

**OCTOBER 22, 2020**



# **POROUS MEDIA VISUALIZATION**

**Mini course based on Digital Rocks Portal data**

---

**DR. MAŠA PRODANOVIĆ AND DR. JAMES MCCLURE**

**THE UNIVERSITY OF TEXAS AT AUSTIN AND VIRGINIA TECH**

# Thanks to Sponsors

 <p><b>SOUTHBDHUB</b></p> <p>Main sponsor: <u>South Big Data Innovation Hub</u></p>	 <p><b>ORS</b>   OBJECT RESEARCH SYSTEMS</p> <p>“Enchanted Rock” sponsor: <u>Object Research Systems</u></p>	 <p>“Town Mountain Granite” sponsor: <u>Kitware</u></p>	 <p>“Austin Chalk” sponsor: <u>Dassault Systèmes</u></p>
---	---	---	---

DragonFly  
software (demo  
today)

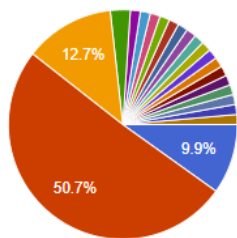
ParaView  
software (demo  
today)

- **10 - 10:10am** Welcome
- **10:10 - 10:50am** Introduction to 2D images
- **11:00 - 11:50am** Introduction to 3D images: slicing the volume and surface visualization
- **12 - 12:15pm** Sponsor demo: Dragonfly software
- **12:25-12:40pm** Sponsor demo: ParaView software
- **12:40-1pm**: Questions

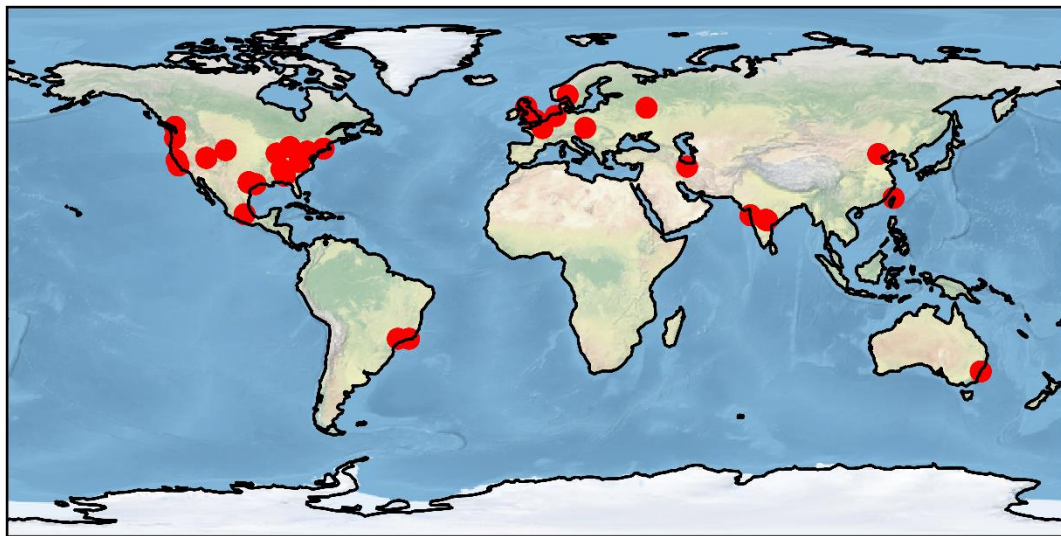
# Rules

- The course is open to anyone who registers and will be recorded (and available on YouTube), and the relevant code posted on GitHub, and both will be linked to Digital Rocks Portal webpage.
- \* We will do demos in Jupyter Notebook, if you want to execute them as we go along, please install Anaconda with Python 3 on your computer: <https://www.anaconda.com/products/individual>. Anyone should be able to execute the code we provide; understanding the details requires basic knowledge of Python and NumPy, however.
- \* The course will cover the basics of relevant 2D and 3D image formats and visualization basics. We will set you up with Jupyter Notebook and Python notebooks that instruct you how to directly download or resample data from Digital Rocks Portal.
- \* 3D visualization software solutions vary and are hardware-intense,. We will demo built-in Python solutions as well as MayaVi (part of Anaconda, alas it currently requires Python 3.7 environment in Anaconda), and sponsor presentations will demo DragonFly and ParaView: you will thus be able to choose.
- \* Please understand that the number of participants will be too large to troubleshoot individual installation problems on the fly beyond some basic questions.

# The audience today is diverse



\*PhD student is a graduate student in USA



Thanks to Javier E. Santos for map!

