McStas and other codes for neutron delivery and full instrument simulations

Peter Willendrup[1,5], Emmanuel Farhi[2], Erik Knudsen[1,5], Uwe Filges[3,6], Linda Udby[4,5], Kim Lefmann[4,5]

[1]Physics Department, Technical University of Denmark, Denmark
[2]Calcul Scientifique, Institut Laue-Langevin, France
[3]Laboratory for Developments and Methods, Paul Scherrer Institute, Switzerland
[4]Niels Bohr Institute, University of Copenhagen, Denmark
[5]ESS design update programme, Denmark
[6]ESS design update programme, Switzerland

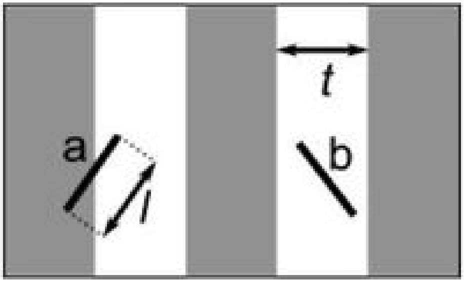
McStas

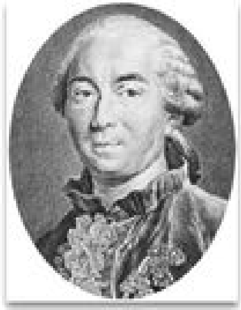
INSIS School, Frascati 2012

Agenda

- An introduction Monte Carlo & raytracing techniques
- An overview of the available packages
- An introduction to McStas
- A tour of the example suite
- Build-along exercises

Numerical experiments before Los Alamos





- In 1777 Georges-Louis Leclerc, comte de Buffon was the first to "numerical experiments" for solving a problem of geometrical probability.
- The experiment involves dropping a needle on a lined surface and can be used to estimate π

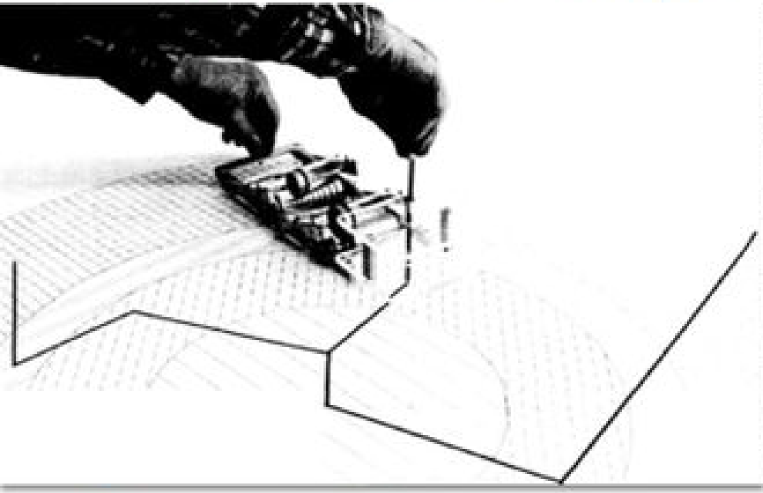
- In the 1930's, Fermi used sampling methods to estimate quantities involved in controlled fission

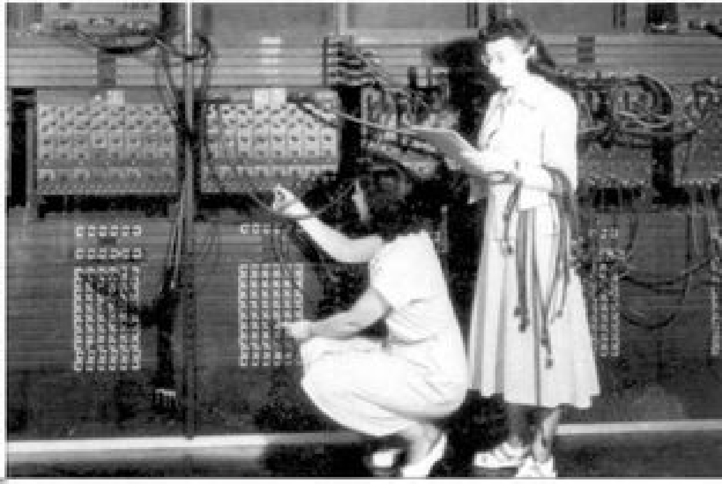
Monte Carlo techniques

- During WW2, "numerical experiments" were applied at Los Alamos for solving mathematical complications of computing fission, criticality, neutronics, hydrodynamics, thermonuclear detonation etc.

- Notable fathers: John v. Neumann Stanislav Ulam Nicholas Metropolis
- Named "Monte Carlo" after Ulam's fathers frequent visits to the Monte Carlo casino in Las Vegas
- Initially "implemented" by letting large numbers of women use tabularized random numbers and hand calculators for individual particle calculations
- Later, analogue and digital computing devices were used

Monte Carlo techniques





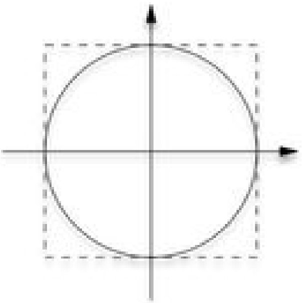
- FERMIAC ENIAC

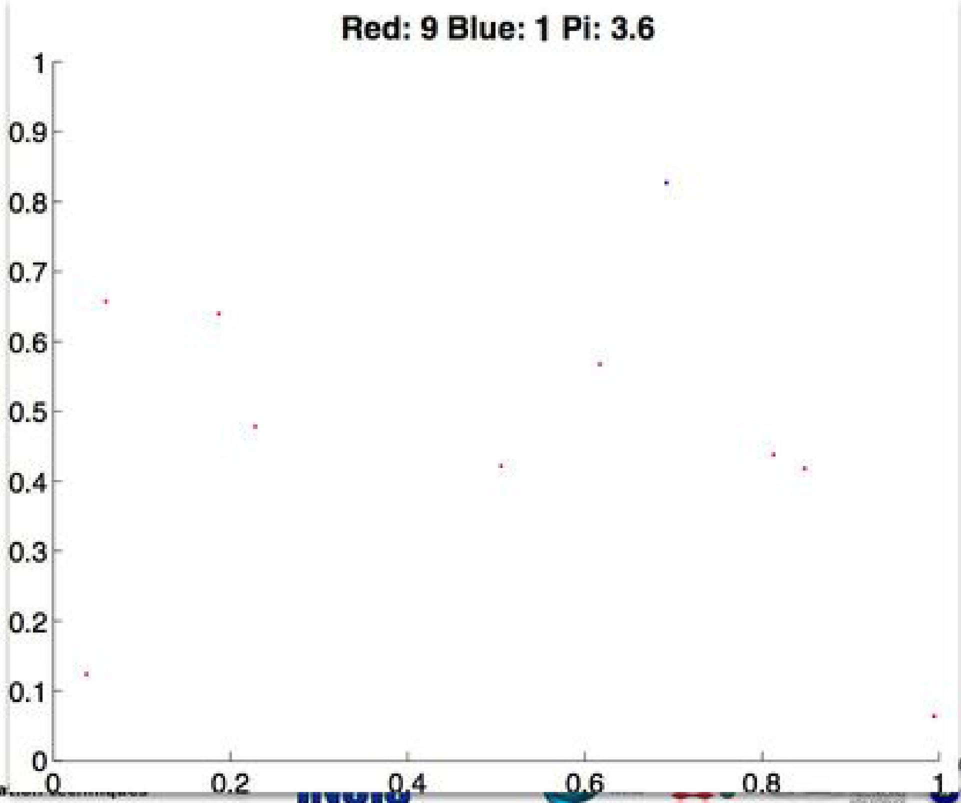
Monte Carlo techniques

- Estimating Pi:

$$\pi = \frac{A_{circ}}{A_{sqr}} = \frac{\pi r^2}{(2r)^2}$$

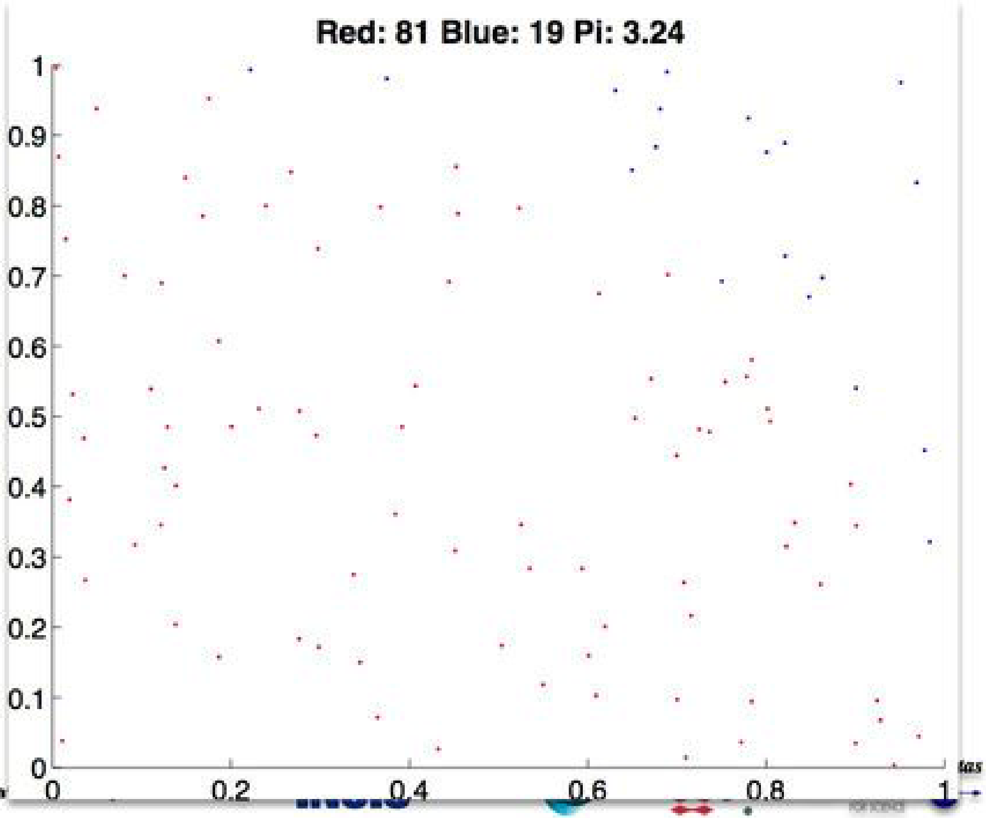
$$\pi \approx 4\frac{9}{9+1} = 3.6$$





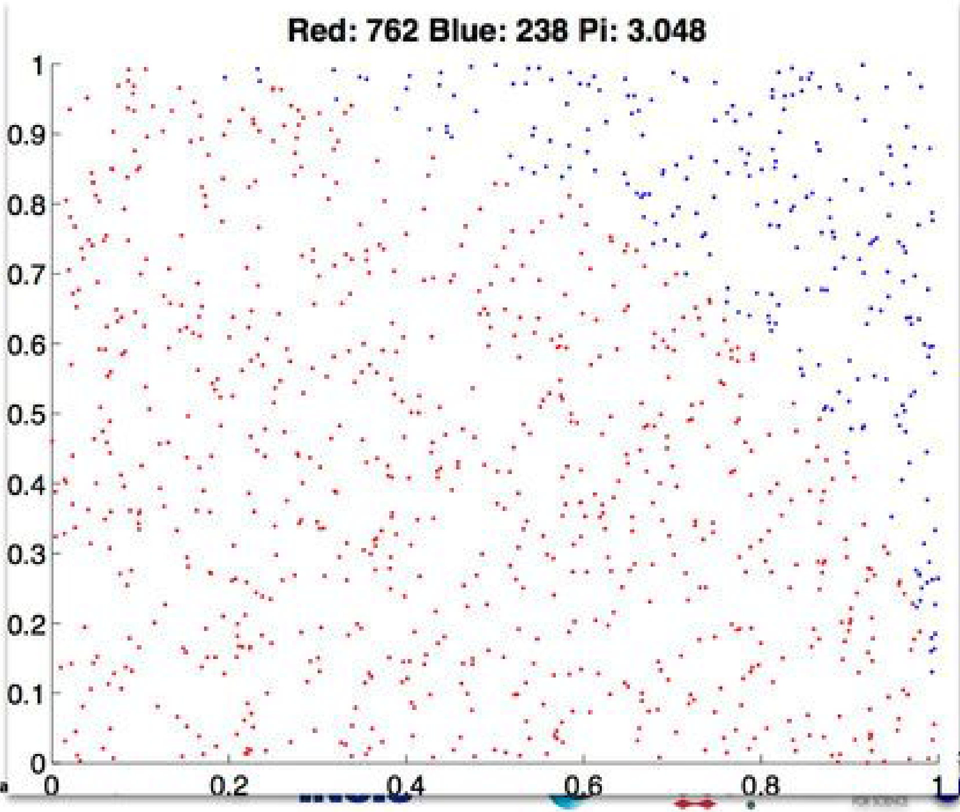
Monte Carlo techniques

- Estimating Pi:



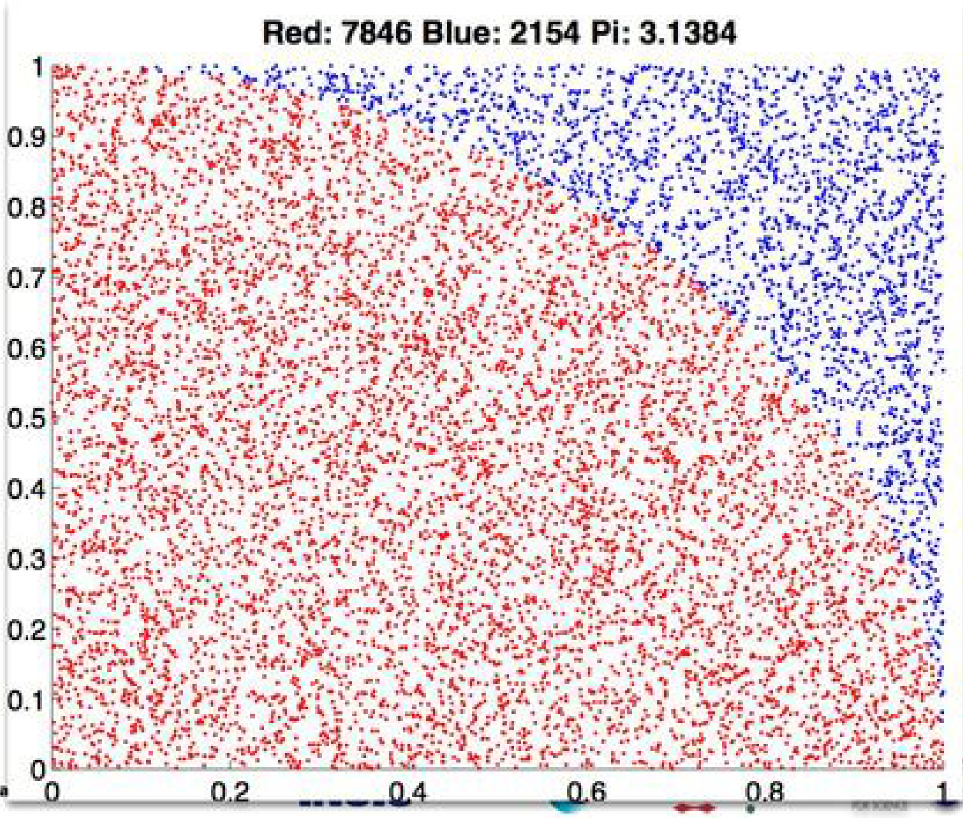
Monte Carlo techniques

- Estimating Pi:



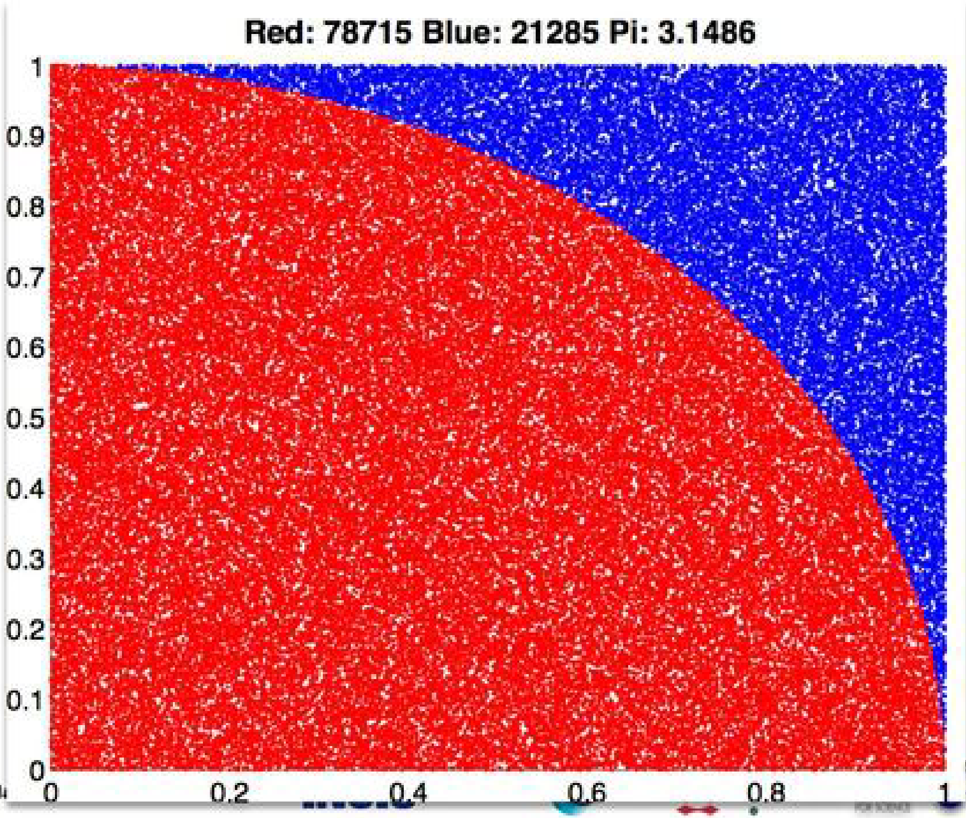
Monte Carlo techniques

- Estimating Pi:



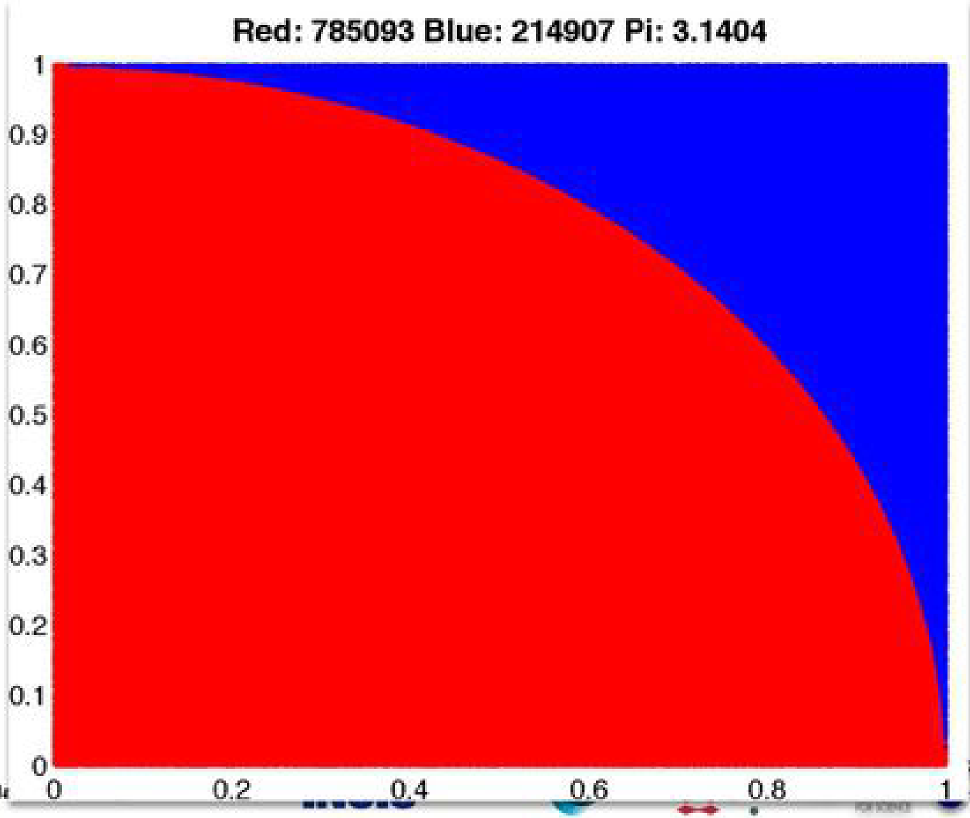
Monte Carlo techniques

- Estimating Pi:



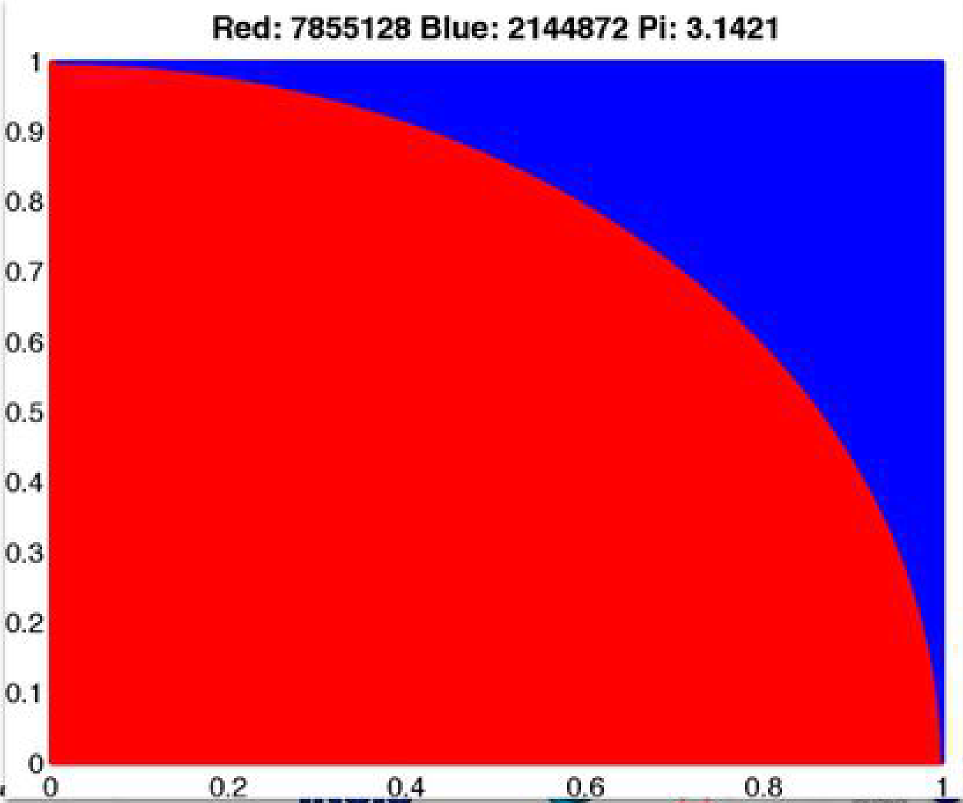
Monte Carlo techniques

- Estimating Pi:



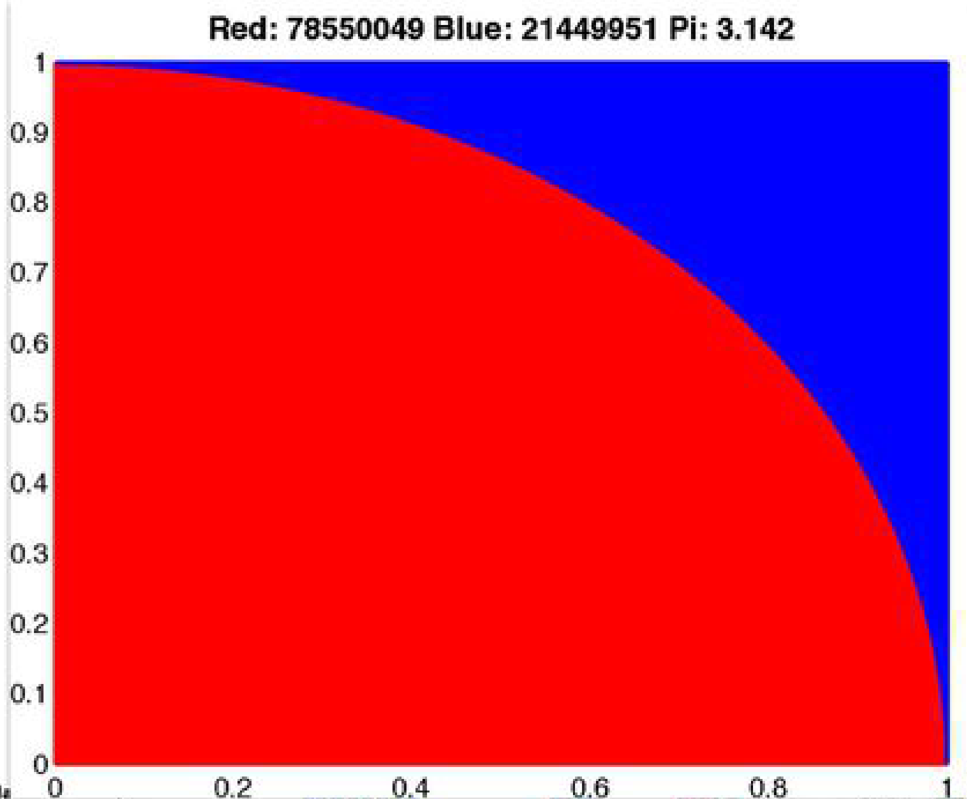
Monte Carlo techniques

- Estimating Pi:



Monte Carlo techniques

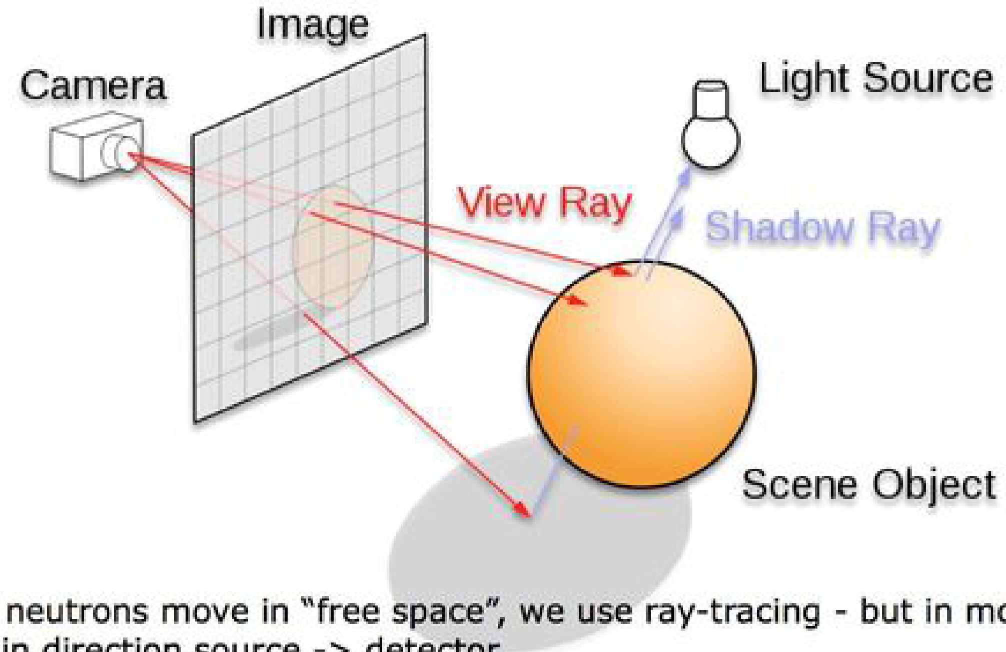
- Estimating Pi:



Monte Carlo techniques

- Los Alamos has since then developed and perfected many different monte carlo codes leading to what is today known as the codes MCNP5 and MCNPX
- State of the art is MCNPX (or soon the merged MCNP6 code) that features numerous (even exotic) particles
- MCNP was originally Monte Carlo Neutron Photon, later N-Particle
- Mainly used for high-energy particle descriptions in weapons, power reactors and routinely used for estimating dose rates and needed shielding
- Does not to date handle coherent scattering of neutrons due to the focus on high energies

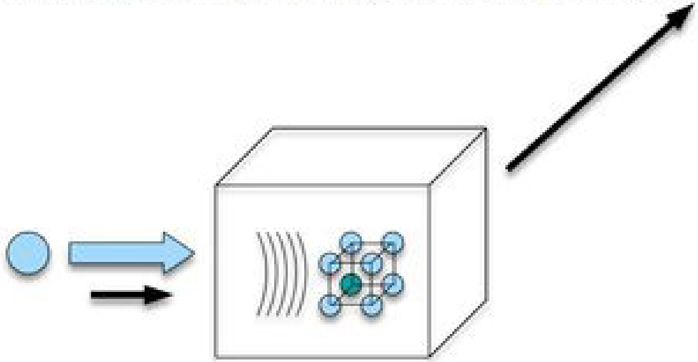
Ray-tracing methods



- When neutrons move in "free space", we use ray-tracing - but in most cases in direction source -> detector
- Of course parabolas rather than straight lines are uses to implement gravity

Elements of Monte-Carlo raytracing

- Instrument Monte Carlo methods implement coherent scattering effects
- Uses deterministic propagation where this can be done
- Uses Monte Carlo sampling of "complicated" distributions and stochastic processes and multiple outcomes with known probabilities are involved
- \- I.e. inside scattering matter
- Uses the particle-wave duality of the neutron to switch back and forward between deterministic ray tracing and Monte Carlo approach



- Result: A realistic and efficient transport of neutrons in the thermal and cold range

Simulation codes for Monte Carlo ray-tracing of neutron instruments (incomplete?)

- US codes:
  - NISP (P Seeger & L Daemen, LANL)
  - IDEAS (X-L Wang ORNL, originally H Lee ORNL)
  - Instrument Builder (JK Zhao ORNL)
  - McVine (engine of the DANSE vnf project) (J Lin Caltech)

- European codes:
  - RESTRAX/SIMRES (J Šaroun, NPI & J Kulda, ILL)
  - VITESS (K Lieutenant et al. HZB plus S Manoshin JINR)
  - McStas (P Willendrup & E Knudsen DTU, K Lefmann KU, E Farhi ILL, U Filges PSI)
  - [NADS (P Bentley ESS)]

- Japanese code:
  - PHITS (MCNP-like) has a few features for instrument beam transport

- There may be others I do not know of...

NISP

- "The mother of all neutron MC-raytracing"
- Builds on MCLIB which was derived from MCNP
- Does almost "anything", i.e. quadratic form geometries of material, has ability to scatter from any region to any region of the instrument
- Defined the de-facto implementation standard for polarized neutron scattering
- Website at http://www.paseeger.com
- Developed by P Seeger (consultant) & L Daemen, LANL
- Windows only (32 bit?)
- License conditions undefined, but a kind of public domain code
- Underlying legacy fortran code run via UI

- My 0.02$: Good stuff, but a limited audience expert code

IDEAS

- Developed for instrumentation work at SNS
- Was used for roughly half the instruments, the other half was mostly McStas and a little VITESS
- Was always a side-activity of the authors
- Website currently gone?
- Developed by H Lee (was ORNL now ANSTO) & X-L Wang, ORNL
- Windows only (32 bit?)
- Unkown license

- My 0.02$: Dead / sleeping project?

Instrument Builder

- Developed by single SNS employee for own project?
- Partial implementation but looks OK
- C++ and PVM/MPI for parallelization
- Website http://neutrons.ornl.gov/instruments/SNS/EQ-SANS/software/ib/doc/
- Developed by JK Zhao (ORNL)
- Windows & Linux
- Unkown license but source code provided

- My 0.02$: Sleeping project?

McVine (in vnf, DANSE)

- Developed for the "virtual neutron facility" which is a part of the DANSE project
- Uses Python "pyre" wraps to create Python versions of the McStas components
- Python script defines the instrument setup
- Allows to use e.g. bvk phonon simulations with instrument simulation
- Websites https://vnf.caltech.edu/vnf/1/ and http://docs.danse.us/MCViNE/1.0-beta/index.html
- Developed by J Lin and B Fultz (Caltech)
- Build to "any" platfrom from source code
- BSD license

- My 0.02$: vnf is really nice but so far implement only a few instruments

RESTRAX & SIMRES

- Builds on RESCAL and TRAX codes for triple axis resolution estimates (Popovici)
- RESTRAX is TAS-only but SIMRES has certain TOF components also
- Has very efficient "detector to source" propagation method
- Exellent monochromator descriptions, including bent Si
- Legacy fortran code with Java UI that lowers the learning curve
- Website http://neutron.ujf.cas.cz/restrax/
- Developed by J Šaroun (NPI) and J Kulda (ILL)
- Windows & Linux
- GPL License

- My 0.02$: RESTRAX is probably the best bet for a TAS code. Milage may vary in other areas.

VITESS

- Virtual Instrumentation Tool for the ESS
- Originally only TOF type instruments but now cover all types
- Had early implementation of polarized neutron techniques
- Is intrisically parallelized by running a set of single executables that talks via pipes, but efficiency can drop somewhat along beamline (now includes options for system-level parallelization via "helper threads"
- Easy to use GUI and no compilation
- Hard(er) to include your own developments
- C-code with little overlap between individually developed modules
- Website http://www.helmholtz-berlin.de/vitess
- Developed by K Lieutenant and his team (HZB) and Sergei Manoshin (JINR)
- Windows, Linux, Mac OS X
- GPL License

- My 0.02$: VITESS has a low learning curve and works very well. Can be hard to do more complex things. Full visualization of the instrument is being developed.

NADS

- Not a Monte Carlo raytracer - uses very quick acceptance diagram techniques instead
- So far decoupled simulation of vertical vs. horizontal beam
- Optics only
- Website http://code.google.com/p/jnads/
- Developed by P Benley (ESS, formerly ANSTO, ILL, HMI)
- "Any" system (java implementation)
- GPL License

- My 0.02$: Has clear potential but is currently not an all-round package.

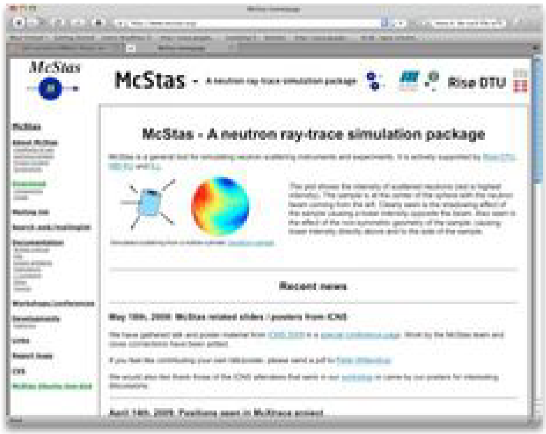
McStas

- The code that I develop.... :)
- Will now progress to discuss that and its features

McStas Introduction

- Flexible, general simulation utility for neutron scattering experiments.
- Original design for Monte carlo Simulation of triple axis spectrometers
- Developed at DTU Physics, ILL, PSI, Uni CPH
- V. 1.0 by K Nielsen & K Lefmann (1998) RISØ
- Currently 2.5+1 people full time plus students

GNU GPL license
Open Source



Project website at
http://www.mcstas.org

mcstas-users@mcstas.org mailinglist

McStas Introduction

McXtrace - since jan 2009 similar in X-rays

•Flexible, general simulation utility for neutron scattering experiments.

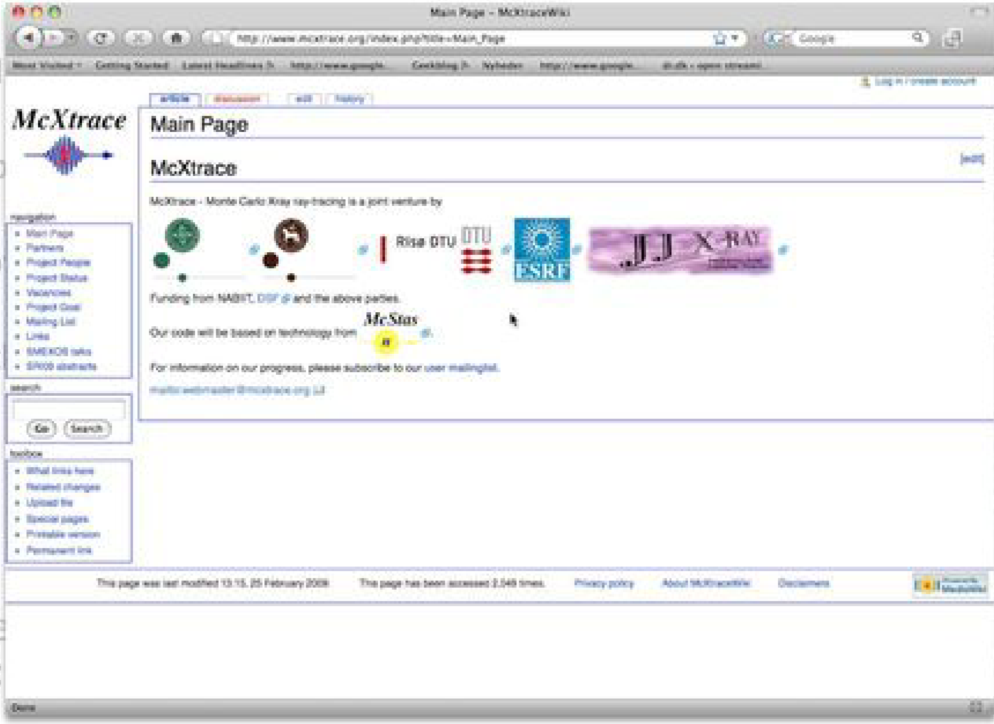
•Original

•Develop

•V. 1.0 b

•Current

Proje
http:
list

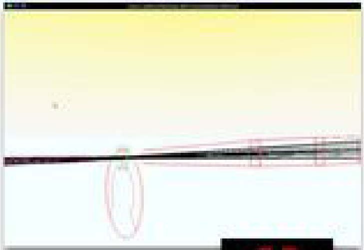


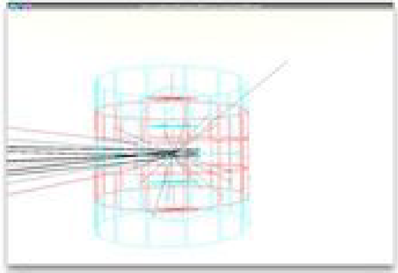
- Synergy, knowledge transfer, shared infrastructure

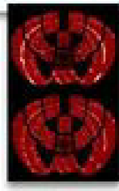
McStas Introduction

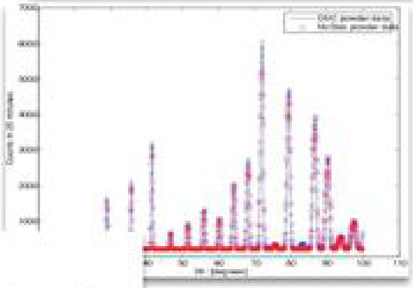
World leading in neutron Monte Carlo

What is McStas used for?



