BIGMECA Data Platform

A. Marano

COPIL – Chaire BIGMECA

26 août 2021

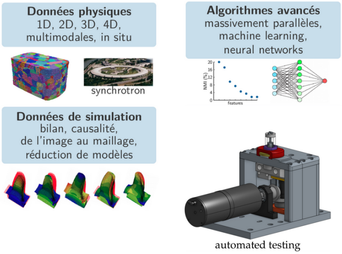
BIGMECA Chair PostDoc

Context

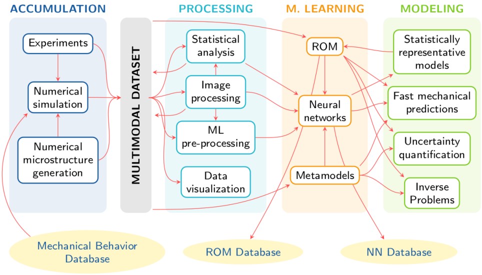
- Rise of automated 4D testing
- Maturity of HPC Material simulation
- 4D Characterization/Simulation of microstructure/damage/plasticity
- Mutltimodal Datasets : EBSD, SEM, DCT, XCT, FEM, FFT, CAD...

Challenges

- Exploit data through complex numerical chains
- Develop a unified (open)data framework
- Applications : structure alloys →lifetime assessment, damage, plastic localization



BIGMECA DATA FLOW



Contents

1. SampleData platform : last updates
   1.1 Data Platform Beta version
   1.2 Data Platform new features
2. Data platform applications
3. Conclusion

Table of Contents

1. SampleData platform : last updates
   - 1.1 Data Platform Beta version
   - 1.2 Data Platform new features

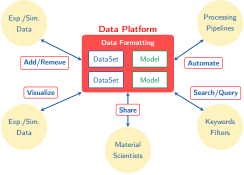
BIGMECA Data Platform

Features

- XDMF/HDF5 – Paraview
- Data management, compression
- High-level user friendly API
- Flexible and rich data model
- GitHub – Full Documentation

Interfaces

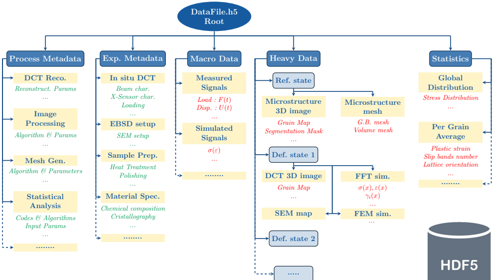
- Generic classes to create interfaces
- F. Nguyen Automatic mesher
- Zset and Amitex_FFTP
- Subclass dedicated for polycrystalline datasets (sim, DCT, EBSD outputs)



README.rst

Pymicro is an open-source Python package to work with material microstructures and 3d data sets.

docs passing | build passing | License MIT



Platform Interfaces

"Intrinsic" interfaces

- Neper
- Dream3D
- OIM (EBSD)
- DCT, LabDCT outputs

Zset

- I/O .geof files
- Automated Zset meshers
- Read and write Zset .ut
- Some Zset FEM commands automated

Automatic Mesher

- 2D and 3D Multi-phase mesher (Matlab)
- Surface mesh of phases contour + volume mesh
- Fully automated

AMITEX FFTP

- Amitex input writing from Microstructure automated
- Automated reading of fields and macro output

Platform Documentation

Pymicro Data Platform: The Sample Data and Microstructure Classes User Guide

The *pymicro* package is based on a data format designed to create, organize and manage efficiently complex multi-modal datasets, used in material science, with a particular focus on the mechanics of microstructures. These data sets are said to be multi-modal because they bring together data from various measurement and numerical simulation techniques, originally produced in very different formats.

- Doc and Tutorial at the same time
- Based on *Jupyter Notebook and NbSphinx*
- Covers all SampleData + Micro usage byt not interfaces
- Online + auto-build *ReadTheDocs*

Now that we have our `numpy.dtype`, and our structured array, we can create the *table*:

```
# create the structured array data item
tab = data.add_table(name='test_table', location='test_group', indexname='tableT', description=sample_type,
                     data=sample_array)

# adding one attribute to the table
data.add_attributes({'tutorial_section':'VI'},'tableT')

# printing information on the table
data.print_node_info('tableT')
```

```
Adding table `test_table` into Group `test_group`

-- Compression Options for dataset test_table

 NODE: /test_group/test_table
====================
 -- Parent Group : test_group
 -- Node name : test_table
 -- test_table attributes :
	 * node_type : structured array
	 * tutorial_section : VI

 -- content : /test_group/test_table (Table(2,)) ''
 -- table description :
{
  "Nature": StringCol(itemsize=25, shape=(), dflt=b'', pos=0),
  "Id_number": Int16Col(shape=(), dflt=0, pos=1),
  "Dimensions": Float64Col(shape=(3,), dflt=0.0, pos=2),
  "Damaged": BoolCol(shape=(), dflt=False, pos=3)}
 -- Compression options for node `tableT`:
	complevel=0, shuffle=False, bitshuffle=False, fletcher32=False, least_significant_digit=None
 -- Node memory size :    63.984 Kb
----------------
```