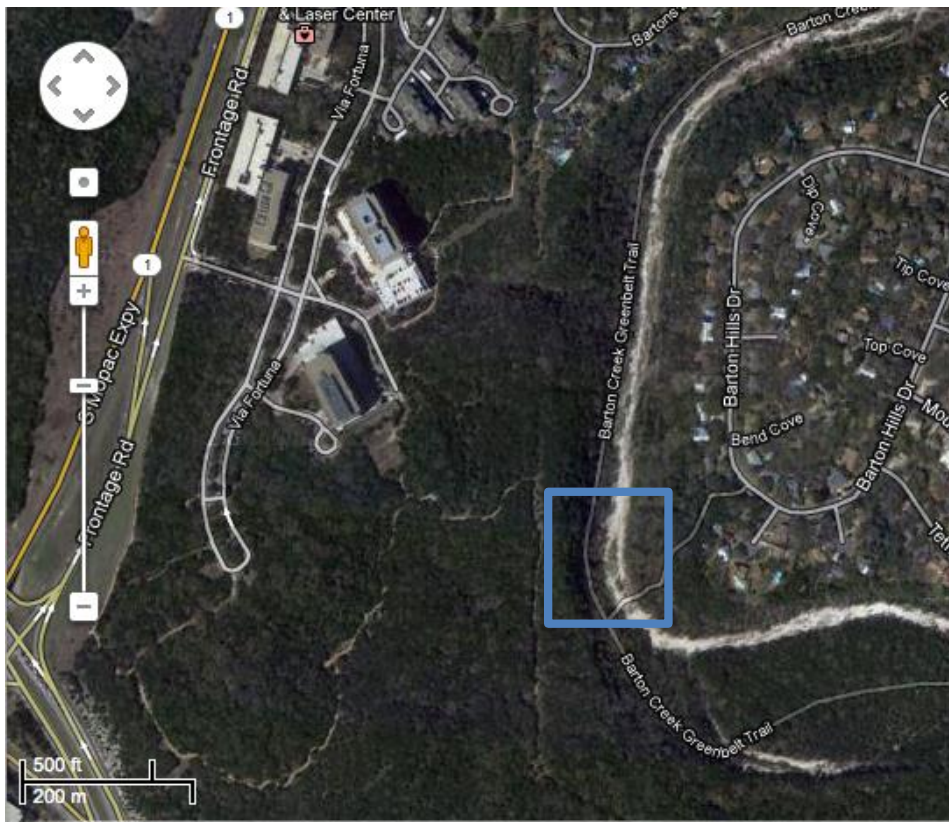


# Quick Poll

- Identify the level of knowledge you have about porous media images (of rocks, soils, etc):
  1. None
  2. Entry level: I encountered on a few occasions
  3. Medium level: I regularly use them but there is space to learn
  4. Expert user of porous media images

# **ROCKS AND FLUIDS IN SUBSURFACE: BRIEF INTRO**

# Barton Creek Greenbelt in Austin...zoom in!





# Barton Creek Greenbelt...zoom in!

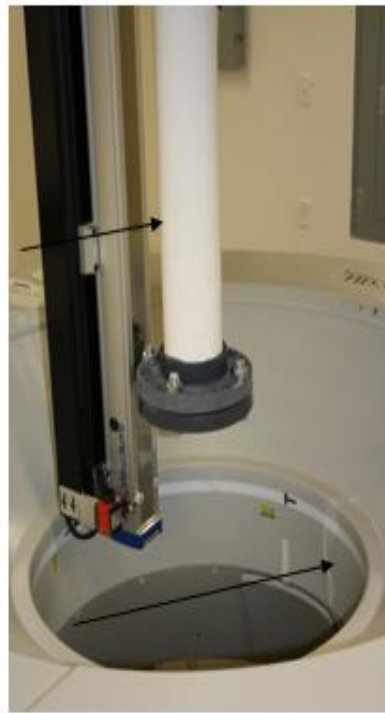
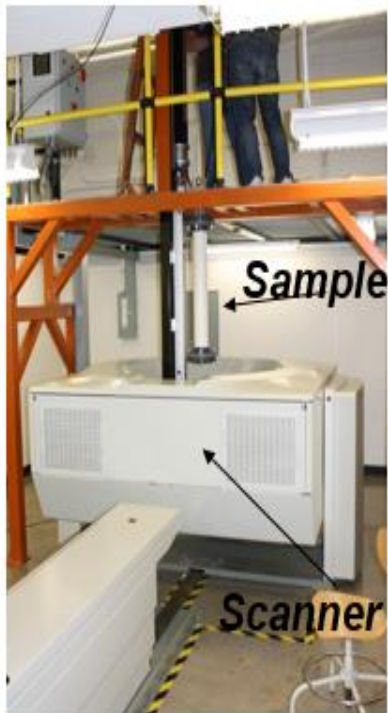


Photo  
Schuyler Nelson-Brown

# Keep zoomin' and do some cuttin'

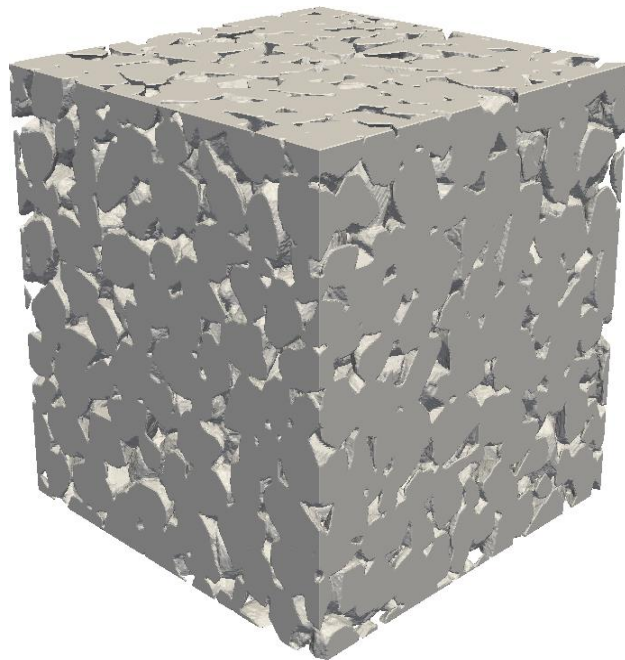
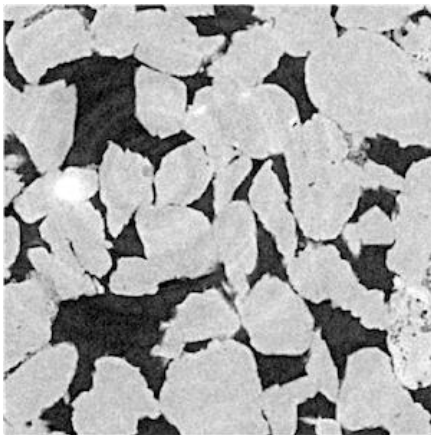
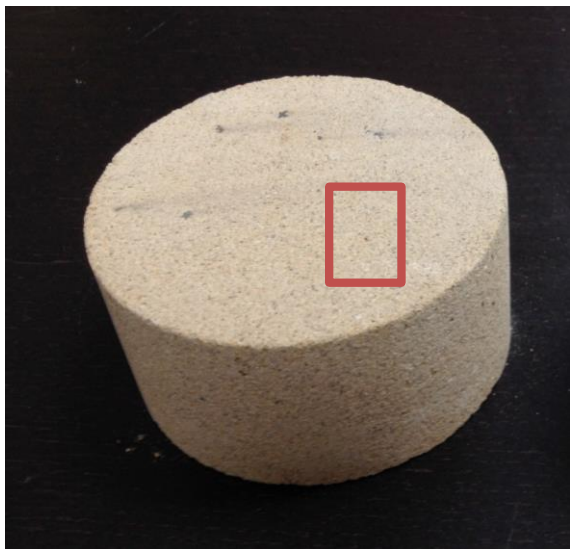


