Peter Willendrup, DTU Physics + ESS DMSC

Sources and Monitors part 2.

Sources: Source model overview

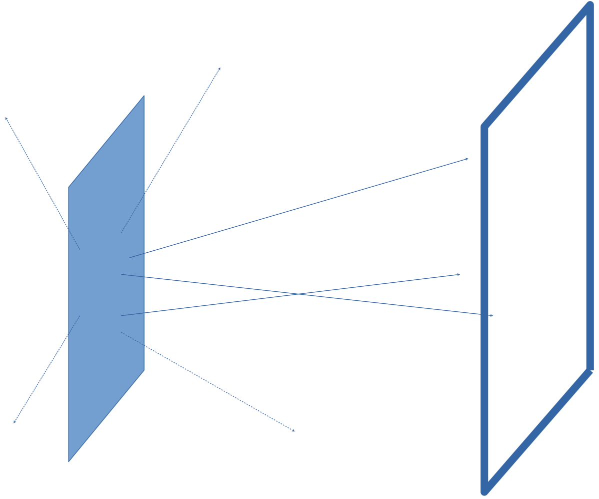
- ***Mathematical**:*
  - *Source_simple.comp*
  - *Source_div.comp*
- ***Pulsed sources**:*
  - *ESS_butterfly.comp*
  - *ESS_moderator.comp*
  - *Moderator.comp*
  - *SNS_source.comp (*)*
  - *SNS_source_analytic (*)*
  - *ViewModISIS (*)*
  - *ISIS_moderator.comp (*)*
- ***Reactors** :*
  - *Source_Maxwell_3.comp*
  - *Source_gen.comp*
  - *Source_gen4.comp*
  - *Source_multi_surfaces.comp (*)*
- *I/O mechanisms:*
- ***MCPL_input/output.comp***
  - *Virtual_input/output.comp*
  - *Virtual_mcnp_ss_input/output.comp*
  - *Virtual_tripoli4_input/output.comp*
  - *Vitess_input/output.comp*

Sources: Source_Maxwell_3

```
COMPONENT source = Source_Maxwell_3(yheight=0.156, xwidth=0.126,
                   Lmin=0.1, Lmax=9.0, dist=1.5, focus_xw = 0.025, focus_yh = 0.12,
                   T1=150.42, I1=3.67E11, T2=38.74, I2=3.64E11, T3=14.84, I3=0.95E11)
```

Parameters from the PSI cold source

Initial position and direction: as for Source_simple



Sources: Source_Maxwell_3

```
COMPONENT source = Source_Maxwell_3(yheight=0.156, xwidth=0.126,
               Lmin=0.1, Lmax=9.0, dist=1.5, focus_xw = 0.025, focus_yh = 0.12,
               T1=150.42, I1=3.67E11, T2=38.74, I2=3.64E11, T3=14.84, I3=0.95E11)
```

Parameters from the PSI cold source

Intensity at a given wavelength drawn from a sum of (up to) 3 normalized Maxwellian distributions:

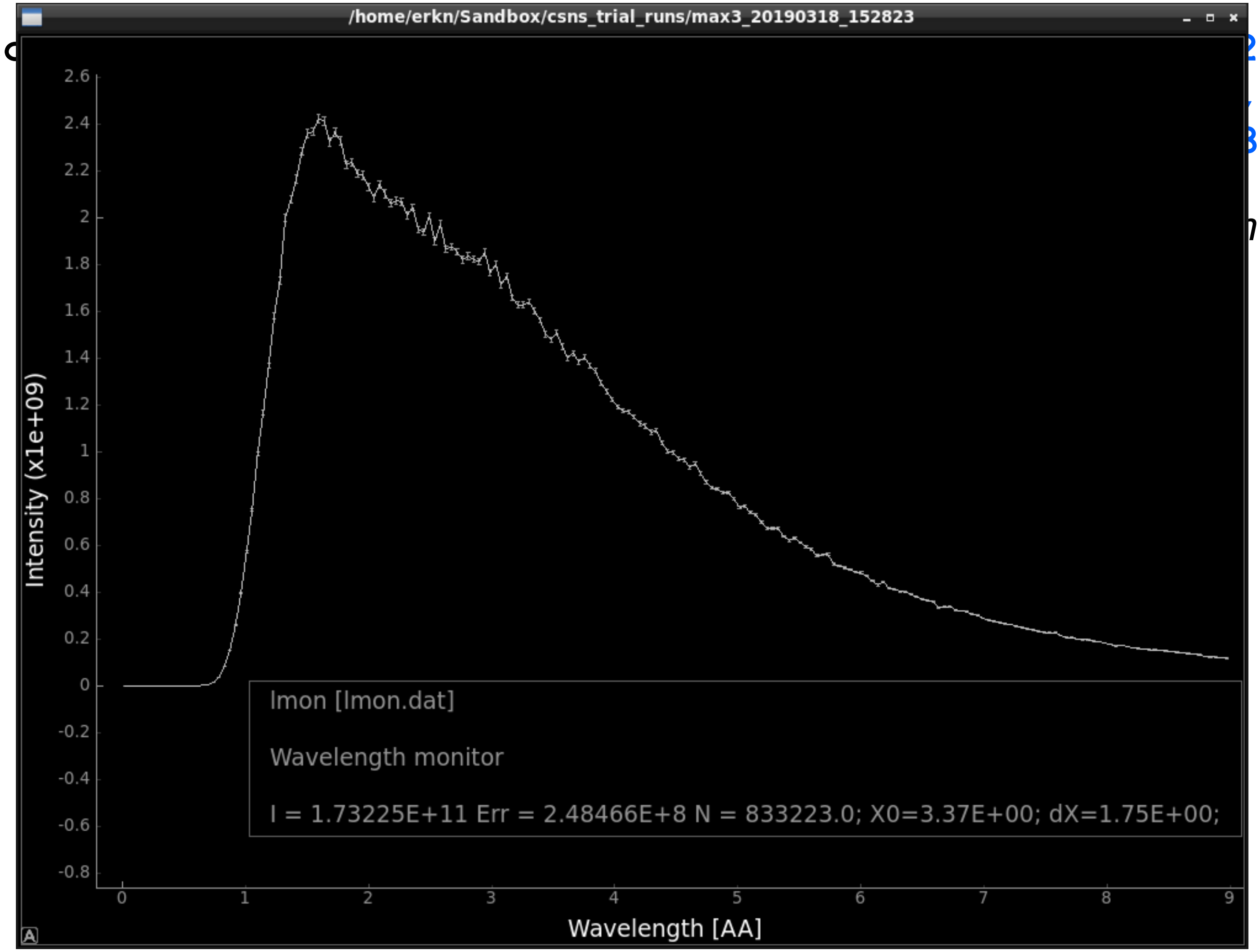
$$I(\lambda) = \sum I_i M\left(\lambda, T_i\right); \qquad M\left(\lambda, T_i\right) = 2\alpha^2 exp\left(\frac{-\alpha}{\lambda^2}\right)/\lambda^5;$$

$$\alpha = 949.0KAA^2/T_i$$

Sources: Source_Maxwell_3

```
COMPONENT so                                                        26,
                                                                 , focus_yh = 0.12,
                                                                 3=14.84, I3=0.95E11)
```

n the PSI cold source



/home/erkn/Sandbox/csns_trial_runs/max3_20190318_152823

Imon [Imon.dat]

