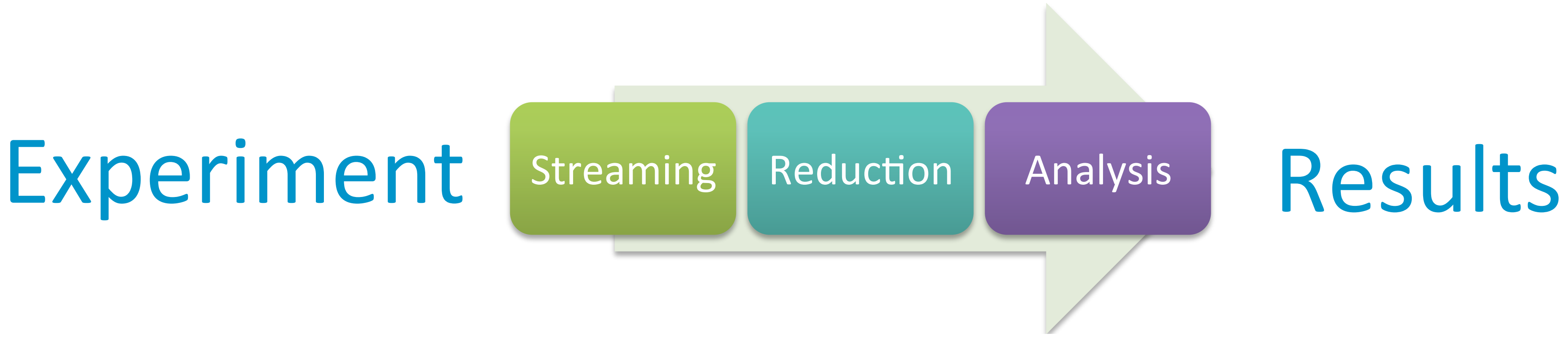
ESS view on SasView

Small Angle Scattering data analysis within the SINE2020

Wojtek Potrzebowski
Data Analysis and Modelling group
Data Management and Software Center
European Spallation Source, ERIC

www.europeanspallationsource.se
14 October 2016

Data analysis that makes user happy



Data analysis software requirements:

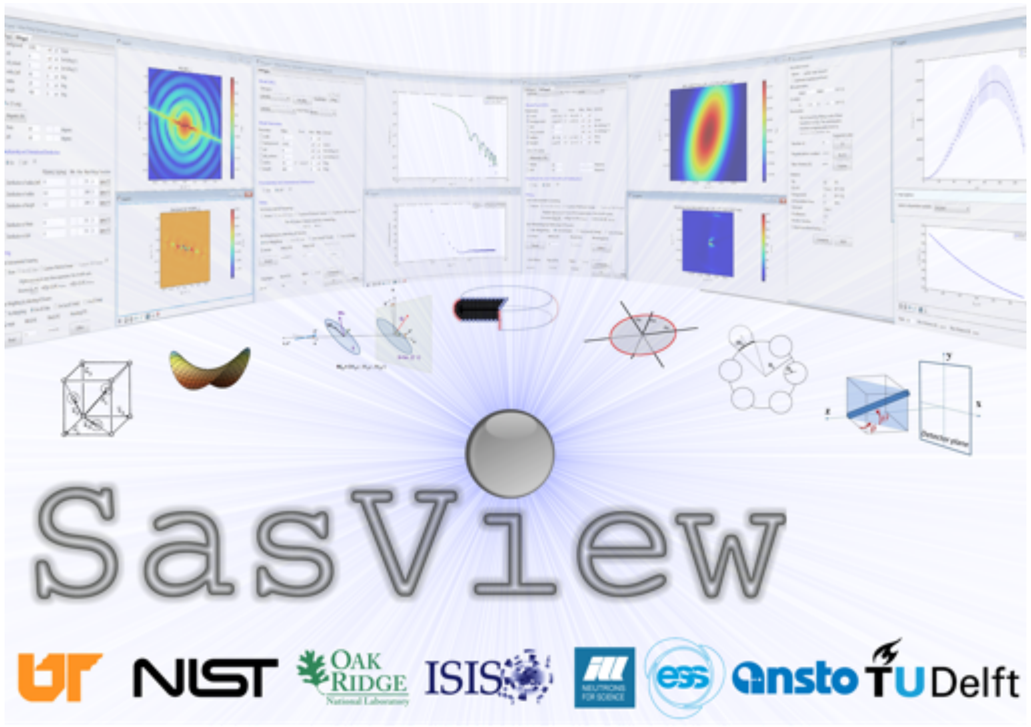
- Exploits underlying science
- Easily extendable with new ideas
- Robust and easy interface
- Up-to-date documentation
- Sufficiently fast
- Maintainable
- Sustainable

Outline

- SasView - Small Angle Scattering analysis software
- SasView development workflow
- Key features of the latest release
- SasView within SINE2020 project