# Need X-ray vision to keep zoomin'



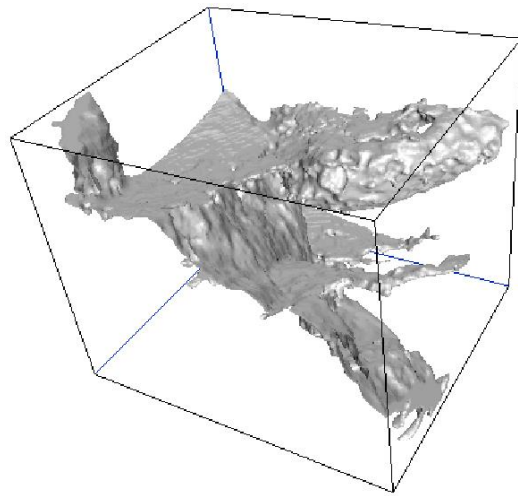
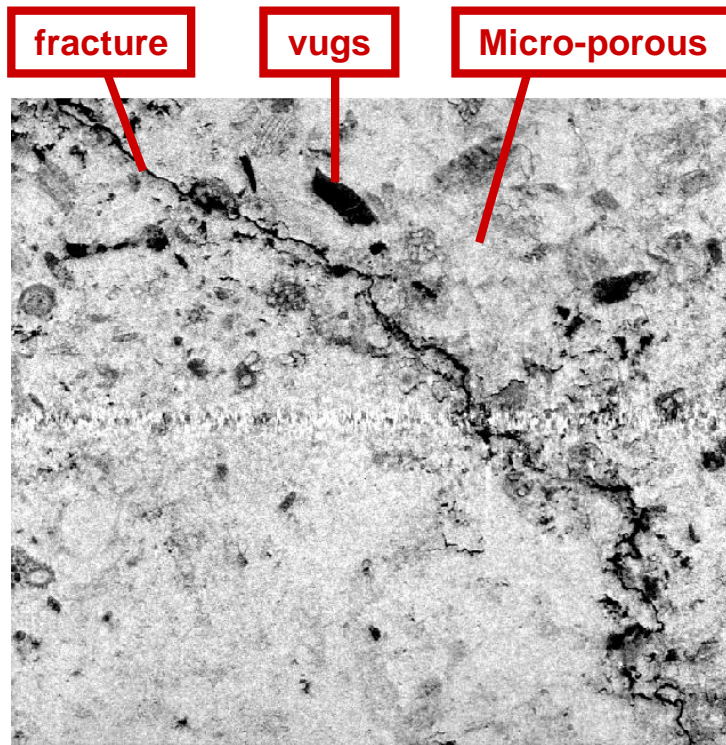
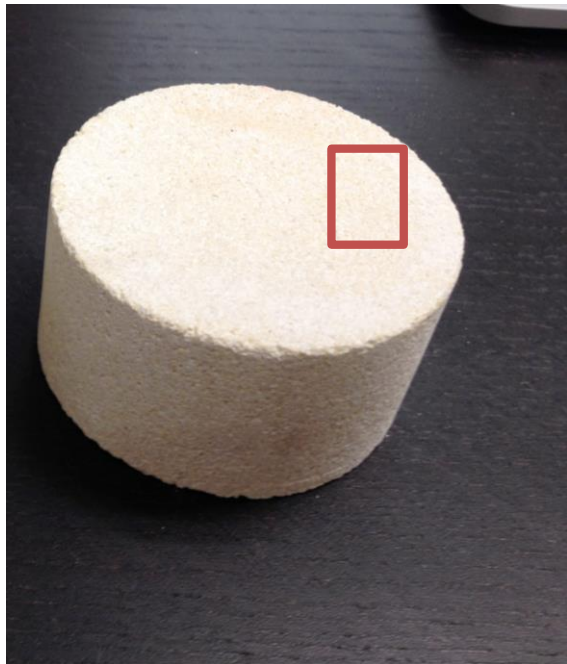
CT Scanner  
PGE Department  
Basement  
Prof. DiCarlo

# What's in a sandstone?

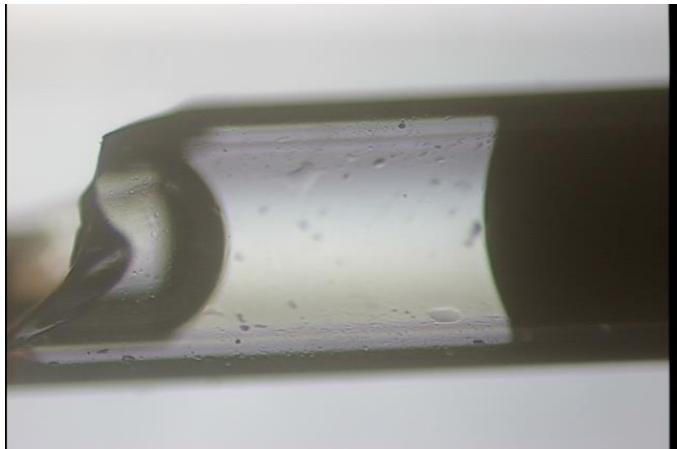
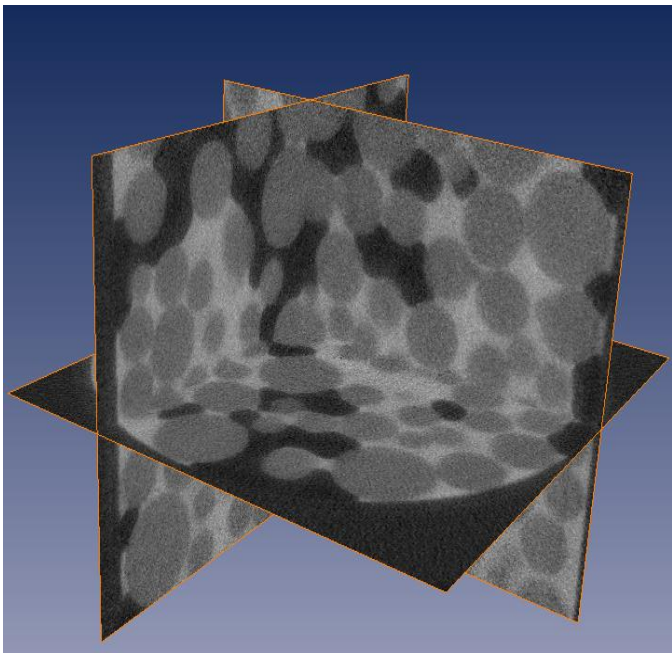




