BornAgain Python API

Gennady Pospelov
Jülich Centre for Neutron Science at MLZ

BornAgain school and user meeting
Garching, December 2018

day_2

Outline

~/BornAgain-tutorial/talks/day_2/python_api_G

|		
Introduction to Python API	talking	pyapi01_introduction.pdf
Minimal simulation example	demo	pyapi02_minimal_example.py pyapi02_jupyter_example.ipynb
Simulating lamellar structure	demo	pyapi03_lamellar.py
Modifying lamellar example	task	pyapi04_lamellar_vertical_solution1.py pyapi04_lamellar_vertical_solution2.py
Introduction to fitting	talking	pyapi05_fitting.pdf
Fitting from Python outside BornAgain context	demo	pyapi06_rosenbrock_minimum.ipynb pyapi06_lmfit_decaying_wave.ipynb
Exact Python equivalent of GUI fit from day_1	demo	pyapi07_hexspheres_fit.py

BornAgain Python API

Application Programming Interface (API) can be used to build and run simulations.

Advantages

- Complex sample construction
- Access to features non existing in GUI
- Possibility to reuse code in other projects
- Less back-compatibility problems
- Git based workflow

Minimal simulation example
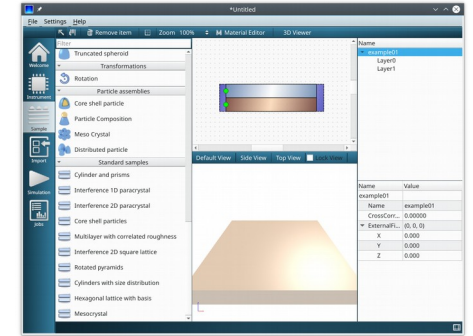
GUI equivalent

```python
from bornagain import *

# --- sample ---

air = Layer(HomogeneousMaterial("Air", 0.0, 0.0))
substrate = Layer(HomogeneousMaterial("Substrate", 6e-6, 2e-8))

multi_layer = MultiLayer()
multi_layer.addLayer(air)
multi_layer.addLayer(substrate)

# --- simulation ---

simulation = GISASSimulation()
simulation.setDetectorParameters(100, -2.0*deg, 2.0*deg, 100, 0.0*deg, 2.0*deg)
simulation.setBeamParameters(1.0*angstrom, 0.2*deg, 0.0*deg)
simulation.setSample(multi_layer)

# --- run and plot ---

simulation.runSimulation()
plot_simulation_result(simulation.result())
```
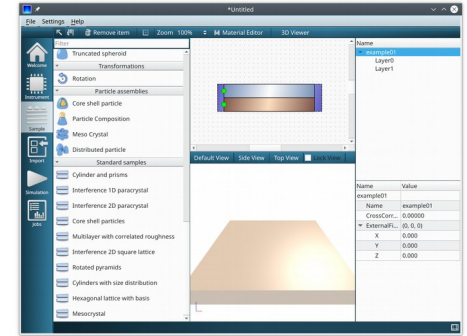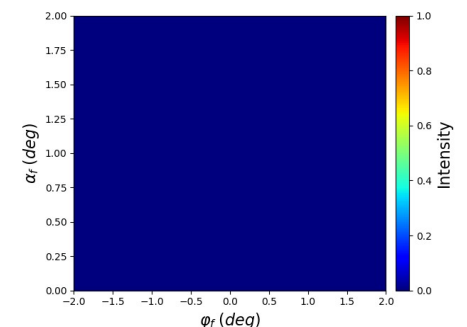
Minimal simulation example

GUI equivalent

```python
from bornagain import *

# --- sample ---

air = Layer(HomogeneousMaterial("Air", 0.0, 0.0))
substrate = Layer(HomogeneousMaterial("Substrate", 6e-6, 2e-8))

multi_layer = MultiLayer()
multi_layer.addLayer(air)
multi_layer.addLayer(substrate)

# --- simulation ---

simulation = GISASSimulation()
simulation.setDetectorParameters(100, -2.0*deg, 2.0*deg, 100, 0.0*deg, 2.0*deg)
simulation.setBeamParameters(1.0*angstrom, 0.2*deg, 0.0*deg)
simulation.setSample(multi_layer)

# --- run and plot ---

simulation.runSimulation()
plot_simulation_result(simulation.result())
```



```
$ python pyapi02_minimal_example.py
```

Demo

Running trivial simulations

```
pyapi02_minimal_example.py
pyapi02_jupyter_example.py
```

Structuring the code

```python
import bornagain as ba


def get_sample():
    multi_layer = ba.MultiLayer()
    ...
    return multi_layer


def get_simulation():
    simulation = ba.GISASSimulation()
    ...
    sample = get_sample()
    simulation.setSample(sample)
    return simulation


def run_simulation():
    simulation = get_simulation()
    simulation.runSimulation()
    return simulation.result()


if __name__ == '__main__':
    result = run_simulation()
    ba.plot_simulation_result(result)
```
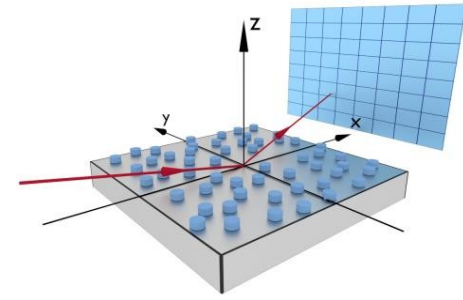
Example on sample construction


```python
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
```
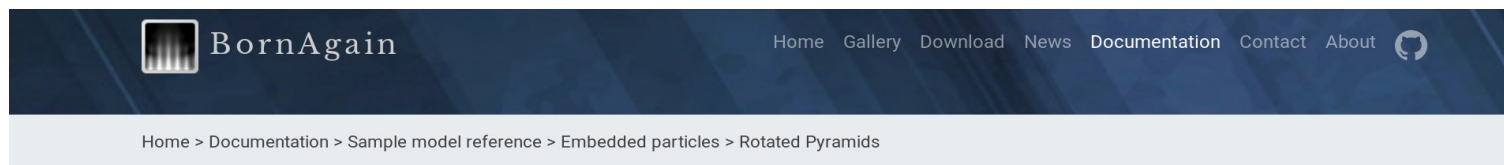
Material definition

