November 13–14, 2024 | Online

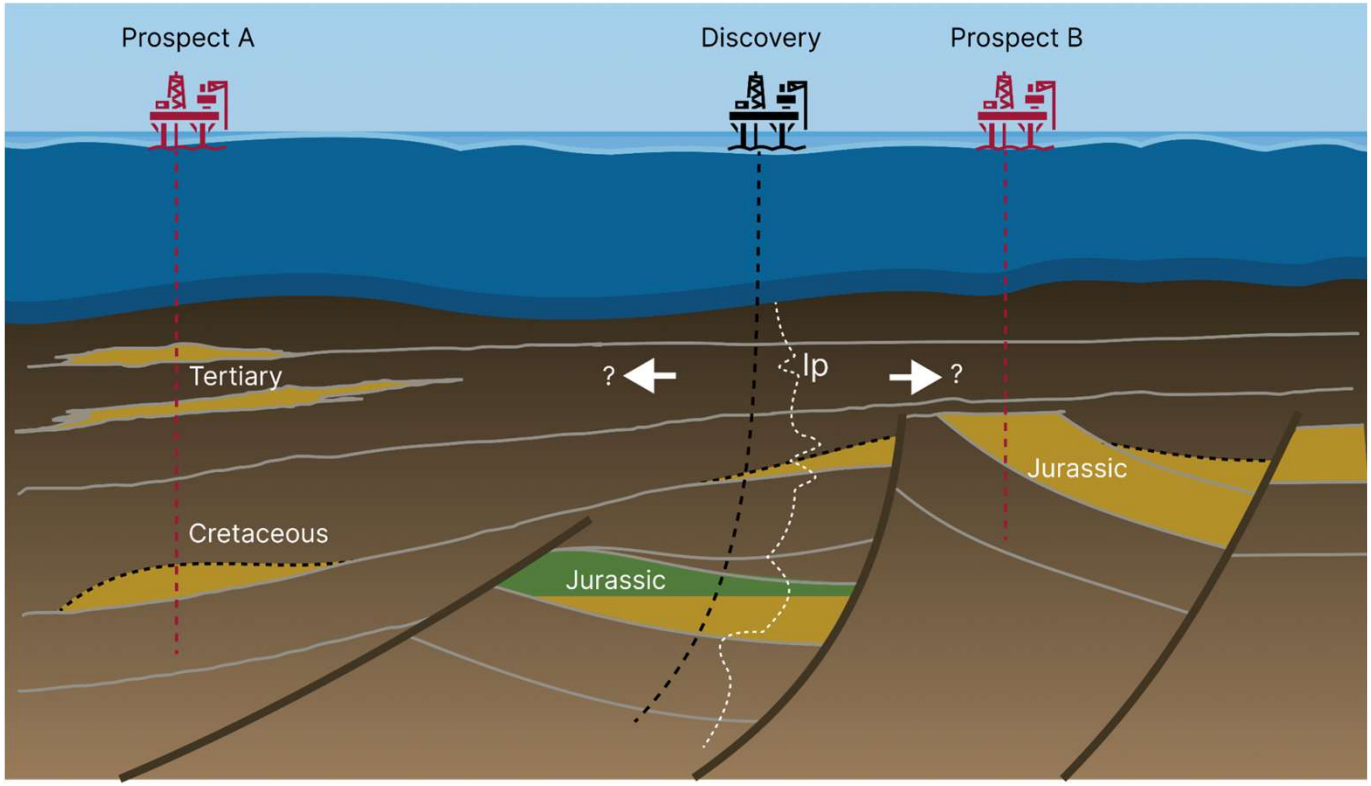
Big Data and MATLAB: Dig Software

Erling Hugo Jensen,
Dig Technology

Ivan Lehocki,
Dig Technology

MATLAB EXPO

Dig Technology helps oil and gas companies map the subsurface



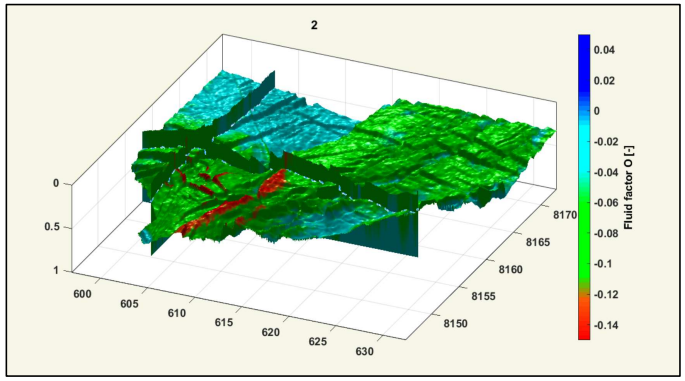
We have many workflows to make predictions of the subsurface

Implementing our workflows in existing software is expensive and not so easy

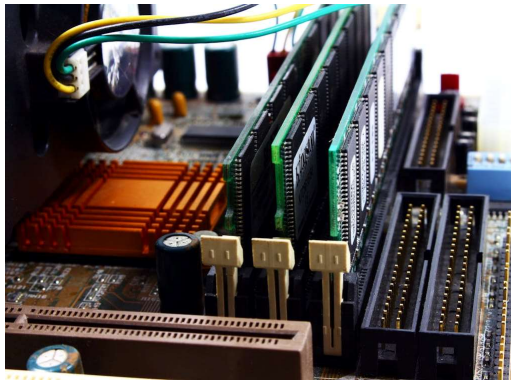
Key challenges we face working with big 3D volume data

