BornAgain framework
EXPERIMENT PLANNING, SIMULATION AND FITTING OF GISAS DATA

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

GISAS 2018, September 4-7, 2018, Gyeongju HICO, Korea
Outline

- Introduction
- Software architecture
- Functionality overview
- What’s new
- Demo
- Project infrastructure
- Future plans
Scientific computing group at MLZ

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

FRM II
20 MW neutron source

More than 30 instruments
Including MARIA, REFSANS, NREX reflectometers

Scientific Computing Group, group leader Dr. J. Wuttke
- Develop and maintain software for data reduction and analysis

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

Project was initiated by Prof. Thomas Brückel
Grazing incidence small angle scattering

GISAS specifics
- Surface sensitive non-destructive technique
- Large area coverage, statistical information
- Reflected and transmitted waves interfere
- Tunable depth probe by changing incident angle

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

\[
\frac{d\sigma}{d\Omega} = \langle |F_{DWBA}|^2 \rangle S(q_z)
\]
Software requirements

Users → Easy-to-use → GUI
Software requirements

- **Users**
  - Easy-to-use
  - Availability
    - Open source
    - No proprietary software
    - Easy-to-install
    - Multi-platform
  - Windows
  - Mac
  - Linux
  - GUI

BornAgain framework

GISAS 2018, Gyeongju, Korea
Software requirements

- Users
  - Easy-to-use
  - Availability
    - Open source
    - No proprietary software
    - Multi platform
      - Windows
      - Mac
      - Linux
      - Clusters
  - Advanced Users
  - Complex models
  - Performance
    - C++
    - OO, decoupling
  - Extensibility
  - Scripting
    - Python

GISAS 2018, Gyeongju, Korea

BornAgain framework
Introduction

Software architecture

Functionality overview

What’s new

Demo

Project infrastructure

Future plans
Software architecture

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

Diagram:

- User
- script.py
- Python bindings
- C++ kernel
- External dependencies: Eigen, fftw3, GSL
- Standalone
- GUI
- External dependencies: Qt5
Object oriented approach

- Sample, beam and detector are defined via building blocks – classes
- Blocks are combined by the user into a hierarchical tree representing a simulation
Sample construction in GUI

BornAgain framework

GISAS 2018, Gyeongju, Korea
import bornagain as ba

def get_sample():
    # defining materials
    air = ba.HomogeneousMaterial("Air", 0.0, 0.0)
    substrate = ba.HomogeneousMaterial("Substrate", 6e-6, 2e-8)
    gold = ba.HomogeneousMaterial("Gold", 6e-4, 2e-8)

    # creating particles
    cylinder_ff = ba.FormFactorCylinder(5*nm, 5*nm)
    cylinder = ba.Particle(gold, cylinder_ff)

    layout = ba.ParticleLayout()
    layout.addParticle(cylinder, 1.0)

    air_layer = ba.Layer(air)
    air_layer.addLayout(layout)
    substrate_layer = ba.Layer(substrate)

    multi_layer = ba.MultiLayer()
    multi_layer.addLayer(air_layer)
    multi_layer.addLayer(substrate_layer)

    return multi_layer
Introduction
Software architecture
Functionality overview
What’s new
Demo
Project infrastructure
Future plans
Functionality overview

- 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

BornAgain framework
Available shapes

- Anisotropic pyramid
- Box
- Cone
- Cone6
- Cuboctahedron
- Cylinder
- Dodecahedron
- Ellipsoidal cylinder
- Full sphere
- Full spheroid
- Hemi-ellipsoid
- Icosahedron
- Prism3
- Prism6
- Pyramid
- Ripple1
- Ripple2
- Tetrahedron
- Truncated cube
- Truncated sphere
- Truncated spheroid

Every shape can be rotated in all 3 directions
- $F(q)$ validated against complex $q$
Complex shapes

Core shell particles

Particles with size distribution
With possibility to link parameters

Particle compositions
collection of particles with fixed inter-particle distance
coherent interference
Hexagonally ordered CoFeO$_4$ nanoparticles