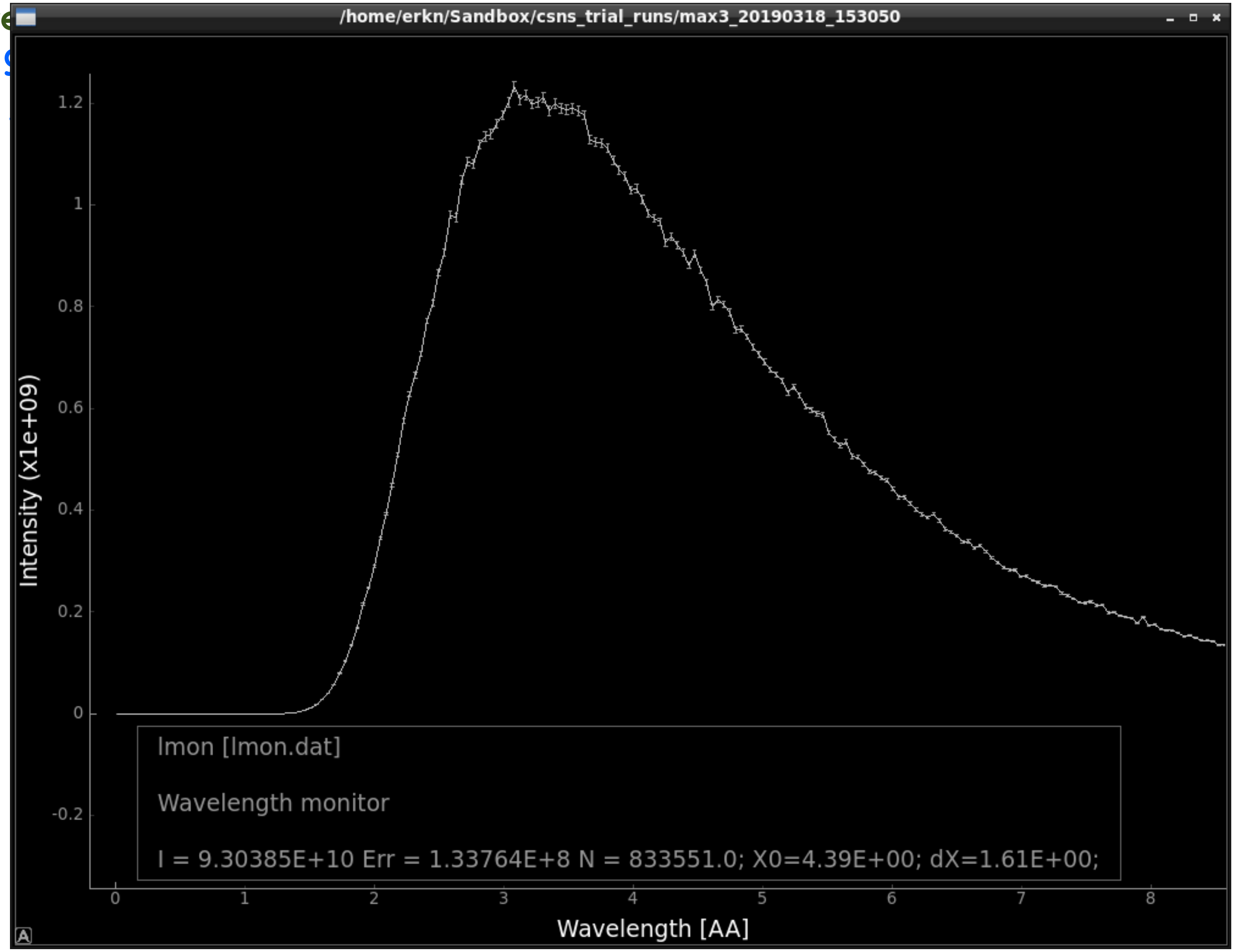
Wavelength monitor

I = 1.73225E+11 Err = 2.48466E+8 N = 833223.0; X0=3.37E+00; dX=1.75E+00;

Sources: Source_Maxwell_3

```
COMPONENT source = Source_Maxwe
                     Lmin=0.1, Lmax=
                     T1=150.42, I1=3
```

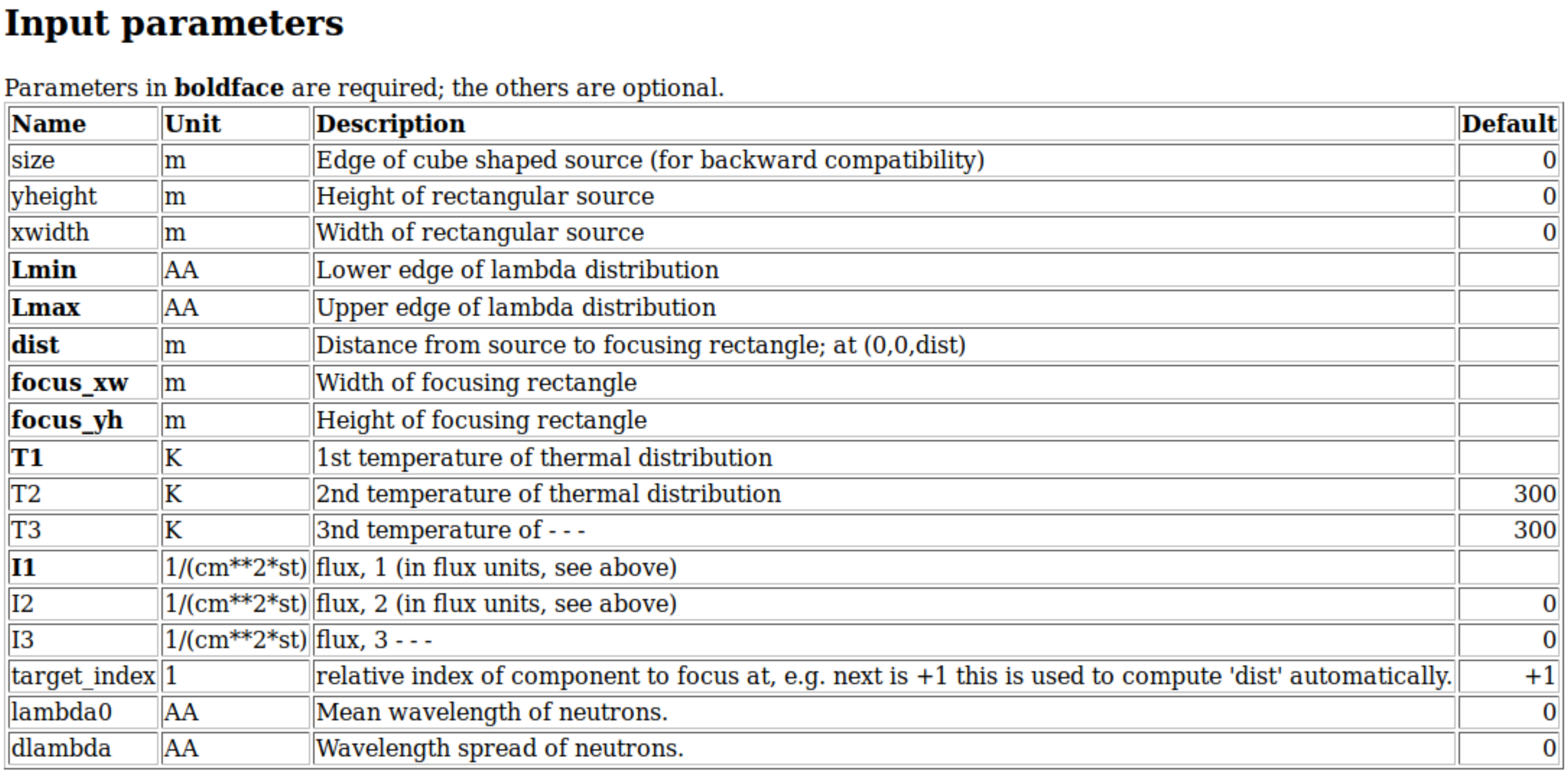
Just for fun – let’s see what happens if we remove the fast peak…



Imon [Imon.dat]

Wavelength monitor

I = 9.30385E+10 Err = 1.33764E+8 N = 833551.0; X0=4.39E+00; dX=1.61E+00;



Input parameters

Parameters in **boldface** are required; the others are optional.