Loading

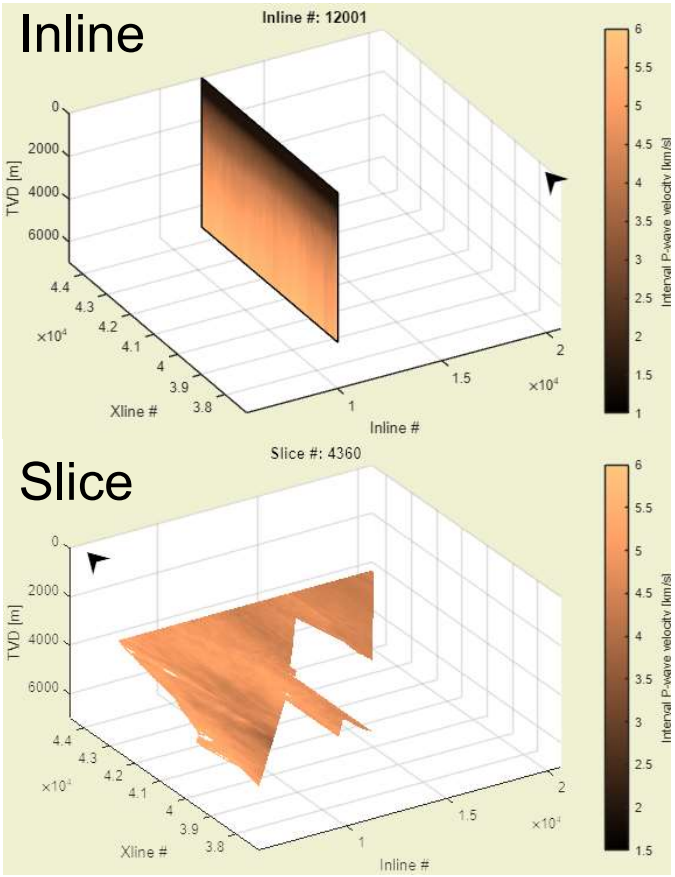
Plotting

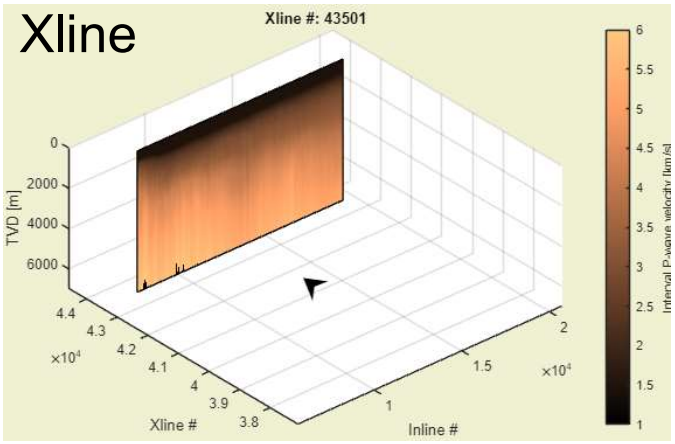


Memory

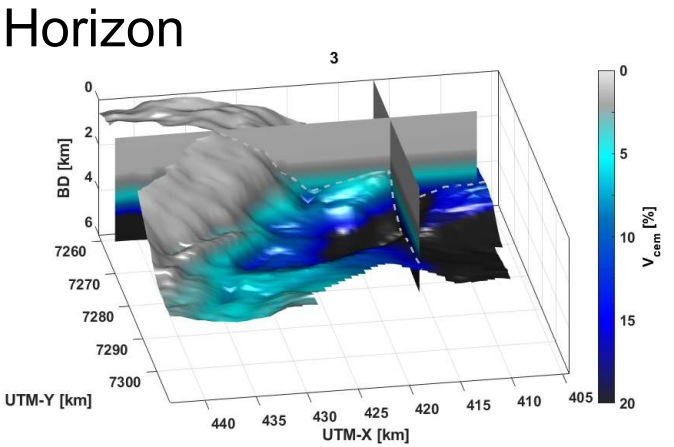


Features associated with subsets of data we extract from the volumes





Profiles



Slices

Some more details on SEGY data file format

Initial file header

Text (3200 bytes) and binary header (400 bytes)