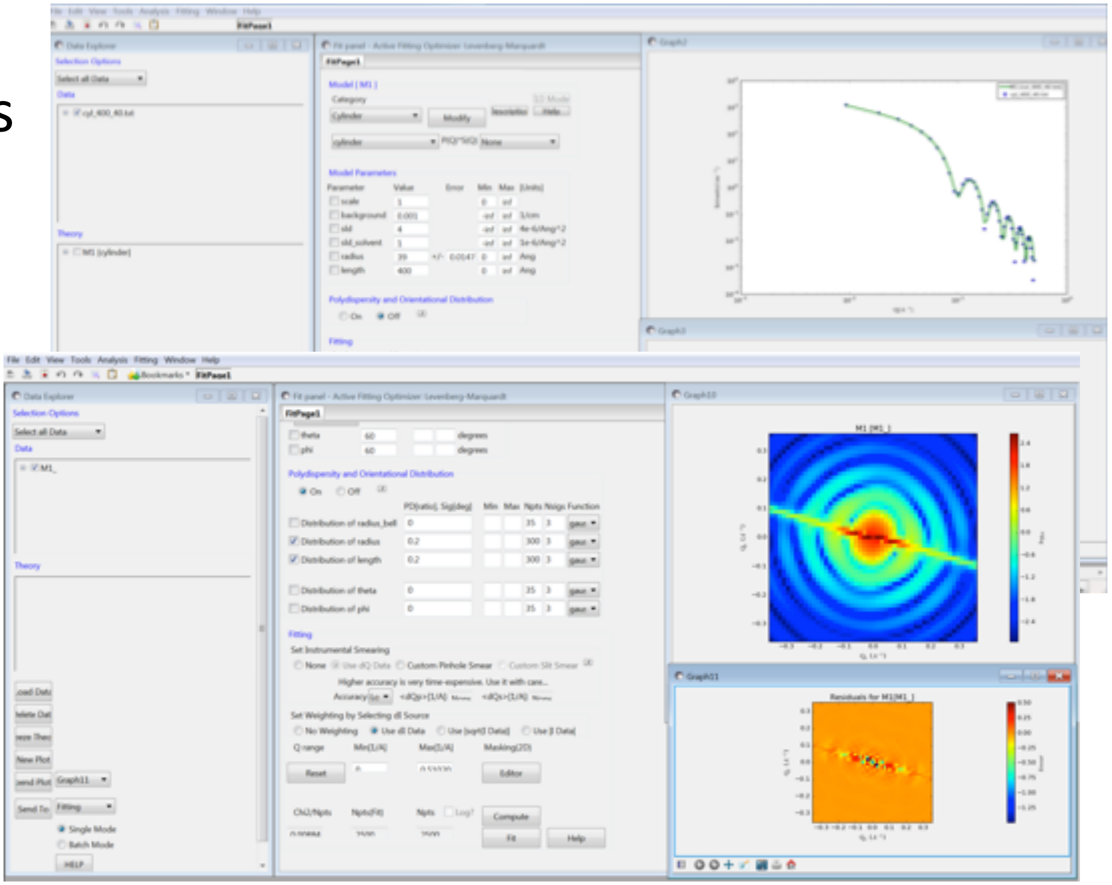
SasView - Small Angle Scattering Analysis Software Package

- Operates on reduced scattering data
- Performs modeling in inverse space
- Data analysis toolbox:
  - Fitting models to data
  - P(r) inversion
  - Model-independent analysis
- Other useful tools



SasView - Fitting

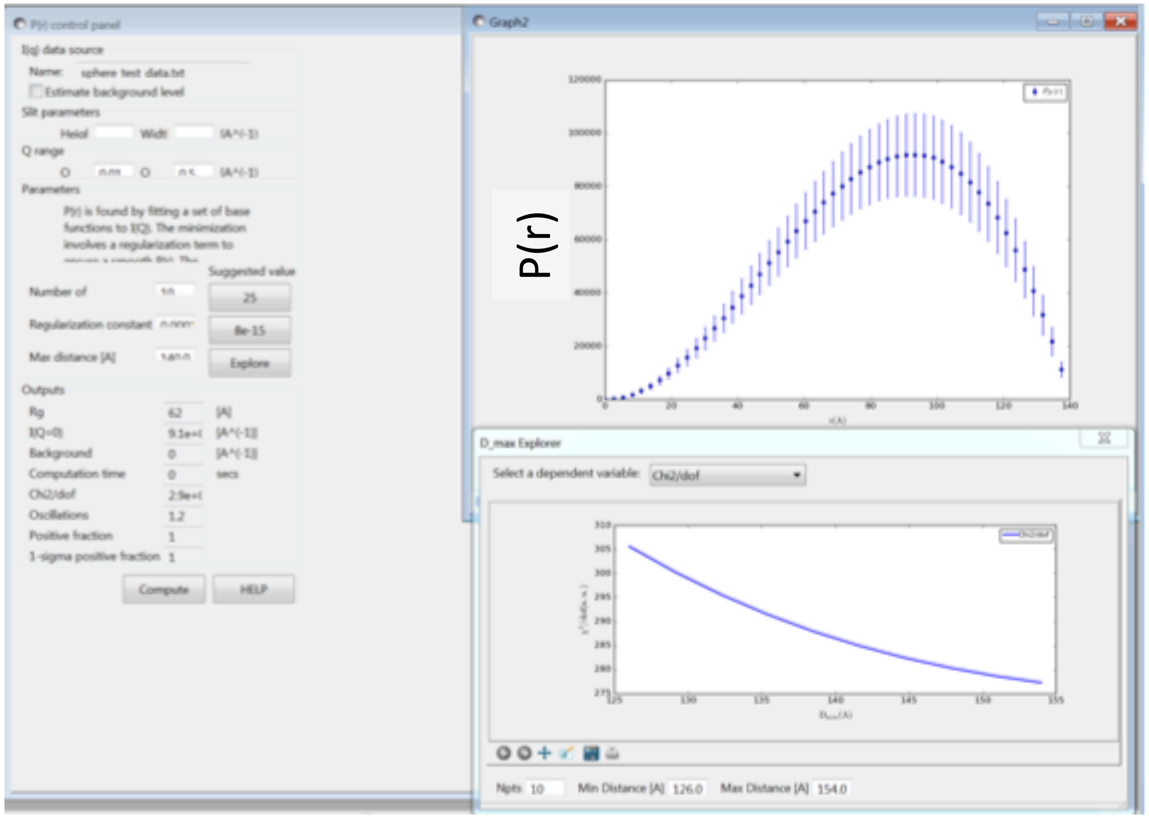
- Handles 1D and 2D data
- Form and structure factors for various particle shapes
- Different optimizers (Bayesian Statistics)
- Allows polydispersity
- Simultaneous and batch fitting
- Plugin models



