio ϵ_k/B_{nl}. The primed terms t' and b' denote dimensionless numbers ϵ_k/m_ec^2 and B_{nl}/m_ec^2 respectively, α is the fine structure constant and a_B is the Bohr radius. In most cases, U_{nl} is not known, although it is common to use the assumption that $U_{nl} = B_{nl}$ [28]. The binding energies of bound electrons in the 1s shell can be found in the tables of Desclaux [29] for atoms, and the ion equivalents may be deduced using a method presented by Carlson *et al* [28, 30].

Once a 1s vacancy has been produced through collisional ionisation, this may be resolved through either radiative or non-radiative transitions. In the case of Auger electrons [31], the vacancy is filled without emission of a photon, and so not all ionisation events lead to Kα production.

The Kα energy spectrum is composed of two line emissions at different energies, Kα_1 and Kα_2, with some line-width applied to each. In a given Kα emission, the probability of creating a photon with the Kα_1 energy is 2/3, and 1/3 for Kα_2. In general, the central photon energy and line-broadening of the Kα_1 and Kα_2 lines change with the background temperature. The energies for these two lines, $E_{k\alpha1}$ and $E_{k\alpha2}$ in the case of copper have been provided by Suxing at the University of Rochester, and take the form

$$\begin{aligned}E_{k\alpha 1} =&(8047.8 + 9.304 \times 10^{-3} - 0.2596 T'_e + 8.725 \times 10^{-3} T'^2_e \\ &- 8.444 \times 10^{-5} T'^3_e + 3.908 \times 10^{-7} T'^4_e - 6.785 \times 10^{-10} T'^5_e) e \qquad (2.100)\\ E_{k\alpha 2} =&(8027.3 - 8.486 \times 10^{-3} - 0.2648 T'_e + 8.837 \times 10^{-3} T'^2_e \\ &- 8.336 \times 10^{-5} T'^3_e + 3.684 \times 10^{-7} T'^4_e - 6.042 \times 10^{-10} T'^5_e) e \qquad (2.101)\end{aligned}$$

where energies have been given in SI units, but T'_e is the background electron temperature in eV. At present, copper is the only solid which can produce Kα photons in the code.

2.8 Photo-electric effect

As photons traverse a medium, they may undergo photo-electric attenuation. This causes photons to be absorbed, and so the number of photons escaping the solid may be less than the number originally created. In general, the rate of absorption depends on the absorbing material properties, and the photon energy.

For the specific case of Kα photons in solid-density heated copper targets, Suxing at the University of Rochester has provided us with a fit to describe the attenuation rate. According to this model, the number of photons N_γ remaining after traversing a distance x is given by

$$N_\gamma = N_{\gamma,0} e^{-\kappa x} \qquad (2.102)$$

where $N_{\gamma,0}$ is the original number of photons, and κ is an attenuation coefficient which scales with the background electron temperature in eV, T'_e. For Kα photons in copper, κ takes the form

$$\kappa = a_0 + a_1 T'_e + a_2 \left(T'_e\right)^2 + a_3 \left(T'_e\right)^3 + a_4 \left(T'_e\right)^4 \qquad (2.103)$$

where the fit parameters a_n are given in Table 2.

a_n	$T'_e < 500$	$T'_e > 500$
a_0	154.633275098830	475.299874660280
a_1	0.684356560314504	-0.781298994640132
a_2	-1.129249310157342$\times 10^{-3}$	4.530821133169535$\times 10^{-4}$
a_3	-9.808822131732512$\times 10^{-7}$	-8.910063459566747$\times 10^{-8}$
a_4	1.505648924357310$\times 10^{-9}$	0.0

Table 2: Parameter fits for the κ absorption co-efficient in the photo-electric attenuation of Kα photons in a heated copper target, from (2.103). When using these a_n with T'_e (background copper electron temperature in eV), equation (2.103) gives κ in units of cm^{-1}.

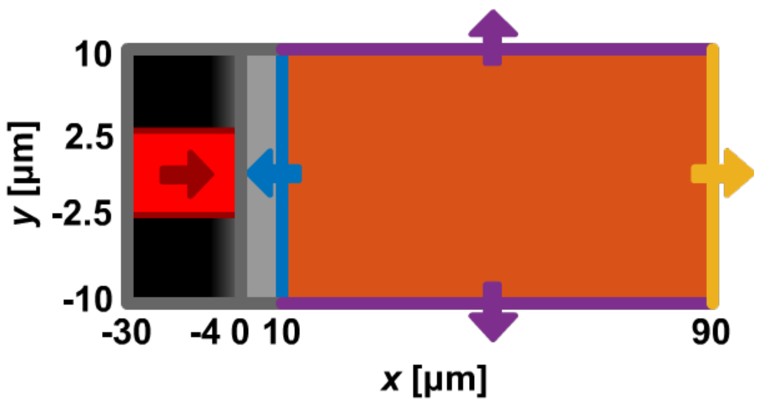


Figure 1: Schematic diagram to show how the end states of particles are characterised, and the path of the laser pulse. We monitor electrons starting in an exponential pre-plasma with 2 μm scale-length between $x = -4$ μm and $x = 0$ μm. Once these electrons pass the rear surface of the solid ($x = 10$ μm), they can **reflux** back into the solid, remain **outside** the solid but within the simulation window, or escape in the **longitudinal** or **transverse** directions.

2.9 Target normal sheath acceleration

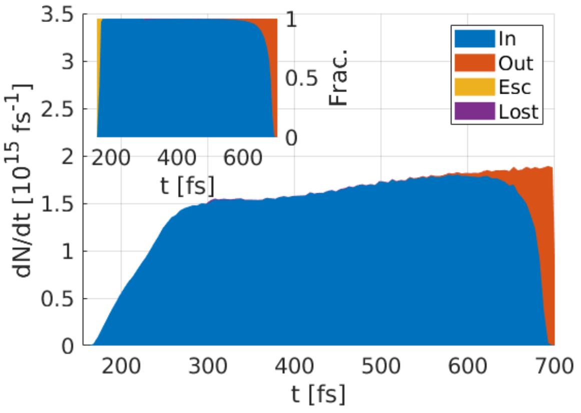
As hot electrons pass from the solid out into the vacuum, they establish negatively charged sheath fields on the solid surface which reflect electrons back into the solid, and accelerate ions out in target normal sheath acceleration (TNSA). Recent studies [3,32] have found that hot electrons lose energy when refluxing in the sheath field, providing an additional energy-loss mechanism in these systems. We have repeated similar simulations to Rusby *et al* [32] in order to roughly characterise the electron behaviour when refluxing in sheath fields.

Our simulations study 10 μm fully ionised carbon targets, and 2 μm gold targets with an ionisation state of 51+. Both targets included a 4 μm exponential pre-plasma on the laser-facing side, with a 2 μm exponential scale length. The targets were simulated with shots from 10^{20} Wcm^{-2} and 10^{22} Wcm^{-2} laser pulses, creating 4 simulations in total.

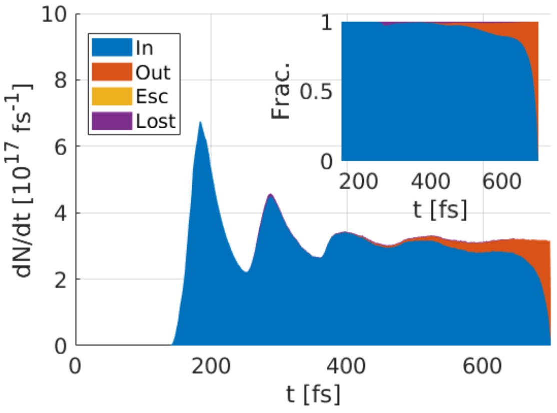
In the C target simulations, square cells of side 20 nm were used, with 300 macroparticles per cell (1 in 6 were macro-ions, the rest macro-electrons). The simulation domain ranged -30 μm to 90 μm in x, and the target and simulation window spanned -10 μm to 10 μm in y. The laser pulse had a Gaussian profile with a 40 fs full width half maximum, and had a focal spot size of 5 μm, as shown schematically in Figure 1.

It was found that Au^{51+} targets required a greater resolution to suppress self-heating, and so these simulations ran with 5 nm cells and 150 ppc, for 160 fs in a smaller simulation window. This reduced window spanned -10 μm to 10 μm in x (pre-plasma $x = -4$ μm to 0, solid $x = 0$ to 2 μm), and from -4 μm to 4 μm in y. The temporal and spatial profiles of the laser pulse remained the same as the C runs.

In our simulations, we used enhanced particle probes which output particle momentum, ID and the time the particle passes the probe. By placing probes at $x = \{-29, 0, 10, 89\}$ μm



(a) C 10^{20}Wcm^{-2}



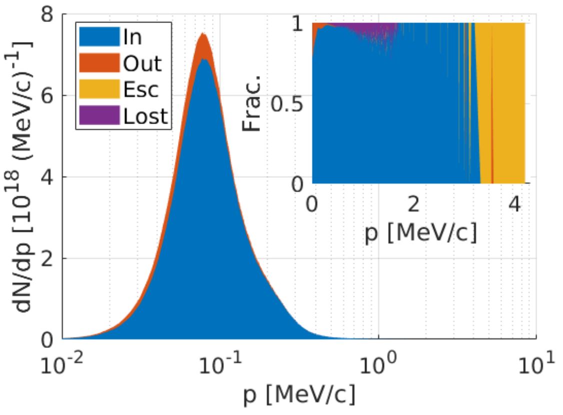
(b) C 10^{22}Wcm^{-2}

Figure 2: The number spectra of electrons leaving the solid through the rear surface for the two carbon simulations, binned by outgoing time. The plots are coloured to show the proportion of particles in each of the four end-states. A normalised fraction of end-states is provided in the sub-figures to show areas which are under-represented in dN/dt.

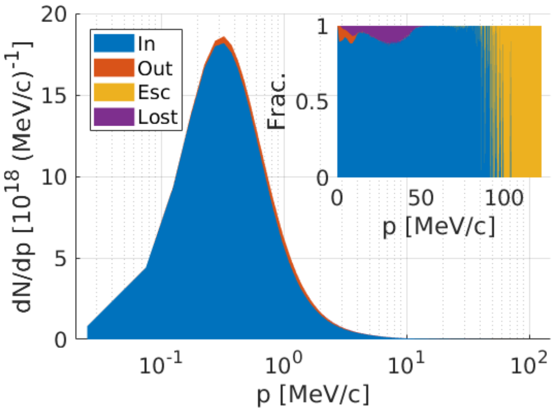
in C simulations, we could categorise electrons leaving the solid by their four possible end-states, also summarised in Figure 1. Electrons could be reflected by the sheath field and reflux back into the target, or could possess enough energy to overcome the sheath field and escape through the $x_{\max}$ boundary. Electrons could also escape through a y-boundary, but this doesn't tell us if they would escape the solid or if they would eventually reflux, so we consider these electrons lost. Finally, electrons could be absorbed into the sheath field, and end the simulation outside the solid region but still within the simulation window. In these characterisation studies, we only track the evolution of electrons which initially started in the pre-plasma ($x < 0$ μm), and only after passing the rear target surface for the first time. This is because the electrons may trigger the front probe multiple times on their way into the target, but they won't act like hot electrons until they possess the energy to fully traverse the solid.

The end-state of electrons varied over the run-time of the simulation. In Figure 2, we bin electrons by the time they leave solid through the 10 μm probe in the C target runs, plotting the number spectrum of outgoing probe hits and colouring this spectrum according to the end-state of particles in each bin. Electrons can be counted more than once if they leave the target again after re-entry.

Figure 2 shows that most electrons reflux back into the target, and we only see particles escaping through $x_{\max}$ early in the simulation before a sheath field is established (as seen in the Figure 2a insert for low t). While some electrons end the simulation in the background region, this mostly occurs for electrons exiting towards the end of the simulation, suggesting they simply haven't had time to reflux. The lack of lost electrons suggests the transverse simulation window size is sufficient for capturing most electron end-states.



(a) C 10^{20}Wcm^{-2}



(b) C 10^{22}Wcm^{-2}

Figure 3: The number spectra of electrons leaving the solid through the rear surface for the two carbon simulations, binned by outgoing total momentum. The plots are coloured to show the proportion of particles in each of the four end-states. A normalised fraction of end-states is provided in the sub-figures to show areas which are under-represented in dN/dp.

We also inspected the end states of particles after binning by outgoing total momentum, as shown in Figure 3. Again, in both C simulations we find the majority of electrons reflux back into the target, except for the highest energy electrons which escape. We find that it's mostly the lower energy electrons which end the simulation outside the solid, which suggests the higher energy electrons don't spend much time in reflux events.

Figures 2 and 3 confirm that electrons mostly reflux back into the target when escaping through the rear solid boundary, but we can also look at how electron energy changes when refluxing. We binned refluxing electrons by the outgoing x-component of momentum on the rear boundary, p_x^{out} in all four simulations, and calculated the average value of $|p_x^{\text{in}}/p_x^{\text{out}}|$ in each bin. These curves, along with a shaded area to show the variation of the momentum change in each bin, have been shown in Figure 4.

We find the momentum loss behaviour is quite similar for targets of different thickness and density, and is qualitatively similar at different laser intensities. The highest energy electrons seem to lose the most energy when refluxing, with the lower energy electrons staying at similar p_x magnitude when returning. However, as seen in Figure 3, the majority of probe-hits occur for these lower energy electrons. It is useful to also look at the total momentum loss in each bin, instead of the average loss per electron.

After binning all refluxing electrons by outgoing momentum, Figure 5 shows total change of momentum component summed over every electron in each bin. Figure 5a shows that most of the longitudinal momentum is lost by the lower energy electrons. A loss of longitudinal momentum agrees with observations by Vyskočil *et al* [3], who suggest the loss would explain the increase in bremsstrahlung emission angle. Our results go further, and suggest that an increased angle can be attributed to both a decrease in p_x, and an increase in p_y when

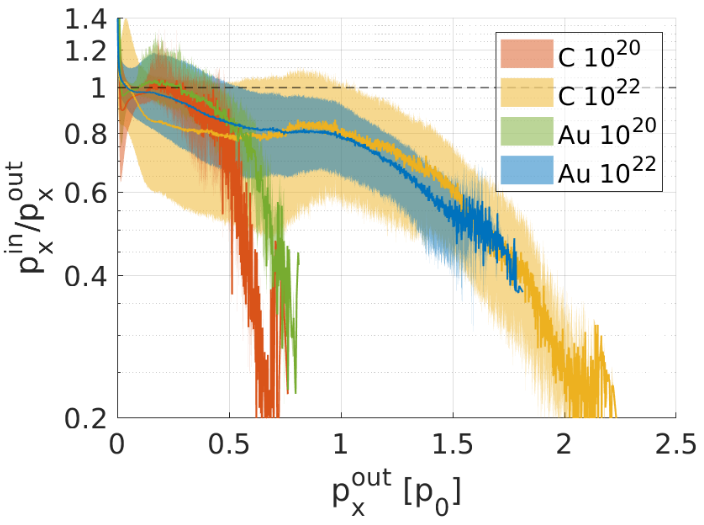
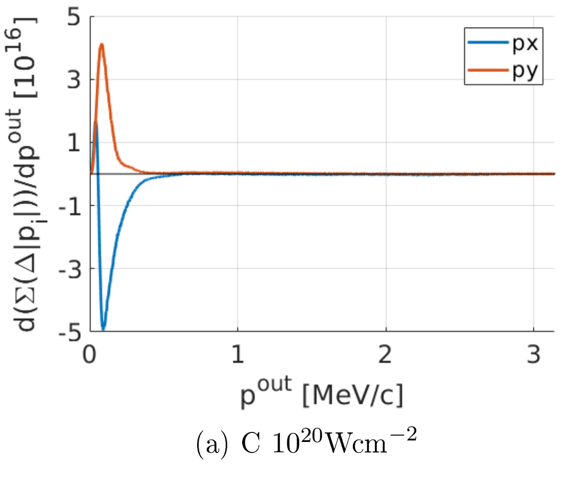
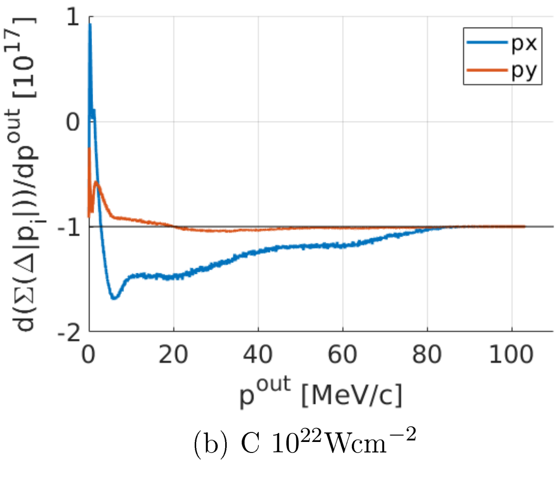


Figure 4: The longitudinal momentum loss in a refluxing event on the rear solid surface, binned by outgoing hot electron momentum. The solid line shows the average momentum change in a bin, and the shaded regions represent the average deviation both above and below the solid line. The dotted line represents no change in the momentum magnitude - everything below has lost momentum, everything above has gained it. The outgoing momentum is in units of the ponderomotive momentum $p_0 = a_0 m_e c$, and simulations are labelled by the target material, and the laser intensity in Wcm^{-2}.