Particle collection

MultiLayer construction

Example on sample construction
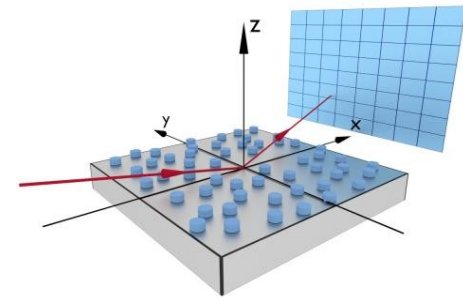

```python
import bornagain as ba


def get_materials():
    # 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)
    return air, substrate, gold


def get_sample():
    air, substrate, gold = get_materials()

    # 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
```

Material definition

Particle collection

MultiLayer construction

Where to start

Python viewer embedded in GUI

Where to start

Sample model reference on website

Where to code

○ Editor of your choice, Python interpreter

○ Integrated development environment (IDE)

• MS Visual Studio Code, PyCharm

• Syntax and error highlighting, debug, code navigation

○ Jupyter notebooks

• Quick prototyping
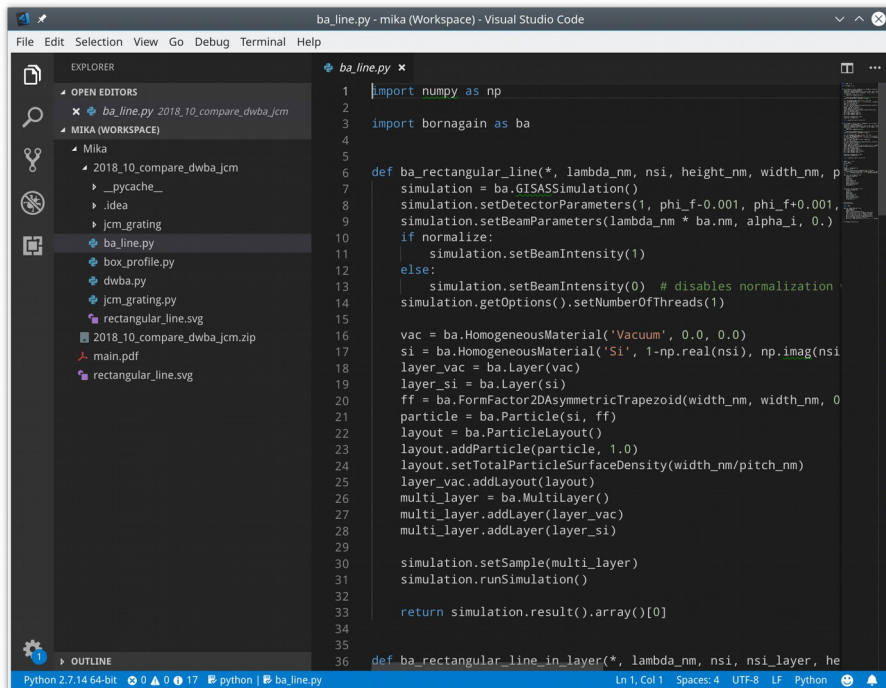
VS Code
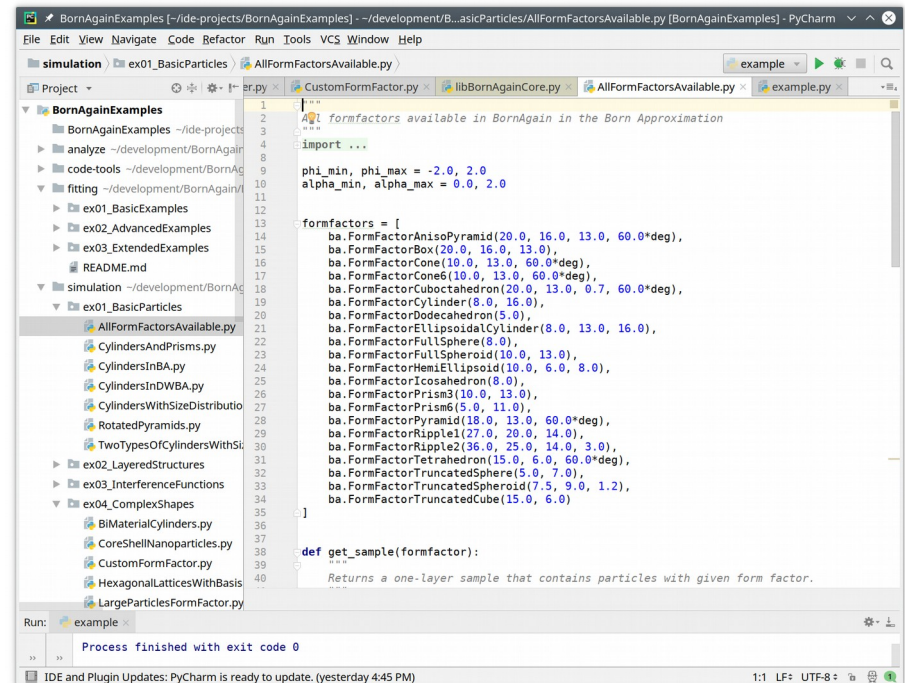
PyCharm

