

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

15 October 2016

Data analysis that makes user happy

Experiment

Streaming

Reduction

Analysis

Results

Data analysis software requirements:

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

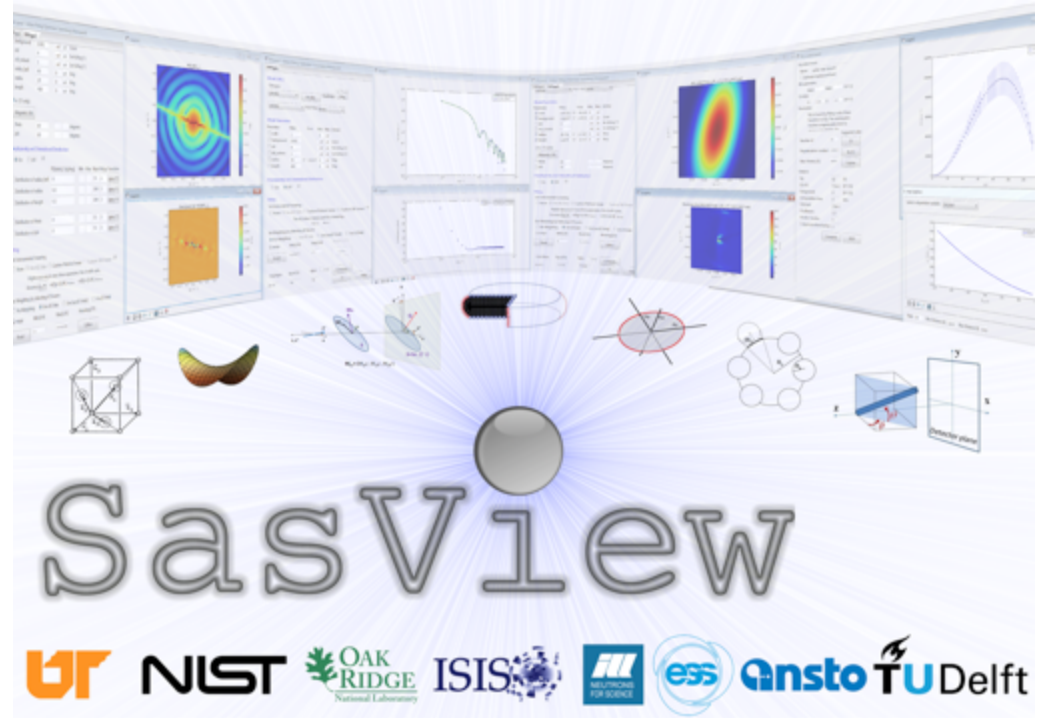


Outline

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

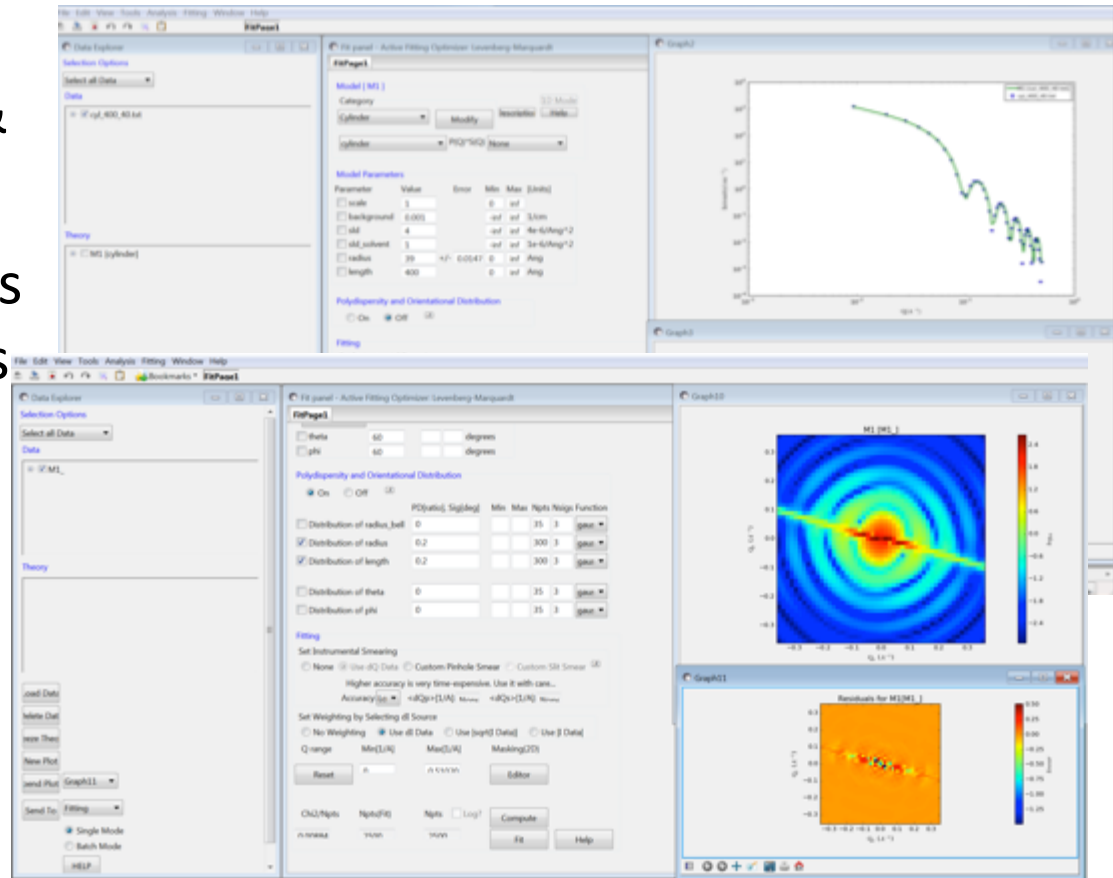
SasView - Small Angle Scattering Analysis Software Package