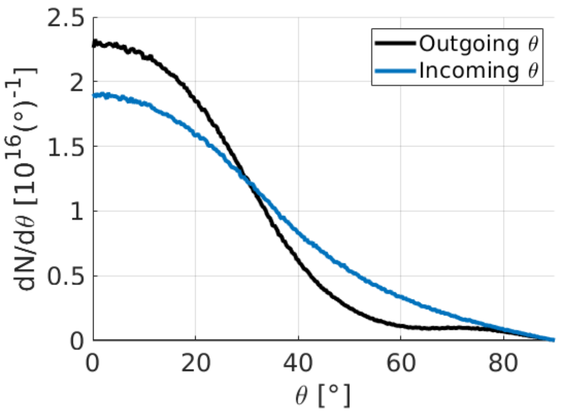


(a) C $10^{20}\mathrm{Wcm}^{-2}$

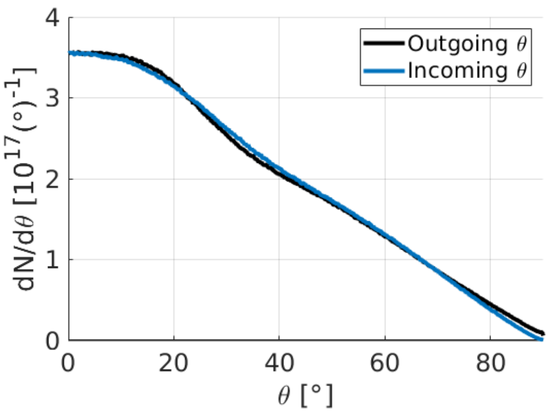


(b) C $10^{22}\mathrm{Wcm}^{-2}$

Figure 5: Electrons which pass the rear surface of the solid and then re-enter are binned by their outgoing total momentum. The changes in the momentum component magnitude ($|p_i^{\text{in}}| - |p_i^{\text{out}}|$) are summed over all electrons in the bin, and plotted for p_x and p_y. No p_z change has been plotted, as this remained 0 for all particles in our 2D simulations.



(a) C 10^{20}Wcm^{-2}



(b) C 10^{22}Wcm^{-2}

Figure 6: Angular distributions of refluxing electrons when escaping the solid target (outgoing) and returning to the solid (incoming).

refluxing. This relationship is less clear in the higher intensity Figure 5b, but we still observe a dominant loss in longitudinal momentum and a gain in transverse momentum.

We also note that the gain in p_y is less than the loss in p_x, and so on average, electrons lose energy when refluxing. The average loss of total momentum during a reflux event has been calculated for each boundary in each simulation, and is given in Table 3.

Target	$\langle\Delta p\rangle$ [keV/c]		
	Front	Rear	Total
10^{20} Wcm^{-2}, C	-28.2766	-3.82309	-7.46456
10^{20} Wcm^{-2}, Au	-112.953	-3.04252	-6.95298
10^{22} Wcm^{-2}, C	-159.474	-196.172	-180.774
10^{22} Wcm^{-2}, Au	-139.194	-141.853	-141.149

Table 3: Mean total momentum changes for all refluxing electrons, broken down into front reflux events (laser and pre-plasma side), rear reflux events (initially solid density to vacuum interface), and all reflux events combined.

As the momentum components are changing in different ways, the angular spectrum of electrons leaving the solid will be different to the spectrum of electrons coming back in. These spectra are compared for rear boundary refluxing electrons in Figure 6 for the two C simulations. In Figure 6a we see the angular spectra tending to a more uniform distribution, while we see little change in the total spectrum in the higher intensity Figure 6b.

We can characterise the angular distributions for refluxing electrons further. In Figure 7, we bin refluxing electrons by their outgoing angle, and compare it to their incoming angle. On average, we find that electrons with low θ typically come back in with a larger angle to the laser axis, and electrons leaving with high θ come back lower.

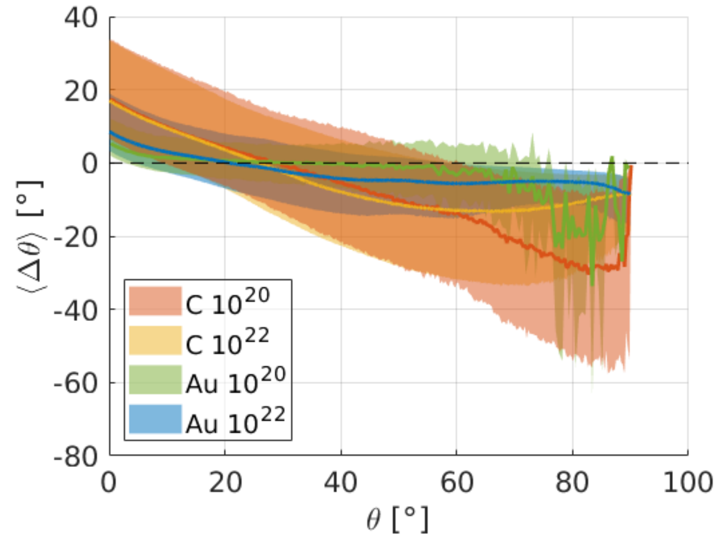


Figure 7: Refluxing electrons were binned by their outgoing angle with respect to the laser axis θ when leaving the solid on either the front or rear surfaces. The mean angle change upon returning to the solid has been calcuated in each bin, and is shown as the solid line for all 4 simulations. The upper and lower shaded regions span up to the average deviation $|\Delta\theta - \langle\Delta\theta\rangle|$ for all electrons above and below $\langle\Delta\theta\rangle$ respectively.

Despite little change in the Figure 6b data, we can see from Figure 7 that there is still some scatter when refluxing. As the shaded area of uncertainty remains a similar size for all outgoing θ, this could provide a useful measure in characterising the amount of scatter in a reflux event. The average values of the bin uncertainty range $\sigma_{\langle\Delta\theta\rangle}$, weighted by the number of electrons in each bin, have been provided in Table 4.

Target	$\sigma_{\langle\Delta\theta\rangle}$ [°]		
	Front	Rear	Total
10^{20} Wcm^{-2}, C	27.609	25.398	27.2838
10^{20} Wcm^{-2}, Au	10.2631	14.8376	10.4962
10^{22} Wcm^{-2}, C	27.3305	34.8999	31.7729
10^{22} Wcm^{-2}, Au	15.0984	30.796	21.7208

Table 4: Mean widths of the shaded error region in Figure 7. This mean-width is calculated using an average weighted by the number of electrons in each bin.

3 Code implementation

Section 2 provides the equations which are used to create the hybrid routines, with references to original sources. This section explores how these equations have been implemented into `EPOCH`, discussing the new hybrid directory in the source code, and the changes made to pre-existing code files.

The hybrid mode consists of two main changes to `EPOCH`, the first of which being the hybrid-PIC loop. When running in hybrid mode, instead of changing the main PIC-loop in epoch$n$d.F90, we pass control to a second PIC-loop found in hybrid/hybrid.F90. This routine is responsible for calling the hybrid subroutines.

Hybrid geometries are specified in the code using a new solid data-type, which stores all the relevant solid parameters. Multiple solids can be defined in the input deck, and compound targets like plastic can be made by spatially overlapping carbon and hydrogen solids.

The following files have been added to `EPOCH`, and house the hybrid routines:

- hybrid/hybrid.F90 - hybrid PIC loop, initialise, finalise, cold e^- removal
- hybrid/hy_elastic_davies.F90 - Davies-style elastic scatter
- hybrid/hy_elastic_urban.F90 - Urban-style elastic scatter
- hybrid/hy_fields.F90 - Hybrid field solver
- hybrid/hy_heating.F90 - Ohmic and collision heating, ion temperature
- hybrid/hy_ionisation_loss.F90 - Ionisation loss and Møller scatter
- hybrid/hy_laser.F90 - Laser-based electron injectors
- hybrid/hy_resistivity.F90 - All resistivity models
- hybrid/hy_shared.F90 - Interpolation routines
- deck/deck_hy_laser_block.F90 - Input for hybrid electron injectors
- deck/deck_hybrid_block.F90 - Switch on/off specific hybrid routines
- deck/deck_solid_block.F90 - Define background solids

Electron Møller scatter requires an additional particle variable (`optical_depth_delta`), and so to prevent slow-down of MPI routines, the hybrid modules can only be accessed through the -DHYBRID preprocessor flag. Sections 3.1-3.10 discuss the core hybrid routines in more detail.

Additional functionality has been added to EPOCH to allow for characterisation of TNSA boundaries, and to allow injection of hot electrons from the post-processing of PIC codes. Particle probes have been updated to include the ID of particles which trigger them, and also the time a particle passes the probe. The injector block now has the option to inject from file, where momenta, weights, ID, injection time and injection position (for 2D and 3D) are read from different files. These improvements are discussed in sections 3.11 and 3.12.

3.1 Hybrid PIC loop

The PIC loop in the main epoch*nd*.F90 file has been positioned inside an IF-statement, which can only be accessed when `use_hybrid` is false. Otherwise, we call the `run_hybrid_PIC` subroutine, which has a similar structure to the standard PIC loop but with a few extra routines. This is shown graphically in the Figure 8 flow-chart.

As in the normal PIC loop, the field update is split into two halves, such that we have time-centred fields for leap-frogging the particle push. This allows us to use the same particle pusher as in the traditional PIC code, with additional momentum-changing scripts like elastic scatter and ionisation loss occurring separately.

The loop starts and finishes with `time` evaluated half a timestep ahead of the particles and fields, and we output with fields evaluated half a timestep behind the particles for consistency with the normal PIC loop.

3.2 Solid type

The hybrid routines describe how electrons pass through solids, which are represented in the code as a new solid type. These are defined in shared_data.F90, which in EPOCH3D reads

```
TYPE solid

  ! Input variables
  REAL(num) :: Iex = -1.0_num
  REAL(num) :: rad_len = -1.0_num
  REAL(num) :: mass_no = -1.0_num
  INTEGER :: Z = -1
  INTEGER :: res_model = 1
  REAL(num), ALLOCATABLE :: ion_density(:,:,:), el_density(:,:,:)

  ! Derived variables
  REAL(num) :: theta_fac, ln_s, Z_prime
  REAL(num) :: Iex_term, dEdx_C
  REAL(num) :: urb_sig, urb_high, urb_el(22), urb_pos(22)
  REAL(num), ALLOCATABLE :: heat_capacity(:,:,:)

END TYPE solid
```

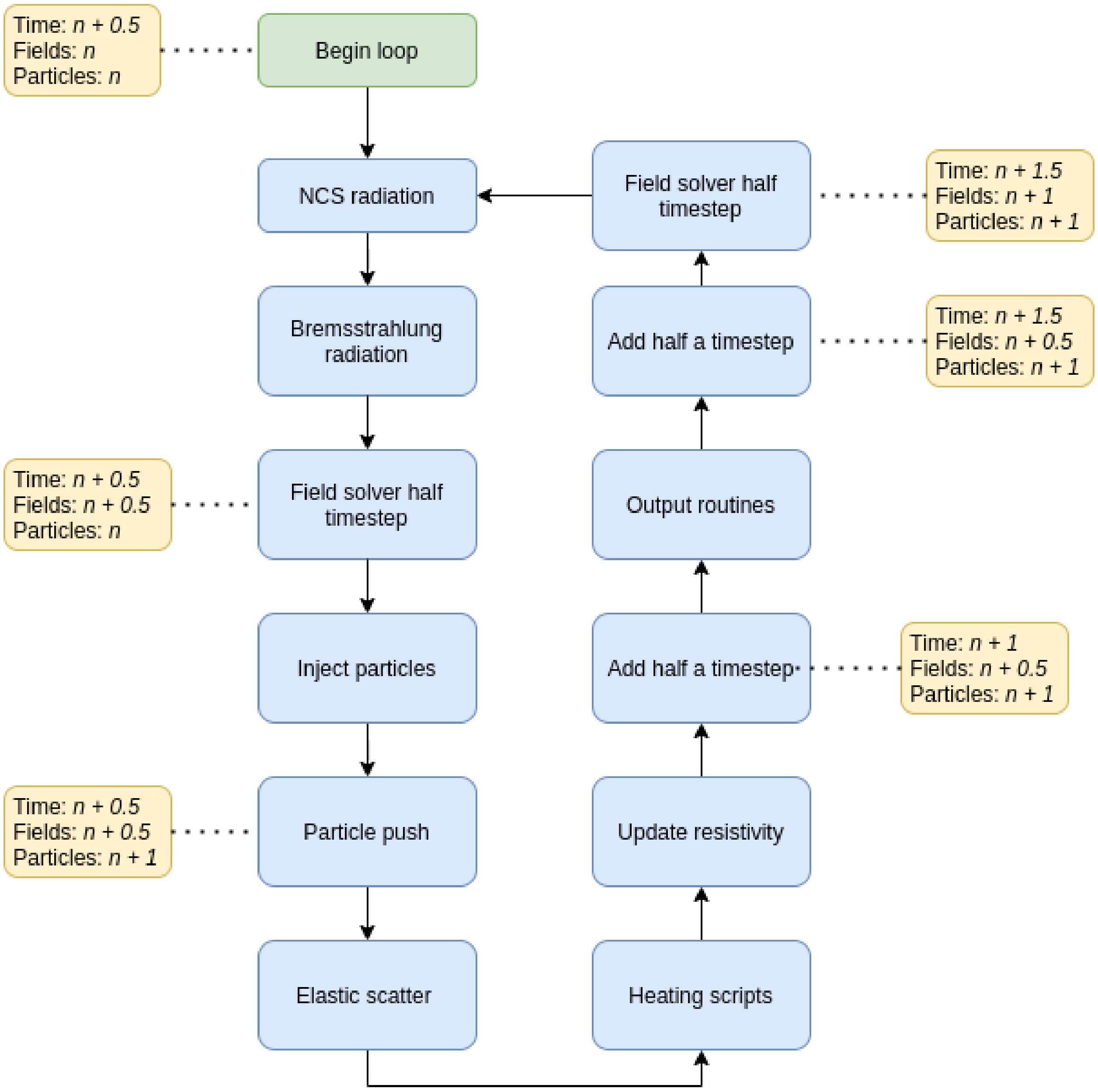


Figure 8: A flow chart which demonstrates the main subroutines called in a hybrid PIC loop. For extra clarity, we also include the current timestep associated with the `EPOCH` time variable, the fields, and the particle positions. Ionisation loss, Ohmic heating, and thermal equilibration of electron and ion species all fall under the label of “heating scripts” in this figure.

Here, mass_no and Z refer to the mass and atomic numbers of the solid respectively. The ion_density array describes the ion number density of the solid in each cell of the simulation window (on the local rank). This value can be set to zero in cells where the solid is not present, allowing the user to control the solid geometries within the simulation window. Here, electron_density corresponds to the total electron number density, and is always set to the product of the solid's ion number density and the atomic number (does not change with ionisation state).