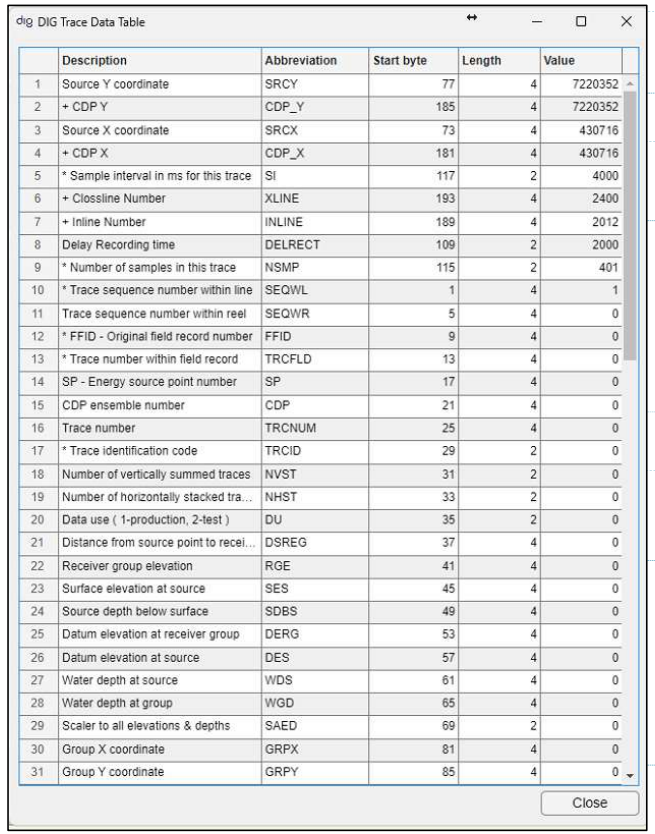
Inline 100, xline 100

Inline 100, xline 101

Inline 101, xline 100

Inline 101, xline 101

Inline 800, xline 180



	Description	Abbreviation	Start byte	Length	Value
1	Source Y coordinate	SRCY	77	4	7220352
2	+ CDP Y	CDP_Y	185	4	7220352
3	Source X coordinate	SRCX	73	4	430716
4	+ CDP X	CDP_X	181	4	430716
5	* Sample interval in ms for this trace	SI	117	2	4000
6	+ Crossline Number	XLINE	193	4	2400
7	+ Inline Number	INLINE	189	4	2012
8	Delay Recording time	DELRECT	109	2	2000
9	* Number of samples in this trace	NSMP	115	2	401
10	* Trace sequence number within line	SEQWL	1	4	1
11	Trace sequence number within reel	SEQWR	5	4	0
12	* FFID - Original field record number	FFID	9	4	0
13	* Trace number within field record	TRCFLD	13	4	0
14	SP - Energy source point number	SP	17	4	0
15	CDP ensemble number	CDP	21	4	0
16	Trace number	TRCNUM	25	4	0
17	* Trace identification code	TRCID	29	2	0
18	Number of vertically summed traces	NVST	31	2	0
19	Number of horizontally stacked tra...	NHST	33	2	0
20	Data use (1-production, 2-test)	DU	35	2	0
21	Distance from source point to recei...	DSREG	37	4	0
22	Receiver group elevation	RGE	41	4	0
23	Surface elevation at source	SES	45	4	0
24	Source depth below surface	SDBS	49	4	0
25	Datum elevation at receiver group	DERG	53	4	0
26	Datum elevation at source	DES	57	4	0
27	Water depth at source	WDS	61	4	0
28	Water depth at group	WGD	65	4	0
29	Scaler to all elevations & depths	SAED	69	2	0
30	Group X coordinate	GRPX	81	4	0
31	Group Y coordinate	GRPY	85	4	0

Trace data (varies in size from file to file)

Trace data (varies in size from file to file)

Trace data (varies in size from file to file)

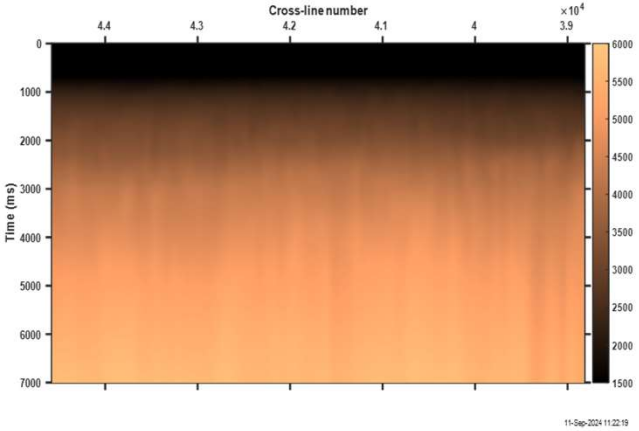
Trace data (varies in size from file to file)

Trace data (varies in size from file to file)

Starting point: Performance using SeisLab v3.01 in MATLAB

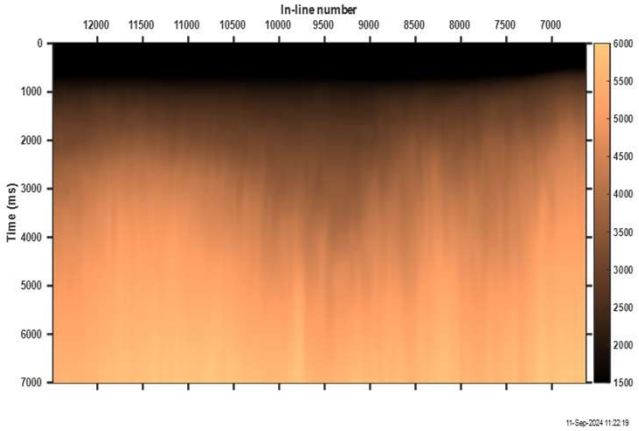
Trace header			
File size	3 GB	20 GB	658 GB
Loading time	12 s	75 s	48 minutes

Inline



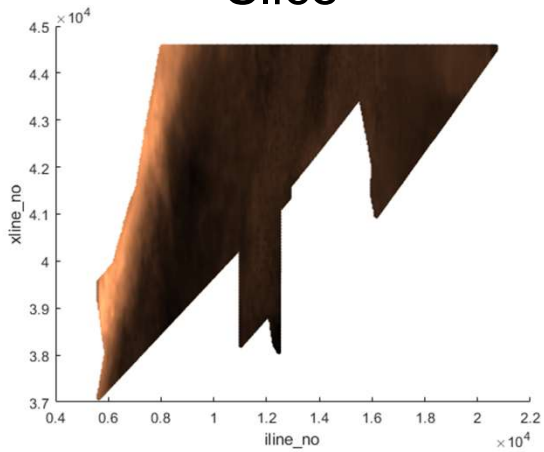
3 GB file: 25 s (8MB data)
20 GB file: 303 s (19MB data)
658 GB file: Out of memory

