Numerical Simulation of Thomson Scattering of an Electron from Multiple Beams

Arthur Campello

June 24, 2019

1 Introduction

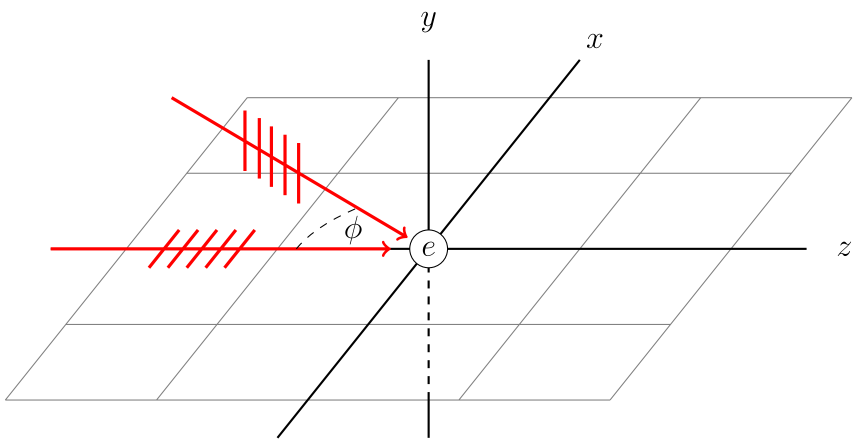
Although one can analytically determine the time-dependent radiated electric field arising from scattering of one beam with ease, finding this for the case of multiple non-parallel beams poses a harder problem. The complexity arises because the term $\mathbf{k} \cdot \mathbf{x}$ no longer remains zero. The non-parallel ploarized beam condition, thus entails solving differential equations of the form $\ddot{\mathbf{x}}(t) \propto cos\left[at + b\mathbf{x}(t)\right]$ instead of the far simpler $\ddot{\mathbf{x}}(t) \propto cos(at)$. For this reason, practically finding the radiated electric field requires a numerical simulation. In addition to the radiated intensity, the polarization and time-like behavior of the field matter greatly for many applications, particularly those in beam-line science. This report details a MATLAB-based program aimed at numerically simulating these behaviors. It includes a description of the assumptions made and the parameters involved, a physics-based justification for the mathematical problem it solves, numerical considerations made to ensure the accuracy of the mathematical solutions, and a detailed structure map of the functions and files involved in producing certain outputs.

2 Assumptions and Parameters

The program relies on an assumption that the electron is at rest at the retarded time, when the beams both reach the particle simultaneously. The user first inputs the propagation and polarization directions of a collection of n beams in to matrices $k \in \mathbb{R}^{3\times n}$ and $pol \in \mathbb{R}^{3\times n}$, respectively (the vectors are encoded in the columns of these matrices). This, of course, means that the ith column of k must be orthogonal to the ith column of pol, as checked by the program before execution. The diagram below illustrates an example setup given by

$$k = \begin{bmatrix} 0 & sin\phi \\ 0 & 0 \\ 1 & cos\phi \end{bmatrix}, \; pol = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

with the incoming beams as red arrows and the polarizations as lines perpendicular to the beams. The parameter $E \in \mathbb{R}^n$ encodes the relative electric fields of all n beams. Additionally, $\omega \in \mathbb{R}^n$ encodes the relative frequencies (proportional to their energies). Finally, $\eta \in \mathbb{R}^n$ represents the phase shift between oscillations of the electric fields of the beams relative to the first (meaning η_1 should be set to zero). The parameters k, pol, E, ω, and η are all provided by the user as inputs. These parameters can vary from experiment to experiment. For the version of this program detailed in this guide, all physical constants are set to one, meaning the results appear in relative quantities.



