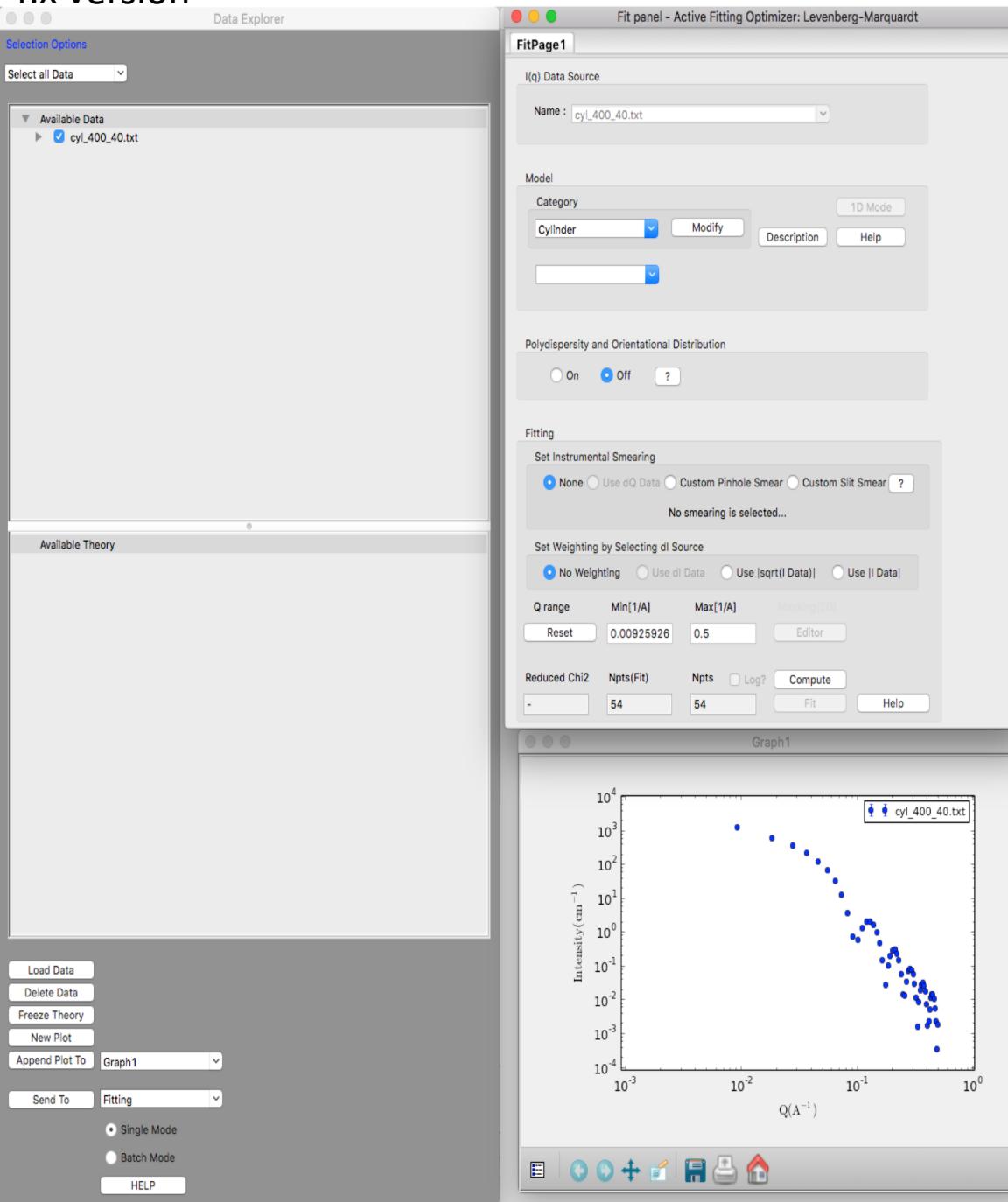


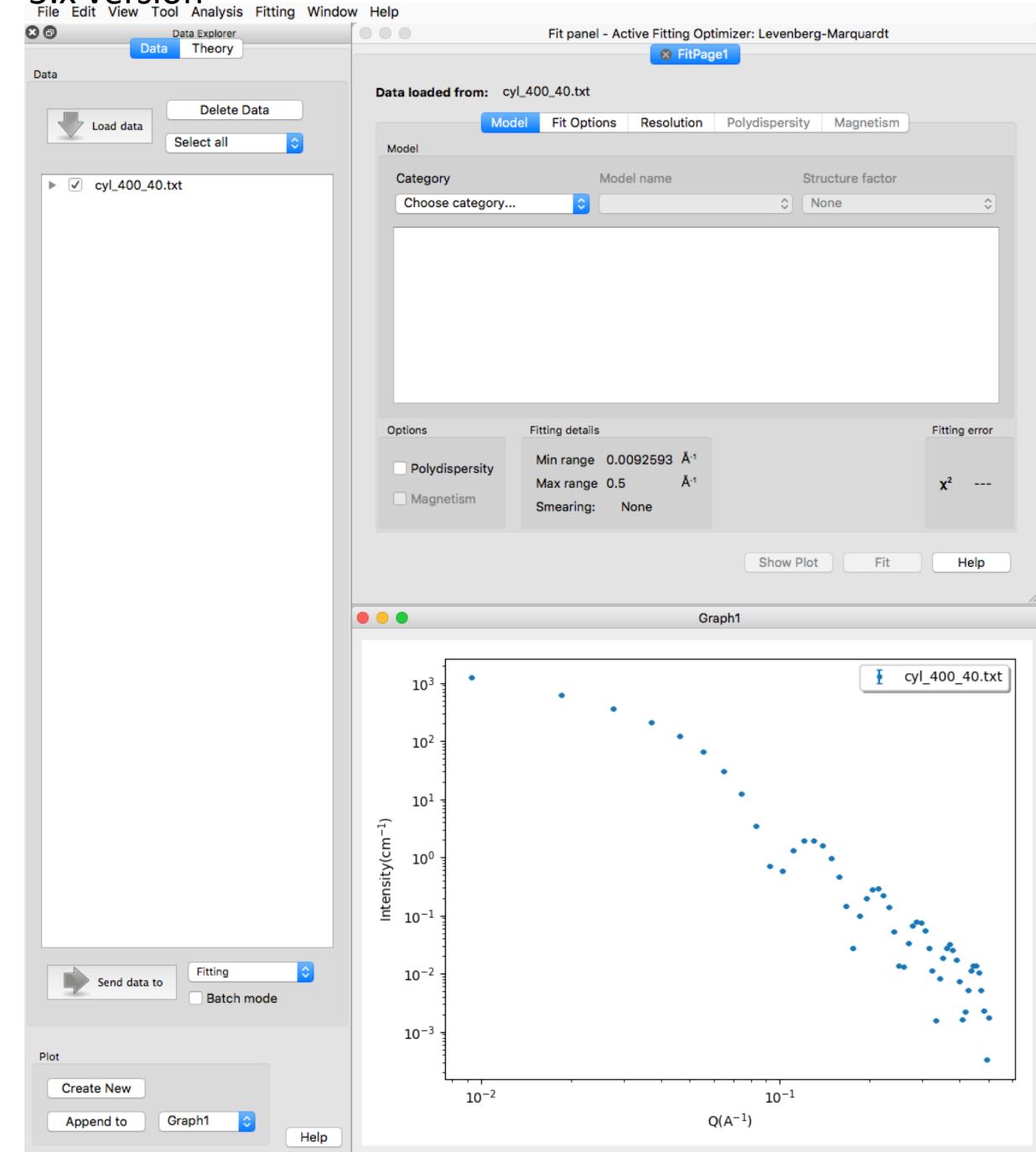
# SasView

## new developers intro

# 4.x version



# 5.x version



# Setting up developers environment

Conda based installation

- <http://trac.sasview.org/wiki/DevNotes/DevGuide/CondaDevSetup5.0>
- <http://trac.sasview.org/wiki/DevNotes/DevGuide/CondaDevSetup4.x>

Github account

Trac account

JIRA account

# Trac and website

**SasView**

Wiki Timeline Roadmap Browse Source View Tickets New Ticket Search

wiki: [WikiStart](#)

**SasView Developer Site**

**Latest Release** [GitHub](#)

**Nightly Builds** [Jenkins](#)

**Source Code**

```
git clone https://github.com/SasView/sasview; git clone https://github.com/Sasview/sasmmodels
```

**The Developers' Corner**

Memos, tools, 'How to' guides, and other resources for developers can be found on these [Developer Notes](#) pages  
Please comment on [SasView 5.0 feedback form](#) to go to sas2018 SasView user meeting attendees.

**For Tutorial Authors**

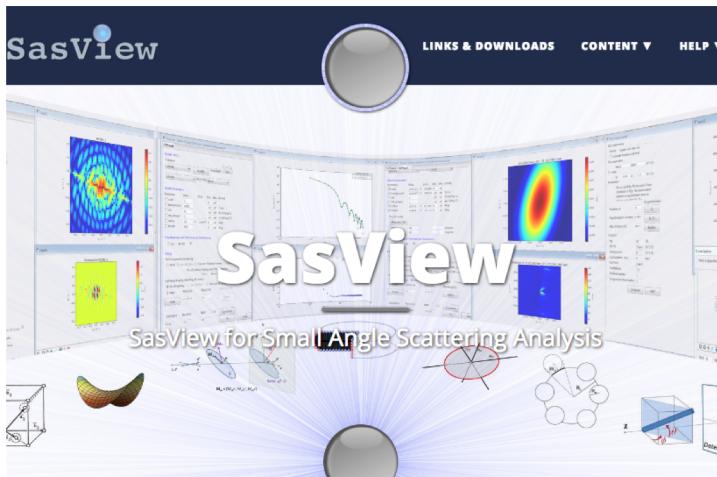
The philosophy behind the Next-Generation Tutorials for SasView are here: [TutorialsTNG](#)  
The current list of Next-Generation Tutorials, and instructions on how to write and contribute them are here: [TutorialsTNGForAuthors](#)

**Code Camps**

- Code Camp-IX will be held in Grenoble France, 25-31 March 2019.
- Code Camp-VIII was held at ESS in Lund, 3-9 September 2018.
- Code Camp-VII was held at ESS/DMSC in Copenhagen, 23-29 October 2017
- Code Camp-VI was held at the EPN Campus (ILL & ESRF) in Grenoble, 4-11 April 2017
- Code Camp-V was held at the Spallation Neutron Source, Oak Ridge, Tennessee, 4-11th October, 2016
- Code Camp-IV was held at the Reactor Institut Delft, 15-23rd March, 2016
- Code Camp-III was held at the ESS DMSC, 11 February - 20 February 2015
- Code Camp-II was held at ISIS, 30 March - 6 April 2014
- Code Camp-I was held at NIST, 3 April - 7 April 2013

[Edit this page](#) [Attach file](#)

<http://trac.sasview.org/>



Download The Latest Release Version of SasView

[Download Version 4.2.1](#)

