Pontifícia Universidade Católica do Rio de Janeiro

Accelerating Curvature Estimate in 3D Seismic Data Using GPGPU

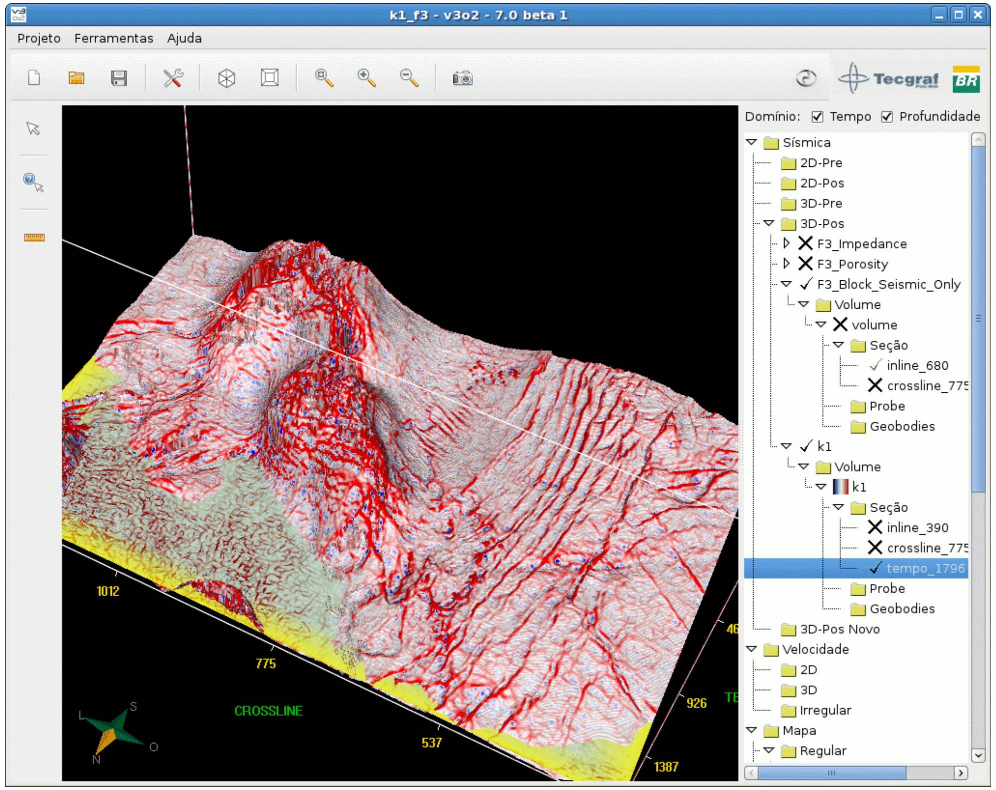
Joner Duarte
jduartejr@tecgraf.puc-rio.br

Tecgraf PUC-RIO

PETROBRAS

Outline

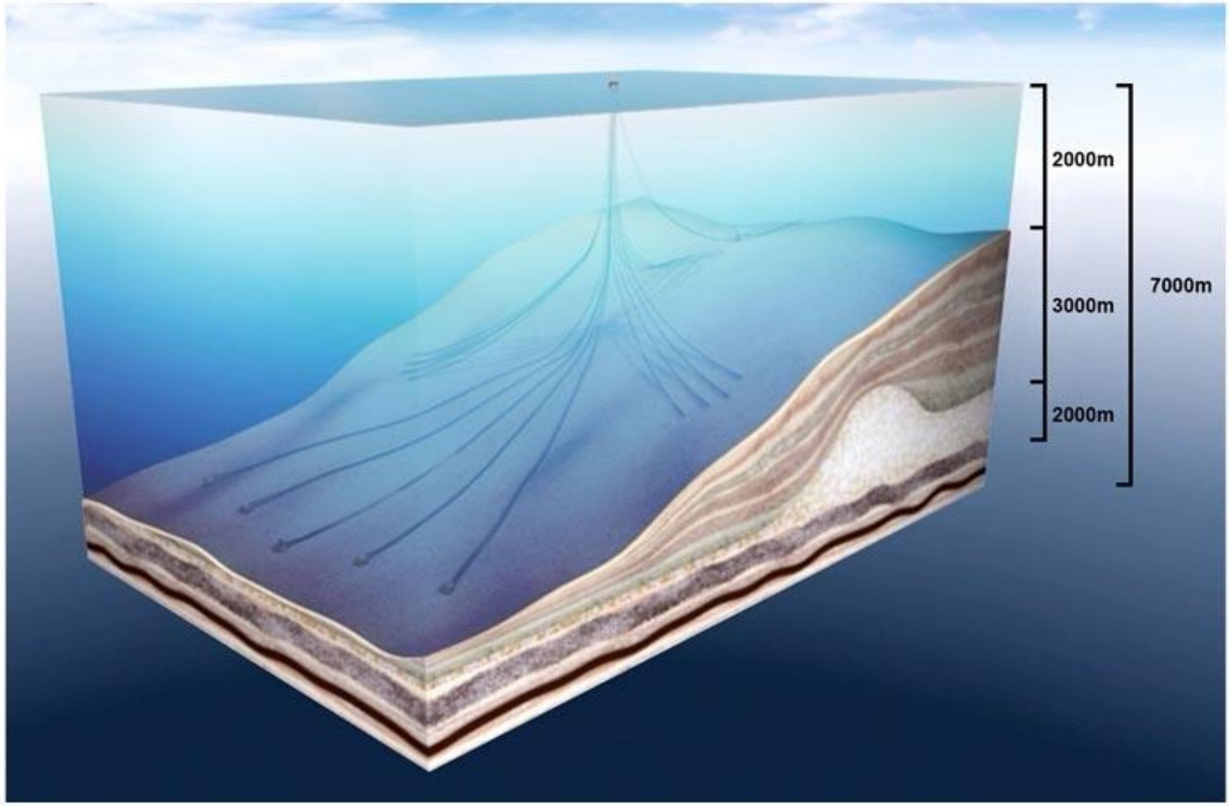
- Introduction
- Volumetric Curvature Estimate
- Parallel Approach
- Results
- Conclusions



Tecgraf PUC-RIO
BR PETROBRAS
NVIDIA
GPU TECHNOLOGY CONFERENCE

Introduction

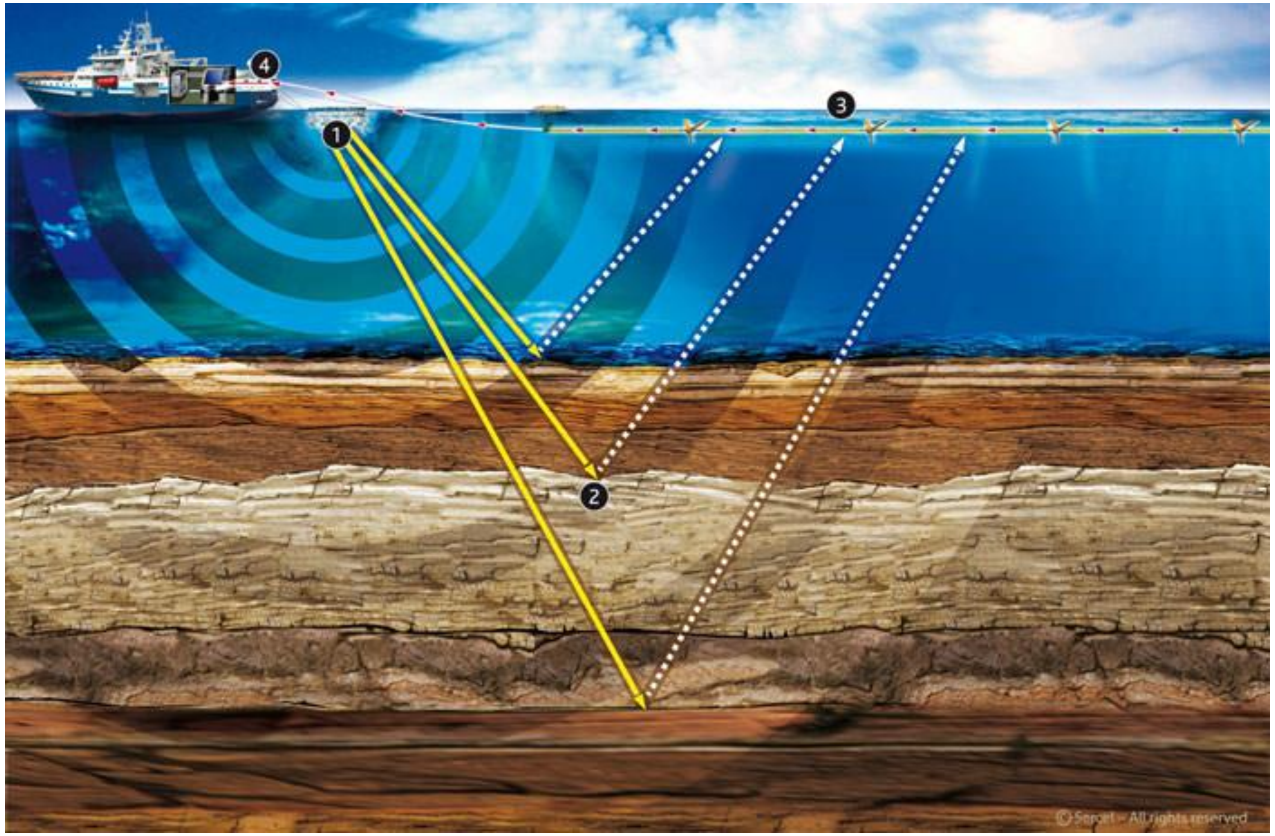
- **3D Seismic Data**



Stratigraphic layers in a seismic acquisition area [Petrobras]