Data Explorer
Fit panel - Active Fitting Optimizer: Levenberg-Marquardt
FitPage1
Model Parameters
Polydispersity and Orientational Distribution
Fitting
Graph10
Graph11
Compute
Fit
Help

Other useful tools

- P(r) inversion
- Model independent analysis
- SLD calculator
- Slit size calculator
- Kiessing thickness calculator
- Q resolution estimator
- Generic scattering calculator



P(r) control panel

I(q) data source

Name: sphere test data.txt

☐ Estimate background level

Slit parameters

Heigt [illegible] Widt [illegible] [A^(-1)]

Q range

Q [illegible] Q [illegible] [A^(-1)]

Parameters

P(r) is found by fitting a set of base functions to I(Q). The minimization involves a regularization term to

		Suggested value
Number of	[illegible]	25
Regularization constant	[illegible]	8e-15
Max distance [A]	[illegible]	Explore

Outputs

Rg	62	[A]
I(Q=0)	9.1e+(	[A^(-1)]
Background	0	[A^(-1)]
Computation time	0	secs
Chi2/dof	2.9e+(	
Oscillations	1.2	
Positive fraction	1	
1-sigma positive fraction	1	

Compute HELP

Graph2

D_max Explorer

Select a dependent variable: Chi2/dof

Npts 10 Min Distance [A] 126.0 Max Distance [A] 154.0

SasView History

2006

2012 Community driven project
Releases after code camps

2014 ESS joined the project

2016 SINE2020 two employees at ESS

SasView manpower 2016

Management Team:

- Paul Butler (NIST)
- Mathieu Doucet (ORNL)
- Andrew Jackson (ESS)
- Steve King (ISIS)

- Jurrian Bakker (TUD)
- Wim Bouwman (TUD)
- Miguel Gonzales (ILL)
- Richard Heenan (ISIS)
- Dirk Honecker (ILL)
- Paul Kienzle (NIST)
- Jeff Kryzwon (NIST)
- Ricardo Leal (ORNL)
- David Mannicke (ANSTO)
- Torben Nielsen (ESS)
- Lewis O'Driscoll (ISIS)
- Steve Parnell (TUD)
- Wojciech Potrzebowski (ESS)
- Piotr Rozyczko (ESS)
- Adam Washington (Sheffield)

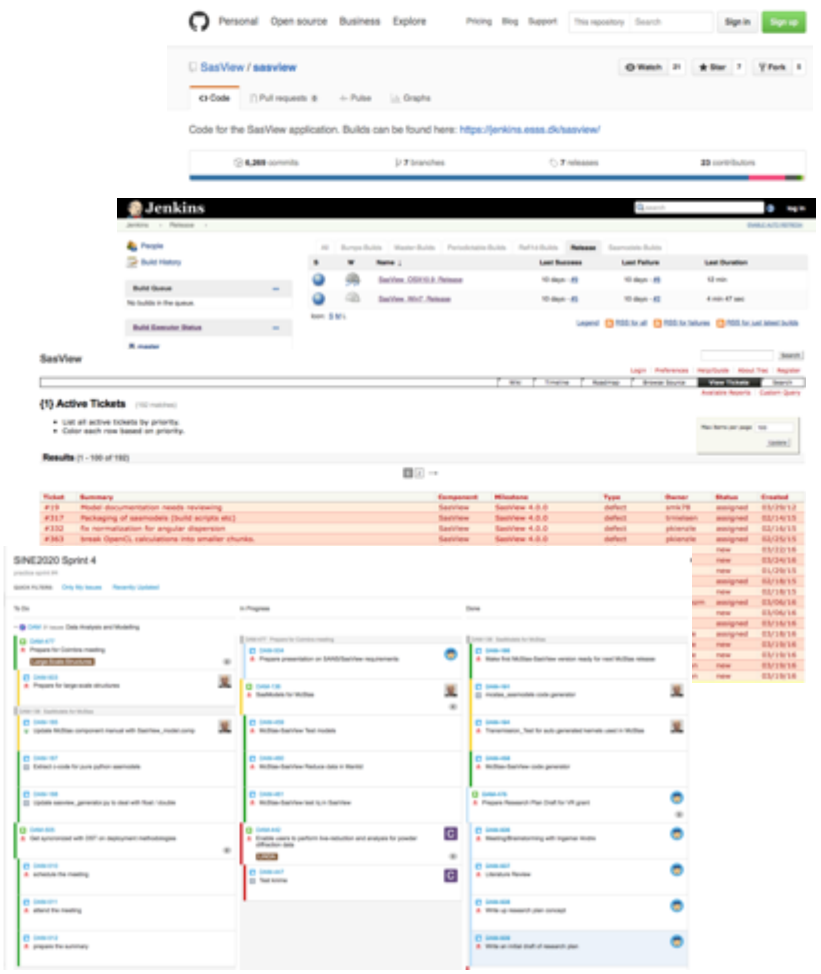
- and thanks to the many previous contributors, particularly Jae Hie Cho and Alina Gervaise

SasView Development Workflow

- Code hosted at github
- Trac issue tracking system
- Build system hosted at ESS -DMSC

- Biweekly video conference
- Code camp once or twice per year

- Web-based and built-in documentation
- Tutorial
- Mailing lists



SasView / sasview

Code for the SasView application. Builds can be found here: https://jenkins.esss.dk/sasview/

6,269 commits · 7 branches · 7 releases · 23 contributors

Jenkins

SasView

{1} Active Tickets

- List all active tickets by priority.
- Color each row based on priority.