3 Physical Calculations

All outputs from the program stem from E_{rad}, the electric field radiated from the acceleration of the electron. The physical equations referenced in this section to clculate E_{rad} all come from Section 11.2 of Chapter 11 in the textbook Introduction to Electrodynamics, Third Edition, by David J. Griffiths. Given the overlapping and crossing input electric fields described in the previous section and letting the vector $\mathbf{x}$ represent the position of the electron relative to its starting position, one can write

$$m\ddot{\mathbf{x}} = eE_1 cos\left(\omega_1 t + \hat{z}\cdot\mathbf{x}\right)\hat{x} + eE_2 cos\left(\omega_2 t + [sin(\phi), 0, cos(\phi)]\cdot\mathbf{x} + \eta\right)\hat{y} \tag{1}$$

with assumptions $\mathbf{x}(t=0) = [0,0,0]$ and $\dot{\mathbf{x}}(t=0) = [0,0,0]$ and use a finite difference approximation to find $\mathbf{x}(t)$. At this stage, the numerical correction detailed in Section 4 is applied. From the corrected $\mathbf{x}(t)$, one can find $\ddot{\mathbf{x}}(t)$. From here, one can make use of radiation from an accelerating particle initially at rest to find $\mathbf{E_{rad}}$ at some position in space $\mathbf{r}$ and time t as

$$\mathbf{E_{rad}}(\mathbf{r},t) = \frac{\mu_0 e}{4\pi|\mathbf{r}|}\left[(\mathbf{r}\cdot\ddot{\mathbf{x}}(t))\mathbf{r} - \ddot{\mathbf{x}}(t)\right]. \tag{2}$$

While this function encodes all necessary information for the electric field, it becomes more practical if written as $\mathbf{E_{rad}}(\theta,\alpha,t)$, where θ and α represent angles of rotation about the x-axis and y-axis, respectively that transform $\hat{z}$ into $\frac{\mathbf{r}}{|\mathbf{r}|}$. This coordinate transformation is needed since the radiated field is observed in the far field at a fixed distance from the sample and as a function of the scattering angle. Since rotation in three-dimensions is non-commutative, I will write this as a rotation of α followed by one of θ. Given some $\mathbf{r}\in\mathbb{R}^{3\times1}$, one can find these angles as

$$R_x(\theta)R_y(\alpha)\frac{\mathbf{r}}{|\mathbf{r}|} = \hat{z} \implies \begin{bmatrix}1 & 0 & 0\\ 0 & cos\theta & -sin\theta\\ 0 & sin\theta & cos\theta\end{bmatrix}\begin{bmatrix}cos\alpha & 0 & sin\alpha\\ 0 & 1 & 0\\ -sin\alpha & 0 & cos\alpha\end{bmatrix}\frac{1}{|\mathbf{r}|}\begin{bmatrix}r_1\\ r_2\\ r_3\end{bmatrix} = \begin{bmatrix}0\\0\\1\end{bmatrix}. \tag{3}$$

From this, one can easily find the radiated intensity $I_{rad}(\theta,\alpha)$ by finding $|\mathbf{E_{rad}}(\theta,\alpha,t)|$ averaged over one period in time and squaring it; i.e.

$$I_{rad}(\theta,\alpha) = \frac{1}{\mu_0 c}\left\langle|\mathbf{E_{rad}}(\theta,\alpha,t)|^2\right\rangle_t \tag{4}$$

The scalar nature of the radiated intensity along some direction means the coordinate system does not alter it. This is not the case, however, for $\mathbf{E_{rad}}(\theta,\alpha,t)$ itself. Since the light radiated from the electron must travel in the radial direction, $\mathbf{E_{rad}}'$ in coordinates with respect to the radial vector serve to represent the field as "observed" from some angle. More specifically, for some $\mathbf{r}$, this coordinate transformation would arise from transforming $\mathbf{E_{rad}}'$ according to the angles θ and α found earlier. This transformation means $\frac{\mathbf{r}}{|\mathbf{r}|} = \hat{z}'$ and, thus, $\mathbf{E_{rad}}'_z = 0$. One can use θ and α along with $\mathbf{E_{rad}}$ (in the original coordinate system) to find $\mathbf{E_{rad}}'$ as

