SES3D.inv

ELEMENTS

SPECTRAL

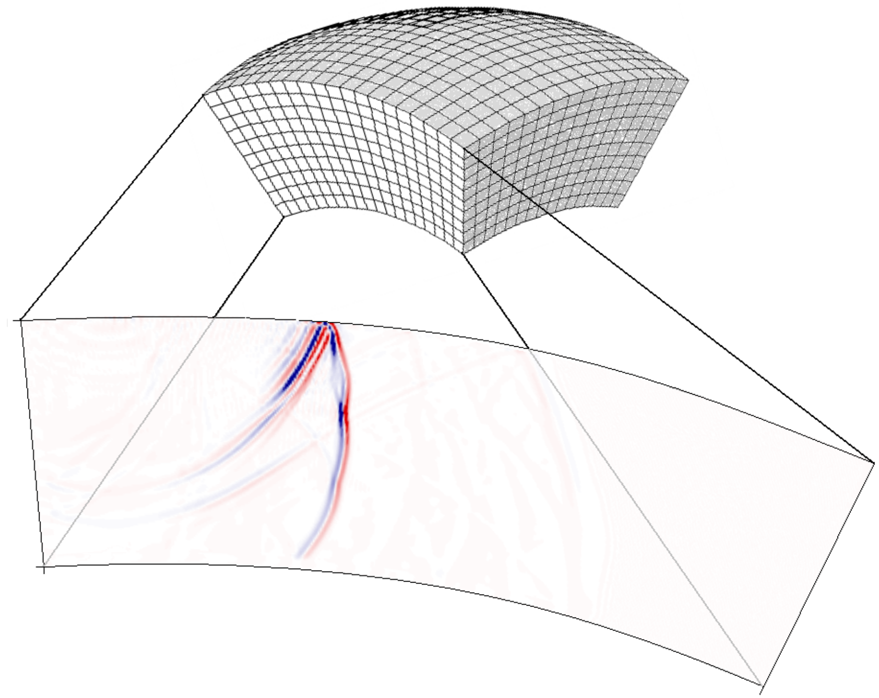
SPHERICAL

BY A FICHTNER

a programme package for seismic waveform modelling and inversion

SES3D.inv

BASIC CHARACTERISTICS

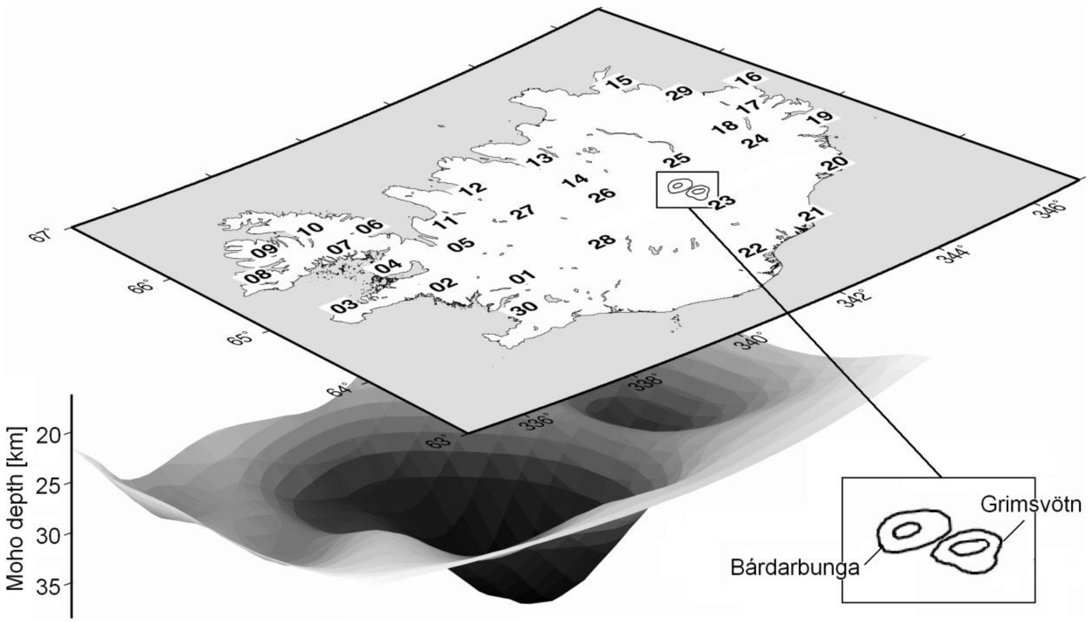


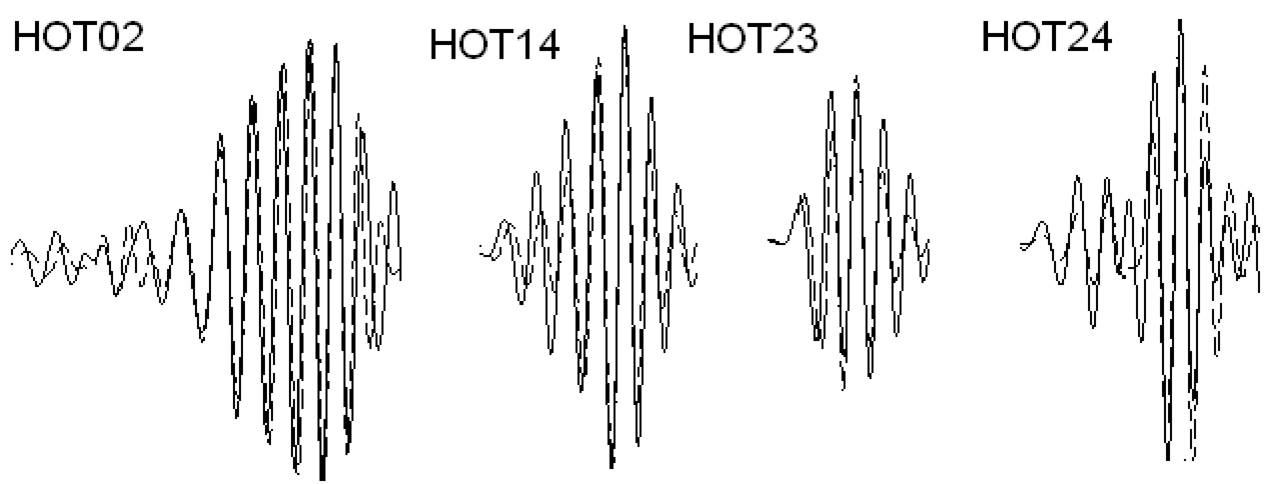
- seismic wave propagation through spherical sections
- including anisotropy and visco-elastic dissipation
- 3D heterogeneities of all Earth model properties
- inversion of seismic observables for 3D Earth structure

A collection of recent applications

A collection of recent applications

modelling volcanic earthquakes on Iceland





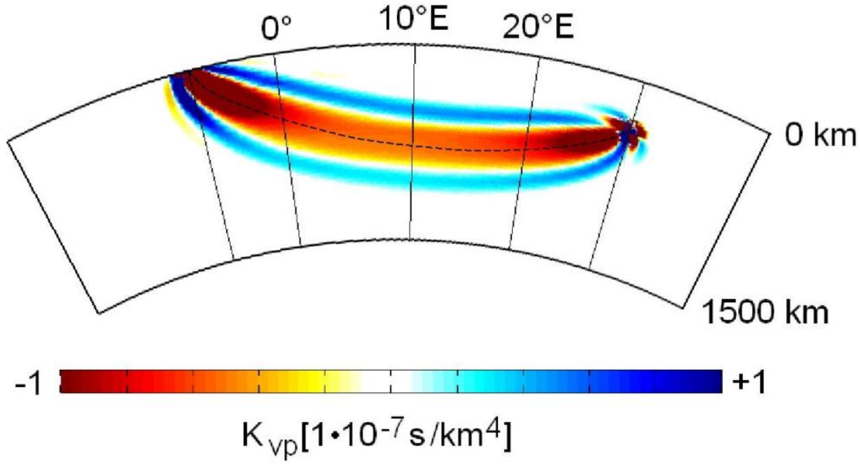
data
synthetic

period: 5 s

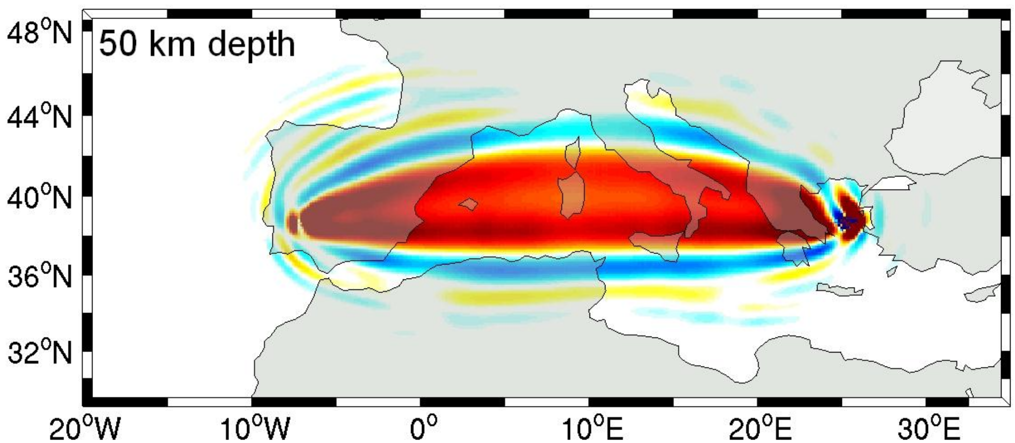
A collection of recent applications

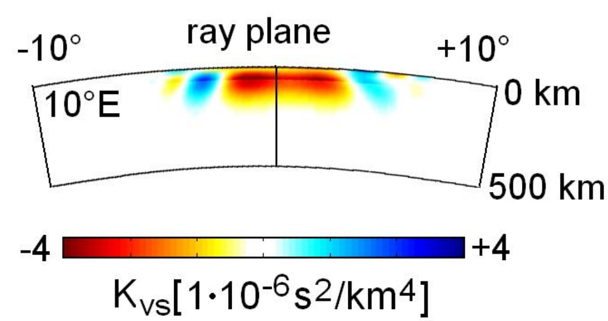
sensitivity analysis:
How is my observable affected by Earth structure?

15 s P wave



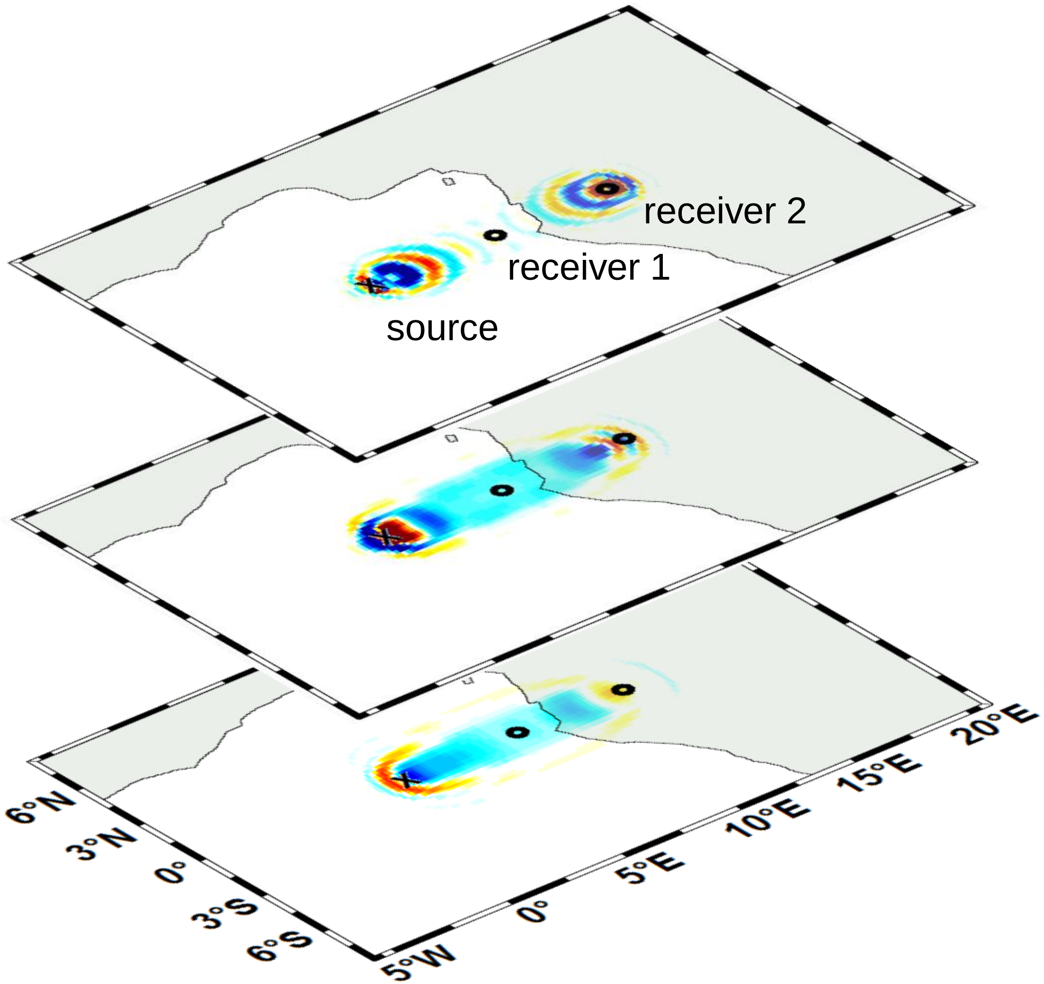
50 s Rayleigh wave





A collection of recent applications

**sensitivity analysis:
How is my observable affected
by Earth structure?**

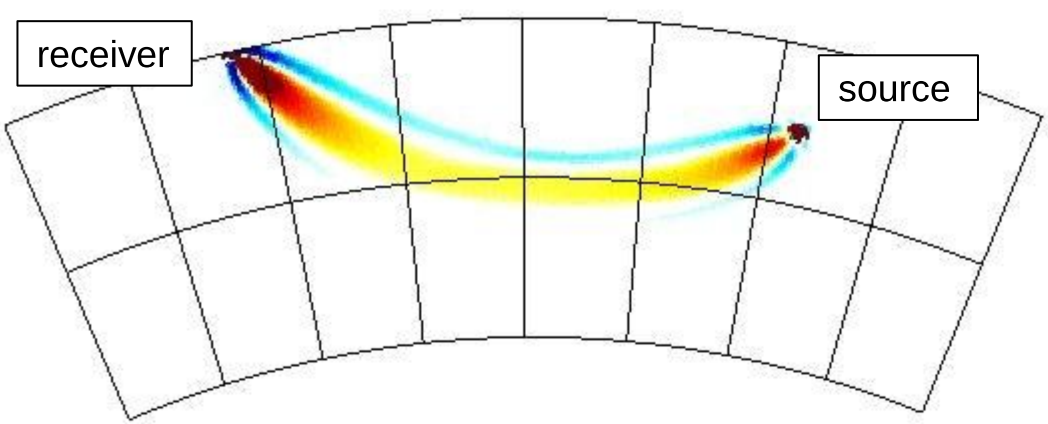


- Cross-correlation measurements for regional scale tomography.
- Validation of the 2-station method.