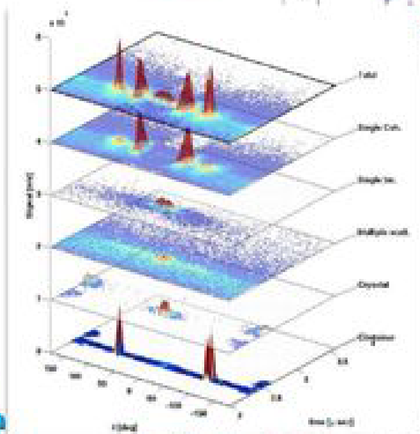


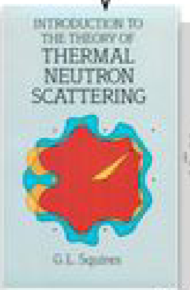


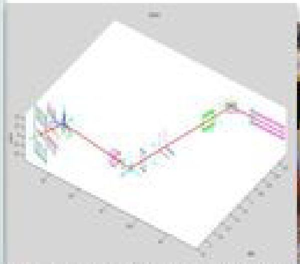


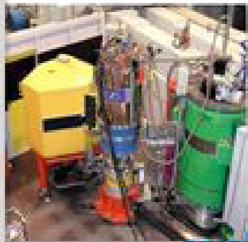
- Instrumentation
- Virtual experiments
- Data analysis
- Teaching

(KU, DTU)



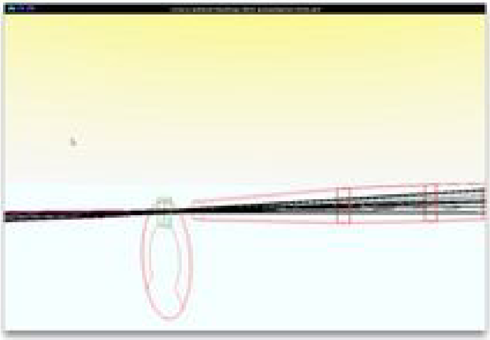


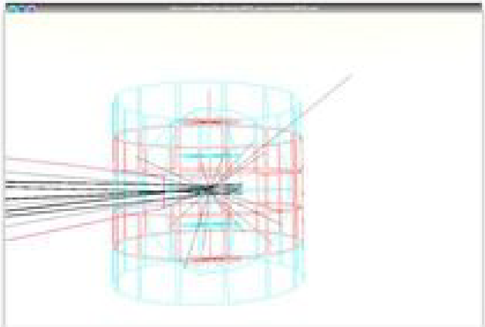


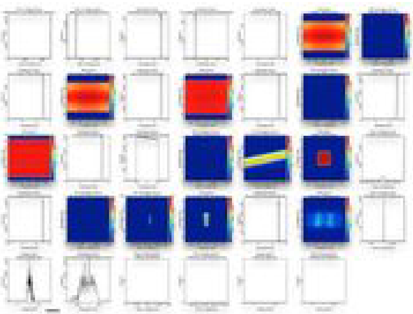


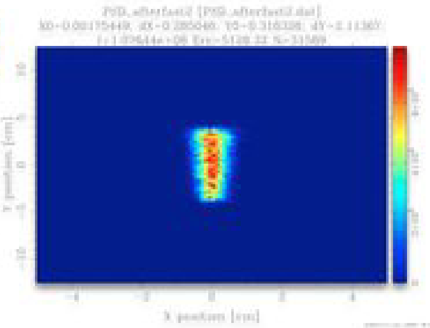
Instrumentation

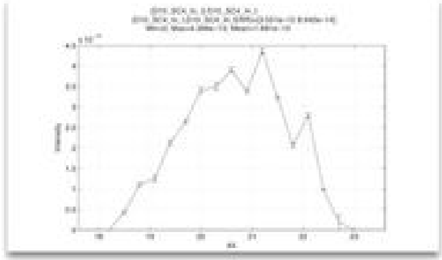
- Design and optimization of instruments







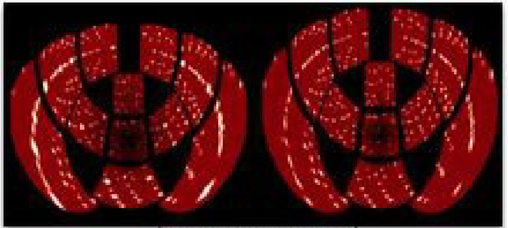




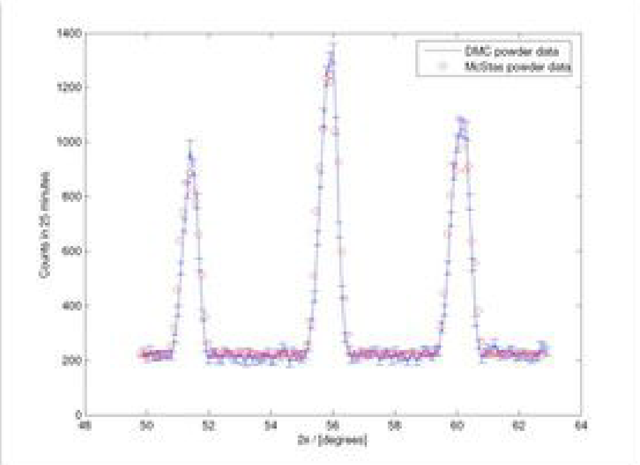
Virtual experiments (VE)

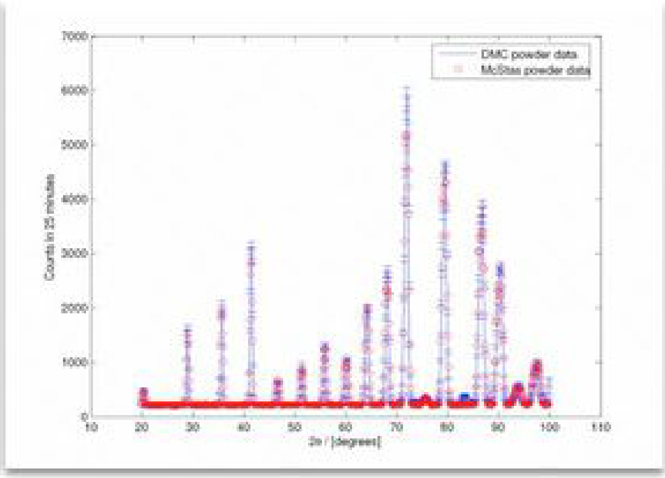
(definition:)

- Simulation of a complete experiment
- … from source to detector
- Ideally controlled like real experiment.
- Data analysed by "real" analysis programs



A. Daud-Aladine, ISIS

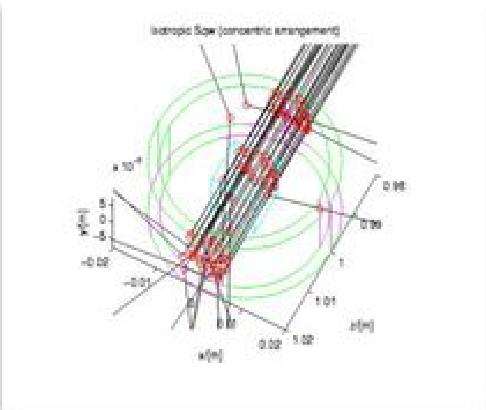


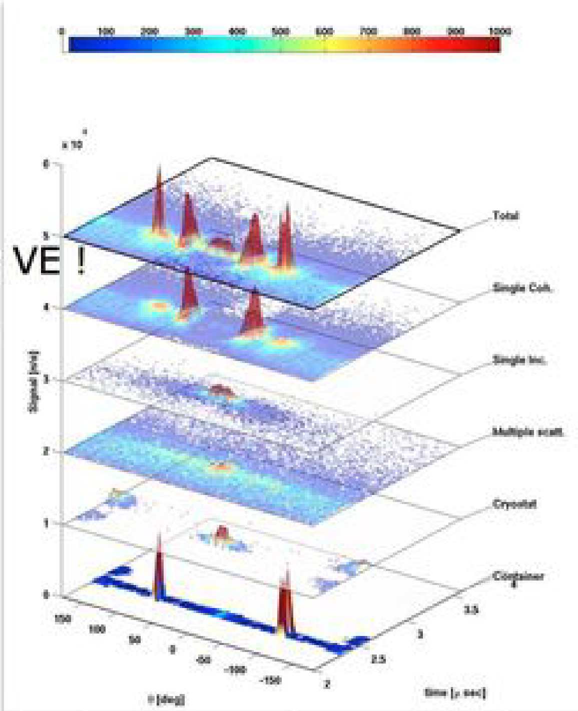


P. Willendrup, Risø DTU; Uwe Filges, L. Keller, PSI

Data analysis (1)
(using VE techiques)

- Virtual TOF exp. at IN6, ILL
- Liquid Ge sample
- Coherent / incoherent
- Multiple scattering
- And sample environment
- All contributions can be separated by VE !

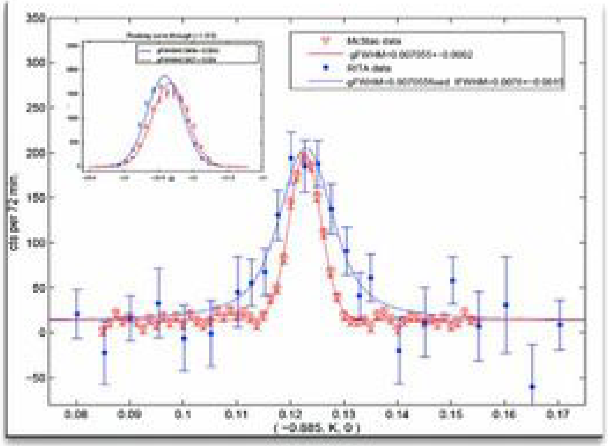




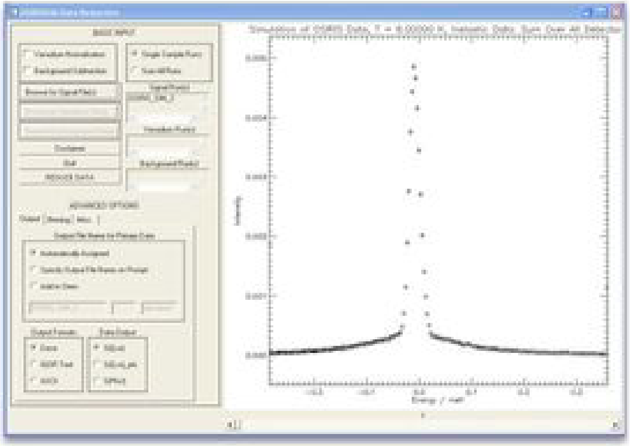
E. Farhi, ILL

Data Analysis (2)
(using VE techniques)

- VE data has been used to test data analysis programs
- … and to check resolution effects