Name	**Unit**	**Description**	**Default**
size	m	Edge of cube shaped source (for backward compatibility)	0
yheight	m	Height of rectangular source	0
xwidth	m	Width of rectangular source	0
Lmin	AA	Lower edge of lambda distribution	
Lmax	AA	Upper edge of lambda distribution	
dist	m	Distance from source to focusing rectangle; at (0,0,dist)	
focus_xw	m	Width of focusing rectangle	
focus_yh	m	Height of focusing rectangle	
T1	K	1st temperature of thermal distribution	
T2	K	2nd temperature of thermal distribution	300
T3	K	3nd temperature of - - -	300
I1	1/(cm**2*st)	flux, 1 (in flux units, see above)	
I2	1/(cm**2*st)	flux, 2 (in flux units, see above)	0
I3	1/(cm**2*st)	flux, 3 - - -	0
target_index	1	relative index of component to focus at, e.g. next is +1 this is used to compute 'dist' automatically.	+1
lambda0	AA	Mean wavelength of neutrons.	0
dlambda	AA	Wavelength spread of neutrons.	0

Sources: Source_gen (Source_gen4)

```
COMPONENT source = Source_gen(yheight=0.156, xwidth=0.126,
                   Lmin=0.1, Lmax=9.0, dist=1.5, focus_xw = 0.025, focus_yh = 0.12,
                   T1=150.42, I1=3.67E11, T2=38.74, I2=3.64E11, T3=14.84, I3=0.95E11)
```

Almost the same as Source_Maxwell_3: but with optional flux-files as input.

MCPL_input/output

Reads/writes events directly from MCPL-format files:

“T. Kittelmann et. al., “”, J. Phys. Comp., 2017

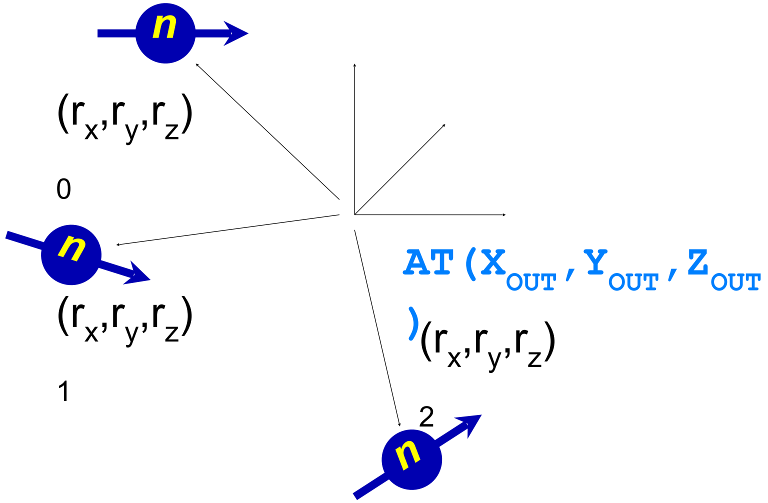
```
COMPONENT vout = MCPL_output(filename=”voutput.mcpl”,
        doubleprec=1,polarisationuse=1,verbose=1)
AT (Xout,Yout,Zout) RELATIVE PREVIOUS

COMPONENT vin = MCPL_input(filename=”voutput.mcpl”,
        polarisationuse=1,verbose=1)
AT (Xin,Yin,Zin) RELATIVE PREVIOUS
```

MCPL_input/output

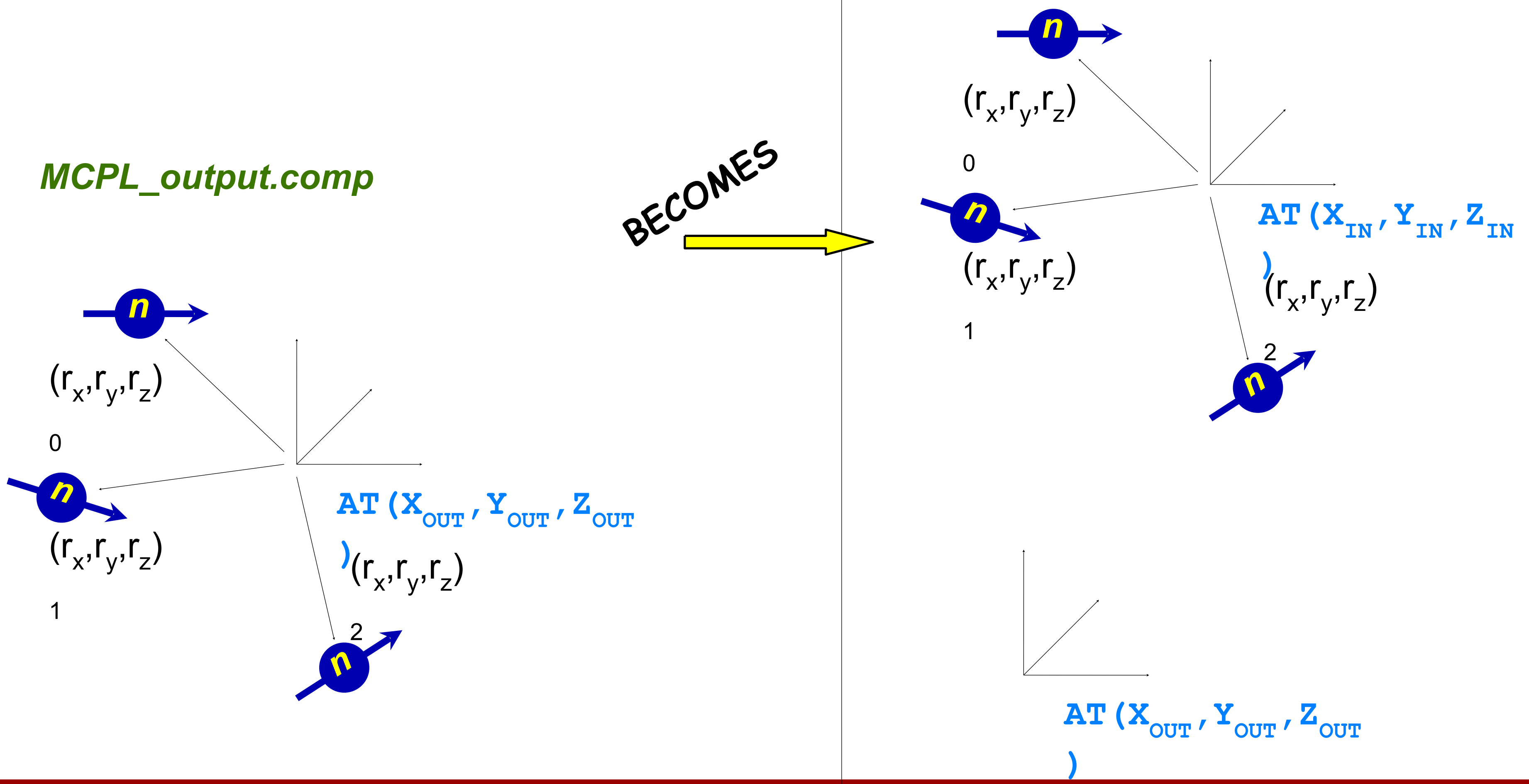
Can include an Implicit Translation:

MCPL_output.comp



MCPL_input/output

Can include an Implicit Translation:



MCPL_output.comp

MCPL_input.comp

Pulsed sources:

Simplest case:

Use a continuous source!

