Creating the Materials Innovation Infrastructure of MGI 2.0: Streaming by design for materials characterization.

Julia Saga Viktoria Camacho Wejbrandt

2022 PARADIM and HEMI REU intern @ Johns Hopkins University

Intern affiliation: Guys Campus/Hospital, King's College London

Program: 2021 Platform for the Accelerated Realization, Analysis, and Discovery of Interface Materials Research Experience for Undergraduates Program at Johns Hopkins University (PARADIM REU @ JHU)
PARADIM REU Principal Investigator: Dr Tyrel McQueen, Head of McQueen Lab, Johns Hopkins University
PARADIM REU Mentor: Dr David Elbert, Hopkins Extreme Materials Institute (HEMI), Johns Hopkins University
Primary Source of JHU Funding: NSF VariMat award number 2129051

Contact: julia.camacho_wejbrandt@kcl.ac.uk , elbert@jhu.edu
Website: REU Participants | PARADIM
Primary JHU PARADIM Tools Used: PARADIM Data Collective (PDC), Laser floating zone, X-ray diffractometer, Laue diffractometer, Physical Property Measurement System (PPMS), Magnetic Property Measurement System (MPMS).

Program: PARADIM, HEMI and McQueen Lab REU Program

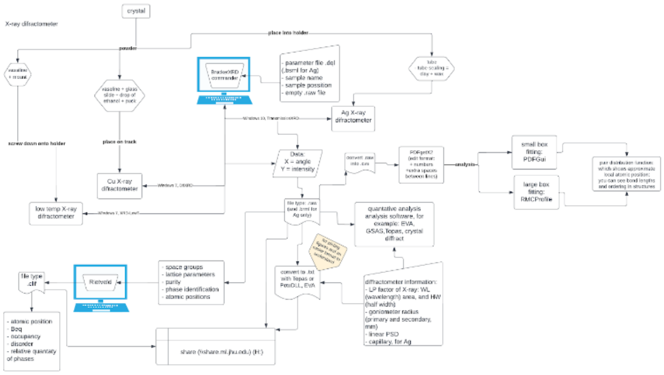
Introduction

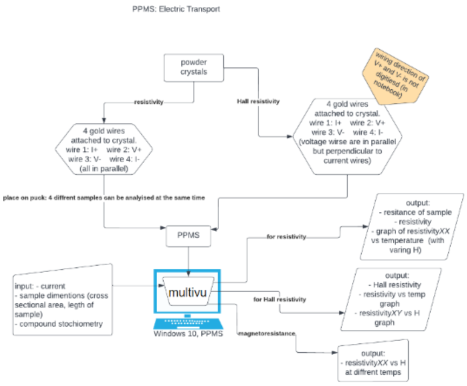
For more than a decade the Materials Genome Initiative (MGI) has put data at the centre of significantly accelerated materials development and deployment. The 2021 MGI Strategic Plan highlighted the need for Materials Innovation Infrastructure and community sharing of data to fuel machine learning and other data-intensive approaches to revolutionize materials research. the fundamental goal of this project has been developing methods and instantiations of the graphical data model (GEMD) used in PARADIM to fully capture materials synthesis and characterization details. Such detailed data model instantiations provide details, including both successful and unsuccessful experiments, of the entire research workflow that are central to developing AI implementation for automation of the research process. With data fully described in GEMD we will be able to make information, scientific findings, and data more FAIR (findable, accessible, interoperable, and reusable) to researchers around the world.

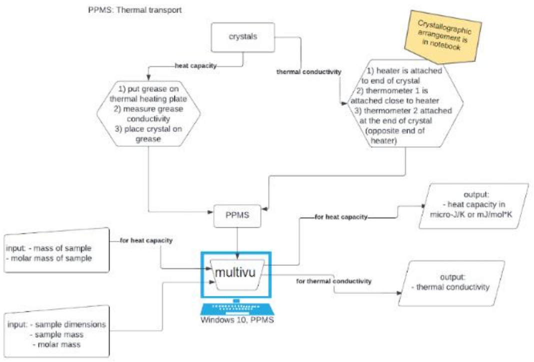
Methodology:

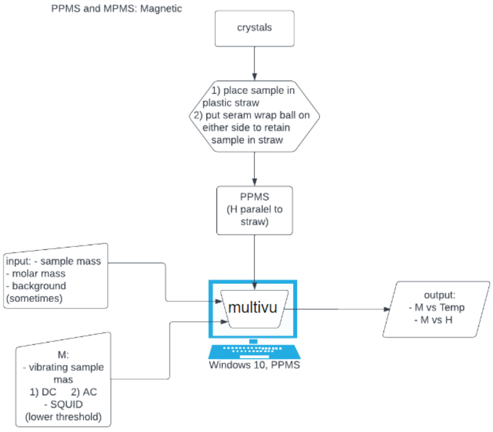
A central part of the research was development of techniques to capture and laboratory procedures and develop codes that is user friendly so experimentalists can translate their workflow into GEMD compliant JSON file representations. Our main focus was on instrument usage; from input variables, set up procedures and output variables, we are able to make a simplified and general flowchart that encompasses the important information for each instrument that is commonly used in the material science field. From there we work on hard coding this information into GEMD objects.

Flowchart examples – these are some of the flowcharts that were made.