Xline



3 GB file: 22 s (8MB data)
20 GB file: 137 s (20MB data)
658 GB file: Out of memory

Slice



3 GB file: 12 s (4MB data)
20 GB file: 75 s (24MB data)
658 GB file: Out of memory

Performance of our Dig SEGY library – Trace headers

Trace header loading times			
File size	3 GB	20 GB	658 GB
SeisLab	12.4 s	74.9 s	48 minutes
Dig SEGY	1.75 s	11.3 s	16 minutes

dig

Can be read once for each volume and stored in the cache

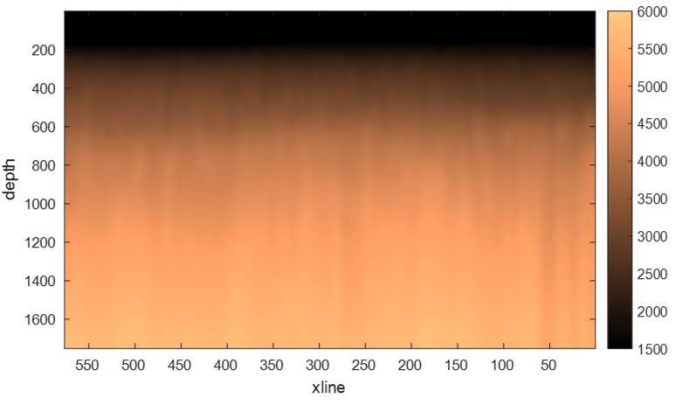
```
% init Cache
Cache = CNTR_Cache;

% get Data from cache
Data = Cache.getObj(label);

% store Data in cache
Cache.update(label, name, Data);
```

Pseudo-singleton design
All instances of CNTR_Cache are the same with the help of “persistent” variable

Performance of our Dig SEGY library – Trace data

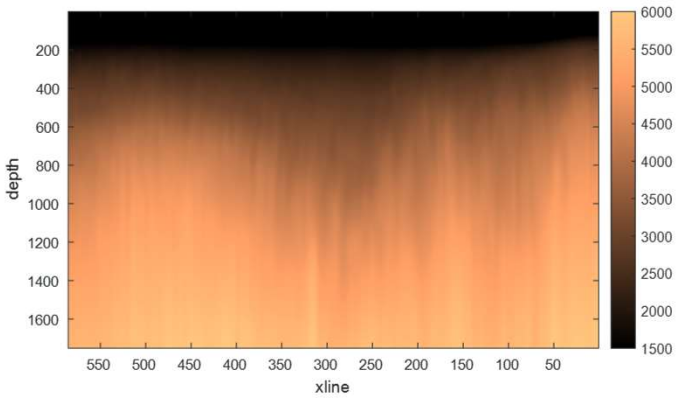
Drastic improvement of loading time, but still problems with slicing the big volume.

Inline



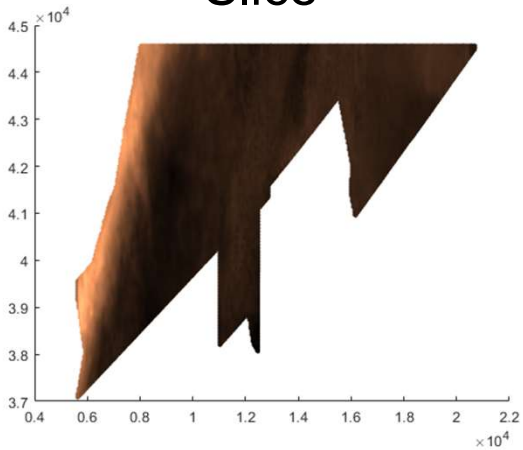
3 GB file: 25 s → 0.056 s
20 GB file: 303 s → 0.069 s
658 GB file: ! → 0.50 s

Xline



3 GB file: 22 s → 0.10 s
20 GB file: 137 s → 0.18 s
658 GB file: ! → 1.28 s

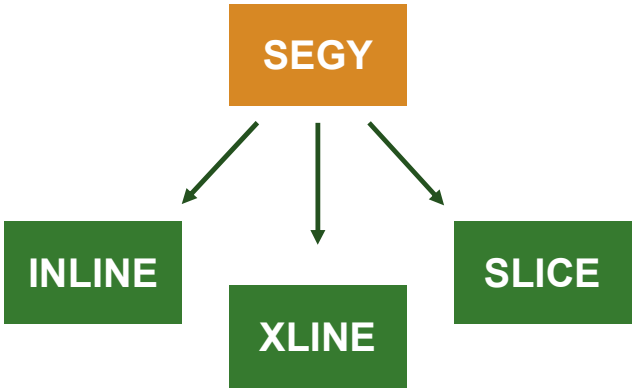
Slice



3 GB file: 12 s → 1.7 s
20 GB file: 75 s → 11 s
658 GB file: Out of memory

We have created our own HDF5 based file format which we reformat the SEGY files to

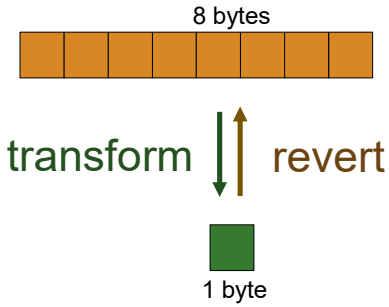
One file per type



Structure

- Inline 100
XL 10, XL 20, … XL N
- Inline 110
XL 20, XL 30, … XL N
…
- Inline M
XL 30, XL 40, … XL N