While individual solid types refer to a single elemental material, compound targets can be created by spatially overlapping solids. For example, plastic could be made by defining a carbon solid and a hydrogen solid in the same cells, with the ion densities of each solid summing to the ion density of the plastic as a whole. The Iex and rad_len variables denote the mean excitation energy (ionisation loss variable) and radiation length (Urban scatter variable) of the material as a whole. This means in the case of plastic, Iex should be set to the mean excitation energy of plastic in both the carbon and hydrogen solids, and not the individual mean excitation energies of carbon and hydrogen respectively. The code uses the mean excitation energy corresponding to the solid with the largest electron number density in that cell.

There are currently 4 resistivity models available in the hybrid code, and different solids can use different resistivities. When updating the resistivity, the code looks for the solid with the highest electron number density in the cell, and uses the res_model variable of that solid to pick which resistivity model to use. This way, multiple resistivity models can be used in the same simulation.

The remaining variables are derived quantities used to speed up the ionisation, elastic scatter, and heating routines:

- theta_fac $= Ze^4/(2\pi\epsilon_0^2)$
- ln_s $= 4\epsilon_0 h/(Z^{1/3}m_e e^2)$
- Z_prime $= Z^{-4/3}k_B/e$
- Iex_term $= 2m_e^2c^4/{I_{ex}}^2$
- dEdx_C $= 1 + 2\ln\left(I_{ex}\sqrt{\epsilon_0 m_e}/(\hbar e)\right)$
- urb_sig is the elastic cross section interpolated to Z for the current solid
- urb_high is a high energy correction interpolated to Z for the current solid
- urb_el(22) is a $\lambda_1(\epsilon_k)$ table for electrons
- urb_pos(22) is a $\lambda_1(\epsilon_k)$ table for positrons
- heat_capacity is the heat capacity of the solid

where we have used the same notation as in the theory sections where these concepts are introduced.

EPOCH stores solids in the solid_array, and stores the number of solids in the solid_count variable.

3.3 Hybrid injectors

The equations in Section 2.1 describe how many electrons we may expect to inject in a given timestep, but the implementation in the code provides more flexibility than outlined in that section. These injection routines use the new hy_laser_block data type, and the syntax is a combination of the laser routines and the injection routines.

In EPOCH3D, the hybrid laser block has the following form:

```
TYPE hy_laser_block

  ! Similar syntax to the laser_block type, but without phase
  INTEGER :: boundary
  INTEGER :: ppc

  ! Only spatial profile can vary spatially
  REAL(num), DIMENSION(:,:), POINTER :: profile

  LOGICAL :: use_time_function, use_profile_function
  LOGICAL :: use_omega_function
  TYPE(primitive_stack) :: time_function, profile_function
  TYPE(primitive_stack) :: omega_function

  REAL(num) :: intensity, omega, t_start, t_end, profile_min
  REAL(num) :: efficiency
  INTEGER :: omega_func_type, species
  LOGICAL :: has_t_end

  INTEGER :: mean, e_dist, ang_dist

  ! User specified energy/weight values
  REAL(num) :: user_mean_KE, user_weight
  REAL(num) :: las_weight_KE
  LOGICAL :: ignore_las

  ! Angular distribution variables
  REAL(num) :: user_theta_max, cos_n_power, top_hat_L
  REAL(num) :: sheng_angle, mean_mult
  REAL(num) :: theta_mean, phi_mean
```

```
    LOGICAL :: use_moore_max
    LOGICAL :: use_sheng_dir

    TYPE(hy_laser_block), POINTER :: next

  END TYPE hy_laser_block
```

where many variables are analogous to those found in the laser block and injector block types. This allows the user to describe a laser in the input deck in much the same way as a normal laser (with spatially varying profile, peak intensity, wavelength and temporal envelope), and to also describe the number of macro-particles to be injected using these laser parameters.

This block allows the user to choose between multiple models for the mean electron energy, energy distribution and angular distribution, as discussed in the following subsections.

3.3.1 Mean energy

There are three options for calculating the mean electron energy, $\langle\epsilon\rangle$ from the laser intensity, I and wavelength, λ currently present in the boundary cell:

- `a0`: $\langle\epsilon\rangle = a_0 m_e c^2$, see (2.4)
- `wilks`: $\langle\epsilon\rangle = \left(\sqrt{1+\frac{I\lambda^2}{1.37\times10^{18}Wcm^{-2}\mu m^2}}-1\right)m_e c^2$, see [33]
- `e_val`: User-defined mean energy.

In the case of `e_val`, the laser properties are not used to calculate the mean energy (although they may still be used to determine how many particles must be injected). Instead, this user defined mean energy is stored in the `user_mean_KE` variable in the hybrid laser block. The choice of mean energy is stored in the `mean` hybrid laser block variable.

3.3.2 Energy distribution

The hybrid laser block is capable of injecting electrons according to one of six energy distribution functions. While there are only three unique energy distributions (`mono`, `exp` and `top_hat`), these distributions can be achieved using different macro-particle weighting methods. The possible energy distributions allowed by the code are characterised by the mean electron energy $\langle\epsilon\rangle$, and can be summarised as:

- `mono`: All electron energies set to $\langle\epsilon\rangle$, and the total number of electrons to inject in this cell and timestep (2.5) is split equally between the injected macro-particle weights.
- `exp`: Macro-particle energies assigned using an exponential probability density function (PDF), $P(\epsilon)$,

$$P(\epsilon) = \frac{1}{\langle\epsilon\rangle}e^{-\epsilon/\langle\epsilon\rangle} \tag{3.1}$$

where all macro-particle weights are equal, and sum to the total number of particles to inject in this cell and timestep according to (2.5).

- top_hat: Macro-particle energies are assigned from a uniform distribution between limits $(1-L)\langle\epsilon\rangle$ and $(1+L)\langle\epsilon\rangle$, where the fractional width, $0 < L \leq 1$ is defined by the user and stored in the top_hat_L variable in the hybrid laser block. The total number of electrons to inject in this cell and timestep (2.5) is split equally between the injected macro-particle weights.

- exp_weight: Macro-particle energies are assigned from a uniform distribution between limits 0 and $M\langle\epsilon\rangle$, where the factor, M, is defined by the user and stored in the mean_mult variable in the hybrid laser block. The weights of all injected macro-particles are exponentially distributed, such that the probability density function of injected real particles matches (3.1), and the weights sum to the total number to be injected (2.5). This distribution is useful if you are interested in the behaviour of high energy electrons from an exponential distribution, as the exp distribution may create too many low energy macro-electrons to get good high energy statistics.

- mono_weight: Macro-particle energies are set to $\langle\epsilon\rangle$, and particle weights are set to a user-defined value stored in the hybrid laser block variable user_weight. Useful for particle beams or bunches.

- mono_las_weight: The total number of electrons to inject in this cell and timestep is calculated from the laser parameters (2.5), and is split equally between the injected macro-particle weights. In this model, the energies assigned to the macro-particles can be different to the $\langle\epsilon\rangle$ value used in calculating the macro-particle weights, and instead takes the user-defined value stored in las_weight_KE. This is useful if you want to inject a low energy beam with the same weight distribution as the electron injection, in case you wish to ensure the total charge in the simulation window remains neutral.

Where appropriate, sampling of these distributions is achieved by converting the probability density function $P(\epsilon)$ to a cumulative distribution function $F(\epsilon)$, using

$$F(\epsilon) = \int_{-\infty}^{\epsilon} P(\epsilon')d\epsilon' \tag{3.2}$$

where $F(\epsilon)$ ranges between 0 and 1. Once an expression for $F(\epsilon)$ is obtained, we rearrange this equation to find an expression for ϵ, and replace the CDF with a random number drawn from a uniform distribution between 0 and 1, x_r. The sampling equations are given in Table 5.

The choice of energy distribution model is stored in the e_dist hybrid laser block variable. The ignore_las variable is used to prevent the laser parameters being calculated when running with a mono_weight distribution.

Distribution	Energy sampling
exp	$\epsilon = -\langle\epsilon\rangle \ln(1 - x_r)$
top_hat	$\epsilon = L\left(x_r - \frac{1}{2}\right) + \langle\epsilon\rangle$
exp_weight	$\epsilon = M x_r \langle\epsilon\rangle$

Table 5: Equations used to sample energies in the non-uniform energy distributions for the hybrid electron injector. Here, x_r refers to a uniformly distributed random number between 0 and 1, and distribution-specific parameters are defined in Section 3.3.2.

3.3.3 Angular distribution

Once the weights and energies of the macro particles have been sampled, we must determine the momentum direction of the incoming particles. In Section 2.1, we introduce the Moore angle for injection cone widths (2.6) and the Sheng angle for the cone axis in oblique laser-solid collisions (2.7). Both of these energy-dependent injection characteristics can be turned on and off using the hybrid laser block variables use_moore_max and use_sheng_dir.

The basic form of injection is into a cone, where the half angle, $\theta_{\max}$ is taken to be the smaller of the Moore angle (if used), or a user-defined cone angle stored in the user_theta_max hybrid laser block variable. Within this cone, we sample θ values using one the following distributions:

- uniform: Electron directions are uniformly distributed within the cone, which corresponds to a probability density function for solid angles $P(\Omega)$ of the form:

$$P(\Omega) = \frac{1}{2\pi(1 - \cos\theta_{max})} \tag{3.3}$$

- cos: Electrons are injected to a cone of maximum cone half-angle $\theta_{\max}$, in a distribution proportional to $\cos^n(\theta)$, where n is a user defined parameter stored in the cos_n_power hybrid laser block variable. The corresponding PDF is:

$$P(\Omega) = \frac{n+1}{2\pi\left(1 - \cos^{n+1}(\theta_{\max})\right)} \cos^n(\theta), \tag{3.4}$$

  where we note this returns the uniform case for $n = 0$.

- beam: All θ values are set to zero, particles are injected in a straight line perpendicular to the boundary.

The choice of angular distribution model is stored in the ang_dist hybrid laser block variable. In all models, the azimuthal angle ϕ is sampled uniformly. As in Section 3.3.2, we can sample from these PDFs by converting them into CDFs, and replacing the CDF with a random number. These functions are given in Table 6

Distribution	Angular sampling
uniform	$\theta = \cos^{-1}(1 - x_r(1 - \cos\theta_{\max}))$
cos	$\theta = \cos^{-1}\left((1 - x_r(1 - \cos^{n+1}(\theta_{\max})))^{\frac{1}{n+1}}\right)$

Table 6: Equations used to sample θ values in the non-beam angular distributions for the hybrid electron injector. Here , x_r refers to a uniformly distributed random number between 0 and 1, and distribution-specific parameters are defined in Section 3.3.3.

After the code has calculated the $\theta_{\max}$ value, sampled a θ, ϕ direction, and applied a Sheng rotation (if used), the user can perform a final rotation to the injector as a whole using the user-defined (θ, ϕ) values stored in the hybrid laser block variables theta_mean and phi_mean.

3.4 Ionisation loss and Møller scatter

As discussed in Section 2.4, ionisation energy loss treatment can be split between a continuous energy loss due to the creation of low energy δ-rays, and discrete emission considering the recoil due to high energy δ-rays. Both approaches are present in the file *hy_ionisation_loss.F90*, and are accessed through the subroutine run_ionisation_loss. This subroutine cycles over all species, and calls the continuous and discrete loss subroutines for every particle in an electron species.

The continuous_energy_loss subroutine calculates the density correction (2.35), then calculates the energy loss, $d\epsilon$ from the stopping power (2.30) and the step size. After applying this loss, $d\epsilon$, the subroutine also outputs $d\epsilon$ for use in the heating routines.

Just as in bremsstrahlung and NCS emission, high energy δ-rays are emitted according to an optical depth model, which requires an additional variable for the particle type: optical_depth_delta. As this could affect the speed of the particle pusher as particles pass between boundaries, all hybrid routines have been hidden behind a pre-compiler flag. This optical depth has also been made a restart variable. At the start of the simulation and after each emission, the hot electrons are assigned an optical depth for emission, τ from an exponential distribution. After each time step, Δt, the particle advances an optical distance $\Delta\tau$,

$$\Delta\tau = \sigma n_e \Delta t \tag{3.5}$$

where σ is the cross section for high energy δ-ray emission (2.39), and n_e is the background total electron density evaluated at the hot electron position.

3.5 Heating routines

The original Davies method of heating in a hybrid PIC code uses the heat capacity (2.86), and the heating equations of Ohmic (2.89) and collisional (2.90) heating [10]. When applied to compound targets, the Davies method used a target-averaged atomic number, Z. However, as the bremsstrahlung cross section scales with Z^2, a compound target and an averaged-Z target would lead to different simulated bremsstrahlung spectra. To fix this, we have extended the Davies model to consider the heating of multiple targets in the same PIC cells.

The heat capacity of each solid is worked out independently using (2.86), and stored in the solid-type variable `heat_capacity`. To reconcile the different solid heat capacities with the need for a single background temperature in each cell, we introduce an effective heat capacity term, C_{eff}

$$C_{\text{eff}} = \sum_{\text{sol}} \left(\frac{Z^{\text{sol}} n_i^{\text{sol}}}{C^{\text{sol}}} \right) \tag{3.6}$$

where we sum over each solid (sol) present in a given cell. Here, Z^{sol}, n_i^{sol} and C^{sol} refers to the atomic number, ion number density and heat capacity of a solid type respectively. As the heat capacity changes with temperature, the effective heat capacity must be recalculated each step. This variable can be used to obtain the solid-averaged inverse heat capacity

$$\langle \frac{1}{C} \rangle_{\text{sol}} = \frac{C_{\text{eff}}}{\sum_{\text{sol}} \left(Z^{\text{sol}} n_i^{\text{sol}} \right)} = \frac{C_{\text{eff}}}{n_e^{\text{tot}}} \tag{3.7}$$

where we have written the total electron number density in the cell as n_e^{tot} for conciseness. Hence, the Ohmic and collisional heating equations for compound targets may be written as

$$\Delta T_e = \frac{\mathbf{j} \cdot \mathbf{j} \eta dt}{(n_e^{\text{tot}})^2 k_B} C_{\text{eff}} \tag{3.8}$$

$$\Delta T_e = \frac{\Sigma_h \Delta \epsilon_h}{(n_e^{\text{tot}})^2 V_{cell} k_B} C_{\text{eff}} \tag{3.9}$$

using the same notation as Section 2.6.

3.6 Field solver

Scipts relating to the hybrid field solver are found in `hy_fields.F90`. Discretisation of the hybrid field equations (2.9 - 2.10) can be achieved in a similar way to the traditional PIC field solver. The step is performed in two half-steps, but with a simple first-order step for the electric field. In 3D, the update for the x-component of the magnetic field in a given cell, $B_x^{n+1/2}(i_x, i_y, i_z)$ reads:

$$\begin{aligned} B_x^{n+1/2}(i_x, i_y, i_z) = B_x^n(i_x, i_y, i_z) &- \frac{\Delta t}{2\Delta y} \left(E_z^n(i_x, i_y + 1, i_z) - E_z^n(i_x, i_y, i_z) \right) \\ &+ \frac{\Delta t}{2\Delta z} \left(E_y^n(i_x, i_y, i_z + 1) - E_y^n(i_x, i_y, i_z) \right) \end{aligned} \tag{3.10}$$

where Δt denotes the timestep, n denotes the time index, and Δy, Δz are the cell sizes in the y and z directions. The hybrid field solver is only a first order method, as the electric

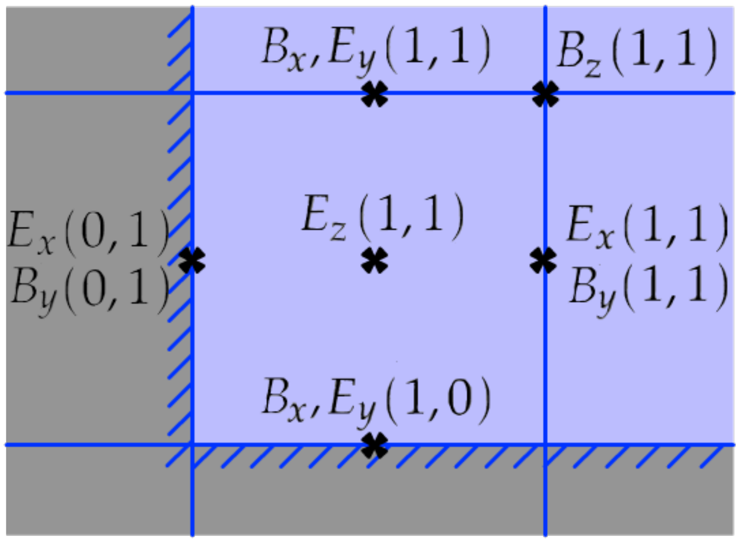


Figure 9: Visualisation of field staggering in an `EPOCH2D` simulation. Here we show how the fields are arranged on the x_{min}, y_{min} simulation window corner, and fields are labelled by their cell indices. The greyed out cells denote ghost cells, starting at index 0.

fields are simply recalculated at the B timestep after B has been updated, via

$$\begin{aligned} E_x^{n+1/2}(i_x, i_y, i_z) = \frac{1}{2}\left(\eta(i_x+1, i_y, i_z) + \eta(i_x, i_y, i_z)\right) \\ \Bigg[\frac{1}{\mu_0 \Delta y}\left(B_z^{n+1/2}(i_x, i_y+1, i_z) - B_z^{n+1/2}(i_x, i_y, i_z)\right) \\ - \frac{1}{\mu_0 \Delta z}\left(B_y^{n+1/2}(i_x, i_y, i_z+1) - B_y^{n+1/2}(i_x, i_y, i_z)\right) \\ - J_x(i_x, i_y, i_z)\Bigg] \end{aligned} \tag{3.11}$$

where we note that the resistivity, η is a cell-centred variable, and the current density, $\mathbf{J}$ shares the same stagger as the electric field. The resistivity and current density in (3.11) are evaluated at different times relative to the B in the two half steps, and could be written as η^n, J_x^n for the update after *Bremsstrahlung radiation*, and $\eta^{n+1/2}$, $J_x^{n+1/2}$ for the update before *NCS radiation* (see Figure 8). Currently, there is no purpose to splitting up the field update before and after radiation - but I believe these field updates could be moved around such that the output routines print fields evaluated at the same time as the particle positions (see discussion of issue #2229, `EPOCH GitLab`). I will therefore leave the field update in two halves to allow for a quick fix in case the main PIC loop is updated.

