Avogadro 2 and Open Chemistry

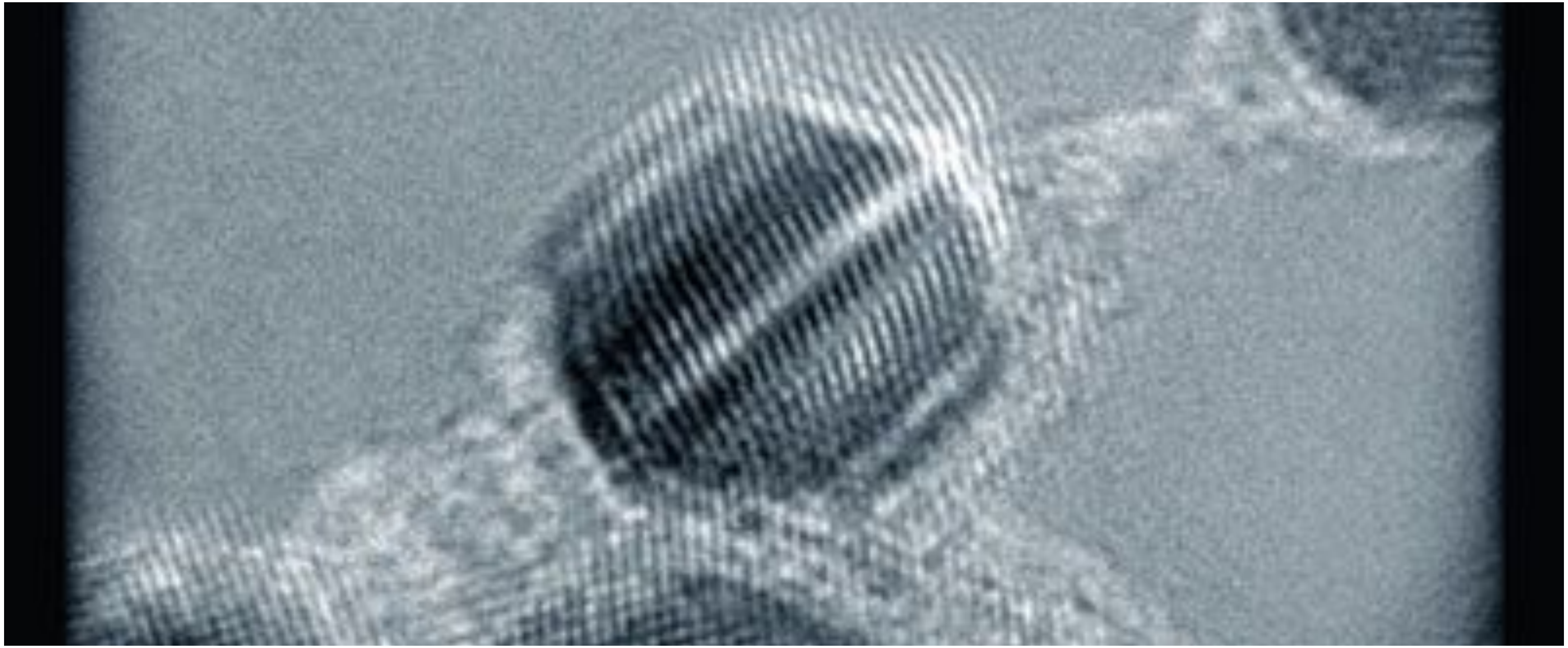
Dr. Marcus D. Hanwell
@mhanwell
Technical Leader
Kitware

Avogadro User Meeting
University of Pittsburgh, PA
25 August, 2018

Some History

- Got a BSc in Physics from the University of Sheffield (2000 - 2003)
  - Did an internship in Silicon Valley in 2002 - servers, computers, simulations, America...
  - Somewhere in about 2002 started packaging for Gentoo Linux on AMD64, scientific packages
- Decided to stay on and do a PhD in Sheffield (2003 - 2007)
  - Nanomaterial Engineering Group working on thiol encapsulated gold nanoparticles
  - Started doing a lot of programming for an experimentalist...data acquisition, simulated nanoparticle packing, helping people get stuff working, C++ plotting of measured surface pressure on Langmuir troughs, simulated X-ray reflectometry, scheduled to finish in 2007
- Thought about taking part in Google Summer of Code since the start
  - Got it together in 2007, went searching for a project in C++ where I could edit molecules
  - Kalzium had the perfect project, started contributing a few patches, talking
  - Submitted a proposal, which was accepted with Benoit Jacob as my mentor

Looking at Nanoparticles in my PhD



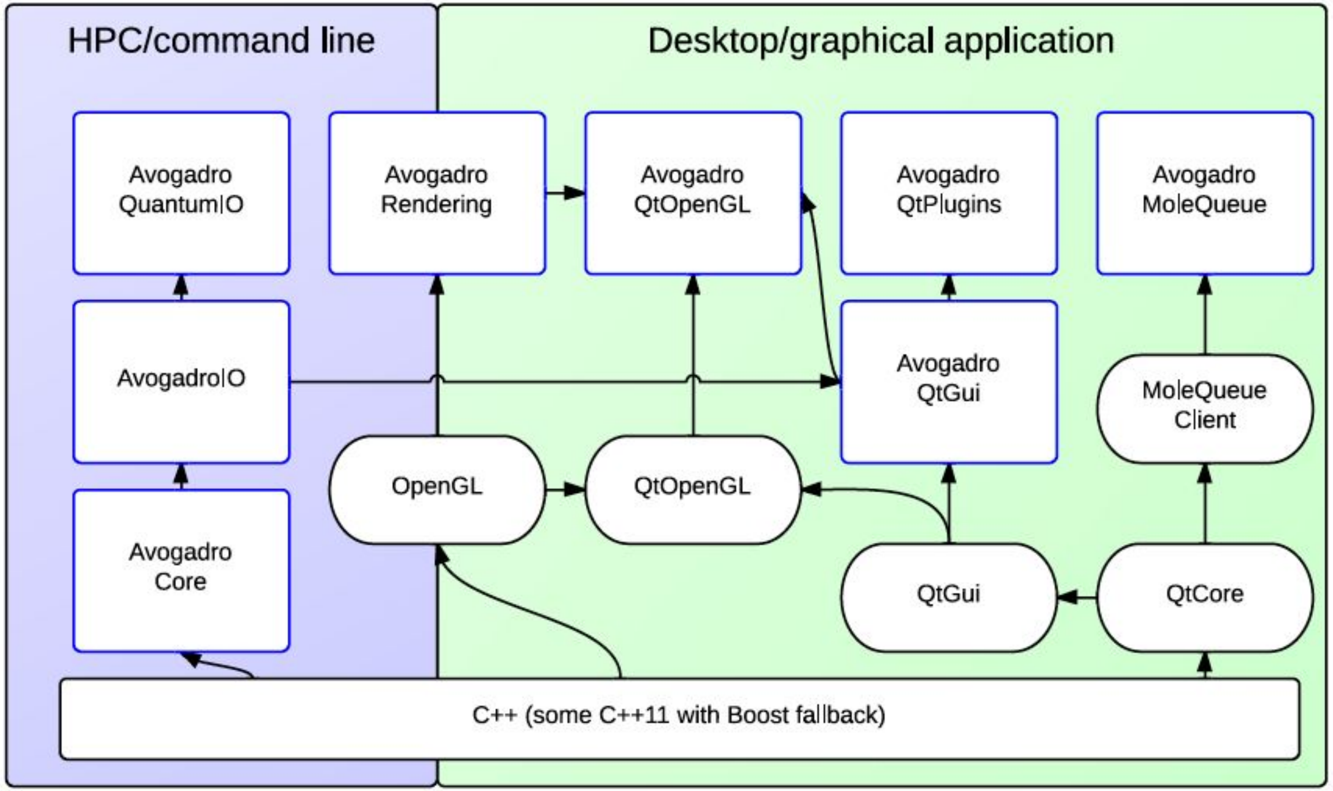
A Little More History...

- Summer of Code project (2007)
  - Kalzium was using Avogadro as its editing library, spent most of my time on that
  - Hacked on code in Paris in the Versailles gardens coming up with eye candy
- Postdoc combining experiment and simulation (2007-2009)
  - Came to Pittsburgh, PA for two years in Geoff’s group as it started up
  - XServe cluster, conductivity rig, AFMs, clean rooms, defect simulation and Avogadro
- Interviewed with Kitware and offered a position (2009 - present)
  - Gave a talk in Jamaica about Avogadro at the first Camp KDE
  - Bill Hoffman was there talking about CMake
- First funding (SBIR) as principal investigator in 2010
  - Founding of broader Open Chemistry project
  - Started rewrite of Avogadro 2, development of MoleQueue, and MongoChem to complement

What Is Avogadro 2?