- 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



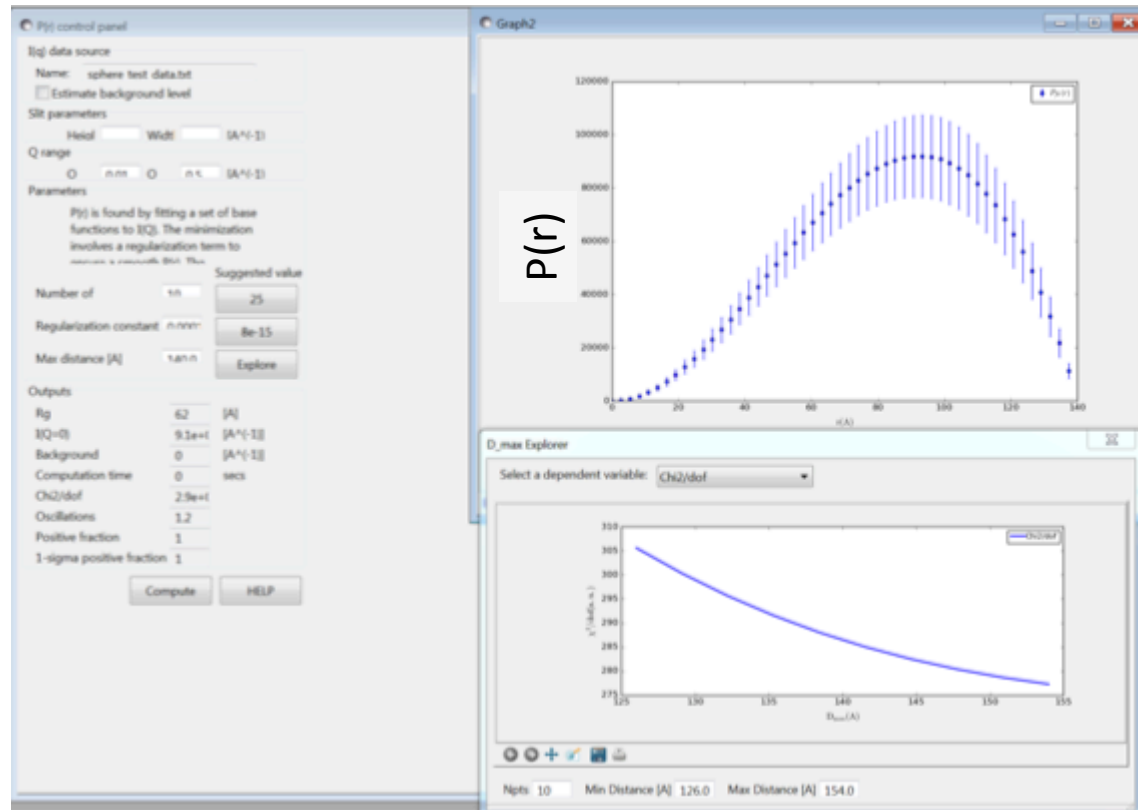
SasView - Fitting

- Handles 1D and 2D data
- support for canSAS XML & NXcanSAS formats
- Form and structure factors for various particle shapes
- Different optimizers (Bayesian Statistics)
- Allows polydispersity
- Simultaneous and batch fitting



Other useful tools

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



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:

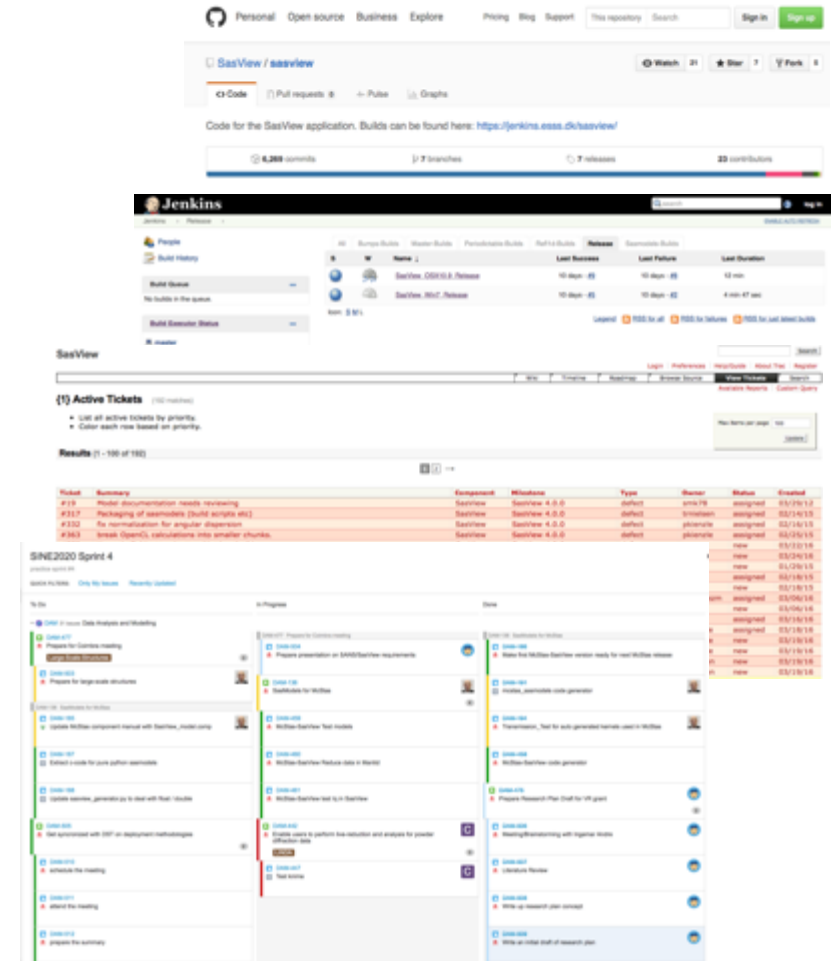
- Paul Butler (NIST)
- Mathieu Doucet (ORNL)
- Andrew Jackson (ESS)
- Steve King (ISIS)



- 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



The image displays three screenshots illustrating the SasView development workflow:

- GitHub Repository:** Shows the SasView / sasview repository with 4,389 commits, 7 branches, 7 releases, and 23 contributors.
- Jenkins Build System:** Displays the Jenkins interface for the SasView project, showing build history and status.
- Trac Issue Tracking:** Shows a list of active tickets (issues) for the SasView project, including details like priority, status, and assignee.