Introduction

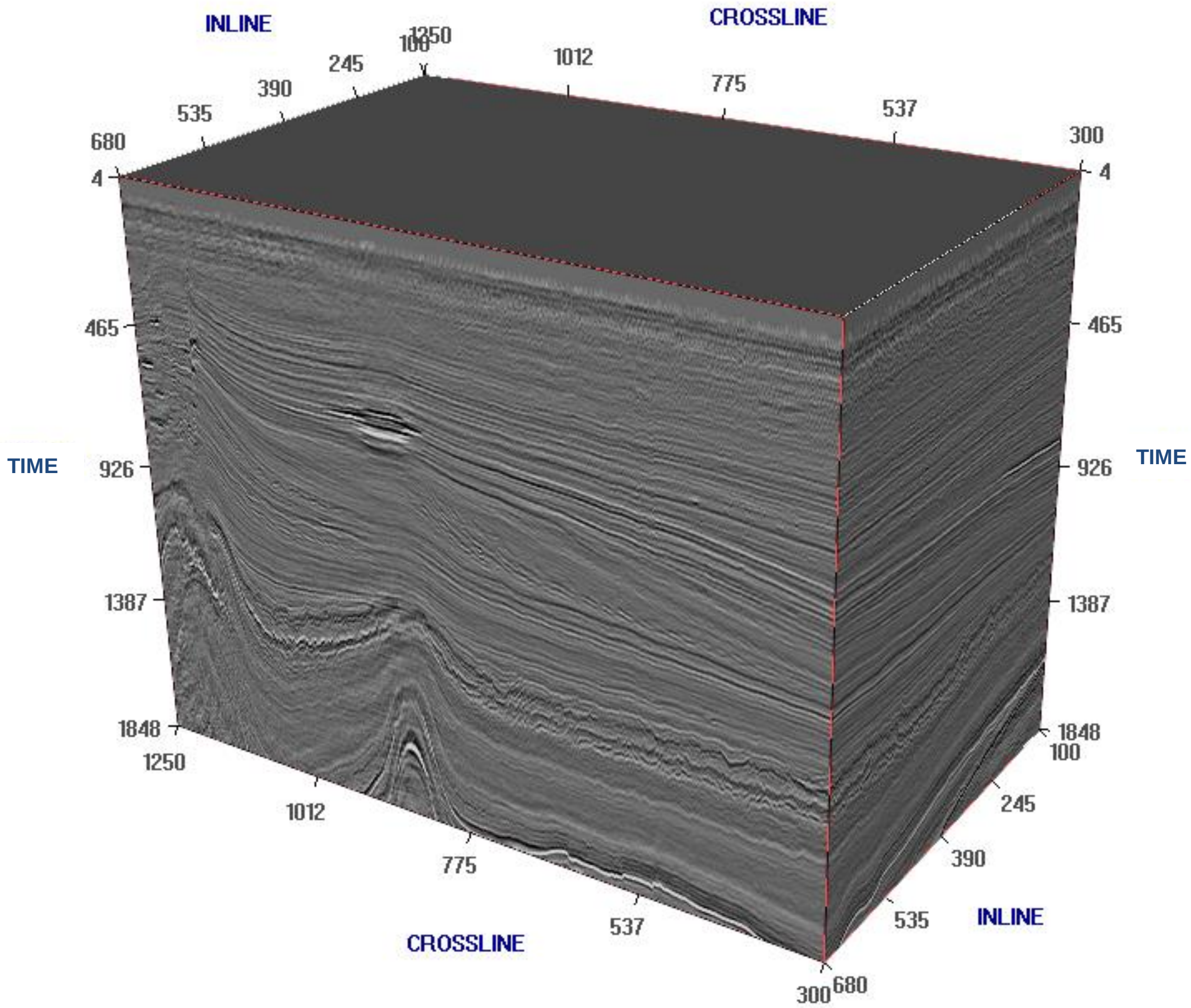
- **3D Seismic Data**



Marine Seismic Acquisition [Sercel]

Introduction

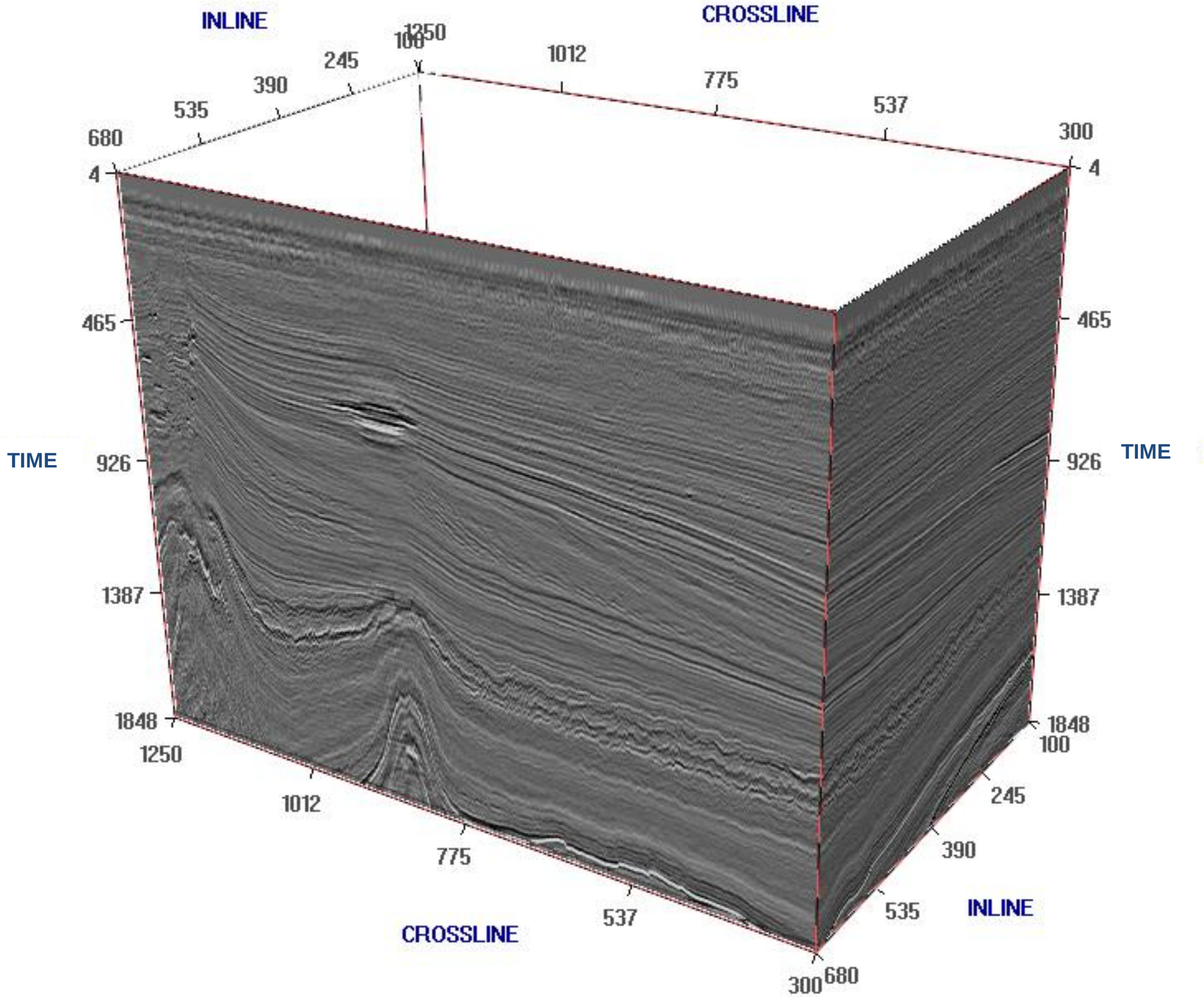
- **3D Seismic Data**



Tecgraf PUC-RIO
BR PETROBRAS
NVIDIA
GPU TECHNOLOGY CONFERENCE

Introduction

- **3D Seismic Data**



Introduction

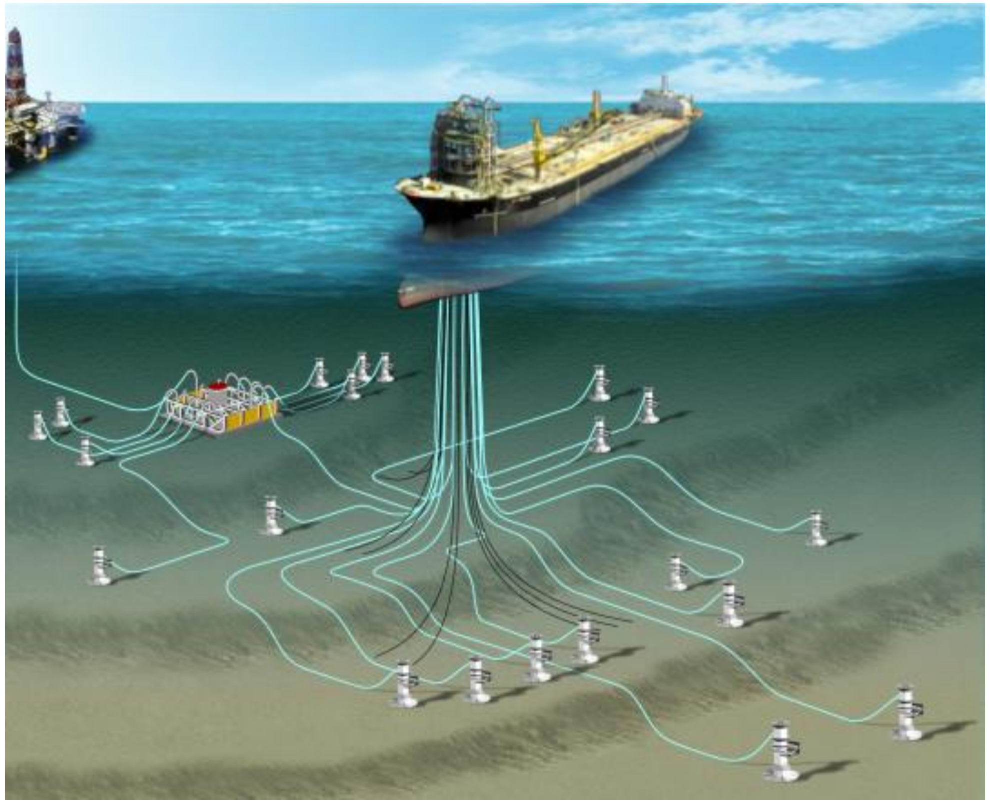
- **Costs to drill an oil well**
  - Pre-salt layer (depth 5000 to 7000 meters)

First well drilled (2005):

- US$ 240 Million
- 1 year

Now, a similar well:

- US$ 60 Million
- 60 days



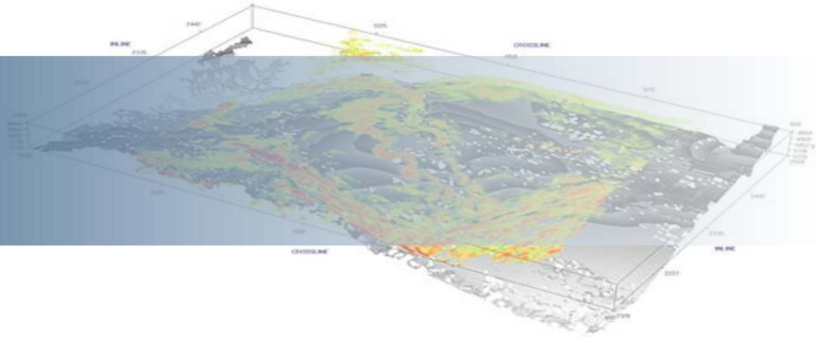
Oil exploration [Petrobras]

