

Efficient Simulation of Secondary Fluorescence via NIST DTSA-II Monte Carlo

Nicholas W. M. Ritchie
National Institute of Standards and Technology
100 Bureau Drive, Gaithersburg, MD 20899-8372
(301) 975-3929
nicholas.ritchie@nist.gov

Abstract

Secondary fluorescence, the final term in the familiar matrix correction triumvirate $Z-A-F$, is the most challenging for Monte Carlo models to simulate. In fact, only two implementations of Monte Carlo models commonly used to simulate electron probe X-ray spectra can calculate secondary fluorescence - PENEPMA (Llovet et al., 2016) and NIST DTSA-II¹ (DTSA-II is discussed herein). These two models share many physical models but there are some important differences in the way each implements X-ray emission including secondary fluorescence. PENEPMA is based on PENELOPE (Salvat et al., 2015), a general purpose software package for simulation of both relativistic and sub-relativistic electron / positron interactions with matter. On the other hand, NIST DTSA-II was designed exclusively for simulation of X-ray spectra generated by sub-relativistic electrons. NIST DTSA-II uses variance reduction techniques unsuited to general purpose code. These optimizations help NIST DTSA-II to be orders-of-magnitude more computationally efficient while retaining detector position sensitivity. Simulations execute in minutes rather than hours and can model differences that result from detector position. Both PENEPMA and NIST DTSA-II are capable of handling complex sample geometries and we will demonstrate that both are of similar accuracy when modeling experimental secondary fluorescence data from the literature.

¹ DTSA-II is a pseudo-acronym.

Introduction

Secondary fluorescence is subtle. Primary characteristic X-ray emission is generated when a core shell of an atom is ionized by energetic electron impact and the vacancy relaxes via the emission of an energetic photon. Primary continuum (Bremsstrahlung) emission is generated when energetic electrons decelerate in the electric field associated with an atom. Secondary fluorescence, on the other hand, is generated when a primary X-ray (either characteristic or continuum) photoionizes an electron from a core shell producing a vacancy which is filled by a valence electron accompanied by the emission of an energetic photon. Since the mean free path of energetic X-rays far exceeds the range of electrons in most materials, secondary fluorescence can come from materials with which the electron beam never interacts. Secondary fluorescence is most efficiently generated when a characteristic line is slightly more energetic than an ionization edge. However, the continuum (Bremsstrahlung) can also generate secondary fluorescence. Because the range of secondary fluorescence is large, materials mounted hundreds of microns or even millimeters from the primary excitation volume can contribute characteristic radiation. This can complicate trace element analysis when the trace element may be present somewhere in the surrounding matrix material.

Monte Carlo models of electron and X-ray transport are an excellent way to simulate primary and secondary fluorescence in samples with complex geometries. Currently only two Monte Carlo codes in common use in the microanalysis community actually simulate secondary fluorescence. These codes are PENEPMA (Llovet et al., 2016) based on PENELOPE (Salvat et al., 2015) and NIST DTSA-II. These two code bases take different approaches to modeling the physics of electron impact X-ray generation. To a large degree these differences can be attributed to the intended audience of the core product. PENELOPE is a very sophisticated general purpose Monte Carlo simulation of electron, positron and photon transport for a generic physics, nuclear, medical and, to a certain extent, as an afterthought, the microanalytical audience. PENELOPE has been specialized for the microanalysis audience through the code base PENEPMA. PENELOPE uses an electron transport model based on simulation of elastic, inelastic and Bremsstrahlung mechanism while DTSA-II uses a simpler continuous slowing down model. PENELOPE and PENEPMA are capable of simulating arbitrary geometries. PENELOPE is capable of simulating relativistic effects such as pair production. This generality is both a blessing and a curse. The generality means that such effects as secondary fluorescence (or even higher order fluorescence) emerge naturally in the model. However, this generality also means that the code does not make certain approximations or optimizations which are both reasonable and expeditious within the microanalytical framework. As a result, PENEPMA simulations require significant more CPU time.

Llovet acknowledged this limitation of PENEPMA and decided to address it (Llovet et al., 2012). They combine Monte Carlo modeling of the primary intensity using PENEPMA with an analytical model to calculate the secondary fluorescence near an interface. This approach mitigates the slow runtime performance of PENEPMA. The Monte Carlo step can be completed in less than an hour. The output of the Monte Carlo becomes input for a second program which fits the simulated intensities from PENEPMA to a series of equations. Finally, a third program is used to calculate the primary and secondary fluorescence from the standard and the specimen. This paper will compare our model with data from various experimental references, PENEPMA directly and the Llovet-2012 approach.

The X-ray generation model within NIST DTSA-II, the focus of this paper, is designed explicitly for simulating electron generated energy dispersive X-ray spectra. The electron transport model on which it is based (Ritchie, 2005) is sufficiently realistic and capable of sufficiently general sample geometries to allow modeling realistic microanalytical experiment geometries and sample structures. The model uses constructive solid geometry to build simple or complex sample shapes from a handful of basic shapes including spheres, cylinders, blocks and regions bounded by planes. The sum and difference of two or more simpler shapes can be combined into a more complex shape like a hemisphere or a dumbbell. Shapes can be embedded within other shapes. A region is defined by a shape and a material defined by elemental composition and density. A selection of common basic sample geometries are available through the application graphical user interface. Electron and X-ray transport is implemented using algorithms from the gaming industry for collision detection (Ericson, 2005). The most critical test is whether and where an electron or X-ray trajectory segment intersects with the surface of a shape. Electron transport was covered previously (Ritchie, 2005) and will not be the focus of this paper. Instead this paper will focus on the third-generation X-ray generation and transport model currently implemented in NIST DTSA-II.

Materials and Methods

Model overview

X-ray generation is an infrequent process. Typically, only a fraction of one percent of electron trajectories lead to ionization followed by subsequent X-ray emission. In a certain sense, this is fortunate because the energy loss associated with individual ionization events can largely be ignored when modeling electron trajectories. However, it does mean that if we model the emission of X-rays in the purest most physically realistic manner, the process is exceedingly inefficient. Hundreds of electron trajectories on average are needed to simulate a single X-ray emission. This is too costly so various approaches have been taken to optimize the problem. PENELOPE uses interaction forcing to enhance the relative number of X-rays generated. Interaction forcing involves systematically increasing the likelihood of certain events and then carefully compensating for the boost later in the calculation. For example, the ionization cross section can be increased by a couple of orders-of-magnitude to increase the likelihood of ionization and therefore also X-ray emission.

In contrast, NIST DTSA-II uses a model based on fractional X-ray emission (Bremsstrahlung and characteristic) along each trajectory segment (Heinrich, 1981). Each trajectory segment, which is nominally the path between sequential elastic scattering events, has a calculable probability of emitting an X-ray. A certain fraction of the emitted X-rays is emitted in the right direction to strike the detector and a fraction of these are absorbed before they reach the detector. The product of these probabilities is the probability that an X-ray will be emitted at some point on the trajectory segment and also strike the detector. This probability is interpreted as a fractional X-ray (a fraction of a photon) and its point of generation is assigned to a randomly selected position on the electron trajectory segment. Since each segment contributes fractional intensity to the resulting spectrum, the simulated spectrum contains hundreds of fractional X-ray emission events per trajectory. So instead of simulated x-ray event statistics limiting simulation precision, the factor limiting the reproducibility of the simulated spectra is variation in the number of backscatter events.

One shortcoming of the fractional X-ray model is that it does not naturally include secondary fluorescence since X-rays are tracked directly to the detector. Fortunately, secondary fluorescence can be added in a straightforward manner. The emitted fractional primary X-ray intensity (both Bremsstrahlung and characteristic) can be propagated from the source in a direction selected at random from a uniform distribution of directions until it is absorbed by

photoionization or it escapes the system. As with electron impact ionization, photoionization is followed by relaxation with the associated probabilities of emitting various characteristic X-rays dependent upon the absorbing element.