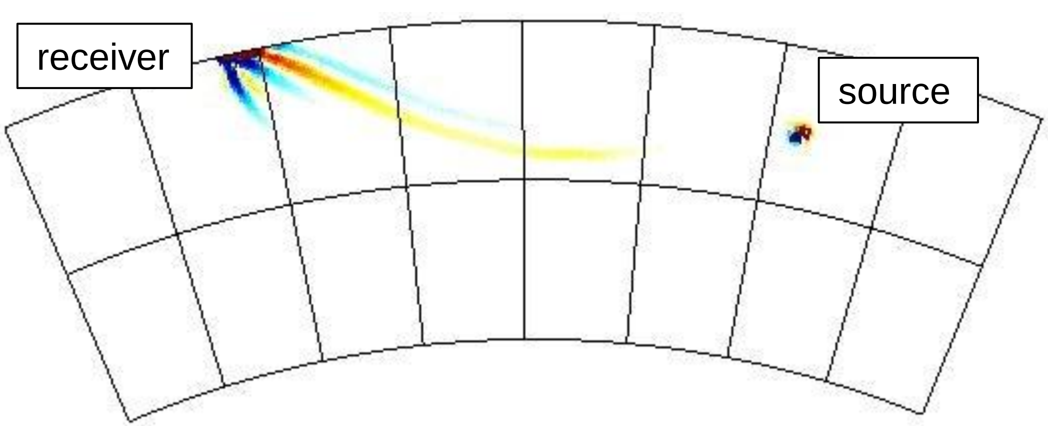
by **Denise de Vos**

A collection of recent applications

sensitivity analysis: How is my observable affected by Earth structure?



$$S-wave\,amplitude$$

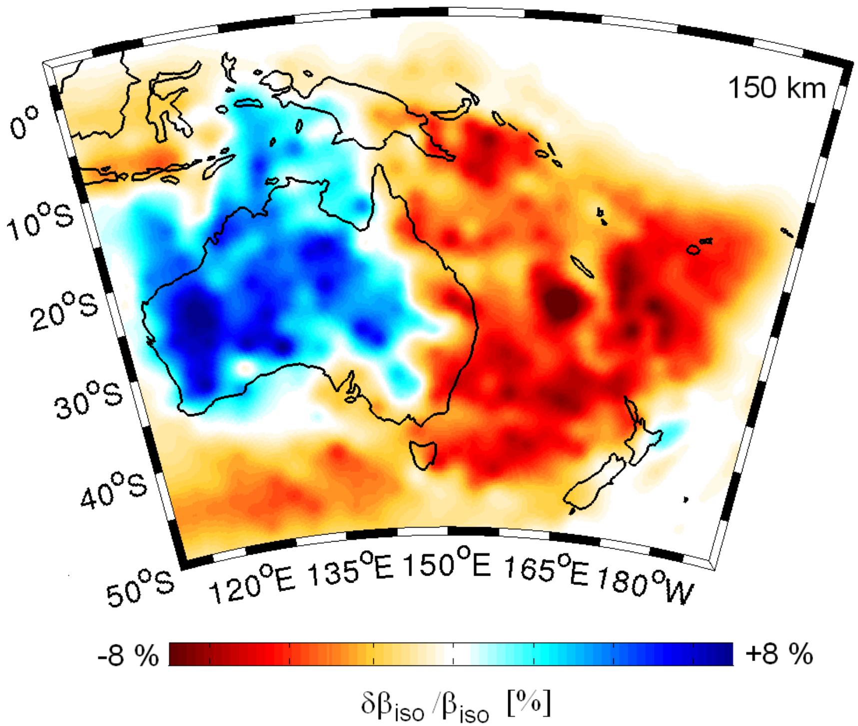


$$apparent\,S\,velocity = \frac{\|\,velocity\,\|}{\|\,rotation\,\|}$$

by **Moritz Bernauer**

A collection of recent applications

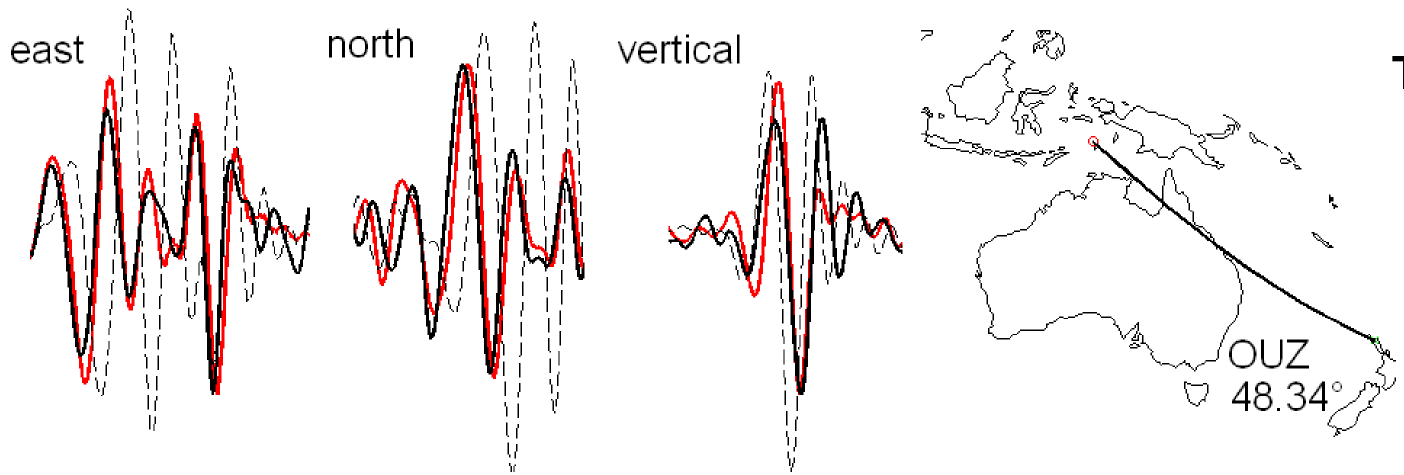
waveform inversion for 3D Earth structure



data

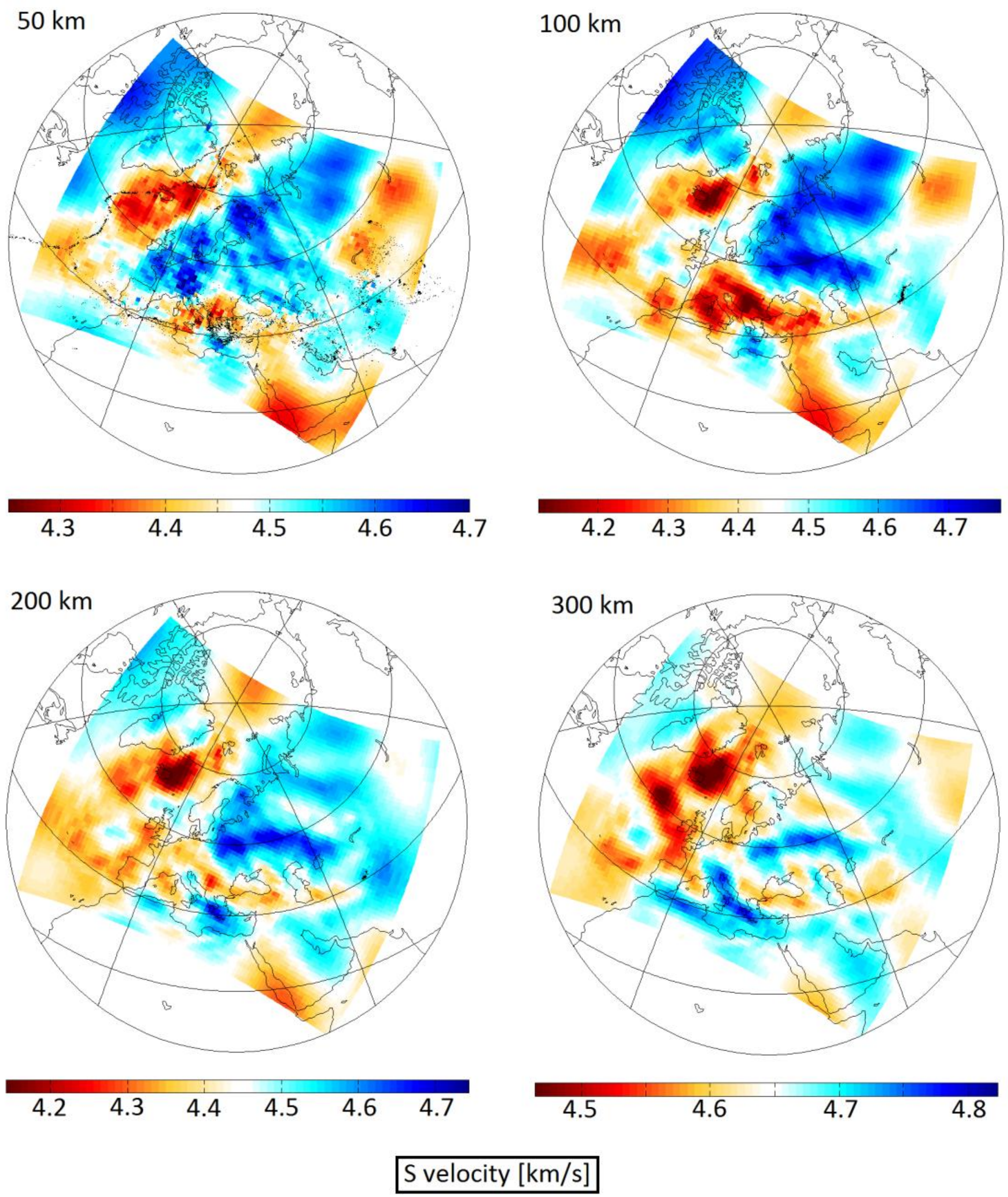
initial model

final model



T = 30 s

A collection of recent applications



waveform inversion for 3D Earth structure

<u>Tomography is a 3-stage process:</u>

(1) forward problem

(2) sensitivities

(3) optimisation

SES3D.inv

forward problem	sensitivities	inversion

forward problem

- seismic wave propagation through heterogeneous Earth models
- dissipation & anisotropy
- spectral-element disctretisation of the seismic wave equation

sensitivities

- how does an observable react to changes of the Earth model?
- Fréchet (sensitivity) kernels
- adjoint techniques & time reversal

inversion

- finding good obervables
- the inversion grid and kernel vs. gradient
- local minima and the multi-scale approach
- regularisation
- non-linear optimisation

SES3D.inv

forward problem	sensitivities	inversion

forward problem

- seismic wave propagation through heterogeneous Earth models
- dissipation & anisotropy
- spectral-element disctretisation of the seismic wave equation

sensitivities

- how does an observable react to changes of the Earth model?
- Fréchet (sensitivity) kernels
- adjoint techniques & time reversal

inversion

- finding good obervables
- the inversion grid and kernel vs. gradient
- local minima and the multi-scale approach
- regularisation
- non-linear optimisation

SES3D.inv

forward problem	sensitivities	inversion

- seismic wave propagation through heterogeneous Earth models
- dissipation & anisotropy
- spectral-element disctretisation of the seismic wave equation

- how does an observable react to changes of the Earth model?
- Fréchet (sensitivity) kernels
- adjoint techniques & time reversal

- finding good obervables
- the inversion grid and kernel vs. gradient
- local minima and the multi-scale approach
- regularisation
- non-linear optimisation

SES3D.inv

forward problem	sensitivities	inversion
seismic wave propagation through heterogeneous Earth models	how does an observable react to changes of the Earth model?	finding good obervables
dissipation & anisotropy	Fréchet (sensitivity) kernels	the inversion grid and kernel vs. gradient
spectral-element disctretisation of the seismic wave equation	adjoint techniques & time reversal	local minima and the multi-scale approach
		regularisation
		non-linear optimisation
Thursday morning	Thursday afternoon	Friday morning

PART I

NUMERICAL SOLUTION OF THE SEISMIC WAVE EQUATION

1. Introduction
2. Spectral-element discretisation
3. Practical
4. Special topics (dissipation, point sources, absorbing boundaries)

Introduction

Introduction

density

elastic tensor

external forces

$$\boxed{\rho\, \ddot{u}_i(x,t) - \partial_j [C_{ijkl}(x,t) * \partial_k u_l(x,t)] = f_i(x,t)}$$

elastic displacement field

Introduction

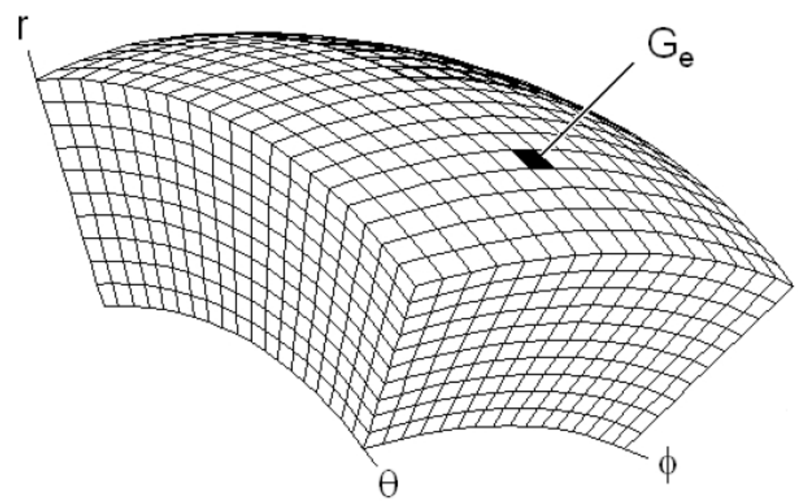
density — elastic tensor — external forces

$$\rho\, \ddot{u}_i(x,t) - \partial_j [C_{ijkl}(x,t) * \partial_k u_l(x,t)] = f_i(x,t)$$

elastic displacement field

SES3D.inv

- **spectral-elements in a spherical section**
- **operates in natural spherical coordinates**
- **visco-elastic dissipation**
- **radial anisotropy**
- **absorbing boundaries: PML**



Introduction

Advantages	Disadvantages
• accurate solutions	• no poles & no core
• easy to use	• only regular-shaped elements
• runs fast when used correctly	• elements become smaller with depth
• code is simple and easy to modify	• no topography
• robust	• no fluid-solid interaction

SES3D.inv is very efficient for a very specific class of problems:

wave propagation on local to continental scales where topography can be ignored

Spectral-element discretisation

Spectral-element discretisation

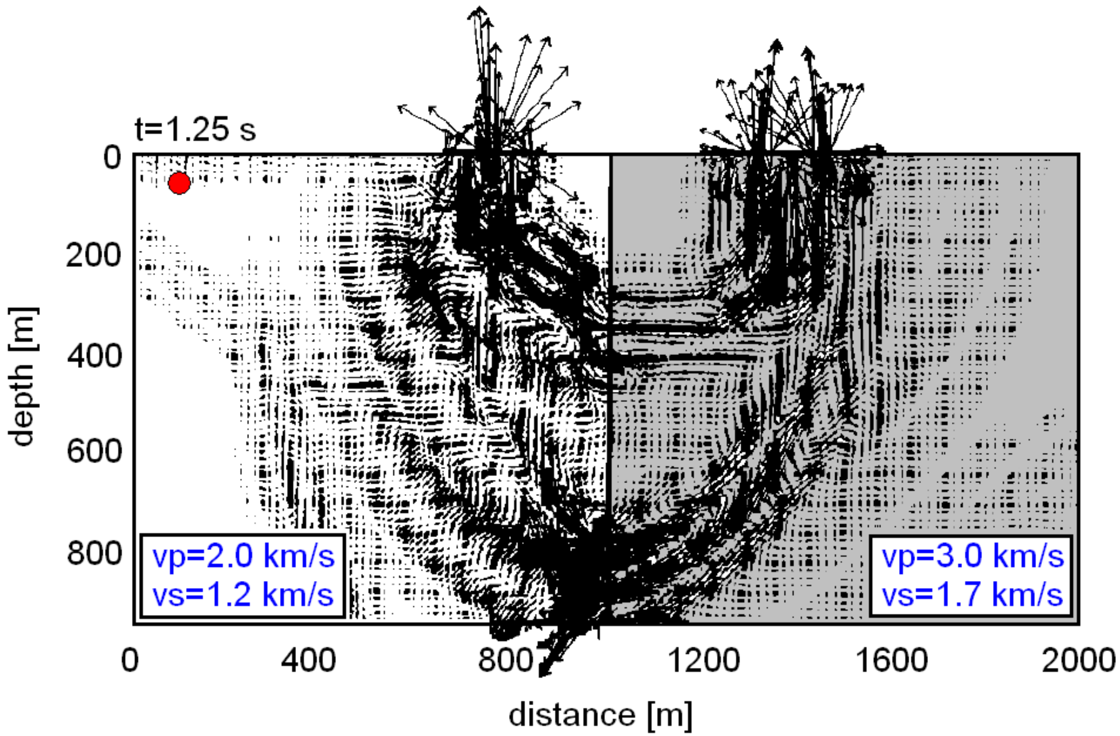
The SEM origins

- originally developed in fluid dynamics (Patera, 1984)

- migrated to seismology in the early 1990‘s (Seriani & Priolo, 1991)