We must also consider the boundary conditions of our new field solver. The distribution of field evaluation points on the staggered grid has been shown for a 2D grid in Figure 9. The field point $E_z(1,1)$ depends on the neighbouring values $B_y(1,1)$, $B_x(1,1)$, and also ghost cell field points $B_x(1,0)$ and $B_y(0,1)$. If these ghost cell points were set to match their closest simulation-cell counterparts, the gradient across $E_z(1,1)$ would always be zero, and no magnetic field could contribute to the electric field in this point. To correct for this, we update all cells from indices `0` to `nx+1` (for x). We then apply zero-curl boundary conditions to cells which are not directly involved in the calculation of simulation window fields, as performed by the `field_zero_curl` subroutine. This boundary condition sets the values of ghost cells to match the first ghost cell, for example, $E_x(-1,1)$, $E_x(-2,1)$, ... $E_x(1-ng,1)$ would be set to $E_x(0,1)$, where `ng` is the number of ghost cells.

3.7 Resistivity models

In Section 2.3, we introduce the Lee-More resistivity model for cold dense solids (2.11). The full model relies on the chemical potential (2.20), but this relies on a computationally costly evaluation of an inverse polylogarithm function for each cell in each timestep, which would be a significant bottleneck for the code. Instead, we use a reduced form of the Lee-More model which avoids the calculation of μ, as done in the hybrid code `ZEPHYROS` which is discussed in Section 3.7.1.

The resistivity models used by Davies [10] are also present in the code, including a fit to experimental data for aluminium (Section 3.7.2), and a heuristic model resistivity model for plastic (Section 3.7.3).

3.7.1 Reduced Lee-More model

In (2.11), the only terms which depend on the chemical potential are the electron relaxation time τ (2.12), and the A^α factor (2.27). The original Lee-More paper [11] quotes these terms in the non-degenerate plasma limit $\mu \to -\infty$,

$$A^\alpha \to \frac{32}{3\pi} \tag{3.12}$$

$$\tau \to \frac{12\pi^2\epsilon_0^2}{e^4}\sqrt{\frac{m_e}{2\pi}}\frac{(k_bT_e)^{3/2}}{(Z^*)^2 n_i \ln\Lambda} \tag{3.13}$$

which are the SI equivalents of the paper's (27 [11]) and (28a [11]) respectively, and the terms have the same meaning as in Section 2.3. The Lee-More paper also provides a low temperature relaxation time, τ_{cold} given by (2.14).

Our `ZEPHYROS`-style reduced Lee More model starts with an effective hot electron relaxation time, $\tau_{\text{hot}} = A^\alpha \tau$ evaluated in the non-degenerate limit. As the temperature falls, τ_{hot} becomes unphysically small and we switch to an effective cold relaxation time, τ_{cold}

$$\tau_{\text{cold}} = \lambda_1 \frac{R_0}{\bar{v}} \tag{3.14}$$

which is the low temperature relaxation time (2.14) with an additional fitting parameter, λ_1. Our reduced Lee-More resistivity formula reads

$$\eta = \frac{m_e}{Z^* n_i e^2 \tau_{\text{eff}}}\lambda_2 \tag{3.15}$$

$$\tau_{\text{eff}} = \max(\tau_{\text{hot}}, \tau_{\text{cold}}) \tag{3.16}$$

where λ_2 is a second fitting parameter for scaling the total resistivity.

3.7.2 Milchberg fit

In the original Davies code [10], the authors use a simpler form for the resistivity, η as a function of electron temperature, T_e. This comes to a fit from data in an experiment ran by Milchberg *et al* [34], which looked at the resistivity of aluminium up to a temperature of 100 eV. The functional form of the Milchberg fit is

$$\eta = \frac{T_{e,eV}}{5 \times 10^6 + 170 {T_{e,eV}}^{5/2} + 3 \times 10^5 T_{e,eV}} \tag{3.17}$$

where $T_{e,eV}$ is the electron temperature measured in eV.

3.7.3 Heuristic plastic model

Davies *et al* also derive a heuristic resistivity model for plastic [35], which varies between a saturation resistivity, η_0, and the Spitzer resistivity. In this paper, Davies explains that while the room-temperature resistivity of an insulator is significantly higher than a conductor, the ionisation brought on by heating enables hot insulators to take on similar charge-carrying characteristics to conductors.

In Boltzmann theory, resistivity, η rises linearly with electron temperature, T_e - but this breaks down at a certain T_e value and η starts to rise slower than T_e. The saturation resistivity refers to the η value at this regime change [36], and can be roughly described by the Ioffe-Regel criteria

$$\eta_0 = \left(\frac{3\pi^2}{n_{\text{con}}^2}\right)^{1/3} \frac{\hbar e^2}{d} \tag{3.18}$$

where n_{con} is the number density of conduction electrons, and d is the interatomic spacing. In the case of plastic, Davies *et al* assumes $n_e \sim 10^{29}$ m^{-3}, and quotes $\eta_0 = 2.3 \times 10^{-6}$ Ωm.

The parallel Spitzer resistivity, $\eta_{\parallel}^{\text{Spitzer}}$ is derived as a correction to the resistivity of an idealised Lorentz gas [37], which translated to SI units reads

$$\eta_{\parallel}^{\text{Spitzer}} = \frac{\pi^{3/2} m_e^{1/2} Z^* e^2 c^4 \ln \Lambda}{2(2k_B T_e)^{3/2}} \frac{10^{-14}}{\gamma_e(Z^*)} \tag{3.19}$$

where γ_e is a correction factor which has tabulated values in the reference. The perpendicular Spitzer resistivity, $\eta_{\perp}^{\text{Spitzer}}$ can be expressed as $\eta_{\perp}^{\text{Spitzer}} = \eta_{\parallel}^{\text{Spitzer}} F(Z^*)$, where $F(Z^*)$ values are tabulated in Table 1 of a paper by Braginskii, [38], which can be approximated to the fit

$$F(Z^*) = \frac{1 + 1.198 Z^* + 0.222 (Z^*)^2}{1 + 2.966 Z^* + 0.753 (Z^*)^2} \tag{3.20}$$

by Kuritsyn *et al* [39]. The Davies heuristic model uses the parallel Spitzer resistivity with $Z^* = 1$ for plastic, where $\gamma_e = 0.582$ and $F(Z) = 1.96$, such that

$$\eta_{\perp}^{\text{Spitzer}} \approx 1.04 \times 10^{-4} \ln \Lambda T_{e,eV}{}^{-3/2} \tag{3.21}$$

where $T_{e,eV}$ is the electron temperature evaluated in eV. For plastic, Davies uses $\ln \Lambda = 7.7$ such that the full heuristic plastic resistivity model reads

$$\eta = \frac{1}{4.3 \times 10^5 + 1.3 \times 10^3 T_{e,eV}^{3/2}} \tag{3.22}$$

which tends towards the saturation resistivity at low temperature, and the Spitzer resistivity at high temperature.

3.8 K-alpha implementation

Kα radiation in the code is implemented as a secondary-particle emission process, much like NCS, pair-production, bremsstrahlung and Möller scatter already present in the code (see Section 3.4). At present, this package allows the production of K-alpha photons **in copper only**. The code identifies a copper solid by reading the atomic number of the solid material.

In practice, hot electrons are assigned an optical depth of ionisation, and an ionisation event is triggered once the electron reaches this distance. Upon triggering a 1s ionisation, the incident electron kinetic energy is reduced by the binding energy of the 1s state. The momentum direction of the incident electron remains the same after the energy loss, and the previously bound electron is not added to the simulation window. At this point, a new optical depth for ionisation is sampled for the hot electron.

By performing this energy reduction on the hot electron, the code may end up over-sampling the energy loss for the hot electrons. Ionisation energy loss from all bound electron shells is already considered by the ionisation loss routines (Section 3.4), while the Kα sub-routines only consider energy loss due to interactions with the 1s shell of copper. Provided this energy loss is significantly less than the full ionisation energy loss, this will not be a problem - otherwise, the user should disable the Kα electron recoil to prevent over-sampling if running with ionisation energy loss.

Only a fraction of ionisation events lead to radiative Kα emission, as some vacancies are filled using Auger electrons. Currently the fraction of non-radiative transitions is a user-defined variable, and a random number is generated to determine whether a given ionisation leads to Kα emission or not. If a photon emission occurs, a second random number determines whether a Kα_1 or Kα_2 photon is created, which is sampled to produce a 2:1 ratio between the two line emissions. Upon emission, the Kα photon is assigned a random direction, sampled from an isotropic distribution.

While equations (2.100) and (2.101) give the central energies of the Kα_1 and Kα_2 emission lines, the line-widths are not given. The code calculates the central energies for a given temperature, and then shifts these using a number drawn from a Gaussian distribution, to spread the line emissions into Gaussian curves. The standard deviation, σ for these Gaussian curves are user-defined, and take the form

$$\sigma = \sigma_0 + \sigma_1(T'_e) + \sigma_2(T'_e)^2 + \sigma_3(T'_e)^3 + \sigma_4(T'_e)^4 \tag{3.23}$$

where the fit parameters σ_n are set by the user in the input deck for both Kα lines.

3.9 Photo-electric effect

To imitate the photon absorption from the photo-electric effect, a new `attenuate` key has been added to the species block. When switched on, and if the species has been identified as a photon species, then particles within the species have an additional step in the hybrid-PIC loop. This is achieved through a call to the subroutine: `run_photo_electric_Cu`. Here the code loops over all solid objects present in the code, and if one of them is copper (identified by atomic number $Z = 29$), the code then loops over all particles in attenuating photon species. The background number density and electron temperature are interpolated to the particle position, and the κ co-efficient is calculated from (2.103). Over a simulation time-step dt, photons are expected to travel a distance $x = cdt$, and so the attenuated fraction can be calculated from (2.102). This is applied as a reduction in the macro-photon weight, to represent absorbed photons.

However, in the hybrid-PIC code, it is possible for the user to represent the solid density copper target as many overlapping solid copper objects of lower density. In this case, the attenuation would be over-estimated if the same attenuation is applied to each copper solid. Instead, this is scaled by the background number density of copper ions, such that the macro-photon weight after attenuation from a copper solid W is

$$W = W_0 \frac{n_i}{n_{i,0}} e^{-\kappa cdt} \tag{3.24}$$

where W_0 is the weight before attenuation from the current solid, and n_i and $n_{i,0}$ represent the ion densities of the current copper solid, and solid density copper (8.4912×10^{28} m^{-3}) respectively. Thus, the attenuation from multiple overlapping low-density copper solids would be the same as the attenuation from the equivalent single copper solid set-up.

3.10 TNSA boundaries

We have implemented simple algorithms to roughly approximate refluxing behaviour, guided by observations in Section 2.9. These algorithms can be accessed using a new `tnsa` boundary type in the boundaries block. Our parametrisation takes the form of three key parameters:

- `tnsa_escape_KE`: Above this kinetic energy, all electrons escape (to reproduce particle fates observed in Figure 3).
- `tnsa_p_loss`: Upon refluxing, this value is subtracted from the total momentum magnitude, without changing the momentum direction. Typical values can be taken from Table 3.

- `tnsa_scatter_angle`: If refluxing with an outgoing polar angle θ, the electron returns with polar angle $\theta + \theta_r$, where θ_r is a random scatter angle drawn from a uniform distribution between $\pm 0.5 \times$ `tnsa_scatter_angle`. This is to reproduce the large scatter uncertainty present in Figure 7, and typical values are given in Table 4.

which are set by the user in the input deck boundary block.

The `tnsa` boundary type behaves the same as the `reflect` boundary type, until the `particle_bcs` subroutine. Here, particles are reflected as normal, then two additional subroutines are called: `tnsa_part_escape` and `tnsa_part_reflect`. The first of these checks the particle total energy against the total energy corresponding to the given kinetic energy threshold, `tnsa_escape_KE`. All electrons above this energy have the `out_of_bounds` variable set to true, and the code treats these as outflow particles.

The `tnsa_part_reflect` routine calculates the momentum direction of the particle, and subtracts the `tnsa_p_loss` amount from this direction. To prevent the particle from gaining momentum in the opposite direction, if `tnsa_p_loss` is greater than the current electron momentum, then all momentum is lost. If the electron still has momentum, and `tnsa_scatter_angle` is greater than zero, then a random θ is drawn as discussed above (along with a uniformly distributed azimuthal angle ϕ), and the electron direction is rotated.

3.11 Enhanced particle probes

The `EPOCH` probes have been updated to dump the particle ID if the `-DPARTICLE_ID` or `-DPARTICLE_ID4` precompiler flag is also active. These modifications have been added to *probes.F90*, and have been added to track particles for the TNSA characterisations of Section 2.9.

We have also added the capability for probes to output the time at which particles cross them. This requires a new particle variable `probe_time` to store the time each particle triggers the probe, so they can be output like any other particle variable. To prevent slowdown of the particle transport routines, this functionality can only be accessed with the `-DPROBE_TIME` precompiler flag.

Probes in `EPOCH` are defined by a normal vector, $\hat{\boldsymbol{n}}$, and the position vector of a point on the probe $\boldsymbol{P}$. If a particle passes a probe, starting the step at position $\boldsymbol{x}_i$, then $\hat{\boldsymbol{n}} \cdot (\boldsymbol{P} - \boldsymbol{x}_i)$ is the component of the distance covered to get to the probe, in the direction of the normal vector. If the particle finishes the step at position $\boldsymbol{x}_f$, then $\hat{\boldsymbol{n}} \cdot (\boldsymbol{x}_f - \boldsymbol{x}_i)$ is the total step size component in the probe normal direction. Hence, the particle passes the probe at the fraction, f of the particle step, where

$$f = \frac{\hat{\boldsymbol{n}} \cdot (\boldsymbol{P} - \boldsymbol{x}_i)}{\hat{\boldsymbol{n}} \cdot (\boldsymbol{x}_f - \boldsymbol{x}_i)}. \tag{3.25}$$

Assuming the particle moves at a constant speed, then time $f dt$ has elapsed since the start of the step. As shown in Figure 8, the `time` variable is evaluated half a timestep ahead of the particles at the start of the particle push, so `probe_time` is equal to `time` $+ dt(f - 0.5)$.

3.12 File injectors

EPOCH currently has the ability to load in particles from a file, but this can only be used to set initial conditions. We have also added the capability to inject particles from files, by extending the injector_block type. This functionality has been added in case the user wishes to inject macro-electrons from a PIC code probe into a hybrid simulation.