Model a source with given wavelength and spatial distribution

and

_________… an infinitely short pulse length. I.e. ***t* = *0*** for all neutron rays.

```
COMPONENT src = Source_simple(
        radius=0.05, lambda0=2.5, dlambda=1.5,
        focus_xw=0.1, focus_yh=0.1, dist=5 )
AT(0,0,0) RELATIVE origin
```

Pulsed sources:

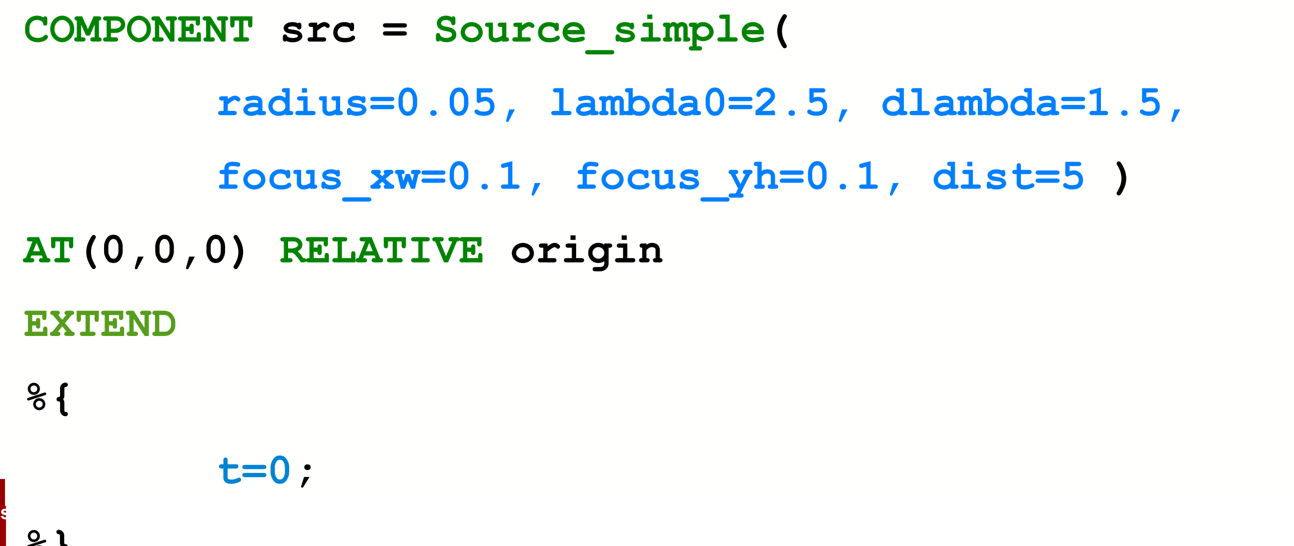
Simplest case:

Use a continuous source!

Model a source with given wavelength and spatial distribution

and

… an infinitely short pulse length. I.e. ***t = 0*** for all neutron rays.



```
COMPONENT src = Source_simple(
        radius=0.05, lambda0=2.5, dlambda=1.5,
        focus_xw=0.1, focus_yh=0.1, dist=5 )
AT(0,0,0) RELATIVE origin
EXTEND
%{
        t=0;
%}
```

Pulsed sources:

Simplest case:

Use a continuous source!

Model a source with given wavelength and spatial distribution

and

… an infinite … tron rays.

Or: Use a chopper

(see later)

```
COMPONENT
        rad…                    …5, dlambda=1.5,
        focu…            1, focus_yh=0.1, dist=5 )
AT(0,0,0) RELATIVE origin
EXTEND
%{
        t=0;
%}
```

Pulsed Sources: Moderator

A flat pulsed source with uniform energy spectrum:

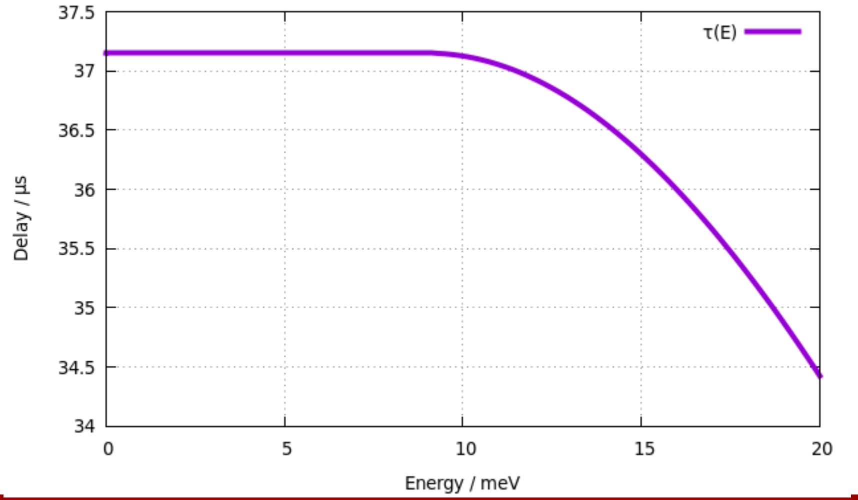
$$x \in U\left[-\frac{xwidth}{2},\frac{xwidth}{2}\right]\ y \in U\left[-\frac{yheight}{2},\frac{yheight}{2}\right]$$

$$|v| = f(\lambda)\ ; \lambda \in U\left[L_{min}L_{max}\right]$$

Time structure is given by energy dependent probability density function:

$$f_t = \frac{1}{\tau}\, exp\left(-\frac{t}{\tau}\right)$$

$$\tau\ = \begin{cases} t_0; & E < E_c \\ t_0\left(\frac{1}{1+\frac{(E-E_c)}{\gamma}}\right); & E \geq Ec \end{cases}$$