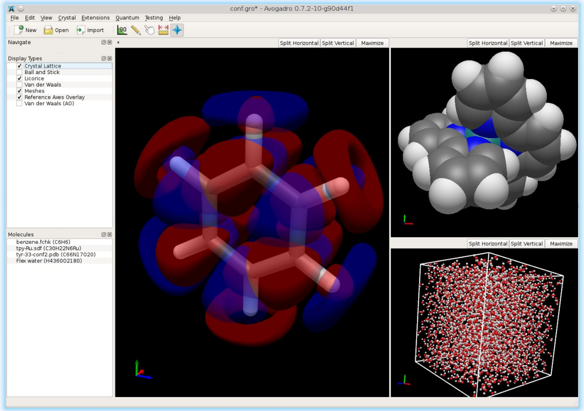
- Ambitious rewrite from the ground up
  - Move to permissive 3-clause BSD license from GPLv2
  - Obtained permission to relicense all Avogado code from contributors
- Using minimal dependencies, multiple focused libraries
  - Core/IO pretty much just C++11
  - Rendering brings in OpenGL
  - Qt classes integrate these things and expose Qt derived classes
  - QtPlugins depend on many things, a lot of functionality in the plugins
- Application in a separate repository as a user of the libraries
- New web-based server code using wrapped core code (no Qt, OpenGL, etc)
- Coded for extensibility, scalability, speed, but also to be useful

High Level Library Dependency Graph



Opportunity to Make Huge Changes

- Underlying data model written for scale
- Rendering engine moved to scene graph
- Use of advanced GLSL impostor rendering
- Completely new input/output code – scale
- Completely new input generators – Python
- Very different APIs, need to port old code
- Move from monolithic single library to smaller, focused libraries
- Focus on executing in external processes where feasible
    - Helps when GPL code is used as we want to remain BSD licensed
    - Helps with stability as external processes will not crash the application
    - Heps with speed as external processes can be executed asynchronously


conf.gro* - Avogadro 0.7.2-10-g90d44f1
File Edit View Crystal Extensions Quantum Testing Help
New Open Import
Navigate
Split Horizontal Split Vertical Maximize

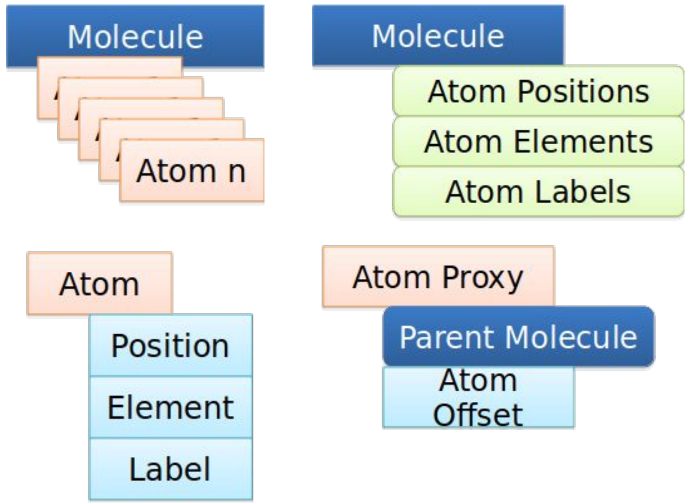
Display Types
- ✓ Crystal Lattice
- ☐ Ball and Stick
- ✓ Licorice
- ☐ Van der Waals
- ✓ Meshes
- ✓ Reference Axes Overlay
- ☐ Van der Waals (AO)

Molecules
- benzene.fchk (C6H6)
- tpy-Ru.sdf (C30H22N6Ru)
- tyr-33-conf2.pdb (C66N17O20)
- Flex water (H436O02180)

Molecule Model and Copy-on-Write

- Molecule is a set of arrays
  - 3D positions, 2D positions, atomic number
  - Custom labels, atom type, others, …
- Atoms/bonds are proxy objects
  - Only contain their index and parent molecule
  - All data resides in molecule
  - Atoms, bonds, etc provided for familiar API
- Only initialize/allocate memory when used
- Everything stored in the molecule
  - x1, y1, z1, x2, y2, z2
  - Atoms refer to their index in parent molecule
- Temporary proxy objects created on demand

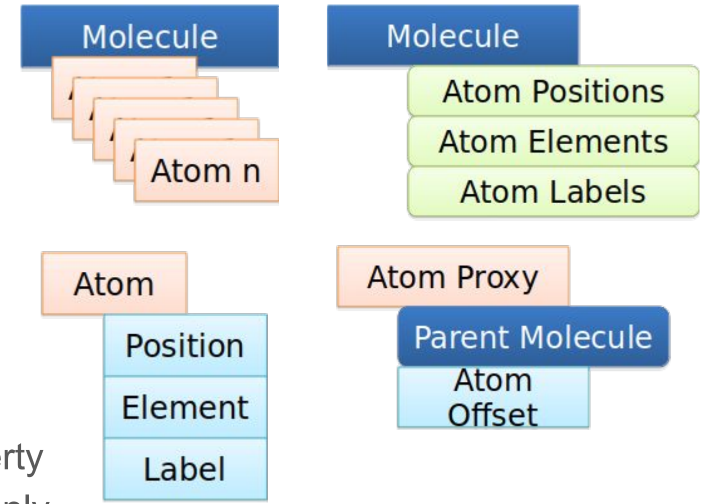
Avogadro 1 versus Avogadro 2



Cheap Copies Until They Are Changed

- When copying a molecule arrays are copied
  - Until data changes array holds no data
  - Refers to thing it is a copy of
  - Point of editing triggers memory copy
- Many copies, but deep copies lazily
  - If you only change atom positions...
    - That is the only array whose memory is copied!
  - Copies are fast – contiguous buffers used!
- Much lower cost for unused properties
  - Old model each atom allocates memory for each property
  - New model creates empty array at the molecule level only
  - Atoms and bonds are ephemeral proxies created to help us work with molecules

Avogadro 1 versus Avogadro 2



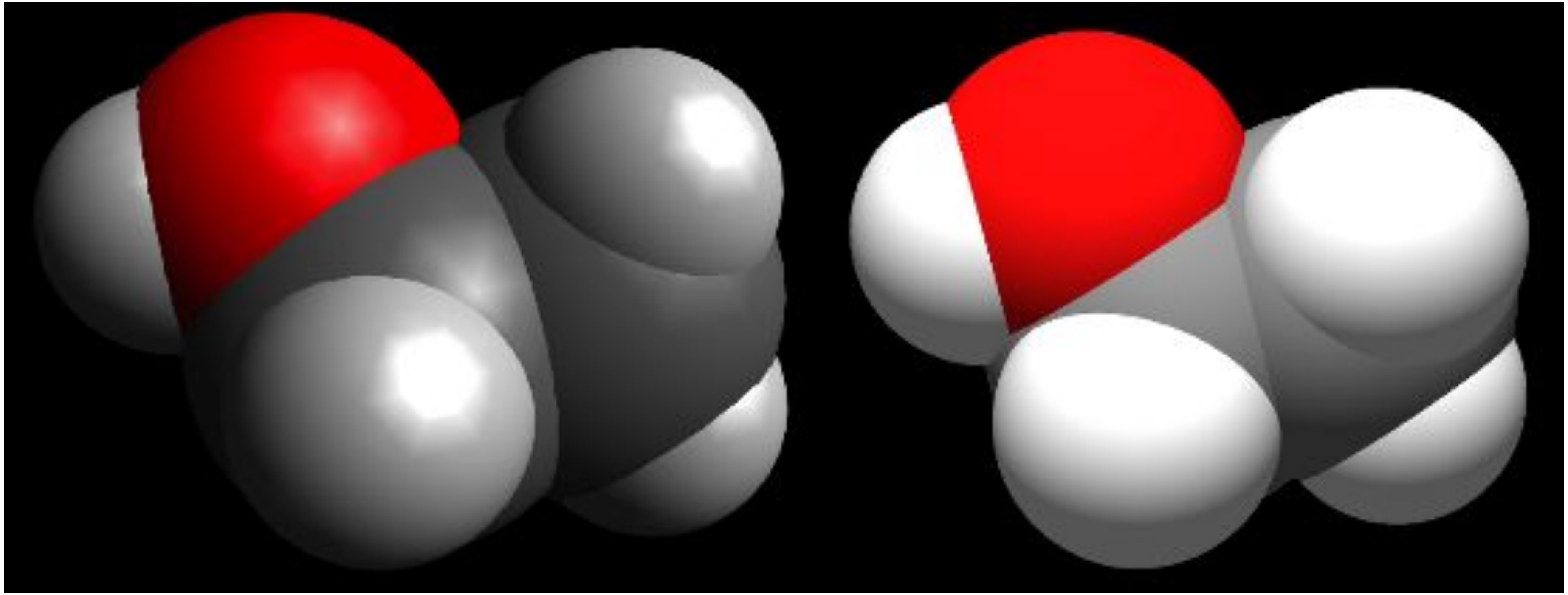
From Painters to Scene Graphs

- Avogadro 1.x has a nice API for adding new visualization engines
  - `painter->setColor(color); painter->drawSphere(center, radius);`
  - Every time we rendered all engines were called for all atoms, bonds, etc
  - Cost of virtual overhead, looking up color for atom type, looking up radius, etc
  - Most renders are just camera changes...
- Avogadro 2 has a nice API for adding new visualization engines
  - `sphereNode->addSphere(center, color, radius);`
  - Only done when the scene changes, after that just render the scene graph
  - Even if quite slow only done if changes to the molecule are made
  - Very amenable to using vertex buffers to store geometry on the graphics card (OpenGL 2.1+)
- The API did not get significantly more complex
  - Can do many more things under the hood to optimize

Using the Best of OpenGL - Impostors

- Avogadro 1.x we rendered spheres using display lists, and level of detail
  - Close spheres used the sphere with the most triangles
  - Far away spheres used the lowest level of detail
  - Had to be calculated for each frame based on the camera
- Avogadro 2 uses impostors - “there is no spoon”
  - All of our spheres are rendered using two triangles
  - Vertex shader ensures it always faces the camera
  - Fragment shader ray traces the shadows pixel by pixel, sets the depth buffer
- Cost for transform, lighting etc went from 100s of vertices to 4!
  - Spheres look better than any we rendered up close
  - Cost is virtually zero when far away or occluded - fragment shader is called per pixel!