Python API technicalities

BornAgain Python API is automatically generated from C++ code
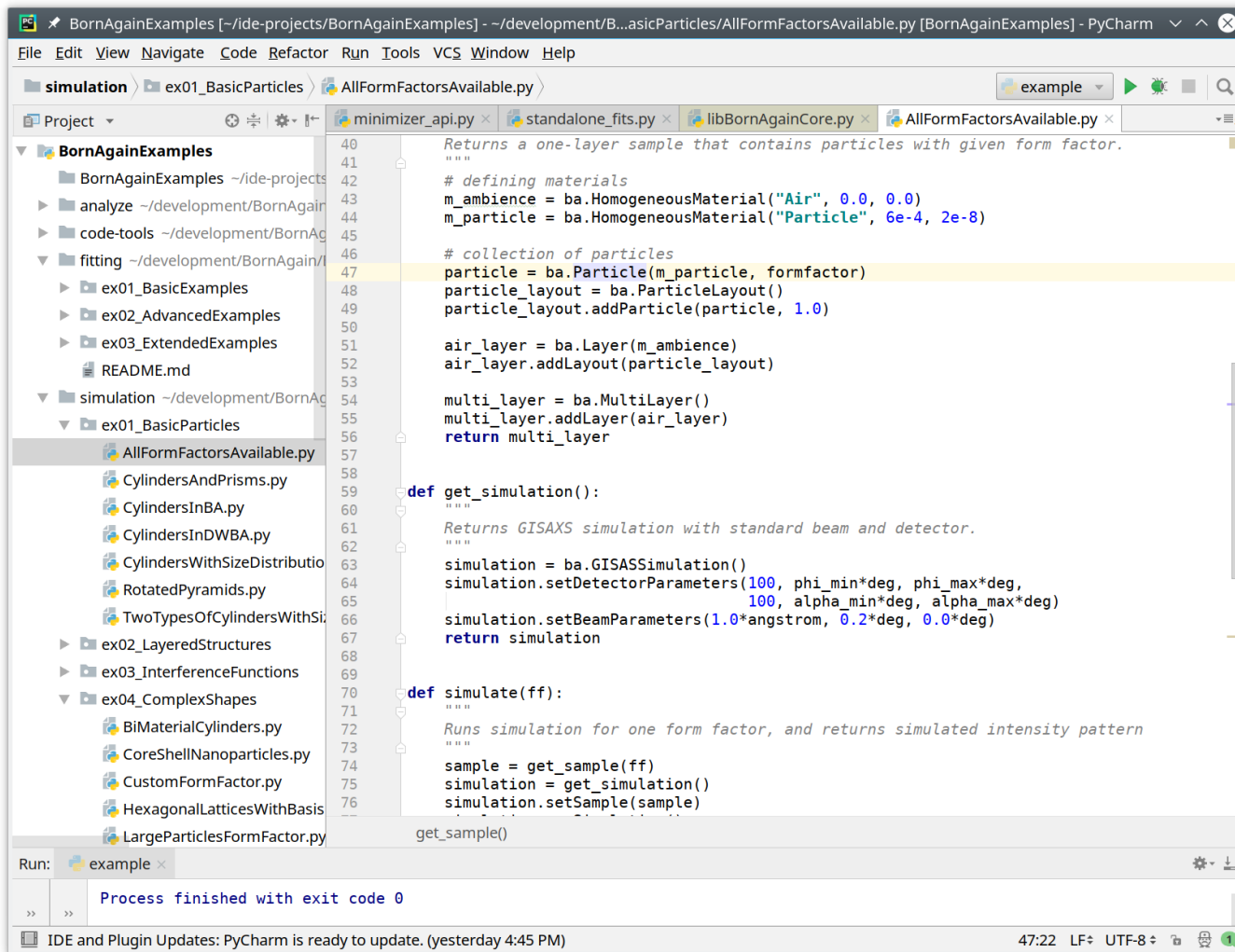
Python API technicalities