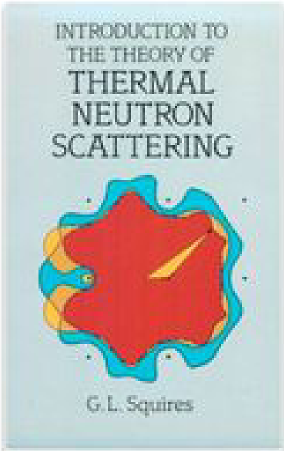
L. Udby, Risø-DTU

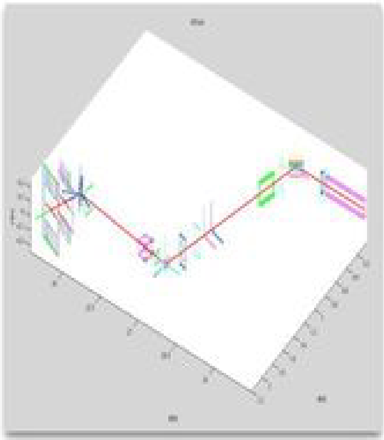


P. Tregenna-Piggott, PSI

Teaching / training purposes

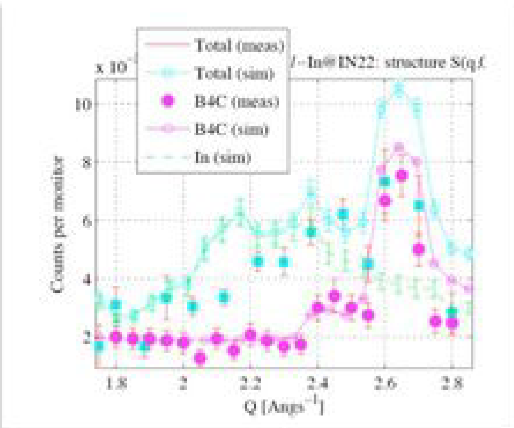
- Workshops
- Teaching
  - University of Copenhagen course on Neutron Scattering



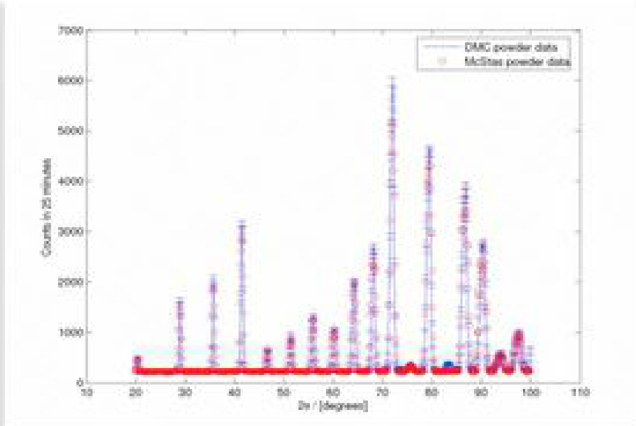


Reliability - cross comparisons

- Much effort has gone into this
- Here: simulations vs. exp. at powder diffract. DMC, PSI
- The bottom line is
- McStas agree very well with other packages (NISP, Vitess, IDEAS, RESTRAX, ...)
- Experimental line shapes are within 5%
- Absolute intensities are within 10%
- Common understanding: McStas and similar codes are reliable



E. Farhi, P. Willendrup et al., in preparation

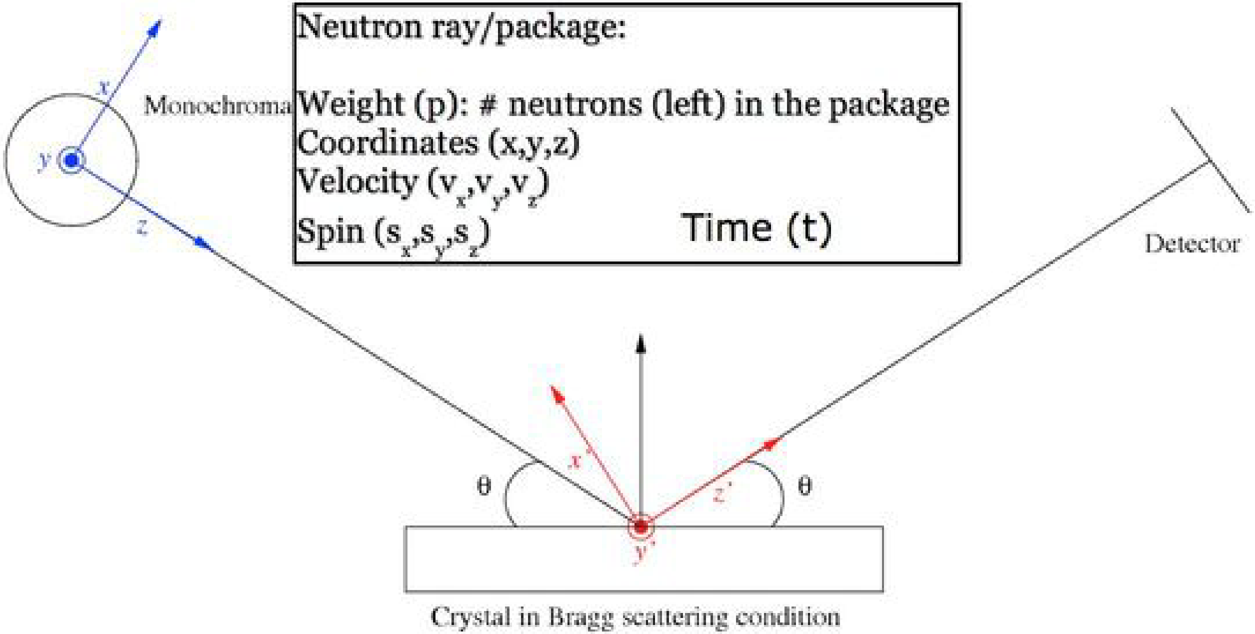


P. Willendrup et al., Physica B, 386, (2006), 1032.

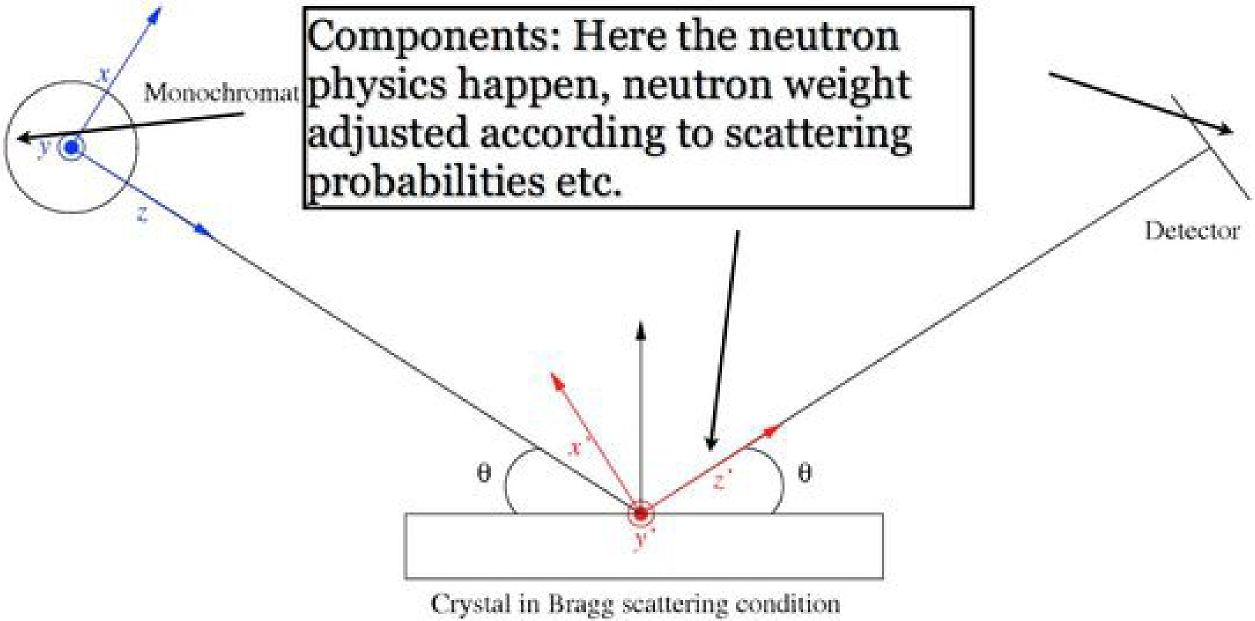
McStas: key concepts

Neutron ray/package:

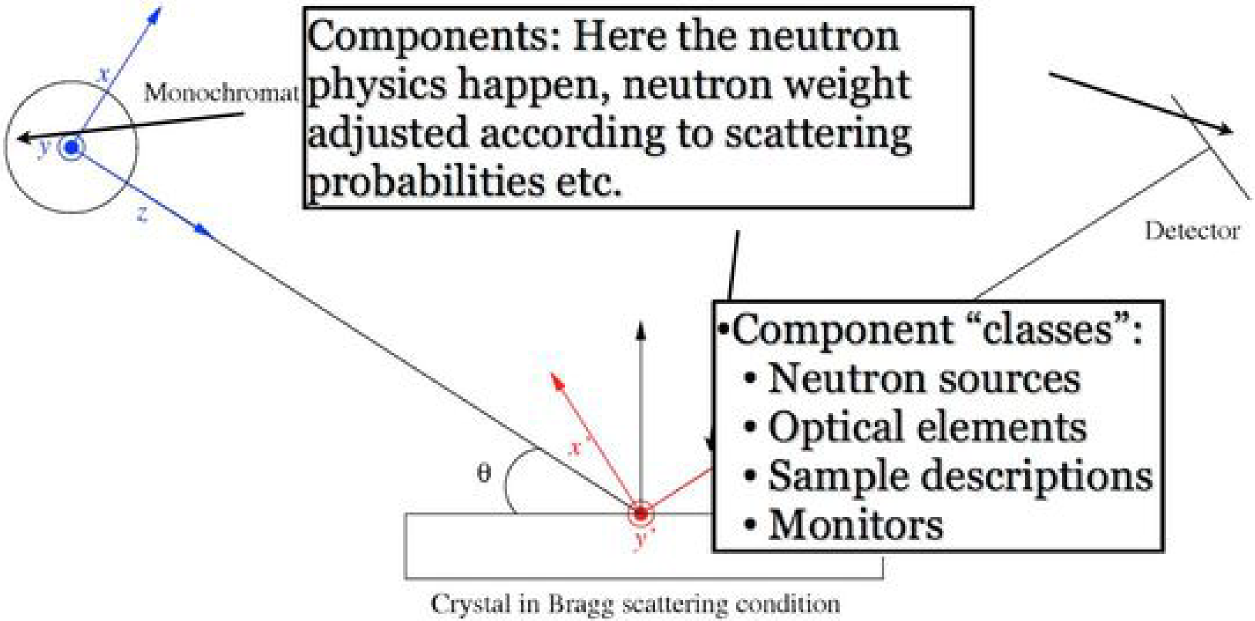
Weight (p): # neutrons (left) in the package
Coordinates (x,y,z)
Velocity (v_x,v_y,v_z)
Spin (s_x,s_y,s_z)
Time (t)



McStas: key concepts



McStas: key concepts

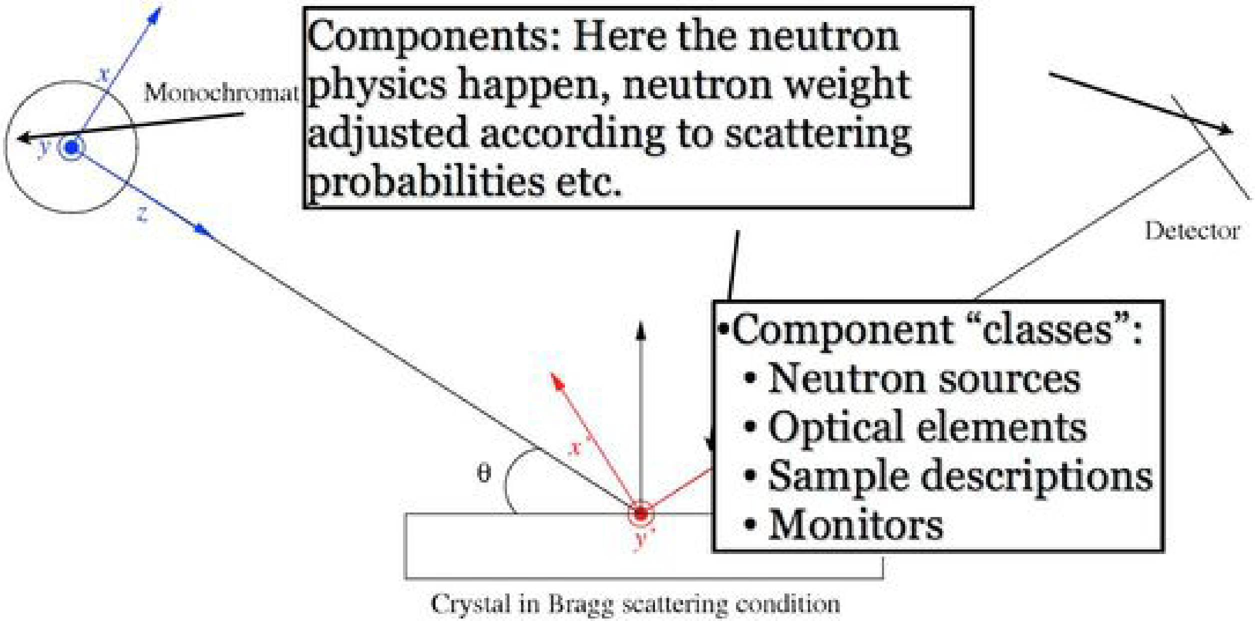


Components: Here the neutron physics happen, neutron weight adjusted according to scattering probabilities etc.