Now that we have our `numpy.dtype`, and our structured array, we can create the *table*:

[39]:
```
# create the structured array data item
tab = data.add_table(name='test_table', location='test_group', indexname='tableT', descriptio
                     data=sample_array)

# adding one attribute to the table
data.add_attributes({'tutorial_section':'VI'},'tableT')

# printing information on the table
data.print_node_info('tableT')
```

```
Adding table `test_table` into Group `test_group`

-- Compression Options for dataset test_table

 NODE: /test_group/test_table
====================
 -- Parent Group : test_group
 -- Node name : test_table
 -- test_table attributes :
	 * node_type : structured array
	 * tutorial_section : VI

 -- content : /test_group/test_table (Table(2,)) ''
 -- table description :
{
  "Nature": StringCol(itemsize=25, shape=(), dflt=b'', pos=0),
  "Id_number": Int16Col(shape=(), dflt=0, pos=1),
  "Dimensions": Float64Col(shape=(3,), dflt=0.0, pos=2),
  "Damaged": BoolCol(shape=(), dflt=False, pos=3)}
 -- Compression options for node `tableT`:
	complevel=0, shuffle=False, bitshuffle=False, fletcher32=False, least_significant_digi
 -- Node memory size :    63.984 Kb
----------------
```

Platform deployment

- For external users :
  - "Beta" version accessible through GitHub online (already a few external users)
  - Enhanced test base
- Deployment at Centre des Matériaux :
  - Currently deployed as a standard tool on CdM workstations and cluster
  - Automatic mesher distributed as separated package with working examples
  - Real Use case maintenance base to test advanced usages
- Future :
  - Support and development help from B. Marchand and L. Lacourt
  - Formation for new users, PhD students and interns
  - GUI

Table of Contents

1. SampleData platform : last updates
   1.1 Data Platform Beta version
   1.2 Data Platform new features

Base functions improvement

- Intuitive access to datasets :

```
# get the test array in a variable with the attribute like access
array2 = data.test_array
```

```
# get array in a variable, using the data item Name
array = data['test_array']
```

- Improved Data Model :
  - *String Arrays*
  - Enhanced support for structured arrays, IP fields, mesh nodes/element sets
  - Improved items naming consistency

- Major bugs solved

Base functions improvement

- Improved prints and get methods :

```
data.print_dataset_content(short=True)
```

```
Printing dataset content with max depth 3
  |--GROUP test_group: /test_group (Group)
     --NODE test_array: /test_group/test_array (data_array) (   64.000 Kb)

  |--GROUP test_image: /test_image (3DImage)
     --NODE Field_index: /test_image/Field_index (string array) (   63.999 Kb)
     --NODE test_image_field: /test_image/test_image_field (field_array) (   63.867 Kb)

  |--GROUP test_mesh: /test_mesh (3DMesh)
     --NODE Field_index: /test_mesh/Field_index (string array) (   63.999 Kb)
    |--GROUP Geometry: /test_mesh/Geometry (Group)
       --NODE Elem_tag_type_list: /test_mesh/Geometry/Elem_tag_type_list (string array) (   63.99
       --NODE Elem_tags_list: /test_mesh/Geometry/Elem_tags_list (string array) (   63.999 Kb)
       --NODE Elements: /test_mesh/Geometry/Elements (data_array) (   64.000 Kb)
      |--GROUP ElementsTags: /test_mesh/Geometry/ElementsTags (Group)
      |--GROUP NodeTags: /test_mesh/Geometry/NodeTags (Group)
       --NODE Node_tags_list: /test_mesh/Geometry/Node_tags_list (string array) (   63.999 Kb)
       --NODE Nodes: /test_mesh/Geometry/Nodes (data_array) (   63.984 Kb)
       --NODE Nodes_ID: /test_mesh/Geometry/Nodes_ID (data_array) (   64.000 Kb)

     --NODE Test_field1: /test_mesh/Test_field1 (field_array) (   64.000 Kb)
     --NODE Test_field2: /test_mesh/Test_field2 (field_array) (   64.000 Kb)
     --NODE Test_field3: /test_mesh/Test_field3 (field_array) (   64.000 Kb)
     --NODE Test_field4: /test_mesh/Test_field4 (field_array) (   64.000 Kb)
```

```
data.print_node_attributes('Tarray')
```

```
-- test_array attributes :
        * empty : False
        * node_type : data_array
        * tutorial_file : 2_SampleData_basic_data_items.ipynb
        * tutorial_section : Section IV
```

