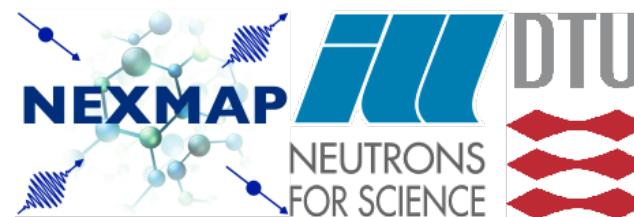
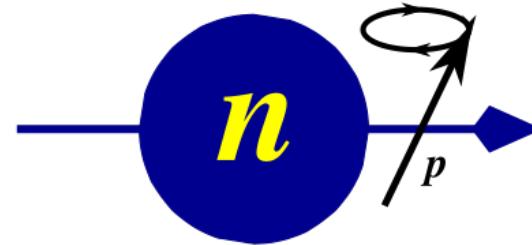


McStas



Simulating Polarized Neutron Scattering Experiments
and Equipment with McStas

Peter Willendrup, DTU Physics & ESS DMSC

Presentation content

- Polarisation in McStas - how is it done
 - Polarised monitors
 - Precession algorithm
- An overview of polarisation comps
- An overview of polarisation instrument examples
- A quick look at 3 of these
- Words of warning

McStas “particle” model

Neutron ray/package:

Weight: (p) # neutrons left in the package

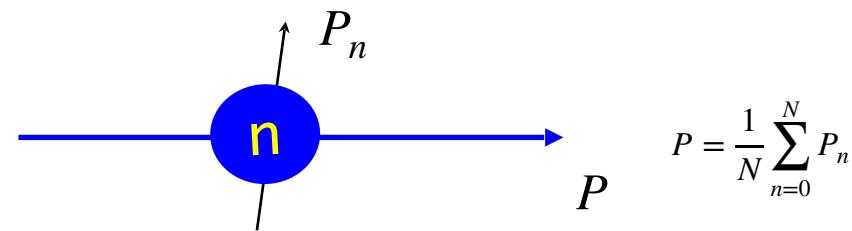
Position: (x, y, z)

Velocity: (v_x, v_y, v_z)

Polarization: (s_x, s_y, s_z)

Time: (t)

$$P_n = \frac{1}{p_n} \sum_i^p P_{i,n}; \quad n = \text{raynumber}$$



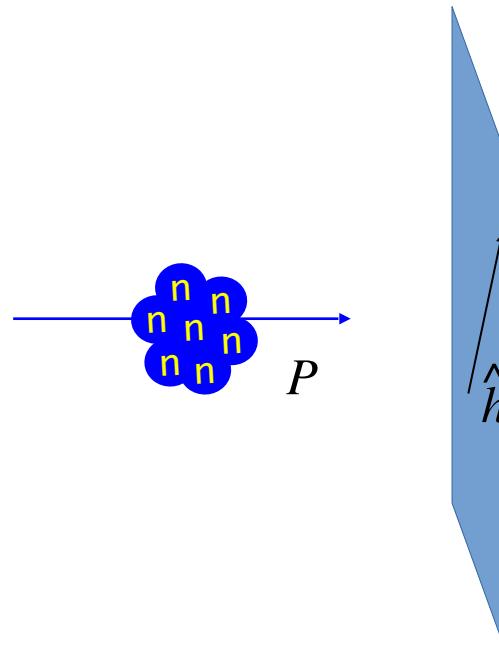
$$P = \frac{1}{N} \sum_{n=0}^N P_n$$

$$P_{i,n} = 2 \left(\left\langle \hat{s}_{x,i} \right\rangle \hat{i}_{x,i} + \left\langle \hat{s}_{y,i} \right\rangle \hat{i}_{y,i} + \left\langle \hat{s}_{z,i} \right\rangle \hat{i}_{z,i} \right)$$

From G. Williams: “*Polarized neutrons*”, Oxford Science Publ., 1988

McStas detectors/monitors

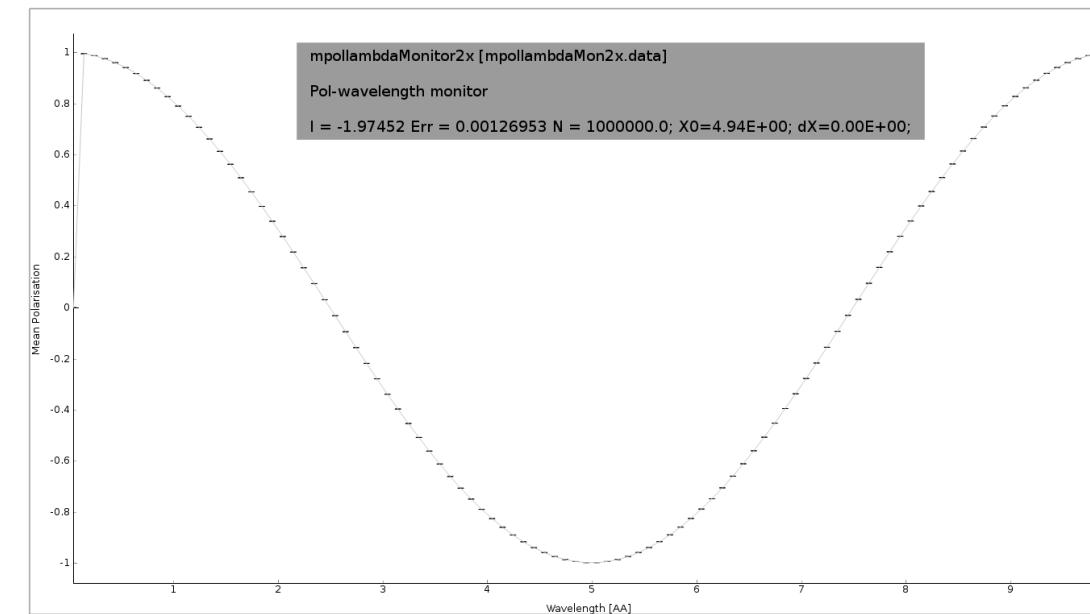
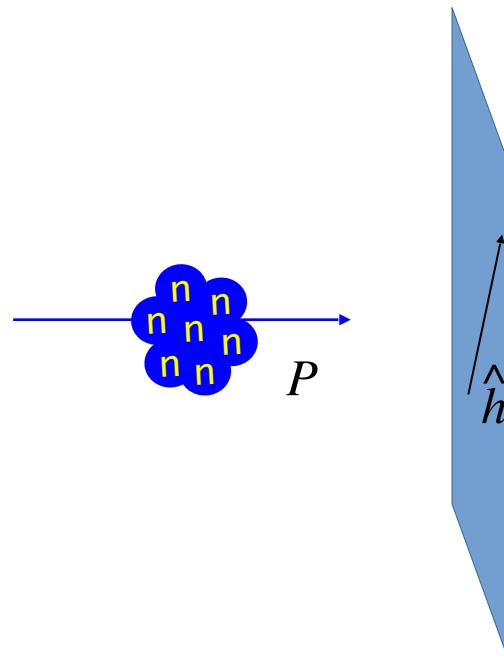
Monitoring: How and What do we monitor?



$$P_{\hat{h}} = \frac{\sum_{n=0}^N p_n P_n \cdot \hat{h}}{\sum_n p_n}$$

McStas detectors/monitors

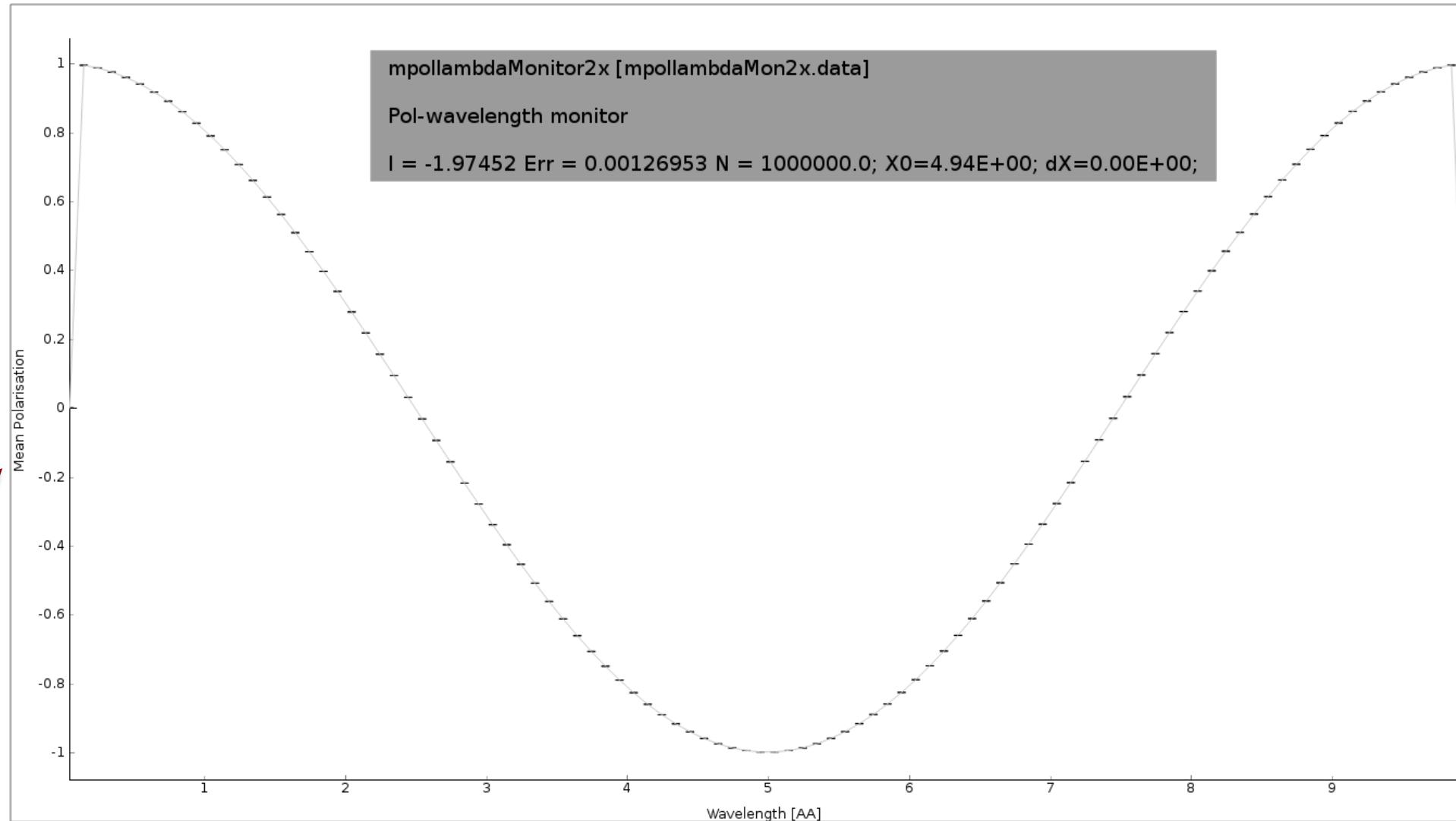
Monitoring: How and What do we monitor?