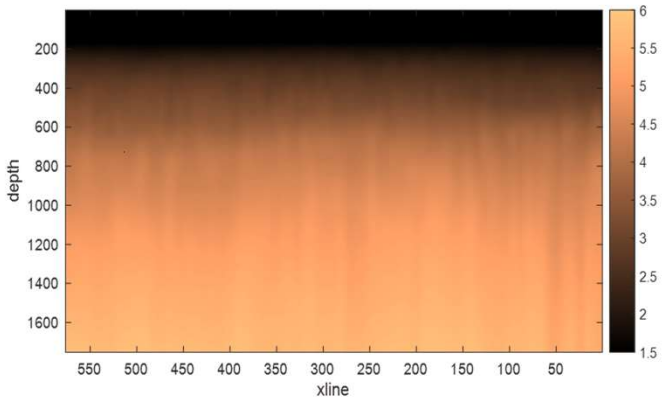
Shift and scale



Performance using DIG HDF5 file format

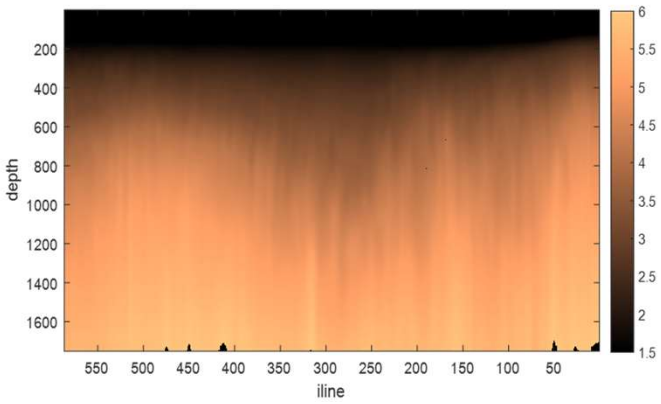
Metadata loading time: 0.3 s (3 GB), 0.5 s (20 GB) and 2 s (658 GB).
Accessing from cache: < 0.001 s.

Inline



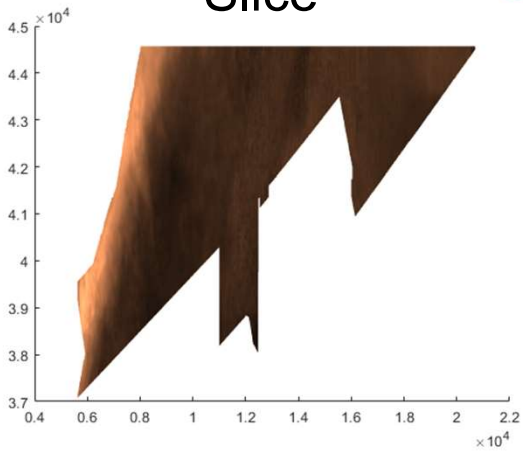
3 GB file: 0.056 s → 0.042 s
20 GB file: 0.069 s → 0.067 s
658 GB file: 0.50 s → 0.16 s

Xline



3 GB file: 0.10 s → 0.055 s
20 GB file: 0.18 s → 0.16 s
658 GB file: 1.28 s → 0.11 s

Slice

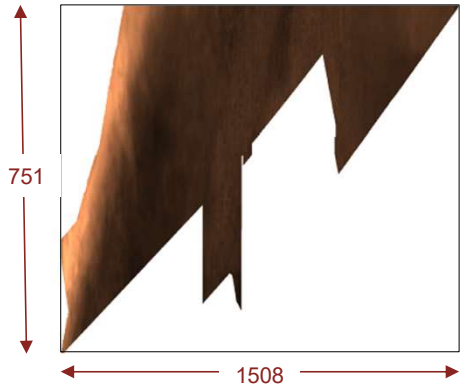


3 GB file: 12 s → 0.14 s
20 GB file: 75 s → 0.16 s
658 GB file: ! → 0.27 s

Drastic improvement of loading time,
but plotting the slice of 658 GB file takes 30 seconds

We have discovered surf-lite

```
% Preparing the Data for plotting
% nrOfTraces = 751x1508 = 1132508 (in case of 3 GB)
Data = single(NaN(nrOfTraces,1));

% IsMemberIndex = true => we have data, false if not
Data(IsMemberIndex) = Trace_Data;
Data = reshape(Data, [751 1508]);
```



3 GB: 751 x 1508
20 GB: 1876 x 3768
658 GB: 16370 x 15071

1)

```
% "Normal" surf function, where
% X, Y, Z, Data has same size: 751 x 1508
[X, Y] = meshgrid(Inline, Xline);
Z = depth*ones(size(Data));
FH = figure;
surf(AH, X, Y, Z, Data, ...
    'FaceColor', 'interp');
```

2)

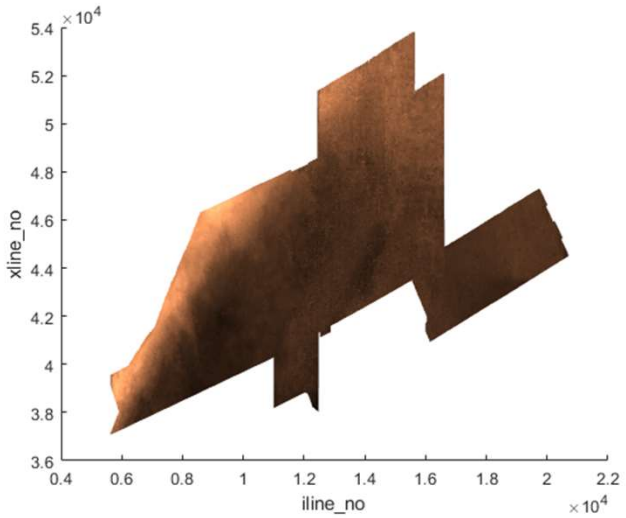
```
% "Surf-lite" function, where X, Y, Z are 2x2 with
% extreme values and Data has size: 751 x 1508
X = [min(Inline) min(Inline); ...
        max(Inline) max Inline)];
Y = [min(Xline) max(Xline); ...
        min(Xline) max(Xline)];
Z = depth*[1 1; 1 1];
FH = figure;
surf(AH, X, Y, Z, Data, ...
    'FaceColor', 'texturemap', ...
    'LineStyle', 'None', ...
    'FaceAlpha', 'texturemap', ...
    'AlphaDataMapping', 'none', ...
    'AlphaData', double(~isnan(Data_)));
```