Results (1 - 100 of 192)

Ticket	Summary	Component	Milestone	Type	Owner	Status	Created
#19	Model documentation needs reviewing	SasView	SasView 4.0.0	defect	smk78	assigned	03/29/12
#317	Packaging of sasmodels (build scripts etc)	SasView	SasView 4.0.0	defect	krzywon	assigned	02/14/15
#332	fix normalization for angular dispersion	SasView	SasView 4.0.0	defect	pkienzle	assigned	02/16/15
#363	break OpenCL calculations into smaller chunks.	SasView	SasView 4.0.0	defect	pkienzle	assigned	02/25/15

SINE2020 Sprint 4

SasView Releases

2006

2012 Community driven project
Releases after code camps

2014 ESS joined the project

2016 SINE2020 two employees at ESS

2006

2011

SasView 1.0 released 2012

SasView 2.0 released

SasView 3.0 released 2013

2014

SasView 3.1 released 2015

SasView 4.0 just released 2016

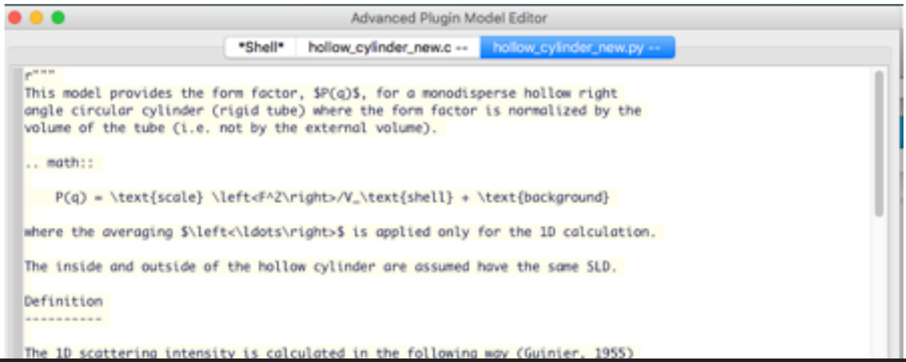
SasView 4.0 is out

- SasView "built-in" models have been separated out into an independent package
- Easy to add custom user models (including advanced)
- Support for OpenCL
- All model documentation has been reviewed and updated
- Number of minor bugs fixed

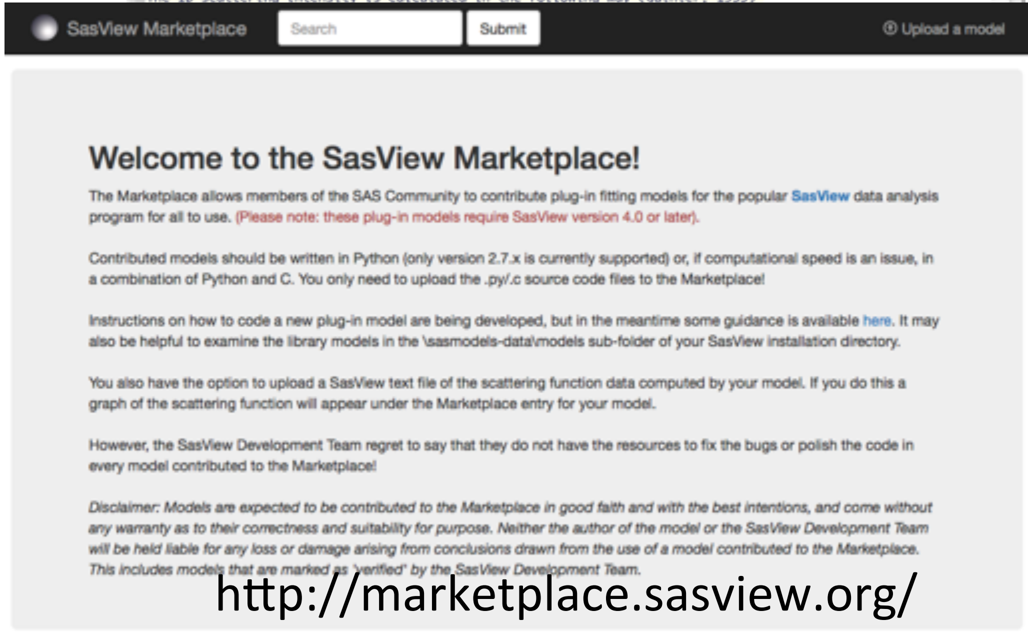
Available from:
https://github.com/SasView/sasview/releases/tag/v4.0

Addition of custom models

- Plugin model editor
- Python and c files
- Syntax and performance testing
- SasView Marketplace



```
r"""
This model provides the form factor, $P(q)$, for a monodisperse hollow right
angle circular cylinder (rigid tube) where the form factor is normalized by the
volume of the tube (i.e. not by the external volume).

.. math::

    P(q) = \text{scale} \left<F^2\right>/V_\text{shell} + \text{background}

where the averaging $\left<\ldots\right>$ is applied only for the 1D calculation.

The inside and outside of the hollow cylinder are assumed have the same SLD.

Definition
----------

The 1D scattering intensity is calculated in the following way (Guinier, 1955)
```



Welcome to the SasView Marketplace!

The Marketplace allows members of the SAS Community to contribute plug-in fitting models for the popular SasView data analysis program for all to use. (Please note: these plug-in models require SasView version 4.0 or later).

Contributed models should be written in Python (only version 2.7.x is currently supported) or, if computational speed is an issue, in a combination of Python and C. You only need to upload the .py/.c source code files to the Marketplace!

Instructions on how to code a new plug-in model are being developed, but in the meantime some guidance is available here. It may also be helpful to examine the library models in the \sasmodels-data\models sub-folder of your SasView installation directory.

You also have the option to upload a SasView text file of the scattering function data computed by your model. If you do this a graph of the scattering function will appear under the Marketplace entry for your model.

However, the SasView Development Team regret to say that they do not have the resources to fix the bugs or polish the code in every model contributed to the Marketplace!