- major advantage: accurate modelling of interfaces and the free surface (with topography)

reflection and transmission of surface waves at material discontinuities



Priolo, Carcione & Seriani: 1994

Spectral-element discretisation

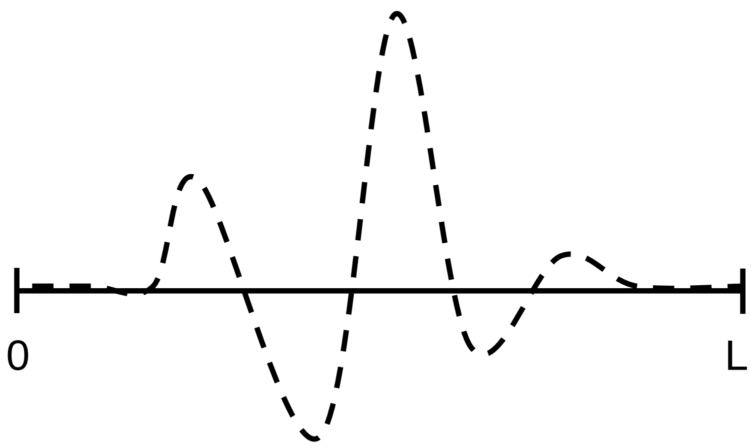
The SEM concept in 1D: Weak form of the wave equation

SEM is based on the weak form of the wave equation

$$\rho\,\ddot{u} - \frac{\partial}{\partial x}\left(\mu \frac{\partial}{\partial x} u\right) = f \qquad \frac{\partial}{\partial x} u(t,0) = \frac{\partial}{\partial x} u(t,L) = 0$$

----------- PDE -----------   -- B.C. (free surface in 3D) --

strong form of the wave equation



vibrating string of length L

Spectral-element discretisation

The SEM concept in 1D: Weak form of the wave equation

SEM is based on the weak form of the wave equation

$$\rho\,\ddot{u} - \frac{\partial}{\partial x}\left(\mu \frac{\partial}{\partial x} u\right) = f \qquad \frac{\partial}{\partial x} u(t,0) = \frac{\partial}{\partial x} u(t,L) = 0$$

----------- PDE -----------   -- B.C. (free surface in 3D) --

strong form of the wave equation

$$\int_0^L \rho\, w\, \ddot{u}\, dx - \int_0^L w \frac{\partial}{\partial x}\left(\mu \frac{\partial}{\partial x} u\right) dx = \int_0^L w\, f\, dx$$

multiply with test function w(x)

integrate over x from 0 to L

Spectral-element discretisation

The SEM concept in 1D: Weak form of the wave equation

SEM is based on the weak form of the wave equation

$$\rho\,\ddot{u} - \frac{\partial}{\partial x}\left(\mu \frac{\partial}{\partial x} u\right) = f \qquad \frac{\partial}{\partial x} u(t,0) = \frac{\partial}{\partial x} u(t,L) = 0$$

----------- PDE ----------- -- B.C. (free surface in 3D) --

strong form of the wave equation

$$\int_0^L \rho\, w\, \ddot{u}\, dx - \int_0^L w \frac{\partial}{\partial x}\left(\mu \frac{\partial}{\partial x} u\right) dx = \int_0^L w\, f\, dx$$

multiply with test function w(x)

integrate over x from 0 to L

$$\int_0^L \rho\, w\, \ddot{u}\, dx + \int_0^L \mu \left(\frac{\partial}{\partial x} w\right)\left(\frac{\partial}{\partial x} u\right) dx = \int_0^L w\, f\, dx$$

integrate by parts

use the boundary conditions

The SEM concept in 1D: Weak form of the wave equation

Solving the weak form of the wave equation means

to find a displacement field u(x,t) such that

$$\boxed{\int_0^L \rho\, w\, \ddot{u}\, dx + \int_0^L \mu \left(\frac{\partial}{\partial x} w \right) \left(\frac{\partial}{\partial x} u \right) dx = \int_0^L w\, f\, dx}$$

is satisfied for any differentiable test function w(x).

The SEM concept in 1D: Weak form of the wave equation

<u>Solving the weak form of the wave equation means</u>

to find a displacement field u(x,t) such that

$$\boxed{\int_0^L \rho\, w\, \ddot{u}\, dx + \int_0^L \mu \left(\frac{\partial}{\partial x} w \right) \left(\frac{\partial}{\partial x} u \right) dx = \int_0^L w\, f\, dx}$$

is satisfied for any differentiable test function w(x).

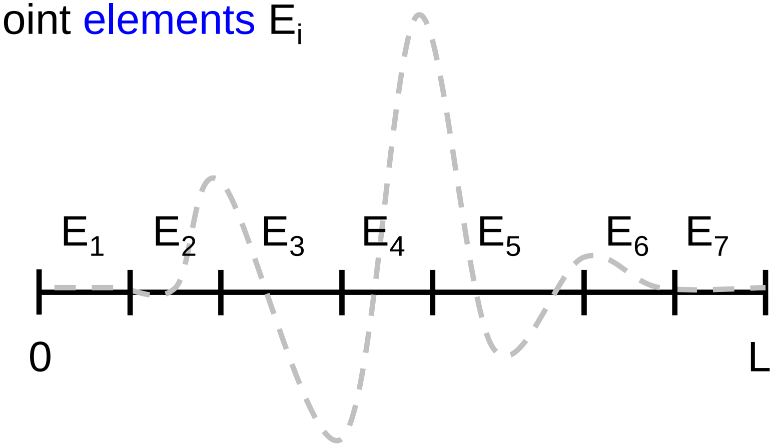
- **Basis of element-based methods** (e.g. finite elements, spectral elements, discontinuous Galerkin)
- **Boundary conditions are implicitly satisfied** (no additional work required, as in finite-difference methods)

Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

1. decompose the computational domain [0, L] into disjoint elements E_i

2. consider integral element-wise



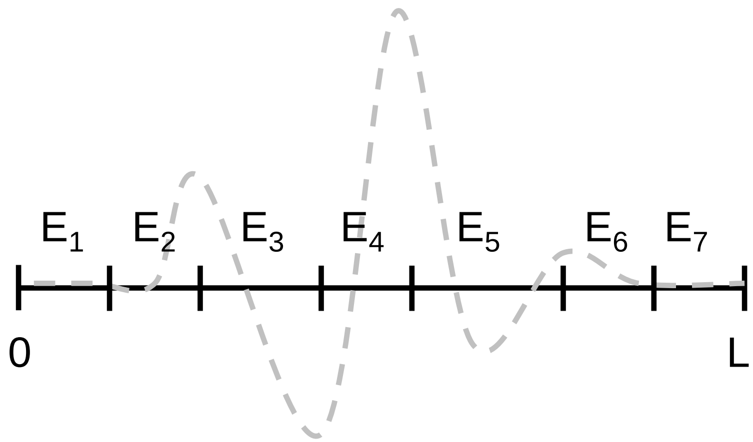
$$\int_{E_i} \rho\, w\, \ddot{u}\, dx + \int_{E_i} \mu \left(\frac{\partial}{\partial x} w\right)\left(\frac{\partial}{\partial x} u\right) dx = \int_{E_i} w\, f\, dx$$

E_i

Spectral-element discretisation

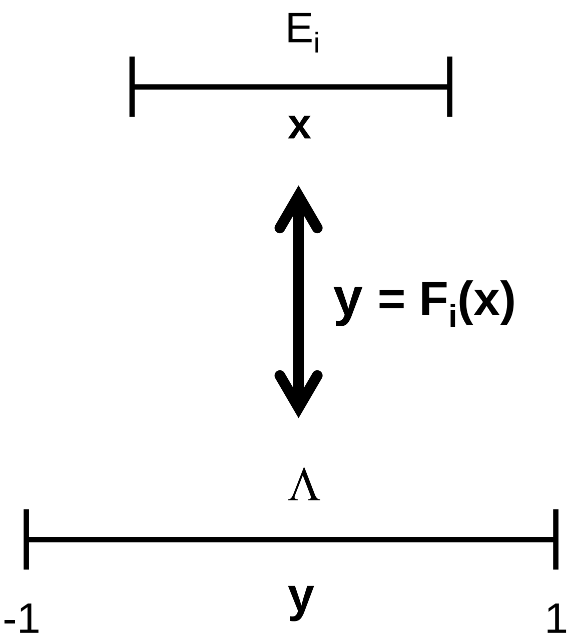
The SEM concept in 1D: Spatial discretisation

3. map each element to the reference interval [-1, 1]



$$\boxed{\int_{E_i} \rho\, w\, \ddot{u}\, dx + \int_{E_i} \mu \left(\frac{\partial}{\partial x} w\right)\left(\frac{\partial}{\partial x} u\right) dx = \int_{E_i} w\, f\, dx}$$

E_i



Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

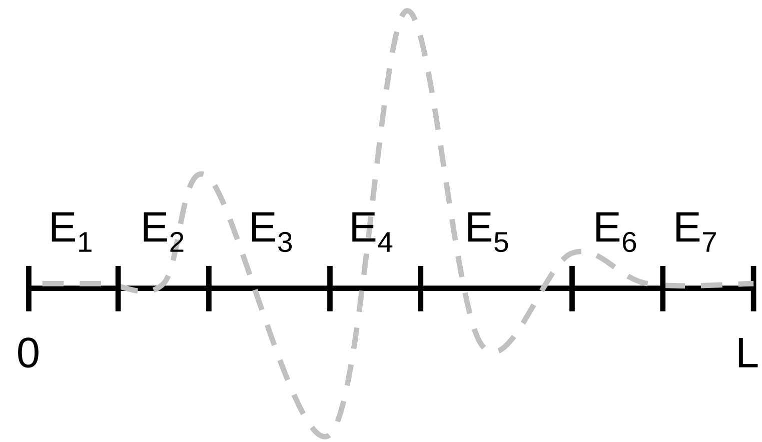
Is the same for every element !!!

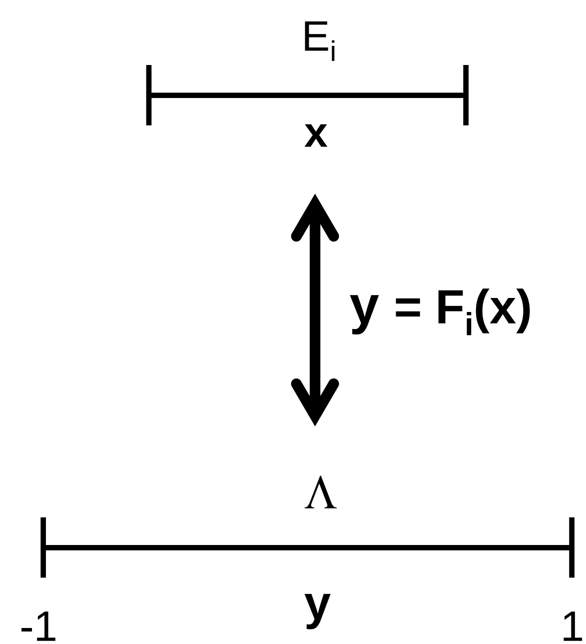
All elements can now be treated in the same way.

$$\int_{-1}^{+1} \rho\, w\, \ddot{u}\, \frac{dx}{dy}\, dy + \dots$$

$$\int_{E_i} \rho\, w\, \ddot{u}\, dx + \int_{E_i} \mu \left(\frac{\partial}{\partial x} w\right)\left(\frac{\partial}{\partial x} u\right) dx = \int_{E_i} w\, f\, dx$$

E_i





Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

4. Approximate u by interpolating polynomials.

$$\boxed{\int_{-1}^{+1} \rho\, w\, \ddot{u} \frac{dx}{dy} dy + \ldots}$$

exact wave field

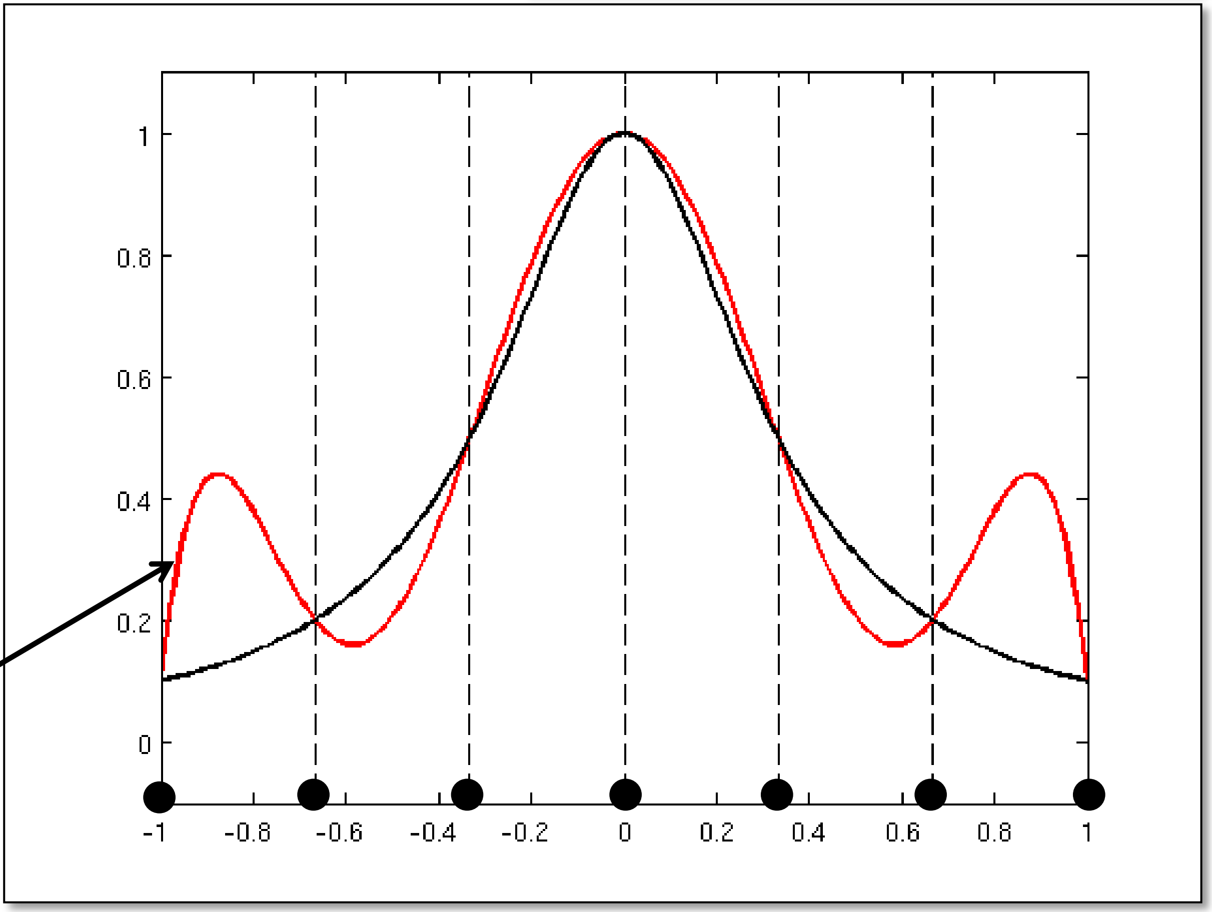
approximation (interpolant)