E.g. polarisation along \hat{h} as fct. of wavelength

McStas detectors/monitors

Monitoring: How and What do we monitor?



Polarisation along \hat{h} as fct. of wavelength

Polarization monitors

- Available monitors:
 - Pol_monitor.comp: 0D
 - PolLambda_monitor.comp: 2D
 - PolTOF_monitor.comp: 2D
 - MeanPolLambda_monitor.comp: 1D

McStas precession algorithm

- Magnetic fields in McStas
- The challenge:
 - Fast beam/ray transport: # rays > 10^6
 - Unknown magnetic field and field strength
 - >1 Magnet → nested fields.

McStas precession algorithm

```

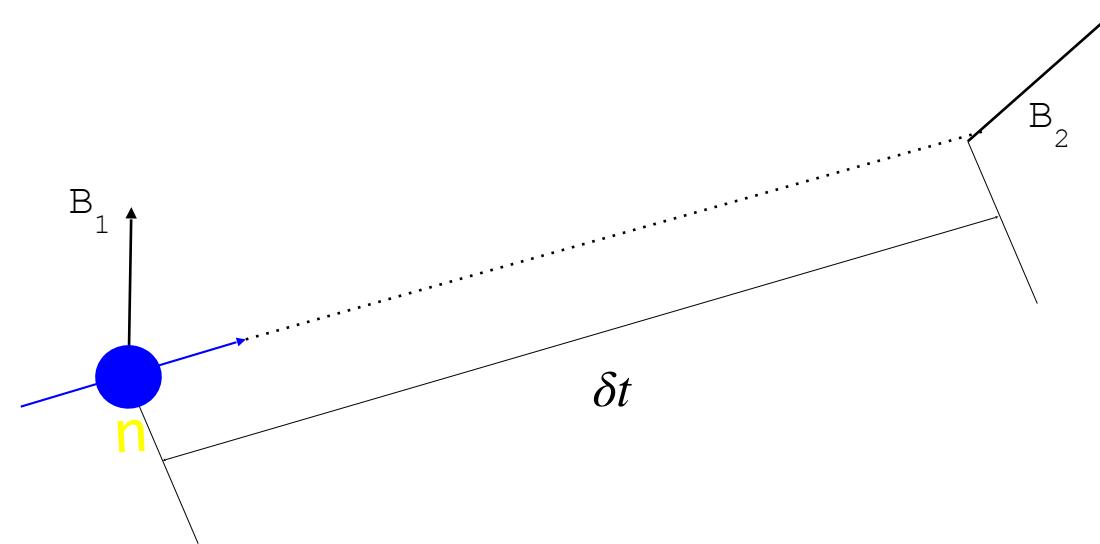
while  $n_t < t_{\text{target}}$  do
    store neutron;
    sample magnetic field:  $\mathbf{B}_1 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
    propagate neutron:  $\delta t(< \Delta t)$ ;
    sample magnetic field:  $\mathbf{B}_2 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
    while  $|\mathbf{B}_1 - \mathbf{B}_2| > \delta B_{\text{threshold}}$  do
        restore neutron;
         $\delta t := \delta t / 2$ ;
        propagate neutron:  $\delta t(< \Delta t)$ ;
        sample magnetic field:  $\mathbf{B}_1 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
        precess polarization:  $\mathbf{P}_n$  by  $\omega$  around  $\frac{\mathbf{B}_1 + \mathbf{B}_2}{2}$ ;

```

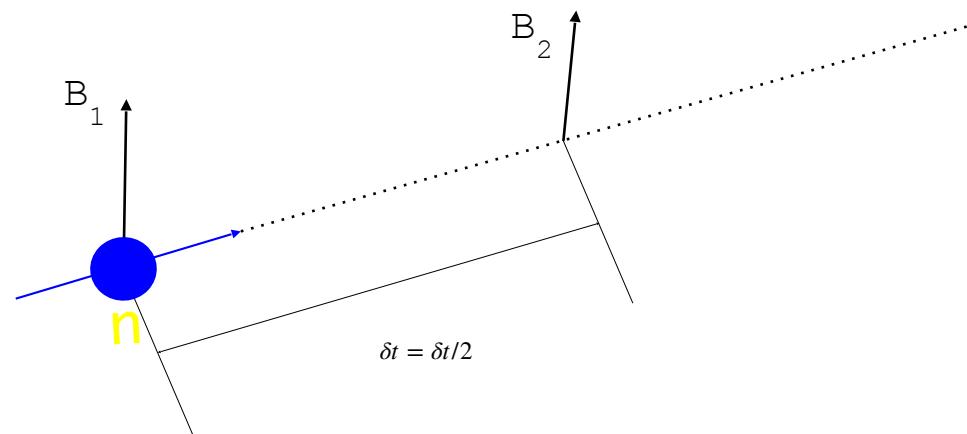
Algorithm 1: SimpleNumMagnetPrecession: Simplistic algorithm for tracking polarization of a Monte-Carlo neutron in a magnetic field. The neutron's state is stored as a position (n_x, n_y, n_z) , a velocity \mathbf{v} , time n_t , and polarization vector \mathbf{P}_n .

From: Knudsen et.al., J. Neutron Research, 2014

McStas precession algorithm



McStas precession algorithm



McStas precession algorithm

```

while  $n_t < t_{\text{target}}$  do
    store neutron;
    sample magnetic field:  $\mathbf{B}_1 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
    propagate neutron:  $\delta t(< \Delta t)$ ;
    sample magnetic field:  $\mathbf{B}_2 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
    while  $|\mathbf{B}_1 - \mathbf{B}_2| > \delta B_{\text{threshold}}$  do
        restore neutron;
         $\delta t := \delta t / 2$ ;
        propagate neutron:  $\delta t(< \Delta t)$ ;
        sample magnetic field:  $\mathbf{B}_1 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
        precess polarization:  $\mathbf{P}_n$  by  $\omega$  around  $\frac{\mathbf{B}_1 + \mathbf{B}_2}{2}$ ;

```

Algorithm 1: SimpleNumMagnetPrecession: Simplistic algorithm for tracking polarization of a Monte-Carlo neutron in a magnetic field. The neutron's state is stored as a position (n_x, n_y, n_z) , a velocity \mathbf{v} , time n_t , and polarization vector \mathbf{P}_n .

From: Knudsen et.al., J. Neutron Research, 2014

McStas precession algorithm

```

while  $n_t < t_{\text{target}}$  do
    store neutron;
    sample magnetic field:  $\mathbf{B}_1 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
    propagate neutron:  $\delta t(< \Delta t)$ ;
    sample magnetic field:  $\mathbf{B}_2 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
    while  $|\mathbf{B}_1 - \mathbf{B}_2| > \delta B_{\text{threshold}}$  do
        restore neutron;
         $\delta t := \delta t / 2$ ;
        propagate neutron:  $\delta t(< \Delta t)$ ;
        sample magnetic field:  $\mathbf{B}_1 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
        precess polarization:  $\mathbf{P}_n$  by  $\omega$  around  $\frac{\mathbf{B}_1 + \mathbf{B}_2}{2}$ ;
    
```

Algorithm 1: SimpleNumMagnetPrecession: Simplistic algorithm for tracking polarization of a Monte-Carlo neutron in a magnetic field. The neutron's state is stored as a position (n_x, n_y, n_z) , a velocity \mathbf{v} , time n_t , and polarization vector \mathbf{P}_n .

From: Knudsen et.al., J. Neutron Research, 2014

McStas precession algorithm

```

while  $n_t < t_{\text{target}}$  do
    store neutron;
    sample magnetic field:  $\mathbf{B}_1 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
    propagate neutron:  $n_t + \Delta n_t$ ;
    sample magnet: void mc_pol_set_angular_accuracy(double domega);
    while  $|\mathbf{B}_1 - \mathbf{B}_2| > \delta B_{\text{threshold}}$  do
        restore neutron;
         $\delta t := \delta t / 2$ ;
        propagate neutron:  $\delta t (< \Delta t)$ ;
        sample magnetic field:  $\mathbf{B}_1 = \mathbf{B}(n_x, n_y, n_z, n_t)$ ;
        precess polarization:  $\mathbf{P}_n$  by  $\omega$  around  $\frac{\mathbf{B}_1 + \mathbf{B}_2}{2}$ ;

```

Algorithm 1: SimpleNumMagnetPrecession: Simplistic algorithm for tracking polarization of a Monte-Carlo neutron in a magnetic field. The neutron's state is stored as a position (n_x, n_y, n_z) , a velocity \mathbf{v} , time n_t , and polarization vector \mathbf{P}_n .

From: Knudsen et.al., J. Neutron Research, 2014

McStas polarization components

Magnetic fields:

- Pol_FieldBox.comp
- (Pol_constBfield.comp 2.x)
- Pol_Bfield.comp
- Pol_Bfield_stop.comp
- Pol_triafield.comp
- (Pol_tabled_field 3.x)

Optics:

- Monochromator_pol.comp
- Pol_bender.comp
- Pol_guide_mirror.comp
- Pol_guide_vmirror.comp
- Pol_mirror.comp
- Transmission_polarisatorABSnT.comp
- Pol_bender_tapering.comp

Monitors:

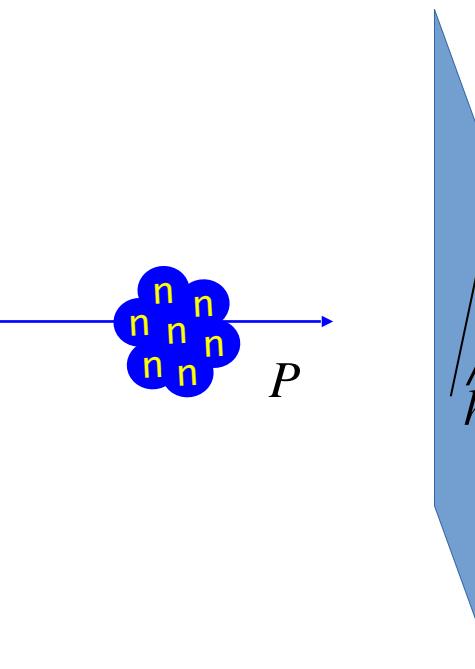
- Pol_monitor.comp
- MeanPolLambda_monitor.comp
- PolLambda_monitor.comp
- PolTOF_monitor.comp

Idealized components:

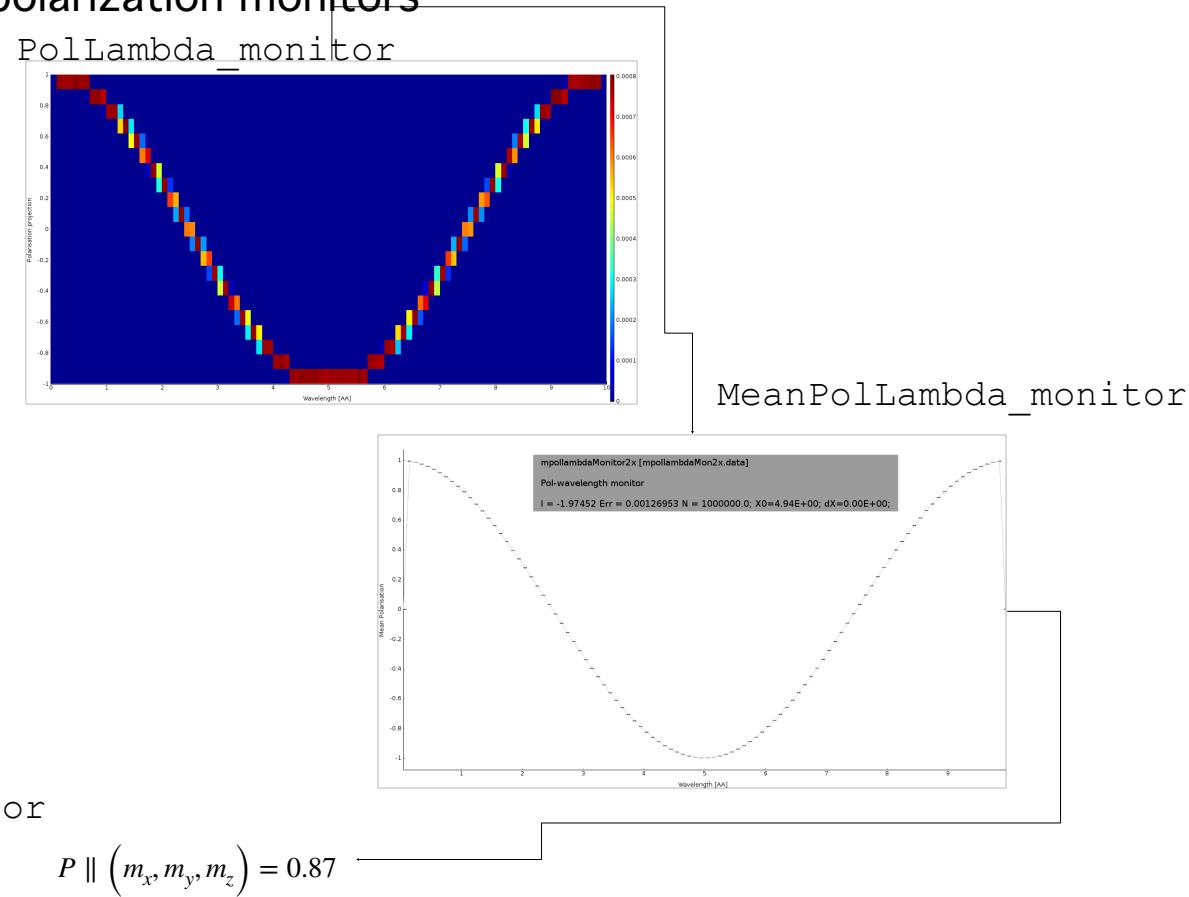
- PolAnalyser_ideal.comp
- Pol_SF_ideal.comp
- Pol_pi_2_rotator.comp
- Set_pol.comp

Contrib:

- Foil_flipper_magnet.comp

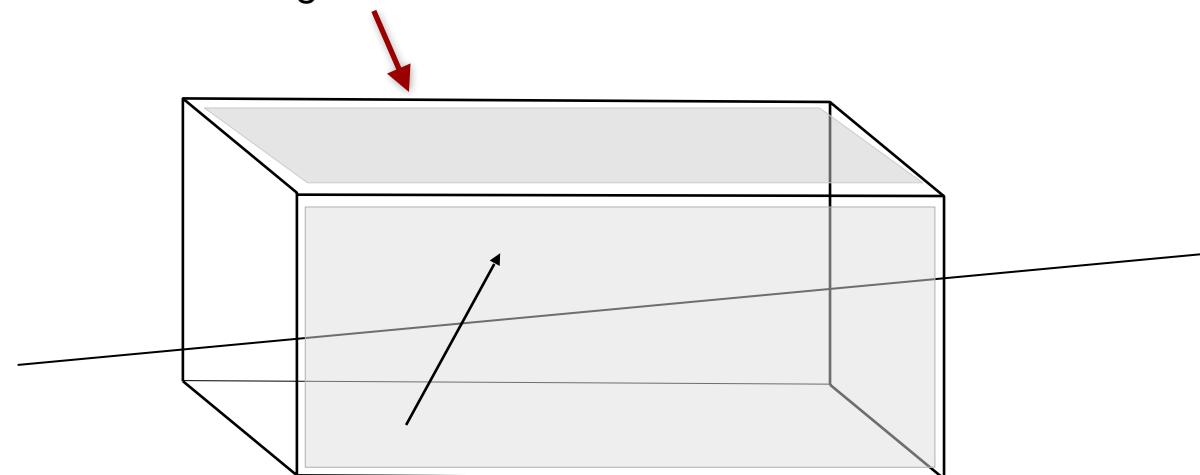


McStas polarization monitors



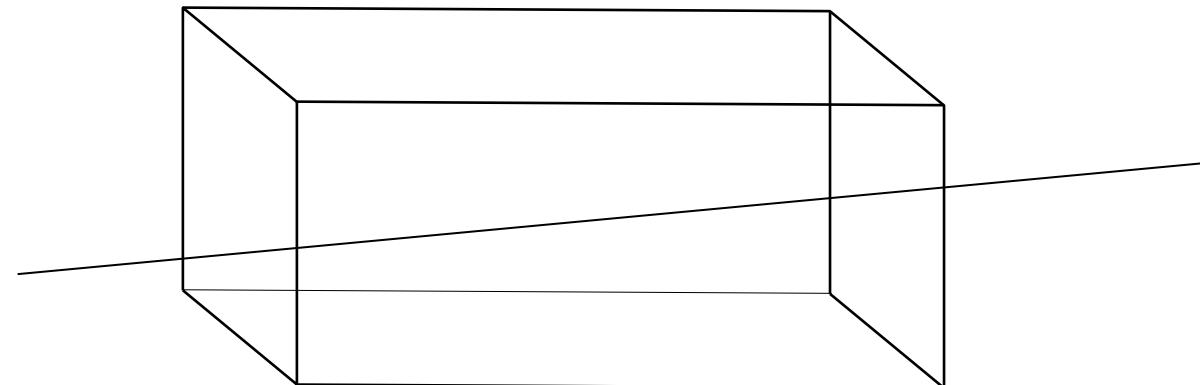
McStas magnetic fields

- Pol_constBfield.comp
- Single constant Magnetic field in a “box”.
- - user may specify a wavelength to flip.
 - “blocking walls”



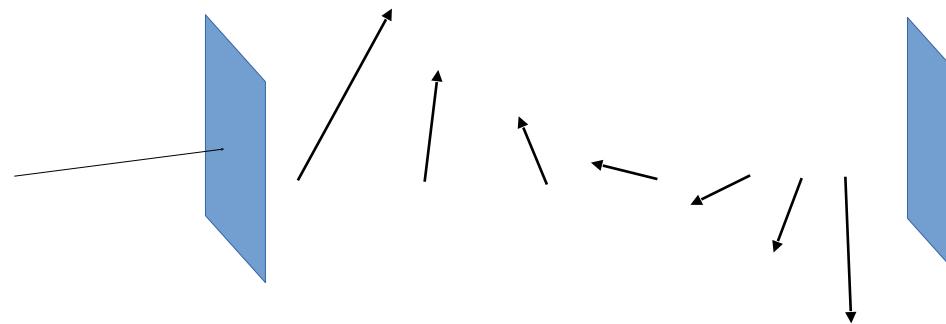
McStas magnetic fields

- Pol_FieldBox.comp
- Single Magnetic field in a “box”
- Constant or tabled magnetic fields



McStas magnetic fields

- Pol_Bfield.comp
- Pol_Bfield_stop.comp
 - Entry/Exit construction allows for nested magnetic field descriptions.
 - Any magnetic field through user supplied c-function
 - Tabled magnetic fields



Standard field types:

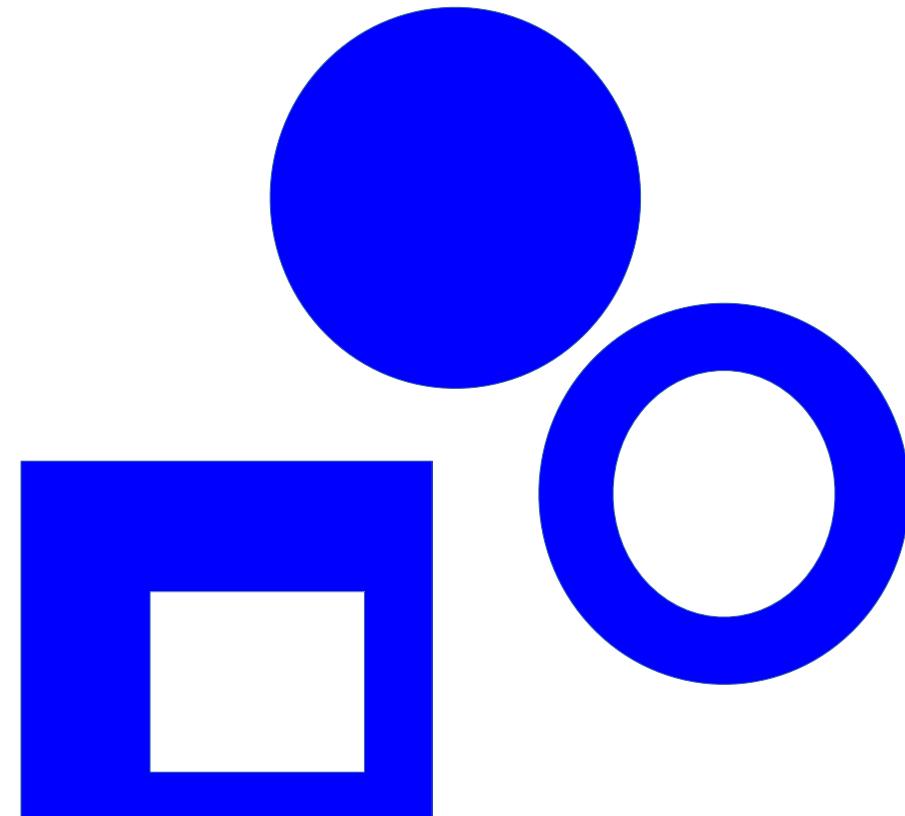
- Constant field
- Rotating field
- Gradient field
- “Majorana” type field

Plus user-defined fields (McStas 2.x only)

See pol-lib.c in share/ and the examples

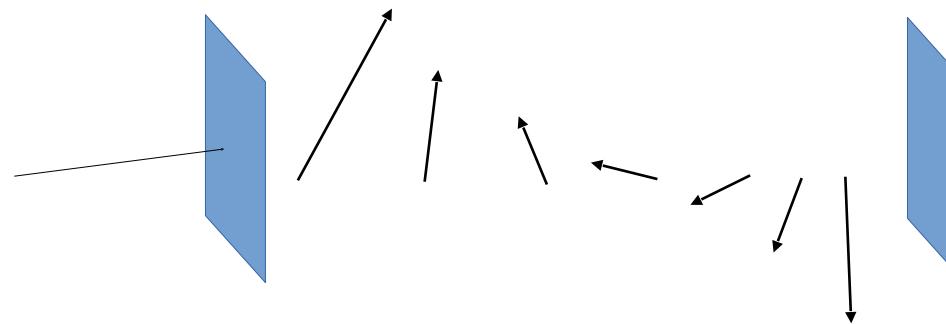
Windows can be many shapes

B-Fields: constant, functional, tabled, ... in
more general shapes



McStas magnetic fields

- Pol_Bfield.comp
- Pol_Bfield_stop.comp
 - Entry/Exit construction allows for nested magnetic field descriptions.
 - Any magnetic field through user supplied c-function
 - Tabled magnetic fields