Mass absorption coefficients are implemented using the photoelectric component of Chantler's FFAST (Chantler et al., 2005) database of calculated mass absorption coefficients. The tabulated values are log-log interpolated. At these energies, there are three dominant processes, photoelectric absorption, Compton (inelastic) scattering and Thompson (elastic) scattering. The photoelectric component alone was used because we are only interested in processes that leave the atom ionized. Furthermore, the Thompson scattering term does not actually change the number of x-ray photons - only redirects them. Some photons heading towards the detector will be scattered away from the detector but an almost equivalent number would otherwise miss the detector but are scattering into the detector.

NIST DTSA-II uses various different techniques falling into the category of "variance reduction" to improve the efficiency of the calculation. Salvat *et al.* (Salvat et al., 2015) discusses various different approaches including interaction forcing, splitting and Russian roulette and range rejection. Interaction forcing involves scaling low probability events so they occur more frequently and then compensating later. NIST DTSA-II does not use interaction forcing. Splitting and Russian roulette are mechanisms to enhance the flux of radiation towards a region of interest and then compensate by discarding a random fraction of the subsequent output. While NIST DTSA-II does enhance the flux of radiation towards the detector, it instead deterministically scales the intensity to account for the bias that was introduced. NIST DTSA-II does implement a simple form of range rejection by eliminating both electrons and photons when they exit a volume containing the sample. The primary variance reduction techniques implemented by NIST DTSA-II are splitting and fractional emission. Fractional emission (Henrich, 1981) is a deterministic mechanism that tracks the probability of X-ray emission rather than stochastically generated discrete emissions. Discrete emissions happen only rarely even when the interaction is forced. Fractional emissions happen every trajectory segment. This leads to orders-of-magnitude more simulated photon trajectories (albeit fractional) to track to the detector. Furthermore, only that fraction of the emitted radiation that contributes to the measured signal is considered. This further increases the efficiency of the calculation. Finally, a small number of electron trajectories are simulated and the desired dose is estimated by scaling the simulated spectrum by the ratio of the desired measurement dose to the simulated dose.

The biggest difference between a direct simulation and the variance enhanced simulation performed by DTSA-II is the large difference in number of electron trajectories. A direct simulation would simulate exactly the same number of electrons as the equivalent real-world

measurement. For example, 1 nA·s is $6.25 \cdot 10^9$ electrons. A DTSA-II simulation, on the other hand, consists of thousands of simulated electron trajectories.

- In samples with complex geometries, the small number of simulated trajectories may not representatively sample the full geometry. Small features may be under or over sampled. For example, no simulated electron trajectories may intersect a very tiny feature. The error is likely to be small in magnitude but may be significant from an interpretational perspective.
- Backscatter events introduce variance into the emitted simulated intensity approximately proportional to $1 / \sqrt{N}$ where N is the number of electron trajectories. The variance actually seen in simulation is less than this naive estimate possibly because backscattered electrons contribute somewhat to the measured X-ray signal. For thousands or tens-of-thousands of simulated electrons, the variance due to backscatter swamps the variance introduced to simulate count statistics in the simulated spectrum.

Modeling X-ray Emission

Each electron trajectory segment is defined by a start point and an end point in three dimensional space and a kinetic energy. Each start/end point is either the point at which an electron enters a sample region or the point at which an elastic scattering event occurs. From the perspective of X-ray generation, it doesn't matter which of these two the start and end point represent. All that matters is the ionization cross-section for the various elements in the material at the specified kinetic energy and the length of the segment. This defines the likelihood of ionization for each shell in each element in the material. The ionization could have occurred with almost equal probability anywhere on the trajectory segment. A point is selected at random between the start and end and this point becomes the point at which the ionization is modeled. DTSA-II uses the analytical expression for the ionization cross section of Bote. (Bote *et al.*, 2008)

After an ionization, a cascade of Auger and/or X-ray emission processes will occur. A core shell ionization is energetically unstable and the electrons in the atom will attempt to redistribute themselves into a more energetically favorable state. The redistribution may involve a single less tightly bound electron falling into the core vacancy. Conservation of energy and angular momentum will dictate that an X-ray will be emitted, or, alternatively, the energy can be carried off by a second bound electron emitted as an Auger electron. The original core vacancy can shift into another shell within the same family via Coster-Kronig transition. So an L_{III} vacancy may transition into an L_{II} or L_I vacancy followed by the emission of either an L_{II} or L_I

Auger electron or X-ray. Furthermore, cascades of X-ray or Auger events are possible. For example, a K vacancy can result in a L_{III} electron filling the K shell leaving an L_{III} vacancy which can lead to a $L_{III}-M_V$ X-ray emission. The entire process is an exceedingly complicated process governed by rates for various different processes. Knowing the rates for each process, it is possible given an initial vacancy to calculate the probability of each of the different possible outcomes from a single initial ionization event. The sum of probabilities for events can exceed one when a single initial ionization can result in more than one Auger electron or X-ray emission event.

DTSA-II (and PENELOPE) uses the Evaluated Atomic Database Library (EADL) (Perkins et al., 1991) to model the core vacancy relaxation process. For each core shell vacancy, the program tabulates the probability of all X-ray emitting events. The generation of characteristic X-rays is assumed to be isotropic and photon polarization is not modeled.

Bremsstrahlung is implemented using Berger and Seltzer's (Seltzer & Berger, 1985, Seltzer & Berger, 1986) tabulated partial and total cross sections for Bremsstrahlung emission. The total cross section determines the fractional number of any energy Bremsstrahlung emission events due to each different element in the material. The partial cross section determines distribution of energies of the generated photons. The implementation is based on PENELOPE (Salvat et al., 2015, Acosta et al., 2002). The cross sections were tabulations provided to Francesc Salvat by Steven Seltzer one element per text file.

The probability of a Bremsstrahlung emission of any energy by electron interaction with element Z is

$$p_z = N_z \sigma(E, Z) \ell, \quad (1)$$

where N_z is the number of atoms of element Z per unit volume, ℓ is the electron path length, and

$$\sigma(E, Z) = \int_{E_{\min}}^E \frac{d\sigma(E, Z)}{dW} dW \quad (2)$$

is the total cross section for Bremsstrahlung production from the incident electron energy of E to a minimum X-ray energy of E_{\min} by element Z . The total probability of Bremsstrahlung emission by any element in the material is

$$p_B = \sum_Z p_Z$$

At each scattering event, the algorithm selects only one element at random weighted by p_Z to generate the full P_B intensity of Bremsstrahlung. The energy of the photon is determined by selecting a randomized energy weighted according to the partial cross-section for element Z

and electron energy $E \frac{d\sigma(E, Z)}{dW}$. In this model, energy is not conserved but because Bremsstrahlung photons are generated so infrequently, these rare deviations should not have a significant impact on the aggregate of simulated electron trajectories.

Both characteristic and continuum x-rays are handled twice to simulate primary and secondary emission. First, the event is tracked back to the detector by propagating it through intervening material. Second, a fraction of these events is emitted isotropically (Knop, 1970). Propagating the X-ray photons twice does not significantly overcount the event since only a tiny fraction of the isotropically emitted photons would have been otherwise detected. Each of these possibilities will be discussed independently.

