

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

John Minter

Started 2018-08-07, Last modified: 2018-08-13

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	1
Get the current simulation time	2
Retrieve the number of showers	2
Compute the total maximum intensity for each element	2
Retrieve the data	2
Compute the maximum total intensity for each element	4
Process the spectra data	5
Load the raw data	5
Plot the spectrum.	5
Write a spectrum file in MSA format	7

Set up for the analysis

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

```
RSEED -10 1 [Seeds of the random-number generator]
REFLIN 30040800 1 1.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)
library(here)
library(readr)
```

Next, set the paths and constants that we need...

```
moved_data <- TRUE # 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 <- 1e-1
data_dir <- here("penepma/penepma16/reflin-test/20-nm-C-250-nm-ZnO-Silica-15kV-reflin-1e-1")
# ori_sim_dir <- "C:/Users/jrminter/Documents/work/penepma16/20-nm-C-250-nm-ZnO-Silica-reflin2"
sim_nam <- "20-nm-C-250-nm-ZnO-Silica-reflin-1e-1"
res_fi <- sprintf("%s/%s", data_dir, "penepma-res.dat")
tim_fi <- sprintf("%s/%s", data_dir, "delta_t.rda")
int_fi <- sprintf("%s/%s", data_dir, "pe-intens-01.dat")
spc_fi <- sprintf("%s/%s", data_dir, "pe-spect-01.dat")
out_ti <- sprintf("%s-%gkV", sim_nam, e0)
msa_fi <- sprintf("%s/%s-%gkV.msa", data_dir, sim_nam, e0)

```

Get the current simulation time

```

# When we have moved the simulation, reload the delta_t.rda
msg <- "Analyzed from data stored in delta_t.rda."
load(tim_fi)

print(msg)

```

```
[1] "Analyzed from data stored in delta_t.rda."
```

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

Time difference of 1.689805 days

Sort summary: this is precision overkill...

Retrieve the number of showers

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

```

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

```

```
1.253511e+06
```

```
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: 24 x 14
```

```
  IZ S0    S1    keV    P.mu    P.se    C.mu    C.se    B.mu
```

```

      <int> <chr> <chr> <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
1       6 K      L2    0.277 8.24e-7 3.66e-7 1.21e-9 2.09e-9 5.23e-10
2       6 K      L3    0.277 1.60e-6 5.09e-7 2.42e-9 2.96e-9 4.42e-10
3       8 K      L2    0.525 1.67e-5 7.57e-6 3.18e-8 1.94e-8 8.55e- 9
4       8 K      L3    0.525 2.90e-5 9.28e-6 8.18e-8 3.85e-8 1.17e- 8
5      30 L3     M1    0.884 2.77e-5 1.00e-6 1.24e-7 2.67e-7 1.12e- 7
6      30 L2     M1    0.906 1.34e-5 6.98e-7 6.04e-9 1.24e-8 9.67e- 9
7      30 L3     M2    0.932 8.06e-8 5.40e-8 0.      0.      1.44e-10
8      30 L3     M3    0.934 6.04e-8 4.68e-8 0.      0.      0.
9      30 L2     M3    0.957 6.04e-8 4.68e-8 0.      0.      0.
10     30 L3     M5     1.01 3.72e-5 1.16e-6 1.31e-7 2.67e-7 2.81e- 8
# ... with 14 more rows, and 5 more variables: B.se <dbl>, 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] 24
```

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    0.00000362 0.000000362 10.00

```

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 %
0.1000491 99.9509116

```

Let's estimate the remaining time

```

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

```

Time difference of 0.000829498 days

Let's store a .rda file with REFLIN info

```

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_1e_m1_ro <- tmp_v
rm(tmp_v)

save(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1_ro,
     file="./C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1_ro.rda")
rm(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1_ro)
load("./C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1_ro.rda")
pander(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1_ro)

```

name	reflin	time	showers	value	target %
20-nm-C-250-nm-ZnO-Silica-reflin-1e-1	0.1	1.689805 days	1253511	0.1	99.95

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

```

# 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_1e_m1 <- prepend_sample_id_max_int_tib(tot_int, out_ti)
# 4
pander(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1)

