Handling Large data in PerGeos

Introduction

We introduce here some techniques to visualize and process large data, mostly targeting the high resolution data acquired by FEI's Heliscan microCT.

The data is considered as "large" in the sense that its size exceeds the size of the GPU memory and/or the size of the RAM of the machine.

The following workflow:

- Segmentation
- Porosity / Connected porosity determination
- Pore separation / Pore Size distribution
- Grain size distribution

is recommended to be applied in memory, if the amount is sufficient (approx. 4 x the size of the dataset, ie 160 GB for a 40GB dataset), with each step independently unloaded from the RAM with the PerGeos *unload from RAM* feature.

If the amount of RAM is not high enough, this workflow can be applied by blocks, with the dedicated SLAB processing modules.

Figure 1 Pore space extracted from a 40 GB carbonate dataset with the slab processing engine

LDM (Large Data Management) and .lda files

Provides:

- Interactive navigation
- Automatic resolution refinement
- Scalable performances
- On-demand data loading

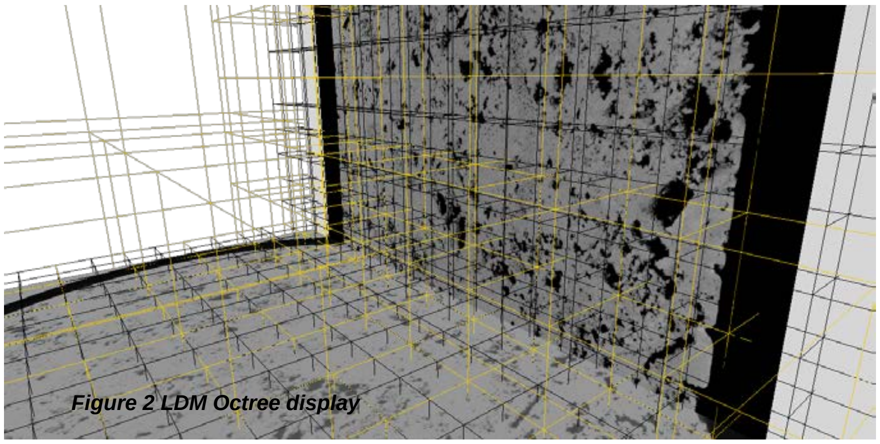


Figure 2 LDM Octree display

The volume data is divided into an octree to allow a real time visualization of the slices and volume rendering. As the GPU has a limited amount of memory, this will prevent the entire volume loading to freeze the machine.

This technology is implicitly used in PerGeos, even if the data is not loaded as an LDM file:

- Data read entirely *in memory*: conversion to LDM on the fly
- Data converted to the LDM format (.lda) (during or prior to loading) and saved on the disk

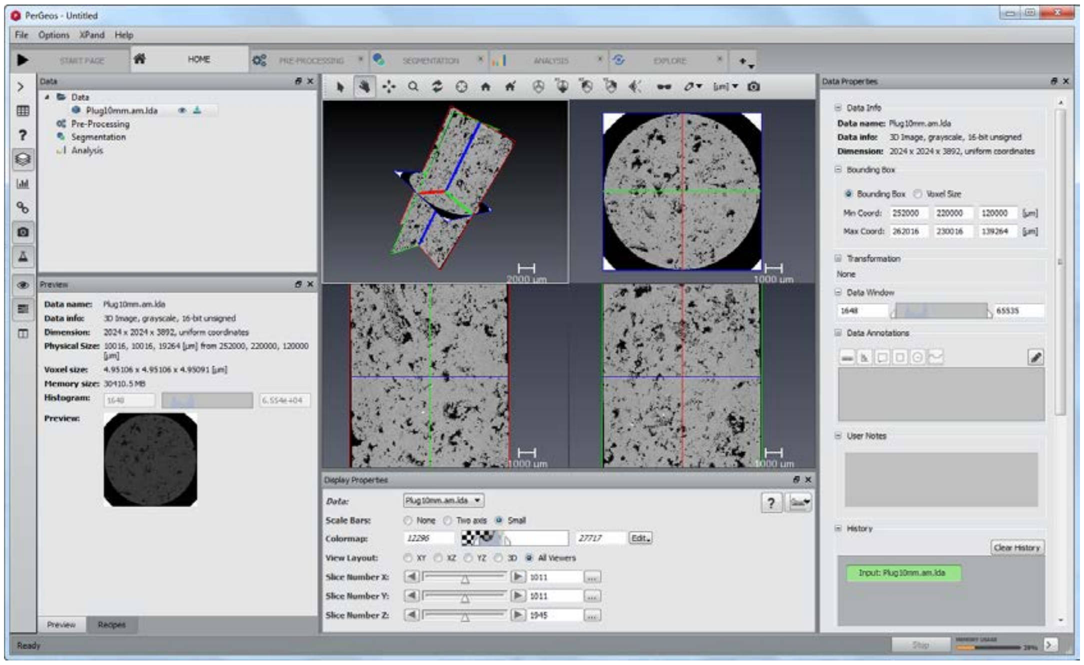
Note that there is no lossy compression of the data, thus it is safe to keep only one LDM reference of the data.