Only a tiny fraction of the X-rays emitted from any point will reach the detector. The algorithm assumes this fraction is determined by the solid angle of the cone subtending the detector and the likelihood of X-ray absorption between the point of emission and the detector. The same collision detection algorithms used to track the path of an electron through sample regions are used to track the path of the X-ray from the point of emission, through each of the intervening materials to the detector. The absorption is

$$\frac{I}{I_0} = \prod_i \exp\left(-\left[\frac{\mu}{\rho}\right]_i \rho_i t_i\right), \quad (3)$$

where the index i reflects the various regions through which the X-ray must pass, $\left[\frac{\mu}{\rho}\right]_i$ is the mass absorption coefficient in the i -th region, ρ_i is the density in the i -th region and t_i is the path length in the i -th region. Absorption is not modeled as a stochastic event but rather as a deterministic reduction in the fractional intensity that reaches the detector.

In addition to the absorption term, there is an additional geometric term to account for the reduction in flux per unit area as the photon travels away from the source. If one photon passes

through a unit of area on a unit sphere, then $1 / r^2$ photons will pass through one unit of area on a sphere of radius r .

For the purpose of calculating the fractional intensity reaching the detector, the detector is assumed to face the point of emission and have a solid angle determined by the active area of the detector. Issues such as window transmission and detector efficiency are handled by the code that models the detector.

Bremsstrahlung, as shown in Figure 1, is not generated isotropically but rather tends to be enhanced in the direction of the electron travel. The distribution is cylindrically symmetric around the axis of the electron trajectory. In the elevation angle, typically parameterized as $\cos \theta$, the fully differential cross section for low energy electrons in a low density material of element Z is

$$\frac{d^2\sigma}{dWd(\cos\theta)} = \frac{d\sigma}{dW} p(Z, e, \kappa; \cos\theta) \quad (4)$$

where $\kappa \equiv W/E$ is the reduced photon energy, $\frac{d\sigma}{dW}$ is the energy loss differential cross section dependent only upon the energy of the photon and $p(Z, e, \kappa; \cos\theta)$ is the shape function which is a function of the atomic number, electron energy, the ratio of the Bremsstrahlung photon energy to the electron energy and the cosine of the emission angle.

$p(Z, e, \kappa; \cos\theta)$ is the shape function and DTSA-II provides two different implementations based on the work of Acosta and Kissel (Acosta et al., 2002, Kissel et al., 1983). The main difference is speed. The first implementation uses cubic interpolation and the second uses linear interpolation. The differences in the shape of the simulated continuum background as demonstrated in Figure 2 are trivial but the linear interpolation model is orders-of-magnitude faster.

Kissel *et al.* computed the angular distribution function $p(Z, e, \kappa; \cos\theta)$ expressed as the linear superposition of Legendre polynomials for

$$\begin{aligned} Z &\in \{2, 8, 13, 47, 79, 92\}, \\ e &\in \{1\text{keV}, 5\text{keV}, 10\text{keV}, 50\text{keV}, 100\text{keV}, 500\text{keV}\}, \text{ and} \\ k &\in \{0.0, 0.60, 0.80, 0.95\}. \end{aligned}$$

Acosta suggested parameterizing the distribution function using the ansatz

$$p(Z, e, \kappa; \cos \theta) = \left[A \frac{3}{8} \left[1 + \left(\frac{\cos \theta - \beta'}{1 - \beta' \cos \theta} \right)^2 \right] + (1 - A) \frac{4}{3} \left[1 - \left(\frac{\cos \theta - \beta'}{1 - \beta' \cos \theta} \right)^2 \right] \right] \frac{1 - \beta'^2}{(1 - \beta' \cos \theta)^2}, \quad (5)$$

where $\beta' = \beta(1 + B)$, $\beta = v/c$, v is the electron velocity at energy e and A and B are fit parameters dependent on Z , e and κ . Acosta *et al.* then suggest using cubic spline interpolation of the quantities $\ln(AZ\beta)$ and $B\beta$ to determine A and B at intermediate values of Z , β and κ . Cubic splines interpolation proved to be very slow compared to linear interpolation but tests demonstrated that both algorithms produce essentially equivalent spectra.

Secondary fluorescence is modeled by selecting a random direction from the point of generation in which to propagate the X-ray and computing the mean free path for photoionization.

$$\lambda = \frac{1}{\left[\frac{\mu}{\rho} \right]_M \rho_M}, \quad (6)$$

where

$\left[\frac{\mu}{\rho} \right]_M = \sum_Z C_Z \left[\frac{\mu}{\rho} \right]_Z$ and C_Z and $\left[\frac{\mu}{\rho} \right]_Z$ are the mass fraction and mass-absorption coefficient (photoabsorption) for element Z .

The radiation is then propagated in a random direction for a randomized distance. If the trajectory segment leaves the current region, then photon is moved to the intersection between the trajectory segment and the region boundary, a new mean free path for photoionization is computed and the X-ray continues along the initial ray. Eventually, the trajectory will either end in a sample region or the X-ray will leave the sample volume. If the X-ray photon terminates in the sample volume, a secondary fluorescence event is simulated.

The computation of secondary fluorescence is a relatively expensive calculation but typically represents a small effect (typically one percent or less). For efficiency, only a fraction \mathcal{F} (nominally 10%) of all X-ray generation events are propagated through the secondary fluorescence code. To compensate, the intensity associated with each secondary event is scaled by the inverse, $1/\mathcal{F}$. \mathcal{F} is chosen to balance the desire for precise secondary emission

intensities with the understanding that secondary emission only represents a small fraction of the total characteristic X-ray intensity. Increasing \mathcal{F} will increase the precision to a level that far exceeds the uncertainty due to uncertainty in primary emission intensity and is typically a waste of CPU time.

Compton scattering is also modeled. Since Compton scattering is not an important phenomenon in most electron excited X-ray spectra, the model used in DTSA-II is fairly simplistic. Compton scattering can be observed (Ritchie et al., 2011) when small samples emitting higher energy (> 10 keV) photons rest on low Z bulk substrates (like a particle on a carbon substrate). Compton scattering can also be observed in X-ray fluorescence spectra when the incident excitation X-rays scatter from the sample into the detector.

Compton scattering is an inelastic, incoherent scattering process where the energy of the outgoing X-ray is determined by the angle between the incident X-ray and the scattered photon. The total cross section for Compton (incoherent) scattering is calculated using McMaster's parameterization (McMaster et al., 1967).

Like Bremsstrahlung, Compton emission is not isotropic and only the unique direction that propagates to the detector is relevant. Thus the angle between the primary radiation and the outgoing photon determines the magnitude of the Compton energy shift and the relative intensity. The fractional energy shift equals

$$P(E, \theta) = \frac{1}{1 + \frac{E}{mc^2}(1 - \cos(\theta))}, \quad (7)$$

where mc^2 is the rest mass of an electron.

This angular dependence is handled by scaling the measured intensity according to the angular dependence in the Klein-Nishina differential cross section (Salvat et al., 2015),

$$\frac{d\sigma}{d\Omega} = \frac{1}{2} \alpha r_e \sin(\theta) \frac{[P(E, \theta) + P(E, \theta)^{-1} - \sin(\theta)^2]}{P(E, \theta)^2}. \quad (8)$$

Like secondary fluorescence, Compton scattering is a relatively expensive calculation for a small effect. To reduce the computation, only a fraction of all generated X-rays are propagate through the Compton scattering code, followed by a scaling to compensate.

Modeling the detector

The X-ray detector is modeled in a two-step process. First, as the electron trajectories are simulated, the incoming X-rays are scaled to account for the detector efficiency at the X-ray energy and then accumulated by energy as a histogram. Second, when a spectrum is desired, the raw accumulated events are scaled to account for the desired electron dose, the result is convolved with the detector resolution function and simulated Poisson count statistics are applied. The detector resolution function is modeled as a Gaussian the width of which depends on photon energy. Incomplete charge collection and other artifacts like escape and coincidence peaks are not modeled. The second step is “evaluated lazily” (i.e. on demand rather than incrementally) as the convolution and simulated statistics are expensive to calculate.