```

Sample.ID	Symbol	IZ	S0	S1	Int.mu	Int.se	Int.snr
20-nm-C-250-nm-ZnO-Silica-reflin-1e-1-15kV	C	6	K	L3	0.000001599	0.000000509	3.141
20-nm-C-250-nm-ZnO-Silica-reflin-1e-1-15kV	O	8	K	L3	0.00002905	0.00000928	3.13
20-nm-C-250-nm-ZnO-Silica-reflin-1e-1-15kV	Zn	30	L3	M5	0.00003732	0.00000119	31.36
20-nm-C-250-nm-ZnO-Silica-reflin-1e-1-15kV	Si	14	K	L3	0.00004103	0.0000181	2.267

```
# 5
save(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1,
     file="C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1.rda")
# delete and reload be sure it worked...
rm(C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_m1)
load("./C_20_nm_ZnO_250_nm_SiO2_15kV_reflin_1e_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.

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

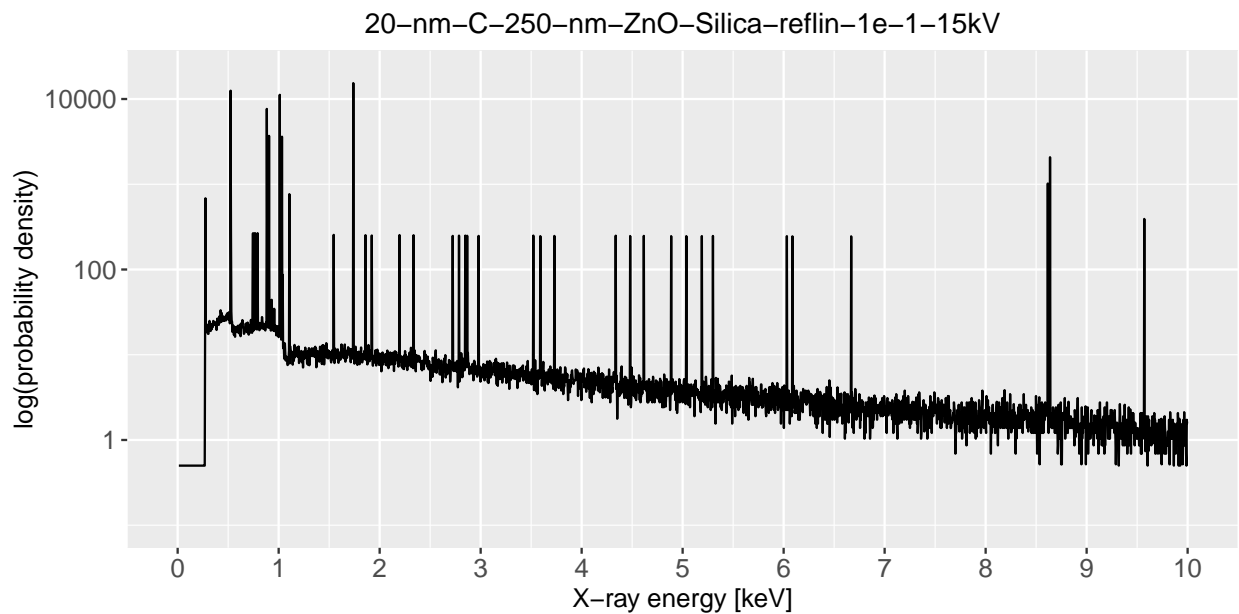
plt <- ggplot(tib, aes(x = keV, y = mu)) +
  geom_line() +
```

```

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)

```



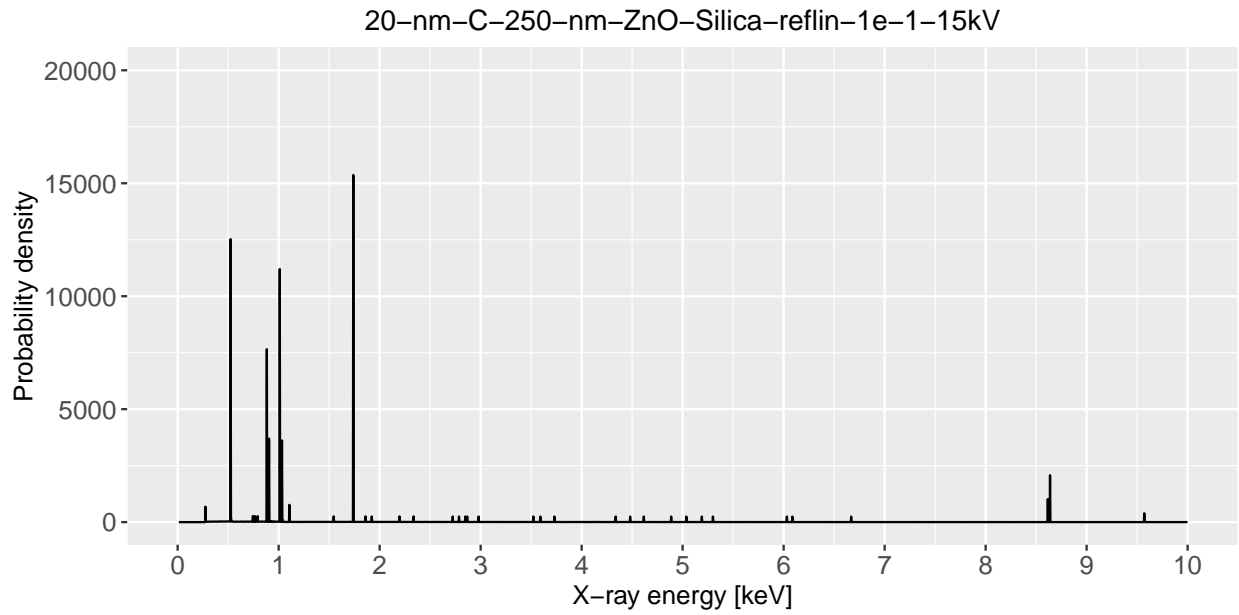
And on a linear intensity scale...

```

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)
```