BornAgain Python API is automatically generated from C++ code
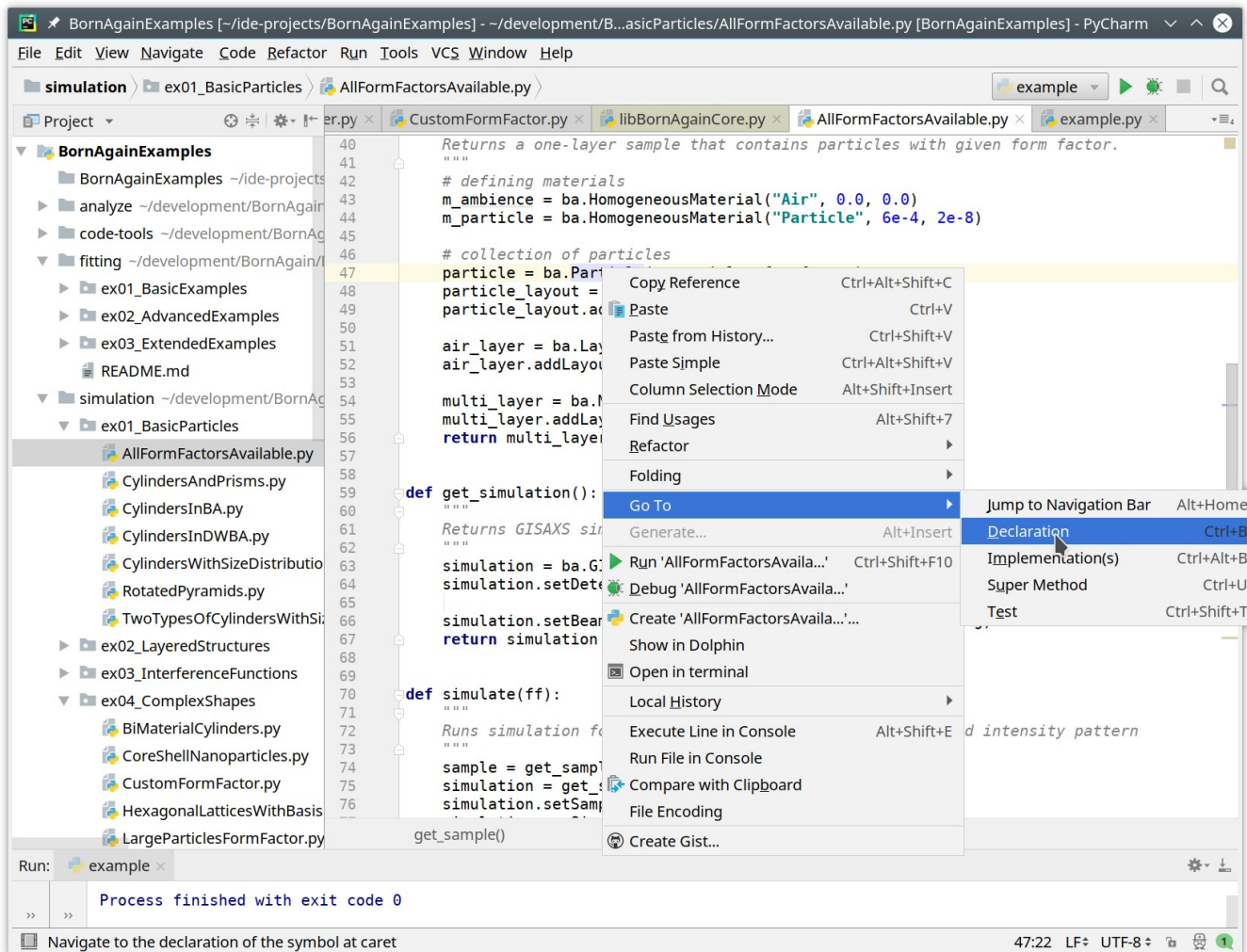
Python API technicalities

BornAgain Python API is automatically generated from C++ code

Python API internals looks unhuman

```
26223    class Particle(IParticle):
26224        """
26225
26226
26227        A particle with a form factor and refractive index.
26228
26229        C++ includes: Particle.h
26230
26231        """
26232
26233        __swig_setmethods__ = {}
26234        for _s in [IParticle]:
26235            __swig_setmethods__.update(getattr(_s, '__swig_setmethods__', {}))
26236        __setattr__ = lambda self, name, value: _swig_setattr(self, Particle, name, value)
26237        __swig_getmethods__ = {}
26238        for _s in [IParticle]:
26239            __swig_getmethods__.update(getattr(_s, '__swig_getmethods__', {}))
26240        __getattr__ = lambda self, name: _swig_getattr(self, Particle, name)
26241        __repr__ = _swig_repr
26242
26243        def __init__(self, *args):
26244            """
26245            __init__(Particle self) -> Particle
26246            __init__(Particle self, Material material) -> Particle
26247            __init__(Particle self, Material material, IFormFactor form_factor) -> Particle
26248            __init__(Particle self, Material material, IFormFactor form_factor, IRotation rota
26249
26250            Particle::Particle(Material material, const IFormFactor &form_factor, const IRotat
26251
26252            """
26253            this = _libBornAgainCore.new_Particle(*args)
26254            try:
26255                self.this.append(this)
26256            except __builtin__.Exception:
26257                self.this = this
26258
```

Documentation on Python API

C++ API documentation sometimes might help …

BornAgain C++ API

Simulate and fit neutron and x-ray scattering at grazing incidence

Class List | Class Index | Class Hierarchy

- ▼ BornAgain
  - User API
  - ▼ Full C++ API
    - ▶ Class List
    - Class Index
    - ▶ Class Hierarchy
  - ▶ File List

Class Index

A | B | C | D | E | F | G | H | I | K | L | M | N | O | P | R | S | T | U | V | W | Z

A

AnisoPyramid
AsymRippleBuilder
AttLimits
Attributes
AxesUnitsWrap
UnitConverterSimple::AxisData
AxisInfo

B

BaseMaterialImpl
Basic2DLatticeBuilder
Basic2DParaCrystalBuilder
BasicLattice
BasicVector3D
Beam
Bin1D
Bin1DCVector
Bin1DKVector
BiPyramid
Box
BoxCompositionBuilder
BoxCompositionRotateXBuilder
BoxCompositionRotateYBuilder
BoxCompositionRotateZandYBuilder
BoxCompositionRotateZBuilder
BoxesSquareLatticeBuilder
BoxStackCompositionBuilder