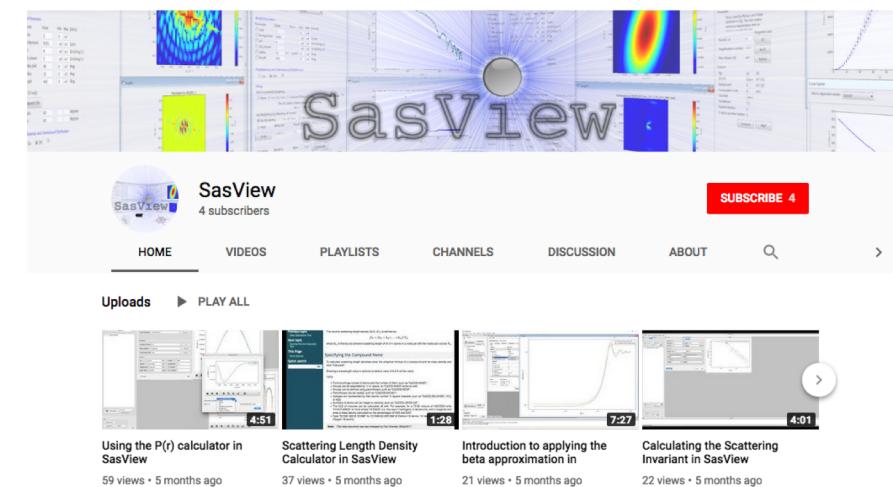
Released on February 18, 2019. [See what's new!](#)

**SasView Code Camp IX**

SasView Code Camp IX will be hosted by the ILL and held at the ESRF... [Read more...](#)

A SAS Community Project launched from the NSF DANSE effort  
SasView is a Small Angle Scattering Analysis Software Package, originally developed as part of the NSF DANSE project under the name SansView, now managed by an

<http://www.sasview.org/>



YouTube channel

# JIRA (SasView 5.x only)

ESS DRAM Kanban

Kanban board

QUICK FILTERS: SasView 5 final QA SasView New GUI Only My Issues SpinW SasView Reduction McStas Imaging DAM Research Partner led projects ESS tasks Construction Board ▾

29 of 183 Backlog

5 of 13 Selected

2 of 15 In Progress Max 10

0 of 10 Blocked

7 of 44 Review Max 4

0 of 170 Done

Release... We're only showing recently modified issues. Looking for an older issue?

**SASVIEW-1156**  
↑ Documentations unclear on how to do Constrained or Simultaneous Fit  
[New SasView GUI](#)  
Documentation, SASVIEW\_5.0\_Final

**SASVIEW-1079**  
↑ QA: testing confunc perspective  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1134**  
↑ Model Documentation missing - Centos  
[New SasView GUI](#)  
Documentation, SASVIEW\_5.0\_Final

**SASVIEW-1111**  
↑ Documentation - fitting  
[New SasView GUI](#)  
Documentation, SASVIEW\_5.0\_Final

**SASVIEW-924**  
↑ Tutorial updates  
[New SasView GUI](#)  
Documentation, SASVIEW\_5.0\_Final

**SASVIEW-1133**  
↑ Model Documentation missing - Ubuntu  
[New SasView GUI](#)  
Documentation, SASVIEW\_5.0\_Final

**SASVIEW-1011**  
✗ volfraction - choose between computed and user-supplied (like ER)  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final, beta-approximation

**SASVIEW-789**  
✗ GUI in Qt: add more flexibility in specifying Plotter/Plotter2D options  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1127**  
✗ Changing fit parameters reverts y axis back to linear  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1173**  
✗ Fit options, Chi2/Npts line edit without function?  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-975**  
✗ Add code signing step to the build process  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1260**  
✗ Create ESS AppleID  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1259**  
✗ Weighting not properly applied to data  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1250**  
✗ Find a solution to conda pulling new module releases and breaking things.  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1173**  
✗ Fit panel -> Show plot plots model and residual without fit  
[New SasView GUI](#)  
QA, SASVIEW\_5.0\_Final

**SASVIEW-1114**  
✗ QA: OSX GUI looks strange  
[New SasView GUI](#)  
QA, SASVIEW\_5.0\_Final

**SASVIEW-1169**  
✗ Crashed on button clicking: FittingWidget  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1136**  
✗ Fontconfig error  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

**SASVIEW-1255**  
✗ Tutorial doesn't open  
[New SasView GUI](#)  
SASVIEW\_5.0\_Final

# Github and build servers

**SasView**  
SasView is a Small Angle Scattering (SAS) analysis package for the analysis of 1D and 2D scattering data directly in inverse space.

<http://www.sasview.org> [developers@sasview.org](mailto:developers@sasview.org)

**Repositories** 11   **People** 37   **Teams** 4   **Projects** 0   **Settings**

Find a repository...   Type: All   Language: All   New

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

Python ★ 23 ⚡ 20 Updated 4 hours ago

**sasmodes**  
Package for calculation of small angle scattering models using OpenCL. Builds here: <https://jenkins.esss.dk/sasview/view/Sasmodes-Builds/>