- Component "classes":
  - Neutron sources
  - Optical elements
  - Sample descriptions
  - Monitors

McStas: key concepts

Local, internal coordinate system!



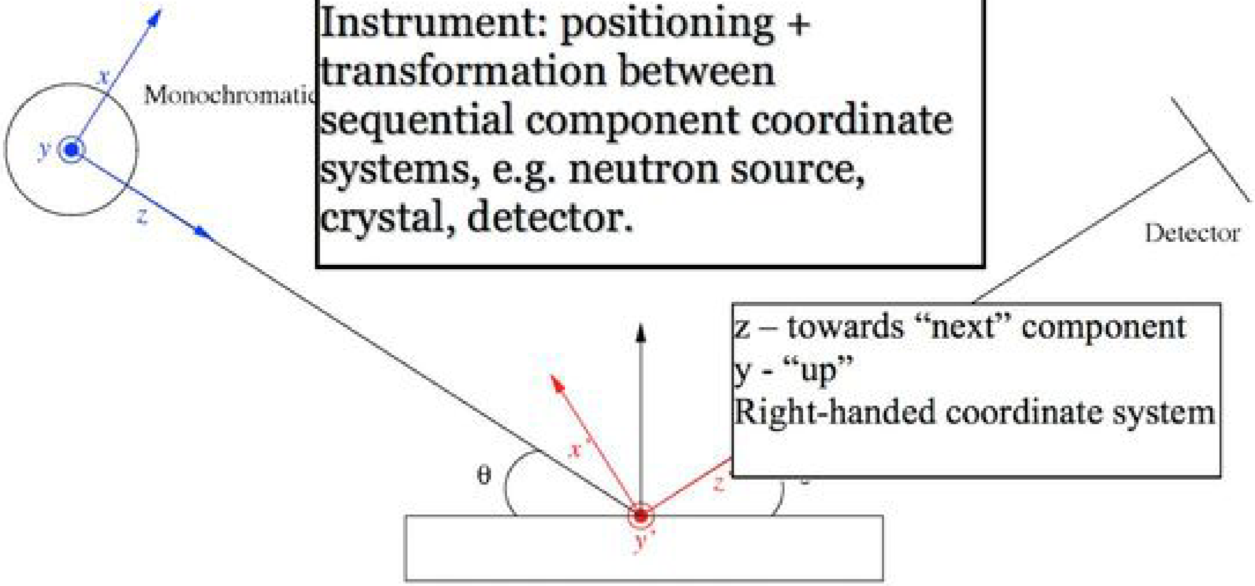
Components: Here the neutron physics happen, neutron weight adjusted according to scattering probabilities etc.

- Component "classes":
  - Neutron sources
  - Optical elements
  - Sample descriptions
  - Monitors

McStas: key concepts

Instrument: positioning + transformation between sequential component coordinate systems, e.g. neutron source, crystal, detector.

z – towards “next” component
y - “up”
Right-handed coordinate system



Crystal in Bragg scattering condition

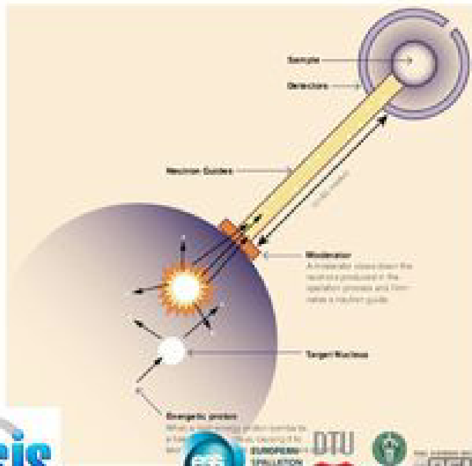
McStas overview

- Portable code (Unix/Linux/Mac/Windoze)

- Ran on everything from iPhone to 1000+ node cluster!

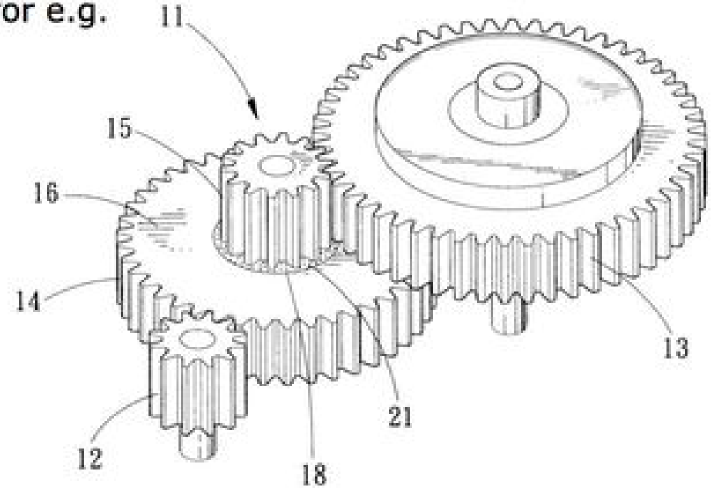
- 'Component' files (~100) inserted from library
  - Sources
  - Optics
  - Samples
  - Monitors
  - If needed, write your own comps

- DSL + ISO-C code gen.



Under-the-hood / inner workings

- Domain-specific-language (DSL) based on compiler technology (LeX+Yacc)
  - Simple Instrument language $\xrightarrow{\text{Code generation}}$ ISO C
- Component codes realizing beamline parts (including user contribs)
- Library of common functions for e.g.
  - I/O
  - Random numbers
  - Physical constants
  - Propagation
  - Precession in fields
  - ...



Implementation

- Three levels of source code:
  - Instrument file (All users)
  - Component files (Some users)
  - ANSI c code (no users)

Instrument file

```
DEFINE INSTRUMENT My_Instrument(DIST=10)

/* Here comes the TRACE section, where the actual      */
/* instrument is defined as a sequence of components.  */
TRACE

/* The Arm() class component defines reference points and orientations  */
/* in 3D space.                                                         */
COMPONENT Origin = Arm()
  AT (0,0,0) ABSOLUTE


COMPONENT Source = Source_simple(
    radius = 0.1, dist = 10, xw = 0.1, yh = 0.1, E0 = 5, dE = 1)
  AT (0, 0, 0) RELATIVE Origin


COMPONENT Emon = E_monitor(
    filename = "Emon.dat", xmin = -0.1, xmax = 0.1, ymin = -0.1,
    ymax = 0.1, Emin = 0, Emax = 10)
  AT (0, 0, DIST) RELATIVE Origin


COMPONENT PSD = PSD_monitor(
    nx = 128, ny = 128, filename = "PSD.dat", xmin = -0.1,
    xmax = 0.1, ymin = -0.1, ymax = 0.1)
  AT (0, 0, 1e-10) RELATIVE Emon

/* The END token marks the instrument definition end */
END
```

Written by you!

Component file

```
/*******************************************************************************
*
* McStas, neutron ray-tracing package
*         Copyright 1997-2002, All rights reserved
*         Risoe National Laboratory, Roskilde, Denmark
*         Institut Laue Langevin, Grenoble, France
*
* Component: Source_flat
*
* %I
* Written by: Kim Lefmann
* Date: October 30, 1997
* Modified by: KL, October 4, 2001
* Modified by: Emmanuel Farhi, October 30, 2001. Serious bug corrected.
* Version: $Revision: 1.22 $
* Origin: Risoe
* Release: McStas 1.6
*
* A circular neutron source with flat energy spectrum and arbitrary flux
*
* %D
* The routine is a circular neutron source, which aims at a square target
* centered at the beam (in order to improve MC-acceptance rate).  The angular
* divergence is then given by the dimensions of the target.
* The neutron energy is uniformly distributed between E0-dE and E0+dE.
*
* Example: Source_flat(radius=0.1, dist=2, xw=.1, yh=.1, E0=14, dE=2)
*
* %P
* radius: (m)   Radius of circle in (x,y,0) plane where neutrons
*               are generated.
* dist:   (m)   Distance to target along z axis
* xw:     (m)   Width(x) of target
* yh:     (m)   Height(y) of target
* E0:     (meV) Mean energy of neutrons.
* dE:     (meV) Energy spread of neutrons.
* Lambda0 (AA)  Mean wavelength of neutrons.
* dLambda (AA)  Wavelength spread of neutrons.
* flux    (1/(s*cm**2*st)) Energy integrated flux
*
* %E
*******************************************************************************/

DEFINE COMPONENT Source_simple
DEFINITION PARAMETERS ()
SETTING PARAMETERS (radius, dist, xw, yh, E0=0, dE=0, Lambda0=0, dLambda=0, flux=1)
OUTPUT PARAMETERS ()
STATE PARAMETERS (x,y,z,vx,vy,vz,t,s1,s2,p)
DECLARE
%{
  double pmul, pdir;
%}
INITIALIZE
%{
  pmul=flux*PI*1e4*radius*radius/mcget_ncount();
%}
```

```
TRACE
%{
 double chi,E,Lambda,v,r, xf, yf, rf, dx, dy;

 t=0;
 z=0;

 chi=2*PI*rand01();                          /* Choose point on source */
 r=sqrt(rand01())*radius;                    /* with uniform distribution. */
 x=r*cos(chi);
 y=r*sin(chi);

 randvec_target_rect(&xf, &yf, &rf, &pdir,
        0, 0, dist, xw, yh, ROT_A_CURRENT_COMP);

 dx = xf-x;
 dy = yf-y;
 rf = sqrt(dx*dx+dy*dy+dist*dist);

 p = pdir*pmul;

 if(Lambda0==0) {
   E=E0+dE*randpm1();                 /*  Choose from uniform distribution */
   v=sqrt(E)*SE2V;
 } else {
   Lambda=Lambda0+dLambda*randpm1();
   v = K2V*(2*PI/Lambda);
 }

 vz=v*dist/rf;
 vy=v*dy/rf;
 vx=v*dx/rf;
%}

MCDISPLAY
%{
  magnify("xy");
  circle("xy",0,0,0,radius);
%}

END
```

Written by developers and possibly you!

Generated c-code

```
/* Automatically generated file. Do not edit.
 * Format:     ANSI C source code
 * Creator:    McStas (http://neutron.risoe.dk)
 * Instrument: My_Instrument.instr (My_Instrument)
 * Date:       Sat Apr  9 15:27:56 2005
 */


/* THOUSANDS of lines removed here.... */

  /* TRACE Component Source. */
  SIG_MESSAGE("Source (Trace)");
  mcDEBUG_COMP("Source")
  mccoordschange(mcposrSource, mcrotrSource,
    &mcnlx, &mcnly, &mcnlz,
    &mcnlvx, &mcnlvy, &mcnlvz,
    &mcnlt, &mcnlsx, &mcnlsy);
  mcDEBUG_STATE(mcnlx, mcnly, mcnlz, mcnlvx, mcnlvy, mcnlvz,mcnlt,mcnlsx,mcnlsy, mcnlp)
#define x mcnlx
#define y mcnly
#define z mcnlz
#define vx mcnlvx
#define vy mcnlvy
#define vz mcnlvz
#define t mcnlt
#define s1 mcnlsx
#define s2 mcnlsy
#define p mcnlp
  STORE_NEUTRON(2,mcnlx, mcnly, mcnlz, mcnlvx,mcnlvy,mcnlvz,mcnlt,mcnlsx,mcnlsy, mcnlsz, mcnlp);
  mcScattered=0;
  mcNCounter[2]++;
#define mccompcurname Source
#define mccompcurindex 2
{   /* Declarations of SETTING parameters. */
MCNUM radius = mccSource_radius;
MCNUM dist = mccSource_dist;
MCNUM xw = mccSource_xw;
MCNUM yh = mccSource_yh;
MCNUM E0 = mccSource_E0;
MCNUM dE = mccSource_dE;
MCNUM Lambda0 = mccSource_Lambda0;
MCNUM dLambda = mccSource_dLambda;
MCNUM flux = mccSource_flux;
#line 58 "Source_simple.comp"
{
 double chi,E,Lambda,v,r, xf, yf, rf, dx, dy;

 t=0;
 z=0;

 chi=2*PI*rand01();                          /* Choose point on source */
 r=sqrt(rand01())*radius;                    /* with uniform distribution. */
 x=r*cos(chi);
 y=r*sin(chi);

 randvec_target_rect(&xf, &yf, &rf, &pdir,
        0, 0, dist, xw, yh, ROT_A_CURRENT_COMP);
```

Written by mcstas!