At any time, by using the *ExtractSubvolume* module the data can be loaded entirely in memory and saved as a raw/3Dtiff/am/… file.

Note: no RAM processing of a data loaded as LDM is possible, a volume has to be extracted in memory first, or the processing has to be applied by blocks with the SLAB processing module.

Visualization

All workspaces will implicitly use LDA. It is then possible to visualize slices even if the data is not loaded entirely in memory.



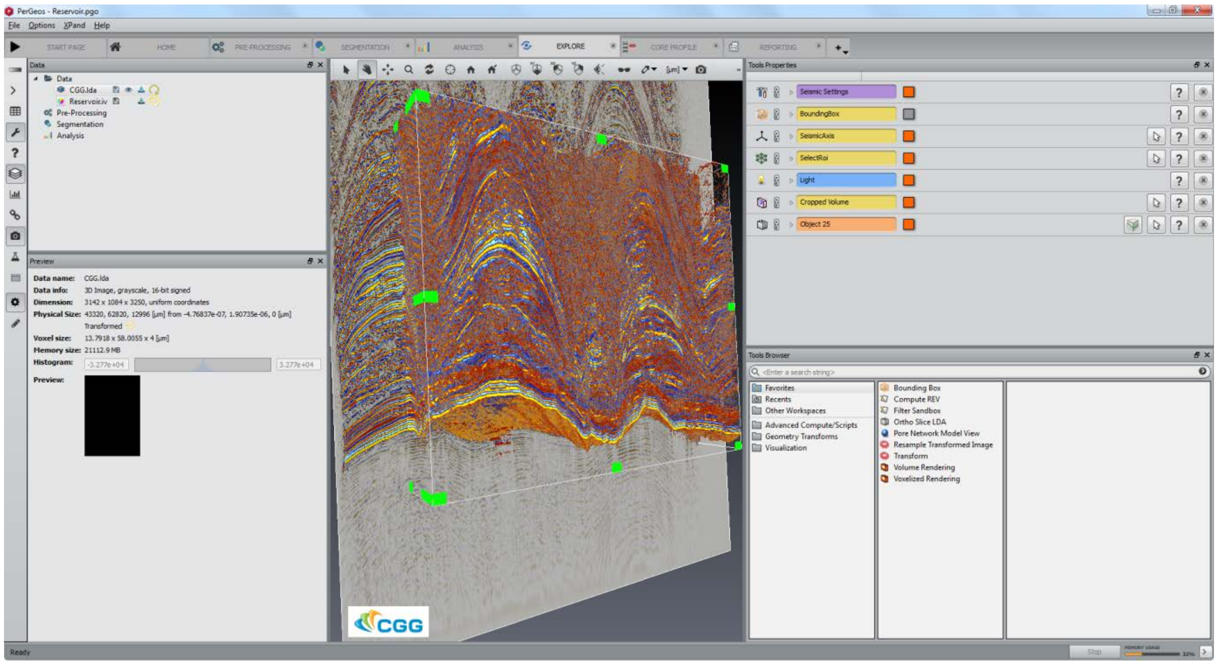
Preview
Data name: Plug10mm.am.lda
Data info: 3D Image, grayscale, 16-bit unsigned
Dimension: 2024 x 2024 x 3892, uniform coordinates
Physical Size: 10016, 10016, 19264 [µm] from 252000, 220000, 120000 [µm]
Voxel size: 4.95106 x 4.95106 x 4.95091 [µm]
Memory size: 30410.5 MB
Histogram: 1648 6.554e+04
Preview:

Display Properties
Data: Plug10mm.am.lda
Scale Bars: None Two axis [✓ Small]
Colormap: 12296 27717 Edit
View Layout: XY XZ YZ 3D [✓ All Viewers]
Slice Number X: 1011
Slice Number Y: 1011
Slice Number Z: 1945