Data Compression

```python
# Set up compression settings with lossy compression: truncate after third digit adter decimal point
compression_options = {'complib':'zlib', 'complevel':9, 'shuffle':True,  'least_significant_digit':2,
                       'normalization':'standard_per_component'}
data.set_chunkshape_and_compression(nodename='Amitex_stress_1', compression_options=compression_options)
```

- Lossy compression with data normalization :
  - Automatic data normalization : centered reduced complete array, or per component
  - Significant digit number reduced –> compression of useless bits
  - Automatic removal when visualizing or getting array


```xml
<Attribute Name="Amitex_stress_1" AttributeType="Tensor6" Center="Cell">
    <DataItem ItemType="Function" Function="($2*$0) + $1" Dimensions="100  100  100  6">
      <DataItem Format="HDF" Dimensions="100  100  100  6" NumberType="Float" Precision="64">Test_compression.h5:/CellData/Amitex_output_fields/Amitex_stress_1</DataItem>
      <DataItem Format="HDF" Dimensions="100  100  100  6" NumberType="Float" Precision="64">Test_compression.h5:/CellData/Amitex_output_fields/Amitex_stress_1_norm_mean</DataItem>
      <DataItem Format="HDF" Dimensions="100  100  100  6" NumberType="Float" Precision="64">Test_compression.h5:/CellData/Amitex_output_fields/Amitex_stress_1_norm_std</DataItem>
    </DataItem>
</Attribute>
```


- Compression ratios
  - Stress field, FFT solver output, 10^6 voxels, $\sim$ 50 Mo.
  - Lossless compression : $\sim$ 21 Mo
  - Lossy compression with 2 significant digits : $\sim$ 12.5 Mo
  - Normalized lossy compression with 2 significant digits : $\sim$ 5 Mo

XDMF temporal grids

```
# instant 0
data.add_field(gridname='image2D', fieldname='time_field', location='image2D', indexname='Field',
               array=temporal_field[:,:,:,0], time=instants[0])
# instant 1
data.add_field(gridname='image2D', fieldname='time_field', location='image2D', indexname='Field',
               array=temporal_field[:,:,:,1], time=instants[1])
# instant 2
data.add_field(gridname='image2D', fieldname='time_field', location='image2D', indexname='Field',
               array=temporal_field[:,:,:,1], time=instants[2])
```

- Field time series :
  - Automatically handled when time information provided
  - Creation of time series when reading Amitex or Zset outputs
  - Paraview animations

```
<Grid Name="first_2D_image" GridType="Collection" CollectionType="Temporal">
  <Grid Name="first_2D_image_T0" GridType="Uniform">
```

Contents

1 SampleData platform : last updates

2 Data platform applications

2.1 Multimodal 4D polycrystalline datasets

3 Conclusion

Pymicro official publication

- Publication goals :
  - Evidence data management issues to highlight need for data platform
  - Proposition of a unified format/tool
  - Promote definition of standards (formats, data models)
  - Official publication for Pymicro package
- Publication content :
  - Litterature Review of data platforms
  - Data platform model, format and capabilities presentation
  - Application datasets
- Journal :
  - *Current Opinion in Solid State & Materials Science*
  - Review oriented. Short articles ($\leq$ 10 pages)
  - Follow up of previous article from H. Proudhon

Table of Contents

2 Data platform applications
  2.1 Multimodal 4D polycrystalline datasets

Dataset presentation

- Material :
  - Grade 2 polycrystalline Ti sample
  - C. Ribart's thesis study material
- Data :
  - X-DCT : pre/post mechanical test
    near field (grain map, orientations), far field (distorsion)
  - In-situ MEB/EBSD mechanical testing
    (lattice rotation, slip bands imaging)
  - Numerical simulation, FFT-based solver, crystal plasticity
  - Microstructure conformal mesh
- Goals :
  - Comparison of fine 2D measures with 3D informed simulation
  - Show how data platform is central to methodology
  - Publish rich multimodal dataset with data model proposition