# What's in a limestone?

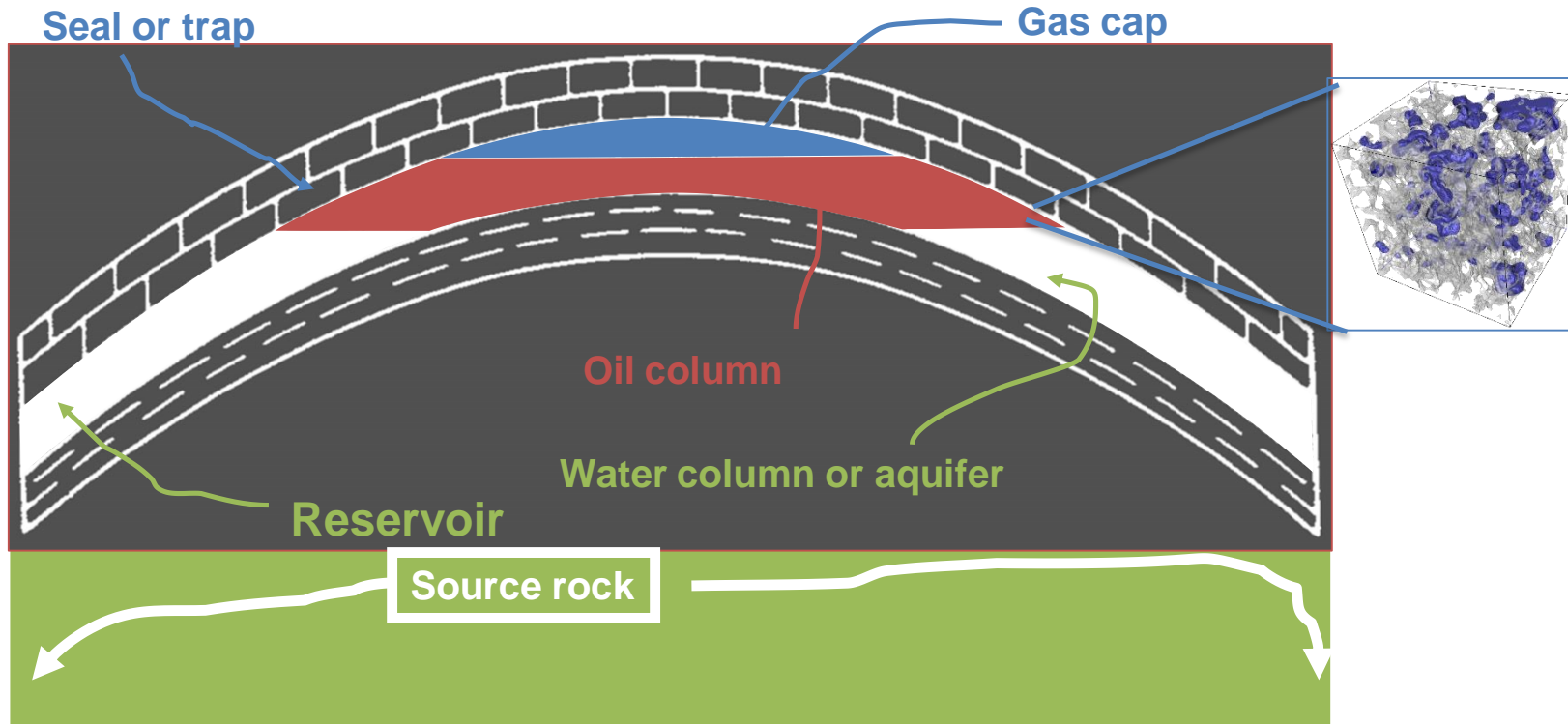


# Just add water, oil and gas within pores





# Reservoir (way) under your feet



# Cool rocks, but not a reservoir

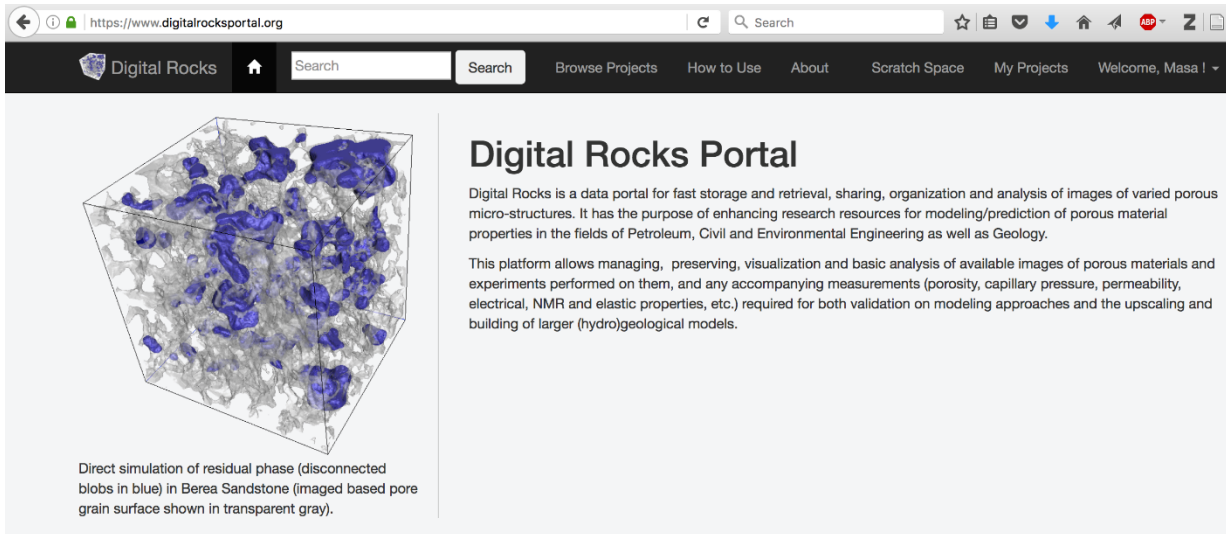


Grand Canyon

<http://www.nps.gov/archive/grca/photos/>

# DIGITAL ROCKS PORTAL

[DRP: https://www.digitalrockportal.org](https://www.digitalrockportal.org)