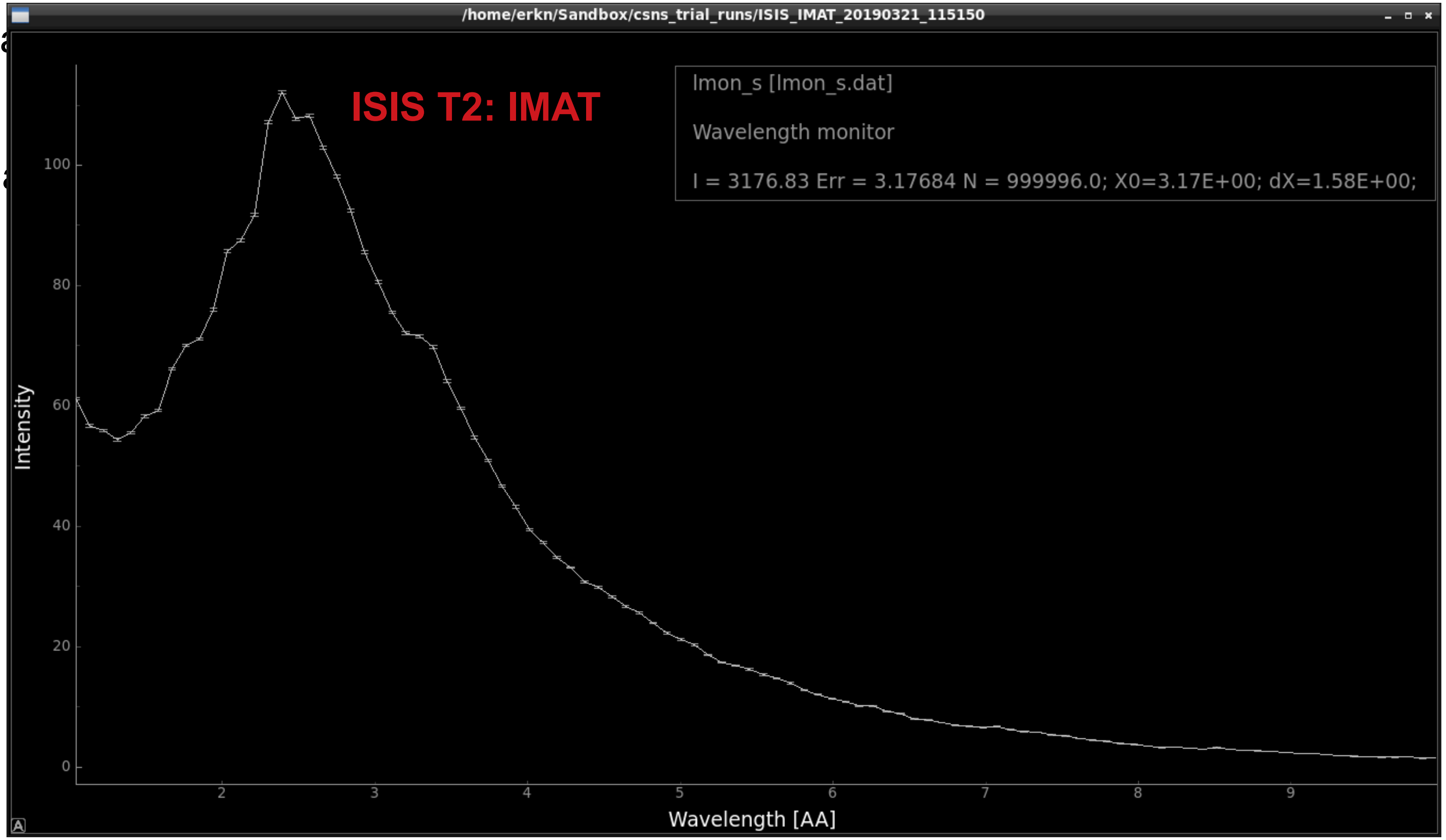


Pulsed Sources: ViewModISIS

- Samples directly from tallies coming from e.g. MCNP target+moderator calculations.
- Data file supplied for each beam port at ISIS.

Pulsed Sources: ViewModISIS

- Sa
- Da



/home/erkn/Sandbox/csns_trial_runs/ISIS_IMAT_20190321_115150

ISIS T2: IMAT

Imon_s [Imon_s.dat]

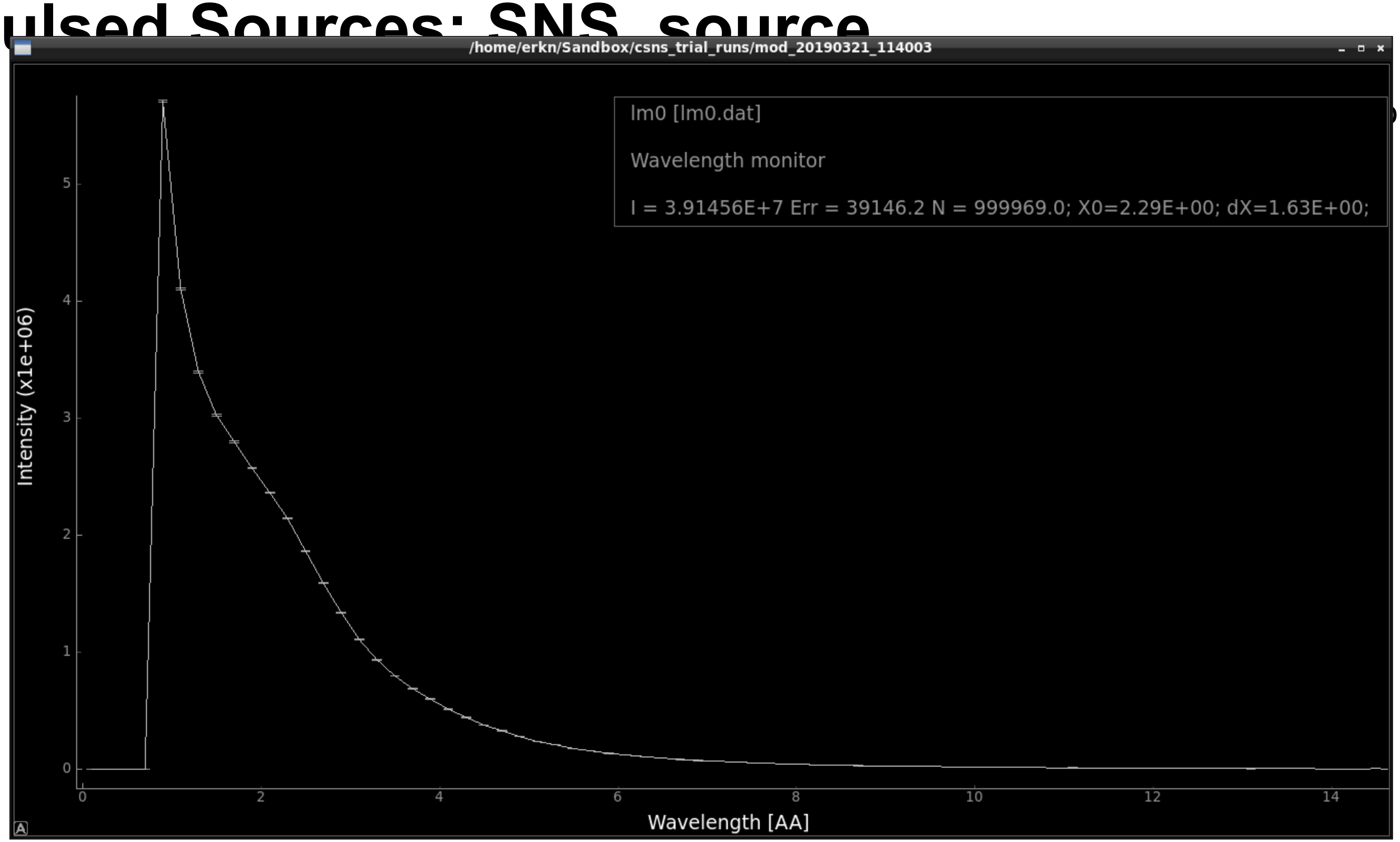
Wavelength monitor

I = 3176.83 Err = 3.17684 N = 999996.0; X0=3.17E+00; dX=1.58E+00;

Pulsed Sources: SNS_source

- Samples directly from tallies coming from e.g. MCNP target+moderator calculations.
- Originally from SNS but also used extensively at J-PARC
- Can be used (with the proper input files) to model CSNS, and likely also ISIS.

Pulsed Sources: SNS_source



lm0 [lm0.dat]

Wavelength monitor

I = 3.91456E+7 Err = 39146.2 N = 999969.0; X0=2.29E+00; dX=1.63E+00;

ns.

Pulsed Sources: SNS_source_analytic

- Samples from fits of Padé-functions to tallies from SNS_source.
- - Requires a complex fitting campaign
- + Much faster than SNS_source
  + “Cleaner” distributions where statistics are sketchy
- Can be used (with the proper input files) to model CSNS-source.

Monitors (some)

1D

- L_monitor → $I(\lambda)$
- TOF_monitor → $I(t)$
- Hdiv_monitor → $I\left(\text{div.}_x\right)$
- MeanPolLambda → $\langle \bar{P} \rangle (\lambda)$
- E_monitor → $I(E)$

2D

- PSD_monitor → $I(x, y)$
- PSD_monitor_4PI → $I(\theta, \phi)$
- PolLambda_monitor → $I\left(\bar{P}, \lambda\right)$
- Divergence_monitor → $I\left(\text{div.}_x, \text{div.}_y\right)$
- DivPos_monitor → $I\left(\text{div.}_x, x\right)$

nD

- Monitor_nD → $I(X)$

or

$I(X, Y)$

or

$Z(X, Y, Z)$

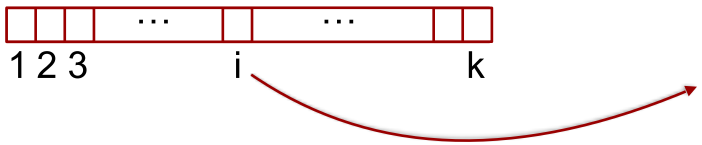
or ...