C

IntegratorReal::CallBackHolder
IntegratorMCMiser::CallBackHolder

FitStatus
FixedBinAxis
FootprintFactorGaussian
FootprintFactorSquare
FormatErrorException (Exceptions)
FormFactorAnisoPyramid
FormFactorBox
FormFactorCoherentPart
FormFactorCoherentSum
FormFactorComponents
FormFactorCone
FormFactorCone6
FormFactorCoreShell
FormFactorCrystal
FormFactorCuboctahedron
FormFactorCylinder
FormFactorDebyeBueche
FormFactorDecoratorDebyeWaller
FormFactorDecoratorMaterial
FormFactorDecoratorPositionFactor
FormFactorDecoratorRotation
FormFactorDodecahedron
FormFactorDot
FormFactorDWBA
FormFactorDWBAPol
FormFactorEllipsoidalCylinder
FormFactorFullSphere
FormFactorFullSpheroid
FormFactorGauss
FormFactorHemiEllipsoid
FormFactorIcosahedron
FormFactorLongBox
FormFactorLongBoxGauss

IClusteredParticles
IComputation
Icosahedron
IdentityRotation
IDetector
IDetector2D
IDetectorResolution
IDistribution1D
IDistribution1DSampler
IDistribution2DSampler
IFactory
IFootprintFactor
IFormFactor
IFormFactorBorn
IFormFactorDecorator
IFresnelMap
IFTDecayFunction1D
IFTDecayFunction2D
IFTDistribution1D
IFTDistribution2D
IFunctionAdapter (Fit)
IHistogram
IIntensityFunction
IIntensityNormalizer
IInterferenceFunction
IInterferenceFunctionStrategy
ILatticeOrientation
ILayerRTCoefficients
ILayout
IMinimizer
IMultiLayerBuilder
INamed
InfinitePlane

M

MagneticCylindersBuilder
MagneticMaterialImpl
MagneticParticleZeroFieldBuilder
MagneticRotationBuilder
MagneticSpheresBuilder
MagneticSubstrateZeroFieldBuilder
Material
MaterialBySLDImpl
PlainMultiLayerBySLDBuilder::MaterialData
MatrixFresnelMap
MatrixRTCoefficients
MesoCrystal
MesoCrystalBuilder
MillerIndex
MillerIndexOrientation
Minimizer (Fit)
MinimizerCatalogue
MinimizerFactory
MinimizerInfo
MinimizerOptions
MinimizerResult (Fit)
Minuit2Minimizer
MPISimulation
MultiLayer
MultilayerInfo
MultiLayerWithRoughnessBuilder
MultiOption
MultipleLayoutBuilder

N

NodeIterator

BornAgain C++ API

FormFactorBox Class Reference

Hard particles

Description

A rectangular prism (parallelepiped).

Definition at line **23** of file **FormFactorBox.h**.

▶ Inheritance diagram for FormFactorBox:

Public Member Functions

	FormFactorBox (double length, double width, double height)
	Constructor of a rectangular cuboid. More...
FormFactorBox *	**clone** () const overridefinal
	Returns a clone of this **ISample** object.
void	**accept** (**INodeVisitor** *visitor) const overridefinal
	Calls the **INodeVisitor**'s visit method.
double	**getLength** () const
double	**getHeight** () const
double	**getWidth** () const
double	**radialExtension** () const overridefinal
	Returns the (approximate in some cases) radial size of the particle of this form factor's shape. More...
complex_t	**evaluate_for_q** (**cvector_t** q) const overridefinal
	Returns scattering amplitude for complex scattering wavevector q=k_i-k_f. More...

BornAgain C++ API

FormFactorBox.cpp

```cpp
1   // *********************************************************************** //
2   //
3   //  BornAgain: simulate and fit scattering at grazing incidence
4   //
5   //! @file      Core/HardParticle/FormFactorBox.cpp
6   //! @brief     Implements class FormFactorBox.
7   //!
8   //! @homepage  http://www.bornagainproject.org
9   //! @license   GNU General Public License v3 or higher (see COPYING)
10  //! @copyright Forschungszentrum Jülich GmbH 2018
11  //! @authors   Scientific Computing Group at MLZ (see CITATION, AUTHORS)
12  //
13  // *********************************************************************** //
14
15  #include "FormFactorBox.h"
16  #include "BornAgainNamespace.h"
17  #include "Box.h"
18  #include "MathFunctions.h"
19  #include "RealParameter.h"
20
21  //! Constructor of a rectangular cuboid.
22  //! @param length: length of the base in nanometers
23  //! @param width: width of the base in nanometers
24  //! @param height: height of the box in nanometers
25  FormFactorBox::FormFactorBox(double length, double width, double height)
26      : m_length(length), m_width(width), m_height(height)
27  {
28      setName(BornAgain::FFBoxType);
29      registerParameter(BornAgain::Length, &m_length).setUnit(BornAgain::UnitsNm).setNonnegative();
30      registerParameter(BornAgain::Width,  &m_width).setUnit(BornAgain::UnitsNm).setNonnegative();
31      registerParameter(BornAgain::Height, &m_height).setUnit(BornAgain::UnitsNm).setNonnegative();
32      onChange();
33  }
34
35  complex_t FormFactorBox::evaluate_for_q(cvector_t q) const
36  {
37      complex_t qzHdiv2 = m_height/2*q.z();
38      return m_height*m_length*m_width *
39          MathFunctions::sinc(m_length/2*q.x()) *  MathFunctions::sinc(m_width/2*q.y()) *
40              MathFunctions::sinc(qzHdiv2) * exp_I(qzHdiv2);
41  }
42
```