Performance using surf-lite

Speed improvement for plotting and interaction on 658 GB volume

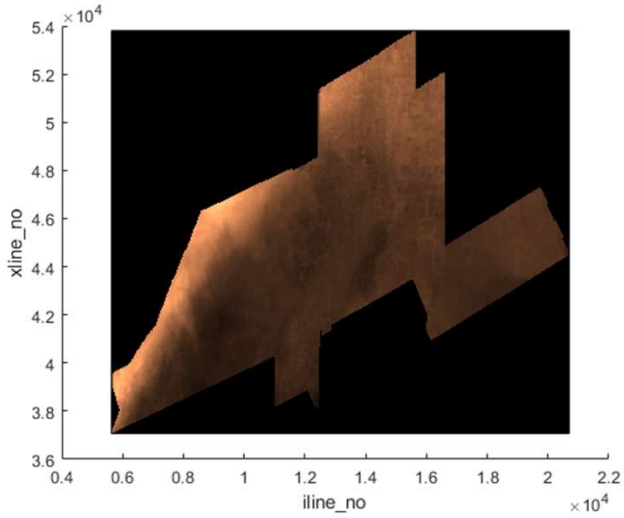
dig

“Normal” surf



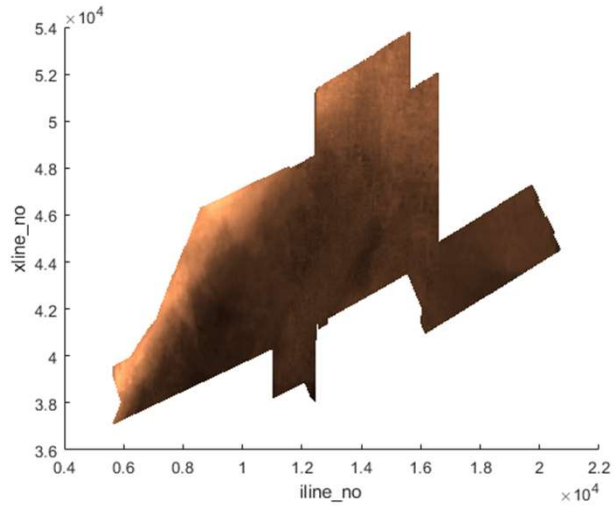
Plotting: 31 s
Rotating: 70 s

“Surf-lite” w/o transparency



Plotting: 6.5 s
Rotating: 4.0 s

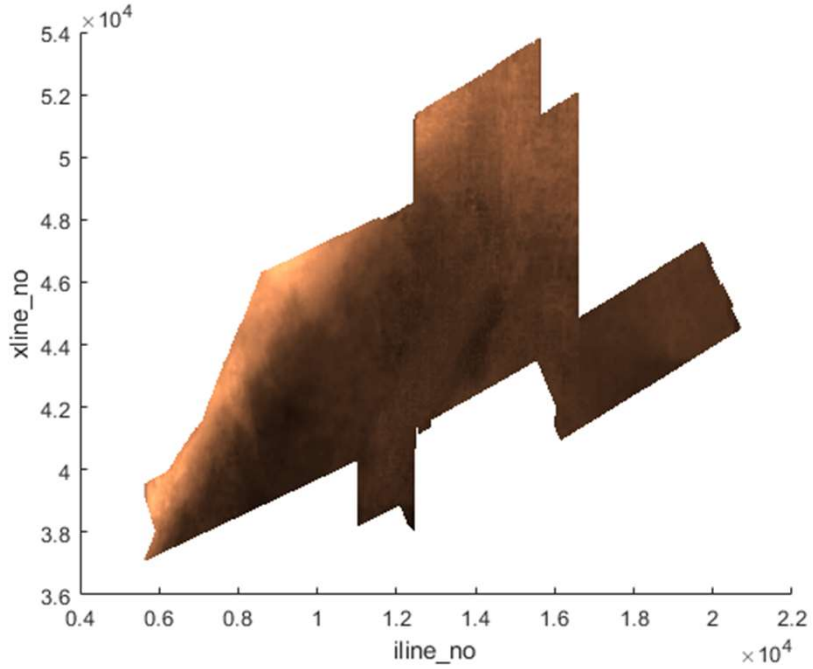
“Surf-lite” w/ transparency



Plotting: 11.6 s
Rotating: 13.2 s

We are resampling the data before plotting

Datapoints 15071 x 16730 squeezed into less than 500 x 400 pixels

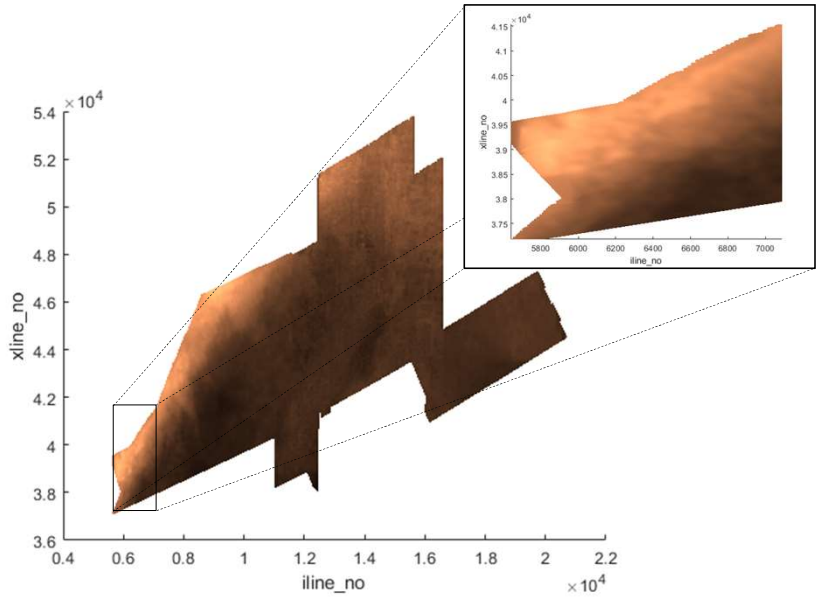


Dig Resampling Procedure:

```
DIG_GFX.stepLengthMultiplier(15);
FH = figure;
AH = gca;
FH.SizeChangedFcn = @DIG_GFX.sizeChangedFcn;
DIG_GFX.surf3Lite(AH, X, Y, Z, Data_, ...
  'FaceColor', 'texturemap', ...
  'LineStyle', 'None', ...
  'AlphaDataMapping', 'none', ...
  'FaceAlpha', 'texturemap', ...
  'AlphaData', double(~isnan(Data_)));

% Resampling is done before plotting and if figure is
% resized, zoomed or panned.