Monitors: Quick examples

```
COMPONENT my_L_monitor = L_monitor(xwidth=0.2, yheight=0.2,
                         nL=20, filename="Output.L", Lmin=2, Lmax=10)
```

In a histogram sense

- Imagine a histogram, e.g. **I**(λ)

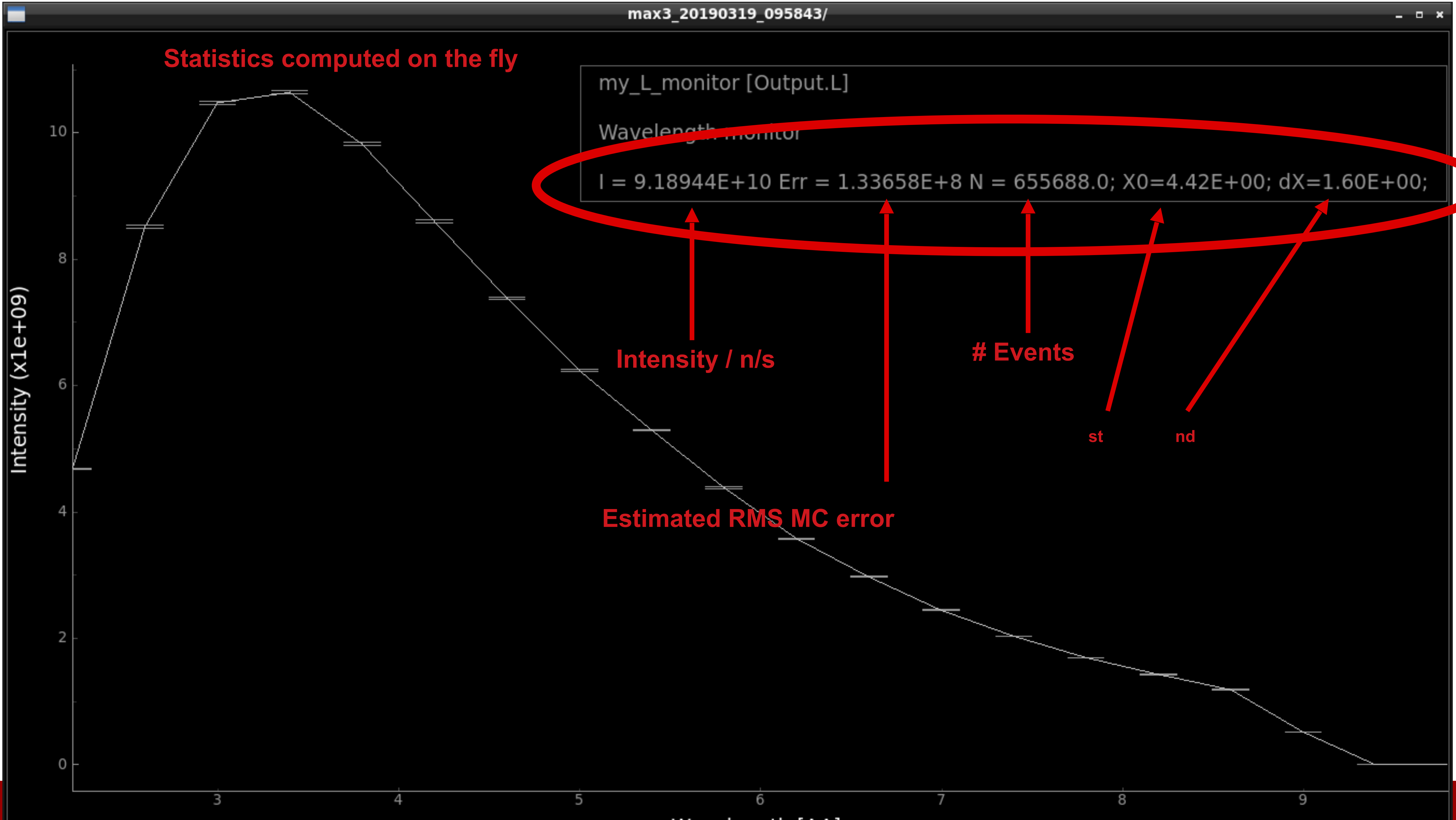


In bin *i*, ***N*** events each carrying a fractional intensity p_j so that

$$I = \sum_{N} p_j$$

- The RMS variance over that set becomes our statistical error bar ***E***

Statistics computed on the fly



my_L_monitor [Output.L]

Wavelength monitor

```
I = 9.18944E+10 Err = 1.33658E+8 N = 655688.0; X0=4.42E+00; dX=1.60E+00;
```