X-ray diffractometer
crystal
powder
place into holder
vaseline + mount
vaseline + glass slide + drop of ethanol + puck
BruckerXRD commander
- parameter file .dql (.bsml for Ag)
- sample name
- sample possition
- empty .raw file
tube
- tube sealing = clay + wax
Ag X-ray diffractometer
Windows 10, TransmissionXRD
screw down onto holder
place on track
Data:
X = angle
Y = intensity
convert .raw into .eva
PDFgetX2 (edit format: + numbers +extra spaces between lines)
analysis
small box fitting: PDFGui
large box fitting: RMCProfile
pair distribution function: which shows approximate local atomic position: you can see bond lengths and ordering in structures
Cu X-ray diffractometer
Windows 7, DIXRD
low temp X-ray diffractometer
Windows 7, XRD Lead
file type: .raw (and .bsml for Ag only)
quantative analysis analysis software, for example: EVA, GSAS,Topas, crystal diffract
file type .cif
Rietveld
- space groups
- lattice parameters
- purity
- phase identification
- atomic positions
convert to .txt with Topas or PowDLL, EVA
diffractometer information:
- LP factor of X-ray: WL (wavelength) area, and HW (half width)
- goniometer radius (primary and secondary, mm)
- linear PSD
- capillary, for Ag
- atomic position
- Beq
- occupancy
- disorder
- relative quantaty of phases
share (\\share.ml.jhu.edu) (H:)


PPMS: Electric Transport
powder crystals
resistivity
Hall resistivity
wiring direction of V+ and V- is not digitiesd (in notebook)
4 gold wires attached to crystal. wire 1: I+ wire 2: V+ wire 3: V- wire 4: I- (all in parallel)
4 gold wires attached to crystal. wire 1: I+ wire 2: V+ wire 3: V- wire 4: I- (voltage wirse are in parallel but perpendicular to current wires)
place on puck: 4 diffrent samples can be analyised at the same time
PPMS
for resistivity
output: - resitance of sample - resistivity - graph of resistivityXX vs temperature (with varing H)
input: - current - sample dimentions (cross sectional area, legth of sample) - compound stochiometry
multivu
Windows 10, PPMS
for Hall resistivity
output: - Hall resistivity - resistivity vs temp graph - resistivityXY vs H graph
magnetoresistance
output: - resistivityXX vs H at diffrent temps


PPMS: Thermal transport
crystals
Crystallographic arrangement is in notebook
heat capacity
thermal conductivity
1) put grease on thermal heating plate 2) measure grease conductivity 3) place crystal on grease
1) heater is attached to end of crystal 2) thermometer 1 is attached close to heater 3) thermometer 2 attached at the end of crystal (opposite end of heater)
PPMS
for heat capacity
output: - heat capacity in micro-J/K or mJ/mol*K
input: - mass of sample - molar mass of sample
for heat capacity
multivu
Windows 10, PPMS
for thermal conductivity
output: - thermal conductivity
input: - sample dimensions - sample mass - molar mass


PPMS and MPMS: Magnetic
crystals
1) place sample in plastic straw 2) put seram wrap ball on either side to retain sample in straw
PPMS (H paralel to straw)
input: - sample mass - molar mass - background (sometimes)
multivu
Windows 10, PPMS
output: - M vs Temp - M vs H
M: - vibrating sample mas 1) DC 2) AC - SQUID (lower threshold)

Encoding GEMD objects

All the code was constructed in Python-kernel Jupyter notebooks using four categories:

User input – Users are presented a series of statements where to manually enter information. Users hit enter to execute the code. . For example:

```
PPMS_ET_setup = {

    #conditions

    ’voltage orientation’: ’parallel’, #either label ’parallel’(to current)or(hall resistivity)then state either ’+V on top’ or ’+V on bottom’

    #parameters

    ’current’: ’6’#in amps}
```

Template – Templates provide controls consistent with laboratory capability and experiment design on each of the inputs given. For example, if there are only two different possibilities that a wire can attach to a sample, then restraining the input to only two answers allows efficient, error-checking representations.

```
temp_ET = ProcessTemplate(

    ’Electric Transport PPMS setup’,

    parameters=[ParameterTemplate(’humidity level in lab’, bounds=RealBounds(0.0,1.0,’kg/kg’) ),

ParameterTemplate(’voltage orientation’, bounds=CategoricalBounds([’+V on top’, ’+V on bottom’,’parallel’,]) )])
```

Spec – Specifications allow users to use templates to build processes for a given experiment. Specs check conformity with the template and then convert to GEMD objects.

```
spec_ET = ProcessSpec(

    name=’Eletric Transport PPMS setup’,

    template=temp_ET,

    conditions=[Condition(’voltage orientation’,value=NominalCategorical(PPMS_ET_setup[’voltage orientation’]))],

    parameters=[Parameter(’current’,value=NominalReal(float(PPMS_ET_setup[’current’]),’amp’),)]

run_ET = make_instance(spec_ET)
```

Convert Spec into a JSONfile – JSON files allows for structured data to be transmitted into the GEMD data modelling platform to construct complete materials history graphs stored in a queryable database.

```
entire_model_fn = ’ET_PPMS_entire_model.json’

with open(pathlib.Path()/entire_model_fn,’w’) as fp :

    fp.write(encoder.dumps([run_ET],indent=2)
```