Tecgraf PUC-RIO BR PETROBRAS NVIDIA GPU TECHNOLOGY CONFERENCE

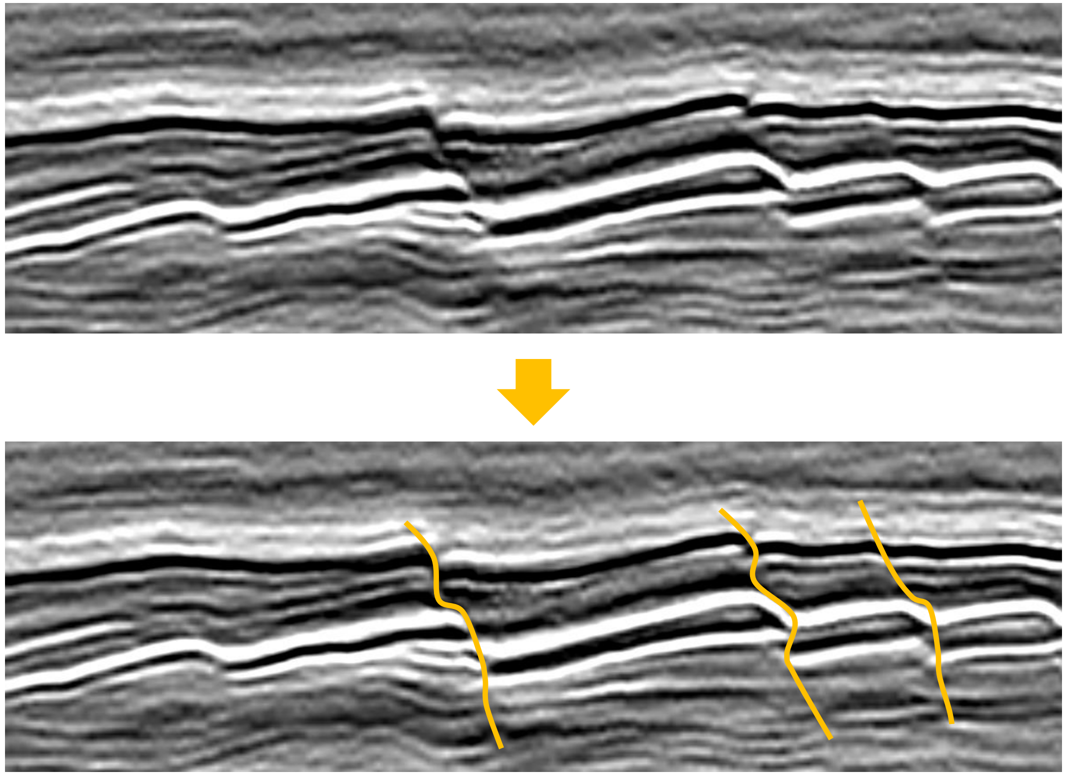
Introduction

- **Seismic Interpretation**
  - **Structural** and **stratigraphic** features
  - Indicates the **presence or absence** of reservoirs

Introduction

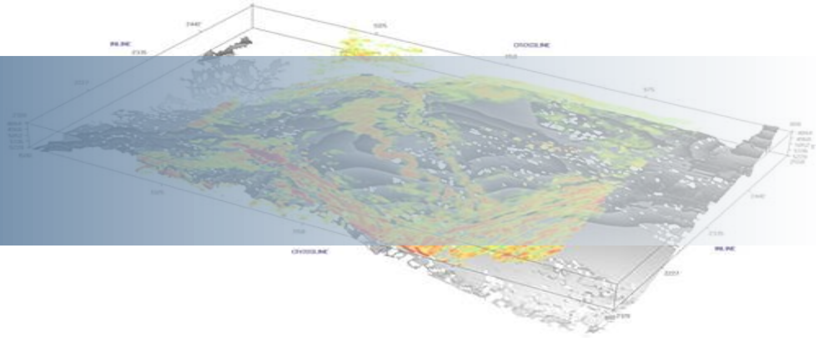


- **Seismic Interpretation**
  - Faults



Fault interpretation process

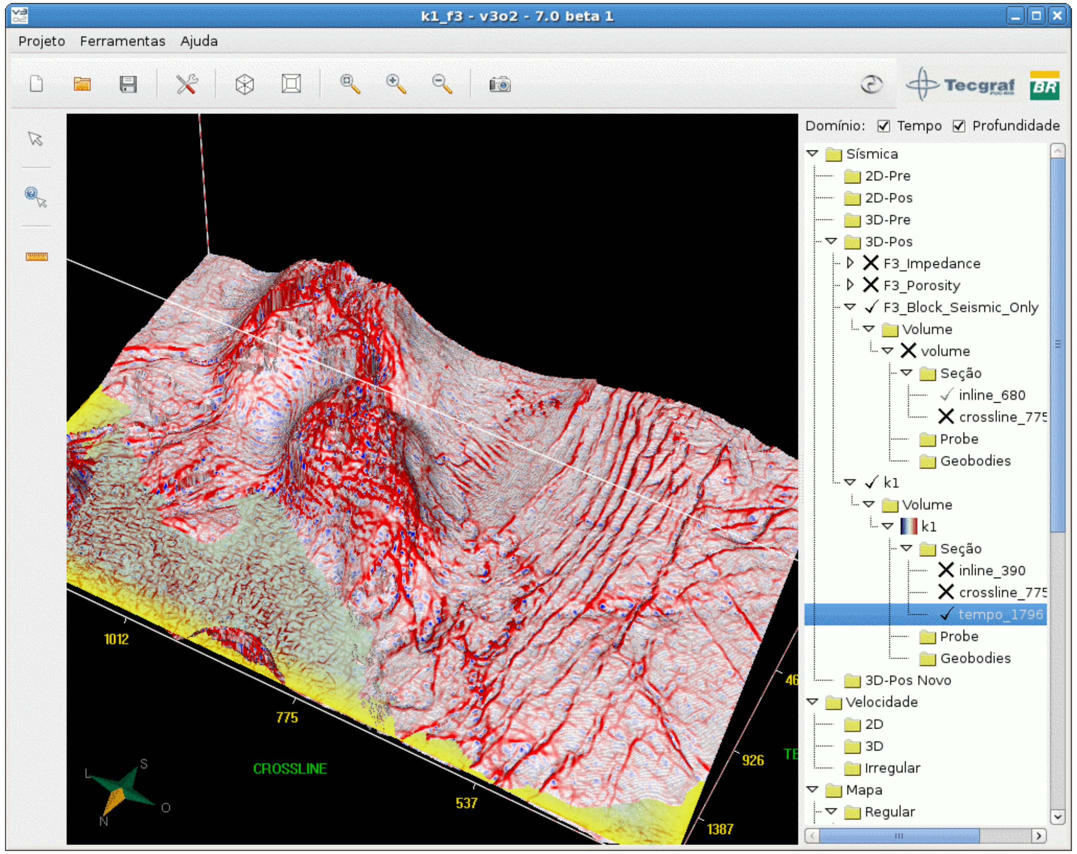
Introduction



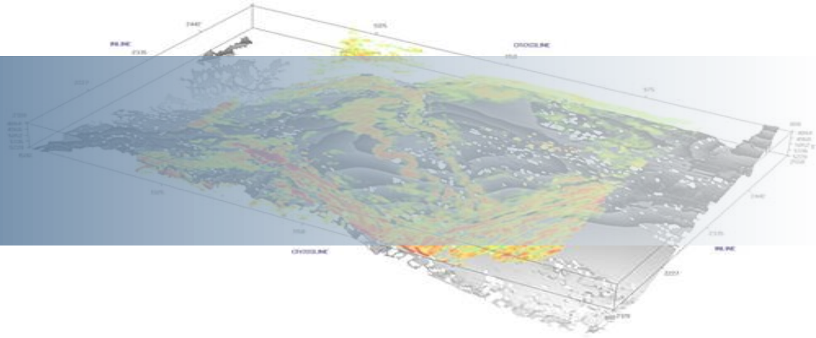
- **Seismic Attributes Estimate**
  - **Highlight** important features
  - Provide **visual aid** on the task of manual interpretation
    - Less susceptible to incorrect interpretations

Introduction

- **Curvature Attributes**



Introduction



But curvature attribute estimate is very **slow**

The interpreter needs to **fine tune** some parameters

Objective

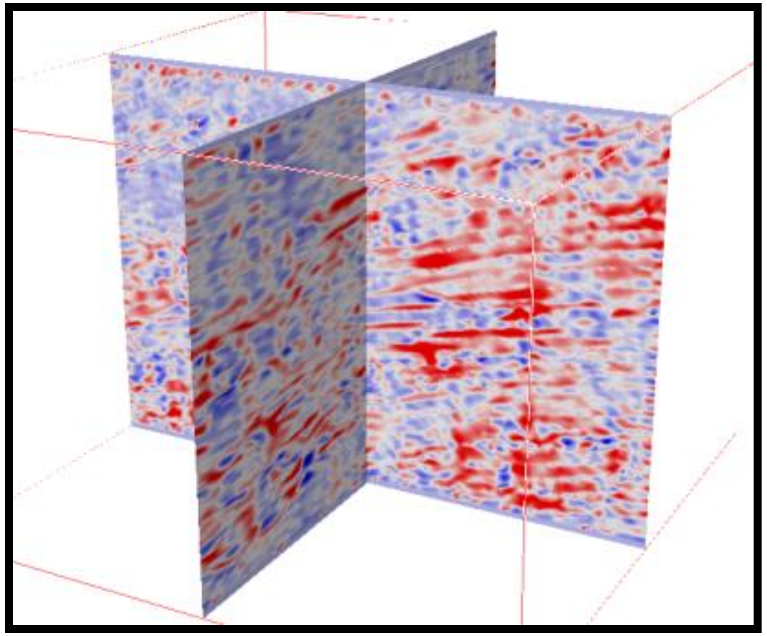
- Enable **visualization at interactive time** of curvature attributes on user workstations
  - Allows fine tune parameters
  - Speeds up the interpretation process
  - Reduces the labor-intensive work
  - Decrease errors due the lack of experience

Volumetric Curvature Estimate

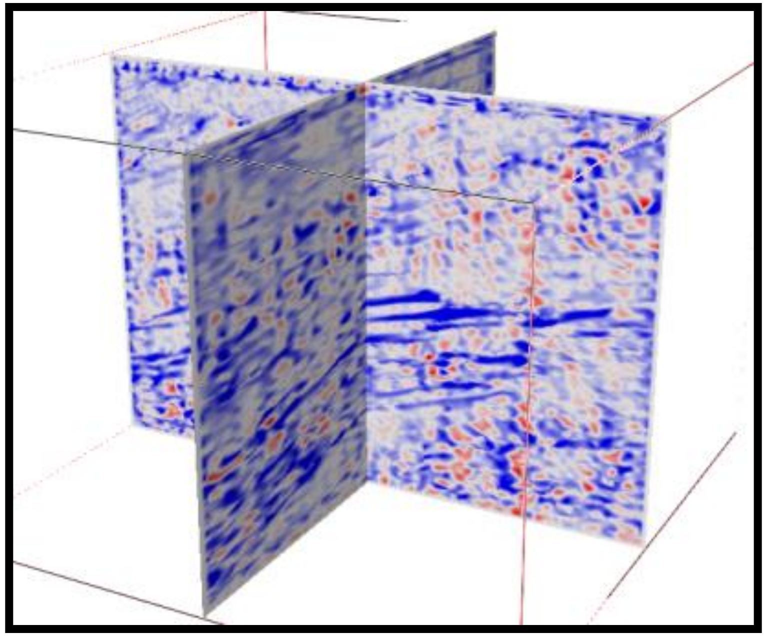
- Second-derivative-based
- Computationally intensive
- It can take several hours on user workstation
- “A method to estimate volumetric curvature attributes in 3D seismic data”, proposed by **Martins et al (2012)**

Volumetric Curvature Estimate (VCE)