Experiment: A. Qdemat, E. Kentzinger et. al., GALAXI

Model:
- nanoparticles in the hexagonal lattice
- local monodisperse approximation
## 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 bornagainproject.org</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, 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>New magnetization formalism, GUI saving mechanism</td>
</tr>
<tr>
<td>1.10</td>
<td>Oct 2017</td>
<td>Mesocrystals in GUI, Plugin mechanism</td>
</tr>
<tr>
<td>1.11</td>
<td>Mar 2018</td>
<td>Off-specular simulation in GUI, specular in Core, SLD materials</td>
</tr>
<tr>
<td>1.12</td>
<td>May 2018</td>
<td>Specular simulaiton in GUI, depth probe simulation, new web site</td>
</tr>
</tbody>
</table>
## 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 bornagainproject.org</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, 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>New magnetization formalism, GUI saving mechanism</td>
</tr>
<tr>
<td>1.10</td>
<td>Oct 2017</td>
<td>Mesocrystals in GUI, Plugin mechanism</td>
</tr>
<tr>
<td>1.11</td>
<td>Mar 2018</td>
<td>Off-specular simulation in GUI, specular in Core, SLD materials</td>
</tr>
<tr>
<td>1.12</td>
<td>May 2018</td>
<td>Specular simulation in GUI, depth probe simulation, new web site</td>
</tr>
</tbody>
</table>
Graded layer approximation

NEW IN RELEASE 1.8, APR 17

- Correlation between particles in different layers
Graded layer approximation

NEW IN RELEASE 1.8, APR 17

- Correlation between particles in different layers
- Particles crossing layer interfaces
Graded layer approximation

NEW IN RELEASE 1.8, APR 17

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

NEW IN RELEASE 1.8, APR 17

- Correlation between particles in different layers
- Particles crossing layer interfaces
- Dense particles: average material for Fresnel calculations
- Graded layer approximation
Graded layer approximation
NEW IN RELEASE 1.8, APR 17

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

NEW IN RELEASE 1.9, JUL 17

Model includes

- Materials with uniform magnetization density
- Can be assigned to any particle shape
- Neutron beam polarization vector
- Neutron analyzer direction, efficiency and transmission

![Polarization analysis](image)

- **Polarization (Bloch vector)**
  - X: 0.000
  - Y: 0.000
  - Z: 0.000

- **Analyzer orientation**
  - X: 0.000
  - Y: 0.000
  - Z: 0.000

- **Analyzer properties**
  - Efficiency: 0.500
  - Transmission: 1.000

BornAgain framework

GISAS 2018, Gyeongju, Korea
Magnetic nanoparticles
NEW IN RELEASE 1.9, JUL 17

Model includes
- Materials with uniform magnetization density
- Can be assigned to any particle shape
- Neutron beam polarization vector
- Neutron analyzer direction, efficiency and transmission
Mesocrystals in GUI

NEW IN RELEASE 1.10, OCT 17

Model includes

- Outer shape of the mesocrystal
- Lattice vectors
- Lattice basis consisting of regular particles at their positions
New material type
NEW IN RELEASE 1.10, Mar 18

Material types in BornAgain
- Homogeneous material based on refractive index
- Homogeneous material based on scattering length density

![Material Editor](image)
Real-space visualization in GUI

NEW IN RELEASE 1.12, MAY 18
Finite 2d lattices
NEW IN RELEASE 1.12, MAY 18

Model includes
- Interference function of finite 2D lattice
- Possibility to compose finite lattice into a superlattice
- For the moment available via Python API only
BornAgain for reflectometry

SINE2020 initiative
- Provide fitting of GISAS, specular, off-specular data in a single framework

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

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

- Instrument effects: wavelength and angular beam divergence
- Footprint corrections
- Full accessibility of all simulation features through GUI
- Possibility to fit reflectometry data via Python API

