BornAgain

Software for simulating and fitting
X-ray and neutron small-angle scattering
at grazing incidence

User Manual
version 0.1.1
September 27, 2013

C. Durniak, G. Pospelov, W. Van Herck, J. Wuttke

Scientific Computing Group
Jülich Centre for Neutron Science
outstation at Heinz Maier-Leibnitz Zentrum Garching
Forschungszentrum Jülich GmbH

Disclaimer

This manual is under development and does not yet constitute a comprehensive listing of `BornAgain` features and functionality. The included information and instructions are subject to substantial change and are provided only as a preview.

Contents

Introduction

BornAgain is a free software package to simulate and fit small-angle scattering at grazing incidence (GISAS). 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 (DWBA) 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.

BornAgain almost completely reproduces the functionality of the widely used program IsGISAXS by R. Lazzari [1].

However, BornAgain also extends this functionality by supporting an unrestricted number of layers and particles, diffuse reflection from rough layer interfaces, particles with inner structures and support for polarized neutrons and magnetic scattering. Adhering to a strict object-oriented design, BornAgain provides a solid base for future extensions in response to specific user needs.

BornAgain is platform-independent software, with active support for Linux, MacOS and Microsoft Windows. It is a free and open source software provided under terms of GNU General Public License (GPL). This documentation is released under the Creative Commons license CC-BY-SA.

The authors will be grateful for all kind of feedback: criticism, praise, bug reports, feature requests or contributed modules. When BornAgain is used in preparing scientific papers, please cite this manual as follows:

> C. Durniak, G. Pospelov, W. Van Herck, J. Wuttke (2013),
> BornAgain - Software for simulating and fitting X-ray and neutron small-angle scattering at grazing incidence, version ⟨...⟩,
> `http://apps.jcns.fz-juelich.de/BornAgain`

This user guide starts with a brief description of the steps necessary for installing the software and running a simulation on Unix and Windows platforms in Section 1. A more detailed description of the installation procedure is given in Section 2. The general methodology of a simulation with BornAgain and detailed simulation usage examples are given in Section 3. The fitting toolkit that is provided by the framework, is presented in Section **??**, while Section 4 provides a brief overview of the software architecture.

Icons used in this manual:

✎: this sign highlights further remarks.

⚠: this sign highlights essential points.

Chapter 1

Quick start

1.1 Quick start on Unix Platforms

This section shortly describes how to build and install BornAgain from source and run the first simulation on Unix Platforms. More details about installation procedure are given in Section 2.

Step I: installing third party software

- compilers: clang versions ≥ 3.1 or GCC versions ≥ 4.2
- cmake (≥ 2.8)
- boost library (≥ 1.48)
- GNU scientific library (≥ 1.15)
- fftw3 library (≥ 3.3.1)
- python-2.7, python-devel, python-numpy-devel

Step II: getting the source
Download BornAgain source tarball from http://apps.jcns.fz-juelich.de/BornAgain or use git repository

```
git clone git://apps.jcns.fz-juelich.de/BornAgain.git
```

Step III: building the source

```
mkdir <build_dir>; cd <build_dir>;
cmake <source_dir> -DCMAKE_INSTALL_PREFIX=<install_dir>
make
make check
make install
```

Step IV: running example

```
cd <install_dir>/Examples/python/ex001_CylindersAndPrisms
python CylindersAndPrisms.py
```

1.2 Quick start on Windows Platforms

Step I: installing third party software
Current version of BornAgain requires Python, numpy, matplotlib to be installed on the system. If you don't have them already, you can use PythonXY installer at https://code.google.com/p/pythonxy with default installation options to install all necessary packages (and more). together.

Step II: using installation package
The Windows installation package can be downloaded from http://apps.jcns.fz-juelich.de/BornAgain. Double click it to start installation process, then follow instructions.

Step IV: running example
Run an example simulation by double-clicking on the python script located in the BornAgain installation directory:

```
python C:/BornAgain-0.9.1/Examples/python/
    ex001_CylindersAndPrisms/CylindersAndPrisms.py
```

1.3 Getting help

Users of the software who encounter a problem in the installation of the framework or in running a simulation can use a web based issue tracking system at http://apps.jcns.fz-juelich.de/redmine/projects/bornagain/issues to provide a bug report. The same system can be used for requests for new features. The system is open for all users in read mode, while submitting of bug reports and feature requests are possible only after a simple registration procedure.