- Curvature Attributes

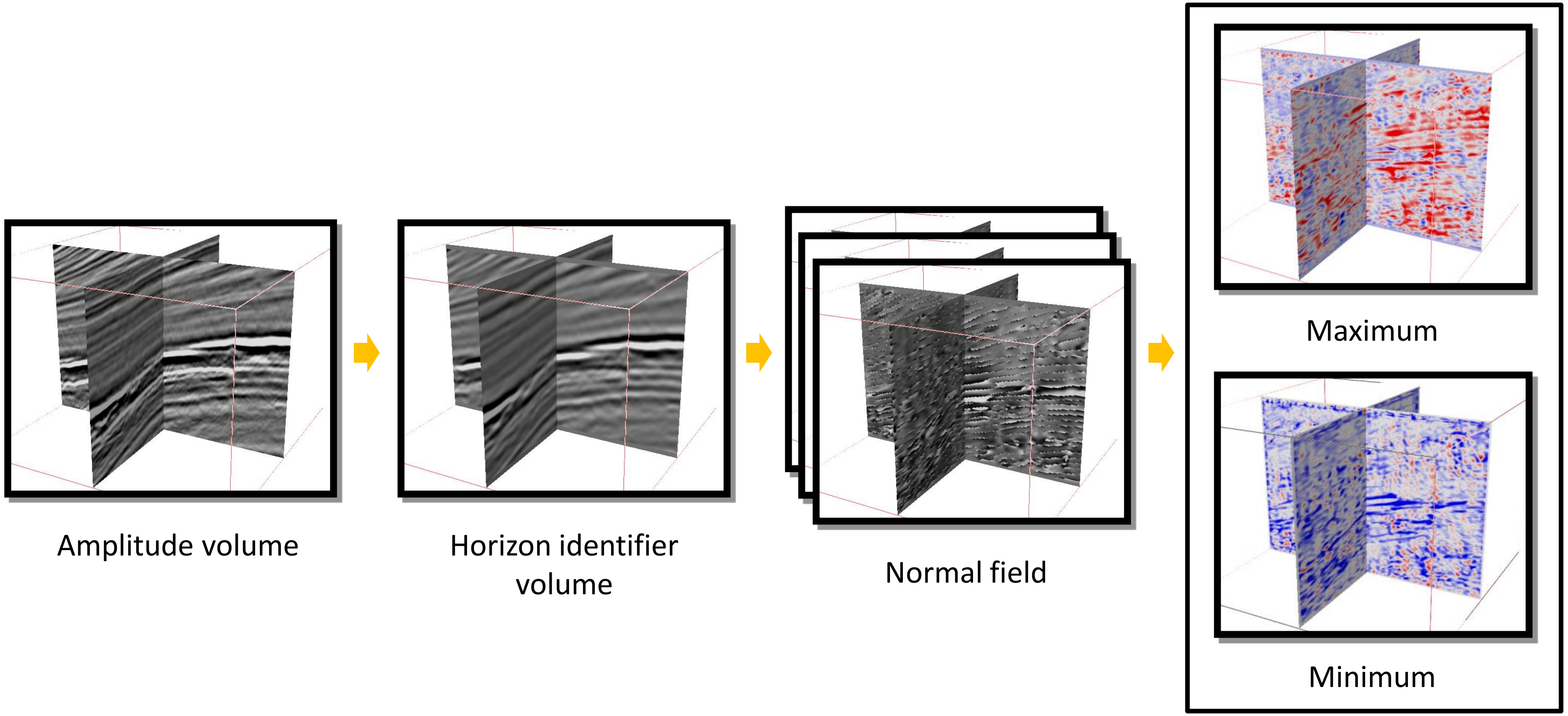


Maximum



Minimum

Volumetric Curvature Estimate



Curvature estimate method

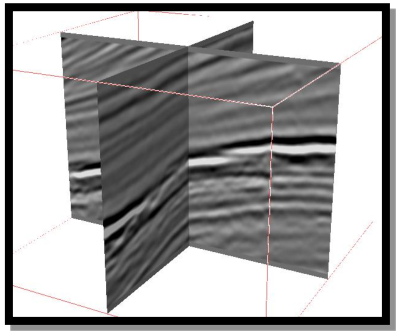
Curvature Attributes

Volumetric Curvature Estimate

- Three steps

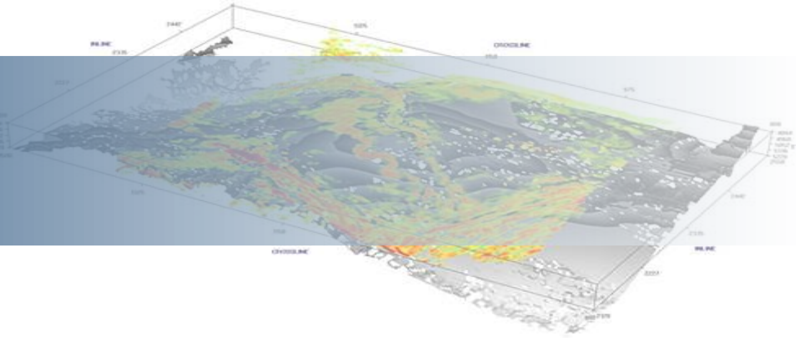
  1º) Computation of horizon identifier attribute

  - Vertical derivative
    - **Improves lateral continuity** of seismic surfaces



Horizon identifier volume

Volumetric Curvature Estimate

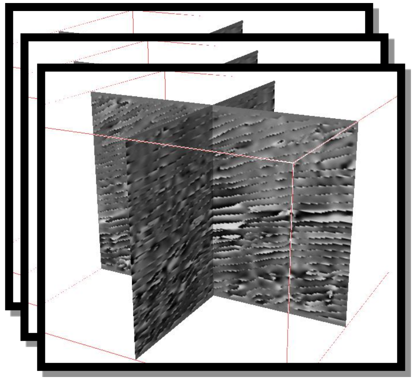


- Three steps

  1º) Computation of horizon identifier attribute

  2º) Normal field estimate

  - Based on the gradient of horizon identifier attribute
  - **Input volume + 3 output** normal volumes



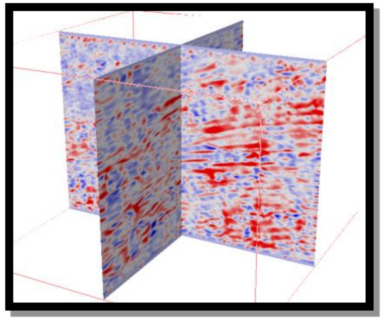
Normal field

$$\nabla F = \begin{pmatrix} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} & \frac{\partial F}{\partial z} \end{pmatrix} = \begin{pmatrix} F_x & F_y & F_z \end{pmatrix}$$

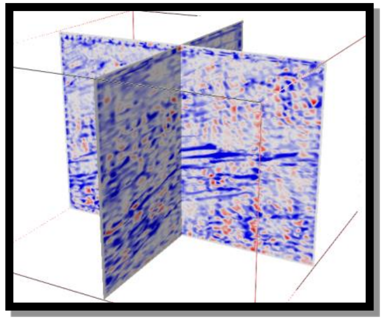
$$N(F) = \begin{cases} \nabla F, & \text{if } F_z \geq 0 \\ -\nabla F, & \text{if } F_z < 0 \end{cases}$$

Volumetric Curvature Estimate

- Three steps
  - 1º) Computation of horizon identifier attribute
  - 2º) Normal field estimate
  - 3º) Curvature estimate
    - Normal field **partial derivatives**



Maximum



Minimum

$$H(T) = \begin{pmatrix} \frac{\partial^2 F}{\partial x^2} & \frac{\partial^2 F}{\partial x \partial y} & \frac{\partial^2 F}{\partial x \partial z} \\ \frac{\partial^2 F}{\partial y \partial x} & \frac{\partial^2 F}{\partial y^2} & \frac{\partial^2 F}{\partial y \partial z} \\ \frac{\partial^2 F}{\partial z \partial x} & \frac{\partial^2 F}{\partial z \partial y} & \frac{\partial^2 F}{\partial z^2} \end{pmatrix} = \begin{pmatrix} F_{xx} & F_{xy} & F_{xz} \\ F_{yx} & F_{yy} & F_{yz} \\ F_{zx} & F_{zy} & F_{zz} \end{pmatrix} = \nabla(\nabla F)$$

$$k_G = \frac{\begin{vmatrix} H^*(F) & N(F)^T \\ N(F) & 0 \end{vmatrix}}{\left|N(F)\right|^4}, k_M = \frac{N(F) * H^*(F) * N(F)^T - \left|N(F)\right|^2 Trace(H^*)}{2\left|N(F)\right|^3}$$

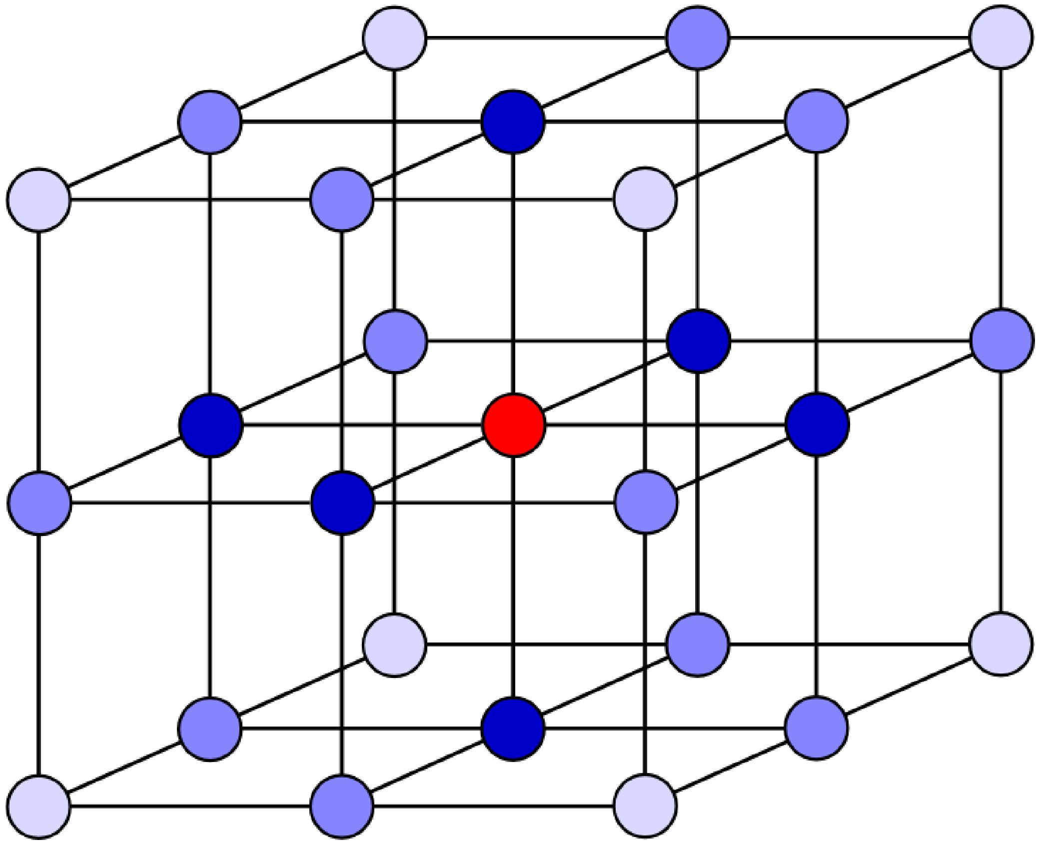
$$\boxed{k = k_M \pm \sqrt{k_M^2 - k_G}}$$

Parallel approach

- **Convolution** of Gaussian derivative filters.
  - Derivative operator size
    - Small: more details, more noise
    - Large: main features, less noise
  - Interpreters usually needs to **vary the derivative operator size** to highlight the features according to theirs needs.