Data Properties
Data Info
Data name: Plug10mm.am.lda
Data info: 3D Image, grayscale, 16-bit unsigned
Dimension: 2024 x 2024 x 3892, uniform coordinates
Bounding Box
[✓ Bounding Box] Voxel Size
Min Coord: 252000 220000 120000 [µm]
Max Coord: 262016 230016 139264 [µm]
Transformation
None
Data Window
1648 65535
Data Annotations
User Notes
History
Clear History
Input: Plug10mm.am.lda

Figure 3 30GB Carbonate plug using only 2G RAM

Seismic volumes can be visualized in the Explore workspace with a *Volume Rendering* module:



Preview
Data name: CGG.lda
Data info: 3D Image, grayscale, 16-bit signed
Dimension: 3142 x 1084 x 3250, uniform coordinates
Physical Size: 43320, 62820, 12996 [µm] from -4.76837e-07, 1.90735e-06, 0 [µm]
Transformed
Voxel size: 13.7918 x 58.0055 x 4 [µm]
Memory size: 21112.9 MB
Histogram: -3.277e+04 3.277e+04
Preview:

Tools Properties
Seismic Settings
BoundingBox
SeismicAxis
SelectRoi
Light
Cropped Volume
Object 25

Tools Browser
<Enter a search string>
Favorites
Recents
Other Workspaces
Advanced Compute/Scripts
Geometry Transforms
Visualization

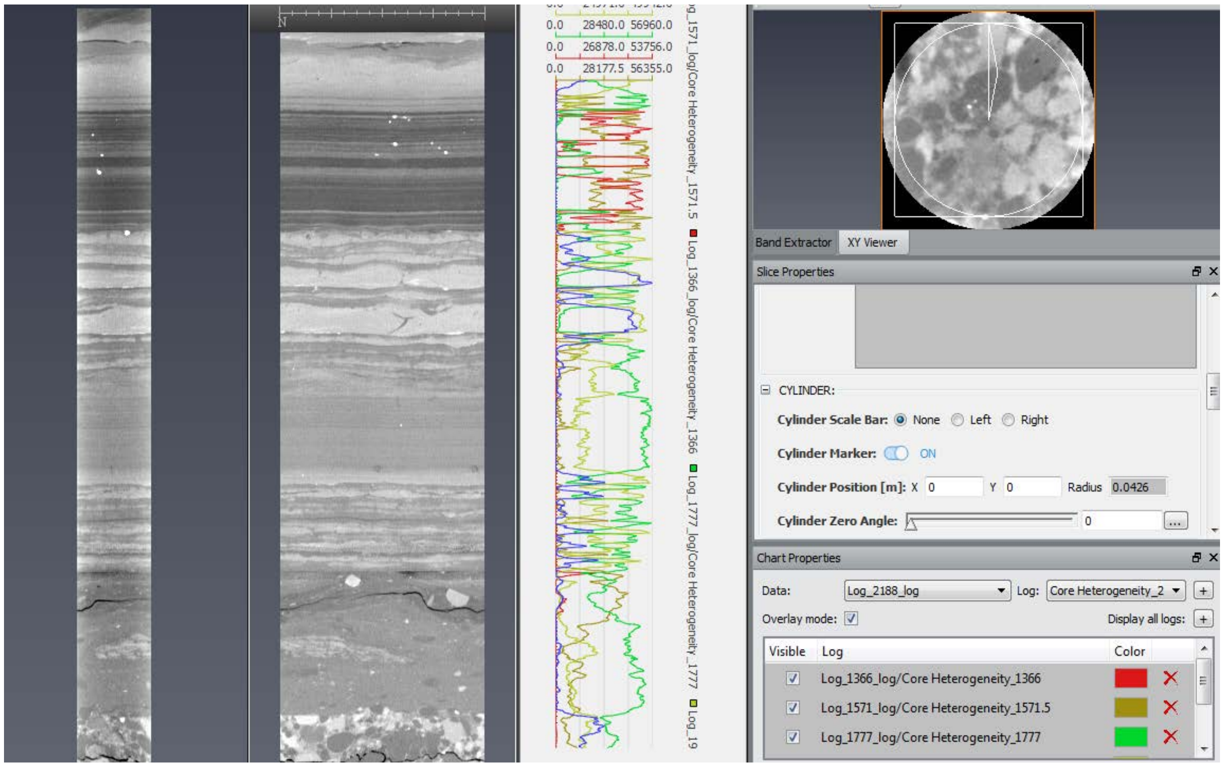
Bounding Box
Compute REV
Filter Sandbox
Ortho Slice LDA
Pore Network Model View
Resample Transformed Image
Transform
Volume Rendering
Voxelized Rendering

Figure 4 Seismic volume display

Core Profile

The core profile can natively handle a .lda stitched core. All visualizations (vertical slice, cylinder slice, XY slice, photo) and processing (logs generation) will be managed by an out of core mechanism.