The efficiency of the detector is calculated from details provided by the user. The program can model many different kinds of X-ray windows by calculating their transparency using construction details and tables of mass absorption coefficients. Alternatively, the program can use tabulated window transparencies such as those provided by Moxtek¹ for their AP3 and AP5 windows as shown in Figure 3.

An X-ray window’s primary purpose is to protect the detector module from damage by environmental gases. An X-ray window must be air tight to ensure that the detector crystal remains under vacuum. X-ray windows may also serve to keep visible or near-visible light from the detector. Light can come from the electron emitter, the sample (cathodoluminescent), an IR camera illuminator, or the environment. Typically, X-ray windows come in two broad classes. Older windows were monolithic layers of a material like silicon nitride, diamond or beryllium. These materials are sufficiently strong that they can maintain the full pressure difference between vacuum and atmosphere. More modern windows tend to consist of a grid supporting a polycarbonate film or some recent ultra-thin windows have been constructed from etched Si_3N_4 . The film may be coated with a layer (often aluminum) to keep out light. The grid can be made of silicon (like a Moxtek AP3.X) or carbon (like a Moxtek AP5.X). Usually, the thickness of the grid is a few hundred microns and covers approximately 25% of the open area. At low X-ray energies, the grid is essentially opaque but at moderate and high energies, the grid becomes increasingly transparent.

¹ NIST Disclaimer: Any mention of commercial products is for information only; it does not imply recommendation or endorsement by NIST.

The detector crystal is modeled as a series of layers of absorbing material over a block of active detector material. Often the detector will have a conductive coating of nickel or gold. In addition, the detector is likely to have a “dead layer”, a layer of silicon that is not semiconducting. X-rays absorbed in this layer contribute less (“incomplete charge collection”) or not at all to the measured X-ray signal.

The detector crystal is modeled as a block of pure silicon. The thickness of the block determines the detection efficiency as a function of energy. The fraction of the X-rays that are absorbed in the block are detected. The fraction that passes right through are not. The effect of all of these layers is readily calculated using the Beer-Lambert law.

Finding the data (like the dead layer thickness, the anode layer thickness or the crystal thickness) necessary to model a detector can be a challenge. Detector vendors may either view this information as proprietary or they may simply not know it. Fortunately, for most purposes a “good enough” model can be constructed from available information and crude guesses. If a better model is required calibrated samples like the BAM TM-EDS001 (Procop & Hodoroaba, 2009), an engineered multilayer multi-element sample, can be used.

The incoming X-ray events are accumulated in energy bins. The width of the accumulator bins matches the channel width of the final spectrum. When a spectrum is requested, the accumulator bins are scaled and then convolved with a detector response function. The detector response function is a Gaussian function whose width varies with energy. The classic detector response function calculates the resolution from the Fano factor and the

electronic noise.
$$R(E) = \sqrt{n^2 + \frac{fE}{\epsilon}}$$
 where $f \gg 0.122$ is the Fano factor, $\epsilon \gg 3.64\text{eV}$ is the energy per electron-hole pair, and n is the electronic noise term. The peak is then modeled as a Gaussian function with this width. Finally, Poisson statistics are simulated. A random deviate is selected based on the nominal number of counts in the channel using a rejection algorithm (Press et al., 1992). This random deviate is added to the number of counts in the channel rounded to an integral value.

Results

Overview

In 2012, Llovet *et al.* did a thorough study of the secondary fluorescence generation accuracy of the program PENEPMA and the Llovet-2012 semi-analytical model. This paper will build on their work. They compared these models with secondary fluorescence data published by Bastin (Bastin *et al.*, 1983), Valovirta (Valovirta *et al.*, 2001), Wark (Wark & Watson, 2006) and Dalton & Lane (Dalton & Lane, 1996). These data sets consider both single elements fluoresced by single elements and compounds fluoresced by compounds. From Bastin *et al.*, they took Cu fluoresced by Co (dominated by characteristic secondary fluorescence) and Co fluoresced by Cu (exclusively continuum fluorescence.) From Valovirta, they took Fe fluoresced by Cu (dominated by characteristic secondary fluorescence) and Cu fluoresced by Fe (exclusively continuum fluorescence). From Wark, they took SiO₂ fluoresced by TiO₂ (dominated by characteristic secondary fluorescence). From Dalton & Lane, they took Ca K-L₃ fluorescence in a range of olivines near diopside. The plots herein are based on Llovet's data with the addition of light green triangles representing the DTSA-II modeled secondary fluorescence.

The simulations were performed using the scripting facility within DTSA-II. For each set of data, one bulk spectrum was computed to be used as a pure element standard for the fluoresced element. Then a series of spectra were simulated at various distances from the interface. Rather than use the simulated spectra directly, the raw X-ray intensity data for the characteristic, characteristic secondary and continuum secondary modes was output to a text file. The k-ratio, defined as the ratio of the intensity from the unknown divided by the intensity from a standard, was calculated from the text data. Alternatively, the same k-ratios could have been computed by fitting the simulated pure element standard to the simulated unknown but this produced less precise results particularly when the signal was small. An additional benefit of using the tabulated data is that the tabulated data separates out the three different contributions to the measured fluorescence whereas they are combined in the simulated spectrum as in the real world.

The script in Script 1 shows an example of a very basic script to calculate the secondary fluorescence at a range of offsets. The syntax is Python and it is executed using the Jython environment² embedded within DTSA-II. Separate scripts were written and executed for each plot or similar set of plots. The simulation takes about 5 minutes per point for 10,000 electrons.

² Jython (<http://www.jython.org/>)

Usually, approximately 1,000 electrons are sufficient to simulate a bulk spectrum with approximately 1 % repeatability, but because of the way DTSA-II prioritizes the secondary fluorescence calculation it is necessary to compute a factor of 10 more simulated electrons. Most of the time secondary fluorescence is a small signal and requires significant effort to compute. So, DTSA-II computes only secondary fluorescence on a fraction of the scattering events and scales the result. This typically produces the best trade off between accuracy and is a good default for most work except when the quantity in which we are most interested is the secondary fluorescence.

The electron trajectories are simulated in the lines `mc3.simulate(...)` (for the standard) and `mc3.interface(...)` (for the interface). The lines starting `fo8_4` and `diopside` use DTSA-II's material definition syntax to define the composition based on oxide fractions. The lines starting `params` configure special options that are passed to the simulation code. The other lines perform bookkeeping such as defining the beam energy, the number of electron trajectories to simulate and which transitions to accumulate.

Comparison with measurements

Each Figure 4 to 8 contains four (or, in one case, five) separate data sets. The violet boxes are measured data. The red line is PENEPMA Monte Carlo. The black line is Llovet's analytical model. The green triangles are NIST DTSA-II Monte Carlo. All three models show good agreement with the measured values. Figure 4 (a) shows the case of Co $K\alpha$ emission when the primary beam is located in copper near a cobalt interface. Secondary fluorescence due to the Cu K characteristic X-rays is the dominant process. Figure 4 (b) shows the case Cu $K\alpha$ emission when the primary beam is located in cobalt near a copper interface. Secondary fluorescence due to Bremsstrahlung generated by the cobalt is the dominant process. The measured data for these two plots comes from Bastin. Figure 5 (a) and (b) are similar except the element pair are iron and copper and the data comes from Valovirta.

The next two plots on Figure 6 are slightly more complex in as much as the materials are binary alloys of copper and cobalt. (a) The electron beam strikes 95.9 % Cu with 4.1 % Co and the other material is 95.9 % Co with 4.1 % Cu. The measured Co $K\alpha$ X-rays result from both primary emission and secondary emission. (b) The electron beam strikes 97.9 % Cu with 2.1 % Co and the other material is 95.9 % Co with 4.1 % Cu. The measured Co $K\alpha$ X-rays result from both primary emission and secondary emission.