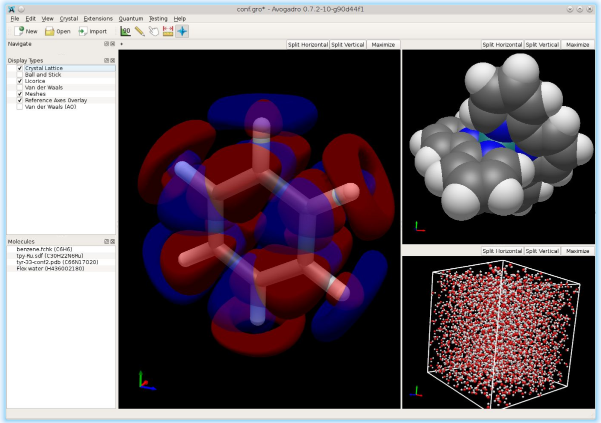
Using the Best of OpenGL - Impostors



Multiple Molecules and Multiple Views

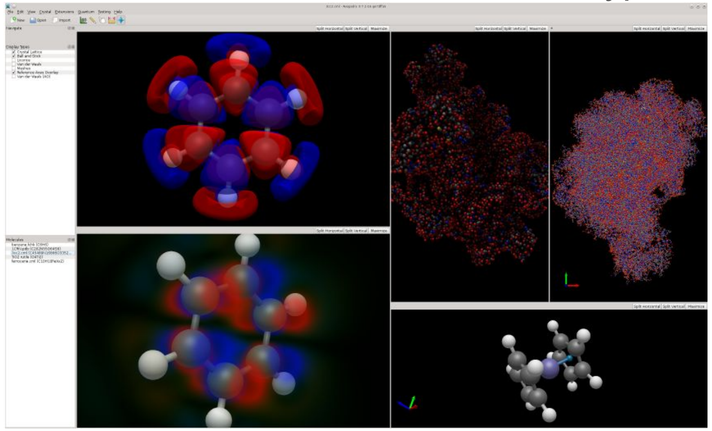
- Support having many molecules loaded
- Easily switch between molecules
- Support different types potentially
- When other applications open molecule
  - No need to save, but switch active view
  - Shift to single window model
- Could shift to pipeline views optionally

- Start with familiar single view
- Split the view horizontally or vertically
  - Dynamically resize views
  - View many molecules, or compare views
- Different view types supported
  - Editor, VTK-based
- Specialize available tools and view types
- Could be extended to cover more types



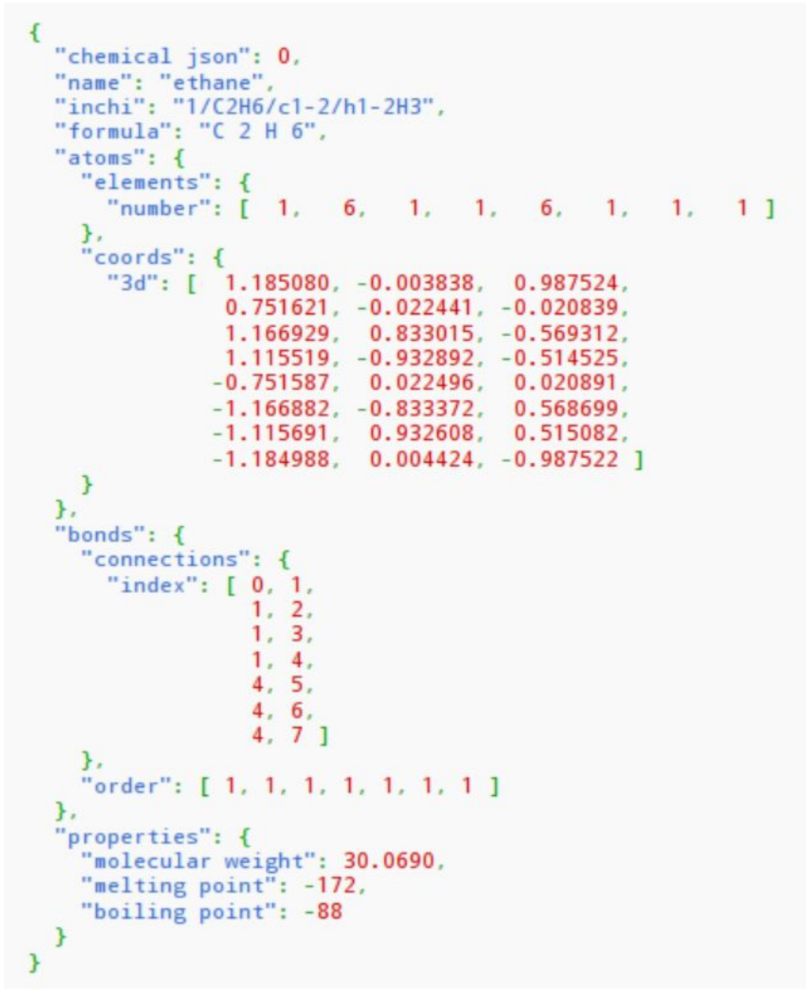
Display Types
- [x] Crystal Lattice
- [ ] Ball and Stick
- [x] Licorice
- [ ] Van der Waals
- [x] Meshes
- [x] Reference Axes Overlay
- [ ] Van der Waals (AO)

Molecules
- benzene.fchk (C6H6)
- tpy-Ru.sdf (C30H22N6Ru)
- tyr-33-conf2.pdb (C66N17O20)
- Flex water (H436002180)



Chemical JSON

- Developed to support projects (~2011)
- Stores structure, geometry, identifiers, descriptors, other useful data
- Benefits:
  - More compact than XML/CML
  - Native to MongoDB, JSON-RPC, REST
  - Easily converted to binary representation
- Can be extended easily
- Unrecognized keys ignored
- **MolSSI JSON schema collaboration - workshop at Berkeley Lab last year**



```
{
  "chemical json": 0,
  "name": "ethane",
  "inchi": "1/C2H6/c1-2/h1-2H3",
  "formula": "C 2 H 6",
  "atoms": {
    "elements": {
      "number": [  1,   6,   1,   1,   6,   1,   1,   1 ]
    },
    "coords": {
      "3d": [  1.185080, -0.003838,  0.987524,
               0.751621, -0.022441, -0.020839,
               1.166929,  0.833015, -0.569312,
               1.115519, -0.932892, -0.514525,
              -0.751587,  0.022496,  0.020891,
              -1.166882, -0.833372,  0.568699,
              -1.115691,  0.932608,  0.515082,
              -1.184988,  0.004424, -0.987522 ]
    }
  },
  "bonds": {
    "connections": {
      "index": [ 0, 1,
                 1, 2,
                 1, 3,
                 1, 4,
                 4, 5,
                 4, 6,
                 4, 7 ]
    },
    "order": [ 1, 1, 1, 1, 1, 1, 1 ]
  },
  "properties": {
    "molecular weight": 30.0690,
    "melting point": -172,
    "boiling point": -88
  }
}
```

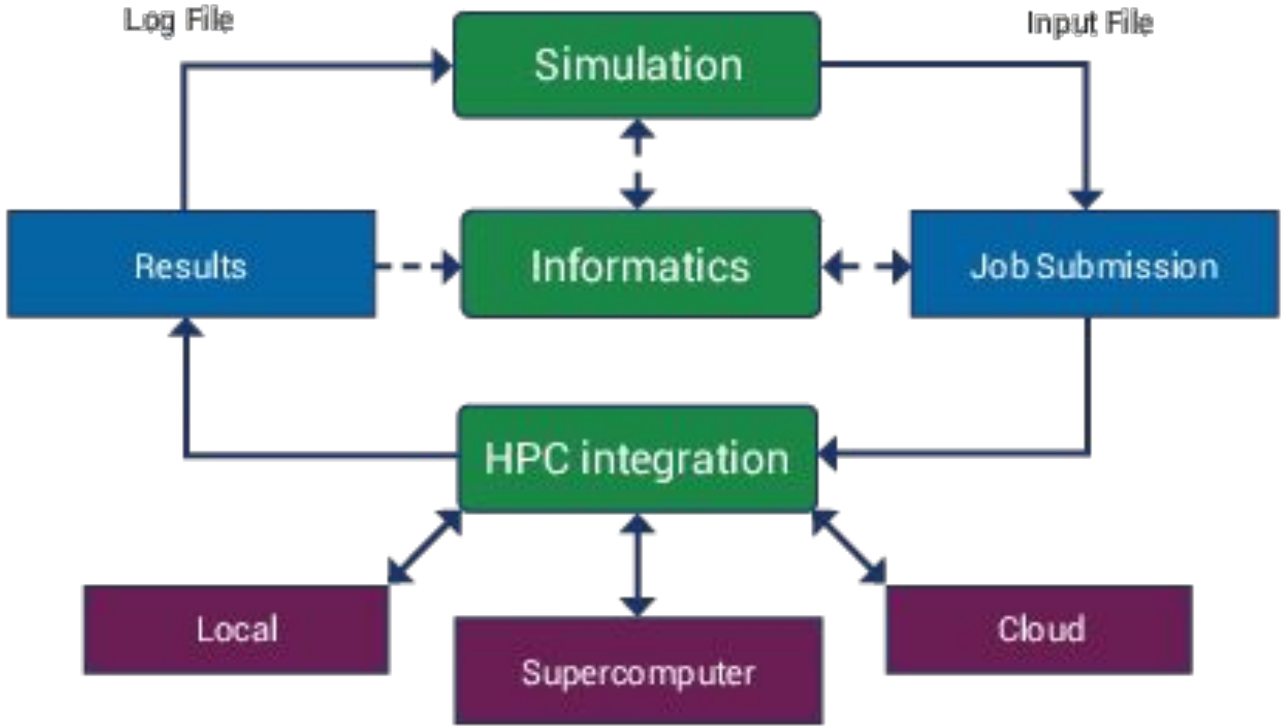
Papers and a Little History on Chemical JSON

- Quixote collaboration with Peter Murray-Rust (2011)
  - “The Quixote project: Collaborative and Open Quantum Chemistry data management in the Internet age”, https://doi.org/10.1186/1758-2946-3-38
- Early work in CML with NWChem and Avogadro (2013)
  - “From data to analysis: linking NWChem and Avogadro with the syntax and semantics of Chemical Markup Language” https://doi.org/10.1186/1758-2946-5-25
- Later moved to JSON, RESTful API, visualization (2017)
  - “Open chemistry: RESTful web APIs, JSON, NWChem and the modern web application”
  - https://doi.org/10.1186/s13321-017-0241-z
- Interested in Linked Data, JSON-LD, and how they might be layered on top
- Use of BSON, HDF5, and related technologies for binary data
- BSD licensed reference implementations

What Is Open Chemistry?

- Umbrella of related projects to coordinate and group
  - Focus on 3-clause BSD permissively licensed projects
  - Aims for more complete solution
- Initially three related projects
  - Avogadro 2 - editor, visualization, interaction with small number of molecules
  - MoleQueue - running computational jobs, abstracting local and remote execution
  - MongoChem - database for interacting with many molecules, summarizing data, informatics
- Evolved over the years but still retains many of those goals
  - GitHub organization with 35 repositories at the last count
- Umbrella organization in Google Summer of Code
  - Three years so far, with 3, 7, and 7 students over a broad range of projects
  - Hope to continue this and other community engagement activities

The Original Open Chemistry Vision



Open Chemistry, Avogadro, Jupyter and Web

- Making data more accessible
- Federated, open data repositories
- Modern HTML5 interfaces
- JSON data format for NWChem data as a prototype, add to other QM codes
- What about working with the data?
- Can we have chemistry from desktop-to-phone
  - Create data, upload, organize
  - Search and analyze data
  - Share data - email, social media, publications
- What if we tied a data server to a Jupyter notebook?
- Can we link generated data to existing government databases?