Python ★ 5 ⚡ 19 BSD-3-Clause Updated 16 hours ago

**sasview.github.io**  
HTML ★ 1 MIT Updated 13 days ago

**documents**  
Documents relating to the SasView project

HTML Updated 18 days ago

**docs**  
Sphinx documents of SasView

HTML Updated 29 days ago

**tutorials**  
This repository is where contributed tutorials for SasView live!

Shell Updated 29 days ago

**sasmode-marketplace**  
A website for sharing custom sasmode files

<https://github.com/SasView>

**Jenkins**

Jenkins >

**People**   **Build History**   **Project Relationship**   **Check File Fingerprint**

**Build Queue (1)**   SasView\_Tutorials

**Build Executor Status**

All	Bumps-Builds	Master-Builds	Periodicable-Builds	Refl1d-Builds
S	W	Name ↓		
●	●	<a href="#">bumps_Ubuntu14.04_test</a>		
●	●	<a href="#">Periodicable-Ubuntu14.04_test</a>		
●	●	<a href="#">Refl1d_Ubuntu14.04_test</a>		
●	●	<a href="#">sasmodes_Ubuntu14.04_test</a>		
●	●	<a href="#">SasView_OSX10.10</a>		
●	●	<a href="#">SasView_OSX10.10_PullRequest</a>		
●	●	<a href="#">SasView_OSX10.10_Release</a>		
●	●	<a href="#">SasView_OSX10.9</a>		
●	●	<a href="#">SasView_OSX10.9_Release</a>		
●	●	<a href="#">SasView_Tutorials</a>		
●	●	<a href="#">SasView_Ubuntu14.10</a>		
●	●	<a href="#">SasView_Ubuntu14.10_test</a>		
●	●	<a href="#">SasView_Ubuntu_PullRequest</a>		
●	●	<a href="#">SasView_Win7</a>		
●	●	<a href="#">SasView_Win7_32bit_PullRequest</a>		
●	●	<a href="#">SasView_Win7_32bit_Release</a>		
●	●	<a href="#">SasView_Win7_64bit</a>		
●	●	<a href="#">SasView_Win7_64bit_conda</a>		
●	●	<a href="#">SasView_Win7_64bit_PullRequest</a>		
●	●	<a href="#">SasView_Win7_Release</a>		

Icon: S M L

<https://jenkins.esss.dk/sasview/>

**Jenkins**

Jenkins >

**People**   **Build History**   **Project Relationship**   **Check File Fingerprint**

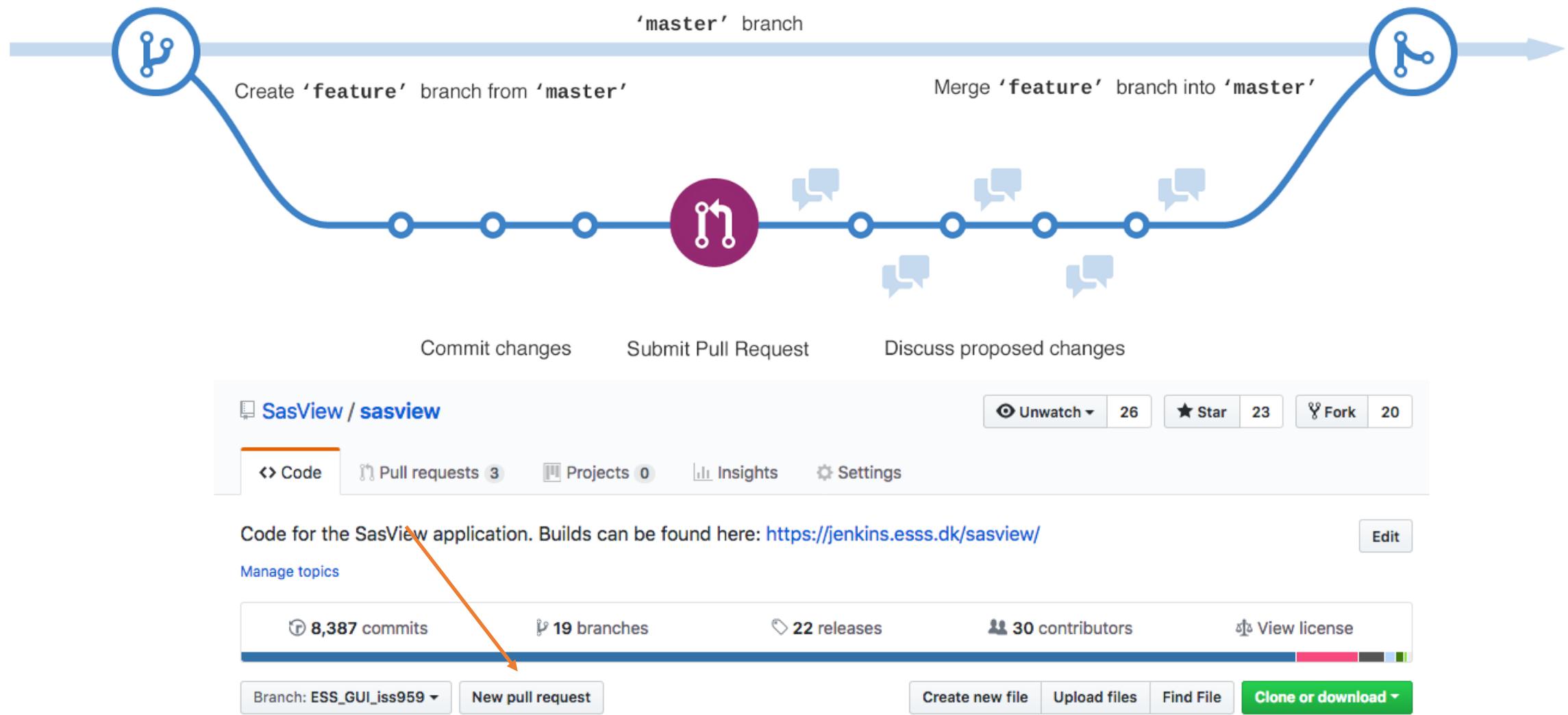
**Build Queue**   No builds in the queue.

**Build Executor Status**

All	W	Name ↓
●	●	<a href="#">SasView-Centos</a>
●	●	<a href="#">SasView-OSX10.13-test</a>
●	●	<a href="#">SasView-OSX1013</a>
●	●	<a href="#">SasView-OSX1013-py27</a>
●	●	<a href="#">SasView-Ubuntu</a>
●	●	<a href="#">SasView-Ubuntu1804</a>
●	●	<a href="#">SasView-Windows10</a>
●	●	<a href="#">SasView-Windows10-beta</a>
●	●	<a href="#">SasView-Windows7</a>
●	●	<a href="#">SasView_Tutorials</a>

<https://jenkins.esss.dk/sasview-beta/>

# Pull Requests



# Pull request testing

Use dQ/|Q| for 2-D resolution #209

Open pkienzle wants to merge 1 commit into release-4.2.2 from ticket-1242-2d-resolution

The screenshot shows a GitHub pull request page for pull request #209. The title is "Use dQ/|Q| for 2-D resolution". The status bar indicates "pkienzle wants to merge 1 commit into release-4.2.2 from ticket-1242-2d-resolution". The commit history shows a single commit "use dq |q| for 2-D resolution" by pkienzle, which has been cherry-picked from master. The pull request has been merged, as indicated by the green checkmark next to the commit message. The right sidebar shows the review status: "Reviewers" (krzywon - Request), "Assignees" (No one—assign yourself), "Labels" (None yet), "Projects" (None yet), and "Milestone" (No milestone). The notifications section shows "1 participant" and an "Unsubscribe" button. The bottom of the page features a "Merge pull request" button and a note about opening it in GitHub Desktop or viewing command line instructions. A comment from rozyczko is visible, adding a commit that referenced this pull request.