Chapter 2

Installation

`BornAgain` is intended to work on x86/x86_64 Linux, Mac OS X and Windows operating systems. It was successfully compiled and tested on

- Microsoft Windows 7 64-bit, Windows 8 64-bit
- Mac OS X 10.8 (Mountain Lion)
- OpenSuse 12.3 64-bit
- Ubuntu 12.10, 13.04 64-bit
- Debian 7.1.0, 32-bit, 64-bit

At the moment we support build and installation from source on Unix Platforms (Linux, Mac OS) and installation using binary installer package on MS Windows 7,8 (see Section 2.1 and Section 2.2). In the next releases we are planning to provide binary installers for Mac OS X and Debian.

We welcome user feedback and/or bug reports related to they installation experience via `http://apps.jcns.fz-juelich.de/redmine/projects/bornagain/issues`

2.1 Building and installing on Unix Platforms.

`BornAgain` uses `CMake` to configure a build system for compiling and installing the framework. There are three major steps to building `BornAgain` :

1. Acquire required third-party libraries.
2. Get `BornAgain` source code.
3. Use `CMake` to build and install software.

The remainder of this section explains each step in detail.

2.1.1 Third-party software.

To successfully build `BornAgain` a number of prerequisite packages must be installed.

- compilers: clang versions ≥ 3.1 or GCC versions ≥ 4.2
- cmake (≥ 2.8)
- boost library (≥ 1.48)
- GNU scientific library (≥ 1.15)
- fftw3 library (≥ 3.3)
- python (≥ 2.7, < 3.0), python-devel, python-numpy-devel

Other packages are optional

- ROOT framework (adds several additional fitting algorithms to `BornAgain`)
- python-matplotlib (allows to run usage examples with graphics)

All required packages can be easily installed on most Linux distributions using the system's package manager. Below we give a few examples for several selected operation systems. Please note, that other distributions (Fedora, Mint, etc) may have different commands for invoking the package manager and slightly different names of packages (like "boost" instead of "libboost" etc). Besides that, the installation should be very similar.

Ubuntu (12.10, 13.04), Debian (7.1)
Installing required packages

```
sudo apt-get install git cmake libgsl0-dev libboost-all-dev
   libfftw3-dev python-dev python-numpy
```

Installing optional packages

```
sudo apt-get install libroot-* root-plugin-* root-system-* ttf-
   root-installer libeigen3-dev python-matplotlib python-
   matplotlib-tk
```

OpenSuse 12.3
Adding "scientific" repository

```
sudo zypper ar http://download.opensuse.org/repositories/science/
   openSUSE_12.3 science
```

Installing required packages

```
sudo zypper install git-core cmake gsl-devel boost-devel fftw3-
   devel python-devel python-numpy-devel
```

Installing optional packages

```
sudo zypper install libroot-* root-plugin-* root-system-* root-
   ttf libeigen3-devel python-matplotlib
```

Mac OS X 10.8

To simplify the installation of third party open-source software on a Mac OS X system we recommend the use of `MacPorts` package manager. The easiest way to install MacPorts is by downloading the `dmg` from `www.macports.org/install.php` and running the system's installer. After the installation new command "`port`" will be available in terminal window of your Mac.
Installing required packages

```
sudo port -v selfupdate
sudo port install git-core cmake
sudo port install fftw-3 gsl
sudo port install boost -no_single-no_static+python27
```

Installing optional packages

```
sudo port install py27-matplotlib py27-numpy py27-scipy
sudo port install root +fftw3+python27
sudo port install eigen3
```

2.1.2 Getting source code

`BornAgain` source can be downloaded at `http://apps.jcns.fz-juelich.de/BornAgain` and unpacked with

```
tar xfz bornagain-<version>.tgz
```

Alternatively one can obtain `BornAgain` source from our public Git repository.

```
git clone git://apps.jcns.fz-juelich.de/BornAgain.git
```

More about Git

Our Git repository holds two main branches called "master" and "develop". We consider "master" branch to be the main branch where the source code of HEAD always reflects latest stable release. `git clone` command shown above

1. gives you a source code snapshot corresponding to the latest stable release,
2. automatically sets up your local master branch to track our remote master branch, so you will be able to fetch changes from the remote branch at any time using "git pull" command.

