20-nm-C-250-nm-ZnO-Silica-15kV-reflin-5e-1

John Minter

Started 2018-08-07, Last modified: 2018-08-11

Abstract

The objective was to measure the time required for this simulation using the `Zn L3-M4` transition to reach the `REFLIN` criterion of an uncertainty of $1.0e^{-1}$. This was a fairly weak transition (0.1140) compared to the stronger `Zn L3-M5` transition (1.0). Still, I want to see how the simulation scales with the value of the uncertainty criterion.

Contents

Set up for the analysis

Note: This simulation used the same `MSIMPA` parameters for all layers and a `REFLIN` parameter (`5.E-1`) to finish.

```
RSEED  -10   1                [Seeds of the random-number generator]
REFLIN 30040800 1 5.E-1     [Zn L3-M4 IZ*1e6+S1*1e4+S2*1e2,dete,tol.]
NSIMSH 1.0e8                    [Desired number of simulated showers]
TIME   1.0e6                       [Allotted simulation time, in sec]
```

First load the libraries we need…

```
library(rpenepma)
library(dplyr)
library(pander)
panderOptions('table.split.table', Inf)
library(ggplot2)
```

Next, set the paths and constants that we need…

```
moved_data  <- FALSE  # set to TRUE when simu is done and .rda with data is moved...
e0          <- 15 # kV
int_lo_lim  <- 1.0e-1
int_hi_lim  <- 1.0e4
reflin_crit <- 5e-1
```

```r
ori_sim_dir <- "C:/Users/jrminter/Documents/work/penepma16/20-nm-C-250-nm-ZnO-Silica-reflin2"
sim_dir     <- "/Users/jrminter/Desktop/test-plt"
sim_nam     <- "20-nm-C-250-nm-ZnO-Silica-reflin-5e-1"
res_fi      <- "./penepma-res.dat"
int_fi      <- "./pe-intens-01.dat"
spc_fi      <- "./pe-spect-01.dat"
out_ti      <- sprintf("%s-%gkV", sim_nam, e0)
msa_fi      <- sprintf("./%s-%gkV.msa", sim_nam, e0)
```

Get the current simulation time

```r
if (moved_data==FALSE){
  # When we have not moved the sim measure the simulation time.
  # Note copying the files to another directory seems to change the time
  # Test a reload.
  delta_t <- penepma_measure_simulation_time(ori_sim_dir)
  # print(delta_t)
  save(delta_t, file="./delta_t.rda")
  rm(delta_t)
  load("./delta_t.rda")
  msg <- "Analyzed from simulation directory."
} else {
  # When we have moved the simulation, reload the delta_t.rda
  msg <- "Analyzed from data stored in delta_t.rda."
  load("./delta_t.rda")
}
print(msg)
```

```
[1] "Analyzed from simulation directory."
```

```r
delta_t_time <- print(delta_t)
```

```
Time difference of 1.410857 hours
```

Sort summary: this is very short and noisy…

Retrieve the number of showers

We want retrieve the number of showers (trajectories) that our `penepma16` simulation calculated.

```r
options(scipen = -6) # force printing in exponential format
num_showers <- penepma_get_number_of_showers(res_fi)
cat(num_showers)
```

```
4.1783e+04
```

```r
options(scipen = 3) # reset to default
```