It is then possible to interactively visualize and process a 100+ GB Whole Core.



CYLINDER:

Cylinder Scale Bar: [✓ None] Left Right

Cylinder Marker: ON

Cylinder Position [m]: X 0 Y 0 Radius 0.0426

Cylinder Zero Angle: 0

Data: Log_2188_log Log: Core Heterogeneity_2

Overlay mode: ✓ Display all logs:

Visible	Log	Color
✓	Log_1366_log/Core Heterogeneity_1366	🟥
✓	Log_1571_log/Core Heterogeneity_1571.5	🟨
✓	Log_1777_log/Core Heterogeneity_1777	🟩

Figure 1 Core profile workspace with whole core and generated logs

Segmentation

Unless having 4x the size of the data in RAM, the segmentation will need to be applied by blocks, with the *binarization by slab* module.

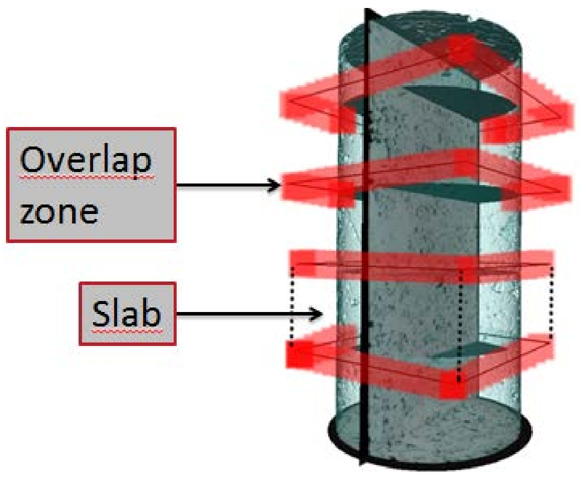


Figure 2 Slab processing mechanism

The following recipe has been specifically designed to segment the pore space of Heliscan data. It uses a combination of masking, 3D marker based Watershed and black tophat.

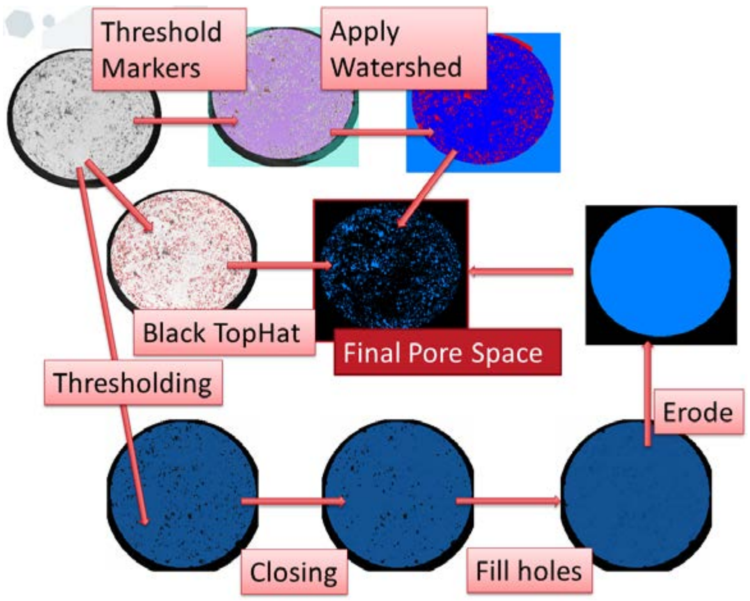
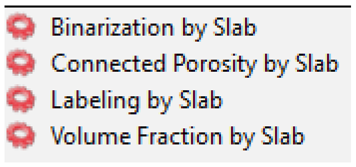


Figure 3 Core Sample porosity segmentation recipe

Applying a recipe by slabs

The dedicated PerGeos modules allow the processing of .lda data by blocks :

a. Binarization by slab
b. Connected porosity by slab
c. Labeling by slab
d. Volume fraction by slab