```

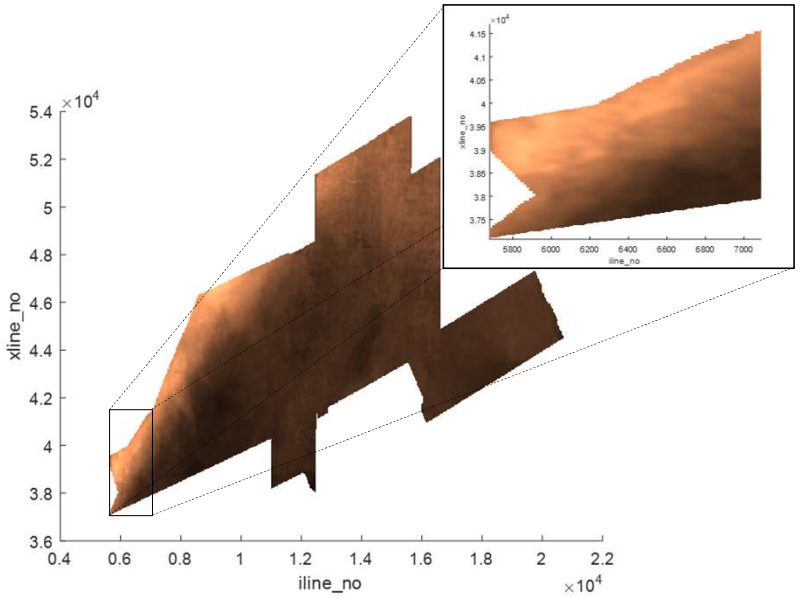
Performance using resampling on 658 GB slice

No resampling



Plotting time: 12 s
Rotating: 13 s

Resampled



Plotting time: 2 s
Rotating: instant

Summary: ca 99% improvement in performance

Trace header / meta data loading times			
File size	3 GB	20 GB	658 GB
SeisLab	12.4 s	74.9 s	48 minutes
Dig SEGY	1.75 s	11.3 s	16 minutes
Dig HDF5	0.30 s	0.5 s	2 s

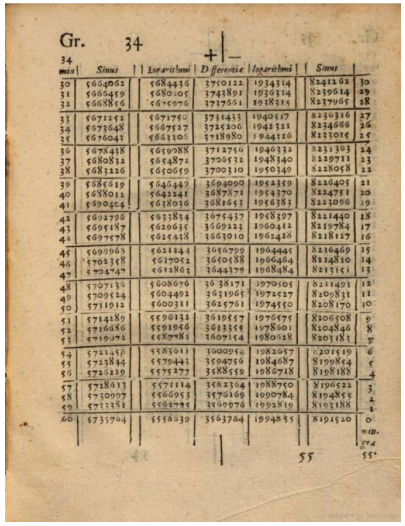
Inline / Xline loading times			
File size	3 GB	20 GB	658 GB
SeisLab	25 s / 22 s	303 s / 137 s	Out of memory
Dig SEGY	0.06 s / 0.10 s	0.07 s / 0.18 s	0.5 s / 1.28 s
Dig HDF5	0.04 s / 0.06 s	0.07 / 0.16 s	0.16 s / 0.11 s

Slice loading times			
File size	3 GB	20 GB	658 GB
SeisLab	12 s	75 s	Out of memory
Dig SEGY	1.7 s	11 s	Out of memory
Dig HDF5	0.14 s	0.16 s	0.27 s

Plotting time for 658 slice can be pushed down to 2s

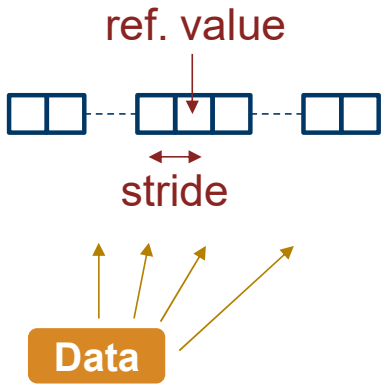
Dig Table: Memoization and lookup table optimization