Features available only in Python

Simulation
- Simulation with distributed parameters
- Some exotic interference functions

Fitting
- Fitting multiple data sets
- Fit along slices
- Third party minimizers
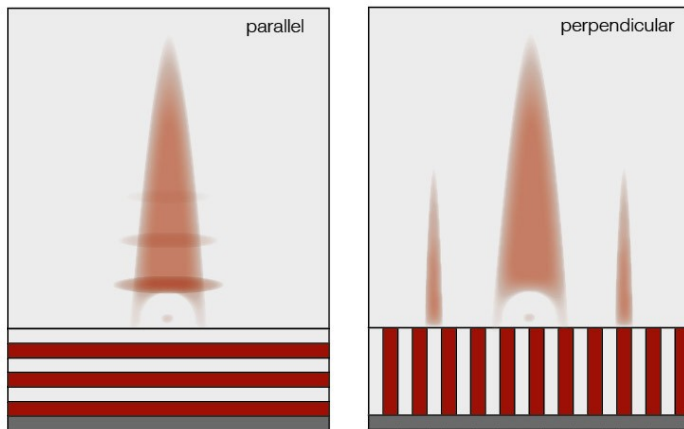- Custom objective functions

Complex workflows

Demo
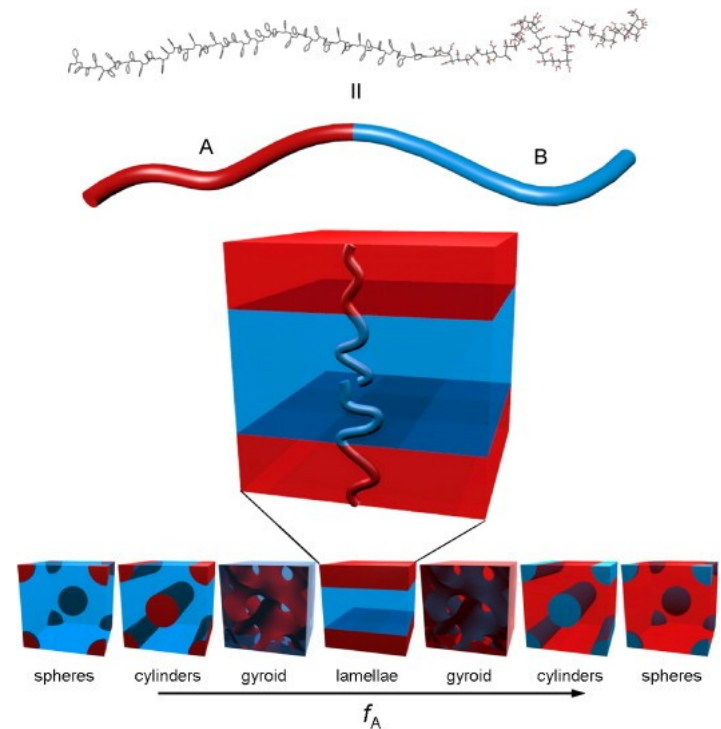
Simulating lamellar structure

`pyapi04_lamellar.py`

Simulating lamellar structures

○ Represents one of ordered phase of block copolymers during self-assembly

○ Alternating layers of different materials in the form of lamellae


https://wiki.anton-paar.com/

The resulting GISAS pattern depends on the size and arrangement of lamellar structure

Simulating lamellar structures

Steps to simulate vertically oriented lamellae in BornAgain using PythonAPI

- Define two materials
  - HomogeneousMaterial
- Define two boxes
  - FormFactorBox
- Define lamellar structure as stack of boxes
  - ParticleComposition
- Add rotation around Z
  - RotationZ
- Apply rotation angle distribution 0-180
  - ParticleDistribution
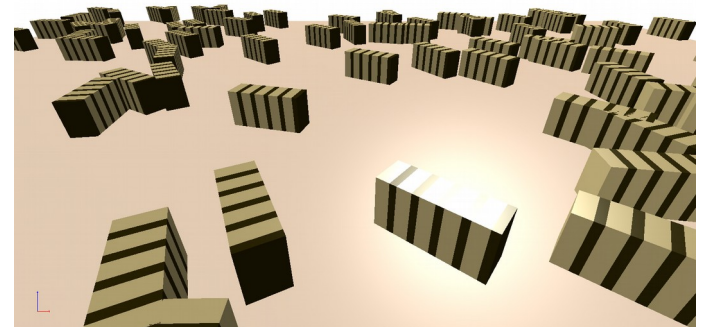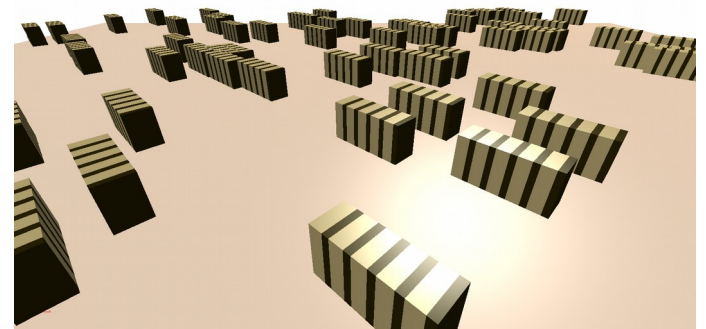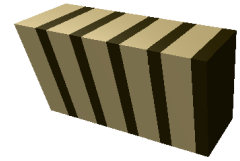
Simulating lamellar structures
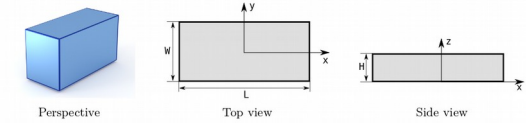

11.1.2 Box (cuboid)

Real-space geometry

Figure 11.5: A rectangular cuboid.