Reflectivity with and without beam divergence

Reflectivity with and without footprint correction
Depth probe simulation
NEW IN RELEASE 1.12, MAY 18

Model includes
- Evanescent wave intensity in the bulk of the sample
- Instrument resolution effects available
- For the moment available via Python API only

\[ l_{ew}(z) = |\psi(z)|^2 = \left| R \cdot e^{ikz} + T \cdot e^{-ikz} \right|^2 \]
Depth probe simulation
NEW IN RELEASE 1.12, MAY 18

**Multilayer Ti/Pt resonator**

H. Frielinghaus, et. al., NIM A 871 (2017) 72–76

---

![Diagram of a multilayer Ti/Pt resonator](image)
DEMO
- Introduction
- Software architecture
- Functionality overview
- What’s new
- Demo
- Project infrastructure
- Future plans
Development infrastructure

- Development organization
  - Source control: github
  - Code review: github
  - Issue tracking: redmine
  - Management of release cycles: redmine

- Code stability
  - Continuous integration: github, travis, appveyor, buildbot
  - Unit testing: google test
  - Functional tests: ad hoc

- Documentation
  - Website: github, hugo
  - Theory manual: pdf, html
  - API documentation: doxygen
BornAgain developers

Scientific Computing Group of MLZ, group leader Dr. J. Wuttke

Main developers
- Gennady Pospelov
- Walter Van Herck

Co-developers
- Jan Burle
- Jonathan Fisher
- Marina Ganeva
- Joachim Wutke
- Dmitry Yurov
- Celine Durniak
- Huan Manuel Carmona Loaiza

Student interns
- Rebecca Brydon
- Sezer Karaca
- Abhishek Khanna
- Mohammad Mahadi Hasan
- David Li
- Ivonna Li
- Anik Halder
GitHub workflow

GitHub cloud

BornAgain central repo

fork

Local computers

clone

Pull

Edits

github.com/scgmlz/BornAgain
GitHub workflow

GitHub cloud

BornAgain central repo

Pull Request

Fork

GitHub builds
- 1 MacOS
- 1 Win
- 1 Linux

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

Local computers

Pull

Edits

Clone

Code Review

github.com/scgmlz/BornAgain
Future plans

- Specular reflectivity
  - SLD profiles
  - Material library
  - Fitting workflow in GUI

- GUI functionality
  - Undo/redo
  - Project files back compatibility
  - Plugin mechanism

- Model for magnetic roughness/domains

- New fitting API

- User requests
BornAgain

Open-source software package to simulate and fit neutron and x-ray small-angle scattering at grazing incidence.

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.

Currently v1.12.0
We are organizing the 2nd BornAgain School and User Meeting in December 2018, in Munich. Subscribe to our mailing list to get notified.
Thank you!
BACKUP
BornAgain for reflectometry
NEW IN RELEASE 1.12, MAY 18

Fitting experimental data
- For the moment, available through PythonAPI only

Ag nanoparticles on silicon substrate with SiO2 coating

Microgel particles on silicon substrate in D2O environment
```python
def get_sample():
    # Defining Materials
    material_1 = ba.HomogeneousMaterial("example03_Air", 0.0, 0.0)
    material_2 = ba.HomogeneousMaterial("example03_Particle", 0.0006, 2e-08)
    material_3 = ba.HomogeneousMaterial("example03_Substrate", 6e-06, 2e-08)

    # Defining Layers
    layer_1 = ba.Layer(material_1)
    layer_2 = ba.Layer(material_3)

    # Defining Form Factors
    formFactor_1 = ba.FormFactorCylinder(5.0*nm, 5.0*nm)

    # Defining Particles
    particle_1 = ba.Particle(material_2, formFactor_1)

    # Defining Interference Functions
    interference_1 = ba.InterferenceFunction2DParaCrystal(20.0*nm, 20.0*nm, 120.0*deg, 0.0*deg, 0.0*nm)
    interference_1.setDomainSizes(20000.0*nm, 20000.0*nm)
```