Master branch is updating approximately once per month. The second branch, "develop" branch, is a snapshot of the current development. This is where any automatic nightly builds are built from. The develop branch is always expected to work, so to get the most recent features one can switch source code to it by

```
cd BornAgain
git checkout develop
git pull
```

2.1.3 Building and installing the code

BornAgain should be build using CMake cross platform build system. Having third-party libraries installed on the system and BornAgain source code acquired as was explained in previous sections, type build commands

```
mkdir <build_dir>
cd <build_dir>
cmake <source_dir> -DCMAKE_INSTALL_PREFIX=<install_dir>
make
```

Here <source_dir> is the name of directory, where BornAgain source code has been copied, <install_dir> is the directory, where user wants the package to be installed, and <build_dir> is the directory where building will occur.

✎ About CMake
Having dedicated directory <build_dir> for build process is recommended by CMake. That allows several builds with different compilers/options from the same source and keeps source directory clean from build remnants.

Compilation process invoked by the command "make" lasts about 10 min for an average laptop of 2012 edition. On multi-core machines the compilation time can be decreased by invoking command "make" with the parameter "make -j[N]", where N is the number of cores.

Running functional tests is an optional but recommended step. Command "make check" will compile several additional tests and run them one by one. Every test contains the simulation of a typical GISAS geometry and the comparison on numerical level of simulation results with reference files. Having 100% tests passed ensures that your local installation is correct.

```
make check
...
100% tests passed, 0 tests failed out of 26
Total Test time (real) = 89.19 sec
[100%] Build target check
```

The last command "make install" copies compiled libraries and some usage examples into the installation directory.

```
make install
```

Troubleshooting

In the case of complex system setup, with variety of libraries of different versions scattered across multiple places (/opt/local, /usr etc.), you may want to help CMake to find libraries in proper place. In example below two system variables are defined to force CMake to prefer libraries found in /opt/local to other places.

```
export CMAKE_LIBRARY_PATH=/opt/local/lib:$CMAKE_LIBRARY_PATH
export CMAKE_INCLUDE_PATH=/opt/local/include:$CMAKE_INCLUDE_PATH
```

2.1.4 Running first simulation

In your installation directory you will find

```
./include - header files for compilation of your C++ program
./lib - libraries to import into python or link with your C++
    program
./Examples - directory with examples
```

Run your first example and enjoy first BornAgain simulation plot.

```
cd <install_dir>/Examples/python/ex001_CylindersAndPrisms
python CylindersAndPrisms.py
```

2.2 Installing on Windows Platforms.

Step I: installing third party software
The current version of BornAgain requires Python, numpy, matplotlib to be installed on the system. If you don't have them already installed, you can use PythonXY installer at https://code.google.com/p/pythonxy which, with default installation options, will contain at least these three packages. The user has to download and install this package before proceeding with BornAgain installation.

Step II: using installation package
The Windows installation package can be downloaded from http://apps.jcns.fz-juelich.de/BornAgain. Double click it to start the installation process, then follow the instructions.

Step IV: running example
Run an example by double-clicking on the python script located in the BornAgain installation directory:

```
python C:/BornAgain-0.9.1/Examples/python/
    ex001_CylindersAndPrisms/CylindersAndPrisms.py
```

Chapter 3

Simulation examples

3.1 General methodology

A simulation of GISAXS using `BornAgain` consists of following steps:

- define materials by specifying name and refractive index,
- define embedded particles by specifying shape, size, constituting material, interference function,
- define layers by specifiying thickness, roughness, material
- include particles in layers, specifying density, position, orientation,
- assemble a multilayered sample,
- specify input beam and detector characteristics,
- run the simulation,
- save the simulated detector image.

User defines all these steps using `BornAgain` API in Python script and then run the simulation by executing the script in Python interpreter. More information about general software architecture and `BornAgain` internal design are given in Section 4.

3.2 Conventions

3.2.1 Geometry of the sample

The geometry used to describe the sample is shown in figure 3.1. The z-axis is perpendicular to the sample's surface and pointing upwards. The x-axis is perpendicular to the plane of the detector and the y-axis is along it. The input and the scattered output beams are each characterized by two angles α_i, ϕ_i and α_f, ϕ_f respectively. Our choice of orientation