McStas is a (pre)compiler!

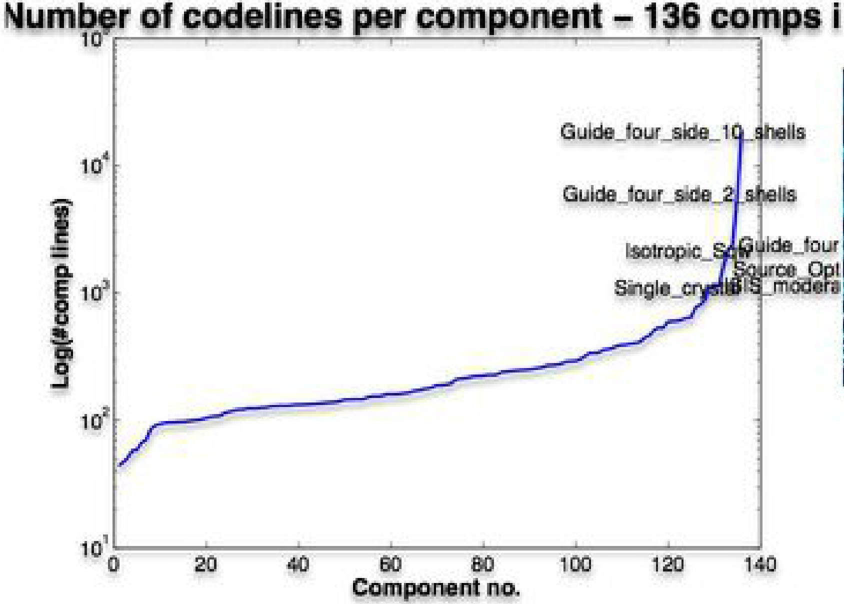
Input is .comp and .instr files + runtime functions for e.g. random numbers

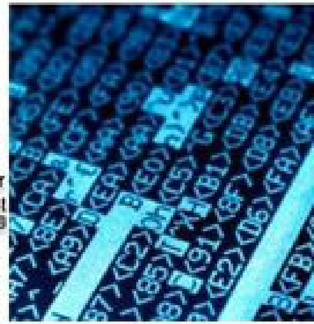
Output is a single c-file, which can be compiled using e.g. gcc.

Can take input arguments if needed.

Writing new comps or understanding existing is not that complex...

- Check our long list of components and look inside... Most of them are quite simple and short... Statistics:





Including user contribs

- Well-developed community support
  - 30-40% of existing and new additions are from users
  - No direct refereeing of the code, but these requirements:
    - At least one test-instrument
    - Meaningful documentation headers (in-code docs)
  - Contributions go in dedicated contrib/ section of library

- Natural life-cycle of contrib's
  - Bug-fixes are applied both by contributor and developers
  - If contributor becomes unavailable either:
    - Many users of comp: Promote to official components, e.g. in optics/
    - Few/no users of comp: Move to obsolete/ until next major release

Advanced language features

- Macros and tricks for your instrument...

DECLARE / INITIALIZE

- Use the DECLARE section define user variables and functions.
  - DECLARE %{
  - double myvar;
  - %}

- Use INITIALIZE for initialization of user variables and calculations.
  - INITIALIZE %{
  - myvar = sqrt(PI*input_var)*rand01();
  - %}

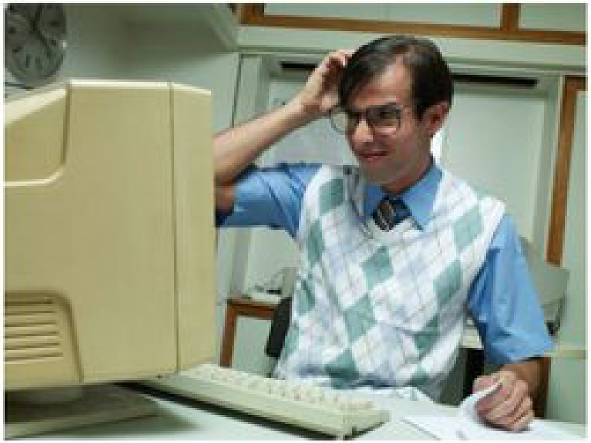
- - Both use normal c-syntax.

- BEWARE: (example) What you do in the c-style areas is c-standard, e.g. trigonometric functions from math.h use radians! - McStas placement specifiers work in degrees, etc...

K & R

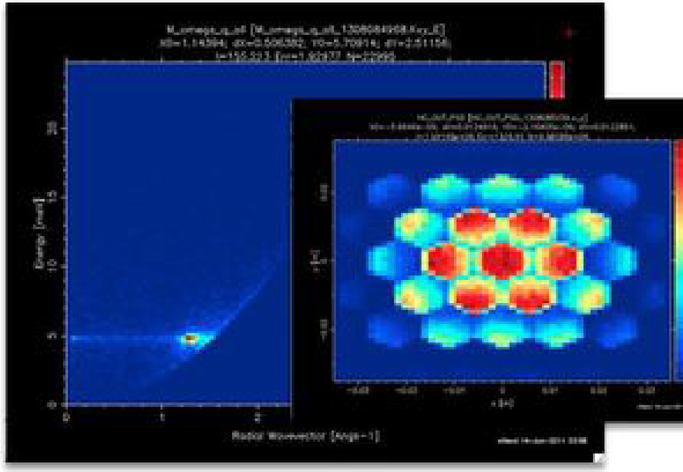
Syntax in one, complex view...

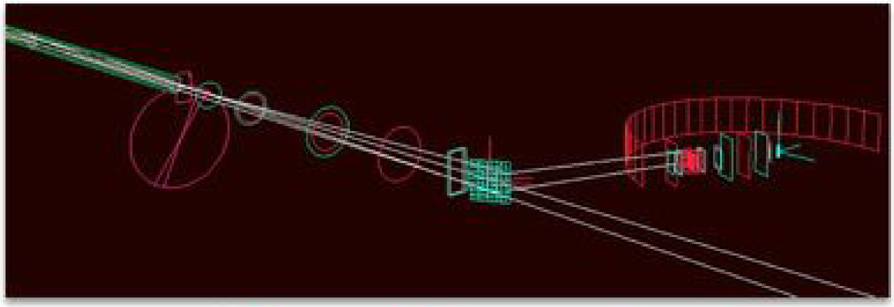
{SPLIT} COMPONENT name = comp(parameters) {WHEN condition}
AT (...) [RELATIVE [reference|PREVIOUS] | ABSOLUTE]
{ROTATED {RELATIVE [reference|PREVIOUS] | ABSOLUTE} }
{GROUP group_name}
{EXTEND C_code}
{JUMP [reference|PREVIOUS|MYSELF|NEXT] [ITERATE number_of_times | WHEN condition] }



Example suite: 7 TOF spectrometers:

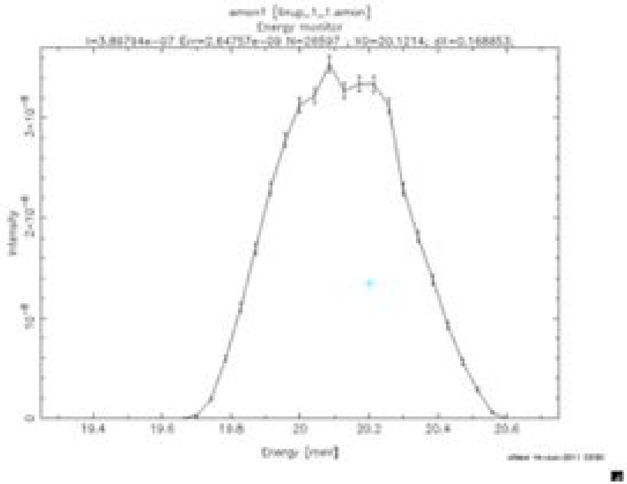
- ESS_IN5_reprate.instr
- ILL_BRISP.instr (Small-angle)
- ILL_H15_IN6.instr
- ILL_H16_IN5.instr
- ISIS_Hetfull.instr
- PSI_Focus.instr
- templateTOF.instr

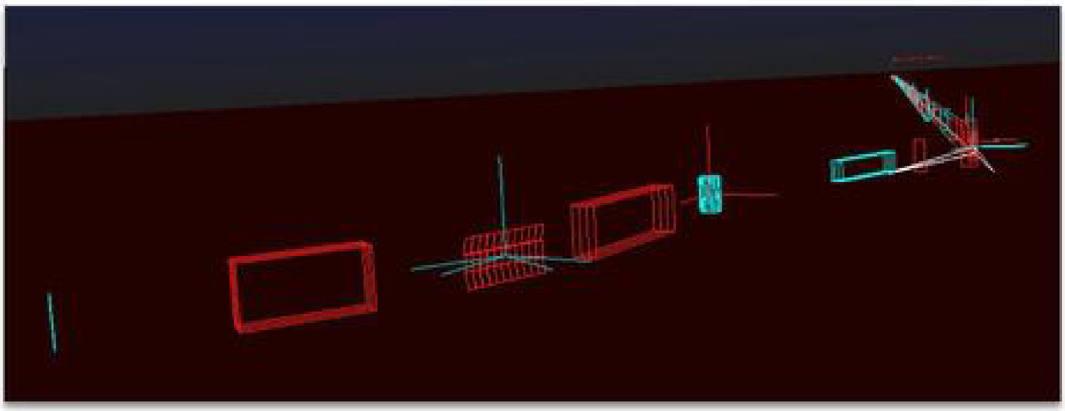




Example suite: 5 TAS

- ILL_H142_IN12.instr
- ILL_H25_IN22.instr
- h8_test.instr
- templateTAS.instr
- linup-1.instr (Risø TAS 1)
- linup-2.instr
- linup-3.instr
- linup-4.instr
- linup-5.instr
- linup-6.instr
- linup-7.instr

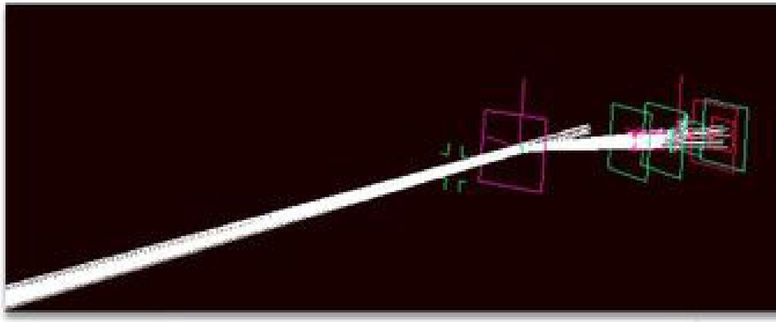


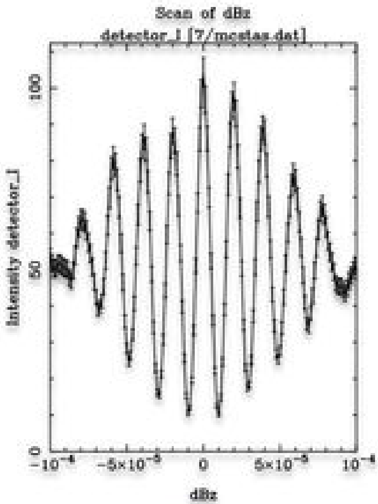


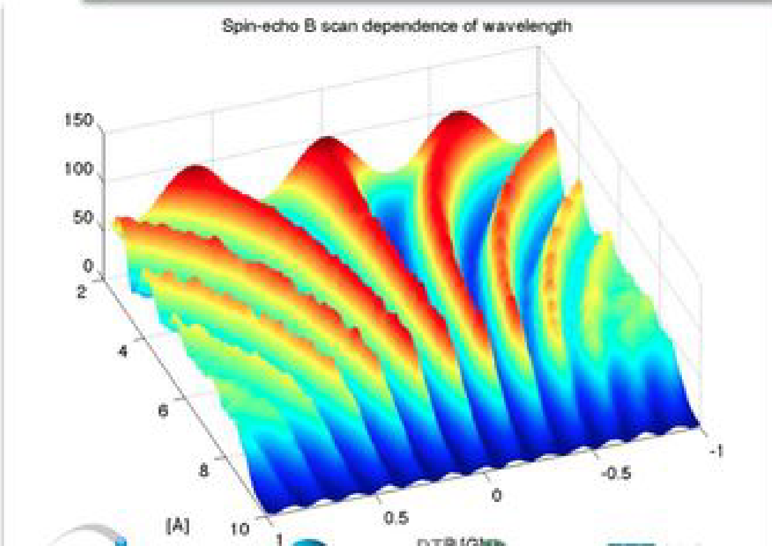
Example suite: 1 Hybrid spectrometer + 1 Spin-echo