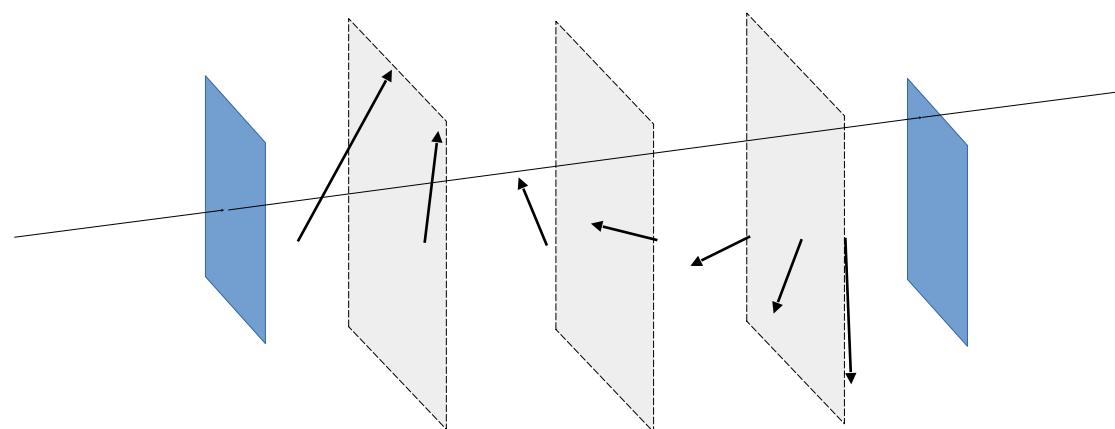
Standard field types:

- Constant field
- Rotating field
- Gradient field
- “Majorana” type field

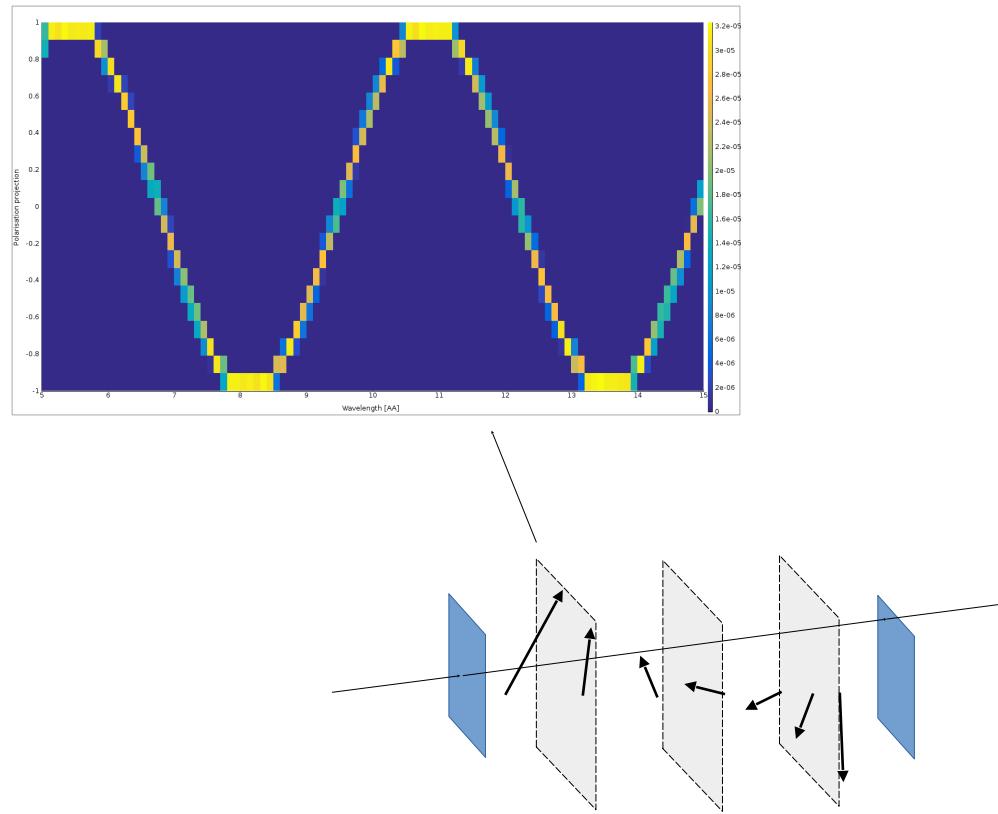
Plus user-defined fields (McStas 2.x only)

See pol-lib.c in share/ and the examples

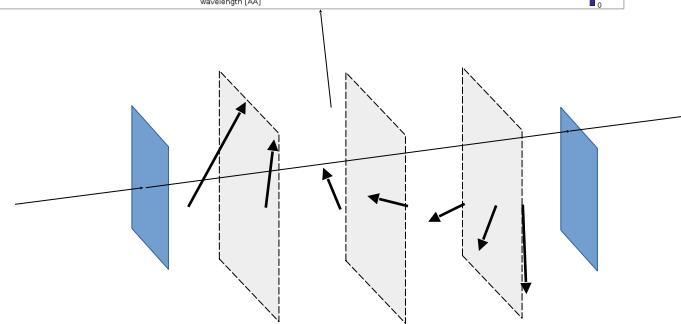
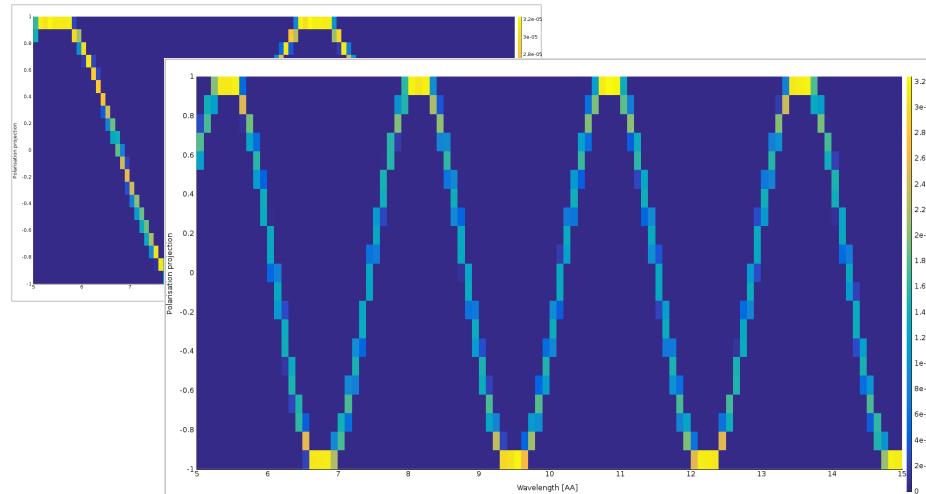
Pol_monitors along the way...



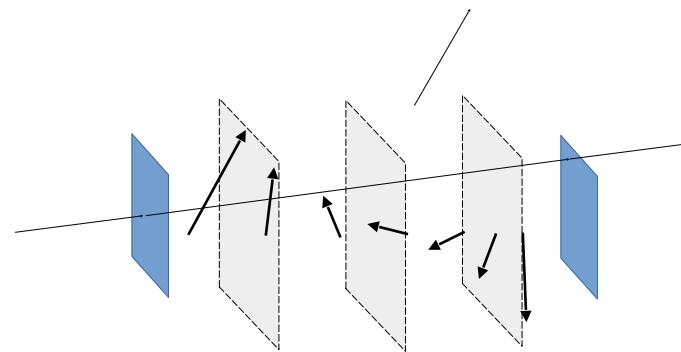
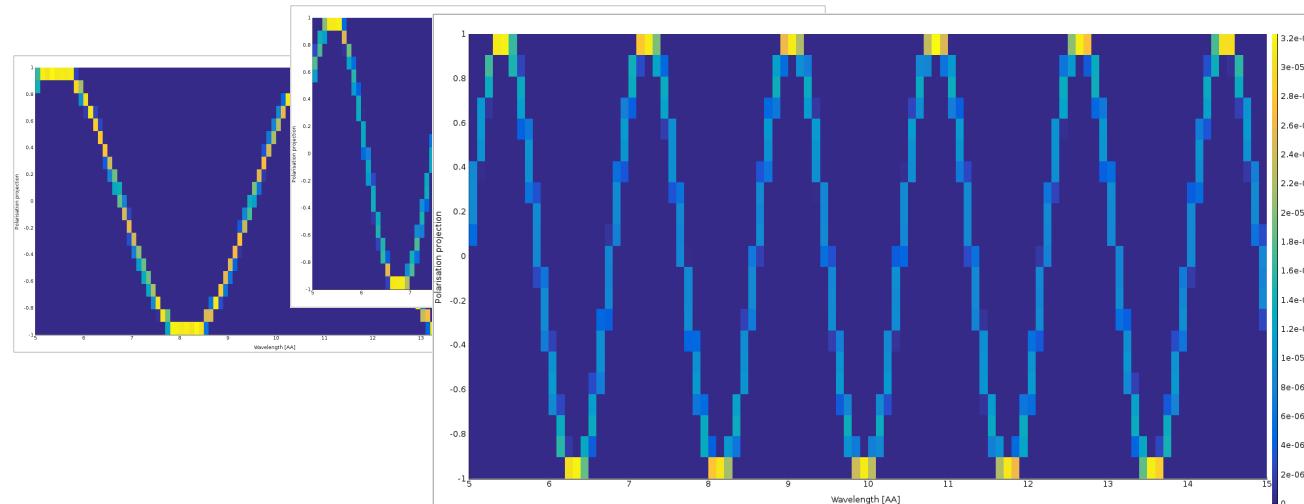
Pol_monitors along the way...



