A simulation and fitting framework for GISAS and reflectometry

Jan Burle, Jonathan Fisher, Marina Ganeva, Gennady Pospelov, Walter Van Herck, Joachim Wuttke, Dmitry Yurov

Scientific Computing Group at MLZ
Jülich Center for Neutron Science

JCNS Workshop 2017, Tutzing, October 10-13, 2017
Outline

- Introduction
- Software architecture
- Milestones
- Recent functionality
- Demo
- Future plans
Grazing Incidence Small Angle Scattering

Experiment

- the beam is directed on a surface with a very small incident angle
- 2D detector records the intensity of scattered wave giving access to lateral and vertical sample structure information

Simulation

- Intensity is calculated from known sample structure using Distorted Wave Born Approximation

\[
\frac{d\sigma}{d\Omega} = \left| \langle F_{DWBA} \rangle \right|^2 S(q_{||})
\]
Welcome to BornAgain

BornAgain is a software package to simulate and fit small-angle scattering at grazing incidence. It supports analysis of both X-ray (GISAXS) and neutron (GISANS) data. Its name, BornAgain, indicates the central role of the distorted wave Born approximation in the physical description of the scattering process. The software provides a generic framework for modeling multilayer samples with smooth or rough interfaces and with various types of embedded nanoparticles.

Read more
Authors

Main developers
- Gennady Pospelov
- Walter Van Herck

Co-developers
- Jan Burle
- Jonathan Fisher
- Marina Ganeva
- Joachim Wuttke
- Céline Durniak
- Dmitry Yurov

Student interns
- Rebecca Brydon
- Sezer Karaca
- Abhishekh Khanna
- Mohammad Mahadi Hasan
- David Li
- Ivonna Li
Outline

- Introduction
- **Software architecture**
- Milestones
- Recent functionality
- Demo
- Future plans
Software Architecture

- Open-source, GPL3 license, 160k lines of code
- Multi-platform (Windows, Mac OS, Linux)
- C++ kernel for simulation and fitting, Python bindings, GUI

**Diagram:**

- **User**
- **script.py**
- **Python bindings**
  - **C++ kernel**
  - **External dependencies:** Eigen, fftw3, GSL
- **Standalone**
  - **GUI**
  - **External dependencies:** Qt5
Software Architecture

- Object oriented approach in sample description
Object oriented approach in sample description

```python
# defining materials
m_air = HomogeneousMaterial("Air", 0.0, 0.0)
m_substrate = HomogeneousMaterial("Substrate", 6e-6, 2e-8)
m_particle = HomogeneousMaterial("Particle", 6e-4, 2e-8)

# collection of particles

cylinder_ff = FormFactorCylinder(5*nanometer, 5*nanometer)
cylinder = Particle(m_particle, cylinder_ff)
prism_ff = FormFactorPrism3(10*nanometer, 5*nanometer)
prism = Particle(m_particle, prism_ff)
particle_layout = ParticleLayout()
particle_layout.addParticle(cylinder, 0.0, 0.5)
particle_layout.addParticle(prism, 0.0, 0.5)

# air layer with particles and substrate form multi layer
air_layer = Layer(m_air)
air_layer.addLayout(particle_layout)
substrate_layer = Layer(m_substrate)
multi_layer = MultiLayer()
multi_layer.addLayer(air_layer)
multi_layer.addLayer(substrate_layer)
```
Established functionality

- X-rays, non-polarized and polarized neutrons
- Arbitrary number of layers
- Rough interfaces
- Simple and composite particles
- Correlated positions
- Nanoparticle assemblies
- Off-specular geometry, beam divergence
Outline

- Introduction
- Software architecture
- **Milestones**
- Recent functionality
- Demo
- Future plans
<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Jan 2015</td>
<td>Graphical user interface, website</td>
</tr>
<tr>
<td>1.1</td>
<td>Apr 2015</td>
<td>New form factors, beam divergence in GUI, export GUI to Python</td>
</tr>
<tr>
<td>1.2</td>
<td>Jun 2015</td>
<td>Working on user manual, GUI real time</td>
</tr>
<tr>
<td>1.3</td>
<td>Jul 2015</td>
<td>New functional test machinery, new tutorials</td>
</tr>
<tr>
<td>1.4</td>
<td>Nov 2015</td>
<td>Rectangular detector, genetic fitting, fitting along slices, new tutorials</td>
</tr>
<tr>
<td>1.5</td>
<td>Feb 2016</td>
<td>C++11 migration, GUI mask editor, new tutorials</td>
</tr>
<tr>
<td>1.6</td>
<td>Jun 2016</td>
<td>Python 3, GUI fitting beta, Windows 32 -&gt; Windows 64</td>
</tr>
<tr>
<td>1.7</td>
<td>Nov 2016</td>
<td>BornAgain school and user meeting, specular peak, GitHub migration, new build server</td>
</tr>
<tr>
<td>1.8</td>
<td>Apr 2017</td>
<td>Graded interfaces, improved fitting support in GUI</td>
</tr>
<tr>
<td>1.9</td>
<td>Jul 2017</td>
<td>Magnetization formalism, GUI saving mechanism</td>
</tr>
<tr>
<td>1.10</td>
<td>Oct 2017</td>
<td>Mesocrystals in GUI, Plugin mechanism</td>
</tr>
</tbody>
</table>
GitHub workflow