The screenshot shows the Digital Rocks Portal homepage. At the top is a navigation bar with a search bar and links for Browse Projects, How to Use, About, Scratch Space, My Projects, and a welcome message. The main content area features a 3D visualization of a porous material structure with blue residual phase blobs. To the right of the image is the title "Digital Rocks Portal" and a description of the platform's purpose and capabilities. Below the main content are two sections: "Browse Published Projects" and "Upload and Publish Data", each with a brief description and a button to access the respective functionality.

**Digital Rocks Portal**

Digital Rocks is a data portal for fast storage and retrieval, sharing, organization and analysis of images of varied porous micro-structures. It has the purpose of enhancing research resources for modeling/prediction of porous material properties in the fields of Petroleum, Civil and Environmental Engineering as well as Geology.

This platform allows managing, preserving, visualization and basic analysis of available images of porous materials and experiments performed on them, and any accompanying measurements (porosity, capillary pressure, permeability, electrical, NMR and elastic properties, etc.) required for both validation on modeling approaches and the upscaling and building of larger (hydro)geological models.

Direct simulation of residual phase (disconnected blobs in blue) in Berea Sandstone (imaged based pore grain surface shown in transparent gray).

**Browse Published Projects**

Research public datasets that are hosted on Digital Rocks. You can view, search, and download metadata, raw and derived data, and find publications related to datasets that Digital Rocks users have uploaded and published.

**Upload and Publish Data**

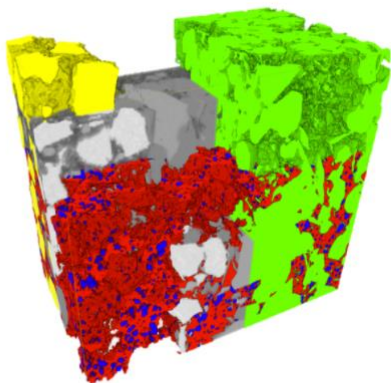
Create a Project and Upload your Data. You can upload originating data, analysis data, and specimen data, as well as publications or other documents relating to your data.

■ **Following**  
[Findable, Accessible, Interoperable, and Reusable data principles](#)

■ **Registered in:**

- [re3data](#),
- [CINERGI](#)
- [COPDESS](#)
- [Geoscience Data Journal](#)

# 117 projects & 654 users: example new additions



## Vaca Muerta FIB-SEM

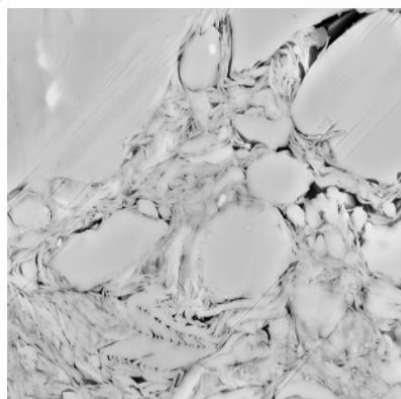
Matthew Andrew  
Carl Zeiss X-ray Microscopy

[View Project](#)

3/2019

Datasets: 2

Downloads: 198



## High-Resolution SE...

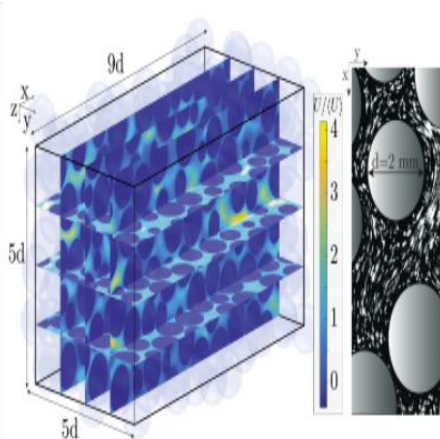
Christopher Landry  
The University of Texas at A...

[View Project](#)

9/2020

Datasets: 5

Downloads: 8



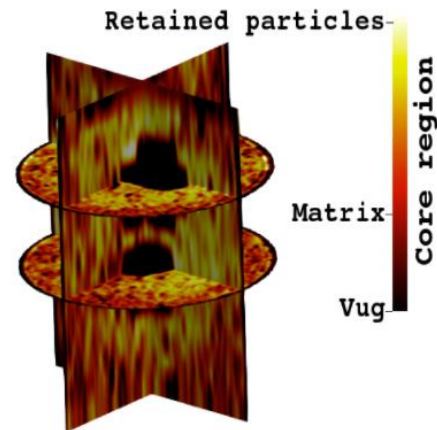
## Experimental 3D Vel...

Mathieu Souzy  
University of Twente

[View Project](#)

Datasets: 6

Downloads: 169



## Particulate straining ...

Hasan Khan  
University of Illinois at Urba...

[View Project](#)

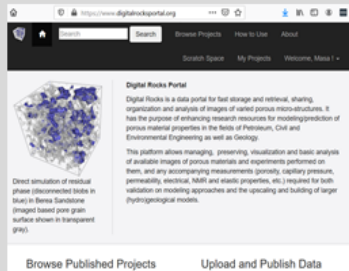
5/2019

Datasets: 34 Downloads: 6577

# Vision: DRP Suite

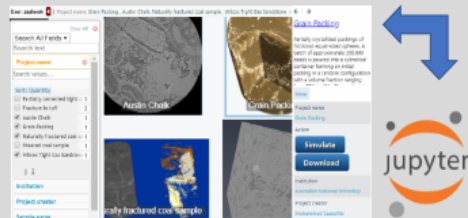
## Digital Rocks Portal (DRP)

- Web-based data repository
- Enable mass upload (**Task 2**)
- Integrate with EarthCube Registries (**Task 2**)