Kitware

Original MongoChem

- Native, cross platform C++ application built with Qt, Avogadro, VTK
- Stored chemical data in a NoSQL MongoDB database

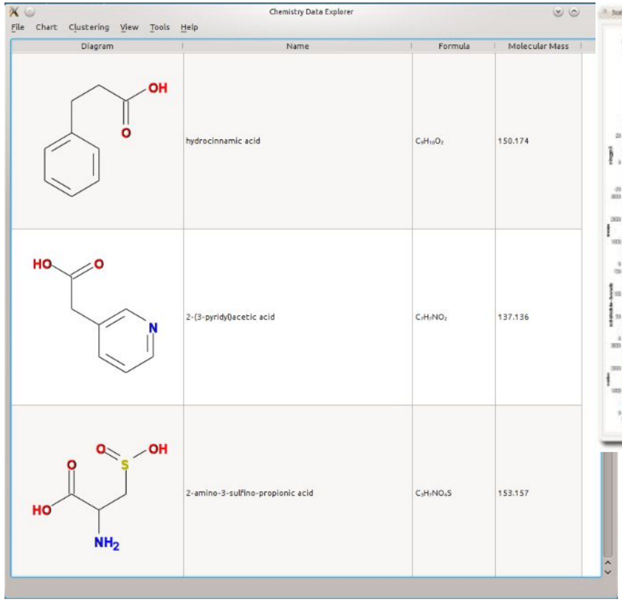
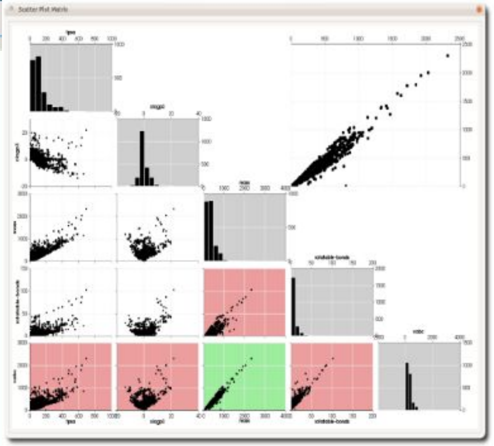
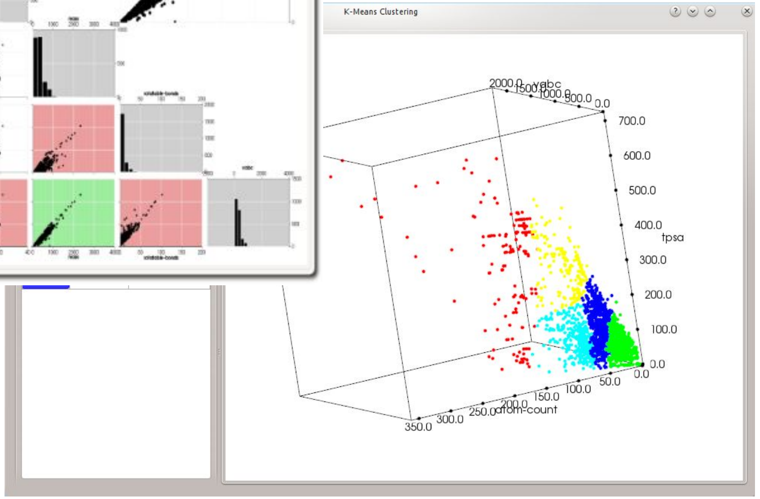
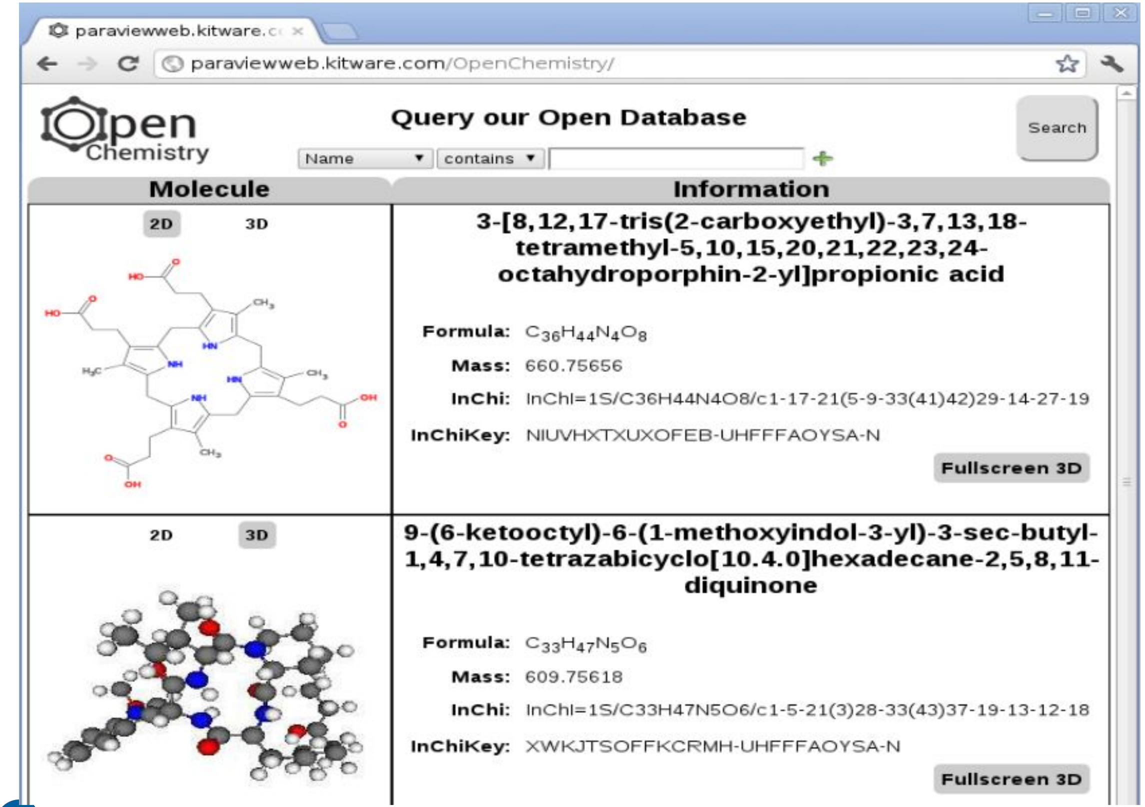


Diagram	Name	Formula	Molecular Mass
	hydrocinnamic acid	$C_9H_{10}O_2$	150.174
	2-(3-pyridyl)acetic acid	$C_7H_7NO_2$	137.136
	2-amino-3-sulfino-propionic acid	$C_3H_7NO_4S$	153.157





ParaViewWeb and Open Chemistry



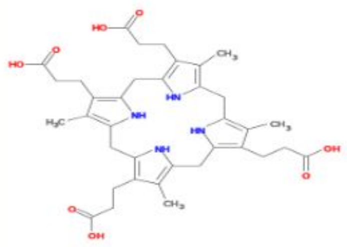
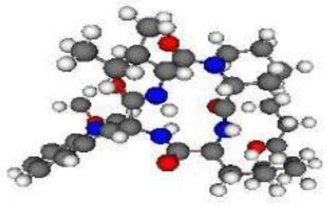
paraviewweb.kitware.com/OpenChemistry/