Magic phrase:  
ready for testing on OSX  
ready for testing on Linux  
ready for testing on Win64  
ready for testing on Win32

# SasView components

SasView

SasModels

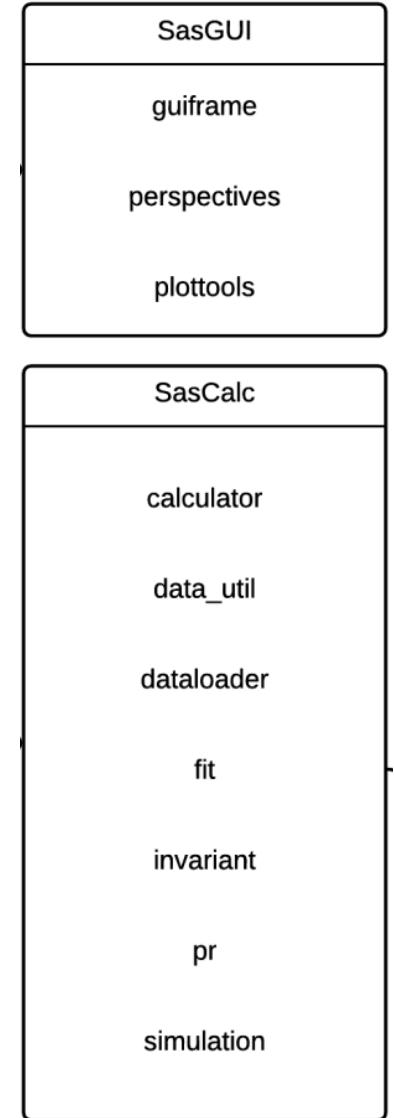
Bumps

SasGui

SasCalc

# SasGUI

- 4.x developed with wx python
- 5.0 developed with Qt: src/sas/sasgui/qtgui in (ESS\_GUI branch)
- SasGui consists of:
  - Perspectives: src/sas/sasgui/perspectives (fitting, invariant, calculators, corfunc)
  - GuiFrame: src/sas/sasgui/guiframe (non-perspective gui elements)
  - PlotTools: src/sas/sasgui/plottols (ploting tools)

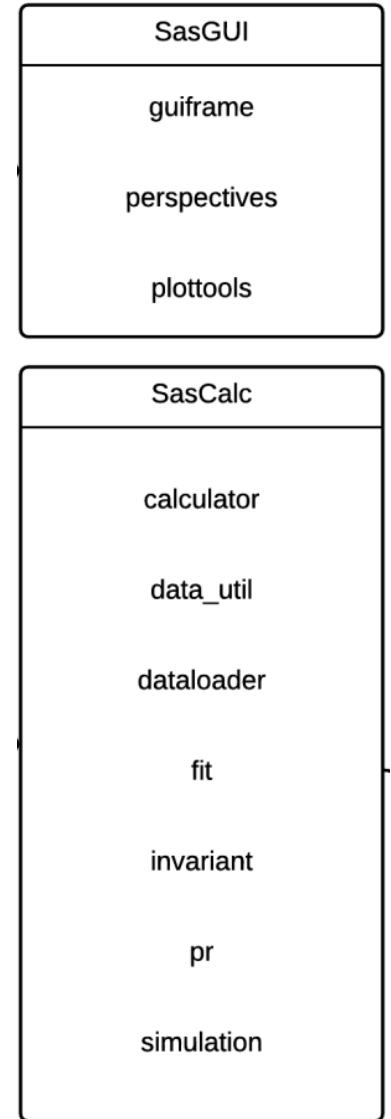


# SasGUI – less obvious cases

- Model editor:  
`src/sas/sasgui/perspectives/calculator/model_editor.py`
- Category manager:  
`src/sas/sasgui/guiframe/CategoryManager.py`
- Settings for OpenCL:  
`src/sas/sasgui/perspectives/fitting/gpu_options.py`
- Fitting options (interface to bumps):  
`bumps/gui/fit_dialog`

# SasCalc

- Back-end calculations for sasgui perspective
- Data fitting:
  - src/sas/sascalc/fit/
- Pr inversion
  - src/sas/sascalc/pr/
- Data loader: src/sas/sascalc/dataloader/loader.py