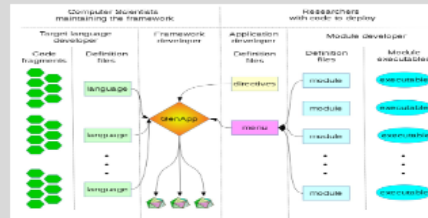
## DRP-Discover (Task 1)

- Find, explore and annotate DRP datasets using SuAVE
- Launch Jupyter workflows for DRP-DB properties
- Manage & send data to DRP-Sim



## DRP-Sim & DRP-Learn (Task 3)

- Build user interfaces
- Manage HPC workflow execution
- Maintain execution history
- Leverage GenApp
- Send output DRP & DRP-DB

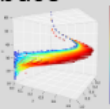


## DRP-Community

- Data reuse, research & discovery
- Sustainability (**Task 5**)
- Education & feedback (**Task 6**)

## DRP-DB (Task 4)

- Computed properties database
- Analyze w. DRP-Discover
- Integrate with DRP





# Digital Rocks Portal updates and newsletter

- Sign up for email updates:
  - <https://digitalrocksportal.us16.list-manage.com/subscribe?u=5062e94dc37c7490120fe65ae&id=92f07372ed>
- Follow us on Twitter: @RocksPortal
  - <https://twitter.com/RocksPortal>

# Virtual machine need

- We can create virtual machine accounts for the purpose of this visualization challenge. If you expect your computer might be too limited and have needs for that then
  - Email: [masha@utexas.edu](mailto:masha@utexas.edu)
  - Subject: Viz Challenge - VM

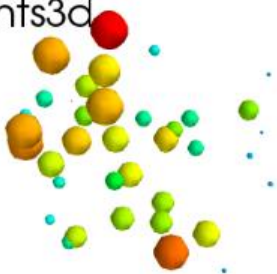
# Data in DRP

- 2D or 3D
- Formats VARY: there is NO ACCEPTED STANDARD
- Geometry is defined by scalar field (image):
  - Grayscale (x-ray, microscopy, MRI) images or
  - Segmented (binary) data where a discrete set of phases (fluids or minerals) have been labeled
- Velocity fields are vector fields (images):
  - There might be three images with x, y, z components

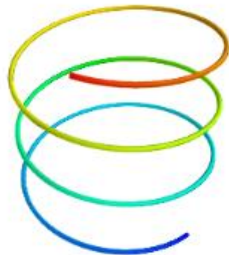
# 3D point cloud plots, line plots or surface plots

- **Just points?** Say I want to plot a number of points in space, and all of them have x,y,z coordinate – I can just place objects (called glyphs) in those locations
- **Lines?** If I want them to be connected by lines, then I need to know the order given by a 1D array).
- **Surface?** Or maybe they are part of a surface (2D object) in which case order is dictated by a 2D grid
- **3D volume?**
- Do I have just data or is the object defined by a function?
- **Position and visibility:** need colors, orientation. and the viewer point of view.

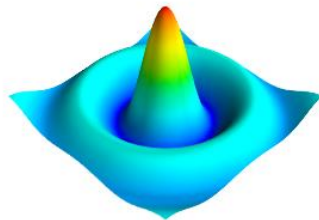
points3d



plot3d



surf

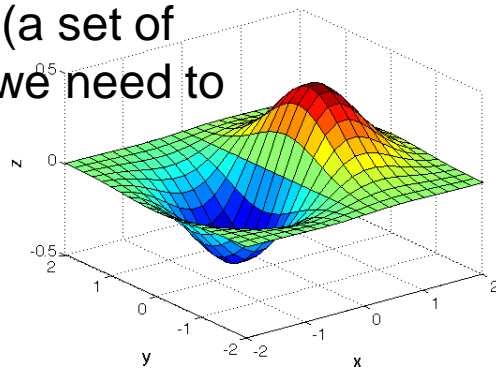


# 3D surface plots

Say I want to plot function of two variables, e.g.  $z(x,y) = -x^*(\exp(-x^{**2}-y^{**2}))$ .

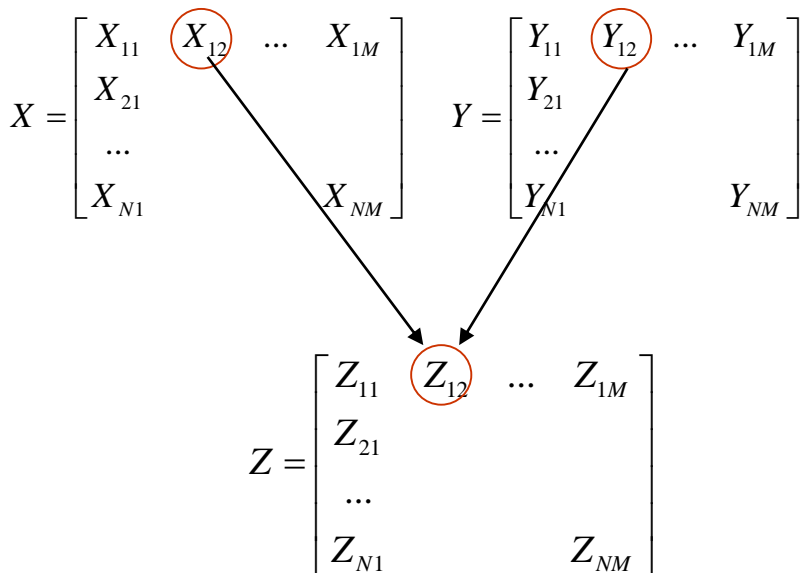
The key issue will be, again, creating a set of points  $(x(i,j), y(i,j), z(i,j))$  that belong to that function/surface in 3D.

For things to look nice, we need to have a nice discretization (a set of points that cover a desired area) in the  $(x,y)$  plane, and then we need to create corresponding  $z$  values for all points.



Use matrices to index locations of points on the surface!