Open Chemistry

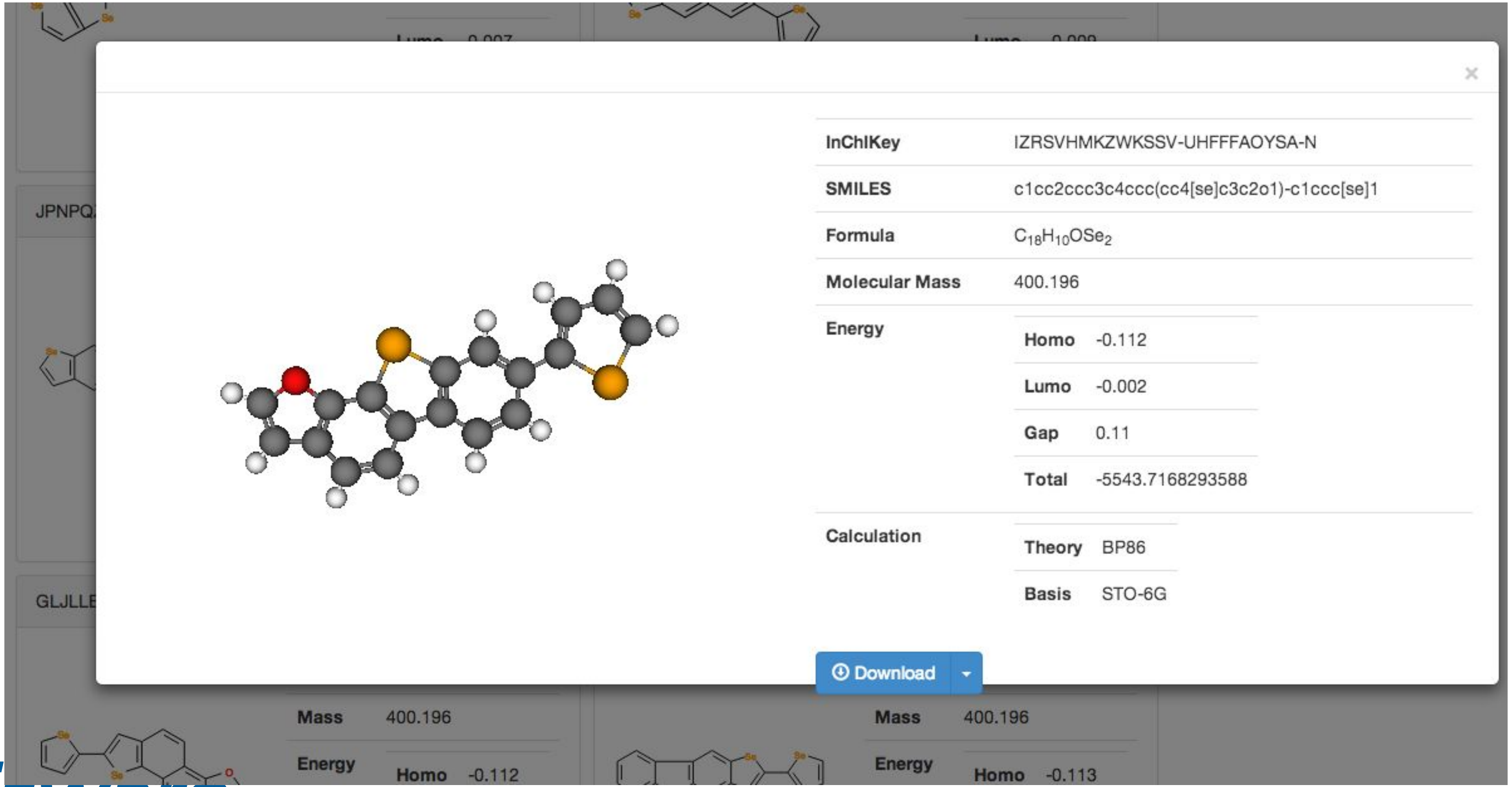
Query our Open Database

Name ▾ contains ▾

Search

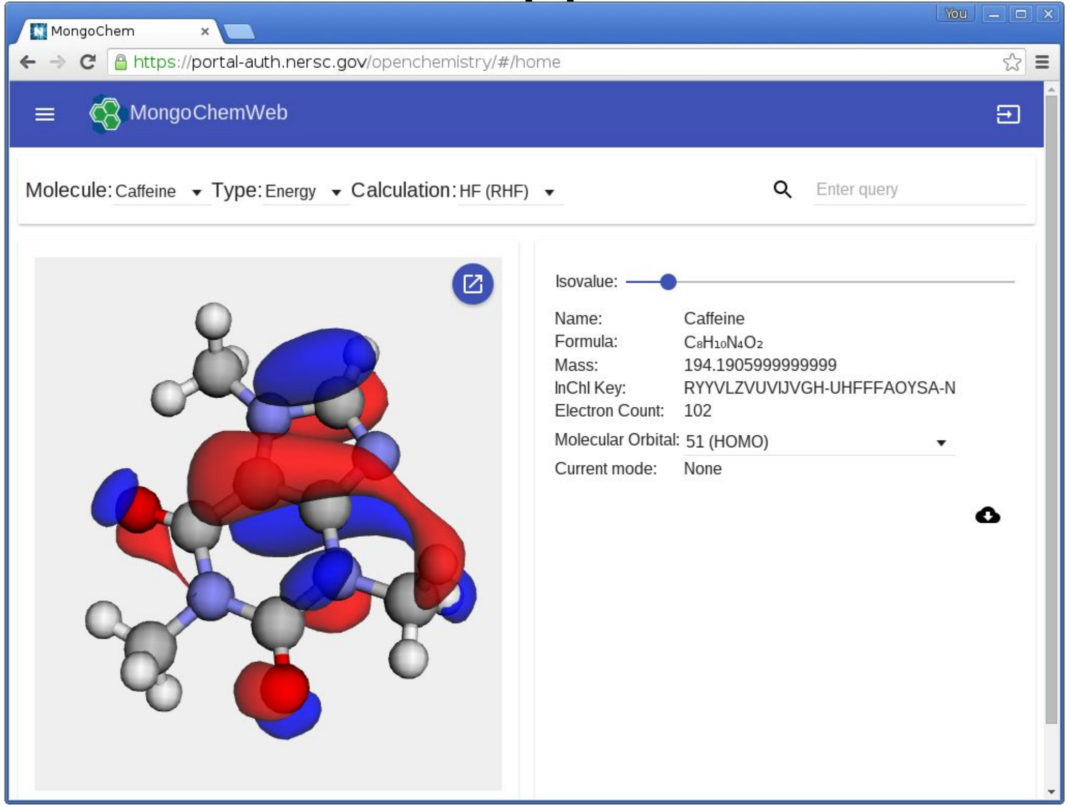
Molecule	Information
	**3-[8,12,17-tris(2-carboxyethyl)-3,7,13,18-tetramethyl-5,10,15,20,21,22,23,24-octahydroporphin-2-yl]propionic acid** **Formula:** $C_{36}H_{44}N_4O_8$ **Mass:** 660.75656 **InChi:** InChI=1S/C36H44N4O8/c1-17-21(5-9-33(41)42)29-14-27-19 **InChiKey:** NIUVHXTXUXOFEB-UHFFFAOYSA-N Fullscreen 3D
	**9-(6-ketooctyl)-6-(1-methoxyindol-3-yl)-3-sec-butyl-1,4,7,10-tetrazabicyclo[10.4.0]hexadecane-2,5,8,11-diquinone** **Formula:** $C_{33}H_{47}N_5O_6$ **Mass:** 609.75618 **InChi:** InChI=1S/C33H47N5O6/c1-5-21(3)28-33(43)37-19-13-12-18 **InChiKey:** XWKJTSOFFKCRMH-UHFFFAOYSA-N Fullscreen 3D

VTKWeb and MongoChemWeb



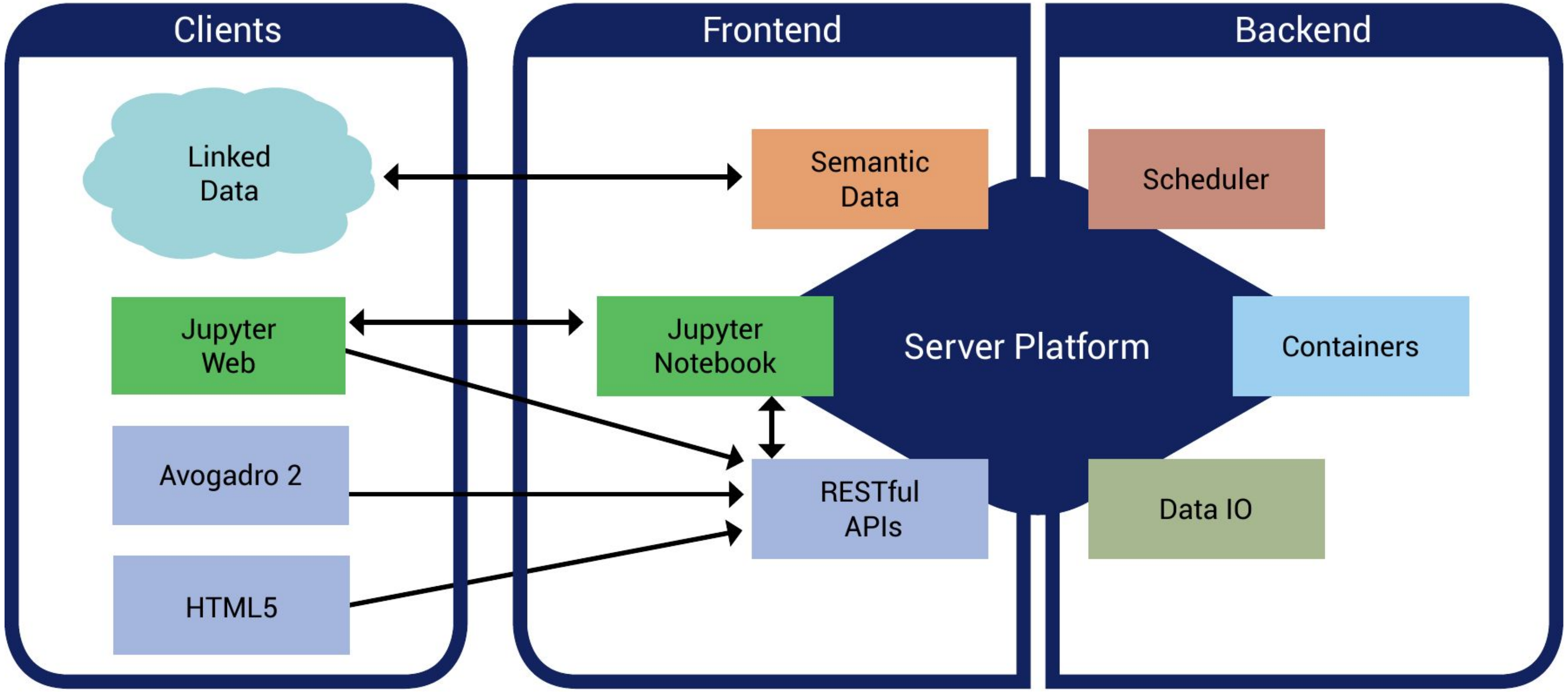
InChIKey	IZRSVHMKZWKSSV-UHFFFAOYSA-N
SMILES	c1cc2ccc3c4ccc(cc4[se]c3c2o1)-c1ccc[se]1
Formula	$C_{18}H_{10}OSe_2$
Molecular Mass	400.196
Energy	Homo -0.112; Lumo -0.002; Gap 0.11; Total -5543.7168293588
Calculation	Theory BP86; Basis STO-6G

AngularJS Modern Web Application



Molecule: Caffeine Type: Energy Calculation: HF (RHF)

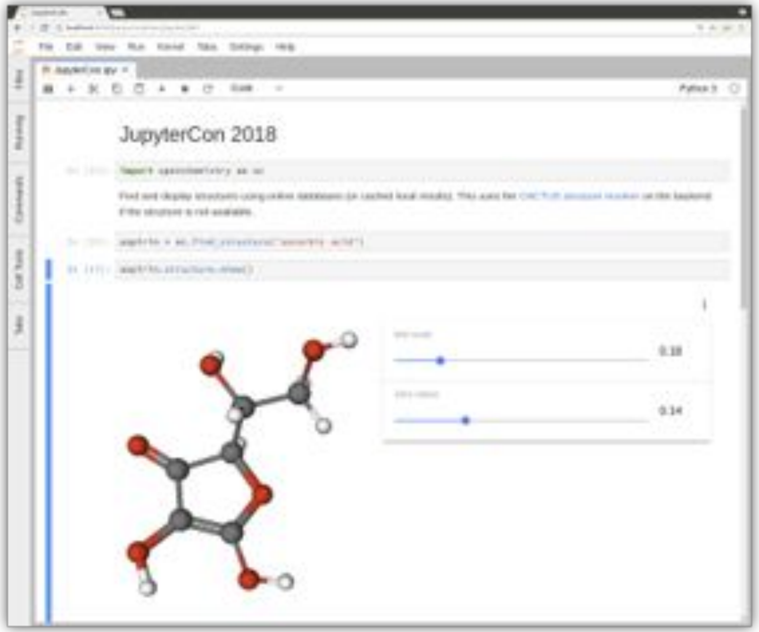
Name:	Caffeine
Formula:	$C_8H_{10}N_4O_2$
Mass:	194.1905999999999
InChI Key:	RYYVLZVUVIJVGH-UHFFFAOYSA-N
Electron Count:	102
Molecular Orbital:	51 (HOMO)
Current mode:	None


Clients
Frontend
Backend
Linked Data
Semantic Data
Scheduler
Jupyter Web
Jupyter Notebook
Server Platform
Containers
Avogadro 2
RESTful APIs
Data IO
HTML5

Kitware

Why Jupyter?