SasView Releases

2006



2012

Community driven project
Releases after code camps

2014

ESS joined the project

2016

SINE2020 two employees at ESS

2006

2011

2012

SasView 1.0 released
SasView 2.0 released
SasView 3.0 released

2013

2014

SasView 3.1 released

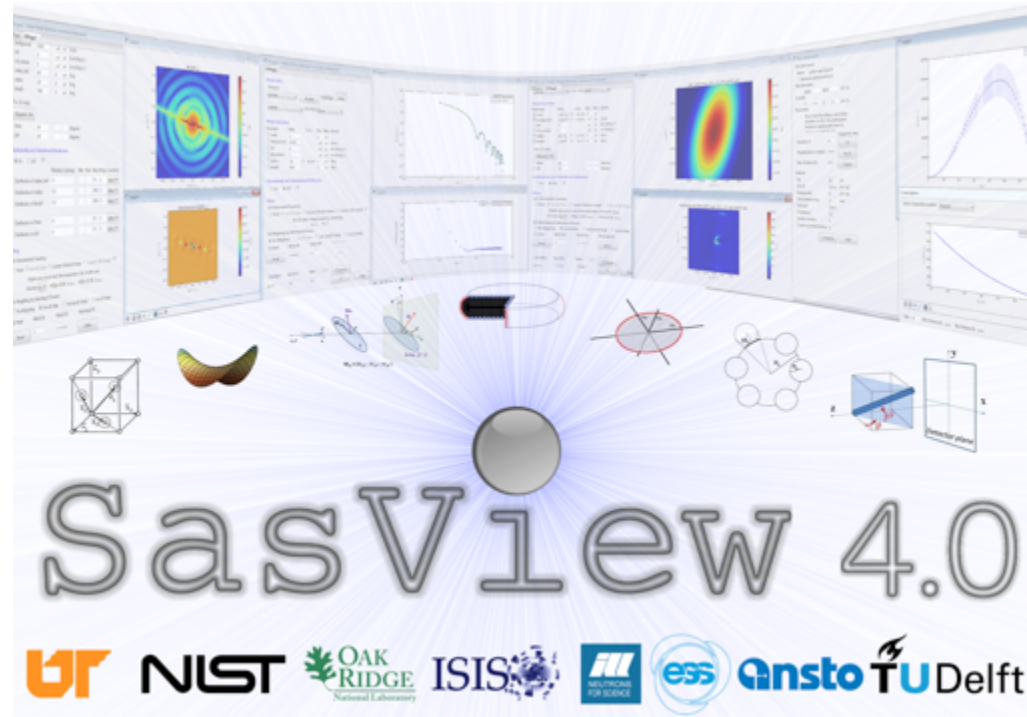
2015

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

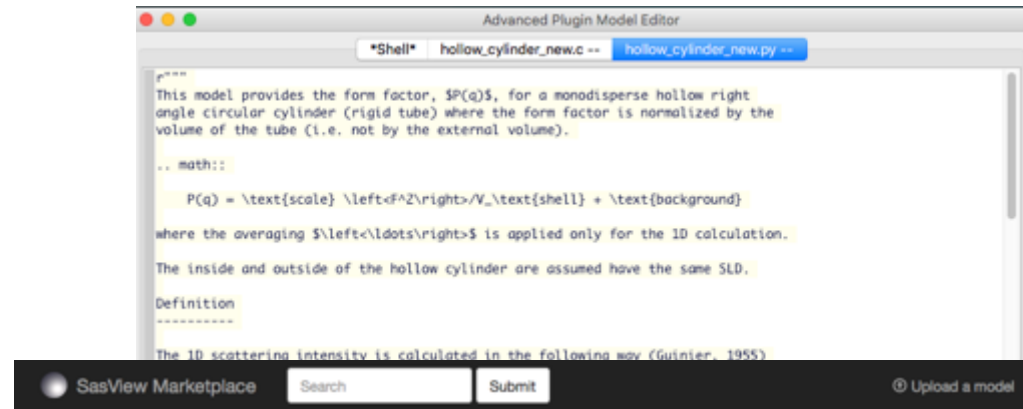


Available from:

<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



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/>

SINE2020 goals

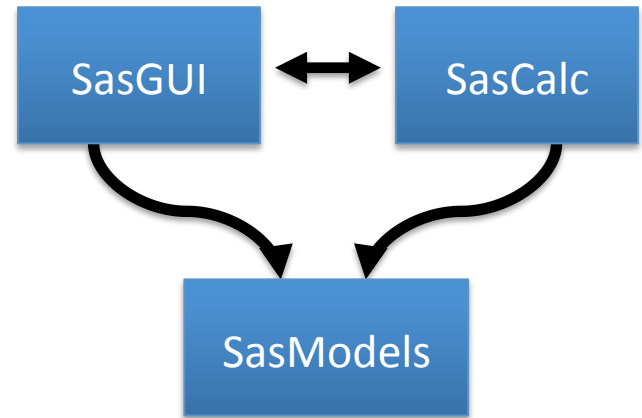


- Code modularization
- New API and CLI
- New GUI
- Optimization of algorithms for real time analysis
- Extension with SASFit models

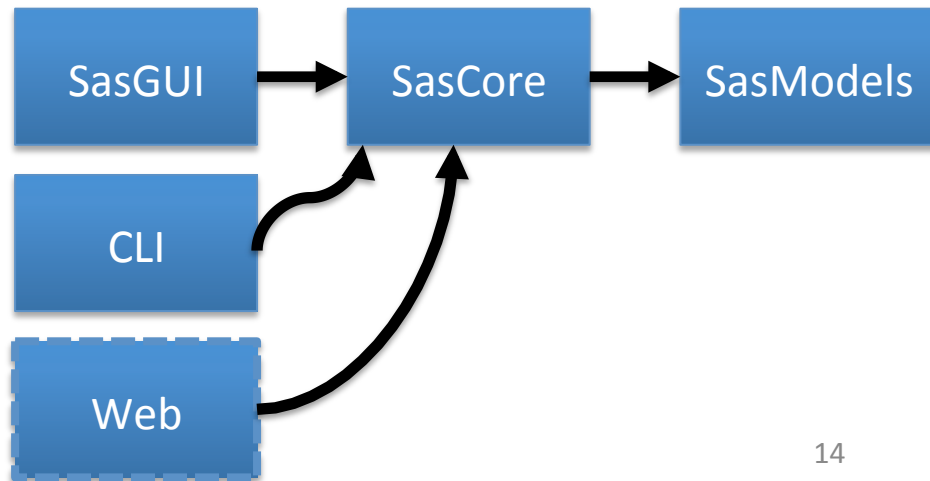
New API and CLI

- 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:



Proposed:



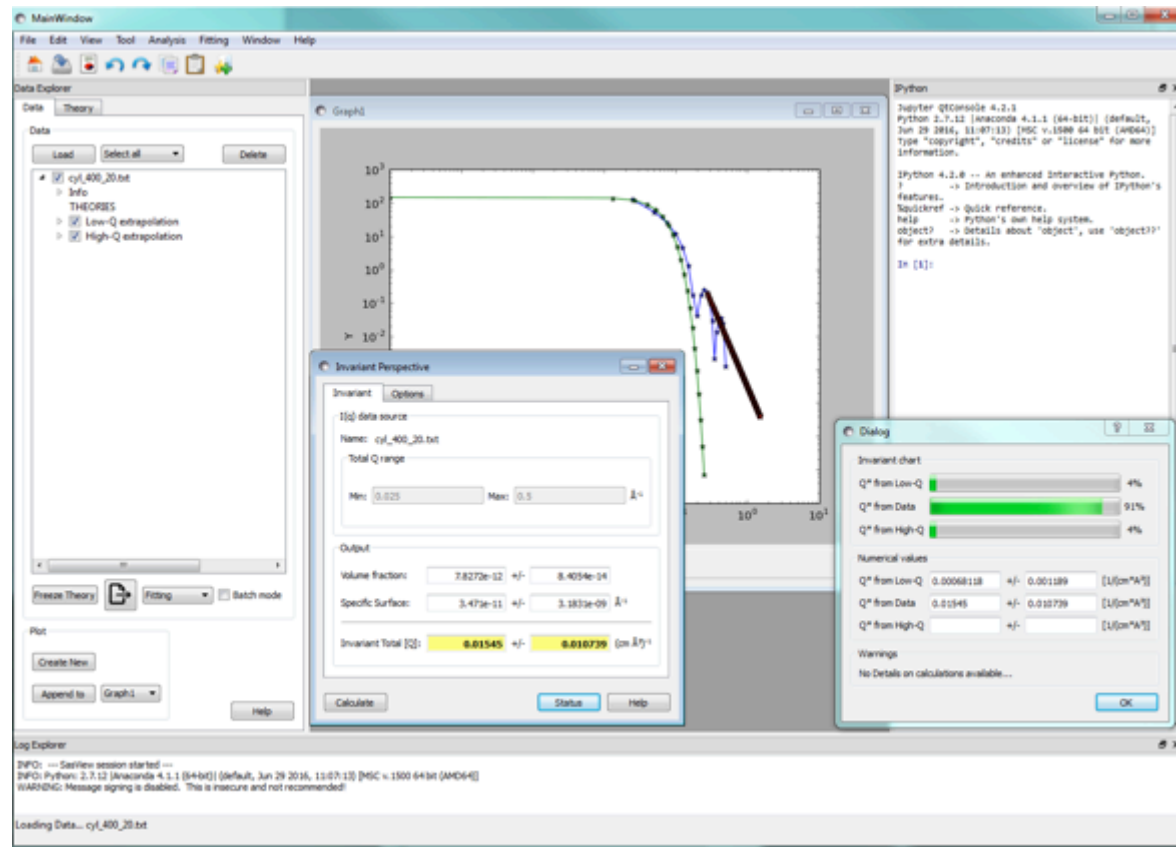
GUI modernization

- Transition from wx-python to PyQt
- Platform consistency dialogs look and behave the same across all platforms
- Maintainability and reliability
- Ease of development (Qt designer)
- Native thread support



GUI modernization

- Working prototype
- Plotting
- P(r) perspective
- 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



Credits: Piotr Rozyczko (ESS, DMSC)