$$\mathbf{E_{rad}}' = R_y(-\alpha)R_x(-\theta)\mathbf{E_{rad}} \implies \begin{bmatrix} E'_{rad_x} \\ E'_{rad_y} \\ E'_{rad_z} \end{bmatrix} = \begin{bmatrix} cos\alpha & 0 & -sin\alpha \\ 0 & 1 & 0 \\ sin\alpha & 0 & cos\alpha \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & sin\theta \\ 0 & -sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} E_{rad_x} \\ E_{rad_y} \\ E_{rad_z} \end{bmatrix}. \tag{5}$$

the transformation constitutes the exact "reverse" transformation from that in equation 3 as the angles are negative and the order of the rotations is swapped. This concludes the description of how the output quantities are determined. The following section describes the aforementioned numerical correction and the last section outlines how the equations appear in the program and how the program presents the calculated outputs.

4 Numerical Considerations

The only numerically relevant consideration made by the user involves the choice of the finite difference resolution (often denoted as a variable "sampps" in the program, which specifies sampling per second). a low resolution will likely amplify numerical errors and produce an incorrect result, while an extremely high resolution will greatly limit the speed of the program. A recommended value for this falls in the range between 100 and 10,000.

For a successful finite difference approximation, the file "ManyBeamField.m" begins by estimating the first two time series values and, from this, it estimates the third as a difference of difference between points (mimicking the second derivative for acceleration). Because of this, only the third value of the time series (indexed at 3) and those beyond are modified. For this reason, and since the acceleration is based on a function of the iterate, terms of $(j - 2.5)$, where j is the iterate, are included, for an optimally accurate estimate of the acceleration rate for that iteration. This has been tested to return correct approximations for functions of the form $y''(t) = cos(at + b)$, for which an analytic solution exists to compare.

5 Program Structure

This section lists the files included in the program as well as their general roles. The files themselves contain comments that reference the equations as numbered in this report. Additionally, I provide a "map" of which files reference other files, where an arrow from one file to another means the second utilizes the first.

Descriptions of files:

- **rotxy.m** This function-file takes in two angles and a vector representing a spatial Cartesian coordinate in three dimensions, and a parameter for the multiplication order. If the order parameter is 0, It produces the vector arising from first rotating the the vector by the first given angle about the first axis (the x-axis) and then by the second angle about the second axis (the y-axis). Otherwise, the vector is produced by the two operations in the alternative order.
- **ManyBeamField.m** This function-file takes in parameters k, pol, ω, η, and E as defined in section 2, sampps as defined in section 4 and a pair of angles representing θ and α as defined in section three. From the frequencies, it determines the time for two periods of the $\mathbf{x}(t)$ functions and returns the calculated $\mathbf{E}'_{\mathbf{rad}}(\theta^*, \alpha^*, t)$, where θ^* and α^* are the fixed input angles. It makes use of equations 1, 2, 3, and 5. The function outputs this as a $3 \times n$ matrix where each column

represents the $\mathbf{E'_{rad}}$ elements (in the transformed coordinate system) ant the column index is the time.

- **ManyBeamFieldPlot.m** This file is not a function-file, but instead a script that allows a user to hard code the parameters used as inputs for the ManyBeamField.m plot and plots the three components of the electric field over time for two periods. As explained in section 3, the $\hat{z}'$ component should always be zero.
- **ManyBeamFieldMovie.m** This script allows a user to hard code all parameters used in ManyBeamField.m except the angle pair. The user also hard codes two ranges of angles for θ and α and a number of angles to sample (at evenly-spaced angles between the specified intervals). The program creates a meshgrid from this information and produces a movie of the x' and y' components of the outgoing field at each angle pair displayed in a grid (the z' component should, again, be zero). The movie spans two periods in time.
- **ManyBeamRadiation.m** This function file takes in the same parameters as ManyBeamField.m does, uses that function to produce an electric field, and makes use of equation 3 to return the the intensity radiated at the input angle pair.
- **ManyBeamRadiationSphere.m** This script allows a user to hard code all parameters used in ManyBeamField.m except the angle pair. The user also hard codes two ranges of angles for θ and α and a number of angles to sample (at evenly-spaced angles between the specified intervals). The program creates a meshgrid from this information and produces a heatmap of the relative radiated intensity for the specified angles. It artificially sets a corner value to zero to ensure the inclusion of no radiation in the color bar.