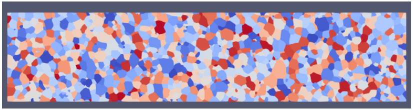
DCT and EBSD data

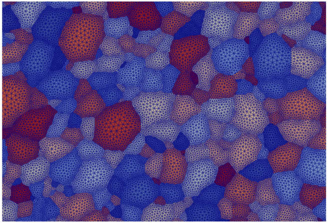
EBSD

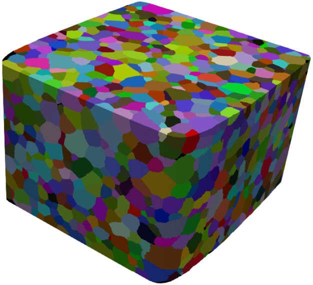
- Automatic load from OIM file
- Automatic meshing (280425 elements, 140924 nodes)
- Current challenge : geometric alignment wit DCT
- $\sim$ 1228 grains

DCT

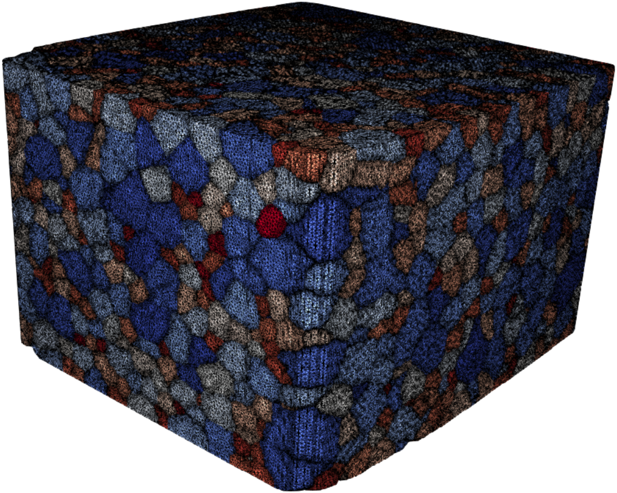
- Loaded, morphological cleaning, automatic meshing
- Microstructure, Mask, Uncertainty map
- Data compression
- $\sim$ 4000 grains, 88 585 238 voxels







DCT Mesh



9 613 886 nodes, 221 035 452 elements

DCT and mesh dataset uncompressed

```
Printing dataset content with max depth 3
  |--GROUP CellData: /CellData (3DImage)
     --NODE grain_map: /CellData/grain_map (None) (  166.875 Mb)
     --NODE grain_map_raw: /CellData/grain_map_raw (None) (  166.875 Mb)
     --NODE mask: /CellData/mask (None) (   83.438 Mb)
     --NODE orientation_map: /CellData/orientation_map (None) (    1.932 Gb)
     --NODE phase_map: /CellData/phase_map (None) (   64.000 Kb)
     --NODE uncertainty_map: /CellData/uncertainty_map (None) (   83.438 Mb)

  |--GROUP CrystalStructure: /CrystalStructure (Group)
     --NODE LatticeParameters: /CrystalStructure/LatticeParameters (None) (   64.000 Kb)

  |--GROUP GrainData: /GrainData (Group)
     --NODE GrainDataTable: /GrainData/GrainDataTable (None) (  255.938 Kb)

  |--GROUP MeshData: /MeshData (emptyMesh)
    |--GROUP grains_mesh: /MeshData/grains_mesh (3DMesh)
       --NODE Field_index: /MeshData/grains_mesh/Field_index (None) ( 1023.984 Kb)
      |--GROUP Geometry: /MeshData/grains_mesh/Geometry (Group)
       --NODE grain_Ids: /MeshData/grains_mesh/grain_Ids (None) (   43.724 Mb)


  |--GROUP PhaseData: /PhaseData (Group)
    |--GROUP phase_01: /PhaseData/phase_01 (Group)
```

Complete size : 4.5 Go (many element sets –> heavy storage, thousands of arrays)