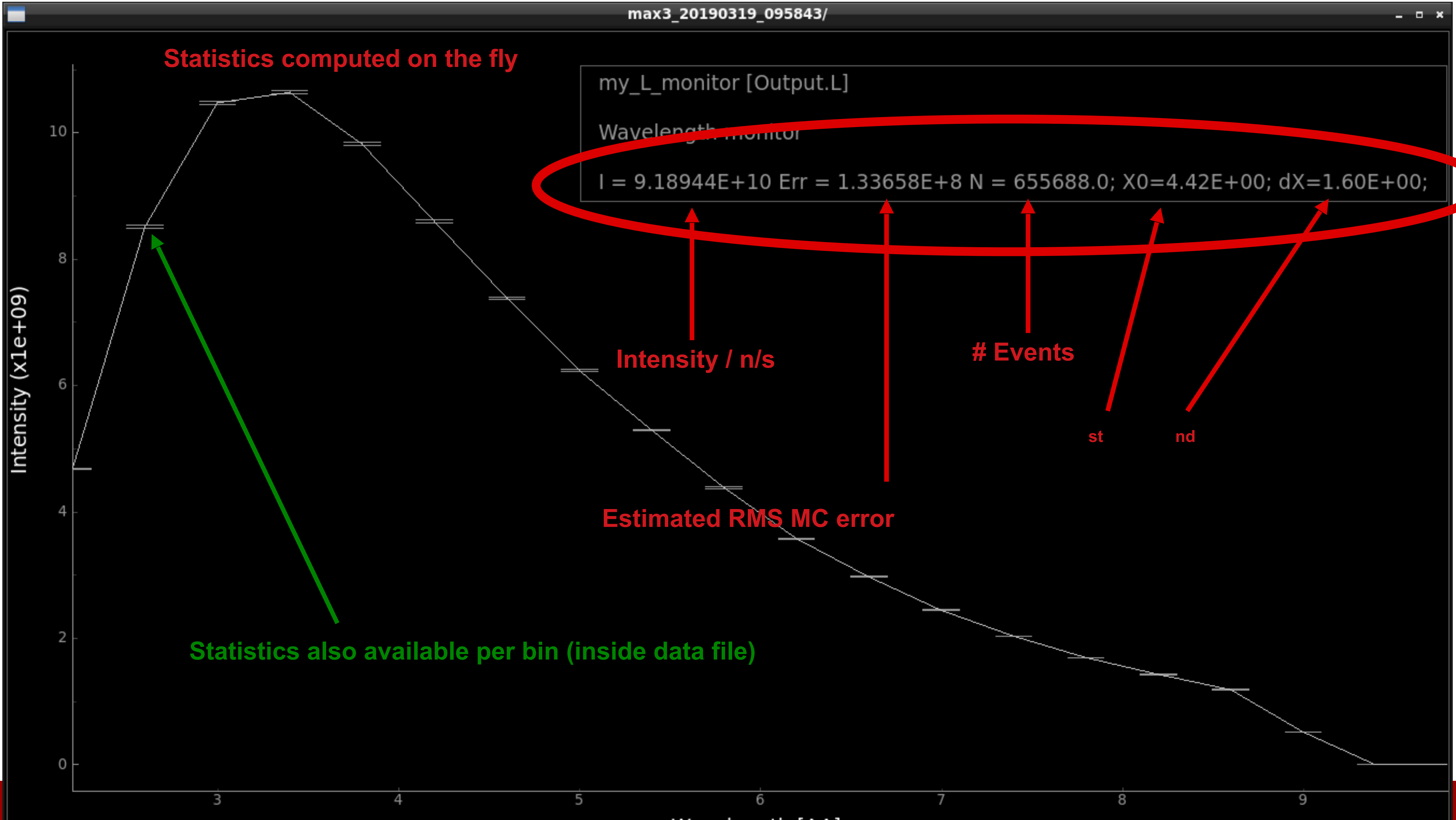
Intensity / n/s

Estimated RMS MC error

# Events

st

nd



Statistics computed on the fly

my_L_monitor [Output.L]

Wavelength monitor

```
I = 9.18944E+10 Err = 1.33658E+8 N = 655688.0; X0=4.42E+00; dX=1.60E+00;
```

Intensity / n/s

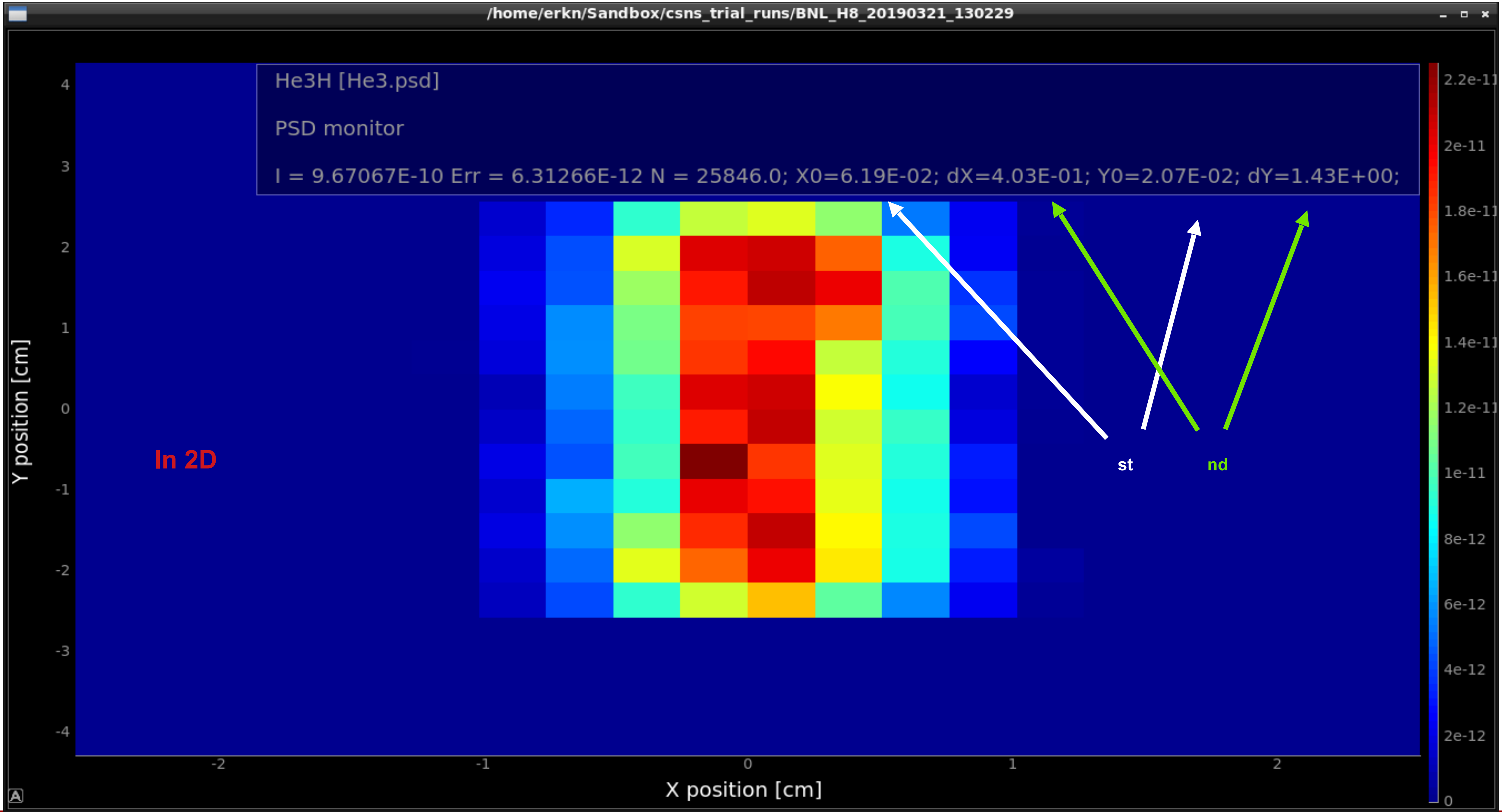
Estimated RMS MC error

# Events

st

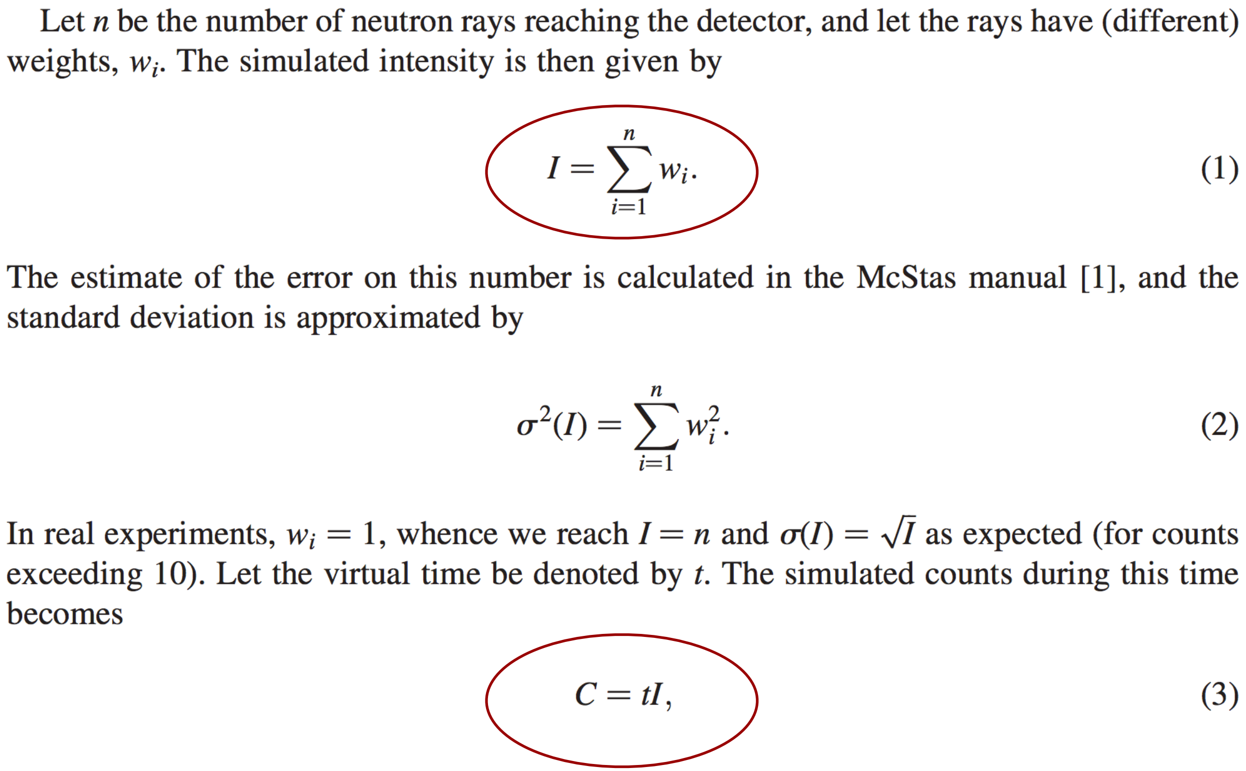
nd

Statistics also available per bin (inside data file)