- Supports interactive analysis while preserving the analytic steps
    - Preserves much of the provenance
- Familiar environment and language
    - Many are already familiar with the environment
    - Python is the language of scientific computing
- Simple extension mechanism
    - Particularly with JupyterLab
    - Allows for complex domain specific visualization
- Vibrant ecosystem and community



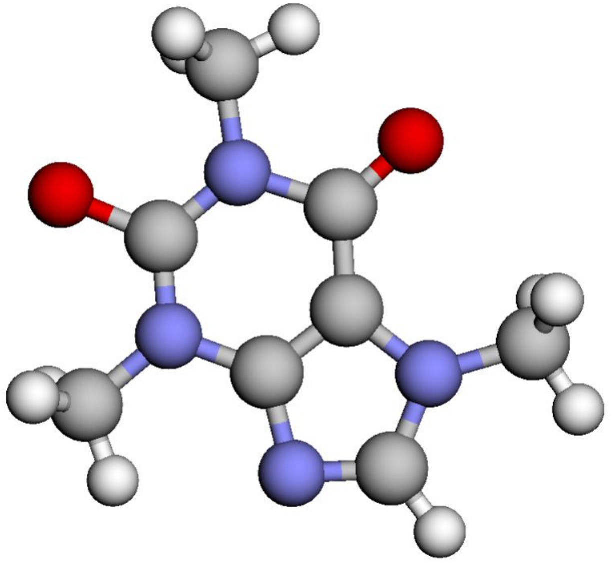
JupyterCon 2018

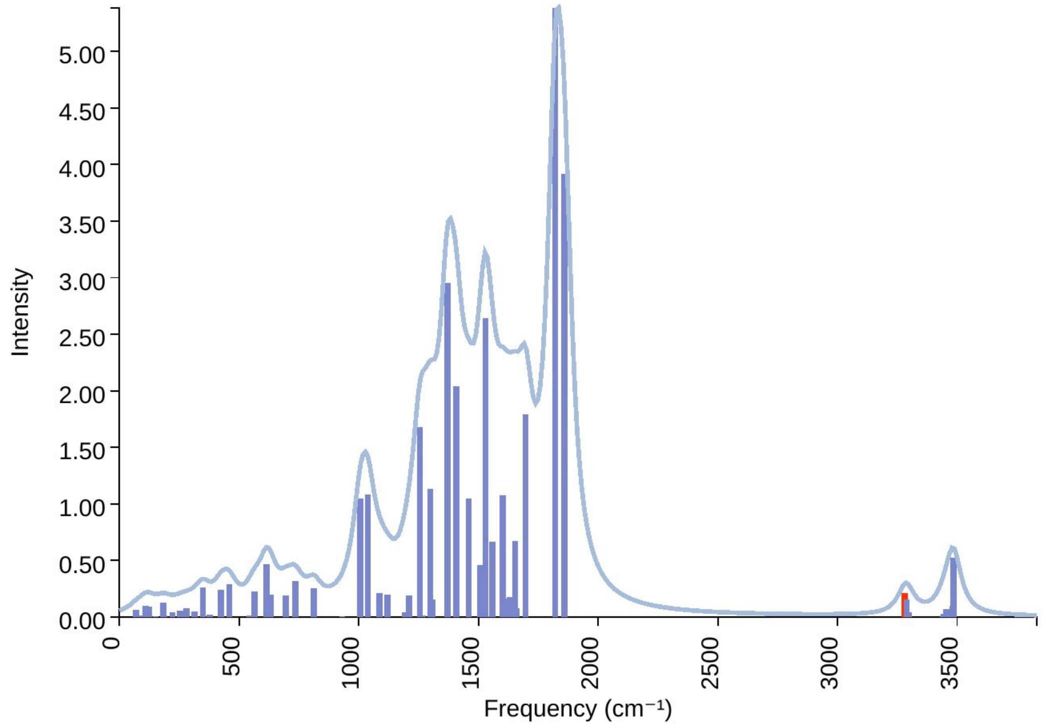
```
import openchemistry as oc
```

[illegible]

Data, Python, Jupyter, Chemistry

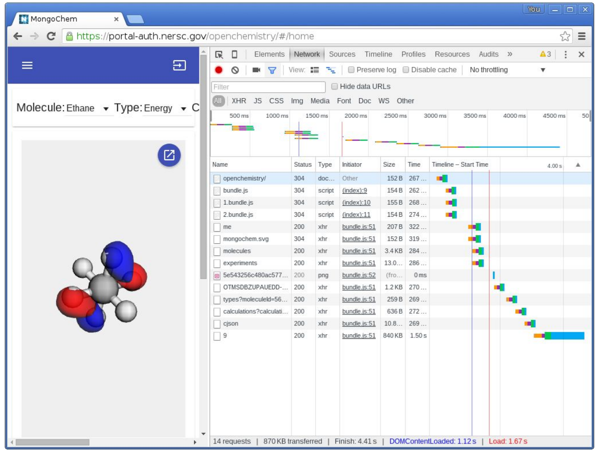
```
[35]: result.frequencies.show(mode=56, animate_modes=True)
```





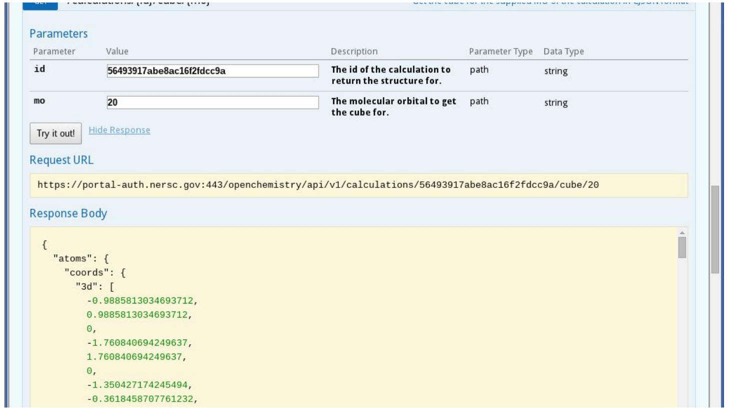
Approach and Philosophy

- Data is the core of the platform
  - Start with a simple but powerful date model and data server
- RESTful APIs are ubiquitous
  - Use from notebooks, apps, command line, desktop, etc
- Jupyter notebooks for interactive analysis
  - High level domain specific Python API within the notebooks
- Web application
  - Authentication, access control, management tasks
  - Launching, searching, managing notebooks
  - Interact with data outside of the notebook



Name	Status	Type	Initiator	Size	Time	Timeline – Start Time
openchemistry/	304	doc…	Other	152 B	267 …	
bundle.js	304	script	(index):9	154 B	262 …	
1.bundle.js	304	script	(index):10	155 B	268 …	
2.bundle.js	304	script	(index):11	154 B	274 …	
me	200	xhr	bundle.js:51	207 B	322 …	
mongochem.svg	304	xhr	bundle.js:51	152 B	319 …	
molecules	200	xhr	bundle.js:51	3.4 KB	284 …	
experiments	200	xhr	bundle.js:51	13.0…	286 …	
5e543256c480ac577…	200	png	bundle.js:52	(fro…	0 ms	
OTMSDBZUPAUEDD-…	200	xhr	bundle.js:51	1.2 KB	270 …	
types?moleculeId=56…	200	xhr	bundle.js:51	259 B	269 …	
calculations?calculati…	200	xhr	bundle.js:51	636 B	272 …	
cjson	200	xhr	bundle.js:51	10.8…	269 …	
9	200	xhr	bundle.js:51	840 KB	1.50 s	

14 requests | 870 KB transferred | Finish: 4.41 s | DOMContentLoaded: 1.12 s | Load: 1.67 s



Parameters

Parameter	Value	Description	Parameter Type	Data Type
id	56493917abe8ac16f2fdcc9a	The id of the calculation to return the structure for.	path	string
mo	20	The molecular orbital to get the cube for.	path	string

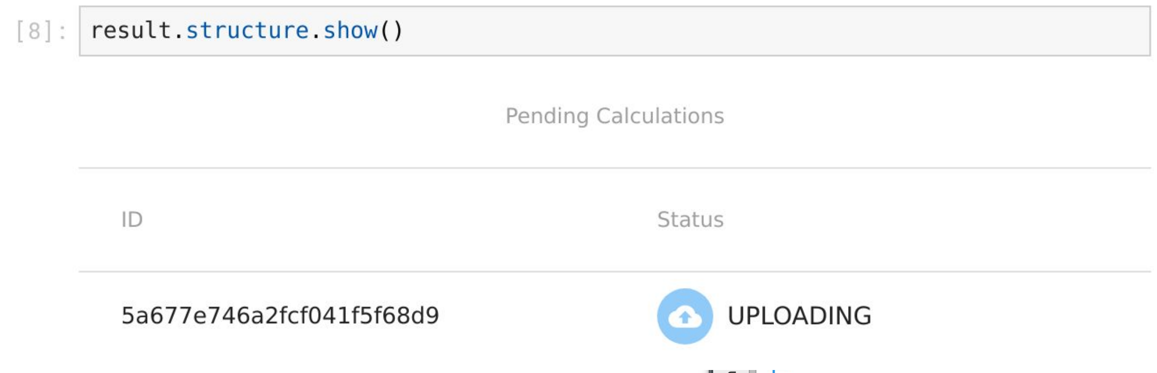
Request URL

```
https://portal-auth.nersc.gov:443/openchemistry/api/v1/calculations/56493917abe8ac16f2fdcc9a/cube/20
```