Disclaimer: Models are expected to be contributed to the Marketplace in good faith and with the best intentions, and come without any warranty as to their correctness and suitability for purpose. Neither the author of the model or the SasView Development Team will be held liable for any loss or damage arising from conclusions drawn from the use of a model contributed to the Marketplace. This includes models that are marked as 'verified' by the SasView Development Team.

http://marketplace.sasview.org/

Credits: Lewis O'Driscoll (ISIS)

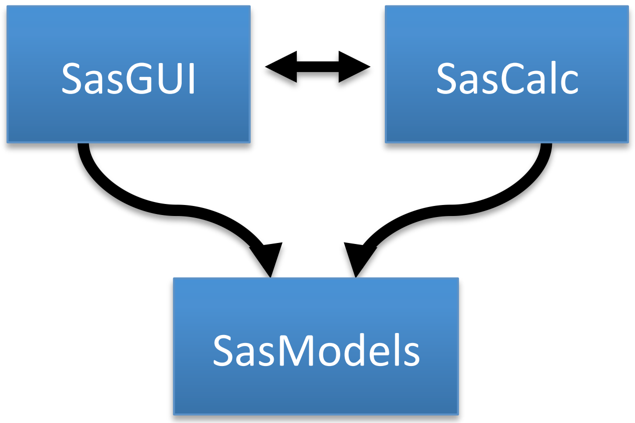
SINE2020 goals

- Code modularization
- New API and CLI
- New GUI
- Optimization of algorithms for real time analysis
- Extension with SASFit models

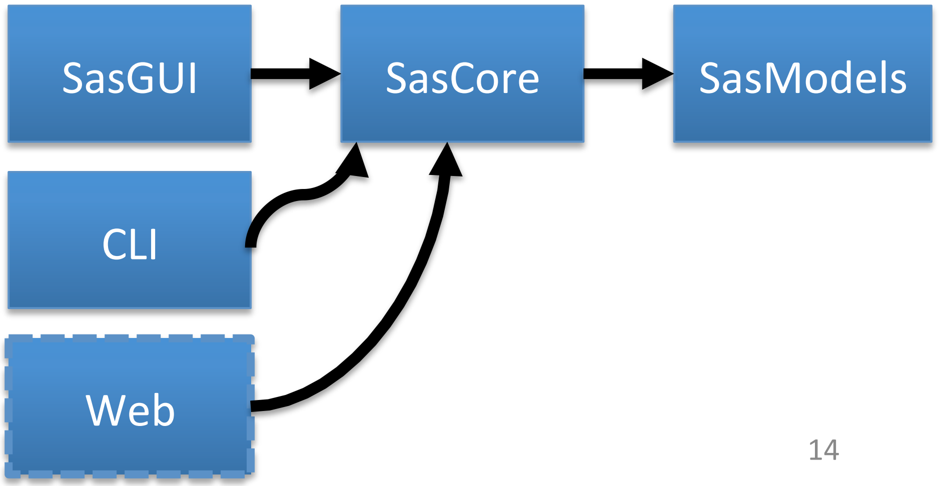
New API and CLI

- SasView "built-in" models moved to an independent package
- Separation of the model calculation code from the GUI
- Module dependencies considerably reduced
- Opens up for use of "built-in" models in pipelines and easy exchange of fitting engines

Before:



Prospposed:

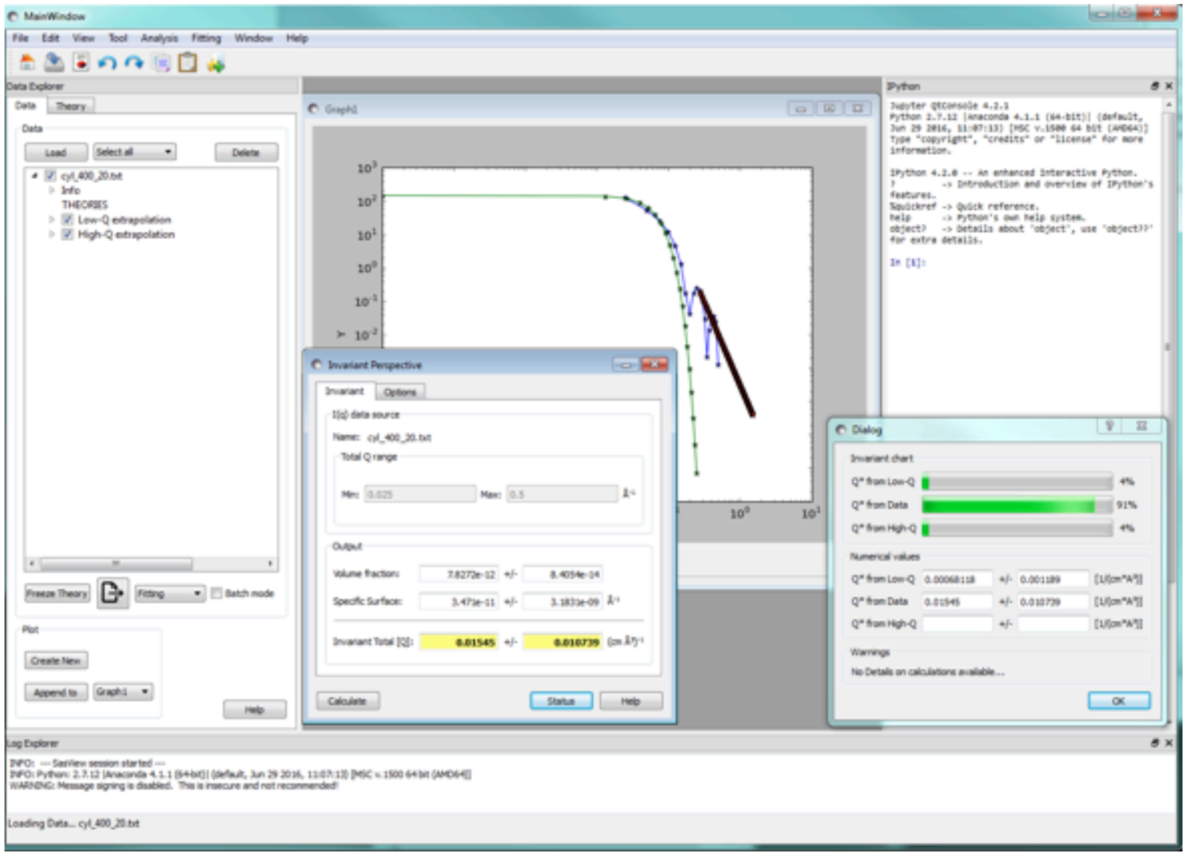


GUI modernization

- Transition from wx-python to PyQt
- Platform consistency dialogs look and behave the same across all platforms
- Long term maintainability
- Ease of development (Qt designer)
- Clean separation of UI and code
- Native thread support

GUI modernization

- Working prototype
- Plotting
- Fitting and P(r) panels
- Plugin model editor
- PyQt4/Qt4 on Python 2.7
- Planned to convert to PyQt5/Qt5 once the code migrated to Python 3
- Multithreading with twisted



```
Jupyter QtConsole 4.2.1
Python 2.7.12 |Anaconda 4.1.1 (64-bit)| (default,
Jun 29 2016, 11:07:13) [MSC v.1500 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more
information.

IPython 4.2.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's
features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??'
for extra details.

In [1]:
```

Invariant Perspective — Invariant | Options

I(q) data source
Name: cyl_400_20.txt
Total Q range
Min: 0.025 Max: 0.5 Å⁻¹

Output

Volume fraction:	7.8272e-12	+/-	8.4054e-14	
Specific Surface:	3.471e-11	+/-	3.1831e-09	Å⁻¹
Invariant Total [Q]:	0.01545	+/-	0.010739	(cm Å³)⁻¹

Dialog

Invariant chart

Q* from Low-Q	4%
Q* from Data	91%
Q* from High-Q	4%

Numerical values

Q* from Low-Q	0.00068118	+/-	0.001189	[1/(cm*A^3)]
Q* from Data	0.01545	+/-	0.010739	[1/(cm*A^3)]
Q* from High-Q		+/-		[1/(cm*A^3)]

Warnings
No Details on calculations available...

Log Explorer

```
INFO:  --- SasView session started ---
INFO: Python: 2.7.12 |Anaconda 4.1.1 (64-bit)| (default, Jun 29 2016, 11:07:13) [MSC v.1500 64 bit (AMD64)]
WARNING: Message signing is disabled.  This is insecure and not recommended!
```

Loading Data... cyl_400_20.txt

Credits: Piotr Rozyczko (ESS, DMSC)

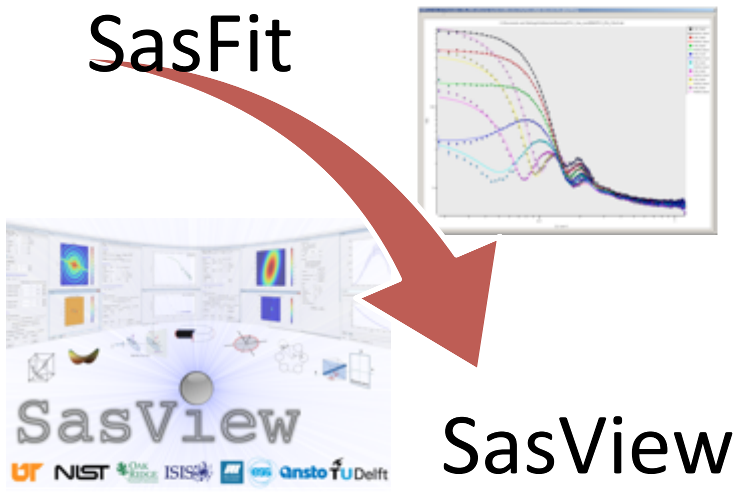
Code optimization

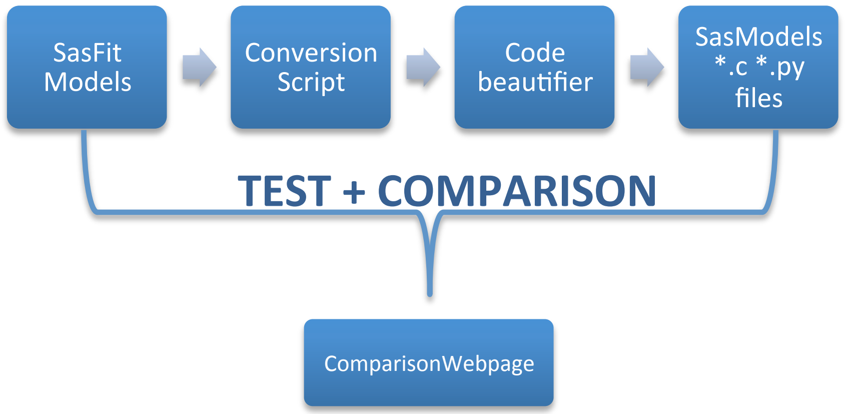
EUROPEAN SPALLATION SOURCE

- Aim to perform real-time data analysis
- Majority of SasView models already ported to GPU
- For modern cards single and double precision enabled
- Planned support for multiple CPUs/GPUs
- Further optimization and testing

