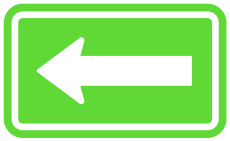
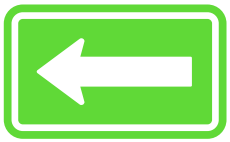
Freely available (DFT) databases
Spawned by MGI
Awareness about license agreement

- **Materials genome initiative**
  - https://www.mgi.gov

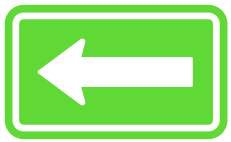


- **Materials Project**
  - https://materialsproject.org



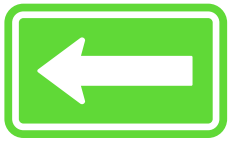
- NOMAD
  - https://nomad-coe.eu

- **Open quantum materials database**
  - http://oqmd.org



- **AFLOW**
  - http://www.aflow.org

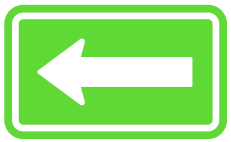
- **Computational materials repository**
  - https://cmr.fysik.dtu.dk



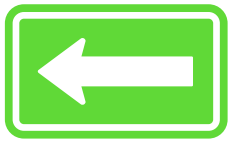
- **novomag**
  - https://www.novomag.physics.iastate.edu/structure-database

- **Novamag**
  - https://zenodo.org/records/3241267

- **Alexandria**
  - https://figshare.com/articles/dataset/Alexandria_DB/27174897?file=49622718



- JARVIS-DFT
  - https://jarvis.nist.gov/



Downloading data

We discussing downloading data from three repositories as example

- From Materials Project
  - MP api
  - Downloading materials by mp-id
  - Downloading materials by properties

- From CMR
  - All projects: Getting familiar with the CMR website
  - Downloading ABS_3 & $ABSe_3$ databases
  - Extracting materials data from these
  - Feature construction from materials structures

- Alexandria
  - Convex hull data
  - 3D materials with PBE

Useful software packages/libraries

- conda
- Atomic Simulation Environment (ASE)
- pymatgen
- DScribe (Arxiv:1904.08875)
- DScribe does not run on python 3.12.x, runs on python 3.9
- Create an environment ‘materials’ for using scikit-learn (for example) which can run with put-on 3.12
    - $ Conda create -n materials
    - Can be activated by: $ conda activate materials
- Install pymatgen in ‘materials’ environment
    - $ conda install -c condo-forge pymatgen
- Also install: custodian, fireworks, ASE, matplotlib in ‘materials’ environment
    - $ conda install -c conda-forge custodian, fireworks, ase, matplotlib
    - Test pymatgen installation:
        - $ python

          >> from pymatgen.core import structure

- DScribe
- Create separate environment for DScribe
    - $ Conda create -n dscribe_env python=3.9
    - $ Conda activate dscribe_env

- Now install DScribe
    - $ conda install -c conda-forge dscribe
    - Check installation
        - $ python
        - >>from describe.descriptors import SOAP

- Install pymatgen within dscribe_env, either
    - $ pip install pymatgen
    - $ conda install -c conda-forge pymatgen

Accessing Materials Project data

- Register
- Unlimited, free access to API, data
- Note your API key on you dashboard
- Data queried and accessed through REST API (Representational State Transfer Application Programming Interface)
    - REST API allows your program to ‘talk’ to MP server using URLs and HTTP

- Access, query, download data from MP
    - Mp-api, REST API (Getting started)
    - Install mp-api: $ conda install -c conda-forge mp-api
    - Importing MPRester client: code fragment, available_fields
    - Documentation for MPRester’s classes and methods
- Examples:
    - Available *endpoints*
    - Data by materials id
    - Summary data with property filter

Available fields

by which to query materials from the summary endpoint

```
['builder_meta', 'nsites', 'elements',
'nelements', 'composition',
'composition_reduced', 'formula_pretty',
'formula_anonymous', 'chemsys',
'volume', 'density', 'density_atomic',
'symmetry', 'property_name', 'material_id',
'deprecated', 'deprecation_reasons',
'last_updated', 'origins', 'warnings',
'structure', 'task_ids',
'uncorrected_energy_per_atom',
'energy_per_atom',
'formation_energy_per_atom',
'energy_above_hull', 'is_stable',
'equilibrium_reaction_energy_per_atom',
'decomposes_to', 'xas', 'grain_boundaries',
'band_gap', 'cbm', 'vbm', 'efermi',
'is_gap_direct', 'is_metal',
'es_source_calc_id',
'bandstructure', 'dos', 'dos_energy_up',
'dos_energy_down', 'is_magnetic',
'ordering',
```

```
'total_magnetization',
'total_magnetization_normalized_vol',
'total_magnetization_normalized_formula_uni
ts', 'num_magnetic_sites',
'num_unique_magnetic_sites',
'types_of_magnetic_species',
'bulk_modulus', 'shear_modulus',
'universal_anisotropy',
'homogeneous_poisson', 'e_total',
'e_ionic', 'e_electronic', 'n',
'e_ij_max',
'weighted_surface_energy_EV_PER_ANG2',
'weighted_surface_energy',
'weighted_work_function',
'surface_anisotropy', 'shape_factor',
'has_reconstructed',
'possible_species', 'has_props',
'theoretical', 'database_Ids']
```

- Combine pymatgen and ASE to write structures and properties
  - Structures as ASE json (Javascript Object Notation)
  - json general example

```
{
  "first_name": "ABC",
  "last_name": "XYZ",
  "is_alive": true,
  "age": 27,
  "address": {
    "street_address": "21 blah Street",
    "city": "New Delhi",
    "state": "",
    "postal_code": "1XXXXX"
  },
  "phone_numbers": [
    {
      "type": "home",
      "number": "011 2xxx-xxxx“
    },
    {
      "type": "office",
      "number": “011 2xxx-xxxx“
    }
  ],
  "children": [
    "M",
    "N",
    “P"
  ],
  “spouse": [
    “Q”
  ],
}
```

  - ase json from MP, example

Tasks (From Materials Project)

1. Query materials for different ‘fields’ for different ‘endpoints’
2. Download materials with different search criteria, and and access different available ‘fields’.
3. Create a database for all materials with band gap up to 2.0 eV, containing up to quaternary materials, and with up to 10 atoms in the unit cell containing their band gaps, formation energy per atom and magnetization in csv format.