Avoid repeating heavy calculations with the same input

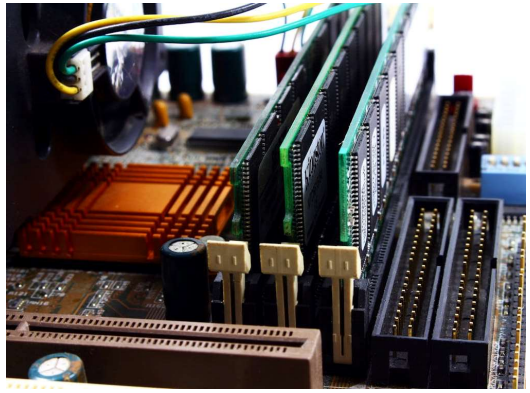


By John Napier - Google Books, Public Domain, https://commons.wikimedia.org/w/index.php?curid=94266674

Lookup table



Discretize data



Caching

Dig Table: Example

- Divide data into subsets
- Get existing results from lookup table
- Calculate missing results and add to lookup table

dig

Processing time			
Approach	For-loop	Parfor-loop	Dig Table
Time	84 s	27 s	13 s

What is best to use depends a lot on number of data points and the operation to be done

Creating a full software in MATLAB is possible

- Practically instant loading and plotting of (big) data
- Responsive design - uigridlayout is your friend
- Other critical factors for Dig Software
  - Prototyping → final product
  - Object oriented programming
  - Custom UI components (ribbons, multi level drop down, status messages)
  - Custom event handler class
  - Strong debugging (breakpoints, step, code-on-the-fly, profiler, …)
  - License manager developed by JenZa

Dig Technology

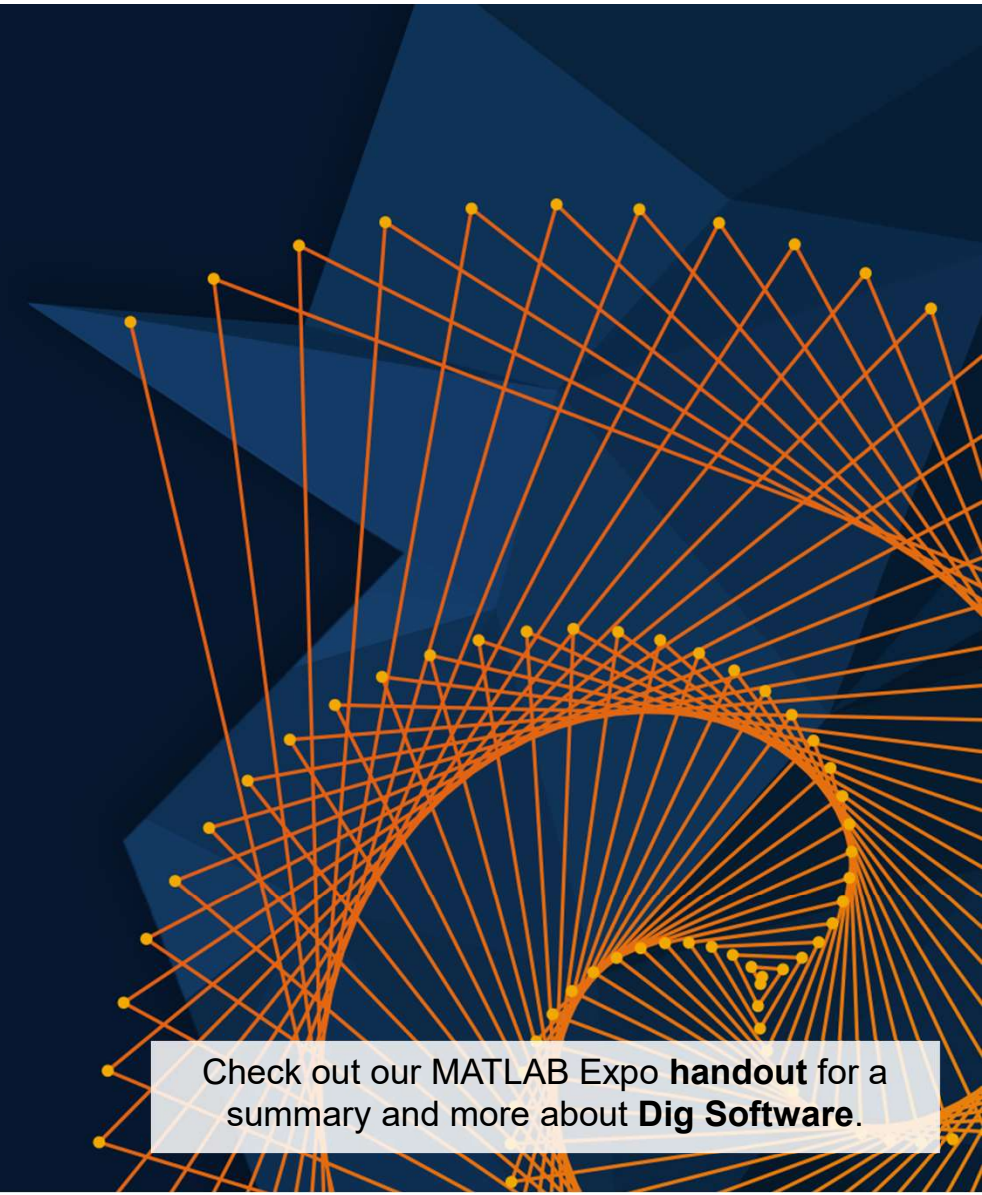
thank you for your interest,

thanks to Aker BP and TGS for allowing us to show their data,

and thank you MathWorks for allowing us to present at

MATLAB EXPO

MathWorks



Check out our MATLAB Expo **handout** for a summary and more about **Dig Software**.