GitHub cloud

BornAgain central repo

- Nightly builds
  - Functional tests
  - 2 MacOS
  - 2 Win
  - 6 Linux

- Pull Request

- GitHub builds
  - 1 MacOS
  - 1 Win
  - 1 Linux

Local computers

fork

- Pull

- Edits

clone

Edits

Code Review
Outline

- Introduction
- Software architecture
- Milestones
- Recent functionality
- Demo
- Future plans
Recent functionality

- Correlation between particles in different layers
Recent functionality

- Correlation between particles in different layers
- Particles crossing layer interfaces
Recent functionality

new in Release 1.8

- Correlation between particles in different layers
- Particles crossing layer interfaces
- Dense particles: average material for Fresnel calculations
Recent functionality

- Correlation between particles in different layers
- Particles crossing layer interfaces
- Dense particles: average material for Fresnel calculations
- Graded layer approximation
Recent functionality

new in Release 1.8

- Correlation between particles in different layers
- Particles crossing layer interfaces
- Dense particles: average material for Fresnel calculations
- Graded layer approximation
Recent functionality

- Uniform B-field versus uniform magnetization density

- Currently only available through Python API

- Validation with simple, well-known samples underway
Recent functionality

- Mesocrystals in GUI
  - Outer shape of the mesocrystal
  - Lattice vectors
  - Lattice basis (consisting of ‘regular’ particles and their positions)

\[
S_{total}(r) = S_{MC}(r) \cdot \sum_{r_i \in \Lambda} S_{NP}(r) \otimes \delta(r - r_i)
\]

\[
F_{total}(q) \propto F_{MC}(q) \otimes \left\{ F_{NP}(q) \cdot \sum_{q_i \in \Lambda^*} \delta(q - q_i) \right\}
\]
Outline

- Introduction
- Software architecture
- Milestones
- Recent functionality
- Demo
- Future plans
Outline

- Introduction
- Software architecture
- Milestones
- Recent functionality
- Demo
- Future plans
3D visualization in BornAgain

New 3D View in graphical user interface
- Improve visual feedback during sample construction
- Visualize abstract entities like interference functions

GUI construction

Python construction

Artistic rendering currently made with Blender
3D visualization in BornAgain
Expanding to reflectometry

BornAgain for reflectometry (SINE2020)

- BornAgain allows to access full R,T info
- Have simple specular peak depicted on top of 2D GISAS image
- Setup off-specular geometries
- Allows flexibly assemble models
- Infrastructure and user community

Planned

- Beam size effects
- Footprint correction
- Rocking curves, omega scans
- Material library, SLD profiles
- Roughness models
Expanding to reflectometry

BornAgain for reflectometry (SINE2020)
- BornAgain allows to access full R,T info
- Have simple specular peak depicted on top of 2D GISAS image
- Setup off-specular geometries
- Allows flexibly assemble models
- Infrastructure and user community

Planned
- Beam size effects
- Footprint correction
- Rocking curves, omega scans
- Material library, SLD profiles
- Roughness models

ESS in-kind contribution
3 years FTE obtained
- Establish BornAgain as a standard software for ESS reflectomeres
- Provide fitting of GISAS, specular, off-specular data in a single framework
Conclusion

BornAgain, release 1.10

- Fitting from graphical user interface
- Python 3 support
- GitHub based development
- Graded layer approximations, mesocrystals, magnetization
- User community

Next steps

- Magnetic nanoparticles in GUI
- Material library
- Reflectometry workflow
- 3D in GUI
Backup
GitHub workflow

github.com/scgmlz/BornAgain

GitHub cloud

BornAgain central repo

Pull Request

fork

Pull

Local computers

Edits

clone
GitHub workflow

GitHub cloud

BornAgain central repo

Pull Request

Pull

fork

GitHub builds
- 1 MacOS
- 1 Win
- 1 Linux

Local computers

clone

Edits

Code Review

github.com/scgmlz/BornAgain
GitHub workflow

**GitHub cloud**
- BornAgain central repo
- Pull Request

**Nightly builds**
- Functional tests
  - 2 MacOS
  - 2 Win
  - 6 Linux

**GitHub builds**
- 1 MacOS
- 1 Win
- 1 Linux

**Local computers**
- Clone
- Edits
- Code Review

**GitHub workflow**
github.com/scgmlz/BornAgain