- 1 Hybrid spectrometer:
- prisma2.instr

- 1 Spin-echo (two implementations)
- SE_example.instr
- SE_example2.instr

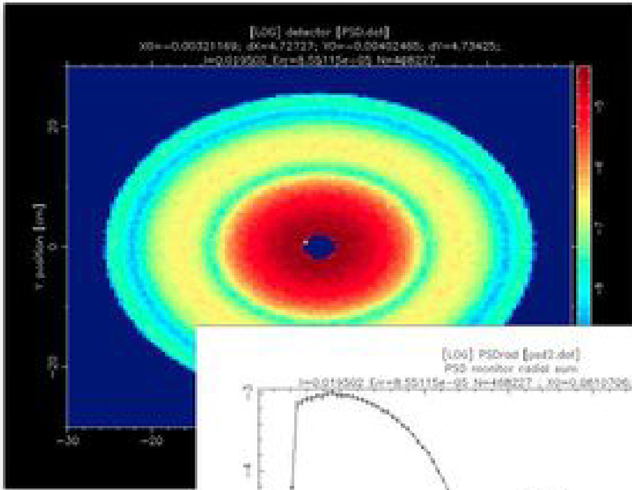


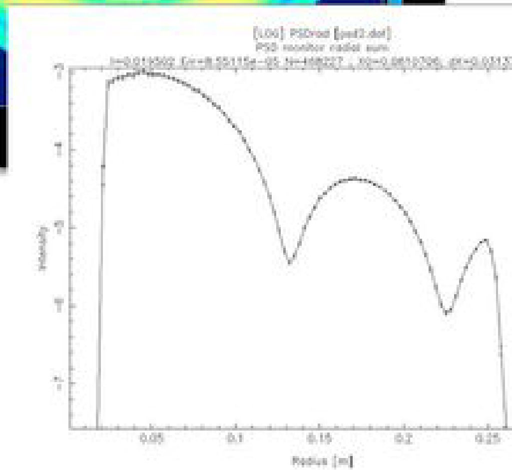


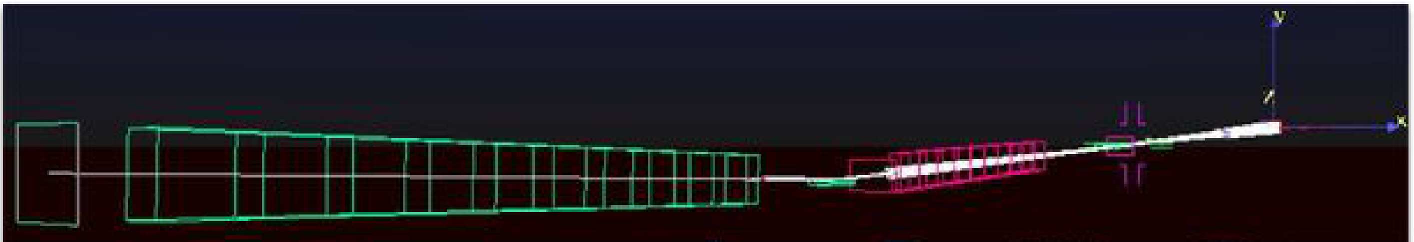


Example suite: Large scale structures

- •2 SANS:
- •FZJ_KWS2_Lens.instr
- •FZJ_SANS_KWS2_AnySample.instr
- •FZJ_SANS_KWS2_DebyeS.instr
- •FZJ_SANS_KWS2_Guinier.instr
- •FZJ_SANS_KWS2_NoSample.instr
- •SANS.instr
- •1 Reflectometer:
- •ISIS_CRISP.instr (Not an accurate model)

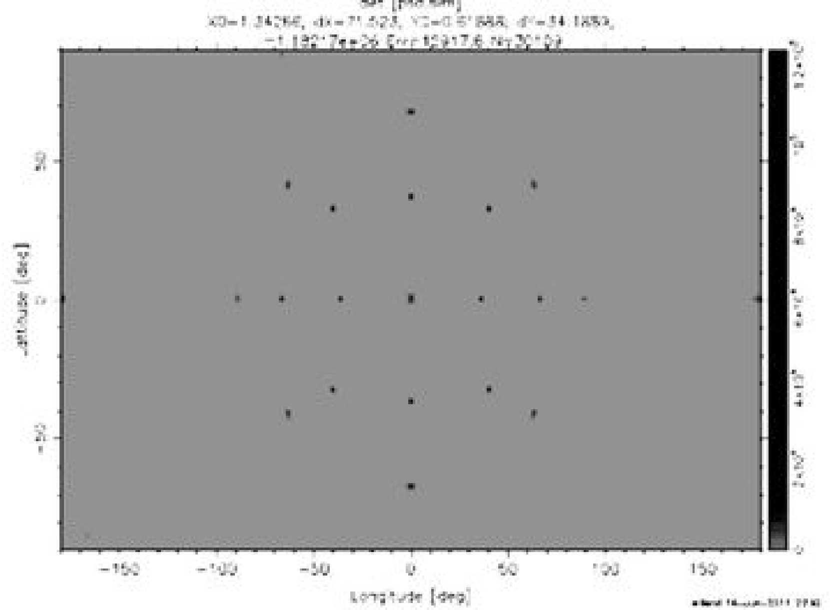


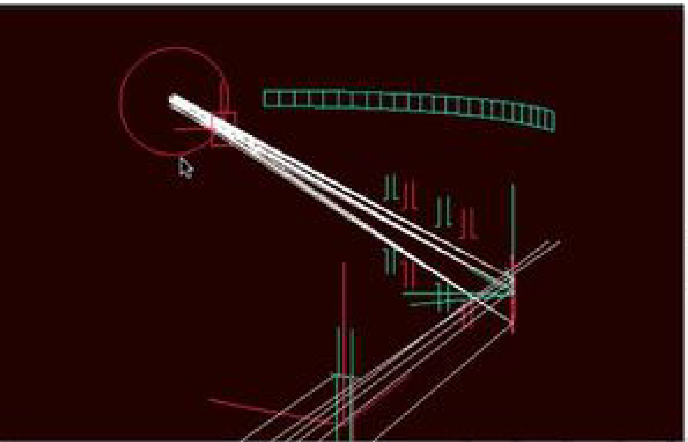


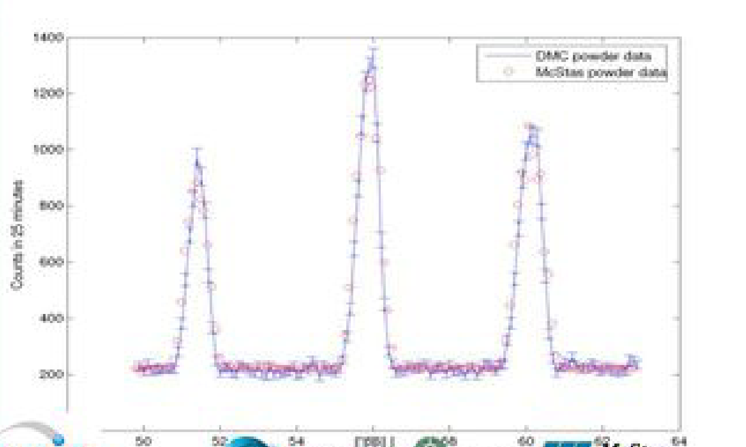


Example suite: Diffractometers

- ILL_D1A.instr
- PSI_DMC.instr
- templateDIFF.instr
- templateLaue.instr

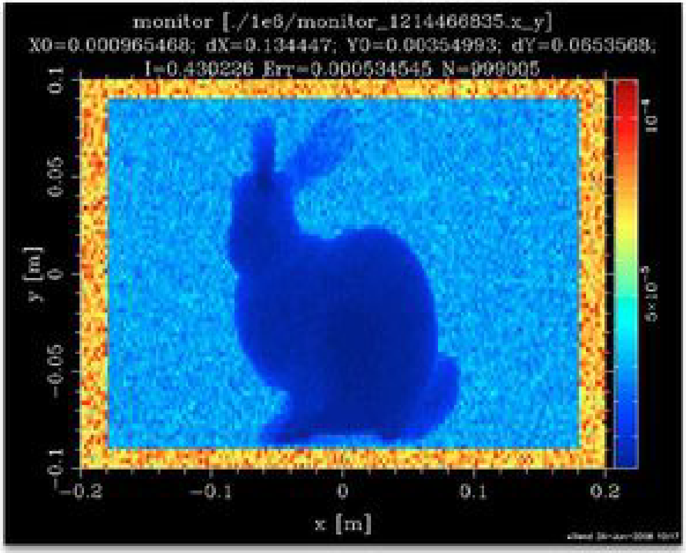


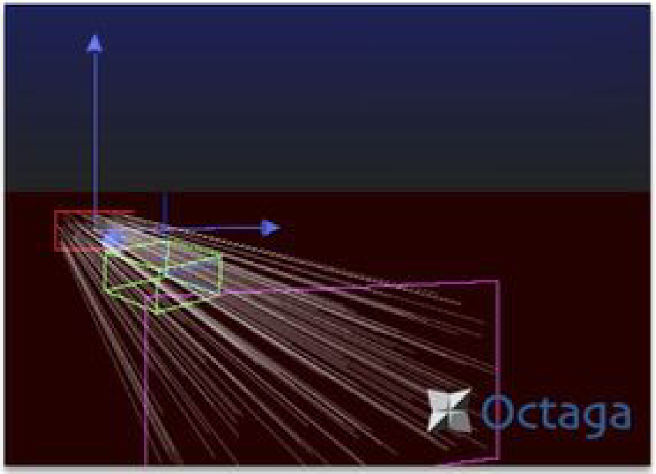


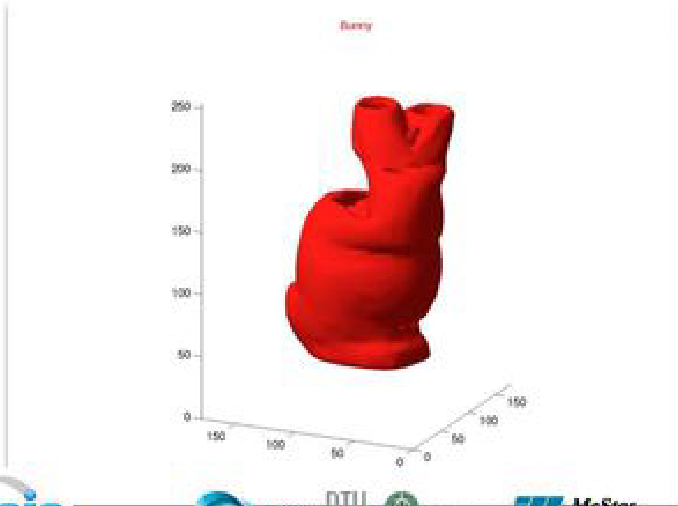


Example suite: Imaging

- Tomography.instr
- - comes with simple
- filtered backprojection
- reconstruction (Matlab)
-







Documentation

- Basic use info is available inside comp & instr codes, extracted by perl to html
- 100+ page manuals documenting
  - Metalanguage
  - What is "under the hood"
  - Examples of practical use plus advanced features
  - Assumptions and algorithms applied in the components
- More than 70 example instruments
- Various tutorial and teach yourself solutions are available

Technique limitations / words of advice

- These codes all describe the "wanted" neutrons / tailor your beam
- Every code has unique features, advantages, disadvantages. If possible, choose the one where someone you trust has experience
- Less easy to discuss instrument backgrounds and spurions etc. but effort is going in this direction on the McStas team (concentric components, interfaces to the MCNPX code etc.)
- These codes can not be considered "black box" utilities, a lot of thinking / calculus required before, during, after simulation

Knowing more / continuing at home

- All development teams are very dedicated and want to help, pass your problems on!

- McStas+VITESS teams often do schools/workshops together where you solve problems with one, the other or both codes. Exercises and code from a workshop in Ven 2010 is available here:
  - http://ven2010.essworkshop.org/storage/

- We will provide you with a take-home Linux Live DVD (ILL/NMI3) where several of the european codes are preinstalled

People

- The success of the project is also about the people:
  - Present McStas team members


K Lefmann


E Farhi


P Willendrup

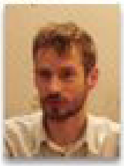

E Knudsen


U Filges

  - Past McStas team members


K Nielsen


PO Åstrand


K Lieutenant


P Christiansen

McStas

Enough Talk!
Let’s see McStas run?