Syntax and parameters

`FormFactorBox(double length, double width, double height)`

```python
def get_horizontal_lamellar():
    mat_a = ba.HomogeneousMaterial("PTFE", 5.20508729E-6, 1.96944292E-8)
    mat_b = ba.HomogeneousMaterial("HMDSO", 2.0888308E-6, 1.32605651E-8)
    length = 30*nm
    width_a = 4*nm
    width_b = 8*nm
    height = 30*nm
    nstack = 5
    stack = ba.ParticleComposition()
    for i in range(0, nstack):
        box_a = ba.Particle(mat_a, ba.FormFactorBox(length, width_a, height))
        box_b = ba.Particle(mat_b, ba.FormFactorBox(length, width_b, height))
        stack.addParticle(box_a, ba.kvector_t(0.0, i*(width_a+width_b), 0.0))
        stack.addParticle(box_b, ba.kvector_t(0.0, (width_a + width_b)/2. + i*(width_a+width_b), 0.0))
```
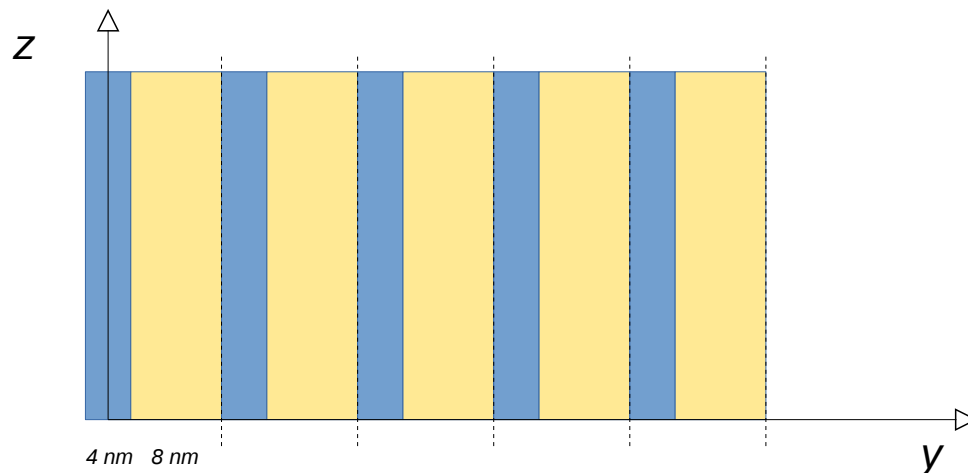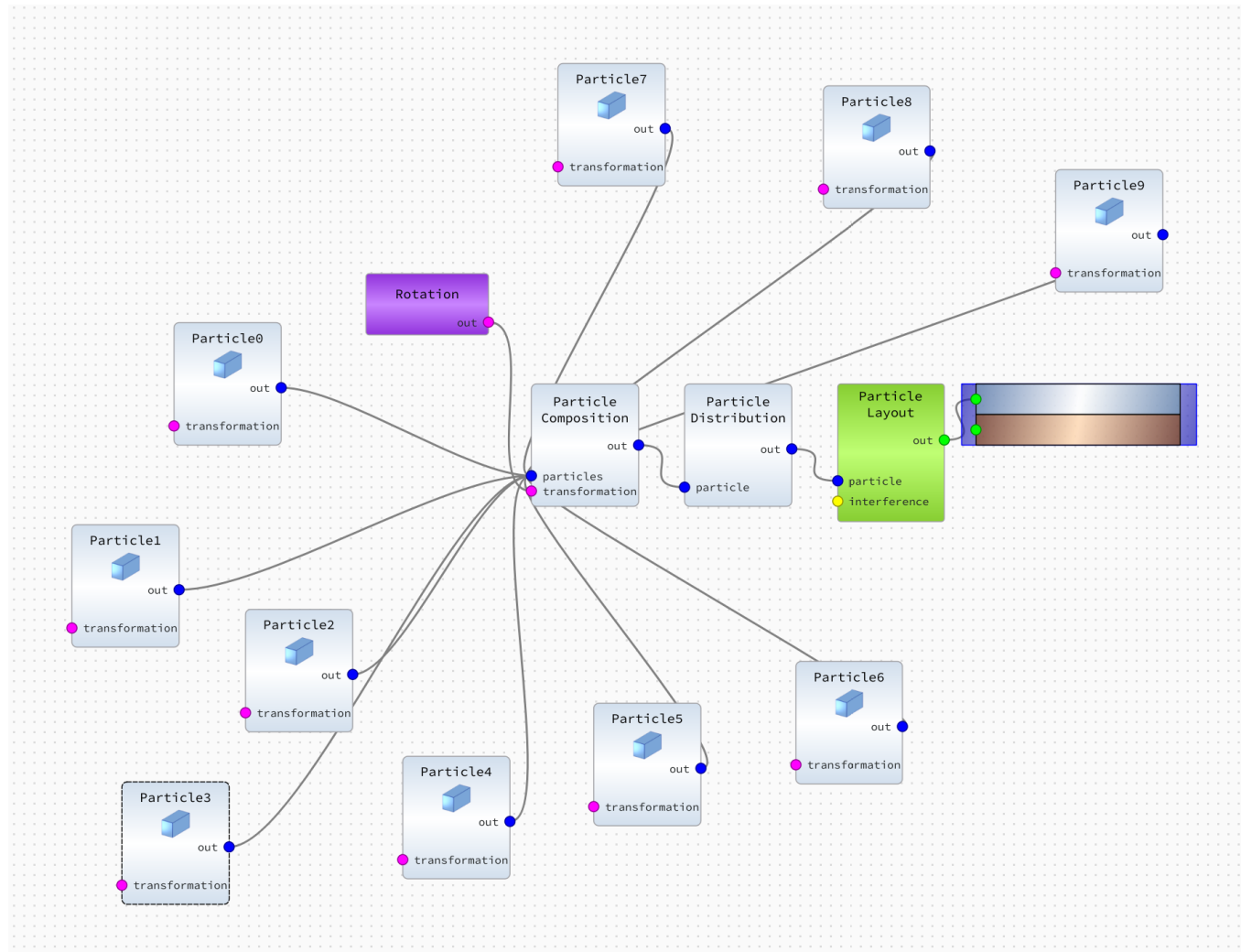
Simulating lamellar structures