- For x coordinate  $X(i,j)$  and y-coordinate  $Y(i,j)$ ,  $Z(i,j)$  is  $f(X(i,j), Y(i,j))$
- Where  $f$  is some function we would like to plot.
- Values of  $X$  and  $Y$  are usually on some regular grid





## numpy.mgrid creates X,Y matrices on a regular grid

- Again, we plot points, not continuous functions. Thus we need to create matrices of point coordinates.
- If I have a function,  $z=f(x,y)$  to plot I need to create the “Z” matrix at specific “x” and “y” points
- `[X, Y] = np.mgrid[xg1:xgm:xspacing, yg1:ygn:yspacing]` creates matrices X and Y on a regular grid where x and y ranges are provided

vector of regular spaced x-  
data

$$X = \begin{bmatrix} xg_1 & xg_2 & \dots & xg_M \\ xg_1 & & & \\ \dots & & & \\ xg_1 & & & X_M \end{bmatrix}$$

vector of regular spaced y-  
data

$$Y = \begin{bmatrix} yg_1 & yg_1 & \dots & yg_1 \\ yg_2 & & & \\ \dots & & & \\ yg_N & & & yg_N \end{bmatrix}$$

# Mayavi for 3D plotting

- Enthought company (based in Austin) specializes in scientific Python tools and maintains 3D visualization software Mayavi; as a result the two are well integrated.
- Mayavi is linked with Python using specific library is called 'mlab': we can directly visualize NumPy arrays.
- Online resources:
  - <https://docs.enthought.com/mayavi/mayavi/>
  - [http://scipy-lectures.org/packages/3d\\_plotting/index.html](http://scipy-lectures.org/packages/3d_plotting/index.html)

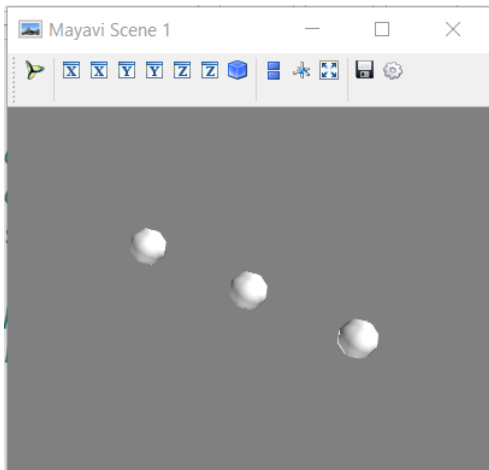
# Installation

- The following commands need to be run as an administrator in Anaconda powershell or Anaconda prompt (on Windows 10, right-click on the prompt to get admin privileges)
  - first update your Anaconda distribution (as that removes issues that could pop up): `'conda update -all'`
  - Install Mayavi: `'conda install mayavi'`
  - If there are issues with versions of Python and Mayavi not being compatible, go to Mayavi website for trouble shooting.
  - Note that all 3D visualizers have a lot of complex dependencies and require good hardware and software: it is common for installations not to work properly.
- **As of October 2020:**
  - **Requires Python 3.7 (create separate environment in Anaconda as necessary)**
  - **Some Mac users have issue with screen showing black color, and no images showing**

# The simplest plot test

```
import numpy as np
import mayavi.mlab as
mlab
```

```
x = np.linspace(1,3,3)
mlab.points3d(x,x,x)
mlab.show()
```



- x coordinates are  $x=[1,2,3]$  (result of that `np.linspace` command), and y, z coordinates are the same for simplicity
- The spheres plotted are called 'glyphs'
- A separate window will open with Mayavi
- You can rotate and interact with that window.

**Omitting `mlab.show()` can cause trouble with the separate window (but not always!). Be sure to include if using mayavi from a different application (Spyder, Jupyter,...)**

# Note: Mayavi is a also standalone visualizer

- If installed, you can run it from Anaconda powershell prompt (or Terminal on Mac) using command 'mayavi2'
- You will note "Python" command window in the lower left corner
- Type in all of the commands from previous slide:

```
import numpy as np
import mayavi.mlab as mlab
```

```
x = np.linspace(1, 3, 3)
mlab.points3d(x, x, x)
mlab.show()
```

- `mlab.show()` is optional, but it does not hurt!
- You can also save these commands in a script and run it.

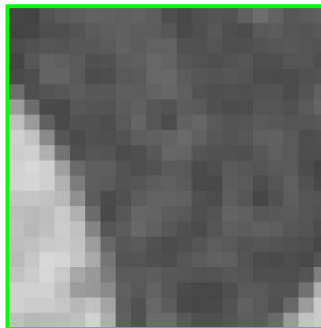
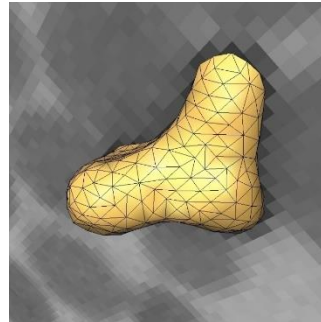
# Refer to Jupyter Notebooks for more examples



# SURFACES IN 3D

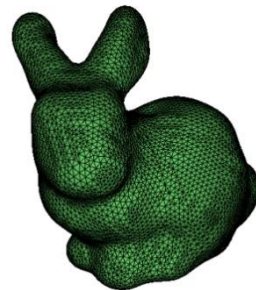
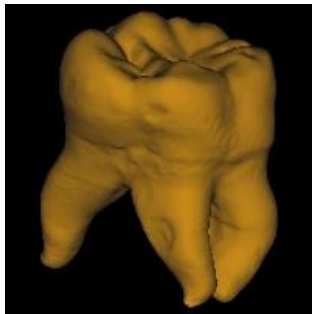
# Surface Generation

- What is an **isosurface**?
  - a surface that represents points of a constant value within a volume of space; in other words, it is a level set of a continuous function whose domain is 3D-space
  - In imaged porous media, we would like to draw a surface (boundary) between two materials
    - Represented by a specific (threshold) grayscale value
    - In segmented images, isosurface 0.5 separates pore (0) from grain (1)
  - In scientific visualization (of either experiments or simulations), isosurface of interest is constant value of pressure, temperature, density....any other field