In addition to the previous variables in the injector block, we also include:

- inject_from_file: Logical switch for file injection.
- inject_from_file: Logical switch to track if any particles are left to inject.
- custom_id: A unique code used for injector file units.
- next_time: Time for the next particle to be injected.
- x_data_given: Records if file has been provided for x injection position (2D, 3D only).
- y_data_given: Records if file has been provided for y injection position (2D, 3D only).
- z_data_given: Records if file has been provided for z injection position (3D only).
- px_data_given: Records if file has been provided with injection p_x values.
- py_data_given: Records if file has been provided with injection p_y values.
- pz_data_given: Records if file has been provided with injection p_z values.
- t_data_given: Records if file has been provided with injection times for each particle.
- w_data_given: Records if file has been provided with the weights of each injected particle (if not compiled with -DPER_SPECIES_WEIGHT).
- id_data_given: Records if file has been provided with injected particle ID values (only if compiled with -DPARTICLE_ID4 or -DPARTICLE_ID).

Our file injectors require a different file for each injected particle variable, which are specified by the user in the input deck. Injection time, weight (if needed) and position (for 2D and 3D injectors) are mandatory variables, the rest can be omitted. These variable files consist of a list of numbers, which correspond to the values of this variable for the particles which are injected. Particles must be listed in the files in order of injection time.

The file injector routines look up the time for the next particle injection. If the particle enters the simulation window in the next time-step, the particle is added to the ghost cells at a position which allows it to pass the simulation boundary at the injection time and position. This is done by assuming the particle moves with the injection momentum during its passage through the ghost cells. This also prevents particles from being injected twice when running from a restart dump, as particles added before the current time-step are skipped. Note that the time at the end of the time-step is `time` $+ 0.5 *$ `dt`, as the `time` variable is evaluated half a time-step ahead of the particle positions when injectors are called (see Figure 8).

Files remain open while there are particles left to inject, and all processors read the files together (but only the processor containing the injection position injects them). This allows new processors to pick up from where old processors left off when we perform a load balance.

As all files remain open while injecting, they must all be assigned a unique file unit. Injector can hold up to 6, 8, or 9 variables in 1D, 2D, and 3D respectively, so we must reserve this number of file units (`custom_var_num`) for each injector. To prevent this routine from taking up all available file units, we arbitrarily limit the number of file injectors to 10^6, and start file units at 10^7. Each injector is given a unique ID, and each variable type is given a unique value between 0 and `custom_var_num`-1. For example, the file unit for the second injector's p_x values would be given by $10^7 + 2 *$ `custom_var_num` $+ 0$, as the unit offset for p_x is 0.

4 Benchmarks

We have rigorously benchmarked all aspects of the hybrid code against experimental literature wherever possible. The input decks used to generate these results are present in the example hybrid deck directory in the EPOCH source code. Our benchmark results look close to identical in all dimensions, so only the EPOCH3D results have been shown - except for the dimensionally dependent results of Section 4.9. Also note that in EPOCH1D, there is no y injector for Section 4.1.1.

If the *Benchmarks* directory has been supplied with this code, you will find sub-directories containing each individual test, broken down into *1D*, *2D* and *3D* sub-directories. These contain the input decks and post-processing MATLAB scripts for each test, along with a *run_test.sh* executable. In order to use this, ensure the *Benchmarks/quick1d.sh*, *Benchmarks/quick2d.sh* or *Benchmarks/quick3d.sh* files have an `epochDir` variable with a path to the EPOCH directory on your machine. Also ensure that you have compiled EPOCH with the `-DHYBRID` precompiler flag.

4.1 Tests

In some cases, experimental evidence was not available, and we instead compare the results of EPOCH against other codes. In other cases we were interested in testing the functionality of our extensions, like the file injectors or the TNSA boundaries. This section contains these non-experimental benchmarks.

4.1.1 File injectors and probe times

An input deck was developed to simultaneously test both the probe-time and the file-injector capabilities, discussed in sections 3.11 and 3.12 respectively. We created files which injected 5 electrons spaced 1 fs apart, into simulations where all hybrid routines were deactivated. To

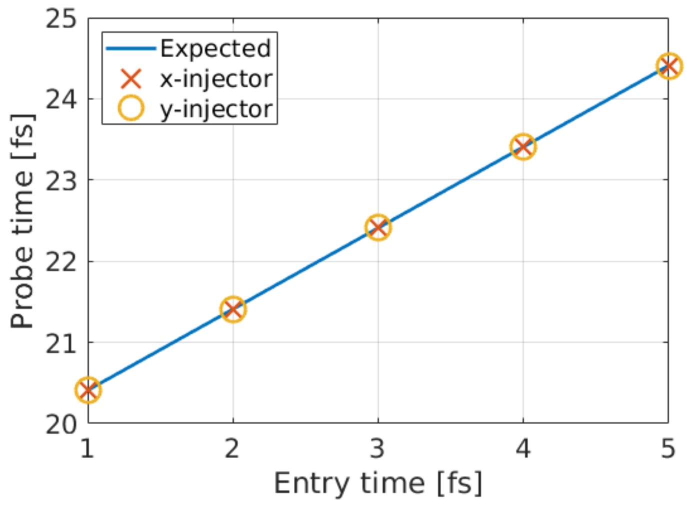


Figure 10: Electrons were injected at different times, travelling at 0.86c, and probes detected the time they passed the 5 μm point. For comparison, the solid line is the time we expect electrons to pass the probes, as a function of injection time. Two separate injectors were tested in the same simulation, one injecting from x_{min} and the other from y_{min}

test the multiple injector capability, 5 electrons were injected from the x_{min} boundary and 5 from y_{min}. The electron momentum was set to 4.588×10^{-22} kgms^{-1} directed perpendicular to the injection boundary in both cases, which corresponded to an electron speed of roughly 0.86c. Probes placed at 5 μm from the injector boundaries output the times our injected electrons passed, and the known injection times and speeds allowed us to calculate the expected times of probe output. Figure 10 shows the times of probe hits and the expected time as a function of injection time.

We see excellent agreement between the expected time and the probe time for both injectors. This test was also performed from a restart dump after the 2 fs electrons were injected, and we found our injectors picked up where they left off and injected the correct number of electrons at the correct times. During testing, an issue with the core `EPOCH` output routines were discovered, as particles inside the simulation window were arriving at the probe at a later time than expected. This was because the particle positions were paired with the wrong time in the SDF file, and were reloaded in the wrong place after the restart (now fixed, see issue #2229).

4.1.2 TNSA boundaries

We wrote an input deck to test the three properties of the TNSA boundaries: escaping electrons, reflux momentum loss and reflux scatter, as discussed in Section 3.10. These were done in hybrid simulations with all the hybrid physics routines switched off. The escape kinetic energy threshold was set to 10 MeV, momentum loss was set to 1 MeV/c, and the

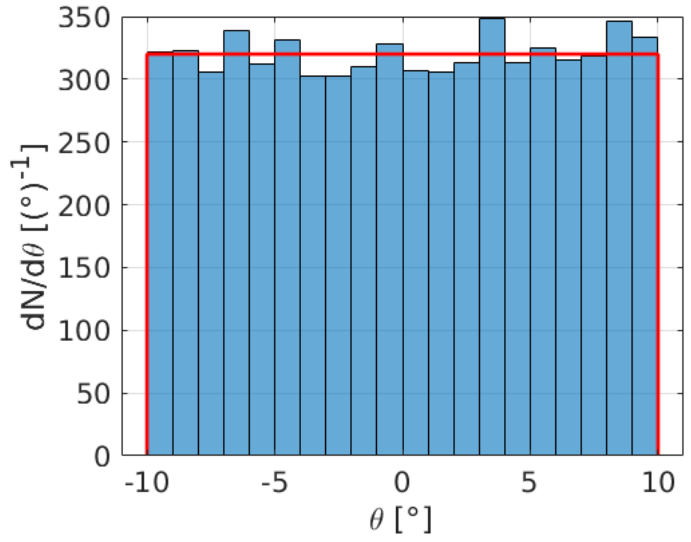


Figure 11: Angular distribution of an electron bunch when refluxing through a hybrid TNSA boundary. The solid red line shows the expected distribution, corresponding to a uniform distribution between ±10°.

scatter angle uncertainty was set to 20°.

To test all tnsa boundary conditions, we injected two electron bunches, one with a mean kinetic energy of 100 MeV, the other with 5 MeV. All 100 MeV electrons passed through the TNSA boundary, and all 5 MeV electrons reflected with a momentum magnitude change of -5.341 $\times$ 10^{-22} kgms^{-1} as expected. Figure 11 shows the outgoing angles are uniformly distributed between ±10°, also as expected.

4.1.3 Thermal equilibration

We were unable to find experimental data for the temporal evolution of electron and ion temperatures in a solid, so instead we built a simple solver for (2.93) in `MATLAB`. This was tested against a hybrid `EPOCH` simulation, with all hybrid physics switched off apart from thermal equilibration. The electron and ion temperatures were initialised to 100 eV and 50 eV respectively, for an aluminium target. Figure 12 shows the temporal equilibration of temperature for both `MATLAB` and `EPOCH`, demonstrating a good agreement between the two codes.

The ion temperature variable is only treated as a restart variable if it is defined in the code. To ensure it is loaded as expected, this test was also performed in two stages with the code restarting from an output dump in the middle. It was found that the ion temperature was correctly loaded, and the evolution matched that of Figure 12.

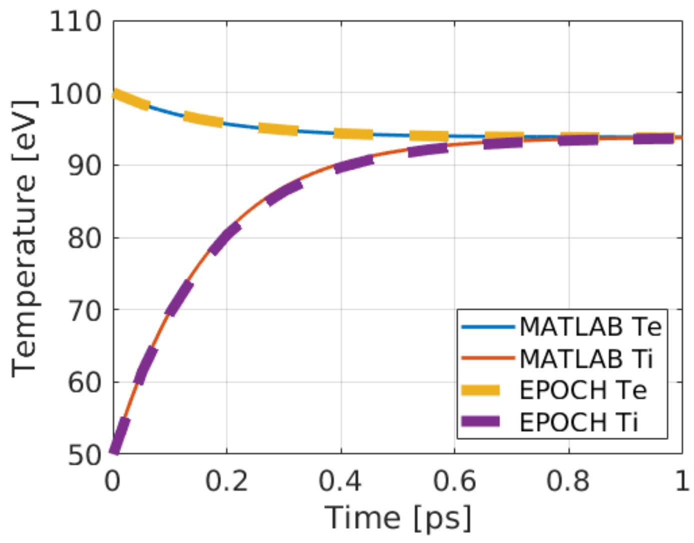


Figure 12: Thermal equilibration of the ions and electrons in an aluminium target. The hybrid routines in `EPOCH` are tested against a `MATLAB` prototype to demonstrate the hybrid routines are behaving as expected.

4.1.4 Geant4 δ-rays

To test the creation of δ-rays, we injected 10^5 electrons with kinetic energy 50 keV into an aluminium target. This was simulated in a hybrid simulation, with only ionisation energy loss switched on. While δ-ray emissions down to 1 keV energy were made to affect electron trajectories, only δ-rays above 50 keV energy were added to the simulation. A probe was positioned at 100 μm to detect passing electrons and δ-rays, and the angular distributions and energy spectra of the particles were recorded.

This set-up was repeated in `Geant4`, for simulations where all electromagnetic physics processes were removed from the physics library, except electron ionisation loss. Here, electron and δ-ray momenta were output as they escaped through the rear surface of a 100 μm aluminium volume. Figure 13 shows the energy and angular spectra of all electrons (both original and δ-rays) passing the 100 μm point.

The two codes show good agreement, with most electrons remaining around 50 MeV, with another low energy peak corresponding to the creation of δ-rays. The angular distributions of electrons passing the probe are similar in both codes, but `EPOCH` over-estimates the large angle scatter. This is because `Geant4` also considers the binding energy of the δ-ray electron when calculating the energy/momentum conservation (2.40). As this is a small effect which mostly affects lower energy electrons, this will be ignored in `EPOCH`.

This test was repeated, split between two runs with a restart in between. This reproduced the original results, which demonstrate the optical depth of δ-ray emission is correctly written to SDF file and reloaded in the restart process.

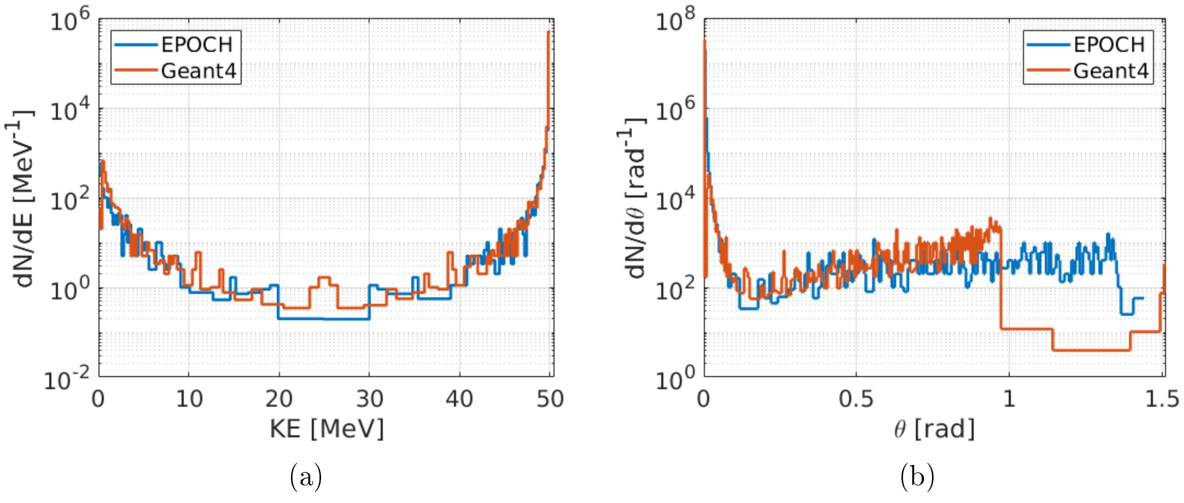


Figure 13: The energy spectrum (a) and angular distribution (b) of electrons passing 100 μm in an Al target. The initial bunch consisted of 10^5 electrons with kinetic energy 50 MeV. Simulations were performed in both `EPOCH` and `Geant4`.

4.2 Hanson elastic scatter

Hanson *et al* [40] performed an experiment looked at the elastic scatter of 15.7 MeV electrons traversing gold foil targets. This benchmark tests our elastic scatter routines by recreating the experimental electron scatter distribution shown in their Figure 3, for the 18.66 mg/cm^2 (9.67 μm) target. Two `EPOCH` input decks were used, both injecting 15.7 MeV (kinetic energy) electron bunches, into gold targets with a particle probe positioned at 9.67 μm. One input deck used the Davies method for angular scatter, and the other used the Urban method adapted for use in a PIC code. To confirm the validity of our Urban implementation, we also include the result from an equivalent Urban `Geant4` simulation. The simulation results are shown in Figure 14, along with the Hanson data for comparison.

The `EPOCH` and `Geant4` Urban curves are in good agreement, which suggest that `EPOCH` cell sizes are small enough to neglect the difference between geometric and true path lengths using this approach. The Davies algorithm is better at describing small angle scatter, but Urban is better at reproducing the large angle scatter tail (unsurprising as the Urban tail parameters are fit to experimental data). Both curves demonstrate a reasonable agreement with the experimental data for the scatter of most electrons.