Parallel approach

- **Convolution** of Gaussian derivative filters
- **Stencil computation**
  - Bandwidth-to-compute
  - Data dependency



Data dependency on a 3 x 3 x 3 stencil computation operator

Parallel approach

- Cost of each convolution
  - 3 x 3 x 3 operator: 27 MADDS
  - 5 x 5 x 5 operator: 125 MADDS
  - 13x13x13 operator: 2197 MADDS

- Curvature estimate
  - 9 x 125 MADDs for a 5 x 5 x 5 operator

Parallel approach

- **CPU implementation**
  - OpenMP
  - Compiler: gcc 4.4.7

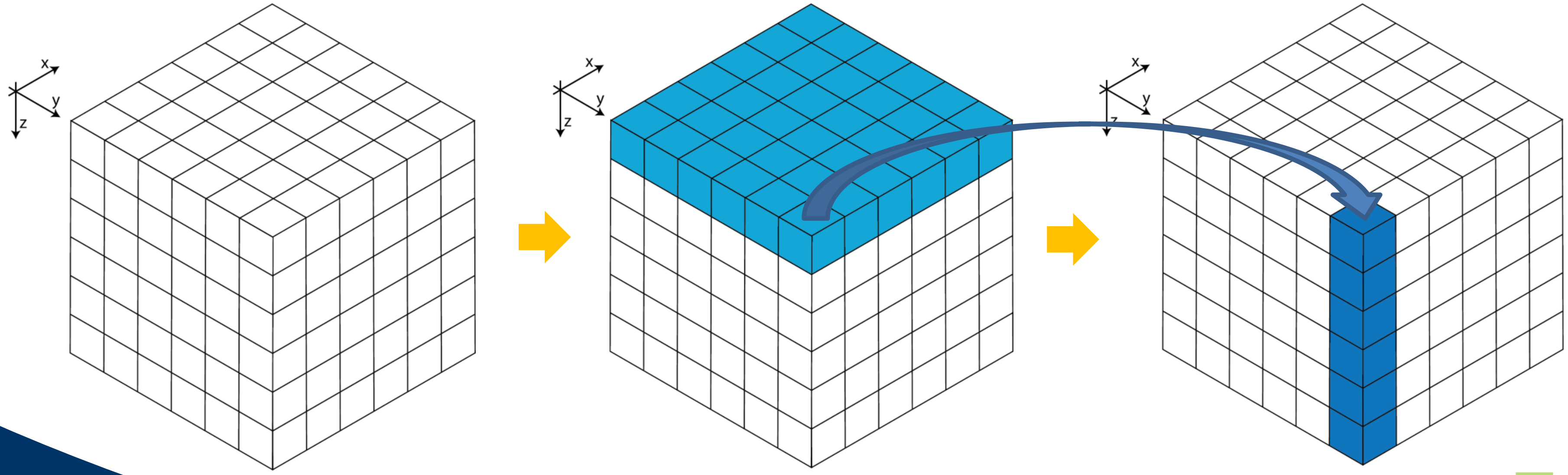
- **GPU implementation**
  - CUDA 6.5
  - Compiler: nvcc

Parallel approach

- **CPU:**
  - **Three loops** to sweep through the volume
  - **Three loops** to sweep through the derivative operator
  - **Blocking** to maximize cache reuse
  - Each thread process a **subset** of blocks
  - Compiler **did not vectorize** the hot spot
  - No manual vectorization with intrinsics

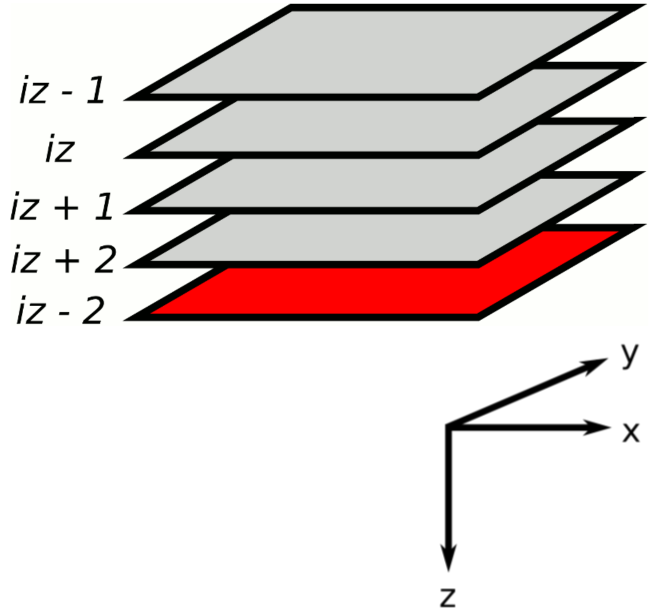
Parallel approach

- **GPU:**
  - Each step in a **different CUDA kernel** (32x16 threads)
  - **60 registers per thread** with no spill
  - Memory access optimization
    - Each thread process a **column of samples**



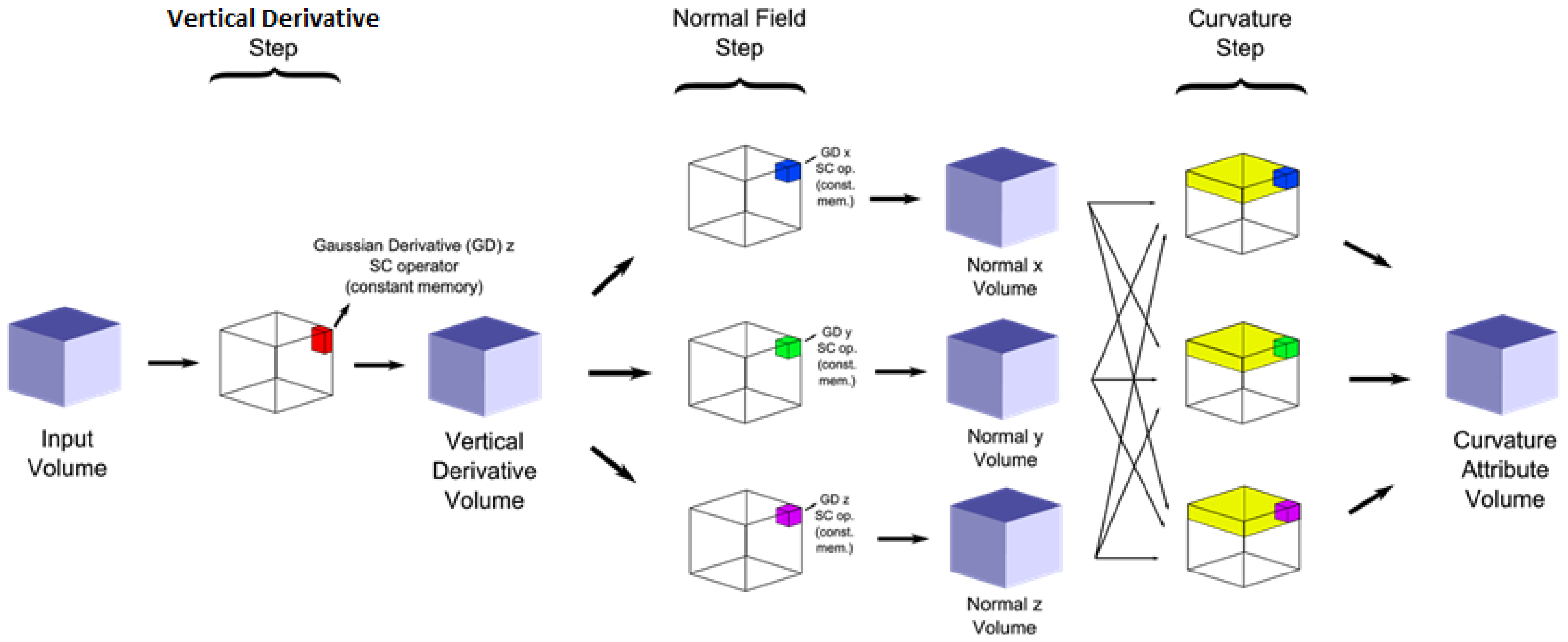
Parallel approach

- 3D shared memory **circular buffer**
  - Based on Paulius Micikevicius (2009) work with RTM (Reverse Time Migration)
  - A single round-robin pointer
  - Operator Size Limited by Shared Mem Per Block