Response Body

```
{
  "atoms": {
    "coords": {
      "3d": [
        -0.9885813034693712,
        0.9885813034693712,
        0,
        -1.760840694249637,
        1.760840694249637,
        0,
        -1.350427174245494,
        -0.3618458707761232,
```

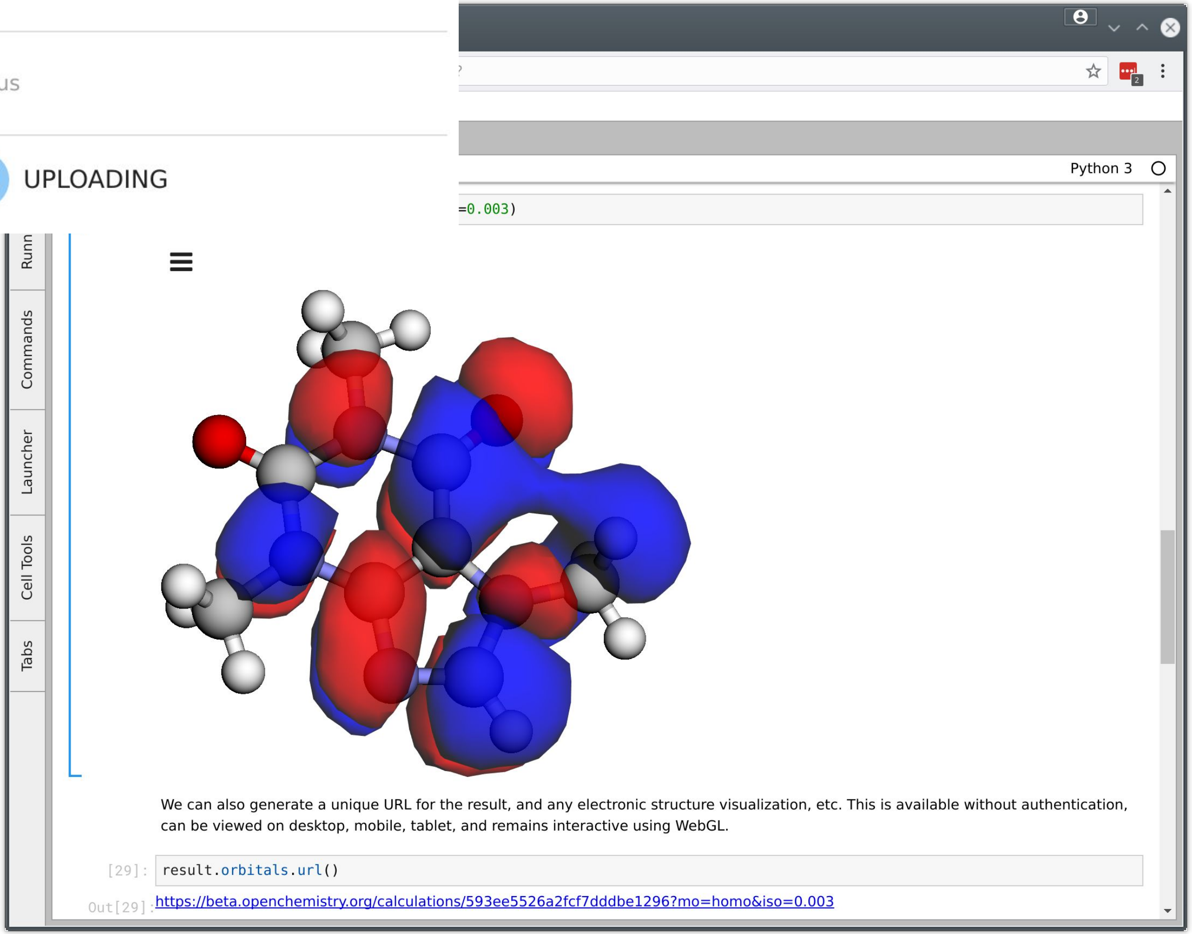
Kitware



```
[8]: result.structure.show()
```

Pending Calculations

ID	Status
5a677e746a2fcf041f5f68d9	UPLOADING

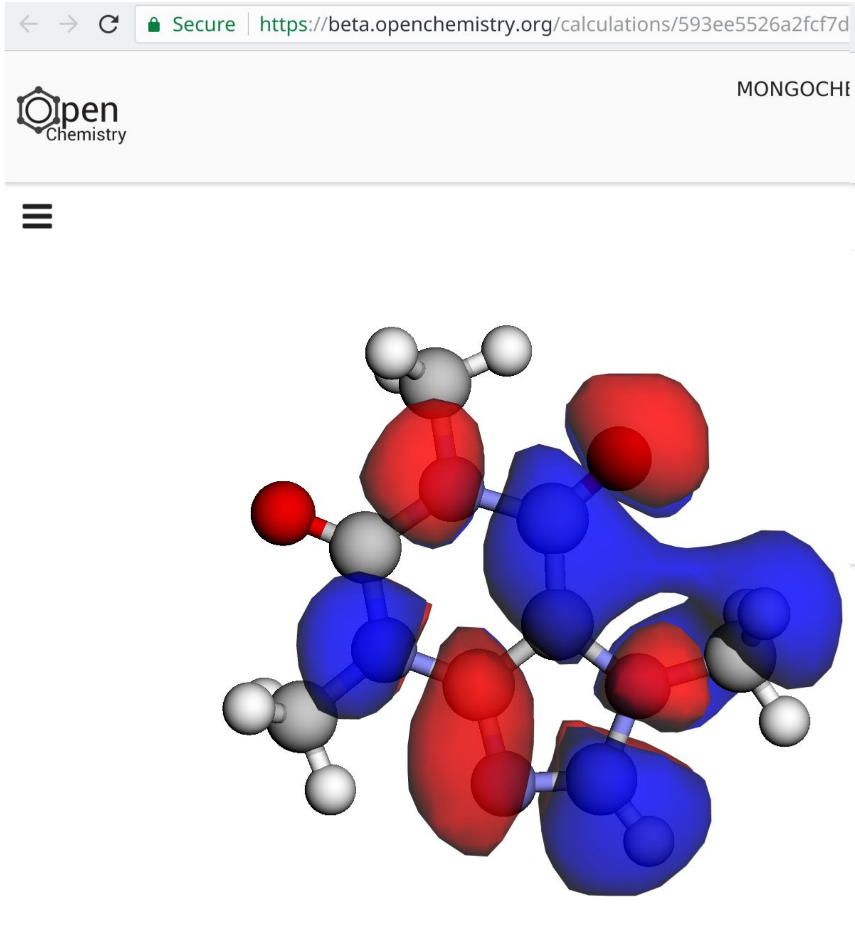


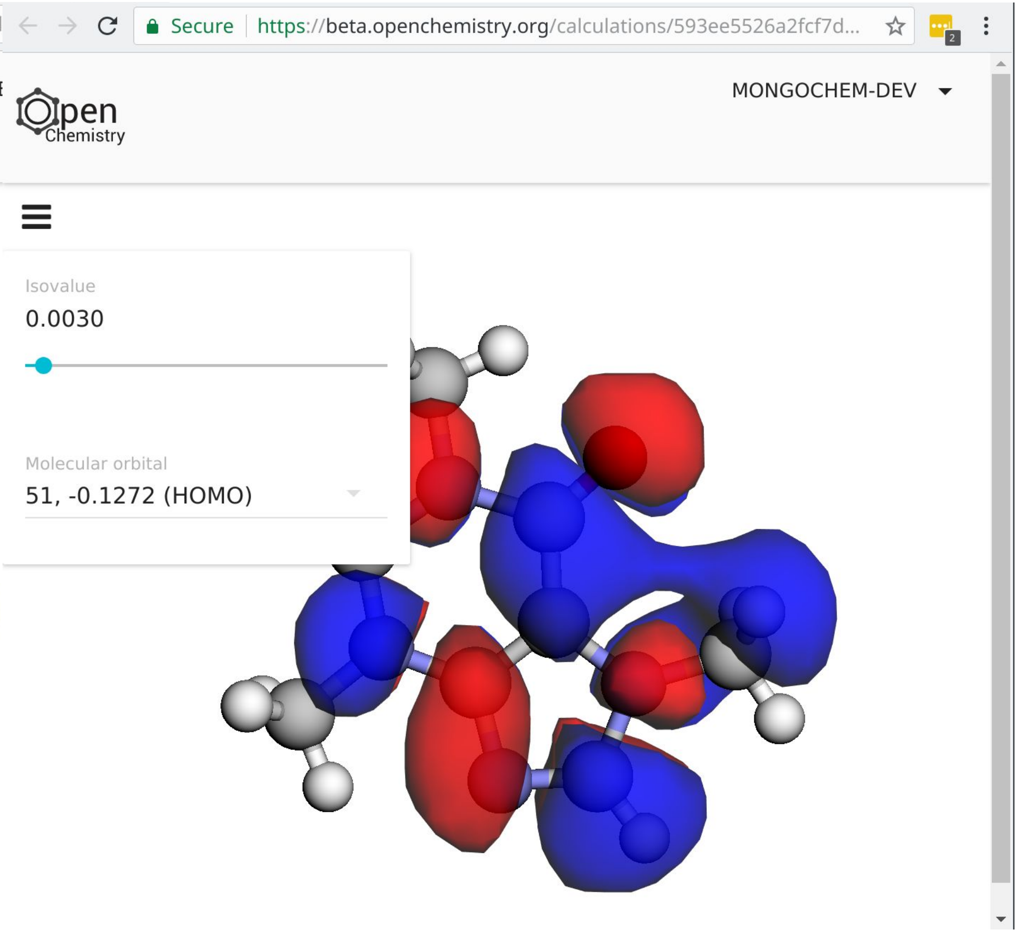
```
=0.003)
```

We can also generate a unique URL for the result, and any electronic structure visualization, etc. This is available without authentication, can be viewed on desktop, mobile, tablet, and remains interactive using WebGL.

```
[29]: result.orbitals.url()
Out[29]: https://beta.openchemistry.org/calculations/593ee5526a2fcf7dddbe1296?mo=homo&iso=0.003
```