# SasCalc

```
from sas.sascalc.dataloader.loader import Loader
from sas.sascalc.pr.invertor import Invertor

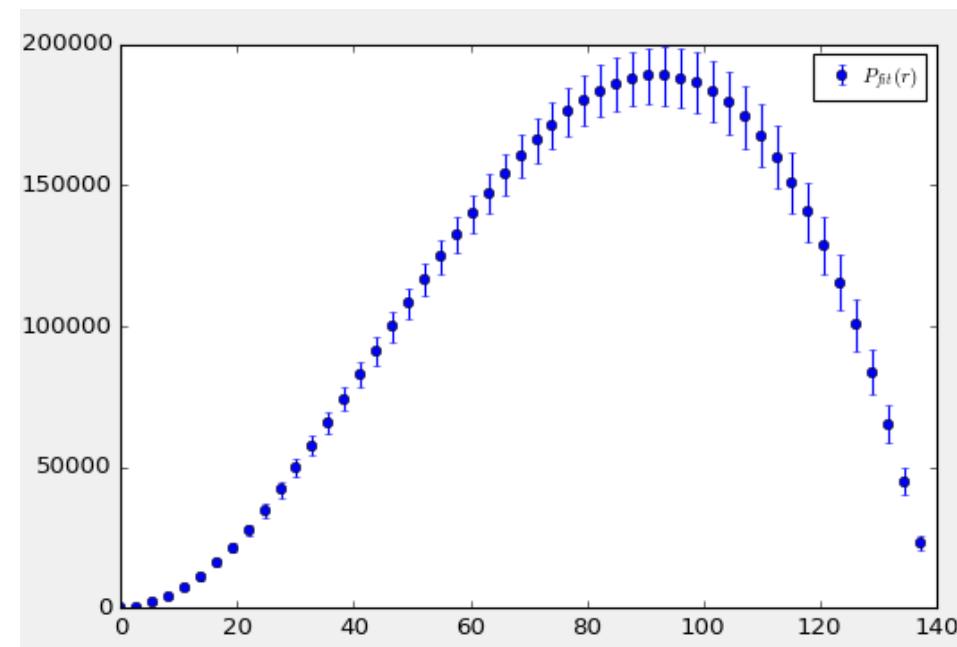
loader = Loader()
test_data = loader.load("sphere_80.txt")

pr = Invertor()

# Set data
pr.x = test_data.x
pr.y = test_data.y

# Perform inversion and show graph
x, y = pr.invert()

import matplotlib.pyplot as plt
plt.plot(x, y)
plt.show()
```



And more at: [https://github.com/SasView/documents/blob/master/Meetings/User\\_Meeting\\_2018/SasView%2BCLI%2Boverview.html](https://github.com/SasView/documents/blob/master/Meetings/User_Meeting_2018/SasView%2BCLI%2Boverview.html)

# SasModels

- Models located in: sasmodels/models
- Three different types of models:
  - Python only: broad\_peak.py
  - Python with embedded C: stickyhardsphere.py
  - Python and separate C: barbell.py and barbell.c

# SasModels essential components

- Model Description: Docstring
- Name, title, short description
- Parameters table
- C files to be included during compilation
- Unit tests

# SasModels functions

- `Iq` - 1D scattering intensity functions for the case where the scatterer is randomly oriented
- `Iqxy` - the 2D scattering intensity functions provide  $I(Q, \phi)$  for an oriented system as a function of a  $Q$
- `Form_volume` - calculates the volume of particle  $V$ , which is used to normalize form factor.
- `ER` – calculates effective radius
- `VR` - calculate particle volume/total volume for shape models
- `INVALID(v)` returns `False` if `v.parameter` is invalid for some parameter or other (e.g., `v.bell_radius < v.radius`).

# SasModels core modules

- Model computation:
  - `sasmmodels/sasview_model.py` – interface to sasview
  - `sasmmodels/kernel.py` -interface to all kernel models
  - `sasmmodels/kerneldll.py` –dll driver for c kernels
  - `sasmmodels/kernelpy.py` – python driver
  - `sasmmodels/kernelcl.py` – opencl driver
  - `sasmmodels/core.py` - prepares the model for the execution
  - `sasmmodels/generate.py` – generates model file based on the template

# Sasmodels compare

- Computes model and compares between different platforms (opencl, dll)
- Allows for comparison with old sasview version
- `./compare.sh modelname -1d`
- `./compare.sh modelname -2d`
- Multiple precision, q ranges options

# Unit tests

- SasView tests are located: sasview/test
  - calculatorview
  - corfunc
  - fileconverter
  - pr\_inversion
  - sascalculator
  - sasdataloader
  - sasguiframe
  - sasinvariant
  - sasrealspace
- Sasmodels unit tests defined inside models