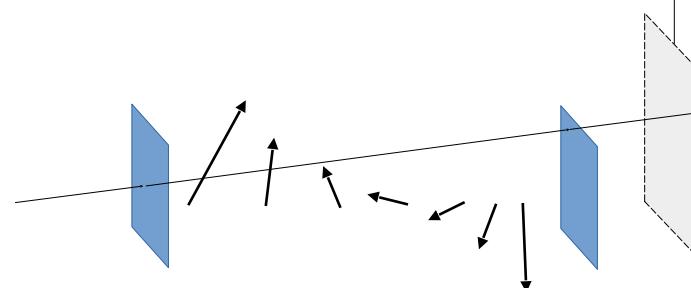
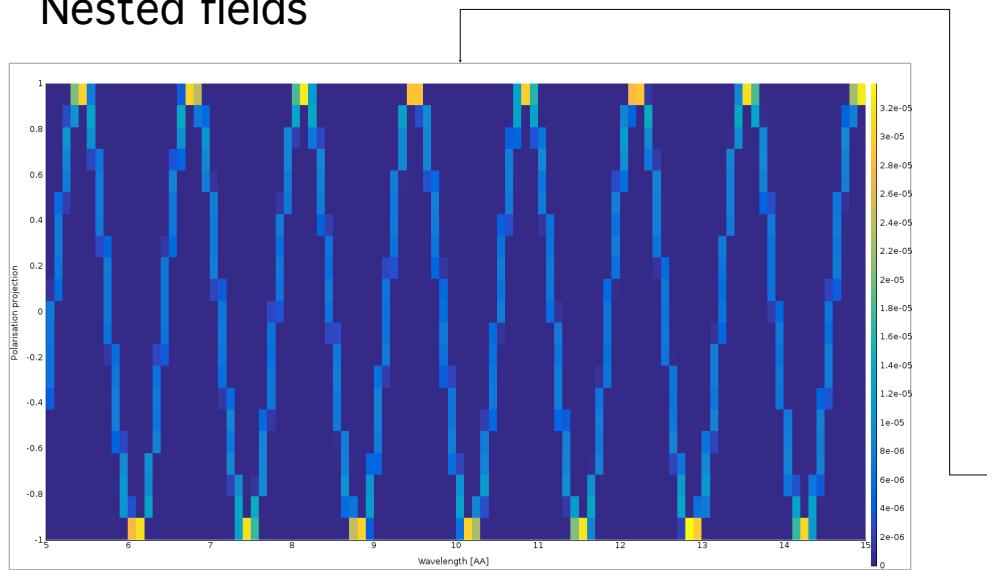
Pol_monitors along the way...



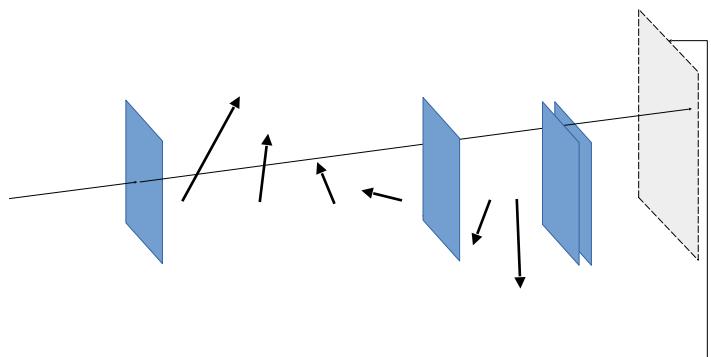
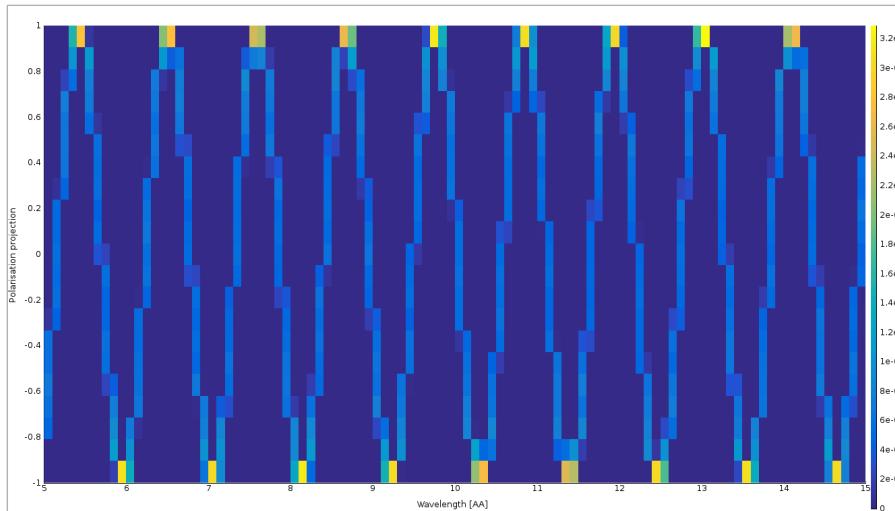
Pol_monitors along the way...



Nested fields



Nested fields



Simple McStas sample component

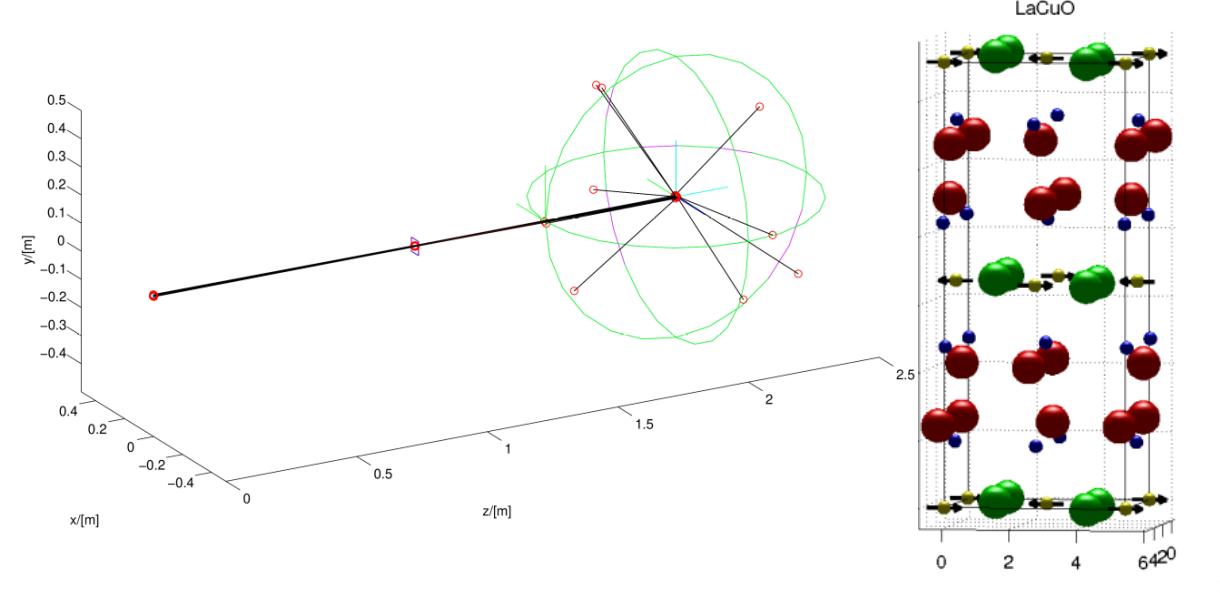
Incoherent.comp has SF / NSF solution

```
/* Polarisation part (1/3 NSF, 2/3 SF) */
sx *= -1.0/3.0;
sy *= -1.0/3.0;
sz *= -1.0/3.0;

SCATTER;
```

McStas sample component (2.x only)

Magnetic single crystal

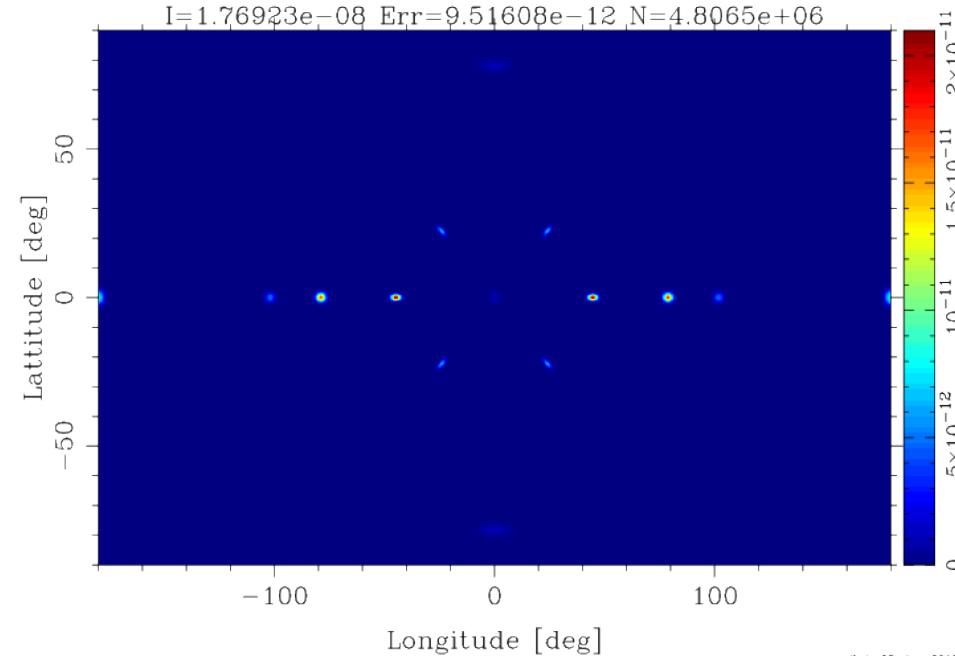


index	iontype	x	y	z	b_{coh} [fm]	g_S	S_x	S_y	S_z	g_L	L_x	L_y	L_z
1	Cu ²⁺	0.5	0.5	0	7.718	2	0	-0.5	0	0	0	0	0
:	:	:	:	:	:	:	:	:	:	:	:	:	:

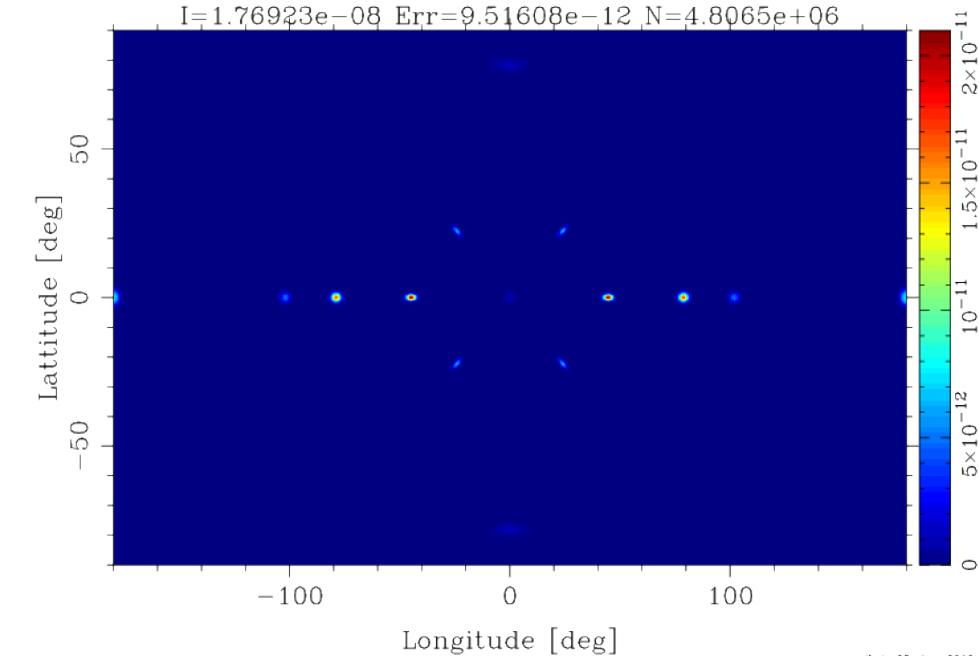
McStas sample component (2.x only)