The next plot Figure 7 represents a pair of stoichiometric compounds SiO_2 (quartz) and TiO_2 (rutile) which are of particular interest in the geological community as a geothermometer to determine the temperature at which the minerals formed. The data is from Wark & Watson.

The next Figure 8 (a)-(b) represent four different compositions of olivine adjacent to diopside. The signal is $\text{Ca K}\alpha$ which is present in the natural olivines in the very most trace quantities. The data is from Dalton & Lane.

Modeling the detector position

In the Figure 4 (b), there is an additional set of points (purple stars) that do not fit the measured data particularly well. Initially, all the DTSA-II secondary fluorescence calculations shown in the plots agreed nicely with the data except for one - $\text{Cu K}\alpha$ in Co. The DTSA-II simulation of this couple came in about a factor of two low; numerous potential explanations were investigated. In the end, the answer was as simple as it was significant. If the azimuthal angle on the X-ray detector is rotated by 180° , the apparent disparity is eliminated. This is demonstrated in Figure 10, which shows how reversing the position of the X-ray detector profoundly influences the measured secondary fluorescence signal. Clearly the quantity and distribution of generated secondary fluorescence has not changed. What has changed is the fraction of the generated fluorescence that reaches the detector. Secondary fluorescence is typically generated much deeper in the sample than characteristic fluorescence (see Figure 9) so is therefore much more susceptible to absorption by the material in the intervening region between the point of generation and detection. Closer examination of Figure 9 can explain the differences seen in Figure 10. As the position of the beam moves away from the interface, the absorption increases as the mean depth of generation increases.

This effect can be used to differentiate secondary fluorescence from a true compositional gradient. A true compositional gradient won't change significantly when the orientation of the sample is rotated by 180° relative to the detector but secondary fluorescence is likely to. Not all pairs of materials are likely to produce such an obvious signal difference and it is worthwhile to simulate both orientations to determine whether the rotation test provides useful information.

This example also demonstrates the power of DTSA-II's method of modeling the detector. In DTSA-II, the detector orientation is explicitly defined by both elevation and azimuthal angle and can be configured to match a given instrument. Since X-rays are propagated directly to the detector, the simulation efficiency does not depend upon solid angle. PENEPM can be configured this way and should be when the sample is not radially symmetric (like

interfaces). However, the more precisely the position of the detector is specified in PENEPMA, the more inefficient the simulation as the number of photons striking the detector scales as the solid angle of the simulated detector.

How many trajectories?

With Monte Carlo simulated X-ray spectra, there is always a question of how many electron trajectories to simulate to get the desired precision. The answer is quite complex as it depends upon many different simulation parameters and physical constants. The rule of thumb developed through experience with NIST DTSA-II is that for characteristic X-rays from a bulk material, approximately 1 000 electrons will produce a variance of approximately 1% in the simulated X-ray intensity. To improve the variance by a factor of 10, it is necessary to run approximately 100 times the number of electrons because the variance scales as the reciprocal square-root of the number of simulated electrons.

When the sample is other than bulk, the number of electrons should be scaled to account for the relative size of the feature generating the X-rays. For example, the number of electron trajectories for a film that is thinner than the excitation depth should be scaled by the ratio of the bulk excitation depth to the film thickness. The graphical user interface associated with NIST DTSA-II automatically applies these rules to scale the default number of electron trajectories to the sample volume. It also scales the number electrons to the desired probe dose although the reason might not be immediately obvious. In a spectrum generated without simulated count statistical variations, ripples are evident in the continuum. These ripples are artifacts of the model and the finite number of simulated electrons. They decrease in magnitude when more electrons are simulated. It is important that simulated count statistics dominate the ripples so when the simulated count statistics are improved by increasing the probe dose, the number of simulated electrons are increased. In DTSA-II's model, the simulated probe dose is not determined by the number of simulated electron trajectories but by a scale factor that depends upon the desired dose divided by the number of simulated electrons.

The best way to determine the optimal number of simulated electrons for the desired secondary fluorescence model precision is to observe the variance. As two examples, Cu $K\alpha$ in the Co-Cu interface (exclusively continuum secondary) and Co $K\alpha$ in the Cu-Co interface (primarily characteristic secondary) are considered. For each, couple 10 spectra were simulated at a distance of 10 μm from the interface using 1 000 simulated electrons each. In the first case, representing exclusively continuum secondary, the observed variance between iterations is 8 % and in the second case, in which the dominant contributor is characteristic secondary

fluorescence and there is a minor contribution from continuum secondary fluorescence, the variance is 11 %. The ten simulated spectra may be averaged together to produce an estimated effective variance of 2.4 % and 3.4 % respectively. Each simulation of 1,000 electrons took about 36 s on an Dell 9010 with an Intel i7-3990 with a single thread Passmark³ rating of 2069. The variance in the data plotted in Figures 4 – 8 is at approximately this level as each point represented 10 000 simulated electrons.

Conclusion

There are three models available that can calculate the secondary fluorescence from an interface. For a simple interface, the choice is a matter of preference as they all appear to do an adequate job of calculating secondary fluorescence. PENEPMA is the most computationally intensive. The actual timing depends upon many factors including the configuration parameters C1 and C2 and the interaction forcing (Pinard et al., 2010). Using the default parameters, PENEPMA takes 12 hours to simulate a particular bulk glass to a precision of 1% while DTSA-II takes about a minute for an equivalent precision- a difference of over 3 orders-of-magnitude on identical hardware. In practice, Llovet's mixed model and DTSA-II are of similar levels of computational intensity. DTSA-II can calculate individual offsets faster than Llovet's model however if a full curve over a range of more than approximately 10 offsets is required then Llovet's model may be more efficient.

Only PENEPMA and DTSA-II can compute the secondary fluorescence from alternative geometries. DTSA-II has a graphical user interface providing easy configuration of many different sample geometries and scripting capabilities for more complex geometries. DTSA-II also provides tools for quantifying and otherwise analyzing X-ray spectra. To the best of the author's knowledge, none of the other Monte Carlo tools for simulating X-ray spectra can handle secondary fluorescence.

Secondary fluorescence can be an important concern and having a variety of tools to compute secondary fluorescence from a variety of different materials and sample geometries will allow researchers to investigate this subtle source of this often hard to interpret X-ray signal.

³ see <https://www.cpubenchmark.net/cpu.php?cpu=Intel+Core+i7-3770+%40+3.40GHz>
[Downloaded: 24-Aug-2016]

References

- ACOSTA, E., LLOVET, X., COLEONI, E., SALVAT, F. & RIVEROS J., A. (1998), 'Monte Carlo simulation of x-ray emission by kilovolt electron bombardment', *Journal of Applied Physics* **83**, 6038--6049.
- BASTIN, G. F., VAN LOO, F. J. J., VOSTERS, P. J. C. & VROLLJK, J. W. G. A. (1983), 'A correction procedure for characteristic fluorescence encountered in microprobe analysis near phase boundaries', *Scanning* **5**(4), 172--183.
- BOTE, D. & SALVAT, F. (2008), 'Calculations of inner-shell ionization by electron impact with the distorted-wave and plane-wave Born approximations', *Physical Review A* **77**(4), 042701.
- CHANTLER, C. T., OLSEN, K., DRAGOSET, R. A., CHANG, J., KISHORE, A. R., KOTOCHIGOVA, S. A. & ZUCKER, D. S. (2005), 'X-Ray Form Factor, Attenuation and Scattering Tables', Technical report, National Institute of Standards and Technology, Gaithersburg, MD, Web Site:~<http://physics.nist.gov/ffast> Available: 2007, May 1.
- DALTON, J. A. & LANE, S. J. (1996), 'Electron microprobe analysis of Ca in olivine close to grain boundaries: the problem of secondary X-ray fluorescence', *American Mineralogist* **81**(1-2), 194--201.
- ERICSON, C. (2004), *Real-time collision detection*, CRC Press.
- GAMMA, E., HELM, R., JOHNSON, R. & VLISSIDES, J. (1995), *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Professional.
- HEINRICH, K. F. J. (1981), *Electron Beam X-Ray Microanalysis*, Von Nostrand Reinhold Company, New York.
- KISSEL, L., QUARLES, C. A. & PRATT, R. H. (1983), 'Shape functions for atomic-field bremsstrahlung from electrons of kinetic energy 1–500 keV on selected neutral atoms $1 \leq Z \leq 92$ ', *Atomic Data and Nuclear Data Tables* **28**(3), 381 - 460.
- KNOP, R. E. (1970), 'Algorithm 381: Random Vectors Uniform in Solid Angle', *Communications of the ACM* **13**(5), 326.