degree 6

7 collocation points

Runge's phenomenon

equidistant collocation points



Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

4. Approximate u by interpolating polynomials.

$$\int_{-1}^{+1} \rho\, w\, \ddot{u}\, \frac{dx}{dy}\, dy + \ldots$$

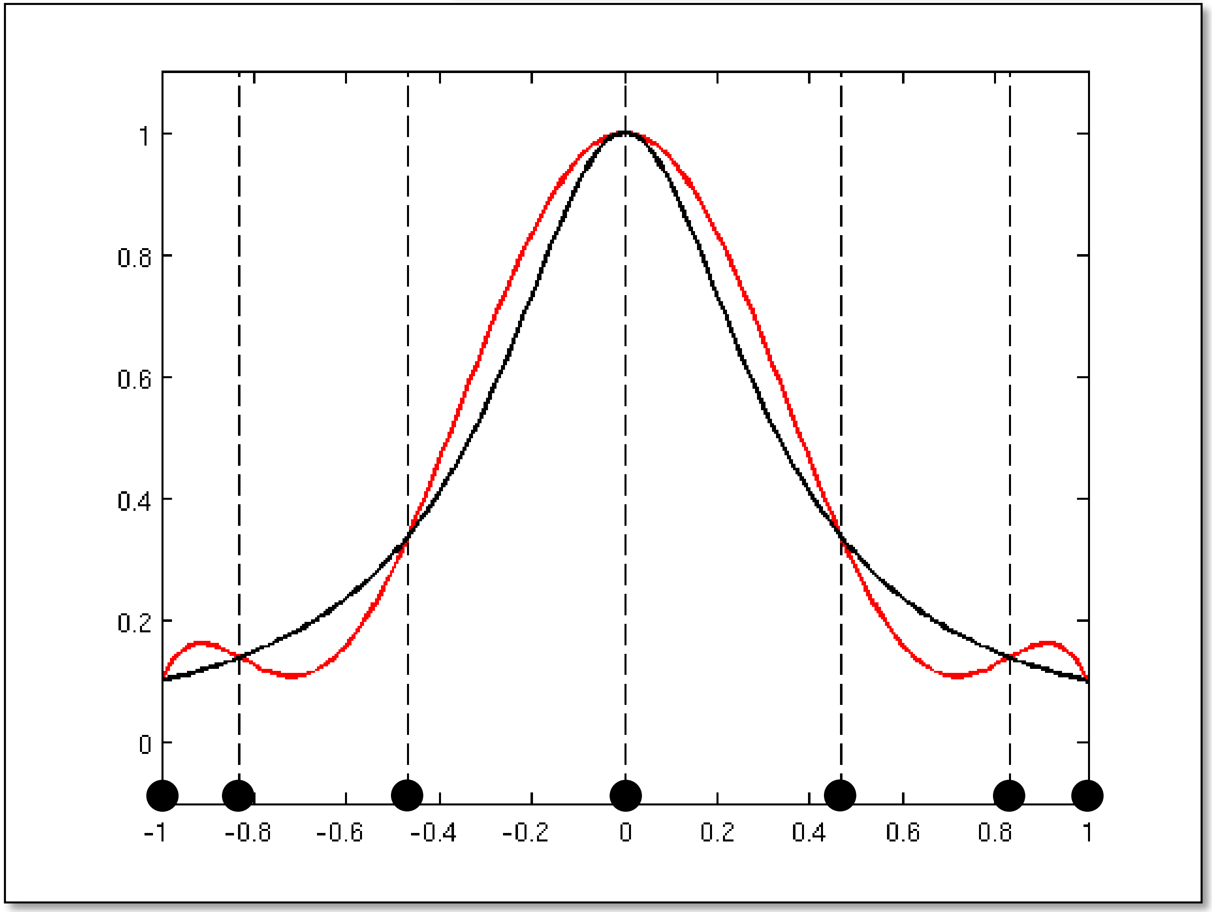
exact wave field

approximation (interpolant)

degree 6

7 collocation points

Gauss-Lobatto-Legendre points



Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

4. Approximate u by Lagrange polynomials of degree N collocated at the GLL points

5. choose Lagrange polynomials also for the test function w

$$\boxed{\int_{-1}^{+1} \rho\, w\, \ddot{u} \frac{dx}{dy}\, dy + \ldots}$$

$$\ell_k(y) \qquad u(y,t) = \sum_{i=0}^{N} u_i(t)\, \ell_i(y)$$

$$\boxed{\int_{-1}^{+1} \rho\, w\, \ddot{u} \frac{dx}{dy}\, dy = \sum_{i=0}^{N} \left[\int_{-1}^{+1} \rho\, \ell_k \ell_i \frac{dx}{dy}\, dy \right] \ddot{u}_i}$$

$$\boxed{\sum_{i=0}^{N} M_{ki}\, \ddot{u}_i(t)}$$

M=mass matrix

Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

4. Approximate u by Lagrange polynomials of degree N collocated at the GLL points

5. choose Lagrange polynomials also for the test function w

6. The integral is approximated using Gauss-Lobatto-Legendre (GLL) quadrature.

- Mass matrix is **diagonal** !!!
- Integration exact for polynomials up to degree 2N-1

$$\boxed{\int_{-1}^{+1} \rho\, w\, \ddot{u} \frac{dx}{dy} dy = \sum_{i=0}^{N} \left[\int_{-1}^{+1} \rho\, \ell_k \ell_i \frac{dx}{dy} dy \right] \ddot{u}_i}$$

$$\boxed{\sum_{i=0}^{N} M_{ki}\, \ddot{u}_i(t)}$$

M=mass matrix

Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

$$\boxed{\int_{E_i} \rho\, w\, \ddot{u}\, dx + \int_{E_i} \mu \left(\frac{\partial}{\partial x} w\right)\left(\frac{\partial}{\partial x} u\right) dx = \int_{E_i} w\, f\, dx}$$

weak form, element-wise

1. mapping to the reference interval [-1 ,1]
2. polynomial approximation (GLL points)
3. numerical integration (GLL quadrature)

$$\boxed{\sum_{i=0}^{N} M_{ki}\, \ddot{u}_i(t)}$$

Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

$$\int_{E_i} \rho\, w\, \ddot{u}\, dx + \int_{E_i} \mu \left(\frac{\partial}{\partial x} w \right) \left(\frac{\partial}{\partial x} u \right) dx = \int_{E_i} w\, f\, dx$$

weak form, element-wise

repeat this for the remaining two terms …

$$\sum_{i=0}^{N} M_{ki}\, \ddot{u}_i(t) \quad + \quad \sum_{i=0}^{N} K_{ki} u_i \quad = \quad f_i$$

stiffness matrix

discrete force vector

Spectral-element discretisation

The SEM concept in 1D: Spatial discretisation

$$\int_{E_i} \rho\, w\, \ddot{u}\, dx + \int_{E_i} \mu \left(\frac{\partial}{\partial x} w\right)\left(\frac{\partial}{\partial x} u\right) dx = \int_{E_i} w\, f\, dx$$

weak form, element-wise

enforce continuity between elements

$$\ddot{\mathbf{u}} = \mathbf{M}^{-1} \cdot (\mathbf{f} - \mathbf{K} \cdot \mathbf{u})$$

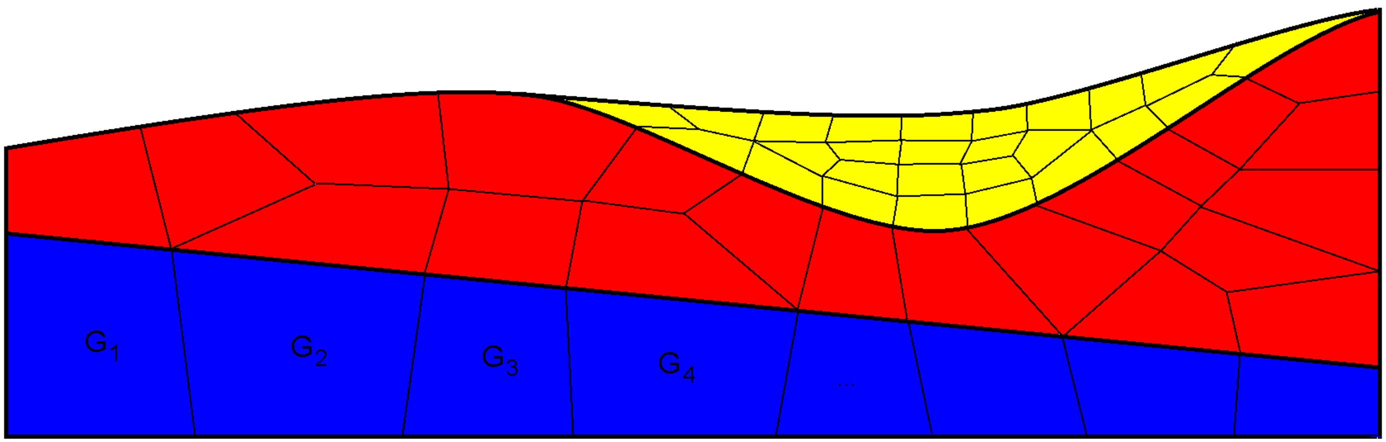
weak form, global

partial differential equation → spatial discretisation → **ordinary differential equation for the polynomial coefficients**

Spectral-element discretisation

The SEM concept in 3D

<u>accurate solutions:</u> discontinuities need to coincide with element boundaries



low velocities: **short wavelength → small elements**

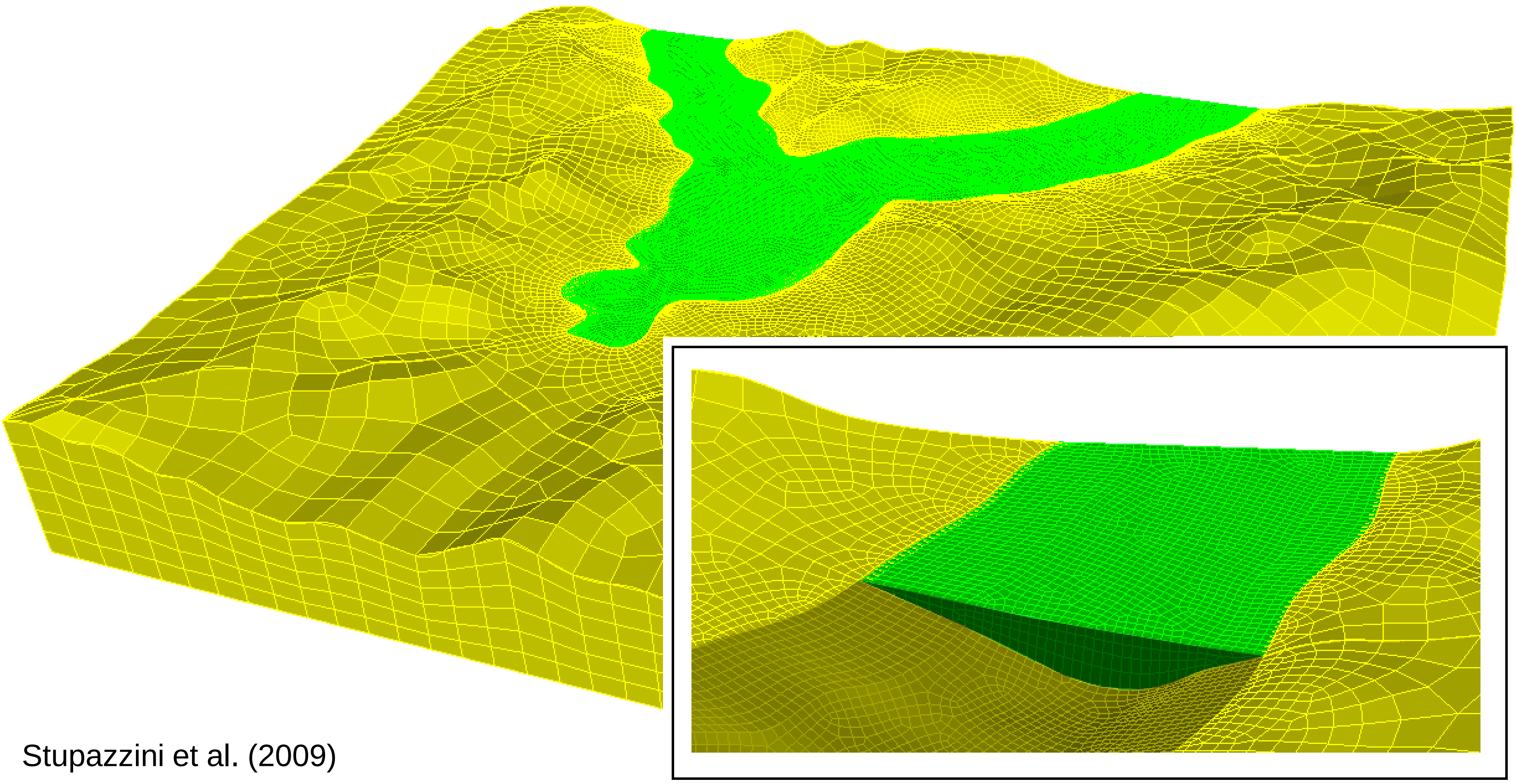
high velocities: **long wavelength → large elements**

many small elements → high computational costs !!!

Spectral-element discretisation

The SEM concept in 3D

Realistic example: The Grenoble valley



Stupazzini et al. (2009)

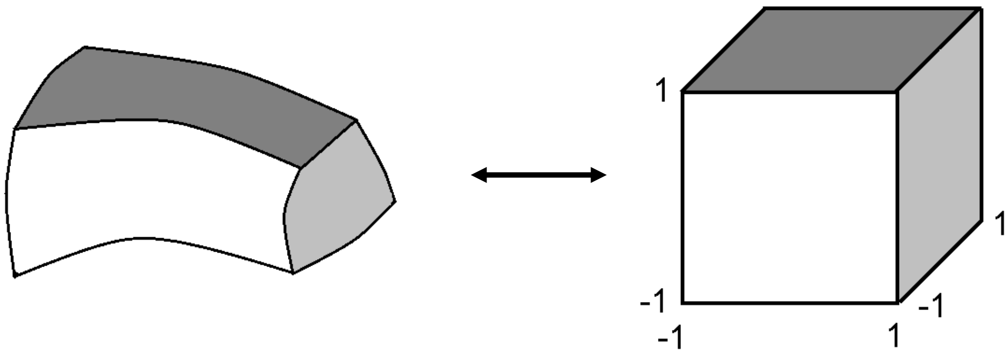
The SEM concept in 3D

Essentially the same as in 1D:

$$\boxed{\rho\,\ddot{u}_i - \frac{\partial}{\partial x_j}\left(C_{ijkl} * \frac{\partial}{\partial x_k} u_l \right) = f_i}$$

1. mapping to the reference cube $[-1\ ,1]^3$
2. polynomial approximation (GLL points)
3. numerical integration (GLL quadrature)

$$\boxed{\ddot{\mathbf{u}} = \mathbf{M}^{-1} \cdot \left(\mathbf{f} - \mathbf{K} \cdot \mathbf{u} \right)}$$



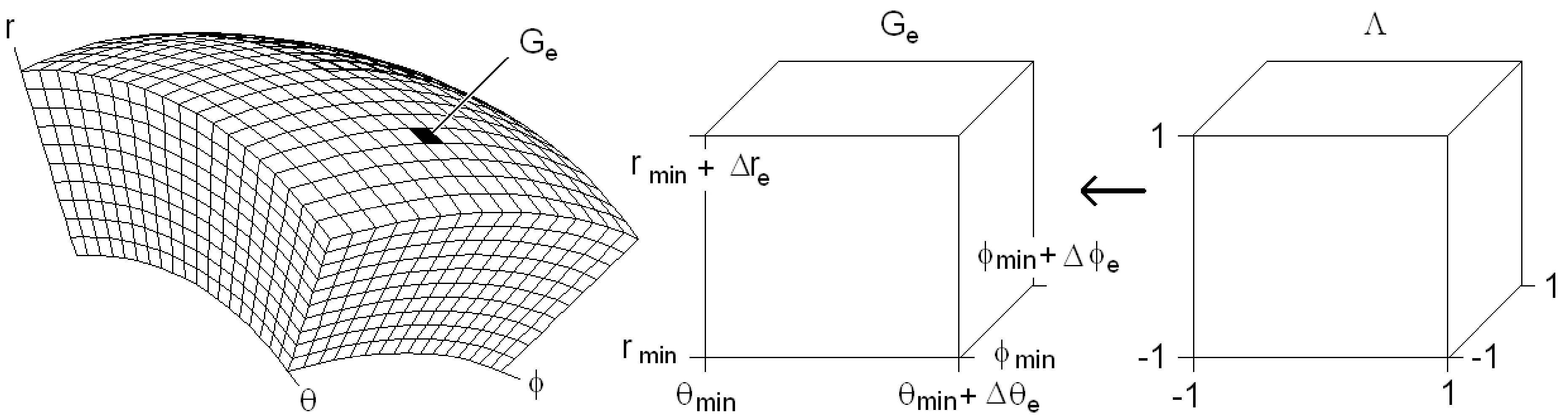
deformed element

reference cube

Spectral-element discretisation

The SEM concept in 3D: SES3D discretisation

- Discretisation of a spherical section without poles and core.
- Each element is a small spherical subsection.