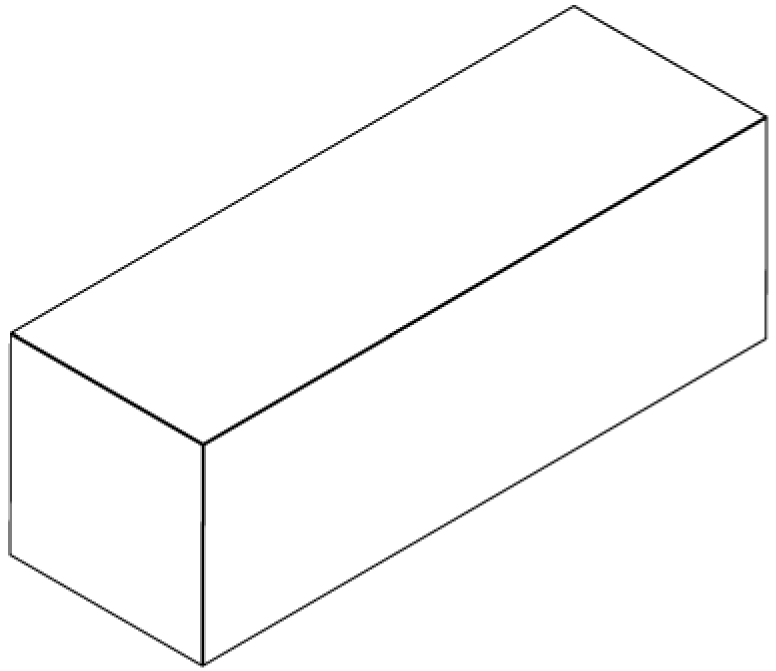
Parallel approach

- Operators and Memory Usage



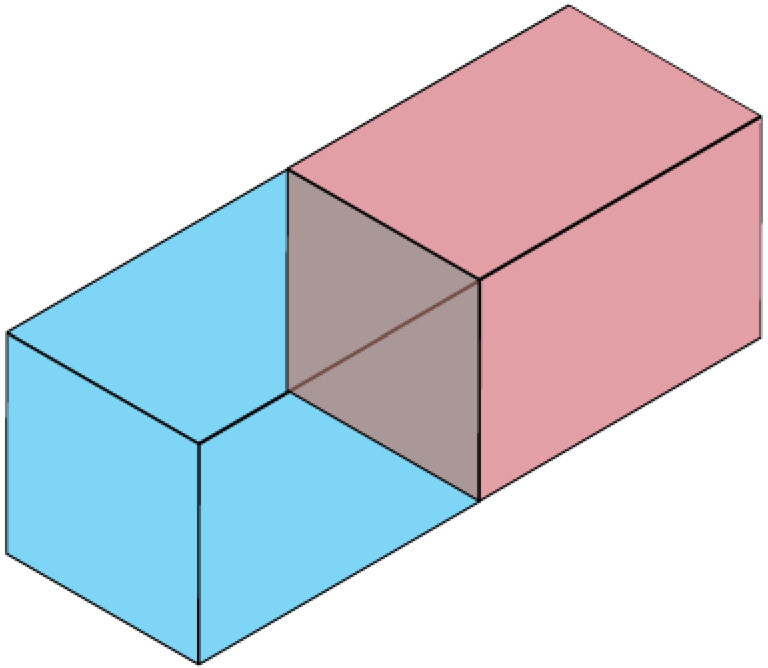
Multi GPU approach

- Split the volume into subvolumes



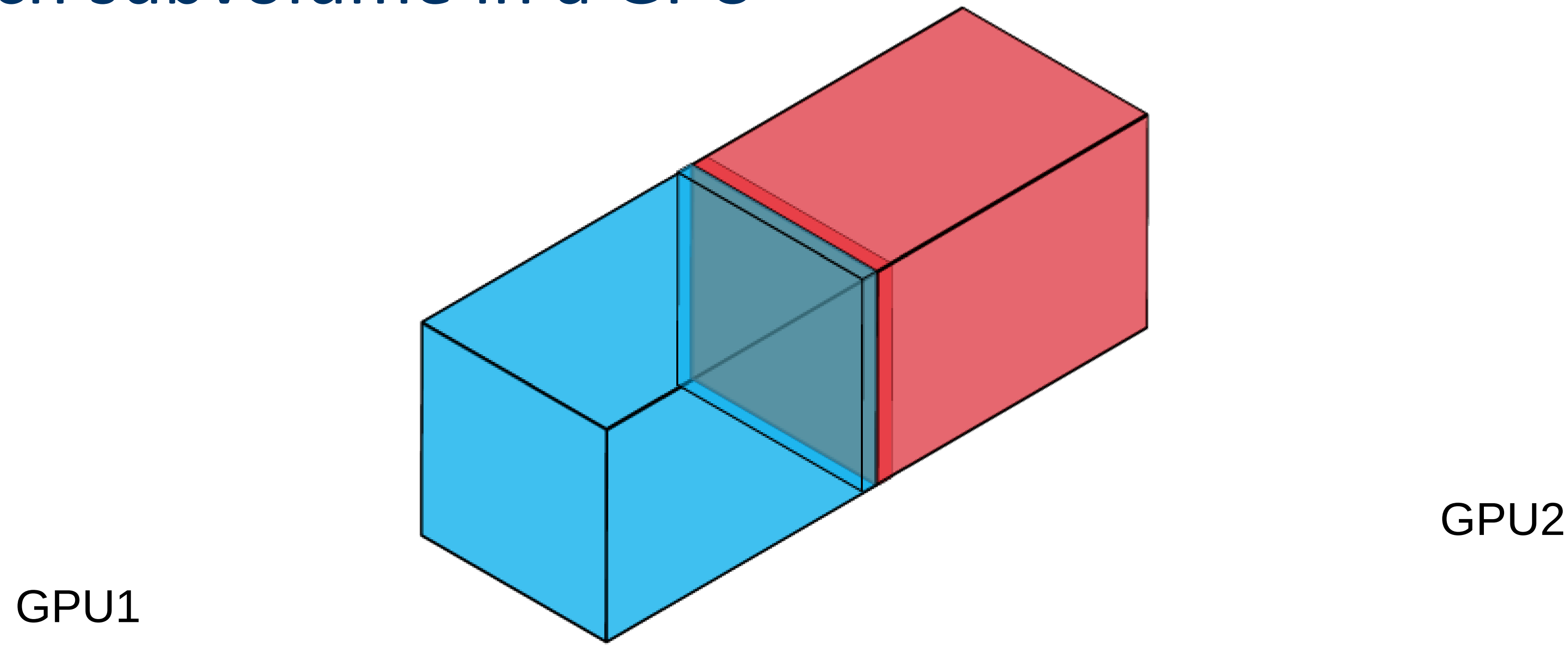
Multi GPU approach

- Split the volume into subvolumes



Multi GPU approach

- Split the volume into subvolumes
  - – Create overlap for edge computations
- Run each subvolume in a GPU



Tecgraf PUC-RIO BR PETROBRAS NVIDIA GPU TECHNOLOGY CONFERENCE

Results

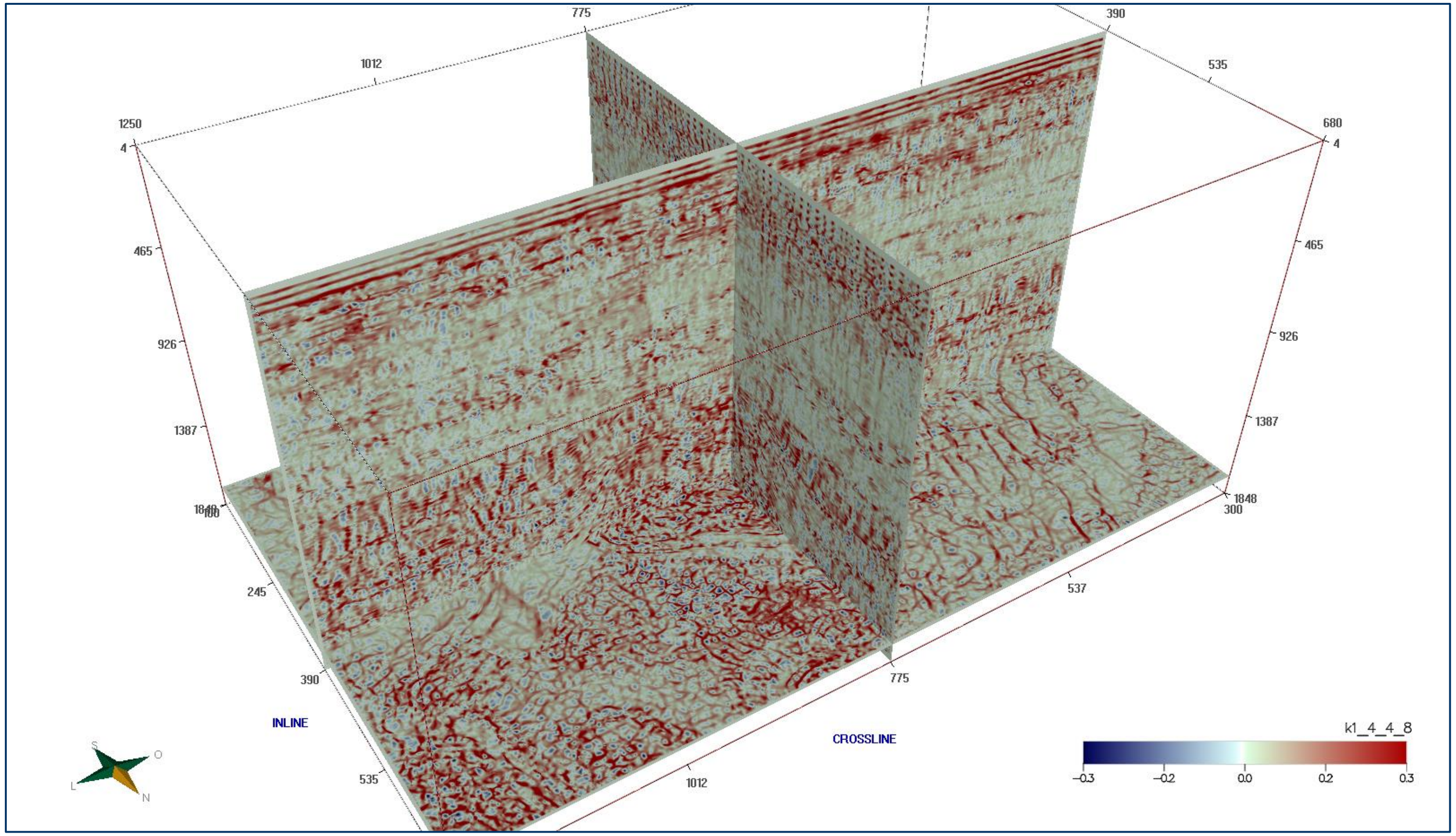
- CPU: i7 3970x
  - 6 cores
- GPU: Tesla K80 Single GPU
  - 2496 cores
  - **3.2 instructions** per clock out of maximum 7.0

Results

- F3 Block
  - North Sea, The Netherlands
  - Resolution: 581 x 951 x 462
  - Seismic file: 1.1 GB
  - https://opendtect.org/osr/pmwiki.php/Main/NetherlandsOffshoreF3BlockComplete4GB