I = 9.67067E-10 Err = 6.31266E-12 N = 25846.0; X0=6.19E-02; dX=4.03E-01; Y0=2.07E-02; dY=1.43E+00;

From "Virtual experiments - the ultimate aim of neutron ray-tracing simulations", K. Lefmann et al., Journal of Neutron Research 16, 97-111 (2008)



Let n be the number of neutron rays reaching the detector, and let the rays have (different) weights, w_i. The simulated intensity is then given by

$$I = \sum_{i=1}^{n} w_i. \tag{1}$$

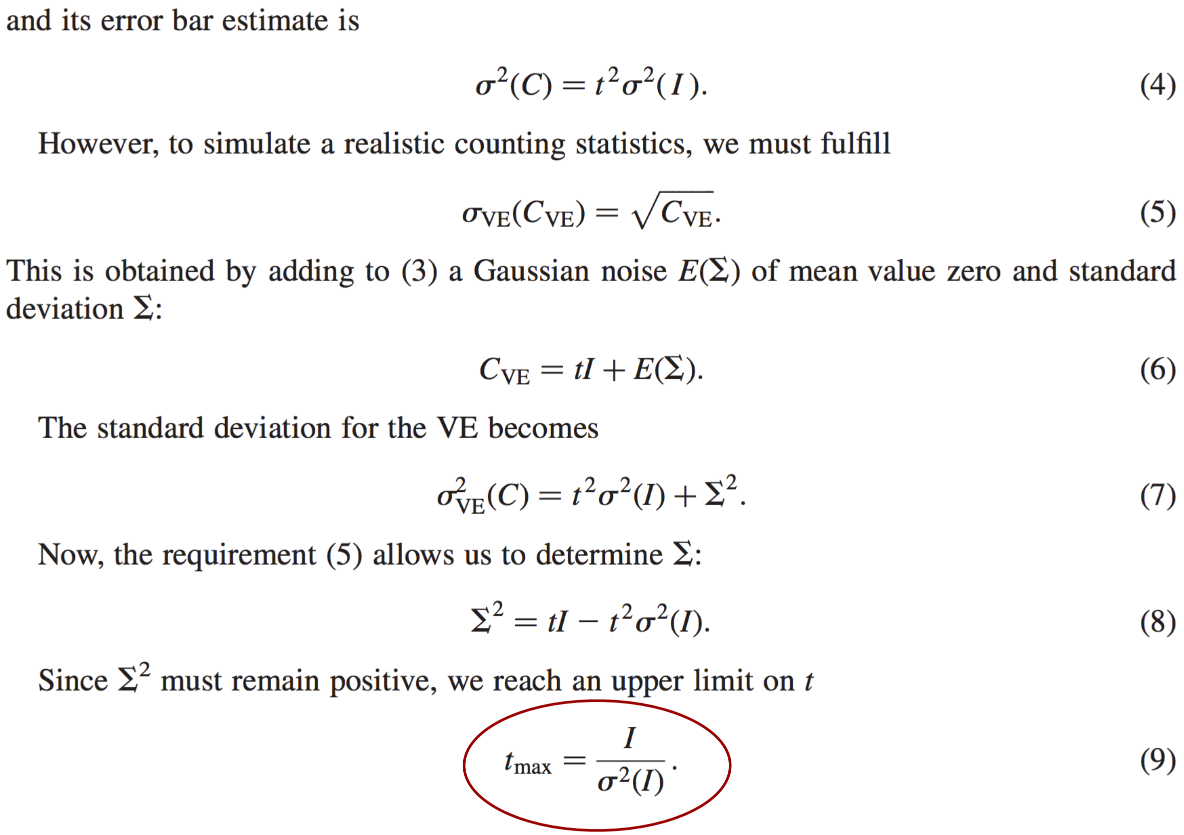
The estimate of the error on this number is calculated in the McStas manual [1], and the standard deviation is approximated by

$$\sigma^2(I) = \sum_{i=1}^{n} w_i^2. \tag{2}$$

In real experiments, $w_i = 1$, whence we reach $I = n$ and $\sigma(I) = \sqrt{I}$ as expected (for counts exceeding 10). Let the virtual time be denoted by t. The simulated counts during this time becomes

$$C = tI, \tag{3}$$

From "Virtual experiments - the ultimate aim of neutron ray-tracing simulations", K. Lefmann et al., Journal of Neutron Research 16, 97-111 (2008)



and its error bar estimate is

$$\sigma^2(C) = t^2\sigma^2(I). \tag{4}$$

However, to simulate a realistic counting statistics, we must fulfill

$$\sigma_{\mathrm{VE}}(C_{\mathrm{VE}}) = \sqrt{C_{\mathrm{VE}}}. \tag{5}$$

This is obtained by adding to (3) a Gaussian noise $E(\Sigma)$ of mean value zero and standard deviation Σ:

$$C_{\mathrm{VE}} = tI + E(\Sigma). \tag{6}$$

The standard deviation for the VE becomes

$$\sigma^2_{\mathrm{VE}}(C) = t^2\sigma^2(I) + \Sigma^2. \tag{7}$$

Now, the requirement (5) allows us to determine Σ:

$$\Sigma^2 = tI - t^2\sigma^2(I). \tag{8}$$

Since Σ^2 must remain positive, we reach an upper limit on t

$$t_{\max} = \frac{I}{\sigma^2(I)}. \tag{9}$$

Sketch of an algorithm…

1. On a given McStas histogram

2. For the non-zero bins, calculate

$$t_{\max} = \frac{I}{\sigma^2(I)}.$$

   The *smallest* $t_{\mathtt{max}}$ defines the "maximal counting time" allowed by your statistics

3. Preferably a "background" should be added - use a "known experimental value" or an estimate…

Monitor_nD

The all-in-one , swiss-army-knife of monitors

Monitor_nD *can have almost any shape, and record any requested standard quantities*

A general monitor for 0D/1D/2D records

Monitor_nD

Examples

```
COMPONENT MyMon = Monitor_nD( xwidth = 0.1, yheight = 0.1, zdepth = 0,
                  options = "intensity per cm2 angle,limits=[-5 5],
                  bins=10,with borders, file = mon1")

            options = "banana, theta limits=[10,130], bins=120, y"

            options = "multiple kx ky kz, auto abs log t, and list all neutrons"
```

Monitor_nD

… or monitor just about anything:

```
COMPONENT MyMon = Monitor_nD(xwidth = 0.1, yheight = 0.1,
    user1=age, username1="Age of the Captain [years]",
    options="user1, auto")
```

Exercise 2:

Head over to the github site and continue the exercise we started before:

https://github.com/McStasMcXtrace/Schools/tree/master/ISIS_April_2021/Tuesday_April_13th/2_Component_Basics/Exercise/