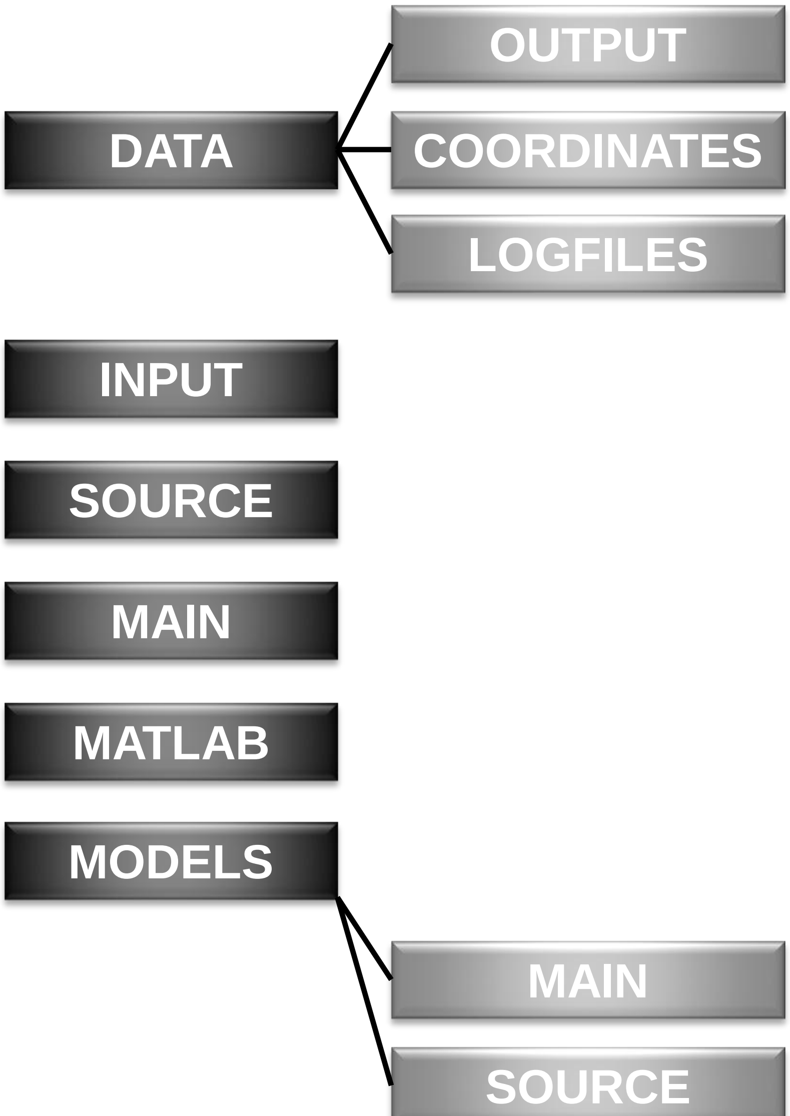
- Speeds up the calculations.
- Simplifies the programme code.
- Reduces flexibility.

Practical

1. SES3D file structure
2. Input files
3. Model generation
4. Forward simulation: wiggly lines – at last!

Practical

File structure



Practical

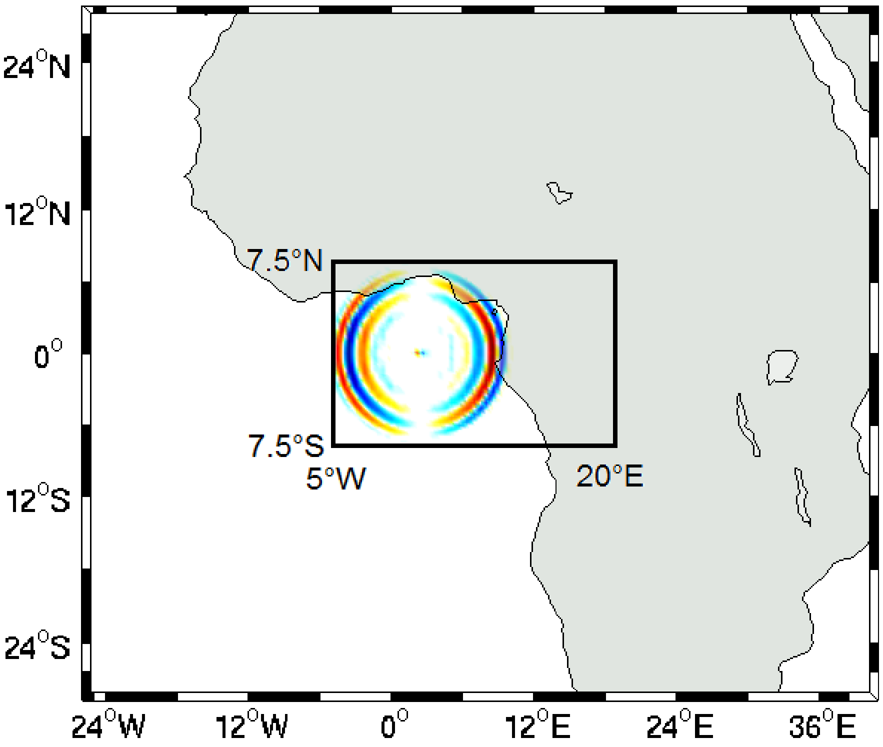
File structure

- **DATA**
  - **OUTPUT** — seismograms, snapshots, kernels
  - **COORDINATES** — files with collocation point coordinates
  - **LOGFILES** — logfiles written during runtime
- **INPUT** — simulation parameters, source time function, receivers
- **SOURCE** — FORTRAN source code
- **MAIN** — executables and scripts for parallel jobs
- **MATLAB** — collection of Matlab (plotting) tools
- **MODELS** — Earth model properties (density, elastic parameters, …)
  - **MAIN** — executables for model generation
  - **SOURCE** — source code for model generation

Practical

Example

- regional-scale wave propagation
- 1 processor (not parallel)



Practical

Input files: Par file

File Edit View Terminal Go Help

```
SIMULATION PARAMETERS ========================================
500                                      ! nt, number of time steps
0.75                                     ! dt in sec, time increment
SOURCE =======================================================
90.00
2.50
30000.0                                                          (m)
10                                                               , 3:f_z, 10:M_ij
0.00e19
0.00e19
0.00e19
0.00e19
0.00e19
1.00e19
MODEL ========================================================
82.50                                    ! theta_min (colatitude) in degrees
97.50                                    ! theta_max (colatitude) in degrees
-5.0                                     ! phi_min (longitude) in degrees
20.0                                     ! phi_max (longitude) in degrees
5371000.0                                ! z_min (radius) in m
6371000.0                                ! z_max (radius) in m
0                                        ! is_aniso
0                                        ! is_diss
2                                        ! model_type
```

- total number of time steps (nt=500)
- time increment in seconds (dt=0.75)
- simulation length = nt·dt (= 375 s)
- dt must satisfy the **stability criterion**:

$$dt < c\frac{dx_{\min}}{v_{\max}}, \qquad c \approx 0.3$$

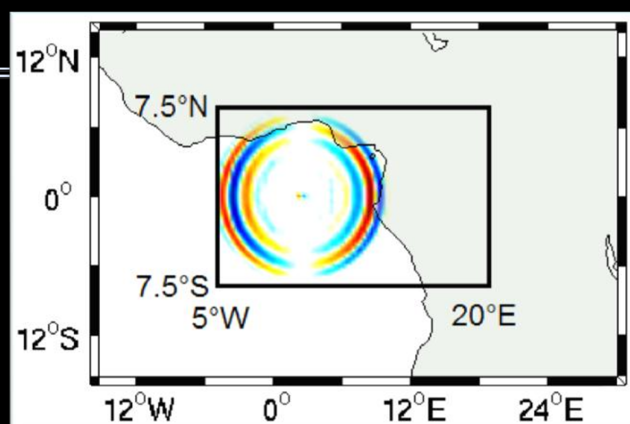
Practical

Input files: Par file

```
File  Edit  View  Terminal  Go  Help
SIMULATION PARAMETERS ======================================
500                                     ! nt, number of time steps
0.75                                    ! dt in sec, time increment
SOURCE =====================================================
90.00                                   ! xxs, theta-coord.
2.50                                    ! yys, phi-coord.
30000.0                                 ! zzs, source depth in (m)
10                                                      , 3:f_z, 10:M_ij
0.00e19
0.00e19
0.00e19
0.00e19
0.00e19
1.00e19
MODEL ======================
82.50                                                   ) in degrees
97.50                                                   ) in degrees
-5.0                                                    n degrees
20.0                                                    n degrees
5371000.0
6371000.0
0
0
2                                       ! model_type
```

source position

- colatitude (90°-latitude) in ° (90°)
- longitude in ° (2.5°)
- depth in m, **not** km (30000 m)



Practical

Input files: Par file

```
File  Edit  View  Terminal  Go  Help
SIMULATION PARAMETERS ========================================
500                                   ! nt, number of time steps
0.75                                  ! dt in sec, time increment
SOURCE =======================================================
90.00                                 ! xxs, theta-coord.
2.50                                  ! yys, phi-coord.
30000.0                               ! zzs, source depth in (m)
10                                    ! srctype, 1:f_x, 2:f_y, 3:f_z, 10:M_ij
0.00e19                               ! M_theta_theta
0.00e19                               ! M_phi_phi
0.00e19                               ! M_r_r
0.00e19                               ! M_theta_phi
0.00e19                               ! M_theta_r
1.00e19                               ! M_phi_r
MODEL ========================================================
82.50
97.50
-5.0
20.0
5371000.0
6371000.0
0                                     ! is_aniso
0                                     ! is_diss
2                                     ! model_type
```

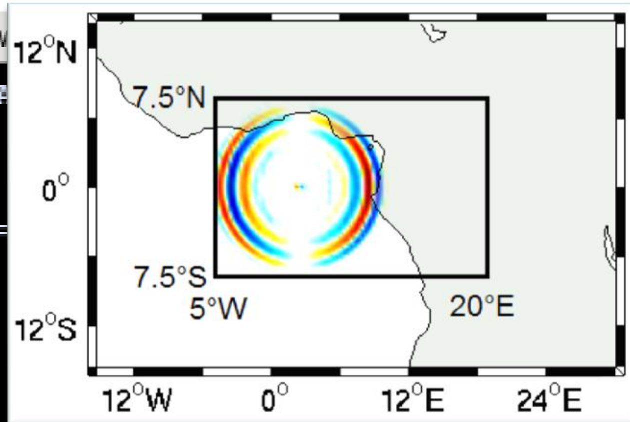
source characteristics

- source type: 1,2,3=single forces, 10=moment tensor
- moment tensor components in Nm
- Mtt, Mpp, Mrr, Mtp, Mtr, Mpr

Practical

Input files: Par file

File Edit View




```
SIMULATION PA                                 =================
500                                           umber of time steps
0.75                                          sec, time increment
SOURCE ======                                 =================
90.00                                         theta-coord.
2.50                                          phi-coord.
30000.0                                       source depth in (m)
10                                            pe, 1:f_x, 2:f_y, 3:f_z, 10:M_ij
0.00e19
0.00e19
0.00e19
0.00e19
0.00e19
1.00e19
MODEL ===========================================================
82.50                                   ! theta_min (colatitude) in degrees
97.50                                   ! theta_max (colatitude) in degrees
-5.0                                    ! phi_min (longitude) in degrees
20.0                                    ! phi_max (longitude) in degrees
5371000.0                               ! z_min (radius) in m
6371000.0                               ! z_max (radius) in m
0                                       ! is_aniso
0                                       ! is_diss
2                                       ! model_type
```

geographic region

- minimum and maximum **co**latitude in° (82.5°, 97.5°)
- minimum and maximum longitude in ° (-5.0°, 20.0°)
- minimum and maximum radius in m (5371e3 m, 6371e3 m)

Practical

Input files: Par file

```
MODEL ========================================================
82.50                                   ! theta_min (colatitude) in degrees
97.50                                   ! theta_max (colatitude) in degrees
-5.0
20.0
5371000.0
6371000.0
0                                       ! is_aniso
0                                       ! is_diss
2                                       ! model_type
COMPUTATIONAL SETUP (PARALLELISATION) =========================
15                                      ! nx_global
25                                      ! ny_global
10                                      ! nz_global
4                                       ! lpd, LAGRANGE polynomial degree
1                                       ! px, processors in theta direction
1                                       ! py, processors in phi direction
1                                       ! pz, processors in r direction
OUTPUT DIRECTORY ============================================
../DATA/OUTPUT/1/
```

- radial anisotropy switched on (=1) or off (=0)
- visco-elastic dissipation switched on (=1) or off (=1)
- structural model (2=PREM)

Practical

Input files: Par file

```
MODEL ==========================================================
82.50                                   ! theta_min (colatitude) in degrees
97.50                                   ! theta_max (colatitude) in degrees
-5.0
20.0
5371000
6371000
0
0
2
COMPUTATIONAL SETUP (PARALLELISATION) ========================
15                                      ! nx_global
25                                      ! ny_global
10                                      ! nz_global
4                                       ! lpd, LAGRANGE polynomial degree
1                                       ! px, processors in theta direction
1                                       ! py, processors in phi direction
1                                       ! pz, processors in r direction
OUTPUT DIRECTORY =============================================
../DATA/OUTPUT/1/
```

Spatial discretisation

- number of elements in theta (colatitude) direction (15)
- number of elements in phi (longitude) direction (25)
- number of elements in vertical direction (10)
- Lagrange polynomial degree

Practical

Input files: Par file

```
MODEL ========================================================
82.50                                   ! theta_min (colatitude) in degrees
97.50
-5.0
20.0
5371000.0
6371000.0
0
0
2
COMPUTATIONAL SET
15
25
10
4
1
1                                       ! py, processors in phi direction
1                                       ! pz, processors in r direction
OUTPUT DIRECTORY ============================================
../DATA/OUTPUT/1/
```

THE rule of thumb

- element size in theta (colatitude) direction: ≈ 110 km
- element size in phi (longitude) direction: ≈ 110 km
- element size in vertical direction: ≈ 100 km

Use at least 2 elements per minimum wavelength!!!

$$T = \frac{\lambda}{v} \approx \frac{2 \times \max \textit{element size}}{\min \textit{velocity}} \approx \frac{200\,km}{3.5\,km/s} \approx 60\,s$$

Practical

Input files: Par file

```
MODEL ========================================================
82.50                                    ! theta_min (colatitude) in degrees
97.50                                    ! theta_max (colatitude) in degrees
-5.0                                     ! phi_min (longitude) in degrees
20.0                                     ! phi_max (longitude) in degrees
5371000.0                                ! z_min (radius) in m
6371000.0                                ! z_max (radius) in m
0                                        ! is_aniso
0                                        ! is_diss
2
COMPUTA
15
25
10
4                                        ! lpd, LAGRANGE polynomial degree
1                                        ! px, processors in theta direction
1                                        ! py, processors in phi direction
1                                        ! pz, processors in r direction
OUTPUT DIRECTORY =============================================
../DATA/OUTPUT/1/
```

Parallelisation

- number of processors in theta (colatitude) direction (1)
- number of processors in phi (longitude) direction (1)
- number of processors in vertical direction (1)

Input files: Par file

```
MODEL ========================================================
82.50                                      ! theta_min (colatitude) in degrees
97.50                                      ! theta_max (colatitude) in degrees
-5.0                                       ! phi_min (longitude) in degrees
20.0                                       ! phi_max (longitude) in degrees
5371000.0                                  ! z_min (radius) in m
6371000.0                                  ! z_max (radius) in m
0                                          ! is_aniso
0                                          ! is_diss
2                                          ! model_type
COMPUTATIONAL SETUP (PARALLELISATION) ========================
15                                         ! nx_global
25                                         ! ny_global
10                                         ! nz_global
4                                          ! lpd, LAGRANGE polynomial degree
1                                          ! px, processors in theta direction
1                                          ! py, processors in phi direction
1                                          ! pz, processors in r direction
OUTPUT DIRECTORY ===
../DATA/OUTPUT/1/
```

directory where the output (e.g. seismograms) is written

Practical

Input files: Par file

```
OUTPUT DIRECTORY ==============================================
../DATA/OUTPUT/1/
OUTPUT FLAGS ==================================================
100                                    ! ssamp, snapshot sampling
1                                      ! output_displacement
0                                      ! output_preco
ADJOINT PARAMETERS ============================================
1                                      ! adjoint flag
10                                     ! samp_ad
../INTERMEDIATE/
```

write snapshots of the wave field (1=yes, 0=no) into the OUTPUT DIRECTORY every ssamp (=100) time steps