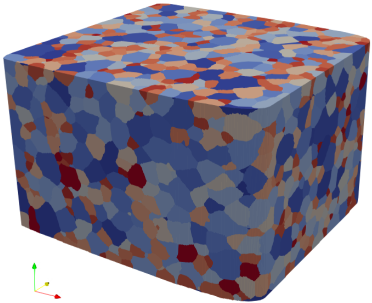
DCT and EBSD dataset compressed

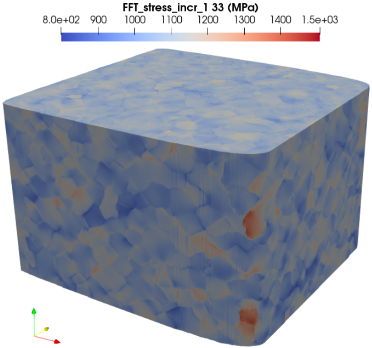
```
Printing dataset content with max depth 3
  |--GROUP CellData: /CellData (3DImage)
     --NODE Field_index: /CellData/Field_index (string array) (   63.999 Kb)
     --NODE grain_map: /CellData/grain_map (field_array) (   15.885 Mb)
     --NODE grain_map_raw: /CellData/grain_map_raw (field_array) (   20.540 Mb)
     --NODE mask: /CellData/mask (field_array) (  961.229 Kb)
     --NODE orientation_map: /CellData/orientation_map (field_array) (   43.185 Mb)
     --NODE phase_map: /CellData/phase_map (field_array) (   64.000 Kb)
     --NODE uncertainty_map: /CellData/uncertainty_map (field_array) (    8.529 Mb)

  |--GROUP CrystalStructure: /CrystalStructure (Group)
     --NODE LatticeParameters: /CrystalStructure/LatticeParameters (None) (   64.000 Kb)

  |--GROUP EBSD_CellData: /EBSD_CellData (3DImage)
     --NODE Field_index: /EBSD_CellData/Field_index (string array) (   63.999 Kb)
     --NODE ebsd_confidence_index: /EBSD_CellData/ebsd_confidence_index (field_array) (    1.664 Mb)
     --NODE ebsd_euler: /EBSD_CellData/ebsd_euler (field_array) (    4.042 Mb)
     --NODE ebsd_grain_map: /EBSD_CellData/ebsd_grain_map (field_array) (  167.799 Kb)
     --NODE ebsd_iq: /EBSD_CellData/ebsd_iq (field_array) (    1.891 Mb)

  |--GROUP GrainData: /GrainData (Group)
     --NODE GrainDataTable: /GrainData/GrainDataTable (None) (  255.938 Kb)

  |--GROUP MeshData: /MeshData (emptyMesh)
    |--GROUP ebsd_grains_mesh: /MeshData/ebsd_grains_mesh (3DMesh)
       --NODE Field_index: /MeshData/ebsd_grains_mesh/Field_index (string array) (   63.999 Kb)
      |--GROUP Geometry: /MeshData/ebsd_grains_mesh/Geometry (Group)
       --NODE grains_mesh_elset_ids: /MeshData/ebsd_grains_mesh/grains_mesh_elset_ids (field_array) (    2.188 Mb)


  |--GROUP PhaseData: /PhaseData (Group)
    |--GROUP phase_01: /PhaseData/phase_01 (Group)
```

Numerical simulation

- Set up :
  - Massively parallel FFT-based solver : AMITEX FFTP
  - National supercomputer : Jean Zay cluster (IDRIS, CNRS)
  - Solver installed on Jean Zay
- Simulation chain
  - Data pre-post processing scripts developed
  - Cluster scripts to run simulations developed
  - Constitutive law developed, currently validated
- Tests :
  - Elastic simulations with DCT volume
  - Currently : subvolume tests with target constitutive behavior

Numerical simulation





Constitutive law implementation

Model

- Finite strain crystal plasticity model
- Hexagonal crystal, prismatic, basal and pyramidal slip
- Details in C. Ribart talk

Implementation

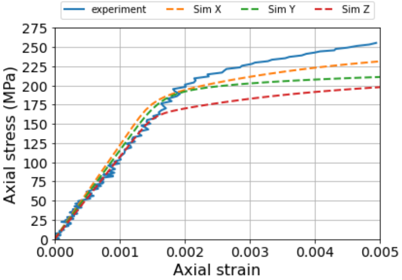
- MFront code generator –> UMAT
- Implicit system, theta method + Newton Raphson

Validation

- Elasticity validation
- Test of schmid law on various orientations

Identification

- Details in C. Ribart talk
- FFT test simulation with 100 grains subvolume
  $\longrightarrow$ large discrepancy with optimization
- Critical shear stresses too low ?
- Bug in constitutive law implementation ?



Contents

1. SampleData platform : last updates
2. Data platform applications
3. Conclusion

Conclusion

- Plate-forme de donnée "version Beta"
  - Package complet en ligne
  - Déployée au Centre des Matériaux
  - Article et applications en cours
- Travail sur les incertitudes liées au jumeau numérique DCT suspendu
- Fin du post-doc
  - Début à l'ONERA au 1er octobre
  - Ingénieur de recherche modélisation des matériaux métalliques
  - équipe de P. Kanouté, au DMAS