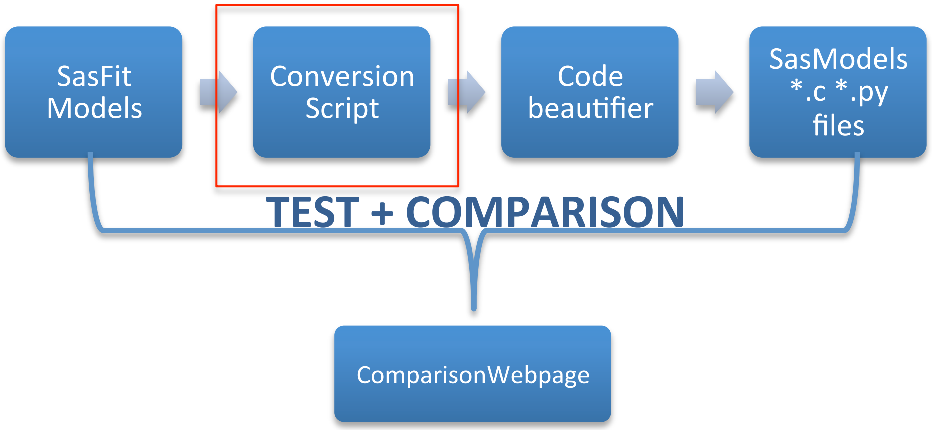
Extending SasView with SasFit models

- SasFit – a software for analyzing and plotting SAS data developed at PSI
- SasFit has a large collection of form and structure factors
- Framework for SasFit to SasView models conversion (including testing and comparison)
- Models will be uploaded to marketplace





Conversion script

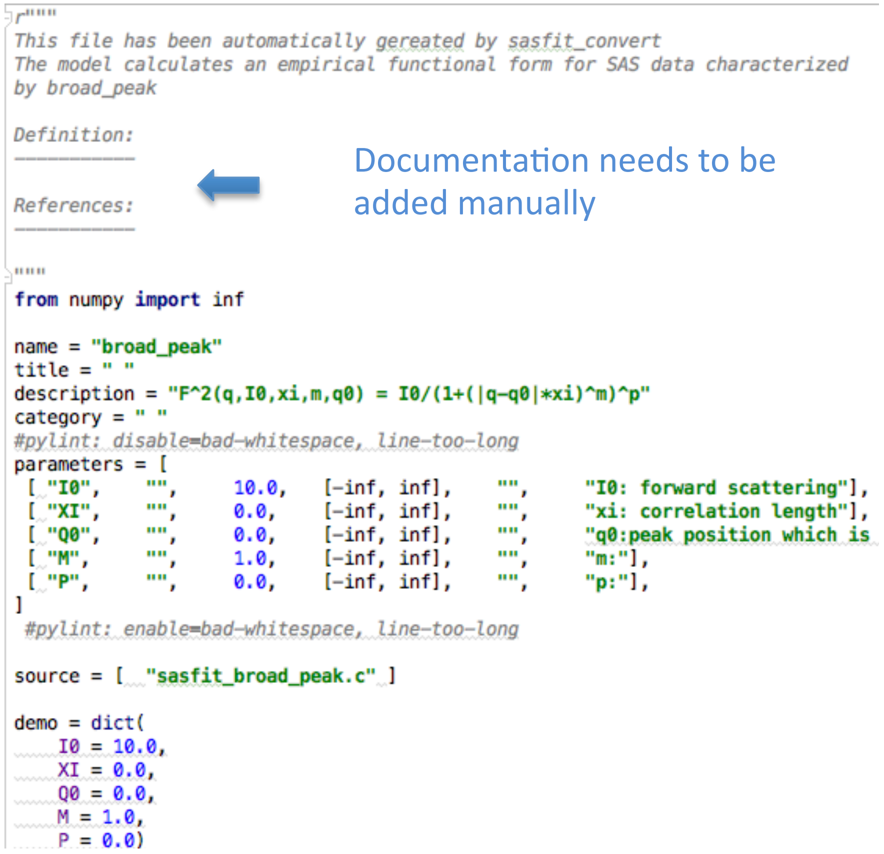


- Extracts code from SasFit models
- Reads in model and parameters description
- Reads in parameters defaults

- Outputs SasView plugin models (*.c and *.py files)
- Creates description, parameters table, demo section, etc.
- Supplies SasModels functions (Iq, Iqxy, form_volume)
- F(q) supplied but not yet fully used by SasView