4.3 Lockwood ionisation loss

This benchmark tests both the angular scatter and continuous ionisation energy loss routines. Lockwood *et al* [41] investigated the energy deposited as a function of target depth for a variety of targets and angles of incidence. Here, we reproduce the depth dose curve for 0.5 MeV electrons in tantalum, as shown in pages 100 and 101 of their report. These simulations injected 0.5 MeV electron bunches, and the deposit energy was inferred from the temperature

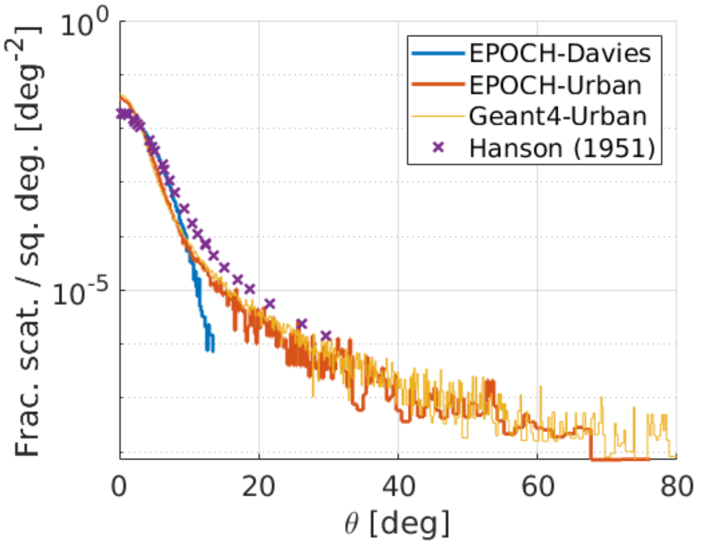


Figure 14: The fraction of injected 15.7 MeV electrons passing a particle probe placed 9.67 μm away from the injection point in a gold target. This number is normalised to the solid angle traced by each θ bin, in units of square degrees. The experimental data [40] is compared against an `EPOCH` simulation using the Davies method for angular scatter, and `EPOCH` and `Geant4` simulations using the Urban method.

increase and the known specific heat of tantalum in `EPOCH`. Figure 15 shows depth-dose curves created using the Davies and Urban algorithms.

This benchmark only tests continuous energy loss, as 0.5 MeV electrons are too low energy to create non-negligible δ-ray or bremsstrahlung X-ray emission. The injected electron current used in the experiment are assumed too small to draw a significant return current. Both algorithms produce good agreement with the experimental data.

4.4 NIST ionisation and bremsstrahlung

In order to test the ionisation and bremsstrahlung energy loss mechanisms for electrons, we repeat the test performed by Wu *et al* [4] and measure the total electron stopping power. This benchmark injected several electron bunches at different energies, and calculated their energy loss and step size each step. In order to get a smoother bremsstrahlung curve, the photon weight was reduced to 0.005 (emission 200 times more likely, for bremsstrahlung macro-photons with 0.005 the weight of the emitting macro-electron). The electron stopping powers are shown in Figure 16 for an Al target, plotted alongside the expected NIST stopping powers [42].

Figure 16 shows excellent agreement between `EPOCH` and NIST, recreating both the ionisation-driven peak at low electron energies, and the high energy bremsstrahlung-driven regime.

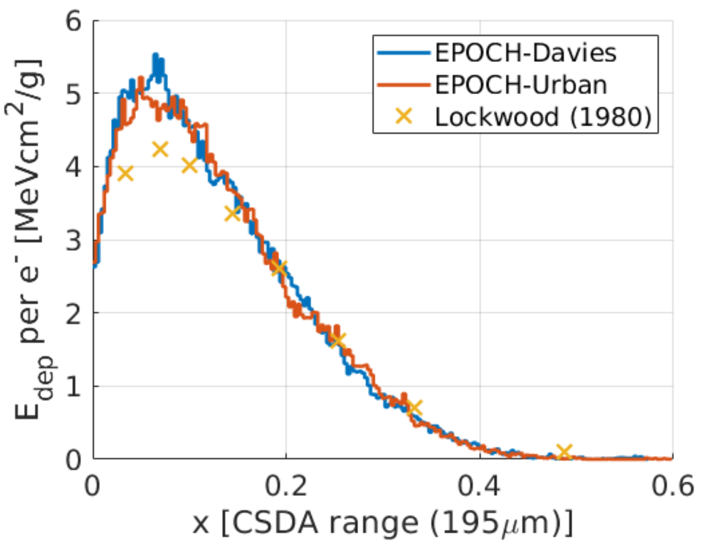


Figure 15: Energy deposition as a function of depth for 0.5 MeV electrons in tantalum. The depth, x is measured in units of expected range of electrons in the material, using the continuous slowing down approximation (CDSA). The energy deposition is quoted as the total energy deposited in MeV per x bin size (cm) per material density (g/cm^3), divided by the total number of incident electrons.

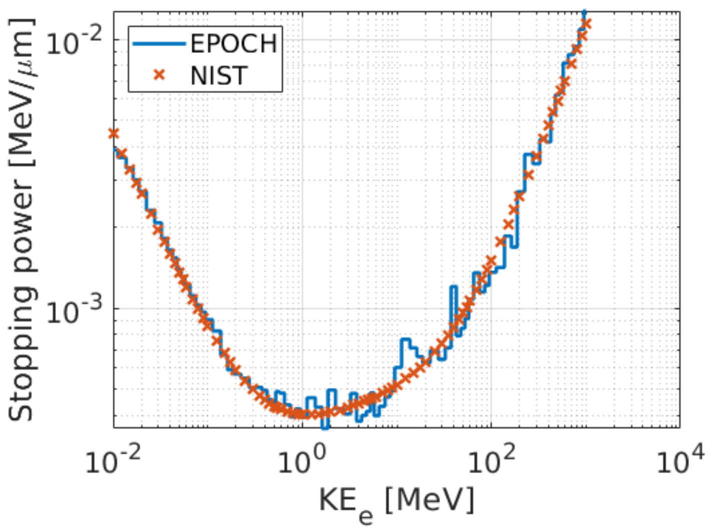


Figure 16: Stopping power of electrons in Al targets, as a function of the electron kinetic energy.

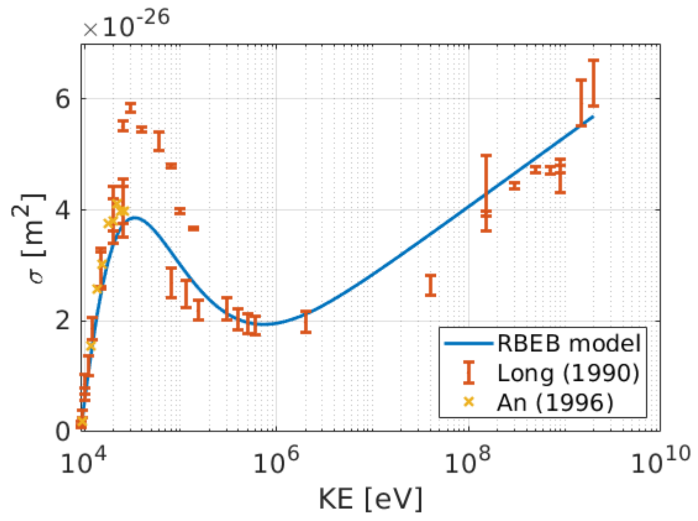


Figure 17: Electron impact ionisation cross section for bound electrons in the 1s shell of copper, as a function of incident electron kinetic energy. The theoretical RBEB model is compared against the experimental data of Long [43] and An [44].

4.5 K-alpha production

K-alpha emission consists of two phases - an ionisation phase, and a photon production phase. To test the validity of the electron impact ionisation part, we may first check the accuracy of the RBEB model in reproducing the experimental copper 1s cross section. This comparison has been made in Figure 17. Here it can be seen that the RBEB model provides a good fit to experimental data, although there is ambiguity in the peak. In the Long [43] data, there is an overlap between low data-points and high data-points at around 100 keV incident electron energy, and the RBEB curve follows the low set. The curve also follows the more recent results of An [44], and so it may be assumed that RBEB is a good model for the ionisation cross section.

A numerical experiment was performed to ensure the correct number of ionisation events were achieved. A 100 keV electron bunch, consisting of 10^6 electrons (N_e) was injected into the simulation window through the x_{min} boundary. The simulation spanned 100 microns in x, L_x, and had a background of solid density copper ($n_i = 8.5 \times 10^{28}$ m^{-3}). At 100 keV incident electron kinetic energy, the RBEB model predicts a cross section of $\sigma = 3.006200 \times 10^{-26}$ m^2, and assuming all ionisation events lead to Kα photons, we would expect the incident electrons to produce a number of photons

$$N_\gamma = N_e \sigma L_x n_i = 255{,}527 \text{ photons} \tag{4.1}$$

as they passed the full length of the simulation window. All other physics packages were switched off. In practice, 259,277 photons were created, within 1.5% of the expected value. Hence, the Kα sampling was behaving as expected.

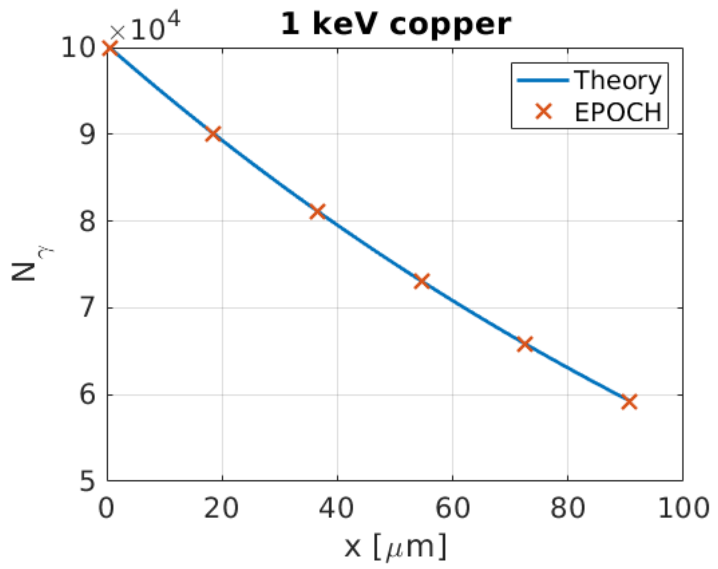


Figure 18: The number of Kα photons remaining after traversing different depths of a solid-density copper target, heated to a background electron temperature of 1 keV. A theoretical curve using the Suxing model has been provided for comparison.

4.6 Photo-electric attenuation

In photo-electric attenuation, photons are absorbed as they traverse a target. Currently, only copper targets are supported for photon attenuation, and the attenuation is only accurate for photons around Kα energies (around 8 keV). For this benchmark, 500 macro-photons representing 10^5 real photons of energy 8 keV were initialised at the x_{min} boundary. The copper background electron temperature was set to 1 keV. The number of real photons remaining as the photon bunch traversed the copper target was tracked, and this has been plotted in Figure 18. Here it can be seen that the attenuation matches the expected attenuation from the Suxing model, and so this form of photon attenuation has been implemented successfully.

4.7 Milchberg resistivity

We can test the `EPOCH` resistivity models by comparing our resistivities against the experimental values found by Milchberg *et al* [34]. The input deck used for this benchmark set up an aluminium target with no injected electrons, and an initial electron temperature which rose linearly with the x direction. This benchmark tested the reduced Lee-More resistivity model in two simulations, one without any model parameter modification, and other where λ_1 and λ_2 have been changed to show better agreement with the data. The resistivity output in each cell was paired with the temperature of that cell to create the curves of Figure 19.

The model parameter λ_1 has the effect of causing the peak resistivity to occur at a higher electron temperature, and λ_2 changes the height of the peak. While the parameterisation

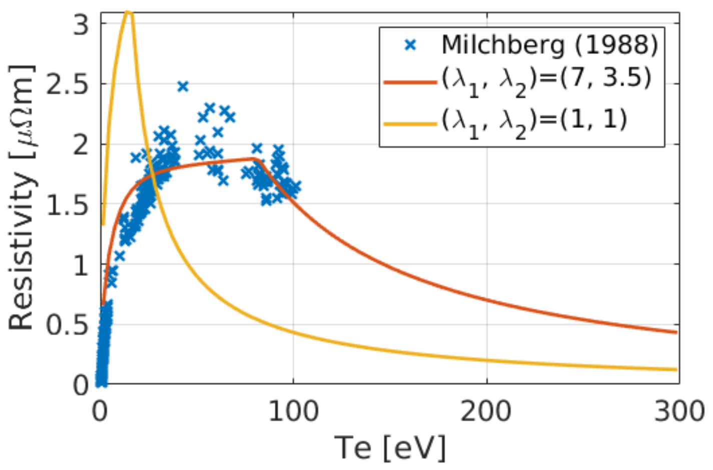


Figure 19: Resistivity curves using the reduced Lee-More resistivity model, plotted against the experimental Milchberg data [34]. The model parameters λ_1 and λ_2 have been varied between the two curves, one modified to overlap with the data, the other without any modification.

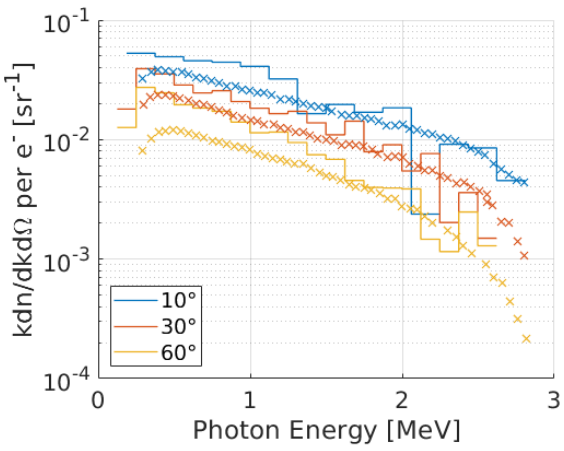
allows for greater control over the low temperature side of the resistivity curve, changing λ_2 from 1 will prevent the high temperature curve reproducing the high temperature limit of the full Lee-More model.

4.8 Rester bremsstrahlung radiation

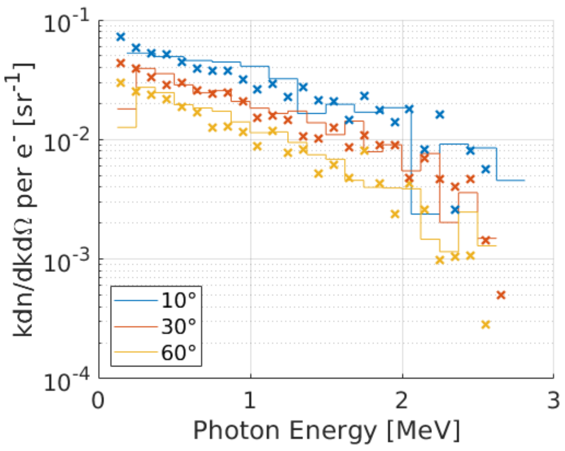
While Section 4.4 considers the electron energy loss due to bremsstrahlung radiation, the emission of X-rays was not tested. We benchmark X-ray production by reproducing the experimental bremsstrahlung spectra shown by Rester *et al* in their Figure 17 [45]. This experiment measured the X-ray energy spectrum created when 2.8 MeV electrons passed through a 2.27 g/cm^2 thick (1.176 mm) gold target, for X-rays escaping the target at 10°, 30° and 60° to the electron injection direction.

The `EPOCH` simulations injected electrons of energy 2.8 MeV into a 1.176 mm gold target, and with elastic scatter, ionisation loss and bremsstrahlung switched on. The minimum photon energy added to the simulation was 200 keV, and no bremsstrahlung macro-photon weight modification was used. Bremsstrahlung X-rays within ±5° of the Rester angles were grouped, and their energy distributions have been plotted in Figure 20.

It was found that using angular bremsstrahlung emissions made almost no difference to the angular distribution of X-rays (elastic scatter of electrons dominated). The `EPOCH` results show generally good agreement with the Rester data, but `EPOCH` overestimates the low energy X-ray spectra, particularly at higher angles of scatter. This benchmark was repeated in `Geant4`, and while full `Geant4` simulations did reproduce the Rester data, we could also reproduce the `EPOCH` data when switching off attenuation from the photo-electric effect. This effect provides an energy loss mechanism for lower energy X-rays which is not included in



(a) `EPOCH` vs Rester (1970)



(b) `EPOCH` vs `Geant4` without photo-electric effect

Figure 20: Energy spectra of bremsstrahlung radiation from 2.8 MeV electrons traversing a 1.176 mm gold target. Bremsstrahlung photons have been grouped by outgoing direction relative to the electron injection axis, θ. The energy spectrum is given as the total X-ray energy in a photon energy bin, divided by the size of the photon energy bin, divided by the total number of injected electrons, divided by the solid angle range of the θ bin (which spans $\pm 5°$ of the quoted angle for simulated data). The solid line in both figures is the `EPOCH` data, and the crosses denote the experimental Rester data in (a), and results from a `Geant4` simulation with all physics switched on apart from the photo-electric effect in (b).