LLOVET, X.; PINARD, P. T.; DONOVAN, J. J. & SALVAT, F. (2012), 'Secondary fluorescence in electron probe microanalysis of material couples', *Journal of Physics D: Applied Physics* **45**(22), 225301.

LLOVET, X. & SALVAT, F. (2016), 'PENEPMA: A Monte Carlo programme for the simulation of X-ray emission in EPMA', *IOP Conference Series: Materials Science and Engineering* **109**(1), 012009.

MCMMASTER, W. H., KERR DEL GRANDE, N., MALLETT, J. H. & HUBBELL, J. H. (1969), 'Compilation of X-ray Cross Sections', Technical report, Lawrence Radiation Laboratory, University of California.

PERKINS, S. T., CULLEN, D. E., CHEN, M. H., RATHKOPF, J., SCOFIELD, J. & HUBBELL, J. H. (1991), 'Tables and graphs of atomic subshell and relaxation data derived from the LLNL Evaluated Atomic Data Library (EADL), Z= 1–100', Technical report, Lawrence Livermore National Lab., CA (United States).

PINARD, P. T., DEMERS, H., SALVAT, F. & GAUVIN, R. (2010), 'An API/GUI for Monte Carlo simulation of EPMA spectra using PENELOPE', *Microscopy and Microanalysis* **16**(S2), 280--281.

PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T., FLANNERY, B. P. & OTHERS (1992), *Numerical recipes in C*, Vol. 2, Cambridge university press Cambridge.

PROCOP, M. & HODOROABA, V.-D. (2009), 'A test material and a quick procedure for the Performance check of X-ray spectrometers attached to the SEM', *Microscopy and Microanalysis* **15**(S2), 1120--1121.

RITCHIE, N. W. M. (2005), 'A new Monte Carlo application for complex sample geometries', *Surface and Interface Analysis* **37**(11), AVS, Sci Technol Soc, Microbeam Anal Soc.

RITCHIE, N. W. M., NEWBURY, D. E. & LINDSTROM, A. P. (2011), 'Compton Scattering Artifacts in Electron Excited X-Ray Spectra Measured with a Silicon Drift Detector', *Microscopy and Microanalysis* **17**(6), 903-910.

SALVAT, F., FERNANDEZ-VAREA, J. M. & SEMPAU, J. (2015), 'PENELOPE-2014: A Code System for Monte Carlo Simulation of Electron and Photon Transport', Technical report, Issy-les-Moulineaux, France: OECD/NEA Data Bank.

SCOTT, V. D., LOVE, G. & REED, S. J. B. (1995), *Quantitative Electron-Probe Microanalysis*, Ellis Horwood.

SELTZER S., M. & BERGER M., J. (1985), 'Bremsstrahlung spectra from electron interactions with screened atomic nuclei and orbital electrons', *Nuclear Instruments & Methods in Physics Research, Section B: Beam Interactions with Materials and Atoms* **12**(1), 95--134.

SELTZER, S. M. & BERGER, M. J. (1986), 'Bremsstrahlung Energy-spectra From Electrons With Kinetic-energy 1 keV-10 GeV Incident On Screened Nuclei And Orbital Electrons Of Neutral Atoms With $Z = 1-100$ ', *Atomic Data and Nuclear Data Tables* **35**(3), 345-418.

SPRINGER, G. (1967), 'The correction for "continuous fluorescence" in electronprobe microanalysis', *Neues Jahrbuch für Mineralogie, Monatshefte* **106**, 241--256.

VALOVIRTA, E., ERLACH, S., LLOVET, X. & HEIKINHEIMO, E. (2001), EPMA of Metal-Metal Diffusion Couples at High Temperature, in 'EMAS 2001–7th European Workshop on Modern Developments and Applications in Microbeam Analysis'.

WARK, D. A. & WATSON, E. B. (2006), 'TitaniQ: a titanium-in-quartz geothermometer', *Contributions to Mineralogy and Petrology* **152**(6), 743--754.

Figure Captions

Figure 1: *Plots of the angular distribution of Bremsstrahlung radiation from iron at $k = 0.75$ as a function of electron energy.*

Figure 2: *Figures (a) and (b) demonstrate the difference between the Bremsstrahlung distribution for three different angular models in a challenging thin-film sample. Thin films accentuate the differences in Bremsstrahlung models because all the electrons are moving one direction - 130° away from the detector. The Acosta model is the most sophisticated and closest to reality. The isotropic is the least sophisticated and should be quite poor particularly at high beam energies when the Bremsstrahlung is significantly forward directed. The Nominal model is a simple interpolated model based on Acosta. It replaces an expensive cubic spline interpolation with a linear interpolation but as Figures (a), (b) and (c) demonstrate, it is almost as good as Acosta while executing many times as fast.*

Figure 3: *Moxtek AP3 and AP5 window transparencies (source: Moxtek, Inc.)*

Figure 4: *These images based on Llovet's Figure 6 compare Co $K\alpha$ (a) and Cu $K\alpha$ (b) secondary fluorescence measured in Cu (a) and Co (b) with three models, Llovet (2012) model, PENEPMA and NIST DTSA-II Monte Carlo. The vertical axis represents the k -ratio relative to pure Co or pure Cu. The horizontal axis represents the position of the incident electron beam relative to the interface. (b) also contains an additional set of points (pink stars) which correspond to repositioning the detector a 180° rotation relative the green dots. The position of the detector cannot influence the generation of X-rays but the relative absorption of Cu $K\alpha$ by Cu ($[m/r] = 47.9 \text{ cm}^2 / \text{g} = 4.7 \text{ m}^2 / \text{kg}$) and Cu $K\alpha$ by Co ($[m/r] = 322 \text{ cm}^2 / \text{g} = 32.2 \text{ m}^2 / \text{kg}$) suggests that more Cu $K\alpha$ X-rays are going to be measured if the detector is located on the Cu side rather than the Co side.*

Figure 5: *These images, based on Llovet's Figure 7, compare Fe K α (a) and Cu K α (b) secondary fluorescence in measured in Cu (a) and Fe (b) with three models, Llovet (2012) model, PENEPMA and NIST DTSA-II Monte Carlo. The vertical axis represents the k-ratio relative to pure Fe or pure Cu. The horizontal axis represents the position of the incident electron beam relative to the interface.*

Figure 6: *These images, based on Llovet's Figure 8, compare Co K α (a) and (b) secondary fluorescence in measured in Cu(4.1 % Co) (a) and Cu(2.1 % Co)(b) with three models, Llovet (2012) model, PENEPMA and NIST DTSA-II Monte Carlo. The vertical axis represents the k-ratio relative to pure Co. The horizontal axis represents the position of the incident electron beam relative to the interface.*