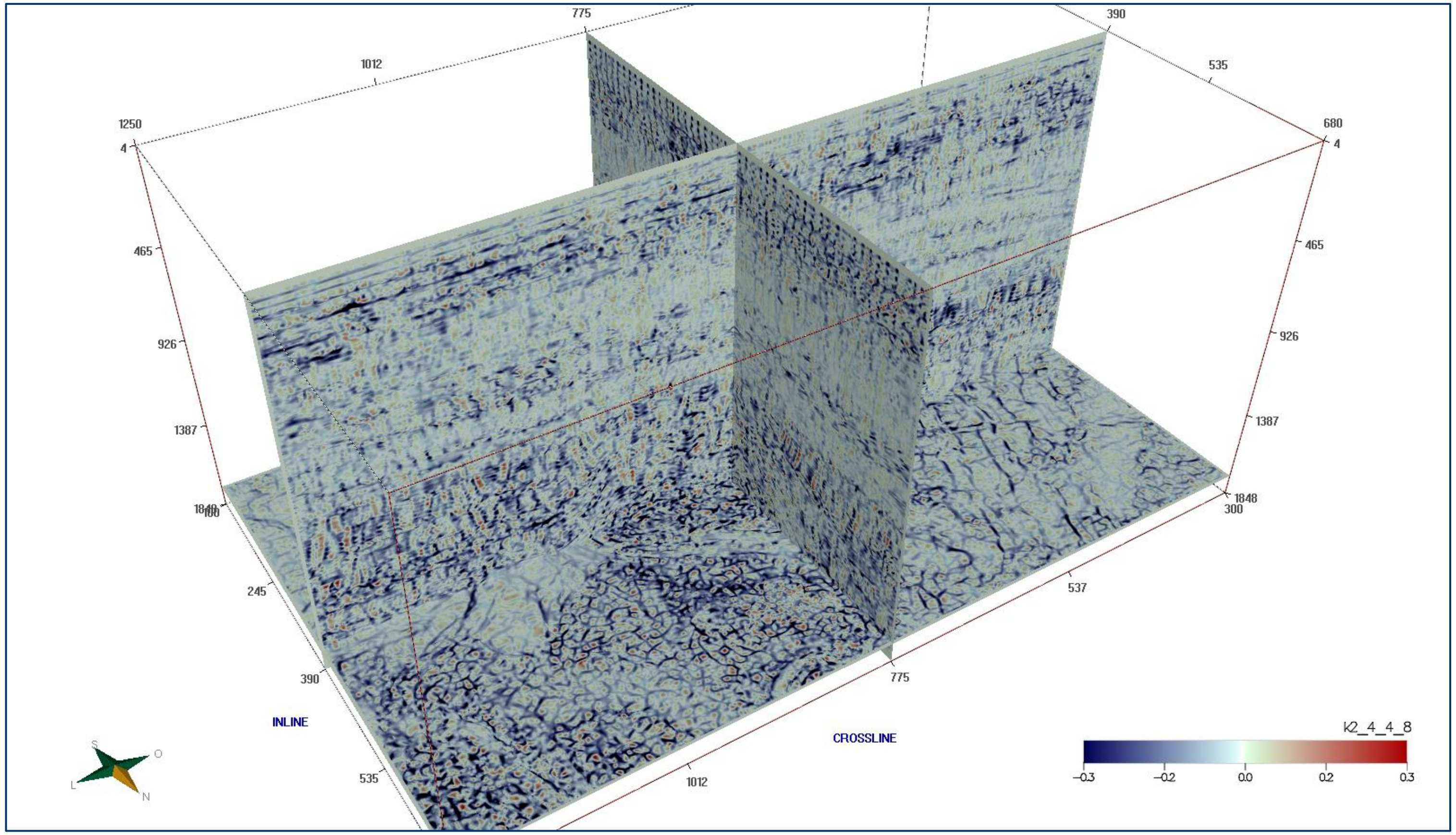
Results

- Maximum Curvature Volume



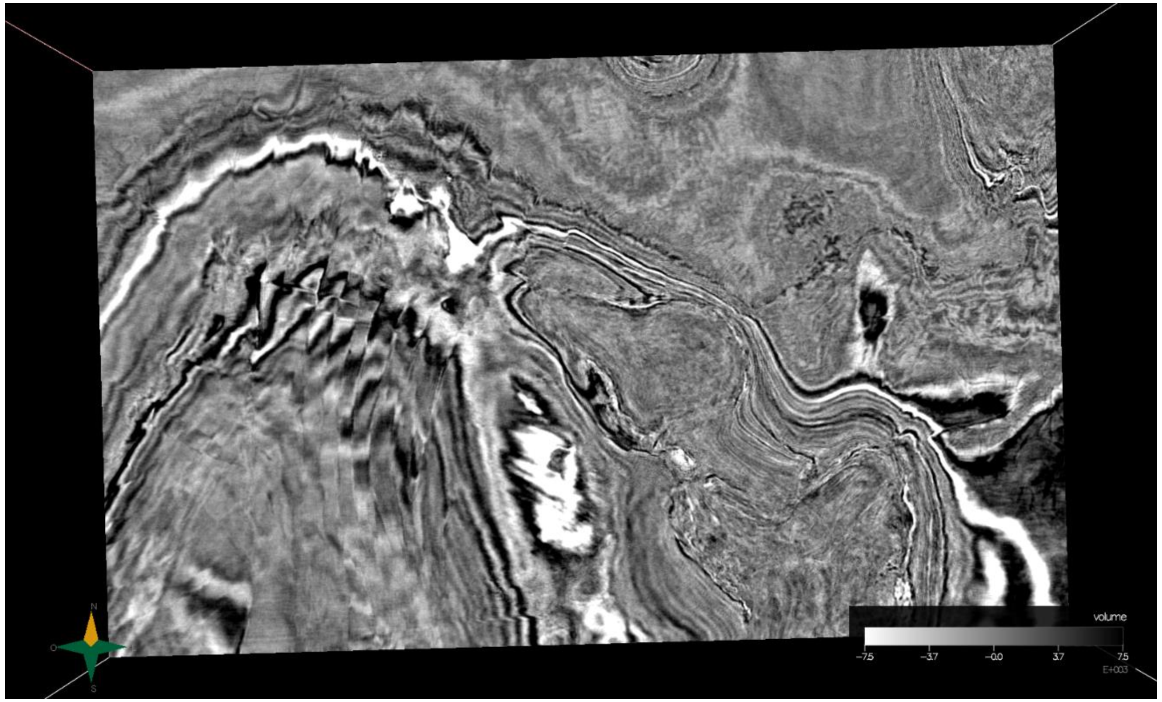
Results

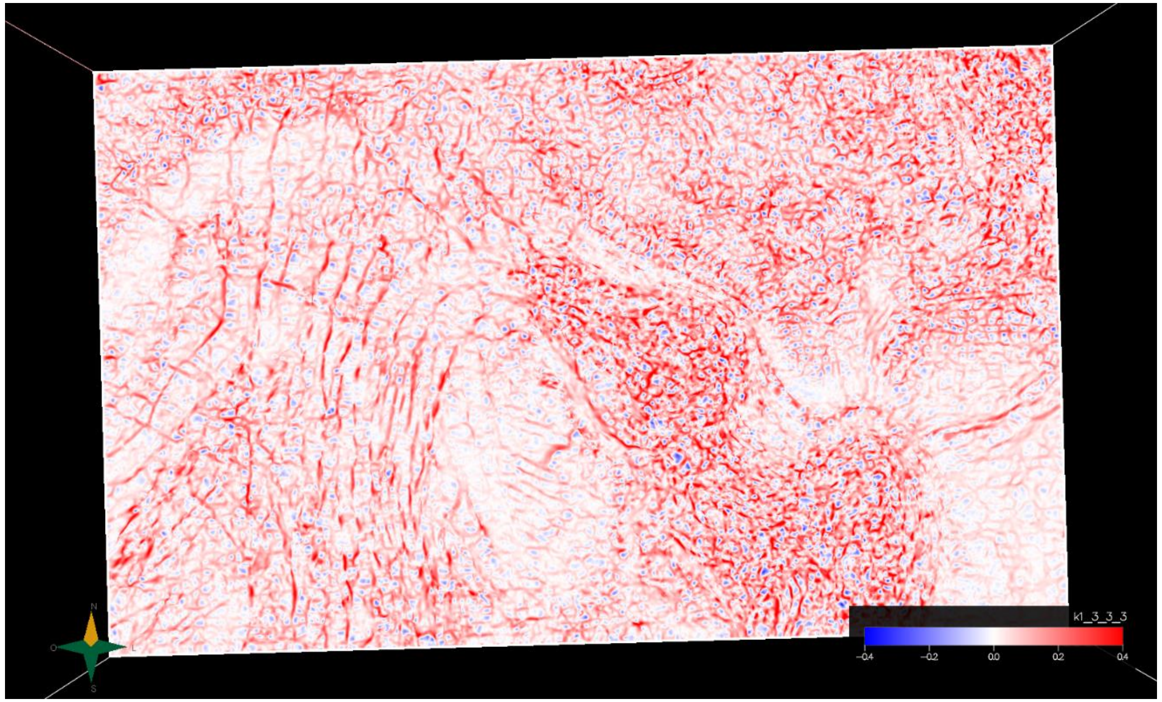
- Minimum Curvature Volume



Results

- Amplitude vs Curvature attribute

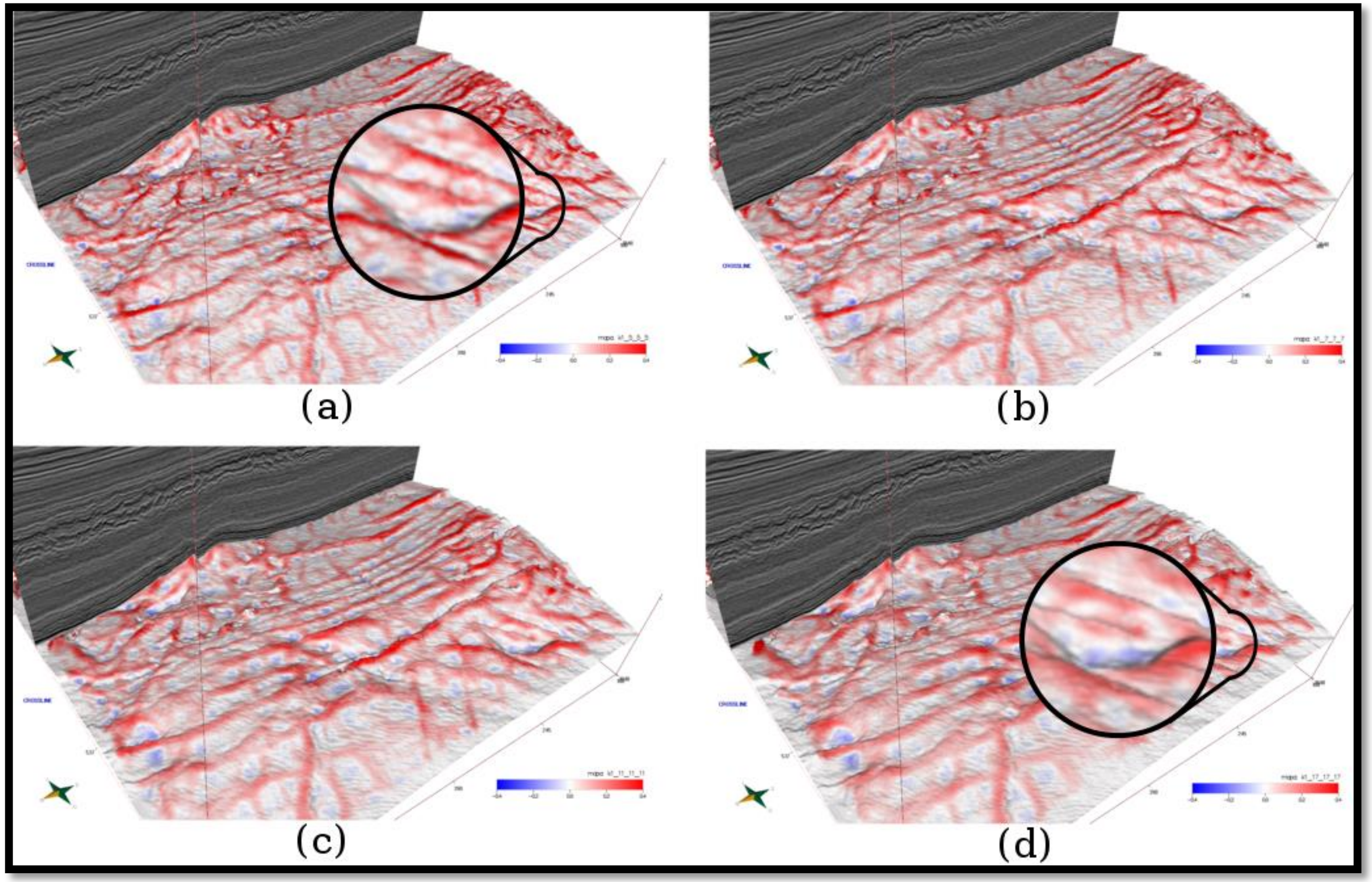




Tecgraf PUC-RIO BR PETROBRAS NVIDIA GPU TECHNOLOGY CONFERENCE

Results

- Operator size effect



(a) 5 x 5 x 5   (b) 7 x 7 x 7   (c) 11 x 11 x 11   (d) 17 x 17 x 17

Results

- CPU x GPU

Operator size	CPU seq. time (s)	CPU with OpenMP time (s)	Single GPU time (s)	Gain
3 x 3 x 3	52.32	9.39	0.51	18.4
5 x 5 x 5	132.40	24.67	0.91	27.1
7 x 7 x 7	313.30	57.89	3.12	18.6
11 x 11 x 11	1095.34	202.36	10.24	19.8
17 x 17 x 17	3832.15	756.29	35.70	21.2

Time spent processing the curvature method
Input volume: F3 Block 1 GB - CPU: i7 3970x - GPU: Tesla K80

Results

- CPU x GPU

Operator size	CPU seq. time (s)	CPU with OpenMP time (s)	Single GPU time (s)	Gain
3 x 3 x 3	52.32	9.39	0.51	18.4
5 x 5 x 5	132.40	24.67	0.91	27.1
7 x 7 x 7	313.30	57.89	3.12	18.6
11 x 11 x 11	1095.34	202.36	10.24	19.8
17 x 17 x 17	3832.15	756.29	35.70	21.2

Even for small volumes of 1GB, at higher operator sizes we can't achieve interactive time.