Magnetic single crystal – Unpolarized beam

4PImon_spinup [250110_SF_NSF_PX0_PY0_PZ0_1e10/PSD4PImon_spinup.
 X0=-0.104202; dX=88.4169; Y0=0.105552; dY=25.2284;
 I=1.76923e-08 Err=9.51608e-12 N=4.8065e+06]

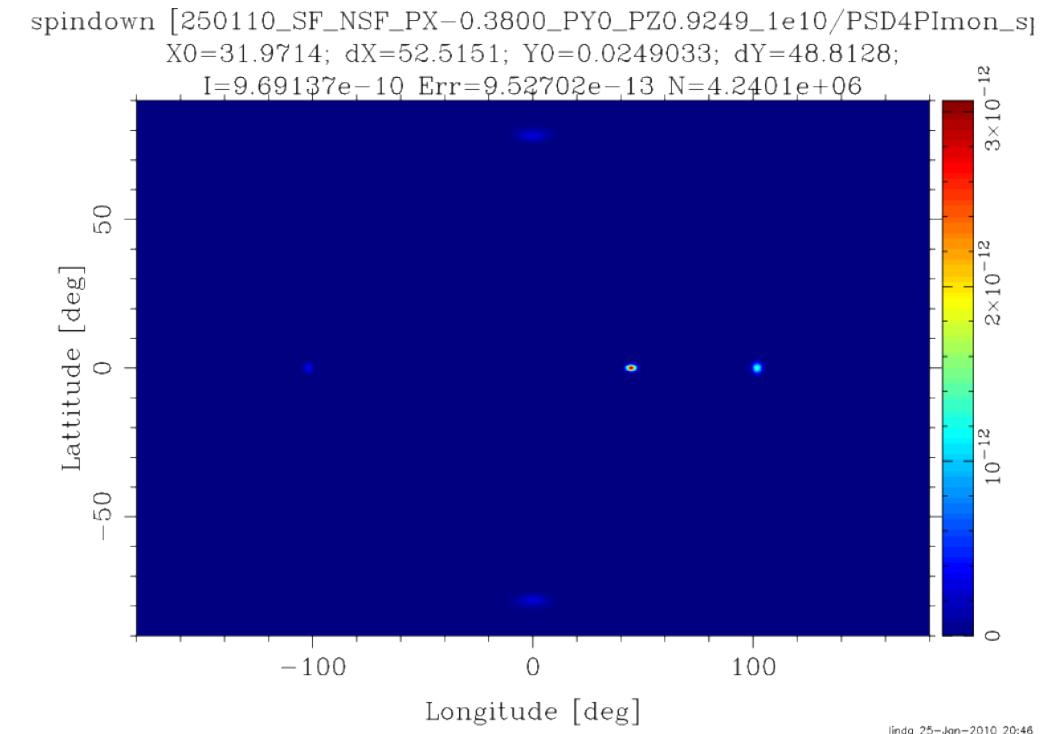
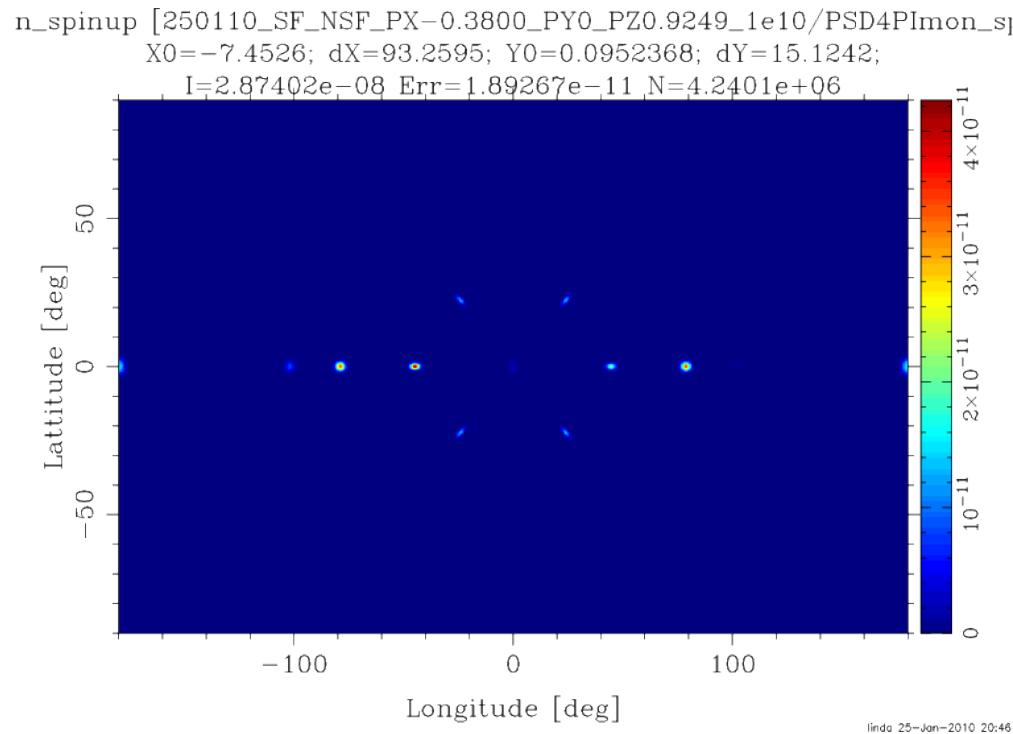


4PImon_spindown [250110_SF_NSF_PX0_PY0_PZ0_1e10/PSD4PImon_spindow.
 X0=-0.104202; dX=88.4169; Y0=0.105552; dY=25.2284;
 I=1.76923e-08 Err=9.51608e-12 N=4.8065e+06]



McStas sample component (2.x only)

Magnetic single crystal – Polarized beam



McStas sample component (2.x only)

Magnetic single crystal

The magnetic scattering cross-section for a sample with localised spin+orbital angular moment $g\mathbf{J} = (g_S + g_L)\mathbf{J} = 2\mathbf{S} + \mathbf{L}$ is:

$$\frac{d^2\sigma}{d\Omega_f dE_f} = \frac{k_f}{k_i} \sum_{i,f} P(\lambda_i) \left| \langle \lambda_f | \sum_j e^{i\mathbf{Q} \cdot \mathbf{d}_j} U_j^{\sigma_i \sigma_f} | \lambda_i \rangle \right|^2 \delta(\hbar\omega + E_i - E_f)$$

where $|\lambda_i\rangle$ and $\langle\lambda_f|$ are the initial and final states of the sample with energies E_i and E_f respectively, $P(\lambda_i)$ is the distribution of initial states and

$$U_j^{\sigma_i \sigma_f} = \langle \sigma_f | b_j - m_j \mathbf{J}_{\perp j} \cdot \boldsymbol{\sigma} | \sigma_i \rangle$$

where $|\sigma_i\rangle$ and $\langle\sigma_f|$ are the initial and final spin states of the neutron, and $\boldsymbol{\sigma}$ are the Pauli spin matrices working on the neutron state.

From: G. Shirane et.al. ,”Neutron Scattering with Triple-Axis Spectrometer”, Cambridge Univ. Press, 2002

McStas sample component (2.x only)

Magnetic single crystal

If $\mathbf{P} = P(\xi, \eta, \zeta) = P\hat{\zeta}$. Thus, the matrix elements of $U^{\sigma_i \sigma_f}$ can now be written

$$\begin{aligned} U^{++} &= b - mJ_{\perp\zeta} \\ U^{--} &= b + mJ_{\perp\zeta} \\ U^{+-} &= -m(J_{\perp\xi} + iJ_{\perp\eta}) \\ U^{+-} &= -m(J_{\perp\xi} - iJ_{\perp\eta}) \end{aligned}$$

where $m = \frac{r_0\gamma}{2}gf(\mathbf{Q})$ with r_0 the classical electron radius, $\gamma = 1.913$, g the Landé splitting factor and $f(\mathbf{Q})$ the magnetic form factor of a particular ion in the sample.

Example instruments:

Magnetic.instr:

```
Test_Magnetic_Constant.instr  
Test_Magnetic_Majorana.instr  
Test_Magnetic_Rotation.instr  
Test_Magnetic_Userdefined.instr (2.x only)  
Test_single_magnetic_crystal.instr (2.x only)
```