Here I include a map detailing which files listed above are used by which others:

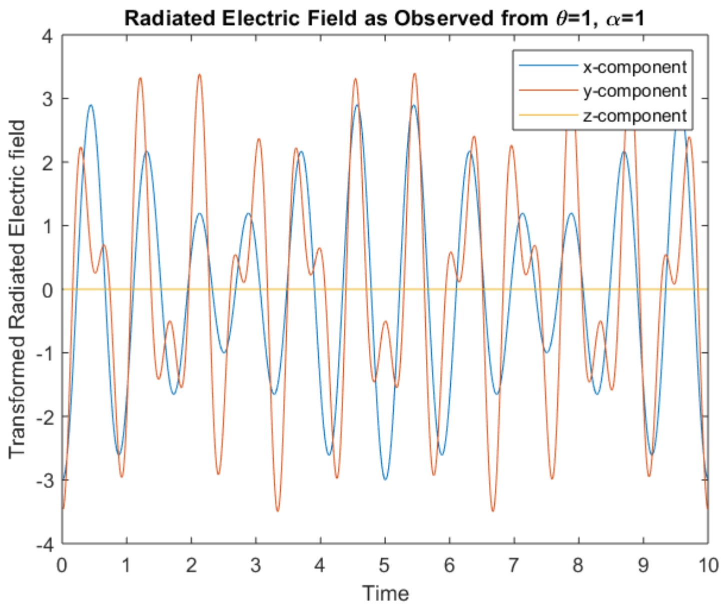
$$\begin{tikzcd}
& \boxed{\text{ManyBeamField.m}} \arrow[rr] \arrow[dd] \arrow[dr] & & \boxed{\text{ManyBeamRadiation.m}} \arrow[dd] \\
\boxed{\text{rotxy.m}} \arrow[ur] & & \boxed{\text{ManyBeamFieldMovie.m}} & \\
& \boxed{\text{ManyBeamFieldPlot.m.m}} & & \boxed{\text{ManyBeamRadiationSphere.m}}
\end{tikzcd}$$

6 Sample Outputs from Estimated Parameters

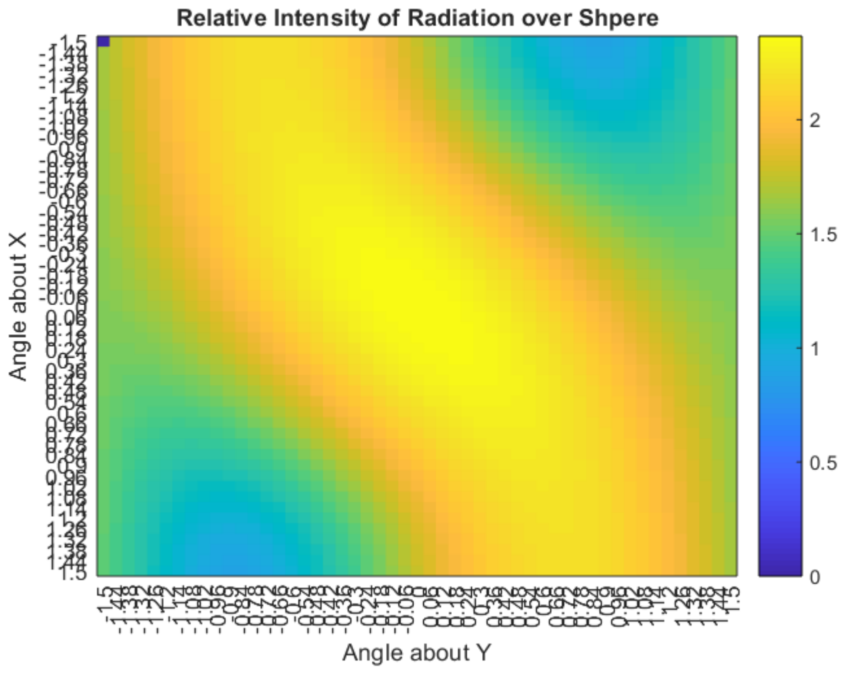
This section shows the outputs generated from different files for the parameters listed as follows:

$$k = \begin{bmatrix} 0 & 1/\sqrt{2} & 0 \\ 0 & 0 & 0 \\ 1 & 1/\sqrt{2} & 1 \end{bmatrix},\ pol = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix},\ E = [1, 1.5, 2],\ \omega = [1, 2.1, 1.2],\ \eta = [0, 0, 0]$$

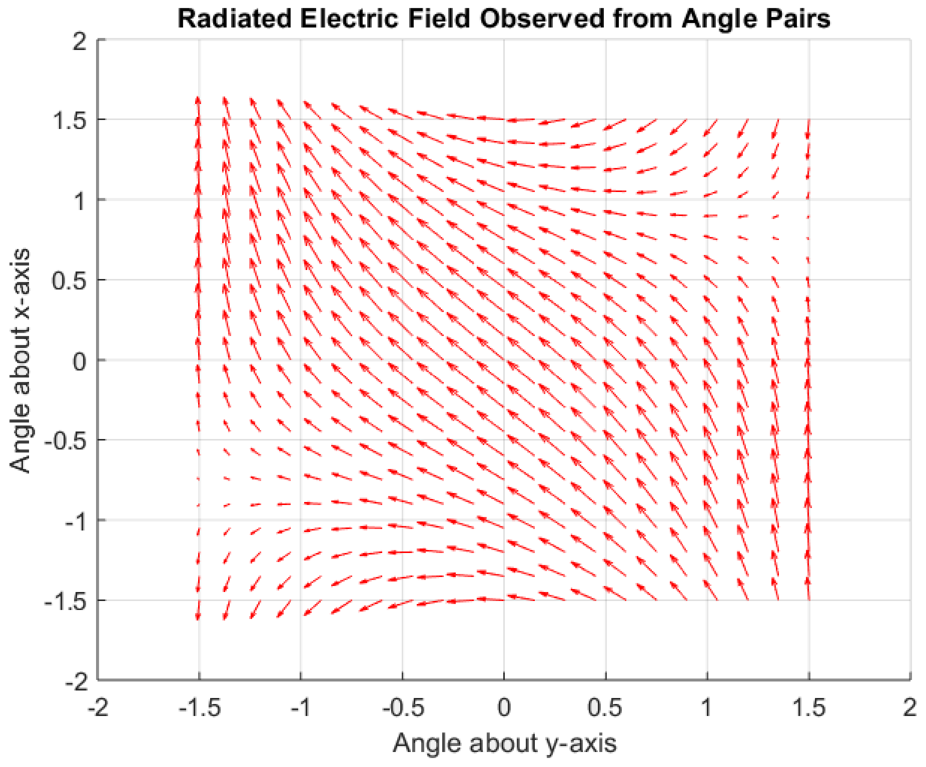
k and pol as in the example in section 2, $\omega = [1, 2.1]$, $\eta = [0, 0]$, and $E_1 = [1, 1]$. These parameters represent the best estimates of the true parameters at the time of the writing of this report. The first plot shows the output from the file ManyBeamFieldPlot.m calculated at $[\theta, \alpha] = [1, 1]$. This is shown below



The second plot, included below, shows the output of ManyBeamRadiationSphere.m sampled at θ and α both ranging from -1.5 to 1.5 at 51 intervals.



The final output from the program is that from the file ManyBeamFieldMovie.m. Since this outputs a movie that plays in time, only a snapshot can be included in this report. This is done in the plot below, showing the output at some point in time for the electric field components sampled at θ and α both ranging from -1.5 to 1.5 at 51 intervals.


Radiated Electric Field Observed from Angle Pairs
Angle about x-axis
Angle about y-axis
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2