Compute the total maximum intensity for each element

Retrieve the data

First retrieve the intensity data as a `tibble` and print a preview.

```
tib <- penepma_get_intensities(int_fi)
print(tib)
```

```
# A tibble: 21 x 14
      IZ S0    S1      keV    P.mu    P.se    C.mu    C.se    B.mu    B.se
   <int> <chr> <chr> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
 1     6 K     L2    0.277 5.50e-7 1.65e-6 0.      0.      0.      0.
 2     6 K     L3    0.277 5.50e-7 1.65e-6 1.21e-8 3.63e-8 1.90e-9 5.69e-9
 3     8 K     L2    0.525 8.82e-6 3.09e-6 0.      0.      4.36e-8 8.10e-8
 4     8 K     L3    0.525 2.05e-5 4.74e-6 8.46e-8 9.59e-8 7.58e-9 1.14e-8
 5    30 L3    M1    0.884 2.57e-5 5.33e-6 2.42e-8 5.13e-8 1.76e-8 1.71e-8
 6    30 L2    M1    0.906 1.72e-5 4.31e-6 0.      0.      1.33e-8 1.50e-8
 7    30 L3    M2    0.932 1.21e-7 3.63e-7 0.      0.      0.      0.
 8    30 L2    M3    0.957 1.21e-7 3.63e-7 0.      0.      0.      0.
 9    30 L3    M5    1.01  3.52e-5 6.18e-6 4.83e-8 7.25e-8 2.46e-8 2.05e-8
10    30 L3    M4    1.01  4.47e-6 2.20e-6 1.21e-8 3.63e-8 1.90e-9 5.69e-9
# ... with 11 more rows, and 4 more variables: TF.mu <dbl>, TF.se <dbl>,
#   Int.mu <dbl>, Int.se <dbl>
```

What features did we measure?

```
print(names(tib))
```

```
 [1] "IZ"     "S0"     "S1"     "keV"    "P.mu"   "P.se"   "C.mu"
 [8] "C.se"   "B.mu"   "B.se"   "TF.mu"  "TF.se"  "Int.mu" "Int.se"
```

How many transitions did we measure?

```
print(nrow(tib))
```

```
[1] 21
```

Measure our test transition ("Zn L3-M4")

```
val <- penepma_get_total_intensity_z_transition(tib, 30, "L3", "M4")
print(val)
```

```
# A tibble: 1 x 7
  Symbol    IZ S0    S1         Int.mu    Int.se Int.snr
  <chr>  <int> <chr> <chr>       <dbl>     <dbl>   <dbl>
1 Zn        30 L3    M4     0.00000449 0.0000022    2.04
```

And see how close we are to the `REFLIN` criterion

```
cur_ref_crit <- val$Int.se/val$Int.mu

rv <- c(cur_ref_crit, 100*reflin_crit/cur_ref_crit)
names(rv) <-c("value", "target %")
print(rv)
```

```
      value    target %
  0.4905128 101.9341364
```

Let's estimate the remaining time

```r
pct <- rv[2]/100.
names(pct) <- c()
tr <- delta_t*(1.0-pct)
print(tr)
```

```
Time difference of -0.02728791 hours
```

Let's store a `.rda` file with `REFLIN` info

```r
tmp_v <- data.frame(as.character(sim_nam),
                    as.numeric(reflin_crit), as.character.POSIXt(delta_t_time),
                    as.numeric(num_showers), as.numeric(cur_ref_crit),
                    as.numeric(100*reflin_crit/cur_ref_crit))
names(tmp_v) <- c("name", "reflin", "time", "showers",
                  "value", "target %")

tmp_v <- as_tibble(tmp_v)
tmp_v$name <- as.character(tmp_v$name)
C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1_ro <- tmp_v
rm(tmp_v)

save(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1_ro,
     file="./C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1_ro.rda")
rm(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1_ro)
load("./C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1_ro.rda")
pander(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1_ro)
```

name	reflin	time	showers	value	target %
20-nm-C-250-nm-ZnO-Silica-reflin-5e-1	0.5	1.410857 hours	41783	0.4905	101.9

Compute the maximum total intensity for each element

1. Compute the maximum total intensity for each element.
2. Stack each row of data together into a tibble.
3. Prepend the sample ID to the data.
4. Print a well-formatted a table using `pander`.
5. Save the final tibble so it can be r-loaded later

```r
# 1
C_t   <- penepma_get_max_total_intensity_z(tib, 6)
O_t   <- penepma_get_max_total_intensity_z(tib, 8)
Zn_t  <- penepma_get_max_total_intensity_z(tib, 30)
Si_t  <- penepma_get_max_total_intensity_z(tib, 14)
# 2
tot_int <- dplyr::bind_rows(C_t, O_t, Zn_t, Si_t)
#3
C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1 <- prepend_sample_id_max_int_tib(tot_int, out_ti)
# 4
pander(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1)
```