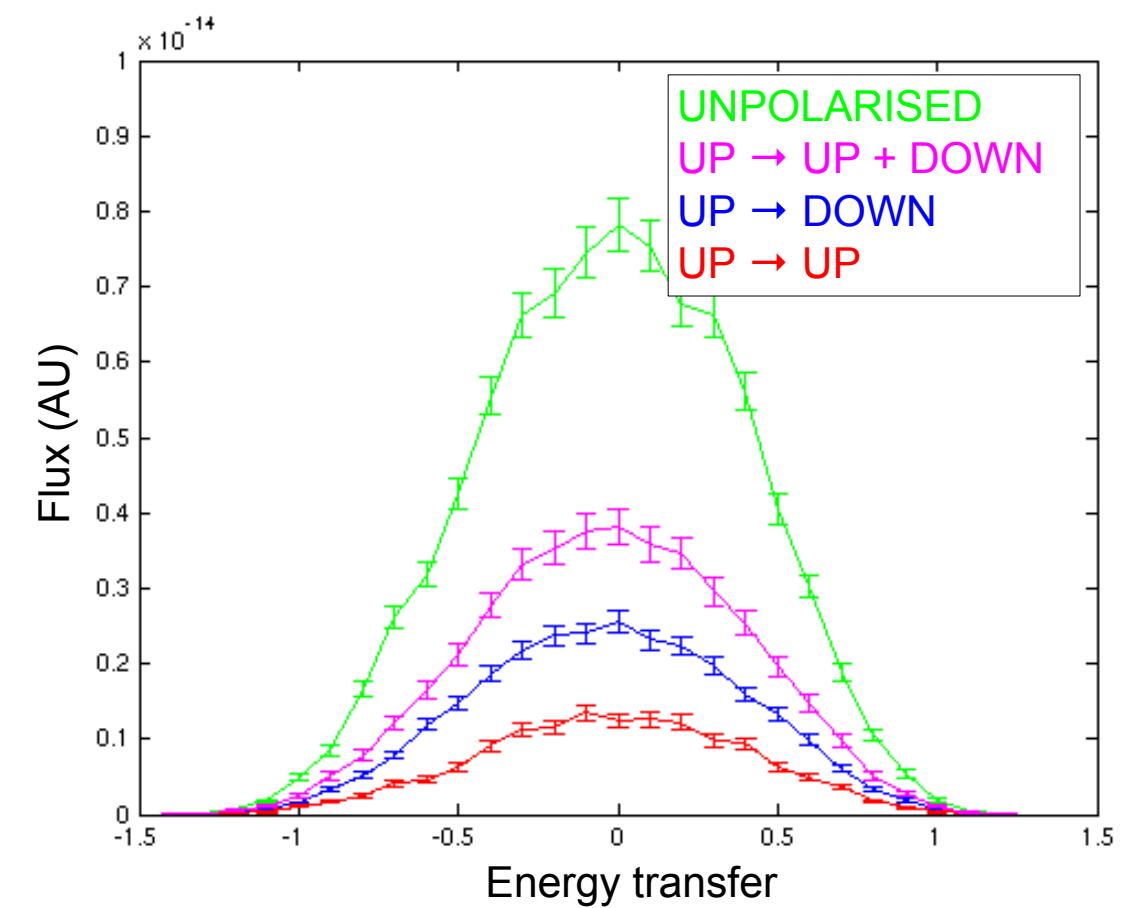
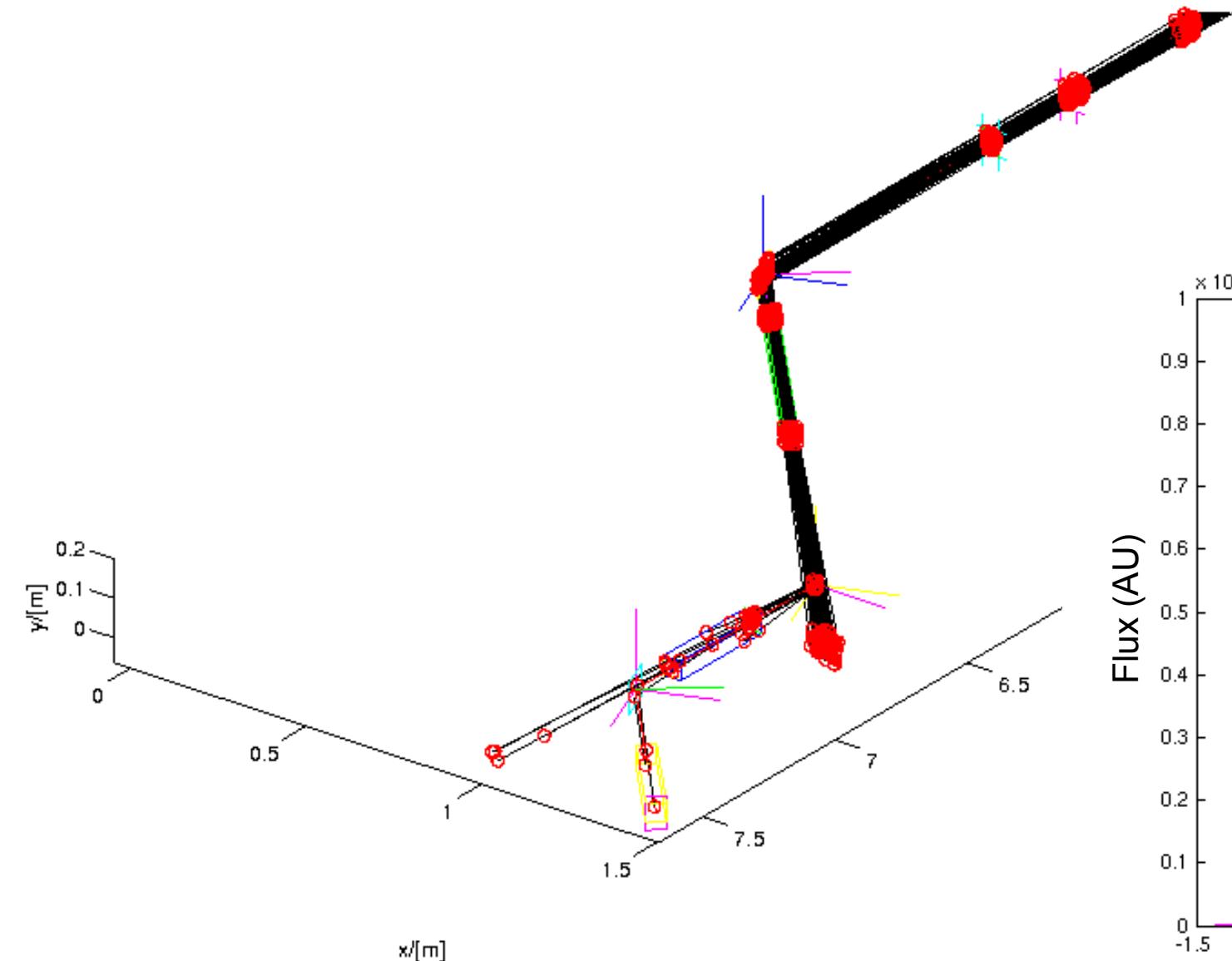
SE*.instr:

```
SEMSANS_Delft.instr  
SEMSANS_instrument.instr  
SESANS_Delft.instr  
SE_example.instr  
SE_example2.instr
```

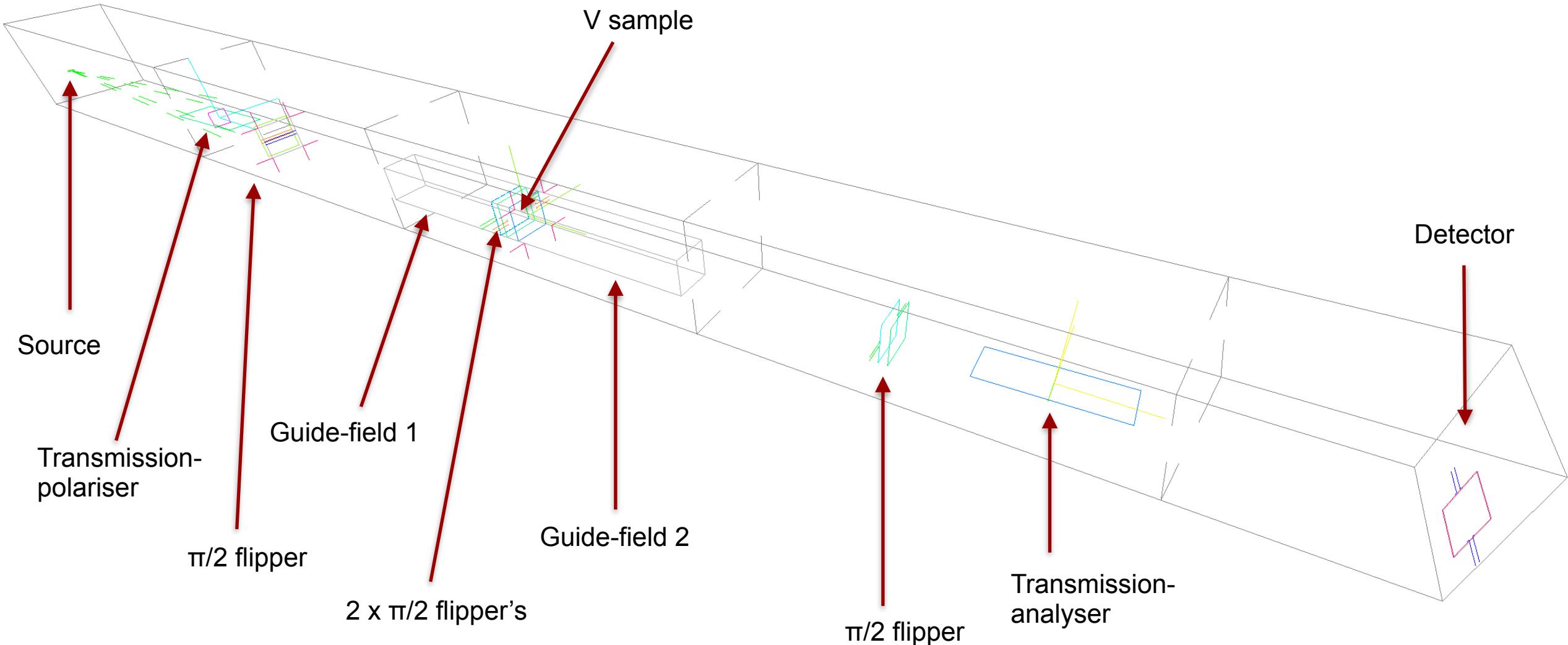
Pol.instr:

```
Test_Pol_Bender.instr  
Test_Pol_Bender_Vs_Guide_Curved.instr  
Test_Pol_FieldBox.instr  
Test_Pol_Guide_Vmirror.instr  
Test_Pol_Guide_mirror.instr  
Test_Pol_MSF.instr  
Test_Pol_Mirror.instr  
Test_Pol_SF_ideal.instr  
Test_Pol_Set.instr  
Test_Pol_Tabled.instr  
Test_Pol_TripleAxis.instr
```

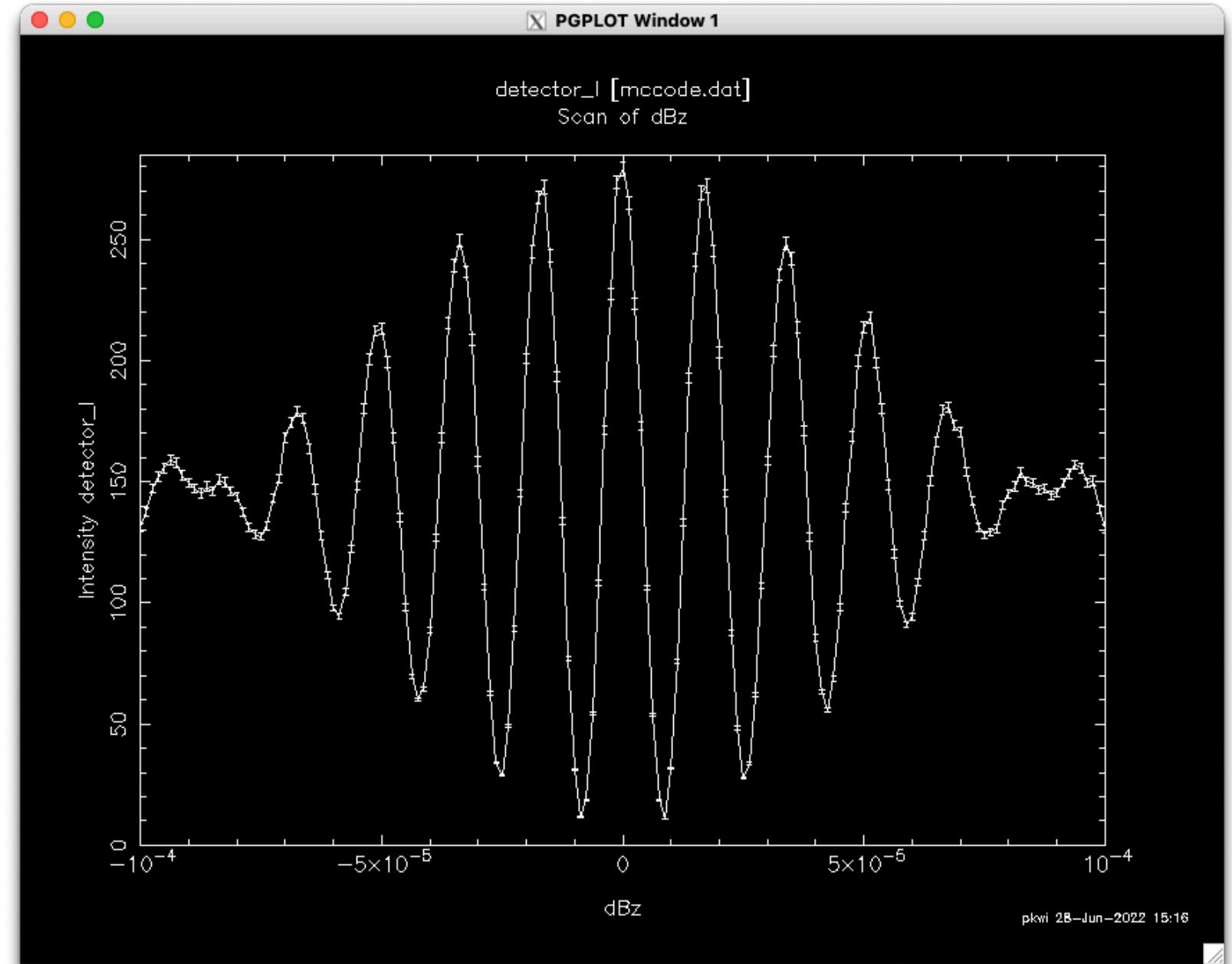
McStas: IN20 TAS model – comparing signals from Vanadium