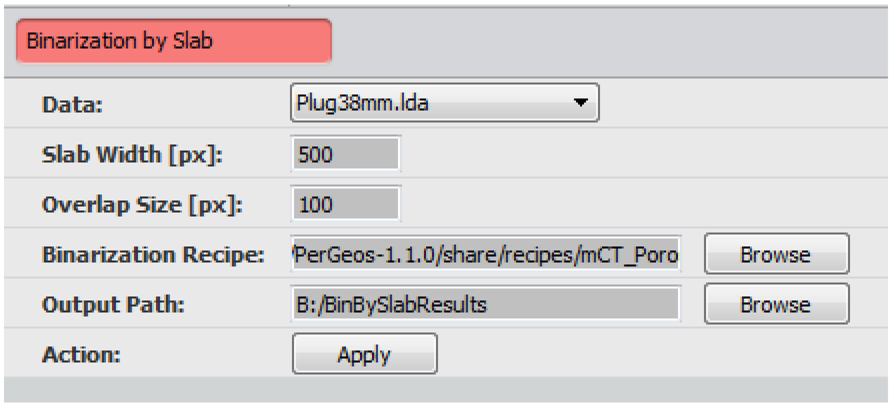
The main one is *Binarization by slab*, responsible of extracting a binary volume as a .lda file from a greyscale .lda data.

A particular attention will be given on the size of the blocks and the overlap, especially when using a Watershed based segmentation.

The data is split into blocks, and each block is processed independently from the others.

The height of each block should be adjusted depending on the available RAM (the maximum Width should correspond to 1/6 of the RAM).

As an example, on a workstation with 32GB of RAM, 500 is a correct size for block height with a 2000x2000xn 16 bits core, with an overlap of 100 to guarantee a continuous segmentation between the blocks (if using a Watershed in the segmentation recipe).


Binarization by Slab

Data:	Plug38mm.lda
Slab Width [px]:	500
Overlap Size [px]:	100
Binarization Recipe:	'PerGeos-1.1.0/share/recipes/mCT_Poro
Output Path:	B:/BinBySlabResults

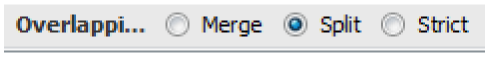
Separating features

If a separation is needed, 2 options are available:

- Incorporate it in the binarization recipe
- apply a second Binarization by slab containing the separation

The separation will be limited by the size of the blocks. Thus, the overlapping must be increased in order to be higher than the size of the features to be separated.

The overlapping mode of Binarization by slab should be set to *Split*.



Overlappi... Merge [✓ Split] Strict

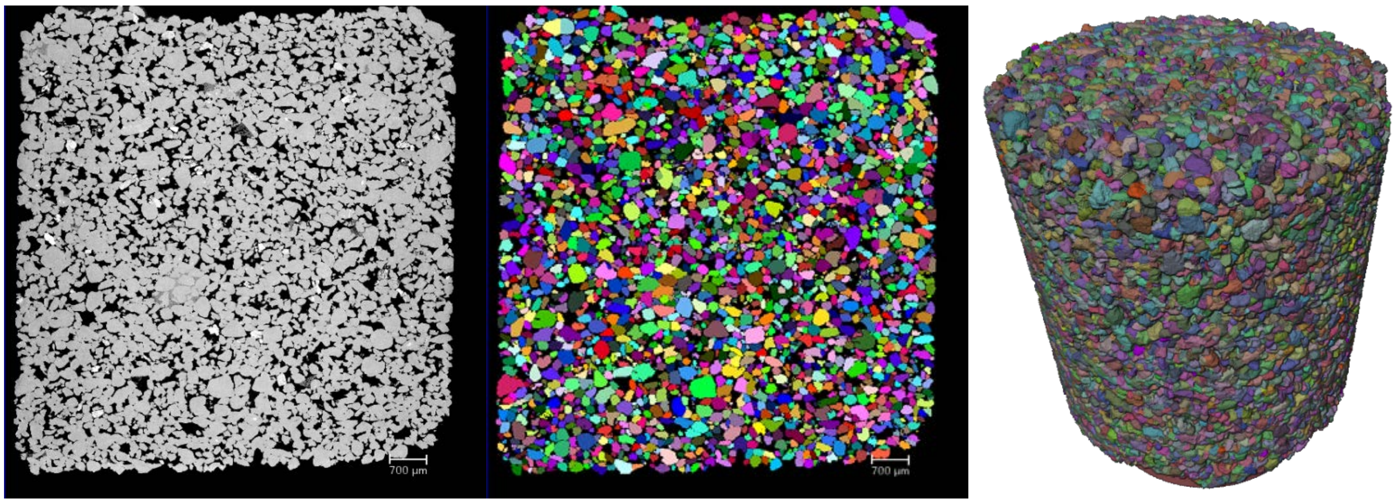


Figure 4 100000+ separated grains from a 40GB sandstone data