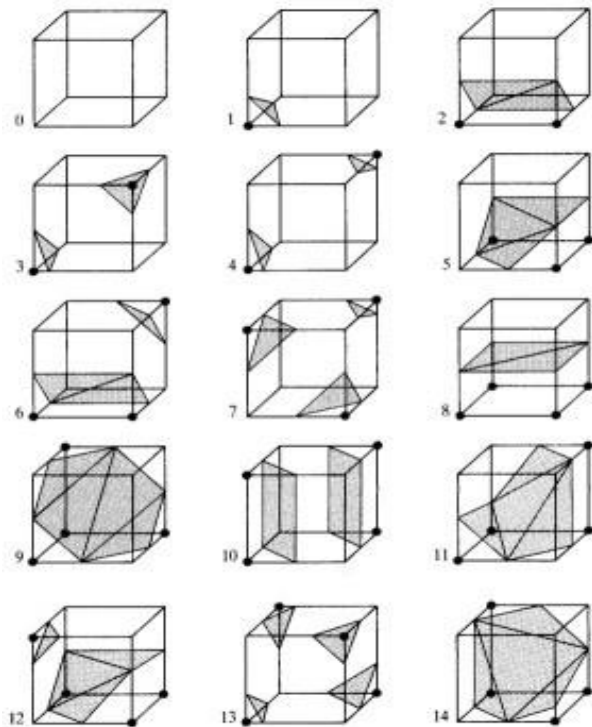
# Surface Generation Algorithms

- Marching cubes algorithm (Lorenson and Cline, 1987)
  - That's pretty much it...
- Adaptive distance gradient filtering (Flin et al., 2005)
  - Adaptive estimation of normals and surface area for discrete 3-D objects: application to snow binary data from X-ray tomography. IEEE Trans. Image Proc. 14(5), 585–596 (2005)
- Porous Media Marching Cubes (McClure et al., 2007)
  - Approximation of interfacial properties in multiphase porous medium systems, **Advances in Water Resources** Volume 30, Issue 3, March 2007, Pages 354-365



# The Marching Cubes Algorithm

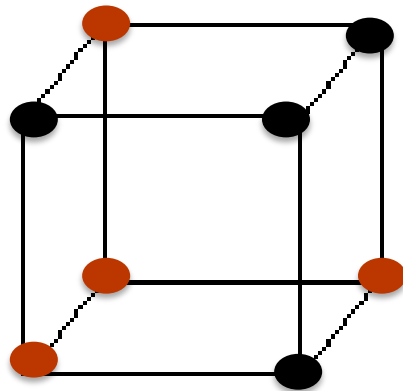
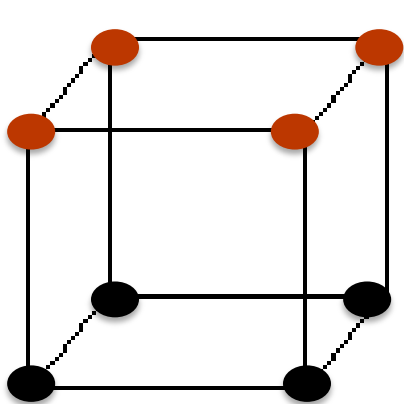
- Introduced by Lorensen and Cline, '87. and implemented in most software
- The algorithm proceeds through the scalar field, taking **eight neighbor locations** at a time (thus forming an imaginary cube), then determining the polygon(s) needed to represent the part of the isosurface that passes through this cube. The individual **polygons are then fused into the desired surface**
- **Original 15 cube configurations ->**
- The MC algorithm **constructs an isosurface** composed of a triangular mesh that corresponds to a chosen isovalue (or threshold)
- Some early-version ambiguities were improved upon in later algorithms (some configurations were identical except for rotation and sign)



W.E. Lorensen and H.E. Cline. Marching cubes: a high resolution 3-D surface construction. *ACM Comput. Graph.* **21** (1987) 163.

# Quiz: marching cubes algorithm

- Draw triangulated surface that separates red and black phases (only voxel centers indicated)



# Exporting surface

- The surface can be Exported from 3D Viewer as an STL (short for stereolithography) file for...
  - visualization and quantification
  - simulation (finite element software can create FEM volume mesh from triangulated surfaces in STL format)
  - 3D printing 😊
- STL files are simply writing down the surface as a set of triangles: sets of vertices and sets triplets of those vertices that define triangles.
- The mesh is likely going to need a clean-up of various inconsistencies before a successful use in modeling: this is a rather hard task: <http://meshrepair.org/>

# Software implementation

- All visualization software has ‘contour’ (in 2D), ‘contour3d’ or ‘isosurface’ (in 3D) routine that invokes creating a surface through data at one or multiple contour or iso “levels”
- If we refer to data in 3D as  $\text{data}[i,j,k]$ , that the triangulated surface output is approximately where  $\text{data} == \text{isolevel}$  based on marching cubes algorithm.

# VISUALIZATION SOFTWARE



# Visualization software

- [MATLAB](#) by Mathworks: Visualization in itself is high end, but is not good for large datasets. Do not forget MATLAB Central exchange which might have a solution to your problem
- [ParaView](#): geared towards **parallel** simulation output, and thus efficient with large datasets; open source but maintained by KitWare
- [VisIt](#): open source, developed for visualizing **parallel** simulation output by Lawrence Livermore National Lab
- [MayaVi](#): recently revived by Austin company Enthought

# Visualization Software (cont'd)

- [Dragonfly](#): free for research
- [Drishti](#): open source
- [PerGeos](#) (not free)
- Python in and of itself is no good at 3D visualization:
  - ParaView and MayaVi readily integrate with Python and Jupyter notebooks; many other have Python scripting capability
  - **PlotLy** and **PyVista** are worth monitoring
- Most of the mentioned software has excellent online resources (demos, videos)

# ImageJ/Fiji

- It was developed originally for 2D microscopy data and processing stacks of 2D slices, and its native visualization kind of operates in terms of manipulating stacks of slices
- It has 3D visualization plugins (BoneJ and 3D Viewer), but in and of itself are not that powerful (Java plugins often bomb) compared to ParaView, for example
- For your research/scientific visualization I suggest:
  - ImageJ for slice views of data
  - ParaView for 3D volumes and surfaces; integrate into Python workflow as necessary

# Hardware issues

- Hardware is important as visualization of 3D Datasets can take A LOT of memory.
- If looking for a personal computer that does visualization well, pay attention to video card, and look for gamer machines.
- Texas Advanced Computing Center provides an [entire Portal](#) to parallel computers that do visualization