Generic spin-echo SE_example2.instr

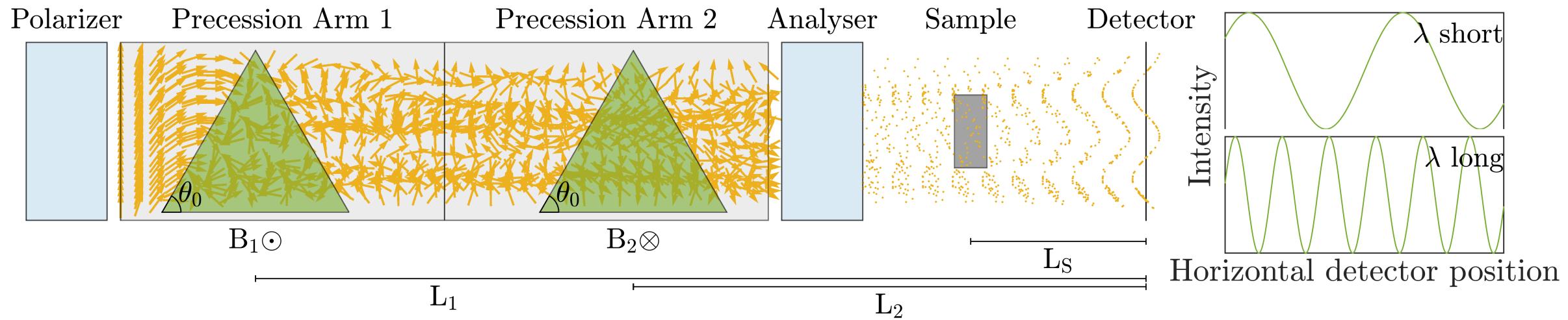


Tuning of guide-field 1 around its value allows us to see spin-echo:



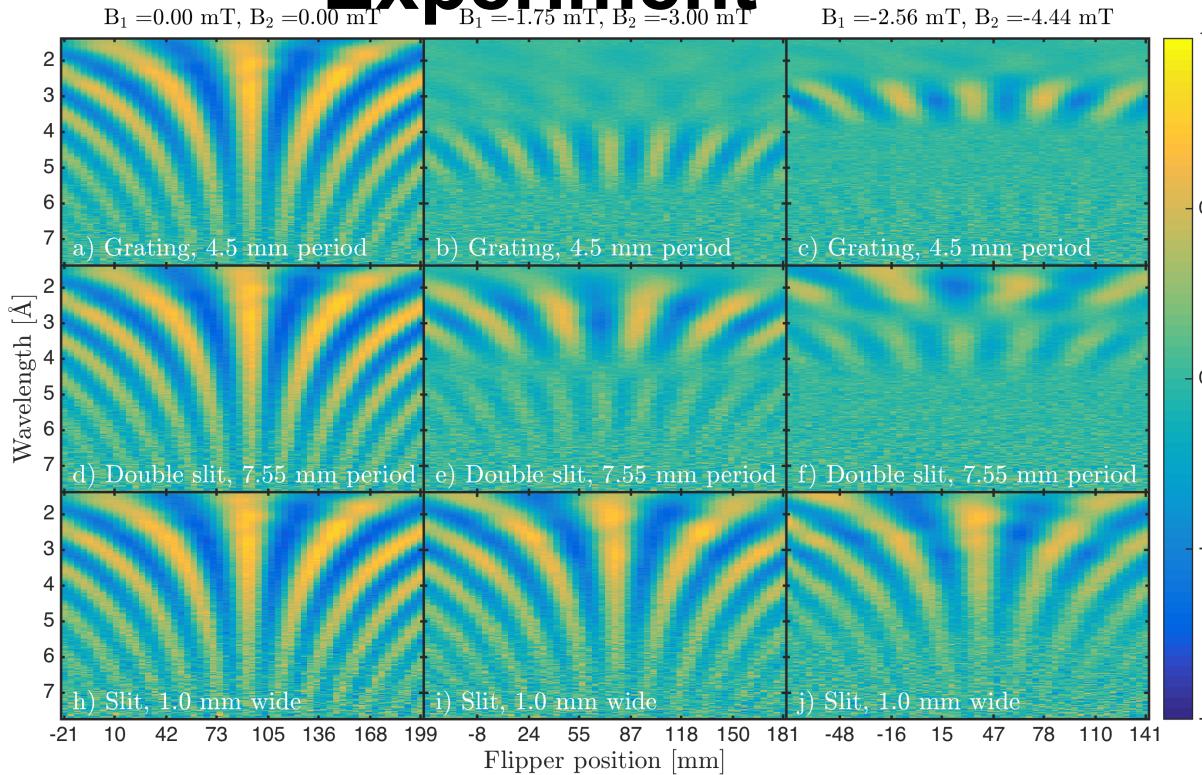
Highlight: McStas example SEMSANS

Courtesy: M. Sales et.al.

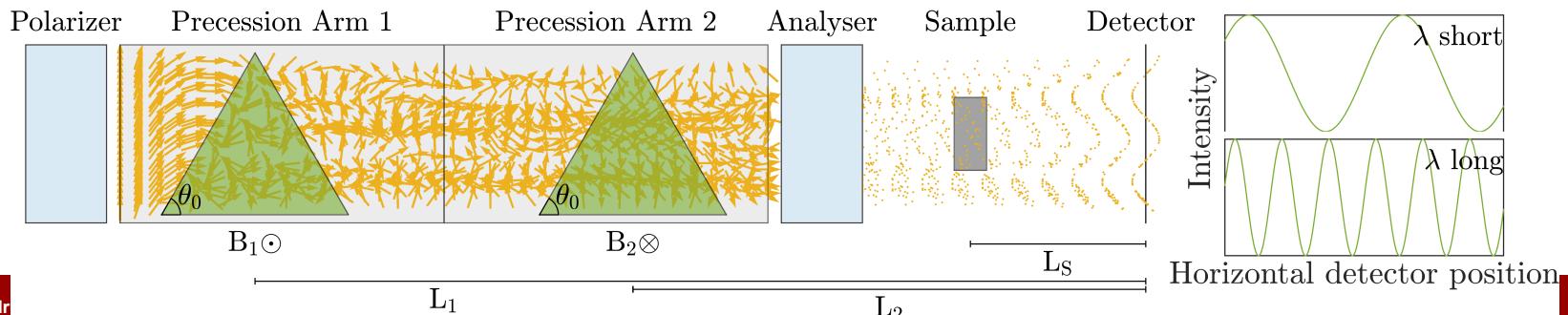
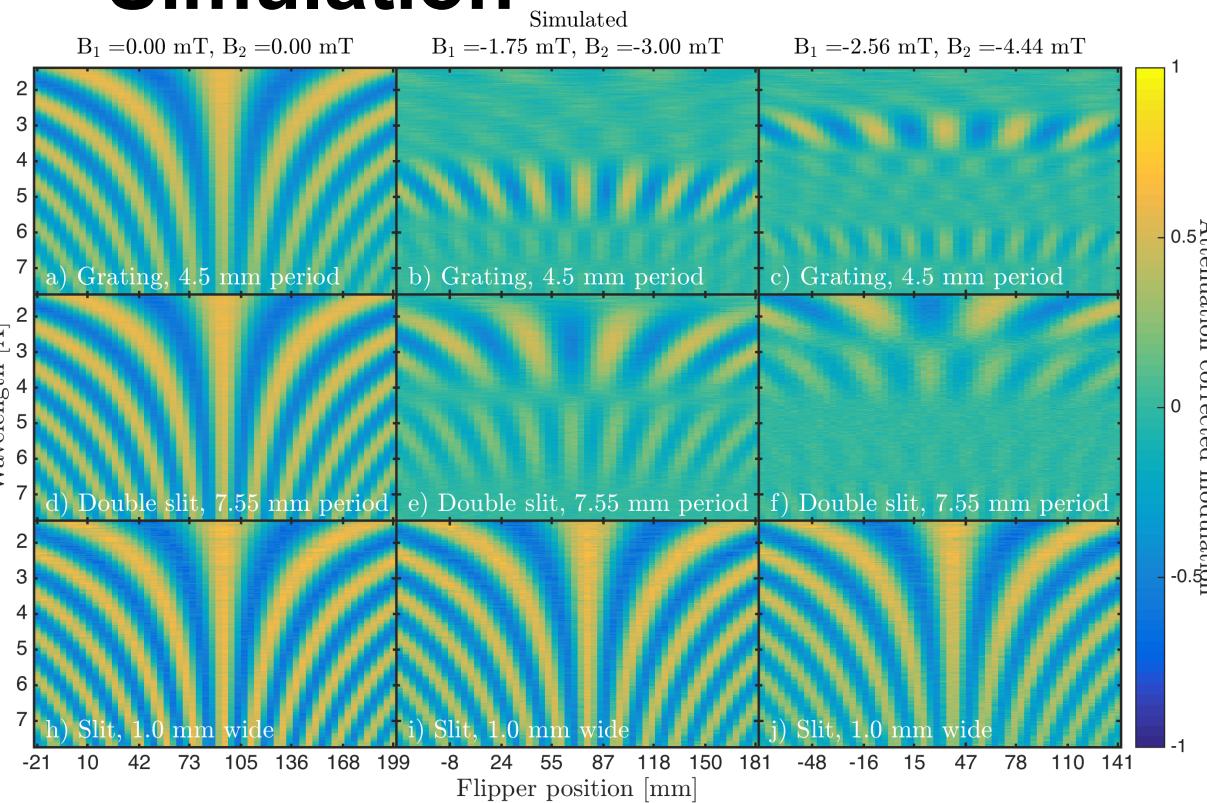


Highlight: McStas example SEMSANS

Experiment



Simulation



Gotcha's

- Polarisation docs in manual are out of date wrt. the current code/algorithm...
- For now, don't combine gravity and polarisation!
(Guide_gravity and Elliptic_guide_gravity can currently not be used within field regions.)
- Memory-leak found in McStas 2.7.1 when using precession regions
(Patch-release later this year, for now patch by using [this updated pol-lib.c](#))
- McStas 3.x does not support
 - User-defined field functions
(No function pointers on GPU... code-generator may rescue us later)
 - Having optics within tabled fields
(tabled fields are for now only available in Pol_tabled_field which can not work in "concentric geometries")
- GitHub issues exist or will be written, these things will eventually all be fixed!