Practical

Input files: Par file

```
OUTPUT DIRECTORY ===============================================
../DATA/OUTPUT/1/
OUTPUT FLAGS ===================================================
100                                        ! ssamp, snapshot sampling
1                                          ! output_displacement
0                                          ! output_preco
ADJOINT PARAMETERS =============================================
1                                          ! adjoint flag
10                                         ! samp_ad
../INTERMEDIATE/
```

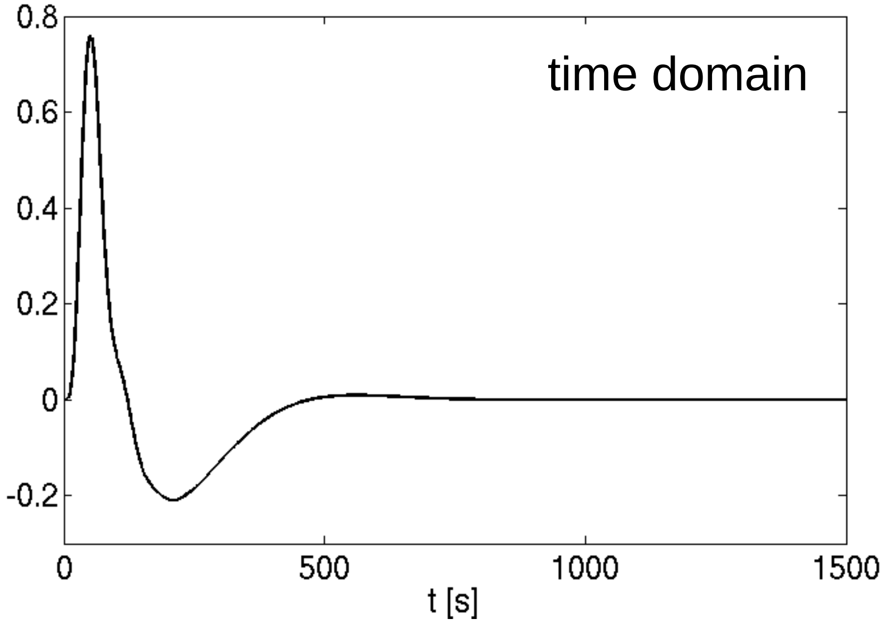
store velocity and strain field (1=yes, 0=no) every samp_ad (=10) time steps in the directory below (../INTERMEDIATE)

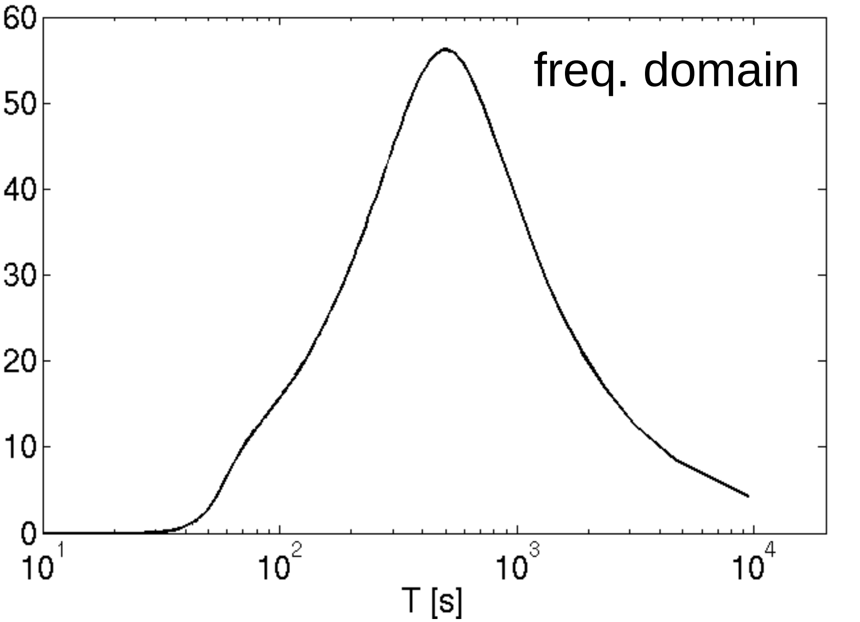
Needed for the computation of sensitivity kernels.

Practical

Input files: source time function (stf)

- Heaviside function, bandpass-filtered between 60 s and 500 s
- lower cutoff period (60 s) dictated by the size of the elements
- upper cutoff (500 s) ensures that the stf returns to zero
- always use bandlimited stf's





Practical

Input files: source time function (stf)

- Heaviside function, bandpass-filtered between 60 s and 500 s
- lower cutoff period (60 s) dictated by the size of the elements
- upper cutoff (500 s) ensures that the stf returns to zero
- always use bandlimited stf's

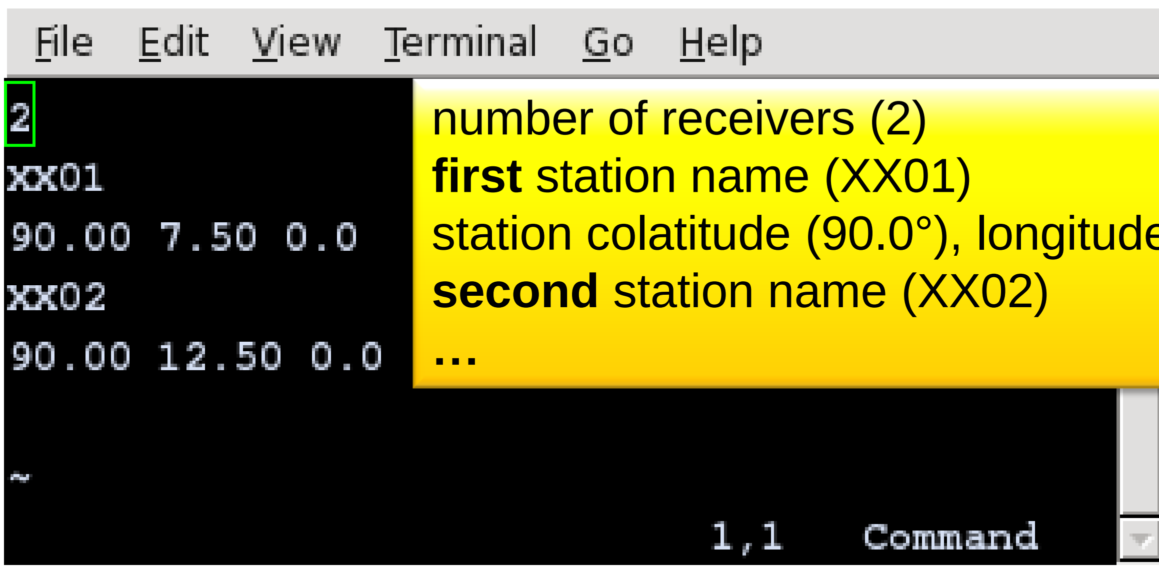
File Edit View Terminal Go Help

```
1.68008e-17
1.17534e-07
1.90214e-06
1.04616e-05
3.60694e-05
9.52114e-05
0.000211155
0.000414113
0.000741076
0.00123537
0.00194598
0.00292674
0.00423538
0.00593246
```

stf: plain ASCII list of numbers

Practical

Input files: receiver configuration (recfile)



number of receivers (2)
first station name (XX01)
station colatitude (90.0°), longitude (7.5°), depth (0.0 m)
second station name (XX02)
…

Practical

Model generation

MODELS

MAIN

SOURCE

run *generate_models.exe*

1 file for each model parameter (lambda, mu, 1/rho) …

… and for each processor.