Sample.ID	Symbol	IZ	S0	S1	Int.mu	Int.se	Int.snr
20-nm-C-250-nm-ZnO-Silica-reflin-5e-1-15kV	C	6	K	L3	0.0000005645	0.00000165	0.3421
20-nm-C-250-nm-ZnO-Silica-reflin-5e-1-15kV	O	8	K	L3	0.00002064	0.00000475	4.344
20-nm-C-250-nm-ZnO-Silica-reflin-5e-1-15kV	Zn	30	L3	M5	0.00003524	0.00000618	5.702
20-nm-C-250-nm-ZnO-Silica-reflin-5e-1-15kV	Si	14	K	L3	0.00002673	0.00008	0.3342

```
# 5
save(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1,
     file="C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1.rda")
# delete and reload be sure it worked...
rm(C_20_nm_ZnO_250_nm_SiO2_15kV_refli2n)
load("./C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_5e_m1.rda")
```

Process the spectra data

Load the raw data

```
tib <- penepma_read_raw_data(spc_fi,min_intensity_clip=5.0e-10)
rownames(df) <- c()
```

Examine the start:

```
pander(head(tib))
```

keV	mu	se
0.01183	0.5	0.5
0.01549	0.5	0.5
0.01915	0.5	0.5
0.02281	0.5	0.5
0.02647	0.5	0.5
0.03013	0.5	0.5

Examine the end:

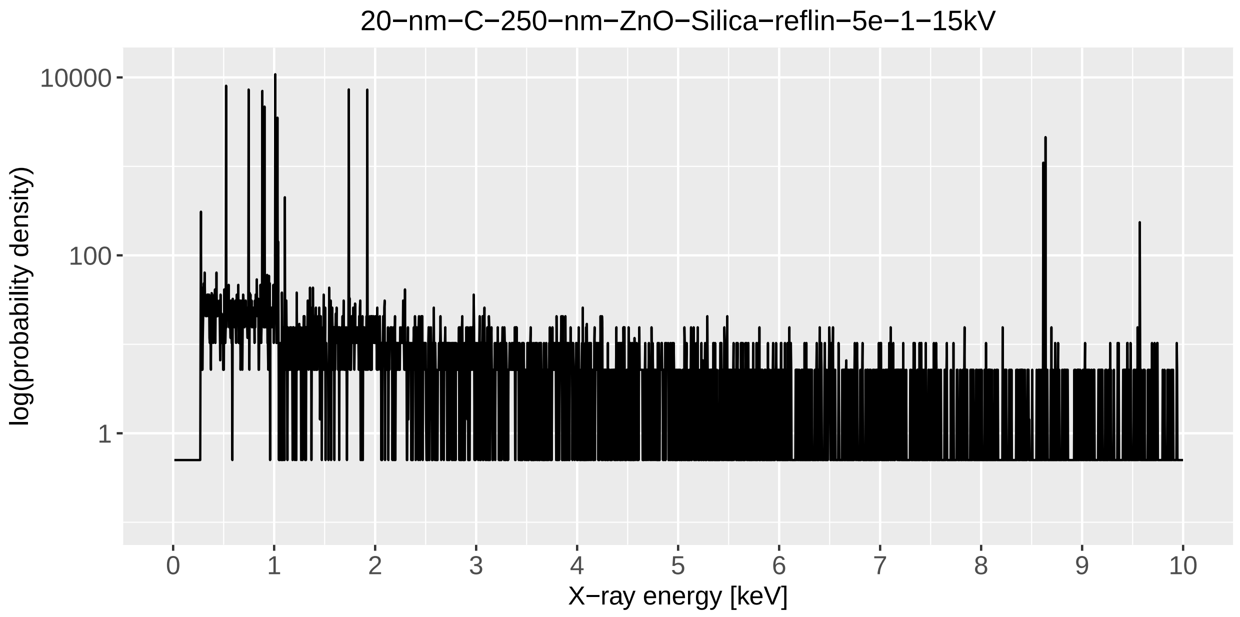
```
pander(tail(tib))
```

keV	mu	se
14.98	0.5	0.5
14.98	0.5	0.5
14.99	0.5	0.5
14.99	0.5	0.5
14.99	0.5	0.5
15	0.5	0.5

Plot the spectrum.