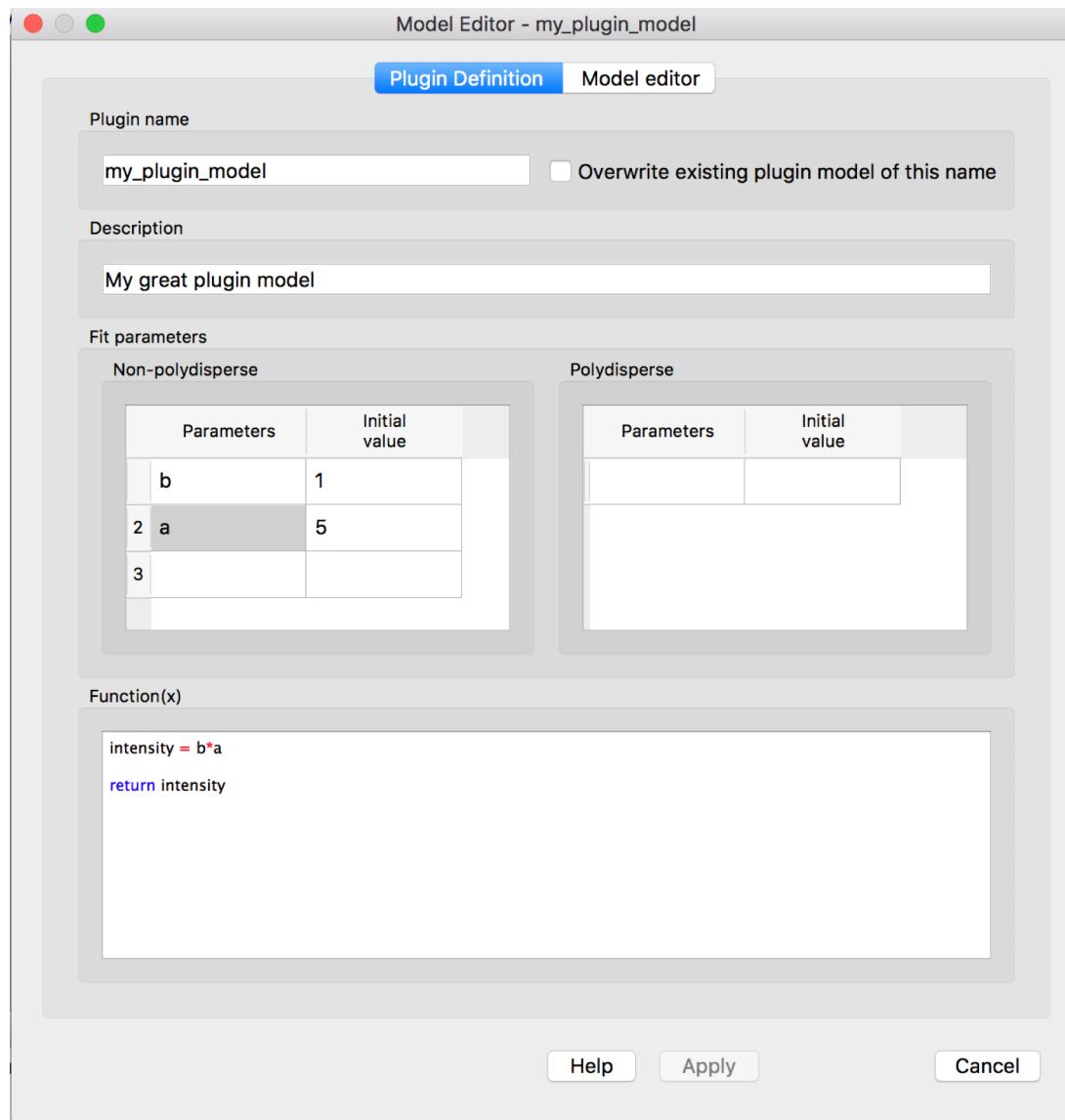
# Running bumps with sasmodels

- Sasmodels example contains data set for oriented rod-like shape
- `sasmodels/examples/fit.py` (sasmodels interface to bumps)
- Running
  - bumps fit.py cylinder –preview
  - Fit.py accepts two arguments: type of model and view (radial and tnagential)

# Custom settings and models

- Local configuration
  - `~/.sasview/config/custom_config.py`
- Plugin models:
  - `~/.sasview/plugin_models/`
- Model categories:
  - `~/.sasview/categories.json`

# Plugin model editor (Fitting->Add Custom Model)



The screenshot shows the 'Model' tab of the Model Editor. It displays the generated Python code for the plugin model:

```
Calculates my_plugin_model.

My great plugin model

References
-----
Authorship and Verification
-----
***Author:*** --- **Date:** 2019YYY-03m-19d
***Last Modified by:** --- **Date:** 2019YYY-03m-19d
***Last Reviewed by:** --- **Date:** 2019YYY-03m-19d
"""

from math import *
from numpy import inf

name = "my_plugin_model"
title = "User model for my_plugin_model"
description = """My great plugin model"""

parameters = [
    # [name, units, default, [lower, upper], type, description],
    ['b', '1.0', [-inf, inf], "float", "b parameter"],
    ['a', '5.0', [-inf, inf], "float", "a parameter"],
]
def Iq(x, b, a):
    """Absolute scattering"""
    intensity = b*a

    return intensity
## uncomment the following if Iq works for vector x
#Iq.vectorized = True

#def Iqxy(x, y, b, a):
#    """Absolute scattering of oriented particles."""
#    ...
#    # return oriented_form(x, y, args)
## uncomment the following if Iqxy works for vector x, y
#Iqxy.vectorized = True
```

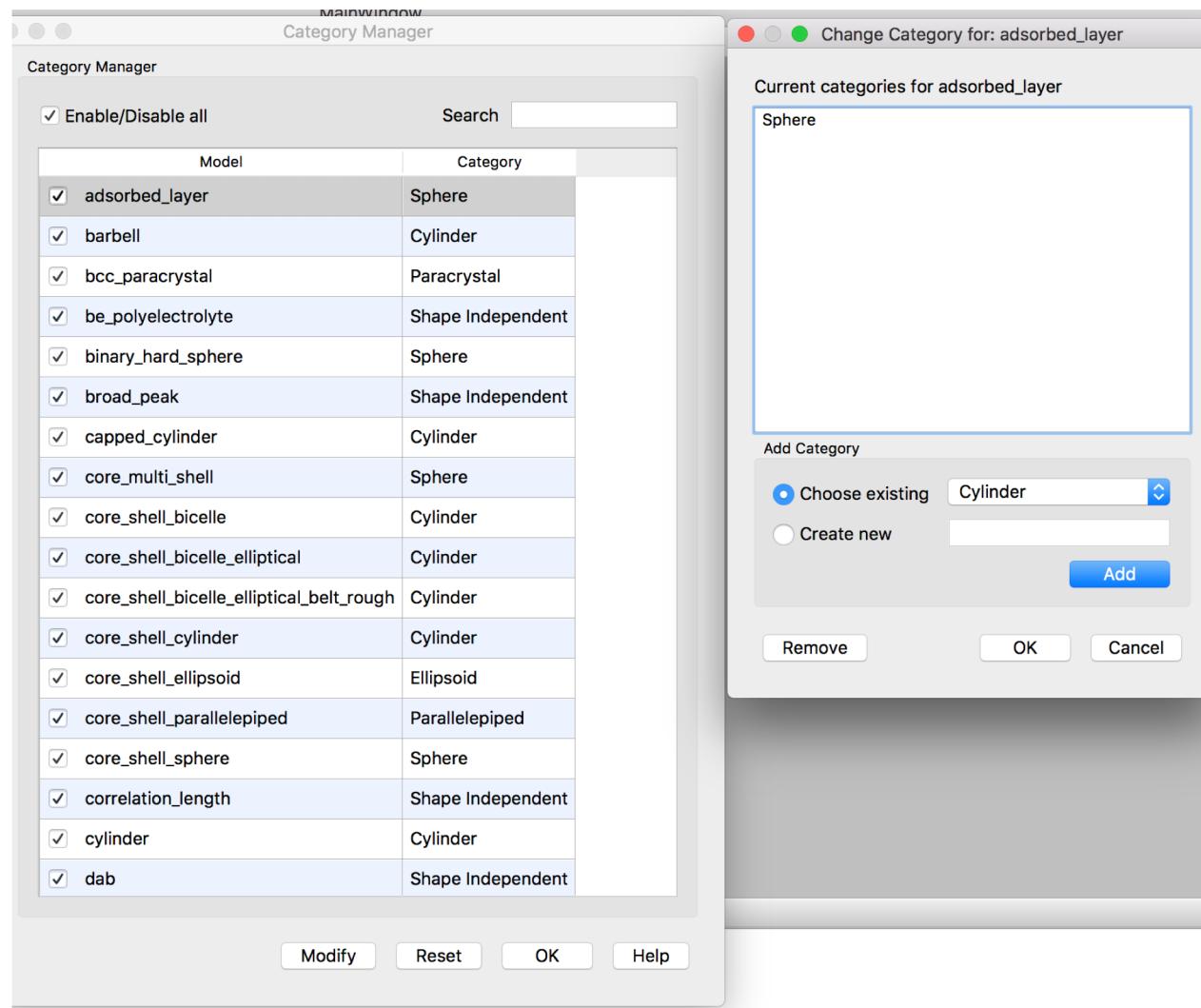
At the bottom are 'Help', 'Apply', and 'Cancel' buttons.