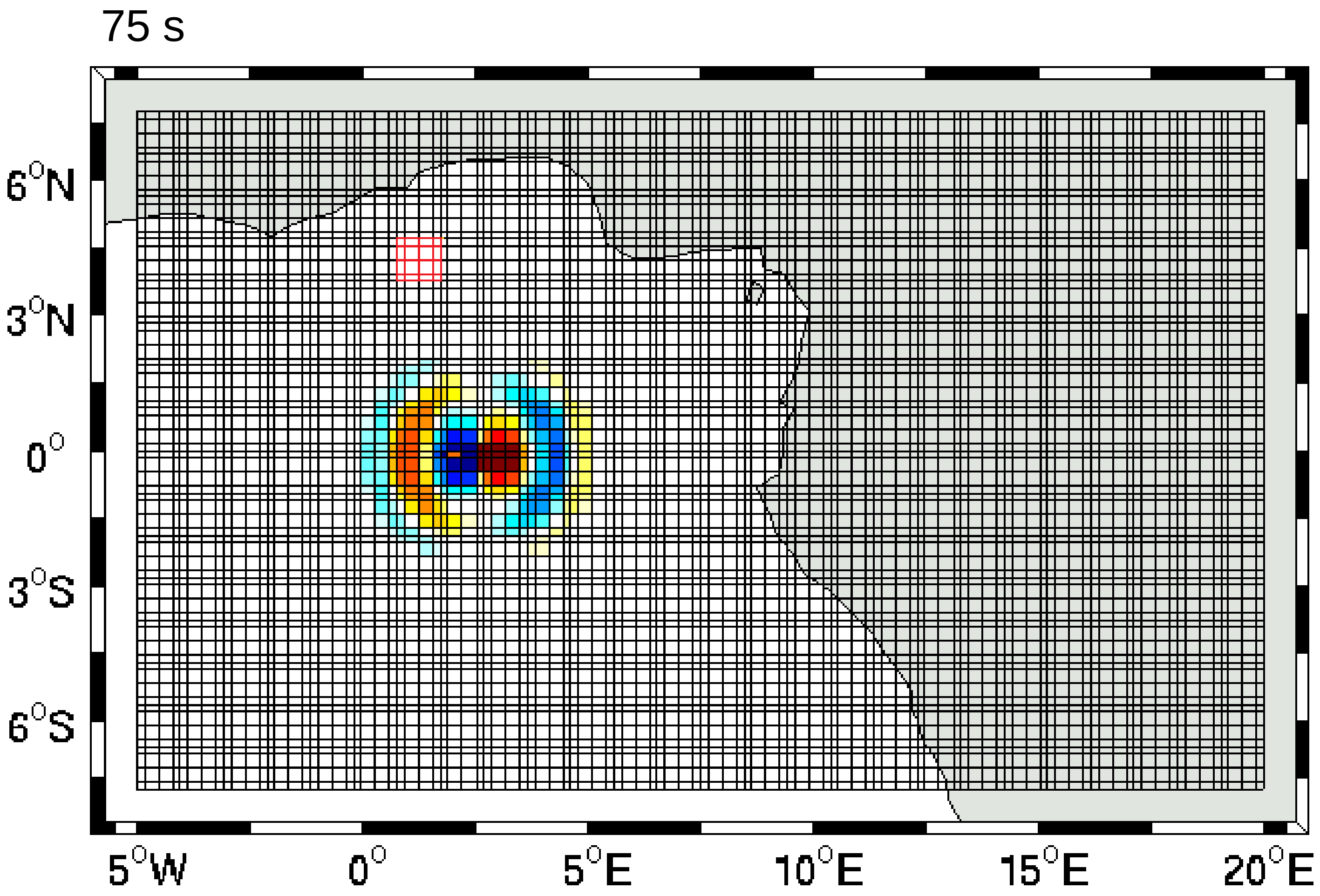
Practical

Run SES3D

MAIN run *main.exe* and wait …

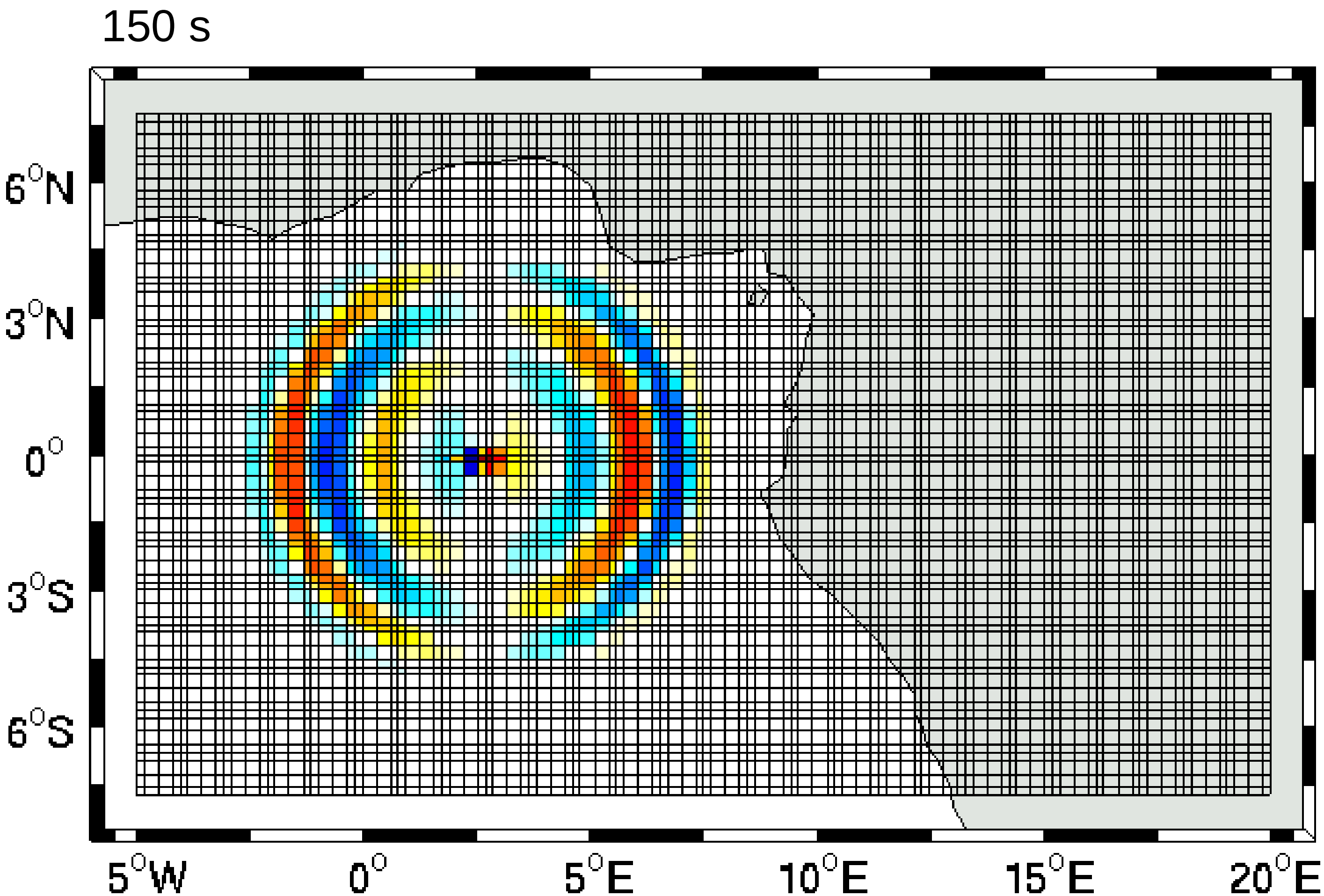
Practical

Simulation



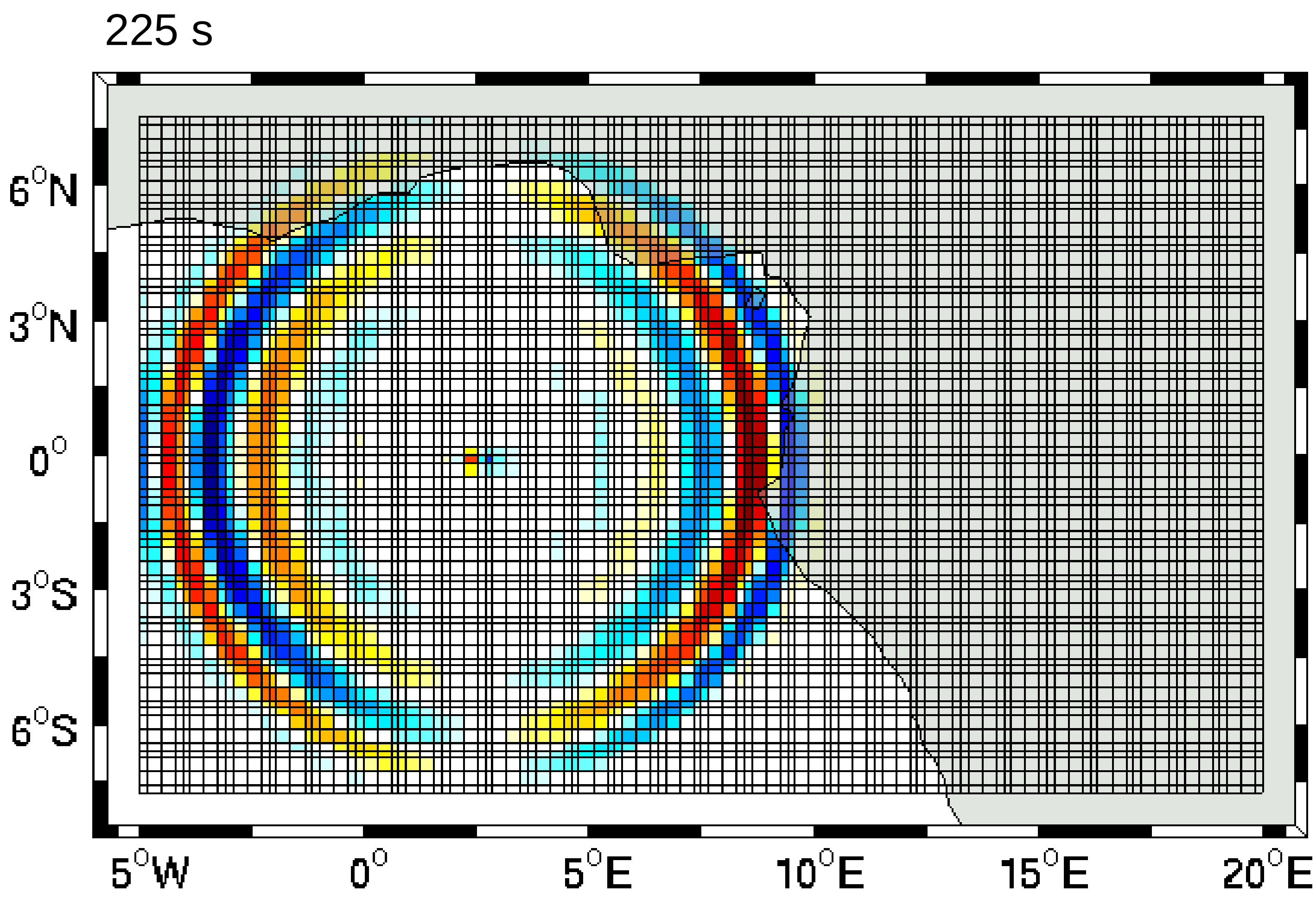
Practical

Simulation



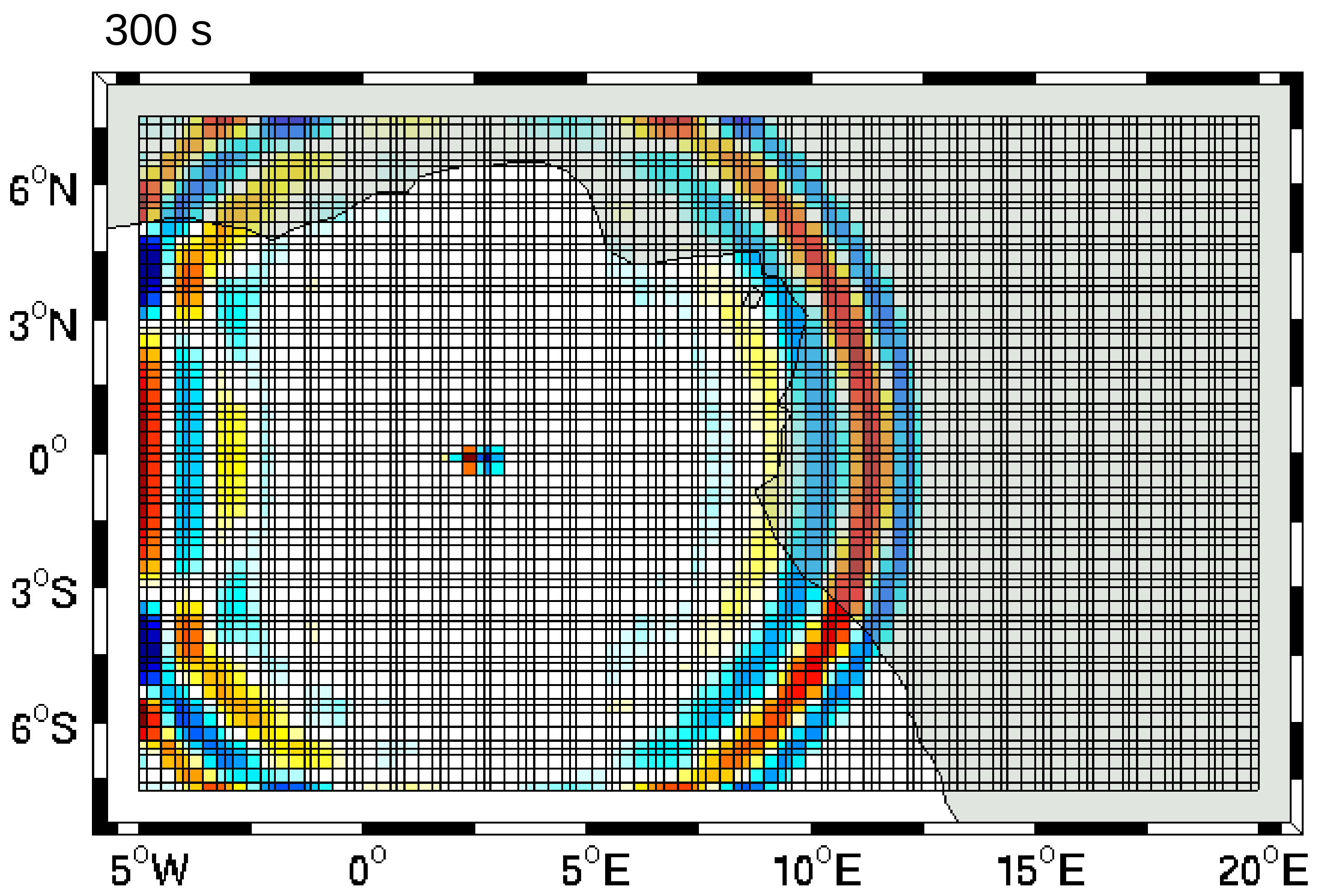
Practical

Simulation



Practical

Simulation



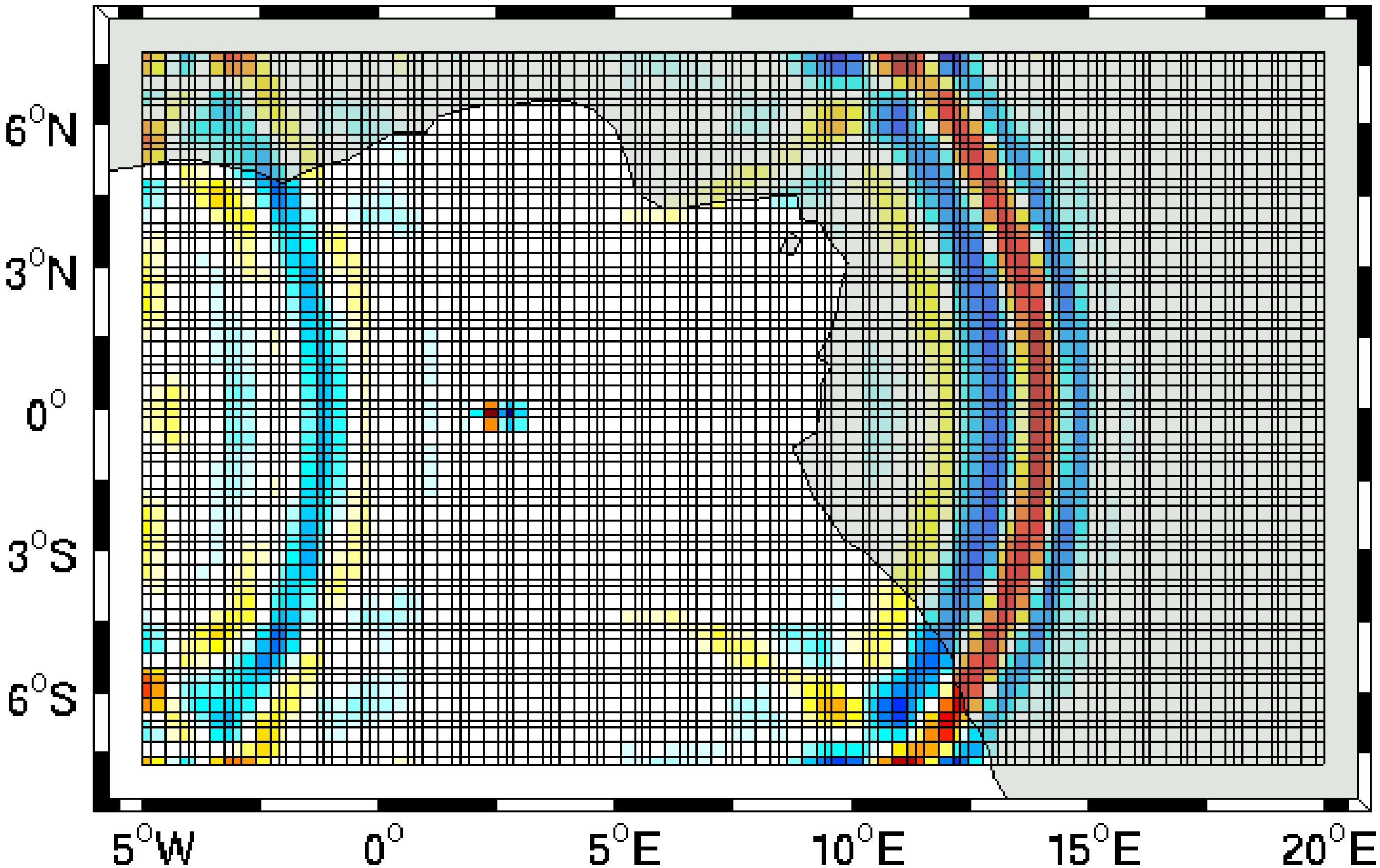
Practical

Simulation

$$v \approx \frac{2\,{}^{3}\!/_{4}\ \text{elements}}{75\,\text{s}} \approx \frac{305\,\text{km}}{75\,\text{s}} \approx 4.1\,\text{km/s}$$

➡ fundamental-mode Rayleigh wave @ 60 s period

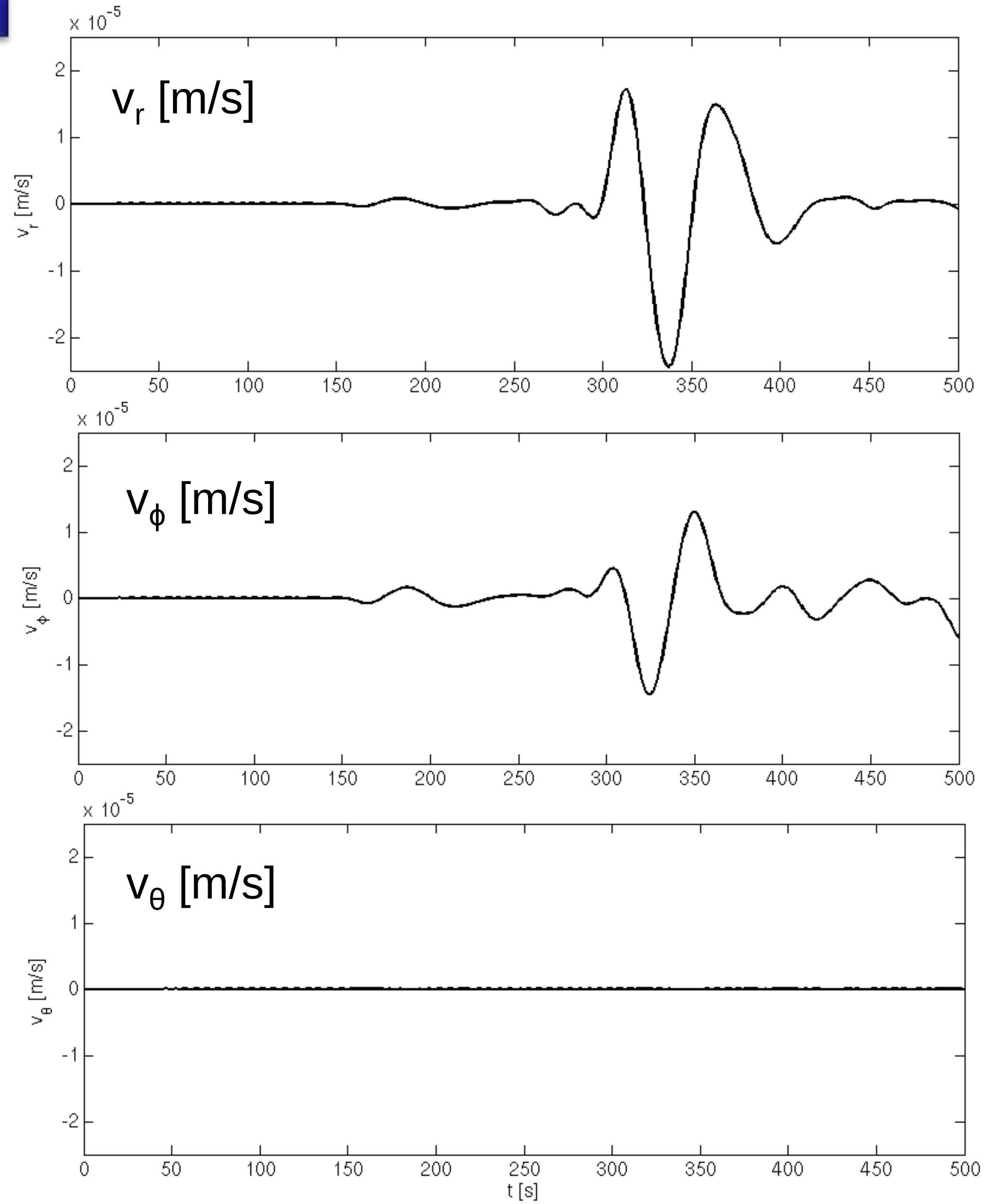
375 s



Practical

Results: Synthetic seismograms

- prominent Rayleigh wave
- no displacement on the N-S component
- pollution by reflections because absorbing boundaries are switched off



Special Topics

1. Visco-elastic dissipation
2. Point sources
3. Absorbing boundaries

Dissipation

- Again, the concept in 1D:
- The Earth is assumed to have a visco-elastic rheology defined as:

$$\boxed{\sigma(t) = (C * \dot{\varepsilon})(t) = \int_{-\infty}^{t} C(t-t')\dot{\varepsilon}(t')\,dt'}$$

- This convolution is very inconvenient in time-domain wave propagation!

Dissipation

- Again, the concept in 1D:
- The Earth is assumed to have a visco-elastic rheology defined as:

$$\boxed{\sigma(t) = (C * \dot{\varepsilon})(t) = \int_{-\infty}^{t} C(t-t')\dot{\varepsilon}(t')\,dt'}$$

- For the stress-relaxation function C(t) one assumes a superposition of standard-linear solids:

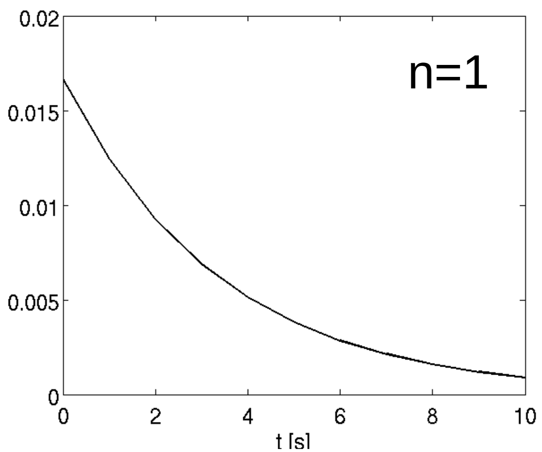
$$\boxed{C(t) = C_r\left[1 + \frac{\tau}{N}\sum_{n=1}^{N} e^{-t/\tau_n}\right] H(t)}$$

- C(t) describes the stress that occurs in response to a unit step strain.
- The parameters of C(t) are chosen such that the corresponding Q(ω) takes a specific form.