EPOCH, and is noticeable in this thick, high-Z target. Hence, EPOCH bremsstrahlung spectra can only be trusted for high energy X-rays, or in thin targets where X-ray attenuation is negligible.

4.9 Evans background heating

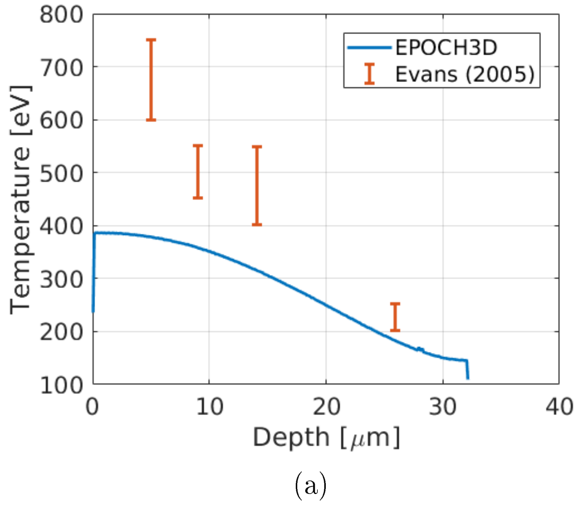
This benchmark attempts to recreate the experimental data of Evans *et al* [46], which measures the temperature as a function of depth in a target after exposure to a high-intensity short-pulse laser. These 800 fs shots focused 300 J of laser energy into a 10 μm focal spot, corresponding to a peak intensity of around 3.1×10^{20} Wcm^{-2}. Plastic targets were used for this experiment, with a 0.2 μm Al tracer layer sandwiched at various depths in different targets for experimental temperature measurements. While we can roughly reproduce the laser parameters, we have had to make assumptions on the electron injection angle (set to 20°), and absorption efficiency (set to 3%), hence we are not expecting a perfect fit to the data.

Our simulations use the Davies model for elastic scatter. We use an exponential distribution of injected electrons, with the energy dependent Moore angle also applied. We also use ionisation loss with Møller scatter, but only add δ-rays over 50 keV kinetic energy to the simulation, otherwise the δ-ray kinetic energy is dumped to the cell for background electron heating. We do not consider thermal equilibration to the ion species, as we are interested in the peak electron temperatures. Bremsstrahlung energy loss and recoil for the electrons is considered, but no X-ray macro-particles are added to the simulation. Perfectly reflective boundaries are used for $x_{\min}$ and $x_{\max}$, and the remainder are open (if present). We use the reduced Lee-More model resistivity for Al, with $(\lambda_1, \lambda_2) = (7, 3.5)$, and the heuristic plastic resistivity for the plastic targets. The full target in our simulation spans 0 to 32.2 μm, with the Al tracer layer present between 28 μm and 28.2 μm. The temperature distributions are shown in Figure 21.

Given the uncertainty in the absorption parameters, we have a fairly accurate qualitative agreement with the experimental data. The influence of the Al layer can be seen by the slight temperature increase at 28 μm in Figure 21a, and in the line of differing temperatures in Figure 21b.

4.10 Benchmarking summary

In general, we are capable of reproducing relevant experimental results which demonstrate good agreement with other simulation codes like Geant4. The main inaccuracy revealed in these tests is the lack of the photo-electric effect, which causes EPOCH to overestimate the X-ray spectra of low energy X-rays escaping thick targets. The code also overestimates the large-angle scattering of δ-ray emission, as we do not consider the binding energy of electrons as they are excited. When running EPOCH-hybrid, the user should be aware of these limitations.



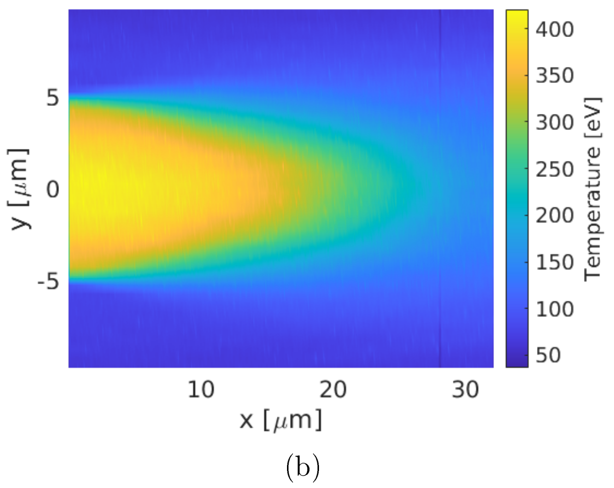


Figure 21: Electron temperature distributions for the CH-Al-CH Evans target after exposure to a 800 fs laser of peak intensity 3.1×10^{22} Wcm^{-2}. We show a line-out of the central temperature (a), and a heatmap of the temperature distribution in the $z = 0$ plane for the EPOCH3D result (b).

5 User manual

This section aims to provide a general overview of the previous sections, along with information on the new blocks for the EPOCH input deck. See the relevant Theory or Implementation sections for a more detailed breakdown of the codes.

EPOCH now has the capability of running as an electron-transport code for laser-solid simulations. Previously, users interested in simulating laser-solid interactions would have to specify a laser boundary, and represent the solid as a cold, dense collection of macro-particles. This approach is computationally expensive, and prone to unphysical behaviour like numerical self-heating. Electron-transport codes treat the solid as a background static fluid, which a small number of high energy macro-particles move through instead.

To start running these hybrid-PIC simulations, the user must become familiar with three new blocks for the input deck: the hybrid block for switching on and off individual subroutines, the solid block for defining solid geometries, and the hy_laser block for electron injection. Explanations of these blocks will be provided in the following sections, along with extensions to existing blocks. An additional Kα radiation block has also been added to the code, which has been written to run in hybrid-PIC mode.

5.1 Hybrid block

The hybrid block determines which subroutines are accessed in the hybrid PIC loop, and is also responsible for the initialisation of some background and model parameters. An example hybrid block is shown below:

```
begin:hybrid

  use_hybrid = T
  use_fields = T
  use_collisions = T
  use_scatter = T
  use_ohmic_heating = T
  use_thomas_fermi = T
  use_ion_temp = F
  produce_delta_rays = T
  electron_temperature = 300
  ion_temperature = 300
  min_delta_KE = 50.0*kev
  min_hybrid_KE = 1.0*ev
  elastic_model = Davies
  rlm_1 = 1
  rlm_2 = 1

end:hybrid
```

use_hybrid - Logical flag to determine if we use the hybrid PIC loop. This must be set to T if the code is to run in hybrid mode. The default is F.

use_fields - Logical flag to switch on the hybrid field solver (default F). If T, the electric and magnetic fields will be updated assuming a return current is drawn from the solid background, otherwise fields will remain at their initial values.

use_scatter - Logical flag to switch on elastic scatter routines for incoming electrons. The default is F (no elastic scatter).

elastic_model - Choose the elastic scatter algorithm to use. The options are `Davies` or `Urban`. Davies is more accurate for small angle scatter and faster to run, while Urban is more appropriate if you are interested in rarer large-angle scatter events. The default is Davies.

use_collisions - Logical flag to switch on ionisation energy loss routines. This is the energy loss from incident electrons due to exciting the atomic electrons in the background solid. The default is F (no ionisation energy loss).

produce_delta_rays - In ionisation energy loss, incident electrons can lose a lot of energy to atomic electrons, exciting them into secondary high-energy electrons (termed δ-rays). Ionisation energy loss routines will always calculate the incident electron recoil due to δ-ray emission, but will only add them to the simulation window if this key is T. Otherwise, the δ-ray energy is used to raise the local background electron temperature. The default is F.

min_delta_energy - Minimum δ-ray total energy to be injected into the simulation [J]. produce_delta_rays must be T to inject any δ-rays at all. The default is $m_e c^2$ (all δ-rays added, very slow).

min_delta_KE - Sets the same parameter as min_delta_energy, but using the minimum kinetic energy instead [J]. Default is 0 (all δ-rays added, very slow).

min_hybrid_energy - Sets the minimum electron energy in allowed in the simulation window [J]. Any electrons below this energy are removed from the simulation, and their energy is dumped into a background electron temperature increase. Due to the heating aspect, this currently requires ionisation energy loss to be switched on in order to work. Default is $m_e c^2$ (only remove electrons at rest).

min_hybrid_KE - Sets the same parameter as min_hybrid_energy, but using the minimum kinetic energy instead [J]. Default is 0 (only remove electrons at rest).

use_ohmic_heating - Logical switch to use Ohmic heating. This involves a background electron temperature increase in each step, based on the local resistivity and hot electron current density. The default is F (no heating).

use_thomas_fermi - Logical switch to calculate the ionisation state of the background solid, using a fit to the Thomas-Fermi distribution. The default is F (no ionisation calculation), but this can be overwritten by routines which require ionisation like the reduced Lee-More resistivity model, or bremsstrahlung radiation with plasma screening switched on.

use_ion_temp - Logical switch to use the ion-electron thermal equilibration routines, which exchange heat between the two particle species. The default is F (no heat exchange).

electron_temperature - Initial background electron temperature distribution, which can vary in space (Kelvin). Default is 0 in all cells.

ion_temperature - Initial background ion temperature distribution, which can vary in space (Kelvin). Default is 0 in all cells.

rlm_1 - The λ_1 parameter used in the reduced Lee-More resistivity model. The default is 7 to match the Milchberg curve.

rlm_2 - The λ_2 parameter used in the reduced Lee-More resistivity model. The default is 3.5 to match the Milchberg curve.

5.2 Solid block

To specify the solid geometries in the hybrid routines, a special solid block has been developed. Solids can be positioned by specifying the ion number density over the simulation window. A solid may only represent one element, but compound materials like plastic can be built by spatially overlapping a carbon and a hydrogen solid. An example solid block is shown below for an Al solid positioned between $x = 0$ and $x = 10$ μm

```
begin:solid

  atomic_no = 13
  mass_no = 27
  I_ex = 166 * ev
  ni = if ((x gt 0) and (x lt 10), 6.022e28, 0)
  rad_len = 0.08897
  resistivity = rlm

end:solid
```

atomic_no - Atomic number of the solid (integer).

mass_no - Mass number of the solid.

I_ex - Mean excitation energy of the solid or compound solid.

ni - Ion number density of the solid.

rad_len - Radiation length of the solid or compound solid (only required if using Urban elastic scatter). The radiation length assigned to each cell will match the radiation length of the solid with the highest background electron number density in this cell at the time of initialisation.

resistivity - Resistivity model for electrons in this solid or compound solid. The resistivity model used will match the resistivity model of the solid with the highest background electron number density in this cell at the time of initialisation. Options are `rlm`, `Milchberg`, `plastic` or `vacuum`.

5.3 Hybrid laser block

When working with hybrid routines, it is useful to have a way to inject electrons which is more general than the Maxwellian injector already present in the code. The `hy_laser` block was designed to produce expected electron distributions based on laser parameters, such that the block shares many keys with a standard laser block (despite being a particle injector). This block is also capable of injecting particle beams and bunches, at a variety of angles using multiple models. For details on the mean energy, particle energy and angular distribution models, consult Section 3.3 and its various subsections. Here we restrict our discussion to the keys in the input deck. When discussing angles, we define θ as an angle to the boundary normal, and ϕ as an azimuthal rotation about the normal. An example `hy_laser` block is shown below:

```
begin:hy_laser

  boundary = x_min
  ppc = 5
  profile = gauss(y,0,5.0e-6)*gauss(z,0,5.0e-6)
  profile_min = 0.5

  t_profile = gauss(time, 730.0e-15, 340.0e-15)
  t_start = 0.0
  t_end = 1460.0e-15

  intensity = 3.1e20 * 1.0e4
  lambda = 1.0e-6
  efficiency = 0.03

  mean_energy = a0
  energy_dist = exp_weight
  angular_dist = uniform

  theta_max = 20.0/180.0*pi
  mean_mult = 10.0
  use_moore_max = T

  species = Electron

end:hy_laser
```

boundary - Boundary of particle injection, can be `x_min` or `x_max`, or higher dimensions in `EPOCH2D` and `EPOCH3D`.

ppc - If a cell injects particles this time step, this key specifies how many macro-particles to inject.

omega - Laser angular frequency. If using laser-based injection, specify this, frequency or wavelength. Can be a function of time.

frequency - Laser frequency. If using laser-based injection, specify this, angular frequency or wavelength. Can be a function of time.

wavelength - Laser wavelength. If using laser-based injection, specify this, frequency or angular frequency. Can be a function of time.

intensity - Peak cycle-averaged laser intensity.

efficiency - Conversion efficiency of laser energy into particle kinetic energy, for determining the weights of macro-particles

profile - Spatial profile of the laser pulse. Must be less than or equal to 1 at all points, otherwise laser intensity will exceed the peak value given by the `intensity` key.

profile_min - If the spatial profile dips below this number, no particles will be added in that cell.

t_profile - Temporal profile of the laser pulse. Must be less than or equal to 1 at all points, otherwise laser intensity will exceed the peak value given by the `intensity` key.

t_start - Time at which particles can start to be injected.

t_end - No particles will be injected after this time.

species - Particle species to populate with injected particles.

mean_energy - Model for calculating the mean injected particle energy. Options are: `a_0` ($a_0 m_e c^2$) or `Wilks` for laser-based mean energies, or `E_val` for a user-defined mean energy.

mean_E - User-defined mean total particle energy for the `E_val` mean energy model. Specify this or `mean_KE`.

mean_KE - User-defined mean particle kinetic energy for the `E_val` mean energy model. Specify this or `mean_E`.

energy_dist - Model for calculating the energy and weight distributions of injected macro-particles. All weights are uniform and calculated from laser parameters unless otherwise stated. The options are: `exp` (exponential energy distribution), `mono` (mono-energetic at mean energy), `top_hat` (uniform energy between two limits), `exp_weight` (uniform energies, exponential weights), `mono_weight` (mono-energetic with user-defined weight), and `mono_las_weight` (mono-energetic, but particle energies can be different from mean energy calculation).

mono_weight - User-defined particle weights for the `mono_weight` energy distribution model.

las_weight_KE - User-defined particle energy for the `mono_las_weight` energy distribution model. Useful if you don't want to inject particles with the same mean energy as that used to calculate the laser injection weights.

top_hat_L - Width of the uniform distribution for the `top_hat` energy distribution model. Particles are injected between ± this fraction of the mean kinetic energy.

mean_mult - Maximum energy modelled by the `exp_weight` distribution, expressed as a multiplication factor applied to the mean kinetic energy.

angular_dist - Model for calculating the angular distribution of injected macro-particles. Options are: `uniform` (isotropic), `cos` ($\propto \cos^n(\theta)$), `beam` (all particles same direction).

cos_n_power - The value of n in $\cos^n(\theta)$ for the `cos` angular distribution model.

theta_max - Maximum θ value for injected particles.

use_moore_max - Logical switch to use Moore scaling on the injected particle angle. This is an energy dependent maximum θ, and particles will be given angles up to the Moore angle or `theta_max`, whichever is lower.

use_sheng_dir - Logical switch to use the energy-dependent Sheng injection angle for oblique laser pulses.

sheng_angle - The angle the laser makes to the target normal, a parameter for the Sheng model.

theta - Global θ rotation of the injected particles after all angular distribution models have been applied.

phi - Global ϕ rotation of the injected particles after all angular distribution models have been applied. The meaning of $\phi = 0$ and the direction of increasing ϕ has different meanings on different boundaries. On boundaries $\{x_{\text{min}}, x_{\text{max}}, y_{\text{min}}, y_{\text{max}}, z_{\text{min}}, z_{\text{max}}\}$, the $\phi = 0$ line points to $\{z_{\text{min}}, z_{\text{min}}, z_{\text{min}}, z_{\text{min}}, x_{\text{max}}, x_{\text{max}}\}$, and the direction of increasing ϕ makes $\phi = \pi/2$ point towards $\{y_{\text{max}}, y_{\text{min}}, x_{\text{min}}, x_{\text{max}}, y_{\text{max}}, y_{\text{max}}\}$.