# Python and C model files

```
cylinder.py x
102
103 import ...
104
105 name = "cylinder"
106 title = "Right circular cylinder with uniform scattering length density."
107 description = """
108     f(q,alpha) = 2*(sld - sld_solvent)*V*sin(qLcos(alpha)/2)
109     /[qLcos(alpha)/2]*J1(qRsin(alpha))/[qRsin(alpha)]
110
111     P(q,alpha)= scale/V*f(q,alpha)^{(2)}+background
112     V: Volume of the cylinder
113     R: Radius of the cylinder
114     L: Length of the cylinder
115     J1: The bessel function
116     alpha: angle between the axis of the
117     cylinder and the q-vector for 1D
118     :the ouput is P(q)=scale/V*integral
119     from pi/2 to zero of...
120     f(q,alpha)^{(2)}*sin(alpha)*dalpha + background
121
122 """
123 category = "shape:cylinder"
124
125 # parameters = [{"name": "sld", "units": "1e-6/Ang^2", "default": 4, "lower": -inf, "upper": inf, "type": "double", "description": "Cylinder scattering length density"}, {"name": "sld_solvent", "units": "1e-6/Ang^2", "default": 1, "lower": -inf, "upper": inf, "type": "double", "description": "Solvent scattering length density"}, {"name": "radius", "units": "Ang", "default": 20, "lower": 0, "upper": inf, "type": "double", "description": "Cylinder radius"}, {"name": "length", "units": "Ang", "default": 400, "lower": 0, "upper": inf, "type": "double", "description": "Cylinder length"}, {"name": "theta", "units": "degrees", "default": 60, "lower": -360, "upper": 360, "type": "double", "description": "cylinder axis to beam angle"}, {"name": "phi", "units": "degrees", "default": 60, "lower": -360, "upper": 360, "type": "double", "description": "rotation about beam"}]
126
127 source = ["lib/polevl.c", "lib/sas_J1.c", "lib/gauss76.c", "cylinder.c"]
128
129 def ER(radius, length):
130     """
131         Return equivalent radius (ER)
132     """
133     ddd = 0.75 * radius * (2 * radius * length + (length + radius) * (length + pi * radius))
134
135
```

```
cylinder.c x
1 #define INVALID(v) (v.radius<0 || v.length<0)
2
3 static double
4 form_volume(double radius, double length)
5 {
6     return M_PI*radius*radius*length;
7 }
8
9 static double
10 fq(double qab, double qc, double radius, double length)
11 {
12     return sas_2J1x_x(qab*radius) * sas_sinx_x(qc*0.5*length);
13 }
14
15 static double
16 orient_avg_1D(double q, double radius, double length)
17 {
18     // translate a point in [-1,1] to a point in [0, pi/2]
19     const double zm = M_PI_4;
20     const double zb = M_PI_4;
21
22     double total = 0.0;
23     for (int i=0; i<GAUSS_N ;i++) {
24         const double theta = GAUSS_Z[i]*zm + zb;
25         double sin_theta, cos_theta; // slots to hold sincos function output
26         // theta (theta,phi) the projection of the cylinder on the detector plane
27         SINCOS(theta , sin_theta, cos_theta);
28         const double form = fq(q*sin_theta, q*cos_theta, radius, length);
29         total += GAUSS_W[i] * form * form * sin_theta;
30     }
31     // translate dx in [-1,1] to dx in [lower,upper]
32     return total*zm;
33 }
34
35 static double
36 Iq(double q,
37      double sld,
38      double solvent_sld,
39      double radius,
40      double length)
41 {
42     const double s = (sld - solvent_sld) * form_volume(radius, length);
43     return 1.0e-4 * s * s * s * orient_avg_1D(q, radius, length);
44 }
```

# Category manager (View->Category Manager)



# SasView marketplace



SasView Marketplace

Search



Log In

## 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 (compatibility with python 2.7.x is currently required) 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 available [here](#). It may also be helpful to examine the library models in the \sasmmodels-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.

*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.*

### Categories:

[Cylinder](#)

[Ellipsoid](#)

[Lamellae](#)

[Other](#)

[Paracrystal](#)

[Parallelepiped](#)

[Shape-Independent](#)

[Sphere](#)

[Structure Factor](#)

[All Models](#)