Figure 7: *This image, based on Llovet's Figure 9, compares Ti K α measured in SiO₂ adjacent to TiO₂ with three models, Llovet (2012) model, PENEPMA and NIST DTSA-II Monte Carlo. The vertical axis represents the k-ratio relative to pure Ti. The horizontal axis represents the position of the incident electron beam relative to the interface.*

Figure 8: (a) - (d) *Ca K α secondary fluorescence from four different compositions of olivine next to diopside. The data is taken from Dalton-Lane (1996).*

Figure 9: *These two images show how the distribution of measured Cu K-L3 X-rays differs depending upon where the detector is located. The tiny orange dot in the center-top of each image represents the locus from which the primary cobalt emission (characteristic and Bremsstrahlung) is being generated. When the detector is located to the right (above the copper), the generated X-rays escape through the copper to the detector while when the detector is located to the left (above the cobalt), most of the generated X-rays must first pass through the copper and then through the cobalt. Cobalt is seven times more opaque to Cu K-L3 X-rays than copper so many fewer X-rays reach a detector mounted on the cobalt side. This can be readily visualized as a diagonal line in the emitted X-rays below which X-rays are greatly attenuated.*

Figure 10: *The position of the detector relative to the orientation of the interface can make a significant difference to the quantity of detected X-rays. The "Correct" location has the detector located on the Cu side of the Cu-Co interface and absorption is less significant. The "Incorrect" location has the detector on the Co side. For Cu K α , Co has a mass absorption coefficient that is roughly seven times the Cu mass absorption coefficient.*

Appendix A: Code Structure

Monte Carlo simulations within NIST DTSA-II can either be performed through the scripting interface or through the graphical user interface (GUI). The GUI is more suited to one-off simulations of common sample geometries including particles, fibers and interfaces. The scripting interface is more flexible and is better suited to iterating through a range of parameters and custom sample geometries. To use the scripting interface, it can be useful to know how the classes implementing the Monte Carlo fit together.

Both modes take advantage of the X-ray detector definitions created through the GUI. Detectors can be created and edited using the *Preferences* dialog accessed through the *File -> Preferences* main menu item. The user may define as many detectors as desired. Detectors are defined in terms of certain fixed physical attributes like the window type, dead layer thickness, take-off angle and azimuthal angle and calibrated aspects like the resolution and energy calibration. The `listDetectors()` function provides access to a complete list of the available detectors and aliases to referred to them in scripts.

The GUI offers 12 different shape models - bulk, thin film, sphere, cube, inclusion, interface, pyramid, cylinder on side, cylinder on end, hemispherical cap, block and equilateral prism. Most of these models can be implemented on a bulk substrate or free-floating. While all of these models show interesting secondary fluorescence effects, the thin film, cube, interface, cylinder on end and block can fully contain the electron beam so that the only mechanism for X-ray production in the substrate is secondary fluorescence. The other models allow electrons to scatter out of the shape into the substrate. Associated with each shape (except bulk) are size parameters. The shapes that are asymmetric about the z-axis can also be rotated about this axis. The user can select instrument parameters like dose, beam energy and whether the beam is a point beam or whether the beam is rastered over the sample area. Additional simulation options allow the user to turn on and off the four different modes of X-ray generation: characteristic, continuum (Bremsstrahlung), characteristic secondary and continuum secondary. The user may select to apply simulated count statistics as appropriate for the dose.

The scripting interface can script the Java classes directly or there is a Python wrapper in the import file `mcSimulate3.py` that facilitates a handful of common usages. This import file was used in the example Script 1. `mc3.simulate(...)` (where `mc3` indicates a function imported from `mcSimulate3` as shown in the example script) simulates a spectrum from a bulk material much like the pre-define function `simulate(...)` that is always available through the scripting interface. The functions `mc3.block(...)`, `mc3.multiblock(...)`, `mc3.sphere(...)`, `mc3.coatedSphere(...)`,

`mc3.multiFilm(...)`, `mc3.embeddedSphere(...)`,
`mc3.embeddedCylinder(...)`, `mc3.interface(...)`,
`mc3.crenellated(...)`, `mc3.verticalLayers(...)`,
`mc3.horizontalLayers(...)` and `mc3.twoParticles(...)` implement different sample geometries. The syntax for each of these can be accessed within the scripting environment using `help(mc3.functionName)` where `functionName` is one of the above listed functions.

Additional shapes can be implemented in a Python function that can be handed to the function `mc3.base(...)` which will connect all the necessary detector sinks, run the simulation and return the results. There are many examples within the `mcSimulate3.py` source file. Additional functions in `mcSimulate3` will calculate ZAF factors or backscatter fractions using the Monte Carlo. The entire electron transport code within DTSA-II is largely taken line-for-line from the earlier NISTMonte (Ritchie, 2005). NISTMonte and DTSA-II use an event-based “observer” design pattern (Gamma et al., 1994) to build connections between various pieces of the code. Certain classes served as sources of event objects that represent things like scattering events, end-of-trajectory events, exiting a material event or X-ray emission event. Other classes connected to these source classes and served as sinks for the source’s events. The sinks perform special actions based on the data they received from the event source object. For example, one sink class `BackscatterStats` (all class names are within the `gov.nist.microanalysis.NISTMonte` package) implements a backscatter detector by counting the number of trajectory start events and the number of backscatter events and calculates the ratio to determine the backscatter fraction. More sophisticated classes serve as both sinks for some event types and sources for others. An X-ray generation class (like `Gen3.CharacteristicXRayGeneration3` or `Gen3.BremsstrahlungXRayGeneration3`) listens for elastic scattering events and then acts as a source for X-ray emission events. Other detector classes can implement sinks that act as energy dispersive X-ray detector models or as accumulators for $\phi(\rho z)$ curves.

The advantage of the source/sink model is that you only need to attach the sinks you need to make the calculation you desire and new kinds of sinks can be developed to add new physics, new accumulators or new detectors without concerns about disrupting pre-existing code. Since many sinks can be attached to a single source, you can compute as many or as few sinks as necessary. For example, if you don’t want to model continuum, you simply don’t attach a `BremsstrahlungXRayGeneration3` source/sink.

The classes implementing X-ray generation and propagation are summarized in Table 1. The X-ray related classes tend to be both sources and sinks. The classes implementing electron and X-ray detectors are listed in Table 2. These classes tend to be sinks for electron scattering events or X-ray generation events.

The class `MonteCarloSS` implements electron transport through arbitrary sample geometries. It generates events each time an elastic scattering event occurs (`ScatterEvent`), each time the electron passes between regions (`NonScatterEvent`) or each time the electron transitions to a point outside a sample region (`BackscatterEvent`.) It also implements bookkeeping events when a fresh electron trajectory starts (`TrajectoryStartEvent`), when the electron trajectory is terminated (`TrajectoryEndEvent`) and a handful of other events itemized in the class. Most of the X-ray generation classes look for `ScatterEvent` (electron scatters from atom) and `NonScatterEvent` (typically transitioning from one material to another) and generate X-ray events (`XRayGeneration`) based on the electron energy, the previous electron trajectory segment and the material in the current region.

The `CharacteristicXRayGeneration3` class computes the probability that a core-shell ionization followed by X-ray emissions will occur on the most recent electron trajectory segment. The `CharacteristicXRayGeneration3` class acts as a source for `XRayGeneration` events with the associated event data consisting of a set of probabilities for characteristic X-ray events. Each `ScatterEvent` / `NonScatterEvent` will emit a set of probabilities for all energetically allowed characteristic X-ray transitions for all elements in the current region's material. This may be tens of different fractional X-ray events for each element in the material. The relative shape of the characteristic line families is influenced by the ionization cross-section for each shell and the valence electron-core hole transition probabilities.