Results
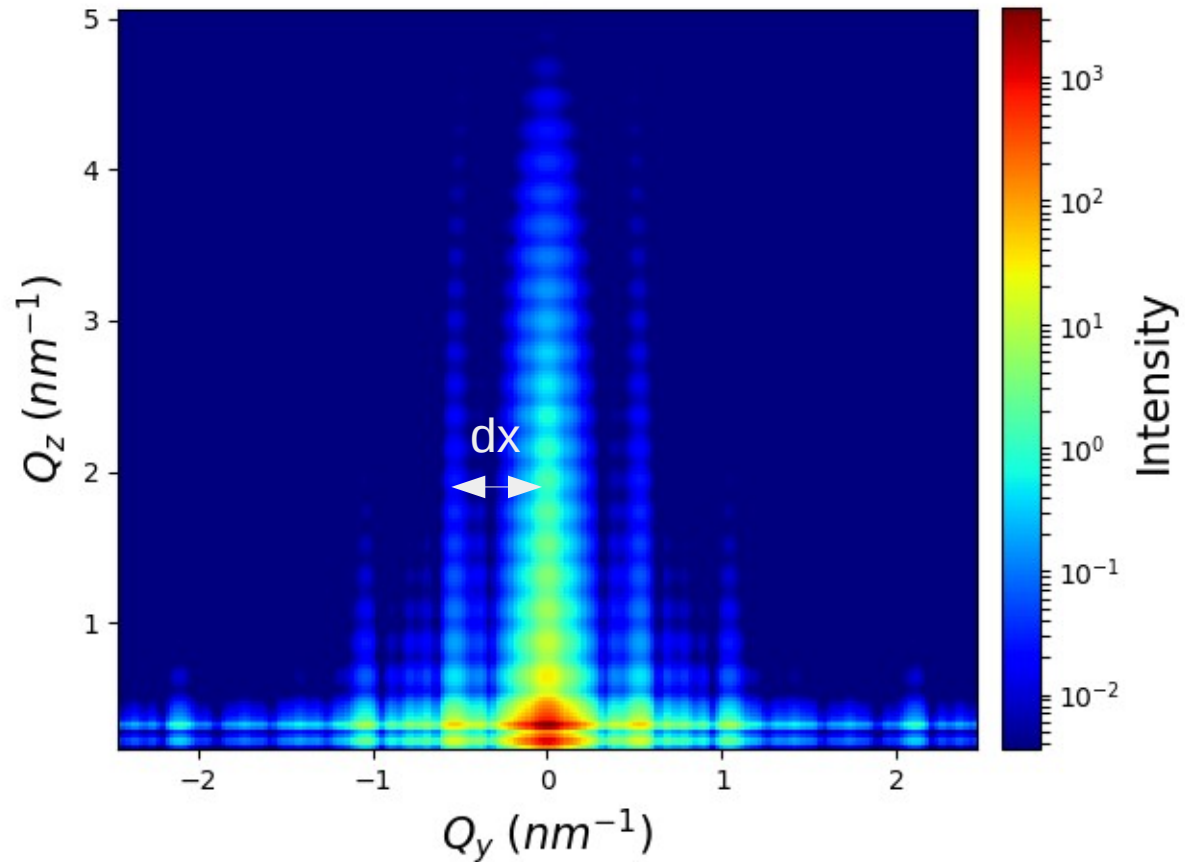
$2*\pi/dx = 12.0$ nm

the value coincide with lamellar period
which was defined in simulation script

Simulating lamellar structures

If we would have to do it in GUI …

Task

Modifying lamellar example

pyapi04_lamellar_vertical_solution1.py

pyapi04_lamellar_vertical_solution2.py

Modifying lamellar example

Task: make lamellar parallel to surface