GISAS analysis with BornAgain

Status and challenges

Jan Burle, Jonathan M. Fisher, Marina Ganeva, Gennady Pospelov, Walter Van Herck and Joachim Wuttke
Overview

- Introduction
- Grazing Incidence Small Angle Scattering
- BornAgain software
- GUI demo
- Challenges
Our group and BornAgain

- Heinz Maier-Leibnitz Zentrum (MLZ) in Garching (Munich)

- Scientific Computing Group: develop and maintain software for data reduction and analysis

- BornAgain: GISAS simulation and fitting software:
  - For both expert and novice users
  - Support for polarized neutrons
  - Extensible: reflectivity, off-specular scans, ...
GISAS setup
GISAS specifics

- In-plane and out-of-plane information
- Surface sensitivity (as in GID)
- Length scales of SAS
- Increased scattering volume
- Reflected and transmitted waves interfere
- Software needed to arrive at quantitative results (simulate & fit)
Waveguide effect
DWBA

- Ideally: solve the Lippmann-Schwinger equation
- Distorted Wave Born Approximation for multilayers
- Exact solution for effectively one-dimensional system
- Everything else as first order perturbation (nanoparticles, roughness)
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

- **Student interns**
  - Rebecca Brydon
  - Sezer Karaca
  - Abhishek Khanna
  - Mohammad Mahadi Hasan
  - David Li
  - Ivonna Li
Lines of code

![Graph showing the growth of Lines of Code from April 2012 to April 2017. The graph includes different categories such as PythonAPI, GUI, Unit Tests, Functional Tests, and Core. The y-axis is in thousands (10^3).]
# Release history

<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>
</tbody>
</table>
BornAgain usage

- Open-source
- Multi-platform
- C++/Python

User

script.py

Python bindings

C++ kernel

External dependencies: Eigen, fftw3, GSL

Standalone GUI

External dependencies: Qt5
BornAgain models
BornAgain 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
Recent functionality

- Correlation between particles in different layers
Recent functionality

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

- 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
User community

- External contributors
- Regularly making concrete suggestions
- Regularly finding and reporting bugs
- Writing us mails
- Regularly running BornAgain
- Generating user requests
- Appearing in statistics
- Visiting website
Citations

- 2017: conservatively extrapolated
Future plans

- Magnetic scattering
- Reflectivity: calculations + GUI integration
- GUI functionality:
  - real-space visualization
  - include polarized scattering
  - further improvements in UX
- New sample models
- Simulations beyond DWBA (e.g. for gratings)
- Documentation
Magnetic scattering

- Uniform B-field versus uniform magnetization density

- Validation with simple, well-known samples
- Magnetic roughness
- Magnetic domain structures
Real-space visualization

- Improve visual feedback during sample construction
- Visualize abstract entities like interference function

Artistic rendering

Representation in GUI and Python

```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 = FormFactorPrism(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)
```
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
  - SLD profiles (fitted across slices)
  - Material library
  - Roughness models
Challenges

- User eXperience for GUI and Python API
Challenges

- Experimental conditions:
  - Beam divergence
  - Finite sample
  - Diffuse specular peak
- More complex sample models (e.g. domain structures)
- Fitting:
  - Reduce time to find (global) minimum
  - Stability of fitting engine
  - Reliability of (global) minimum
- Validation and stability of simulations
  - Functional testing
  - Agreement with experimental data
  - Cross validation of software
- Beyond DWBA
Thank you!

www.bornagainproject.org
github.com/scgmlz/BornAgain
Development infrastructure

- Source control: github
- Code review: github
- Continuous integration: github, buildbot
- Management of release cycles: Redmine
- Issue tracking: Redmine
- Unit testing: googletest
- Functional tests: ad hoc
- Documentation: drupal website, theorymanual and API documentation
Basic theory

• Lippmann-Schwinger equation

\[ |\Psi^\pm\rangle = |\Phi\rangle + G_0^\pm V |\Psi^\pm\rangle \]

• Solution methods:
  • Direct numerical methods
  • In case of spherical symmetry of V: partial wave analysis
  • High energies or weak potential: Born series (or DWBA)
  • Method of continued fractions
  • Schwinger-Lanczos method
  • ...