Results

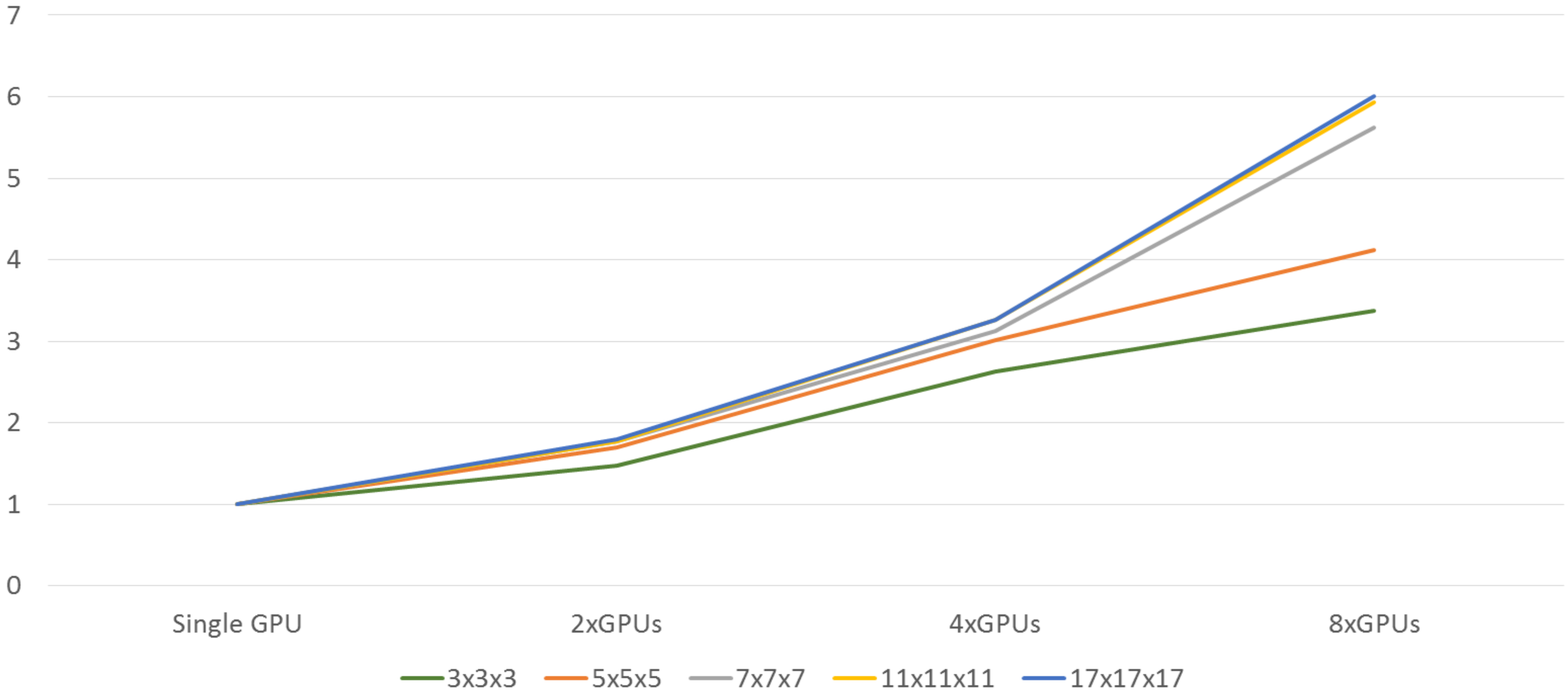
- Multi-GPU

Operator size	Single GPU time (s)	2 x GPUs time (s)	4 x GPUs time (s)	8 x GPUs time (s)
3 x 3 x 3	0.51	0.35	0.20	0.15
5 x 5 x 5	0.91	0.54	0.30	0.22
7 x 7 x 7	3.12	1.76	1.00	0.55
11 x 11 x 11	10.24	5.69	3.14	1.70
17 x 17 x 17	35.80	20.14	10.94	5.97

Time spent processing a volume using multi GPU
Input volume: 1GB – Resolution: 581 x 951 x 462
GPU: Tesla K80

Results

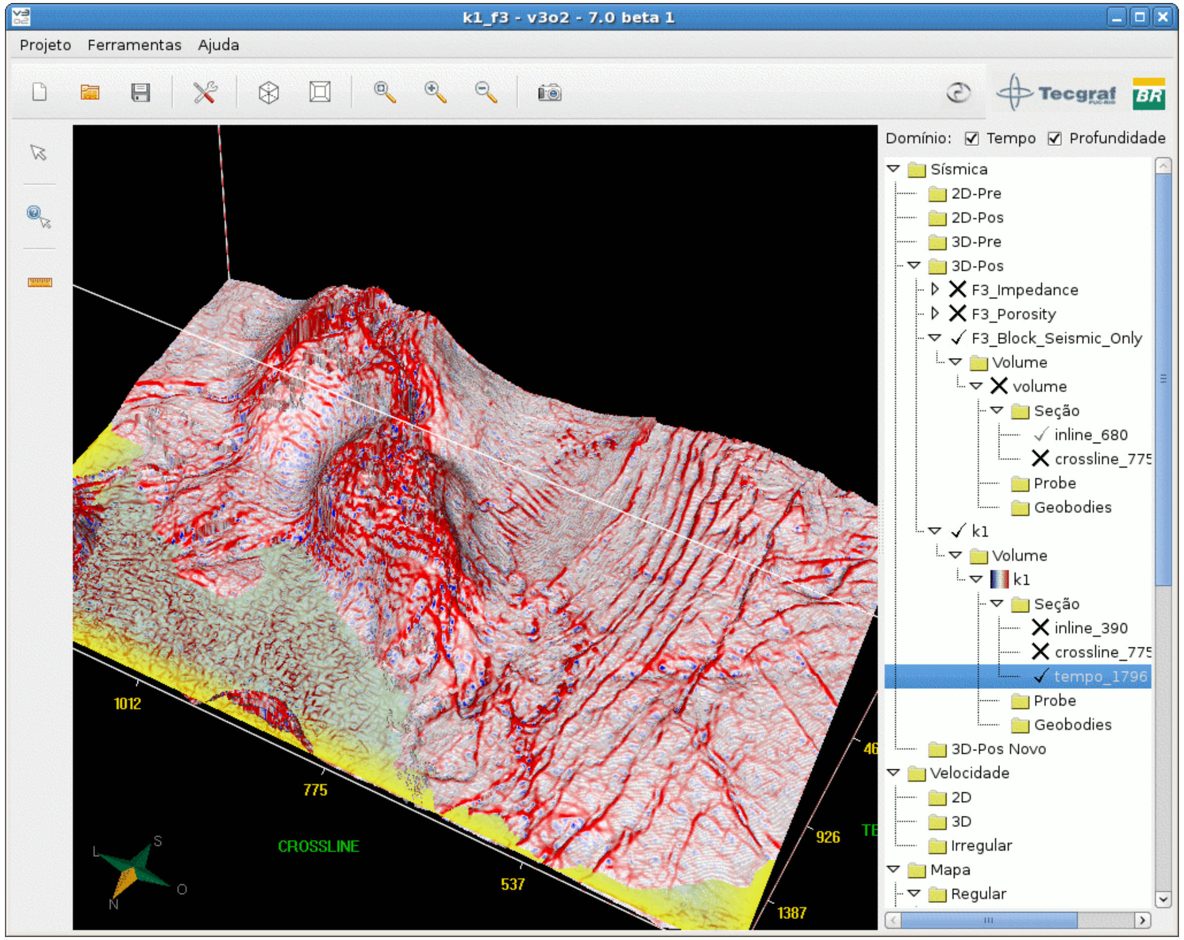
- Speedup comparison using Multi-GPUs



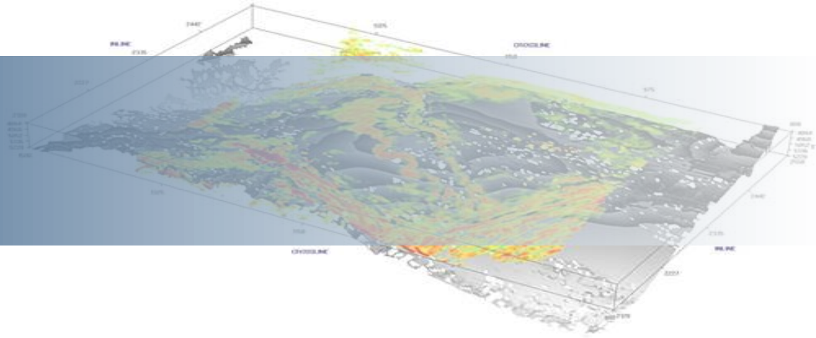
Input volume: 1GB – GPU: Tesla K80

Results

- Processing a single slice of a 1GB volume



Results



- Single slice

Operator size	CPU with OpenMP time (ms)	Single GPU time (ms)	Gain
3 x 3 x 3	110	7	15.71x
5 x 5 x 5	341	13	26.23x
7 x 7 x 7	1012	38	26.63x
11 x 11 x 11	5109	102	50.08x
17 x 17 x 17	26203	501	52.30x

Time spent processing a single inline slice
Input volume: 1GB – Inline resolution: 951 x 462
CPU: i7 3970x - GPU: Tesla K80

Conclusion

- The use of a **massively parallel** architecture to calculate the curvature attribute can lead to **great improvement**
- GPU architecture fits better to stencil computation compared to CPU
- The use of domain characteristics, can avoid compute a lot of data that might never be used

Acknowledgements

Questions???