Kitware


Secure
https://beta.openchemistry.org/calculations/593ee5526a2fcf7d
Open Chemistry
MONGOCHE

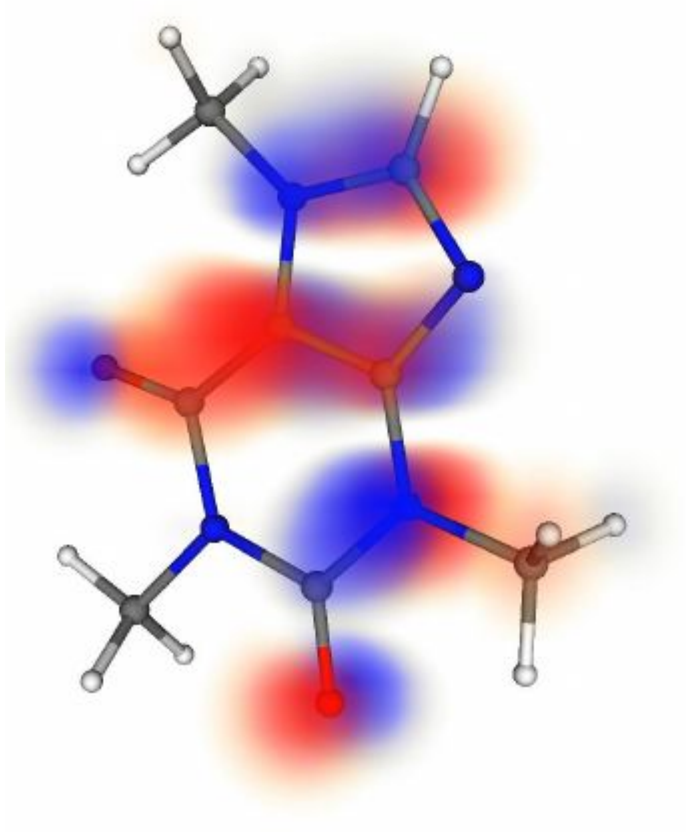

Secure
https://beta.openchemistry.org/calculations/593ee5526a2fcf7d...
Open Chemistry
MONGOCHEM-DEV

Isovalue
0.0030

Molecular orbital
51, -0.1272 (HOMO)

Deployment

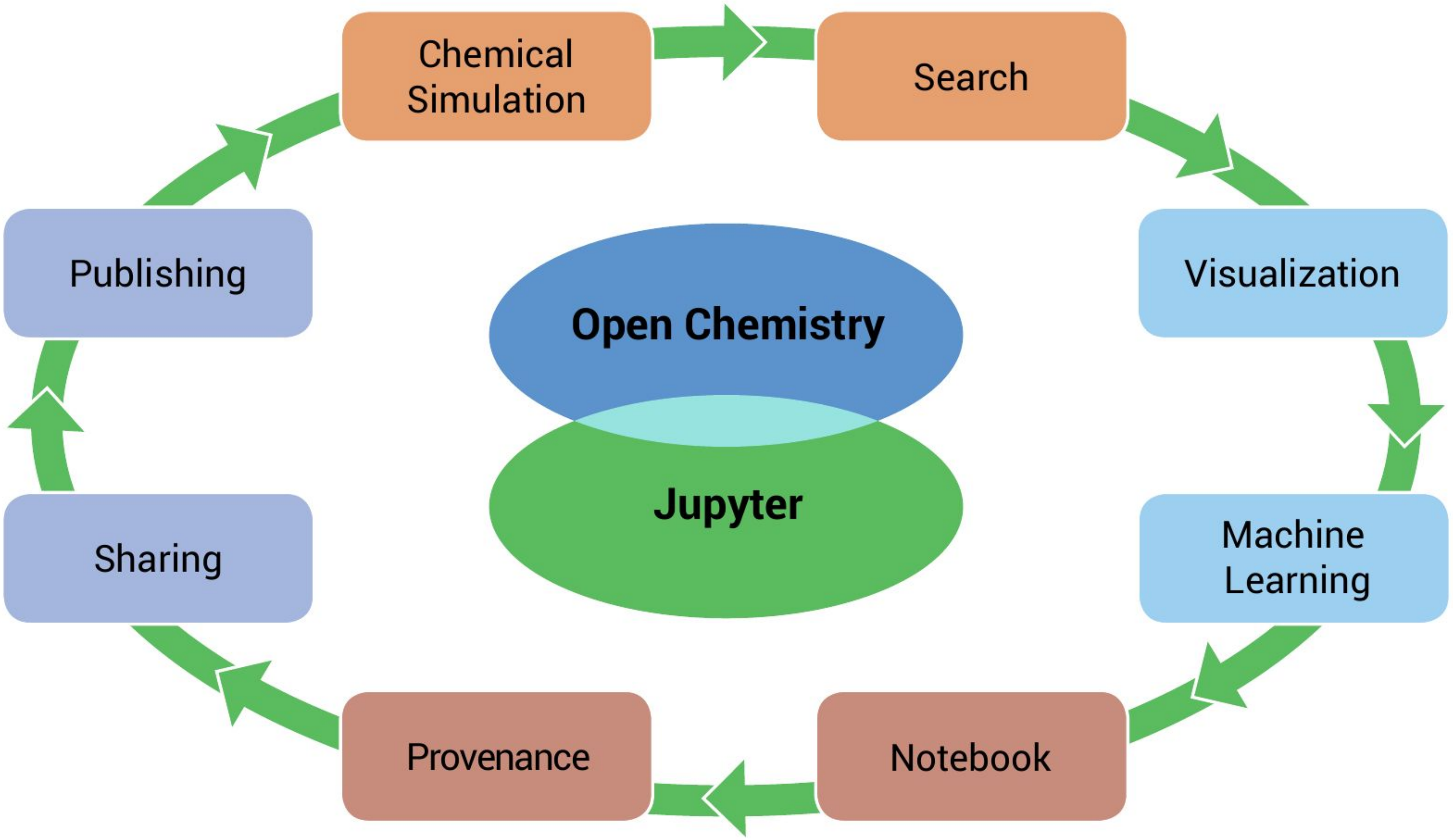
- Docker containers for the components
- Use docker-compose for coordination
- Ansible for runtime configuration
- AWS deployment
  - Running jobs on a small cluster
- NERSC deployment
  - Uses NERSC login credentials
  - Jobs run on Cori nodes
  - SBIR allocation to support development
- Development deployments are being improved



Reproducibility for Chemical-Physics Data

- Dream - share results like we can currently share code
- Links to interactive pages displaying data
- Those pages link to workflows/Jupyter notebooks
- From input geometry/molecule through to final figure
- Docker containers offer known, reproducible binary
  - Metadata has input parameters, container ID, etc
- Aid reproducibility, machine learning, and education
- Federate access, offer full worked examples - editable!

Kitware


Chemical Simulation
Search
Visualization
Machine Learning
Notebook
Provenance
Sharing
Publishing
Open Chemistry
Jupyter

Pillars of Phase II SBIR Project

1. Data and metadata
    - JSON, JSON-LD, HDF5 and semantic web
2. Server platform
    - RESTful APIs, computational chemistry, data, machine learning, HPC/cloud, and triple store
3. Jupyter integration
    - Computational chemistry, data, machine learning, query, analytics, and data visualization
4. Web application
    - Management interfaces, single-page interface, notebook/data browser, and search
5. Avogadro and local Python
    - Python shell integration, extension of Avogadro to use server interface, editing data on server

Regular automated software deployments, releases with Docker containers

Kitware

Moving to the Modern Web

- Aggressively target HTML5, client-side rendering, asynchronous app
    - React-based web widgets, client side state, asynchronous calls, websockets, JSON, etc
    - Client side WebGL JavaScript based rendering for data with interactivity
    - Client side D3 charting with interactivity and linking
- Modern data server using Python as a basis coupled with microservices
    - Ansible orchestration of deployment, Docker containers, microservices
    - No HTML generated by the server - RESTful APIs, JSON, data endpoints
    - Static web assets downloaded by clients using “assembled” JavaScript bundles
- Multiple frontends using language agnostic endpoints
    - Jupyter, modern web application, Python, and enabling app integration, i.e. Avogadro
- Open, extensible, modular, modern, engineered architecture
    - Reusing C++, Fortran, C etc on server side using containers, schedulers, data-centric

Kitware

Collaboration and Community

- Phase II project partners/direct subcontracts
    - Bert de Jong at Berkeley Lab - links to NWChem, diverse projects with data/viz needs
    - Johannes Hachmann at SUNY Buffalo - focused on machine learning, chemical libraries
- Deployments on diverse infrastructure
    - NERSC science gateway, Amazon EC2, soon university deployments
    - NERSC requires NIM account, Amazon much more open, university likely more limited
- Engaging MolSSI, already sponsored a workshop last year on JSON schema
- Building a community around user interfaces, reproducibility, data
    - Engage with the academic, lab-based and industrial communities
- Open framework with licensing friendly to both open and proprietary codes
    - Commercialization approaches customizing to new codes, environments, compute cluster, etc

Closing Thoughts

- Avogadro 2 already has a number of exciting new features
- What are the pain points preventing us from releasing 2.0?
- Successfully reusing components of Avogadro Libraries server-side
- Integration of Avogadro 2 with the Open Chemistry Jupyter platform coming
- Volume rendering using VTK is working in Avogadro 2
  - Developing new API to support “active objects” to enable more dynamic extensions
  - Number of technical issues had to be solved
- Eleven years of Avogadro, open source, and data
- Position our open source tools for the future
- Embrace Python in more diverse ways to increase engagement
- This open source thing might just win out, despite the resistance to it ;-)