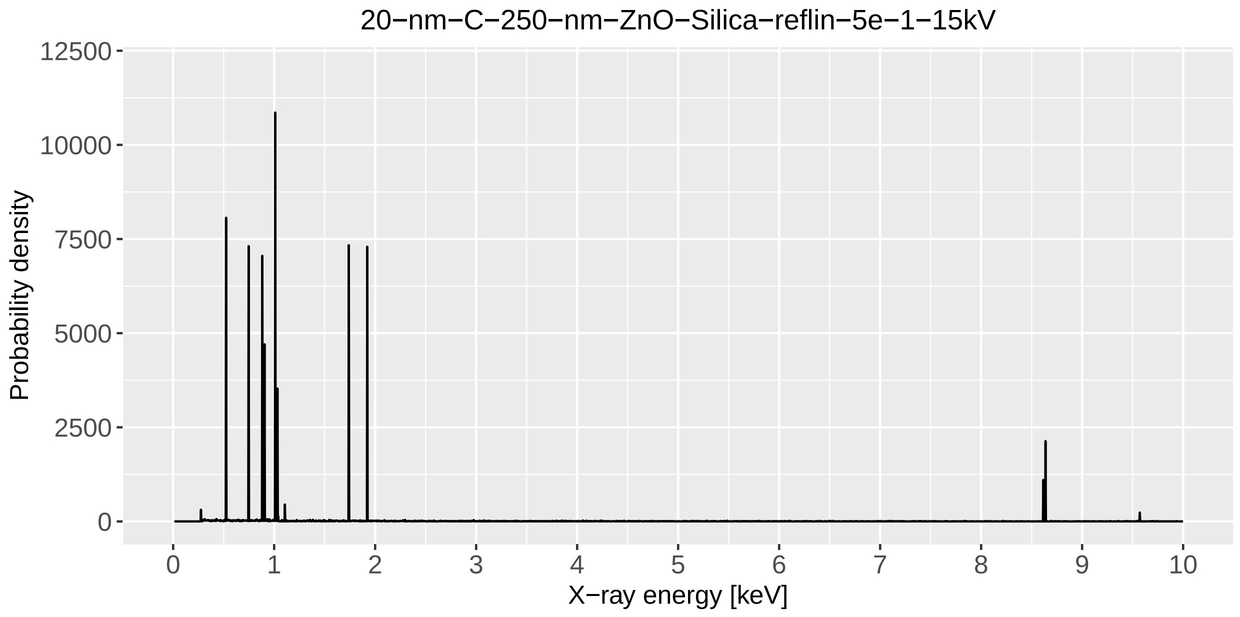
There is a very large dynamic range for both the **probability density** and the **uncertainty**. Penepma sets a lower limit for data at **1.0e-35**. Missing values are set to zero. We want to remove values from the dataframe that are below a useful limit. We do this below and plot a copy of the dataframe that is limited to the useful values.

```r
m_t <- max(pretty(tib$mu))

plt <- ggplot(tib, aes(x = keV, y = mu)) +
       geom_line() +
       scale_x_continuous(breaks = seq(from = 0, to = e0-5, by = 1),
                          limits = c(0,e0-5)) +
       scale_y_log10(limits = c(int_lo_lim, m_t)) +
       xlab(label="X-ray energy [keV]") +
       ylab(label="log(probability density)") +
       # (1/(eV*sr*electron)") +
       ggtitle(out_ti) +
       theme(axis.text=element_text(size=12),
             axis.title=element_text(size=12),
             # center the title
             plot.title = element_text(hjust = 0.5))

print(plt)
```



And on a linear intensity scale...

```r
plt <- ggplot(tib, aes(x = keV, y = mu)) +
       geom_line() +
       scale_x_continuous(breaks = seq(from = 0, to = e0-5, by = 1),
                          limits = c(0,e0-5)) +
       scale_y_continuous(limits = c(int_lo_lim, m_t)) +
       xlab(label="X-ray energy [keV]") +
       ylab(label="Probability density") +
       # (1/(eV*sr*electron)") +
       ggtitle(out_ti) +
       theme(axis.text=element_text(size=12),
             axis.title=element_text(size=12),
             # center the title
             plot.title = element_text(hjust = 0.5))

print(plt)
```



Write a spectrum file in MSA format

```
penepma_to_msa(spc_fi, msa_fi,e0, out_ti)
```