Conversion script

Python file

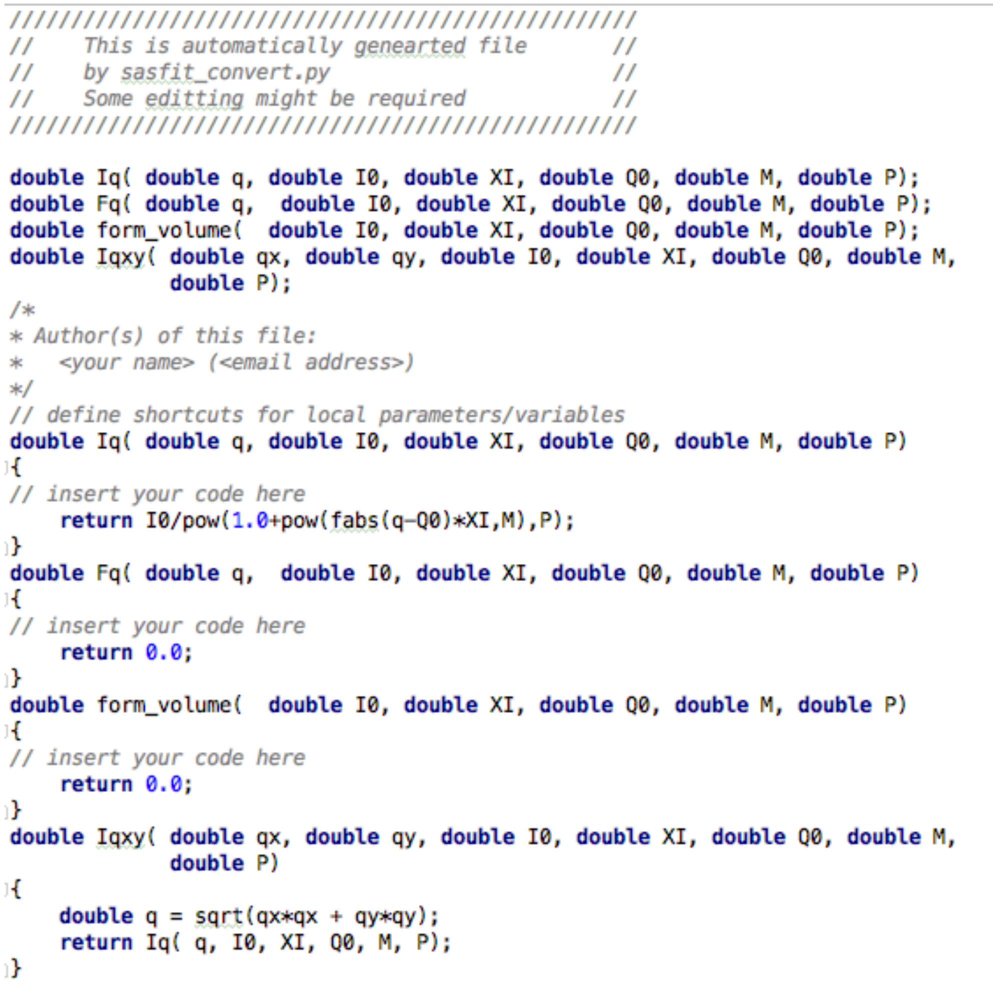


```
r"""
This file has been automatically gereated by sasfit_convert
The model calculates an empirical functional form for SAS data characterized
by broad_peak

Definition:
-----------

References:
-----------

"""
from numpy import inf

name = "broad_peak"
title = " "
description = "F^2(q,I0,xi,m,q0) = I0/(1+(|q-q0|*xi)^m)^p"
category = " "
#pylint: disable=bad-whitespace, line-too-long
parameters = [
 [ "I0",     "",     10.0,   [-inf, inf],    "",     "I0: forward scattering"],
 [ "XI",     "",     0.0,    [-inf, inf],    "",     "xi: correlation length"],
 [ "Q0",     "",     0.0,    [-inf, inf],    "",     "q0:peak position which is
 [ "M",      "",     1.0,    [-inf, inf],    "",     "m:"],
 [ "P",      "",     0.0,    [-inf, inf],    "",     "p:"],
]
 #pylint: enable=bad-whitespace, line-too-long

source = [  "sasfit_broad_peak.c" ]

demo = dict(
    I0 = 10.0,
    XI = 0.0,
    Q0 = 0.0,
    M = 1.0,
    P = 0.0)
```

Documentation needs to be added manually

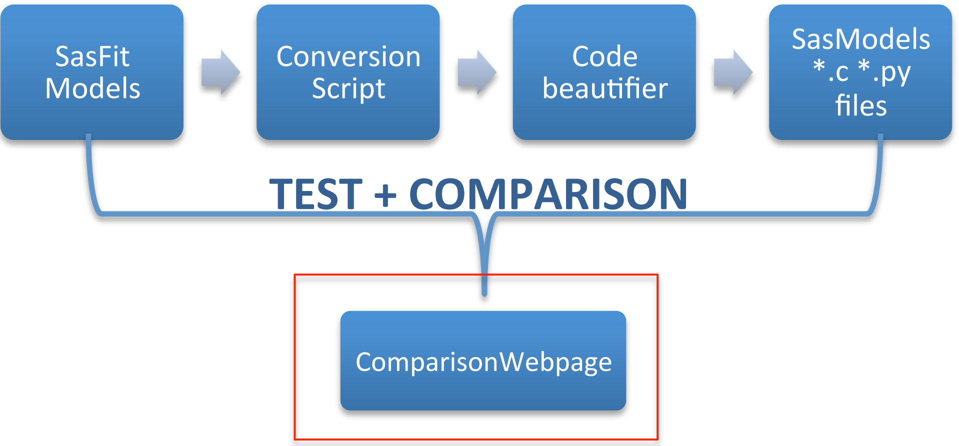
Unit tests needs to be added manually

C file



```
////////////////////////////////////////////////
//    This is automatically genearted file      //
//    by sasfit_convert.py                      //
//    Some editting might be required           //
////////////////////////////////////////////////

double Iq( double q, double I0, double XI, double Q0, double M, double P);
double Fq( double q,  double I0, double XI, double Q0, double M, double P);
double form_volume(  double I0, double XI, double Q0, double M, double P);
double Iqxy( double qx, double qy, double I0, double XI, double Q0, double M,
             double P);
/*
* Author(s) of this file:
*    <your name> (<email address>)
*/
// define shortcuts for local parameters/variables
double Iq( double q, double I0, double XI, double Q0, double M, double P)
{
// insert your code here
    return I0/pow(1.0+pow(fabs(q-Q0)*XI,M),P);
}
double Fq( double q,  double I0, double XI, double Q0, double M, double P)
{
// insert your code here
    return 0.0;
}
double form_volume(  double I0, double XI, double Q0, double M, double P)
{
// insert your code here
    return 0.0;
}
double Iqxy( double qx, double qy, double I0, double XI, double Q0, double M,
             double P)
{
    double q = sqrt(qx*qx + qy*qy);
    return Iq( q, I0, XI, Q0, M, P);
}
```

SasView-SasFit model comaprison



- There is a number of overlapping models
- Unique models need to be identified
- Comparison by name is not sufficient
- Requires community effort

Summary

- SasView is an open source, collaboratively developed software for the analysis and the modeling of small angle scattering
- SasView 4.0 is out!
- Sine2020 efforts involve:
  - Code modularization and optimization
  - New GUI development
  - Extension with new models
- Our goals:
  - Easier maintainability
  - Increased reliability
  - Better user experience
  - Real-time data analysis