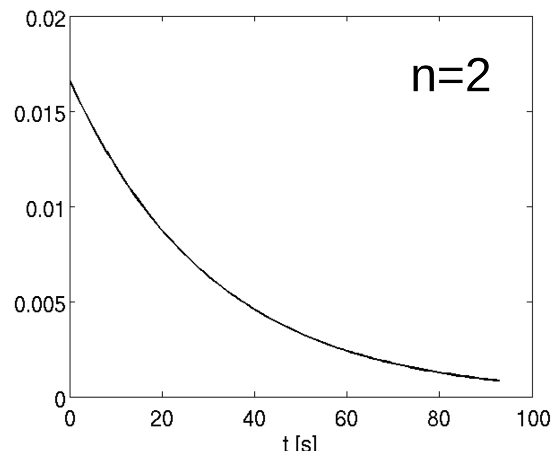
Dissipation

$$C(t) = C_r \left[1 + \frac{\tau}{N} \sum_{n=1}^{N} e^{-t/\tau_n} \right] H(t)$$

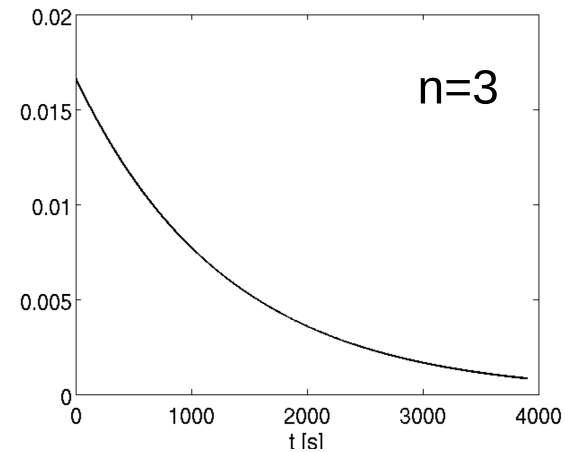
Example for N=3:



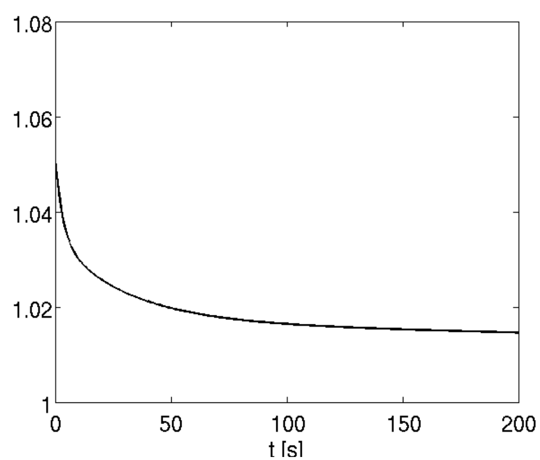
+



+

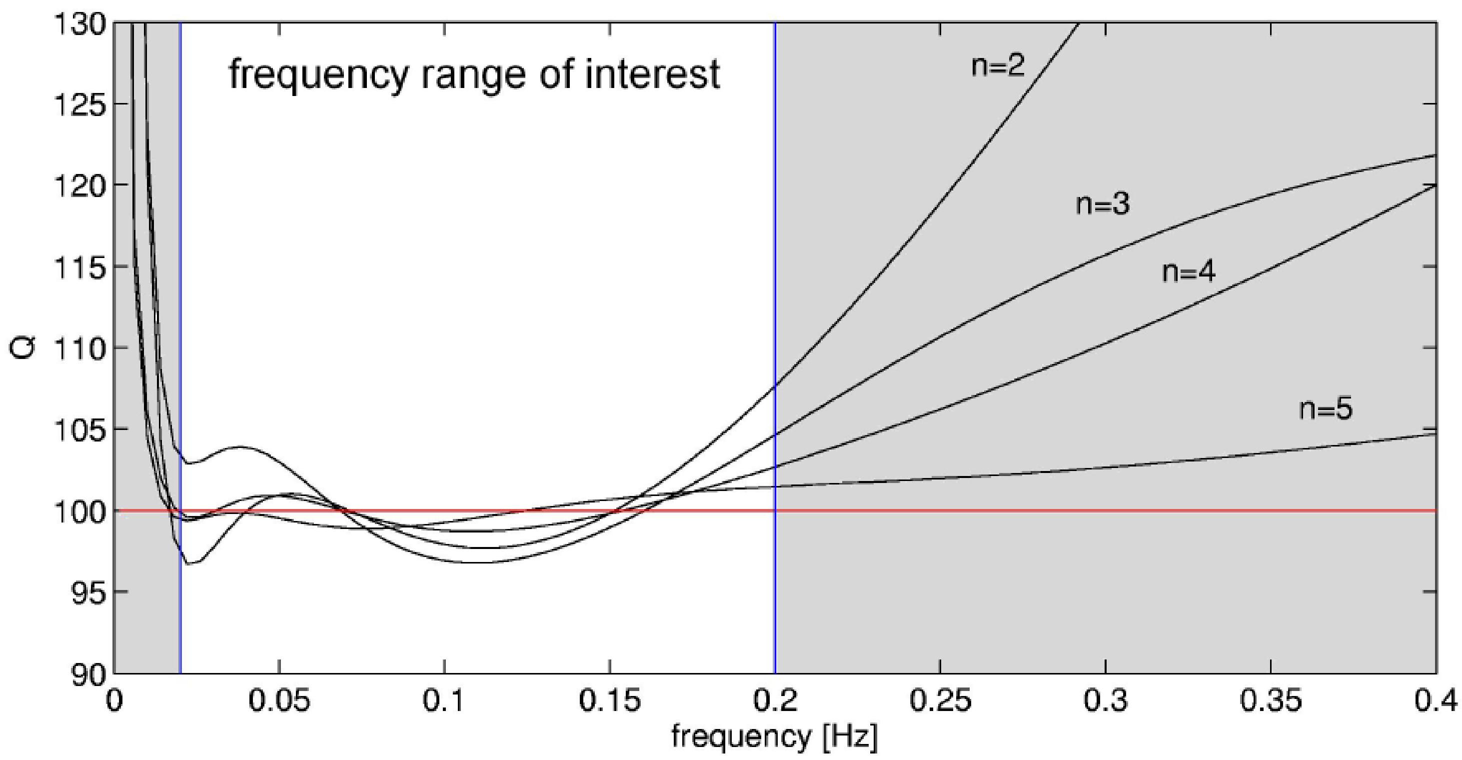


+ constant =



Dissipation

- The example was designed to have Q=100=const within a frequency range from 0.02 Hz to 0.2 Hz.



- More mechanisms increase the quality of the Q-const approximation and also the computational costs.

Point sources

- There are generally two ways of implementing a point source – each with its advantages and disadvantages:

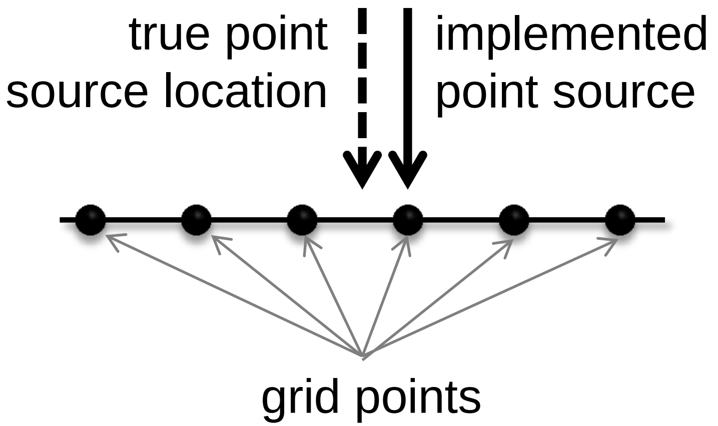
Point sources

- There are generally two ways of implementing a point source – each with its advantages and disadvantages:

1. Grid point implementation

Force acts at the grid point that is closest to the true point source location.

- easy to implement
- correct near field
- numerical error when the true location of the point force is too far from the nearest grid point



Point sources

- There are generally two ways of implementing a point source – each with its advantages and disadvantages:

2. Lowpass filtered δ-function (implemented in SES3D)

The δ-function is approximated by a polynomial (degree and collocation points as in the spatial discretisation)

- correct solution in the far field for any source position
- implementation more difficult
- near-field inaccurate (within ≈ 2 elements around the source)

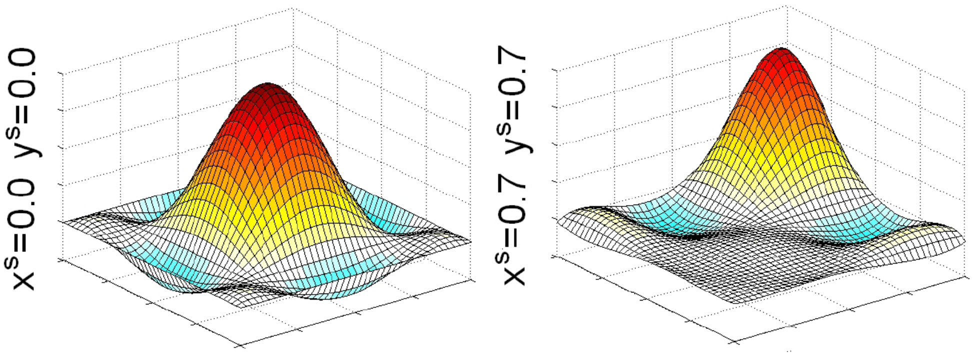


Fig.: Polynomial approximation of the δ-function (degree 4) within one 2D element, for different source positions (x^s, y^s).

Absorbing boundaries

- Methods to avoid reflections from unphysical boundaries fall into 2 categories:

1. Absorbing boundary conditions

Boundary conditions that prevent the popagation of energy into the medium.

- easy and elegant implementation
- highly inefficient for large angles of incidence
- can be unstable

2. Absorbing boundary layers (implemented in SES3D)

Boundary layers where the amplitudes of incoming waves decay rapidly

- efficient for all angles of incidence
- implementation more involved
- can be unstable

Absorbing boundaries

2. Absorbing boundary layers (implemented in SES3D)

- **P**erfectly **M**atched **L**ayers (PML) are the most popular and efficient absorbing technique.
- Wave equation is modified within the boundary region so that incoming waves decay exponentially:

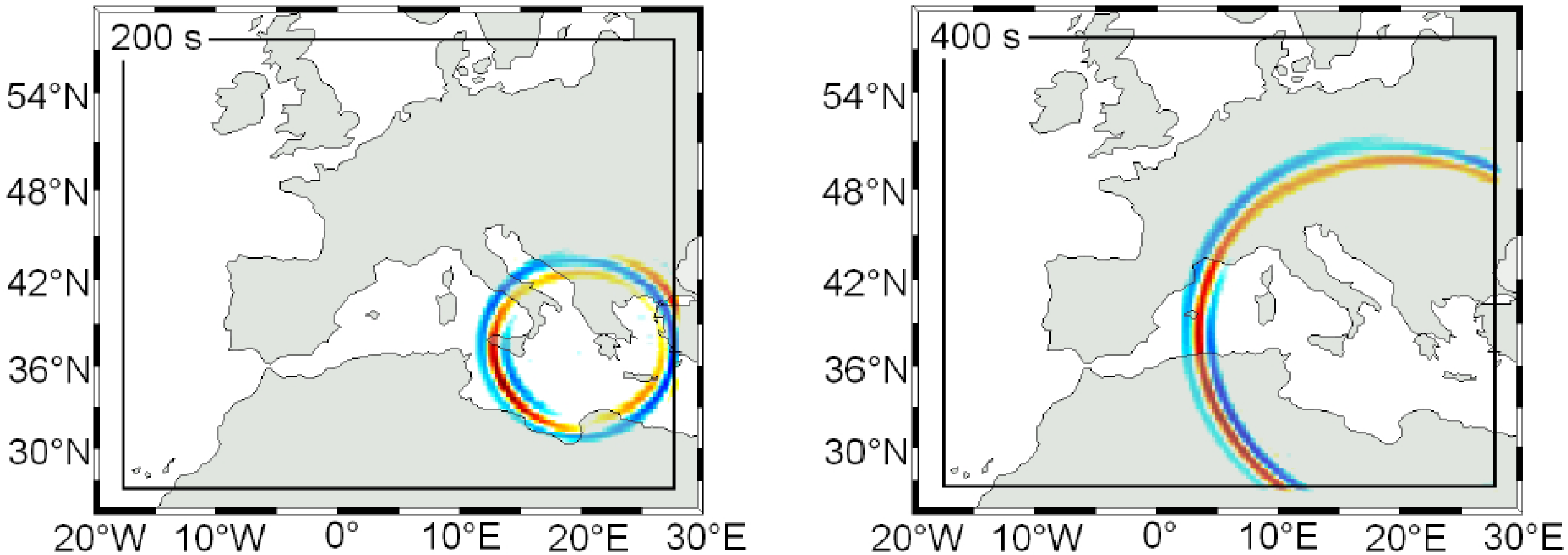


Fig.: SES3D wavefield snapshots, illustrating the absorption of energy within the PML region.

Absorbing boundaries

2. Absorbing boundary layers (implemented in SES3D)

- **P**erfectly **M**atched **L**ayers (PML) are the most popular and efficient absorbing technique.
- Wave equation is modified within the boundary region so that incoming waves decay exponentially:

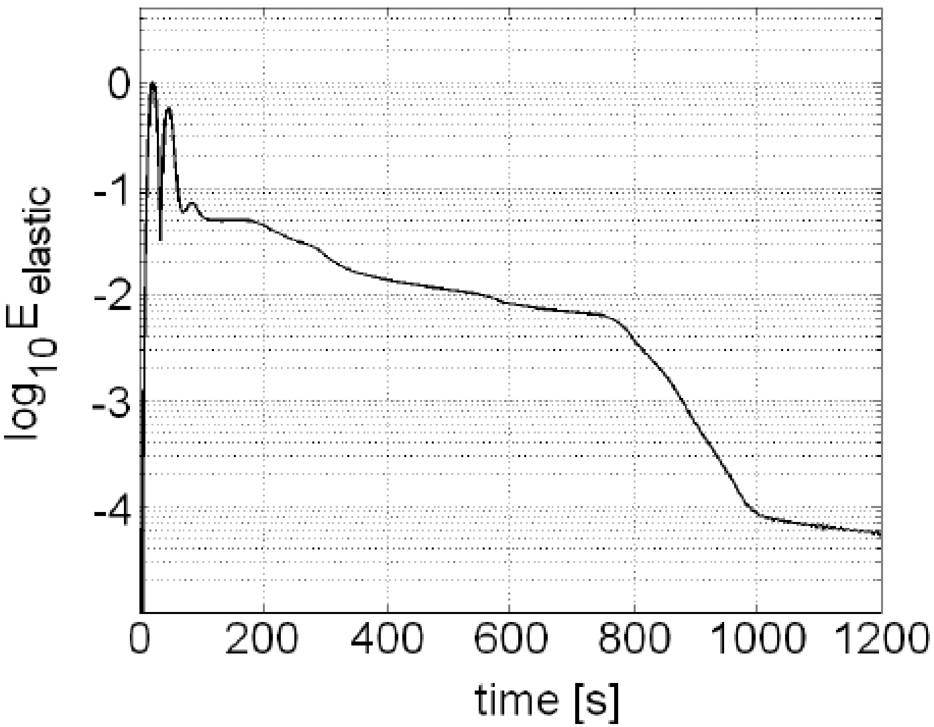


Fig.: Total kinetic energy within the model as a function of time. The energy decreases rapidly but does not reach 0 due to imprefect absorption.