Typically, the `CharacteristicXRayGeneration3` class serves as a source for the `XRayTransport` and `FluorescenceXRayGeneration3` classes.

`BremsstrahlungXRayGeneration3` acts as a sink for scattering events and as a source for `Bremsstrahlung` X-ray emission events.

For each X-ray generation event `XRayTransport3` performs two actions. The photon is projected to the detector (taking into account the relative solid angle of detection) and a randomly-selected fraction of the photons are projected in a random direction to simulate secondary fluorescence.

In addition to handling the absorption of X-ray intensity, `XRayTransport3` as handles the anisotropy of Bremsstrahlung emission and Compton scattering events. Placing this logic in `XRayTransport3` rather than `BremsstrahlungXRayGeneration3` and `ComptonXRayGeneration3` might seem to be inelegant but is necessary as neither `BremsstrahlungXRayGeneration3` nor `ComptonXRayGeneration3` knows where the detector is located. Information travels from the source to the sink but not the other direction. The direction of the outgoing photon is known to `XRayTransport3` and to `FluorescenceXRayGeneration3`. Typically, other Monte Carlo simulations randomly select a propagation direction distributed according to $p(Z, e, \kappa; \cos \theta)$, the distribution of Bremsstrahlung emission as a function of Z , atomic number, e , incident electron energy, fractional Bremsstrahlung x-ray energy and angle θ . Instead DTSA-II forces a propagation direction towards the detector or selected at random to simulate secondary fluorescence. The non-isotropic distribution of radiation is accounted for by scaling the fractional intensity of the emitted radiation by the value of the shape function at the appropriate $\cos \theta$, where θ is simply the angle between the electron trajectory segment and the emitted photon.

`FluorescenceXRayGeneration3` is similar to `XRayTransport3` in as much as it observes the production of X-ray photons. `XRayGeneration3` propagates the photon to the detector while `FluorescenceXRayGeneration3` propagates the photon in a random direction until it is photoabsorbed by surrounding material or exits the model. Typically, the `FluorescenceXRayGeneration3` class acts as a source for the `XRayTransport3` class which now serves to propagate the secondary emission photon to the detector.

Tables

Table 1

Table 1: *The main Java classes implementing electron transport, X-ray generation and X-ray transport.*

<i>Class</i>	<i>Description</i>	<i>Sink for</i>	<i>Source for</i>
MonteCarloSS	This class implements the electron transport code.	Nothing	Scattering events
BaseXRayGeneration3	The base class for all algorithms that emit simulated X-ray photons. It's primary purpose is to keep track of the energy, emission point, intensity and transition (when relevant) of a list of events. It also manages event listeners for the derived classes.	Scattering events in MonteCarloSS	Nothing
CharacteristicXRayGeneration3	Extends BaseXRayGeneration3 to simulate characteristic X-ray emission by simulating ionization followed by X-ray emission.	Scattering events in MonteCarloSS	X-ray generation events
BremsstrahlungXRayGeneration3	Extends BaseXRayGeneration3 to simulate Bremsstrahlung X-ray emission.	Scattering events in MonteCarloSS	X-ray generation events

FluorescenceXRayEmission3	Extends BaseXRayGeneration3 to track isotropically emitted X-rays through photoabsorption and re-emission of secondary fluorescence.	X-ray emission events in a class derived from BaseXRayGeneration3	X-ray generation events
ComptonXRayEmission3	Extends BaseXRayGeneration3 to initiate Compton scattered X-rays.	X-ray emission events in a class derived from BaseXRayGeneration3	X-ray generation events
XRayTransport3	Extends BaseXRayGeneration3 to tracks trajectories of X-rays. Implements modeling of absorption between start and end point. This class also handles the non-isotropic character of the generation of Bremsstrahlung and Compton scattering photons.	X-ray emission events in a class derived from BaseXRayGeneration3	X-ray generation events

Table 2

Table 2: *The main Java classes implementing event sinks for X-ray and electron related events. These classes typically provide a mechanism to tabulate and report various characteristics of electron and X-ray processes.*

<i>Class</i>	<i>Description</i>	<i>Sink for</i>
AnnularDetector	This class implements a annular shaped detector which responds to electron trajectories	ScatterEvent, NonScatterEvent and BackscatterEvent

	passing through in one direction.	
BackscatterStats	Accumulates statistics such as energy, azimuthal and elevation angle for electrons that backscatter or forward scatter from the sample.	BackscatterEvent
DiagnosticListener	This class provides a mechanism to dump the electron trajectories to a tab-separated text file. Each line represents a step. Most steps represent scattering but some will represent boundary crossing.	All electron trajectory related events.
EnergyLossListener	This class is designed to listen for electron scattering events and to tabulate the energy loss as a function of position in the sample. The listener voxelates a region of space.	ScatterEvent and NonScatterEvent
GridDetector	A square electron detector divided into a grid like a chess board. The detector is designed to record the position of passage of electrons from high Z to lower Z through a Z-plane.	BackscatterEvent
ScatterStats	Collects statistics on the electron trajectory including the total trajectory path length and number of scattering events.	ScatterEvent and BackscatterEvent

Efficient Simulation of Secondary Fluorescence via Monte Carlo

TimeListener	Times individual electron trajectories and the entire simulation.	TrajectoryStartEvent and TrajectoryEndEvent
TrajectoryImage	Creates a 2D projection image of the X-ray trajectories.	ScatterEvent, NonScatterEvent and BackscatterEvent
TrajectoryVRML	Creates a Virtual Reality Markup Language file containing visualizations of the sample and the electron trajectories.	ScatterEvent, NonScatterEvent and BackscatterEvent
EmissionImage3	Creates a 2D heat map image showing the density of detected X-ray emission in profile.	XRayGeneration
PhiRhoZ3	Accumulates and summarizes the generation and emission data necessary to generate a $\phi(\rho z)$ curve.	XRayGeneration
VoxelatedDetector	The voxelated detector divides a region of space into voxels and counts the number of X-ray events that occur in each voxel.	XRayGeneration
XRayAccumulator3	Accumulates the generated and emitted intensities for a specified set of characteristic lines.	XRayGeneration
EDSDetector	Simulates an EDS detector. Produces a simulated spectrum as output.	XRayGeneration

Script 1

```

import dtsa2.mcSimulate3 as mc3

# define extra parameters to output images and tabulated X-ray intensity data
xrts = ( transition("Ca K-L3"), transition("Mg K-L3") )
params = mc3.configureEmissionImages( xrts, 100.0e-6, size=1024)
params.update(mc3.configureXRayAccumulators( xrts, charAccum=True,
charFluorAccum=True, bremFluorAccum=True))

# Simulation parameters
e0 = 15
nTraj = 10000

# simulate the Ca standard
mc3.simulate(material("Ca",4.5),d1, e0, nTraj=traj, dose=600.0, sf=True,
bf=True, xtraParams = params)

# Define the olivine
denFayalite = 4.329
denForserite = 3.3
fo8_4 =
material("0.0336*MgO+0.2977*SiO2+0.00028*CaO+0.0106*MnO+0.6556*FeO",denFayali
te*0.916+0.084*denForserite)
fo8_4.setName("Fo8.4")

# Define diopside
diopside =
material("0.0020*Na2O+0.1589*MgO+0.0477*Al2O3+0.5117*SiO2+0.2511*CaO+0.0029*T
iO2+0.0279*FeO",3.278)
diopside.setName("Diopside")

# Simulate the interface
for offset in range(0, 80, 2):
    spec = mc3.interface(diopside, -1.0e-6*offset, fo8_4, d1, e0, nTraj=nt,
dose=600.0, xtraParams=params)

```

Script 1: An example showing how DTSA-II can be scripted using Python language syntax to simulate spectra from complex materials and geometries.