5.4 K-alpha block

The Kα radiation block follows a similar syntax to other radiation blocks like bremsstrahlung and non-linear Compton scatter. At present, only Kα emission from hot electrons traversing **copper** is supported by the code. To identify a solid as copper, set its atomic number to 29. A photon species must also be specified to populate with Kα macro-particles. An example is provided below, including both the Kα block, and a species block to assign generated Kα photons to. The line `identify:k_alpha_photon` identifies this species as the one to populate with Kα macro-particles.

```
begin:k_alpha

  use_k_alpha = T
  produce_photons = T
  photon_dynamics = F
  start_time = 0.0
  use_k_alpha_recoil = T
  min_photon_energy = 0.0
  auger_frac = 0.2
  photon_weight = 1.0
  sig_ka1_0 = 3.0
  sig_ka2_0 = 3.0

end:k_alpha


begin:species

  name = Ka_Photon
  npart = 0
  dump = T
  identify:k_alpha_photon

end:species
```

use_k_alpha - Determines whether to run Kα scripts or not. Default is F.

produce_photons - Logical flag to determine whether macro-photons are added to the Kα photon species. If this is F while use_k_alpha is T, then electrons will still recoil when triggering ionisation, but no Kα macro-photons will be created. The default is F.

photon_dynamics - Logical flag to determine whether Kα macro-photons move when they are created. If F, they will remain at the creation position. Note that this flag overrides the immobile flag in the species block. The default is F.

start_time - The user may specify a delay before the Kα routines switch on. This may speed up the code if electrons traverse a large gap before reaching a copper solid. The default is 0.

use_k_alpha_recoil - If T, this reduces the energy of incident hot electrons when they ionise a bound 1s electron in copper. The user may wish to disable this if running with ionisation energy loss, as this package already considers energy loss due to collisions with all bound electrons, including copper 1s. The default is T.

min_photon_energy - Determines the minimum photon energy which will be added to the Kα photon species. Photons with energy below this threshold can still cause hot electron recoil, but they will not be tracked by the code. The default is 0.

auger_frac - The fraction of copper 1s ionisation events which do **not** lead to Kα emission. The default is 0.

photon_weight - If this is not specified, Kα macro-photons will have the same weight as the incident electron. This key may be used to improve statistics by sampling more Kα macro-photons of reduced weight, through increasing the cross section of emission. For example, setting this to 0.1 would increase the number of macro-photons by a factor of 10, while decreasing the macro-photon weight by the same factor to conserve real particle number. Note there are limitations to this key, as a macro-electron may only trigger one ionisation event per time-step - setting this too high will numerically saturate the emission, causing an underestimate of Kα yield. The default is 1 (no weight correction).

sig_ka$i$_n - Kα emissions can be made to emit photons from a Gaussian distribution, centred on the Kα energies at the current cell temperature. The standard deviation for this Gaussian must be set by the user. Here, i refers to the Kα line, and can be either 1 or 2. It is assumed the standard deviation can be described by a polynomial fit $\sigma = \sum \sigma_n (T_e')^n$, where n may range from 0 to 4. The T_e' terms refer to the background copper temperature in eV, and this key sets the σ_n variables, assuming they are defined such that σ is also in eV. All these σ_n terms are set to 0 by default, so the code would have Kα line emissions.

5.5 Hybrid keys in existing blocks

Finally, some new hybrid functionality is present in existing input deck blocks. The remainder of this section will cover these new extensions.

5.5.1 Photo-electric attenuation

For the attenuation of photons in a copper target due to the photo-electric effect, the user must activate a key in the control block, and identify photon species which are to be attenuated. This only applies the correct attenuation to copper Kα photon energies (around 8 keV energy). An example is provided below:

```
begin:control

  nx = 100
  ny = 8
  nz = 8
  t_end = 300e-15
  x_min = 0
  x_max = 100e-6
  y_min = -4.0e-6
  y_max =  4.0e-6
  z_min = -4.0e-6
  z_max =  4.0e-6
  use_photo_electric = T

end:control


begin:species

  name = Ka_Photon
  npart = 0
  dump = T
  attenuate = T
  identify:k_alpha_photon

end:species
```

use_photo_electric - Control block variable. This is a logical switch to run photo-electric attenuation of photons. The default is F.

attenuate - Species block variable. This is a logical switch to identify a species as a photon which will undergo photo-electric attenuation when traversing copper. The default is F.

5.5.2 TNSA boundary

We have implemented a new type of boundary condition to mimic the behaviour of electrons refluxing in the sheath field. This involves allowing high energy electrons to escape, while reflecting lower energy electrons with some momentum loss and scatter. An example block with TNSA boundaries is given below:

```
begin:boundaries

  bc_x_min = tnsa
  bc_x_max = tnsa
  bc_y_min = reflect
  bc_y_max = reflect
  bc_z_min = reflect
  bc_z_max = reflect

  tnsa_escape_KE = 10e6 * ev
  tnsa_p_loss = 5.341e-22
  tnsa_scatter_deg = 20

```

```
end:boundaries
```

The TNSA keys are currently shared between all boundaries with TNSA boundary conditions. A breakdown of these new keys is provided here:

tnsa_escape_KE - TNSA boundaries are reflective, but particles above this kinetic energy will not be reflected and escape the simulation window.

tnsa_p_loss - Reflecting particles will have their momentum magnitude reduced by this amount on tnsa boundaries (or set to zero if their momentum is too low).

tnsa_scatter_angle - The θ angle of reflecting particles will be changed by a number drawn from a uniform distribution between $\pm$ the value of this key, in radians.

tnsa_scatter_deg - Same as tnsa_scatter_angle, but in degrees.

5.5.3 Hybrid output variables

The hybrid routines introduce a new particle variable which can be written to SDF files from the output block:

delta_optical_depth - The optical depths associated with emission of a δ-ray.

We also have 5 new field variables to output properties of the hybrid solid background:

hy_Te - Background electron temperature (Kelvin).

hy_Ti - Background ion temperature (Kelvin).

hy_ni - Background ion number density.

hy_ion_charge - Background average ion charge state (from Thomas-Fermi ionisation).

hy_resistivity - Resistivity of the background solids.

The optical depth and electron temperature are restart variables, and ion temperature is only a restart variable if it has been allocated in the code.

5.5.4 File injector

The injector block has been extended to now inject particles defined in files. Ensure each particle variable is given its own file, and particles are ordered by injection time. An example block is provided below:

```
begin:injector

  boundary = x_min
  species = Electron

  inject_from_file = T

  y_data = "../injectors/y.txt"
  z_data = "../injectors/z.txt"
  px_data = "../injectors/px.txt"
  w_data = "../injectors/w.txt"
  t_data = "../injectors/t.txt"
  id_data = "../injectors/id.txt"

end:injector
```

inject_from_file - Override normal injector block routines and inject particles from file.

`{...}_data` - The path to the file containing a list of variable values for injecting particles. In higher dimension `EPOCH` versions we can define the injection position with `x_data`, `y_data` and `z_data` if appropriate. Injection momentum is set using `px_data`, `py_data` and `pz_data` (assumed 0 if omitted). The weight of injected particles is given by the `w_data` file, the injection times are given by `t_data` and we can assign particle IDs to injected particles using `id_data`.

References

[1] D Woodbury, L Feder, V Shumakova, C Gollner, R Schwartz, B Miao, F Salehi, A Korolov, A Pugžlys, A Baltuška, et al. Laser wakefield acceleration with mid-ir laser pulses. *Optics letters*, 43(5):1131–1134, 2018.

[2] CP Ridgers, TG Blackburn, D Del Sorbo, LE Bradley, C Slade-Lowther, CD Baird, SPD Mangles, P McKenna, M Marklund, CD Murphy, et al. Signatures of quantum effects on radiation reaction in laser–electron-beam collisions. *Journal of Plasma Physics*, 83(5), 2017.

[3] J Vyskočil, O Klimo, and S Weber. *Plasma Phys. Contr. F.*, 60(5):054013, 2018.

[4] D Wu, XT He, W Yu, and S Fritzsche. *High Power Laser Sci.*, 6, 2018.

[5] F Wan, Chong Lv, M Jia, H Sang, and B Xie. *Eur. Phys. J. D.*, 71(9):236, 2017.

[6] DA Hammer and N Rostoker. Propagation of high current relativistic electron beams. *The Physics of Fluids*, 13(7):1831–1850, 1970.

[7] CI Moore, A Ting, SJ McNaught, J Qiu, HR Burris, and P Sprangle. A laser-accelerator injector based on laser ionization and ponderomotive acceleration of electrons. *Physical review letters*, 82(8):1688, 1999.

[8] Z-M Sheng, Yasuhiko Sentoku, Kunioki Mima, Jie Zhang, Wei Yu, and Juergen Meyer-ter Vehn. Angular distributions of fast electrons, ions, and bremsstrahlung x/γ-rays in intense laser interaction with solid targets. *Physical review letters*, 85(25):5340, 2000.

[9] JR Davies, AR Bell, MG Haines, and SM Guerin. Short-pulse high-intensity laser-generated fast electron transport into thick solid targets. *Physical Review E*, 56(6):7193, 1997.

[10] JR Davies. How wrong is collisional monte carlo modeling of fast electron transport in high-intensity laser-solid interactions? *Physical Review E*, 65(2):026407, 2002.

[11] Yim T Lee and RM More. An electron conductivity model for dense plasmas. *The Physics of fluids*, 27(5):1273–1286, 1984.

[12] RM More. Pressure ionization, resonances, and the continuity of bound and free states. In *Advances in atomic and molecular physics*, volume 21, pages 305–356. Elsevier, 1985.

[13] MJ Berger, M Inokuti, HH Anderson, H Bichsel, JA Dennis, D Powers, SM Seltzer, and JE Turner. Report 37. *Journal of the International Commission on Radiation Units and Measurements*, (2):NP–NP, 1984.

[14] Sea Agostinelli, John Allison, K al Amako, John Apostolakis, H Araujo, P Arce, M Asai, D Axen, S Banerjee, G 2 Barrand, et al. Geant4âĂŤa simula-

tion toolkit. *Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250–303, 2003.

[15] John Allison, Katsuya Amako, JEA Apostolakis, HAAH Araujo, P Arce Dubois, MAAM Asai, GABG Barrand, RACR Capra, SACS Chauvie, RACR Chytracek, et al. Geant4 developments and applications. *IEEE Transactions on nuclear science*, 53(1):270–278, 2006.

[16] J Allison, Katsuya Amako, John Apostolakis, Pedro Arce, M Asai, T Aso, E Bagli, A Bagulya, S Banerjee, G Barrand, et al. Recent developments in geant4. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 835:186–225, 2016.

[17] GEANT Collaboration et al. Physics reference manual. *Version: geant4*, 10(9), 2016.

[18] Harry Messel and David F Crawford. *Electron–Photon Shower Distribution Function: Tables for Lead, Copper and Air Absorbers*. Elsevier, 2013.

[19] László Urbán. A model for multiple scattering in geant4. Technical report, 2006.

[20] HW Lewis. Multiple scattering in an infinite medium. *Physical review*, 78(5):526, 1950.

[21] D Liljequist, M Ismail, F Salvat, R Mayol, and JD Martinez. Transport mean free path tabulated for the multiple elastic scattering of electrons and positrons at energiesâĽď 20 mev. *Journal of applied physics*, 68(7):3061–3065, 1990.

[22] Ricardo Mayol and Francesc Salvat. Total and transport cross sections for elastic scattering of electrons by atoms. *Atomic Data and Nuclear Data Tables*, 65(1):55–154, 1997.

[23] D Attwood, P Bell, S Bull, T McMahon, J Wilson, R Fernow, P Gruber, A Jamdagni, K Long, E McKigney, et al. The scattering of muons in low-z materials. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 251(1):41–55, 2006.

[24] AR Bell. New equations of state for medusa. Technical report, Science Research Council, 1980.

[25] Lyman Spitzer. *Physics of fully ionized gases*. Courier Corporation, 2 edition, 1962.

[26] AGR Thomas, M Sherlock, C Kuranz, CP Ridgers, and RP Drake. Hybrid vlasov–fokker–planck–maxwell simulations of fast electron transport and the time dependance of k-shell excitation in a mid-z metallic target. *New Journal of Physics*, 15(1):015017, 2013.

[27] Yong-Ki Kim, José Paulo Santos, and Fernando Parente. Extension of the binary-encounter-dipole model to relativistic incident electrons. *Physical Review A*, 62(5):052710, 2000.

[28] F Pérez, L Gremillet, A Decoster, M Drouin, and E Lefebvre. Improved modeling of relativistic collisions and collisional ionization in particle-in-cell codes. *Physics of Plasmas*, 19(8):083104, 2012.

[29] JP Desclaux. *At. Data Nucl. Data Tables*, 12(4):311–406, 1973.

[30] Thomas A Carlson, CW Nestor Jr, Neil Wasserman, and JD McDowell. *At. Data Nucl. Data Tables*, 2:63–99, 1970.

[31] F Pérez, L Gremillet, A Decoster, M Drouin, and E Lefebvre. Improved modeling of relativistic collisions and collisional ionization in particle-in-cell codes. *Physics of Plasmas*, 19(8):083104, 2012.

[32] DR Rusby, CD Armstrong, GG Scott, M King, P McKenna, and D Neely. Effect of rear surface fields on hot, refluxing and escaping electron populations via numerical simulations. *High Power Laser Science and Engineering*, 7, 2019.

[33] SC Wilks, WL Kruer, M Tabak, and AB Langdon. Absorption of ultra-intense laser pulses. *Physical review letters*, 69(9):1383, 1992.

[34] HM Milchberg, RR Freeman, SC Davey, and RM More. Resistivity of a simple metal from room temperature to 10 6 k. *Physical review letters*, 61(20):2364, 1988.

[35] J. R. Davies, A. R. Bell, and M. Tatarakis. Magnetic focusing and trapping of high-intensity laser-generated fast electrons at the rear of solid targets. *Phys. Rev. E*, 59:6032–6036, May 1999.

[36] O Gunnarsson, M Calandra, and JE Han. Colloquium: Saturation of electrical resistivity. *Reviews of Modern Physics*, 75(4):1085, 2003.

[37] Lyman Spitzer Jr and Richard Härm. Transport phenomena in a completely ionized gas. *Physical Review*, 89(5):977, 1953.

[38] SI Braginskii. Transport processes in a plasma. *Reviews of plasma physics*, 1, 1965.

[39] A Kuritsyn, M Yamada, S Gerhardt, Hantao Ji, R Kulsrud, and Y Ren. Measurements of the parallel and transverse spitzer resistivities during collisional magnetic reconnection. *Physics of plasmas*, 13(5):055703, 2006.

[40] AO Hanson, LH Lanzl, EM Lyman, and MB Scott. Measurement of multiple scattering of 15.7-mev electrons. *Phys. Rev.*, 84(4):634, 1951.

[41] Grant J Lockwood, Laurence E Ruggles, Glenn H Miller, and JA Halbleib. Calorimetric measurement of electron energy deposition in extended media. theory vs experiment. Technical report, Sandia Labs., Albuquerque, NM (USA), 1980.

[42] MJ Berger, JS Coursey, and MA Zucker. Estar, pstar and astar: Computer programs for calculating stopping-power and range tables for electrons, protons and α-particles (version 1.2. 2). *NIST, Gaithersburg*, 2000.

[43] Xianguan Long, Mantian Liu, Fuqing Ho, and Xiufeng Peng. Cross sections for k-shell ionization by electron impact. *Atomic data and nuclear data tables*, 45(2):353–366, 1990.

[44] Z An, TH Li, LM Wang, XY Xia, and ZM Luo. Correction of substrate effect in the measurement of 8–25-kev electron-impact k-shell ionization cross sections of cu and co elements. *Physical Review A*, 54(4):3067, 1996.

[45] DH Rester, WE Dance, and JH Derrickson. Thick target bremsstrahlung produced by electron bombardment of targets of be, sn, and au in the energy range 0.2–2.8 mev. *J. Appl. Phys.*, 41(6):2682–2692, 1970.

[46] RG Evans, EL Clark, RT Eagleton, AM Dunne, RD Edwards, WJ Garbett, TJ Goldsack, S James, CC Smith, BR Thomas, et al. Rapid heating of solid density material by a petawatt laser. *App. Phys. Lett.*, 86(19):191505, 2005.