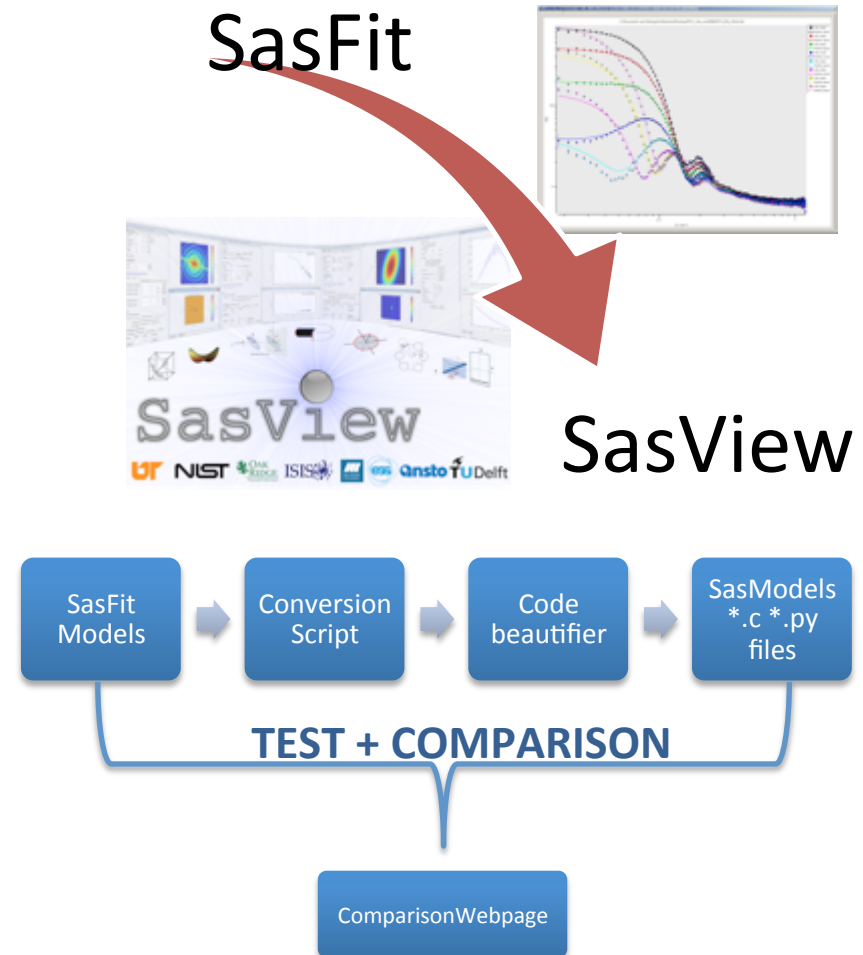
Code optimization

- 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

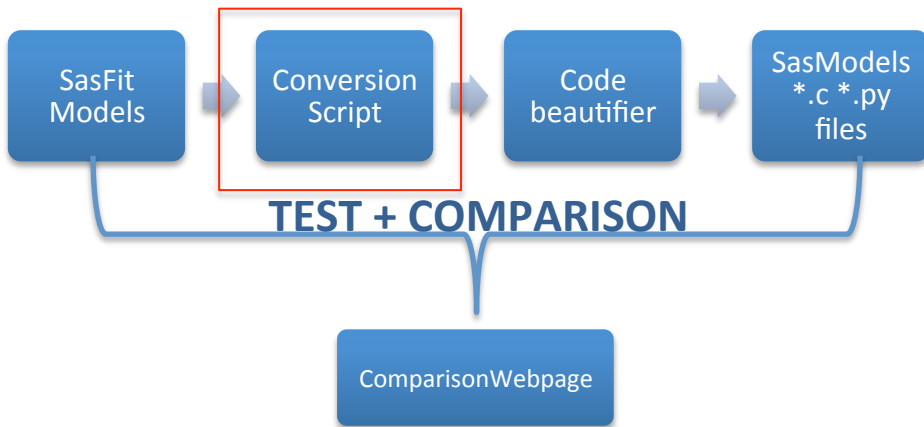


Extending SasView with SasFit models

- 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 SasView functions (Iq, Iqxy, form_volume)
- F(q) supplied but not yet fully used by SasView

Conversion script

Python file

```
"""
This file has been automatically generated 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 generated file //
// by sasfit_convert.py //
// Some editing 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);

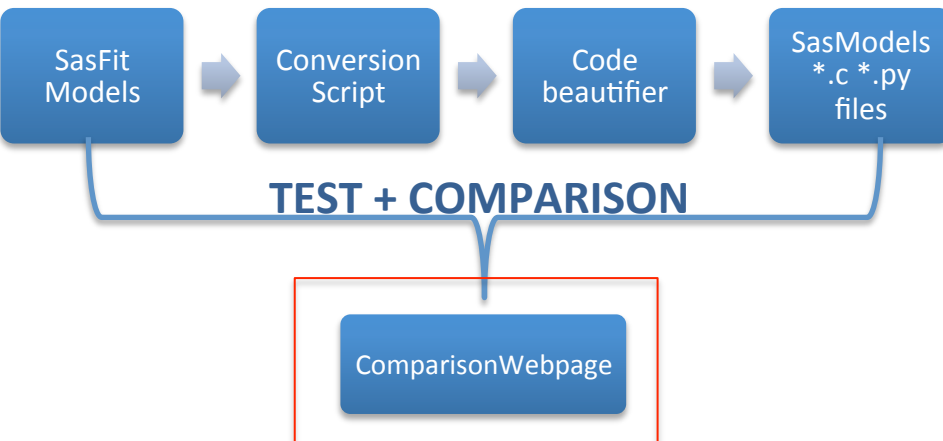
/*
 * Author(s) of this file:
 * <your name> (<email address>)
 */
// define shortcuts for local parameters/variables
double Iq( double q, double I0, double XI, double Q0, double M, double P)
{
    // insert your code here
    return I0/pow(1.0+pow(fabs(q-Q0)*XI,M),P);
}

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 comparison



- 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 SAS data
- SasView in Sine2020 project
 - Code modularization and optimization
 - New GUI development
 - Extension with new models
- SasView 4.0 is out!
- SasView 4.1 is coming soon:
 - Correlation functions
 - SESANS