February 20th-25th, 2011

SeisSol Course

From elastic wave equations to the ADER-DG method

Christian Pelties & Martin Käser

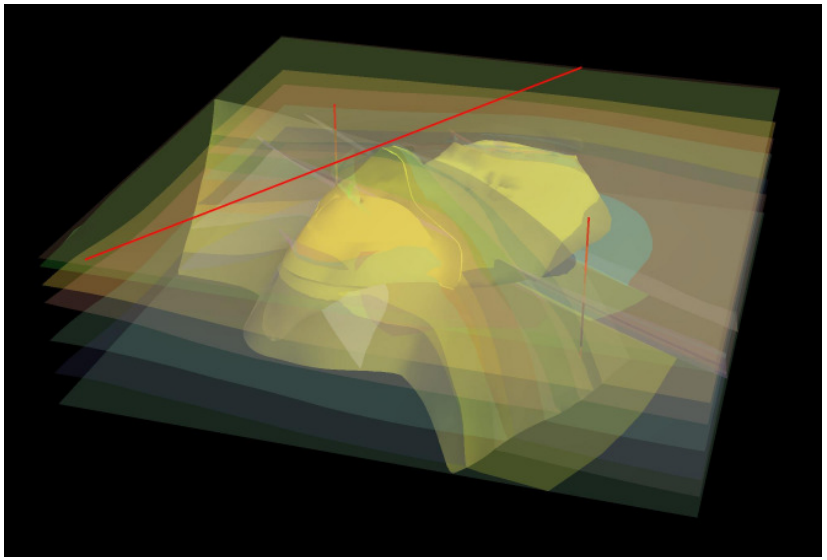
Department of Earth and Environmental Sciences, LMU München, Germany

General Overview – Possibilities

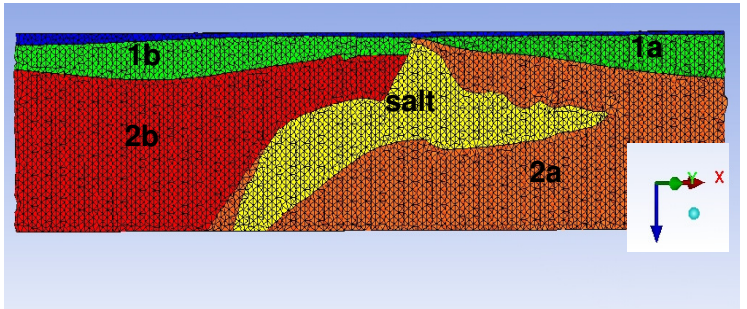
Part I

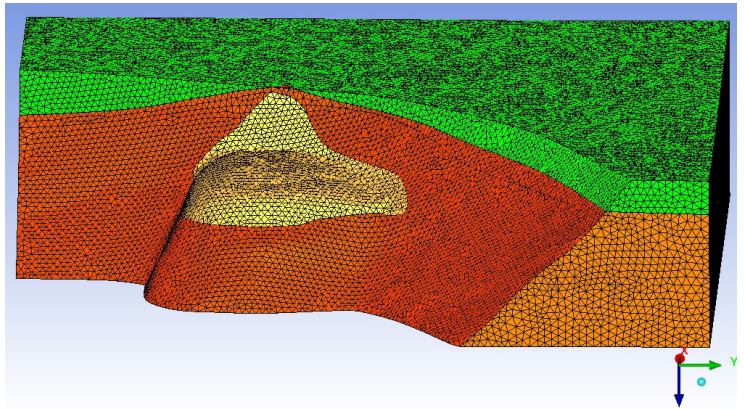
Characteristics of the SeisSol simulation software:

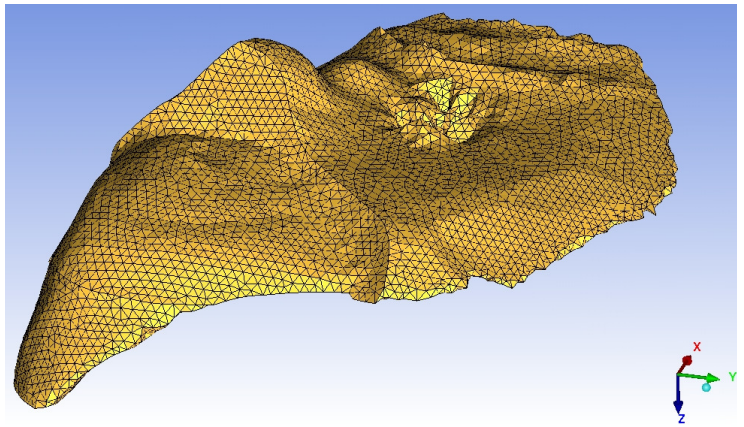
• use of tetrahedral and hexahedral meshes to approximate complex 3D model geometries

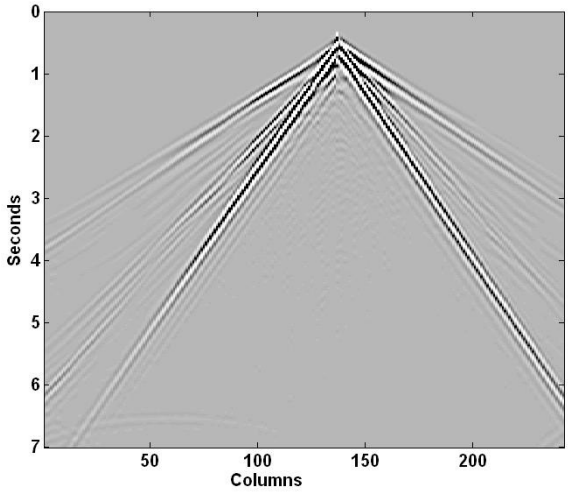


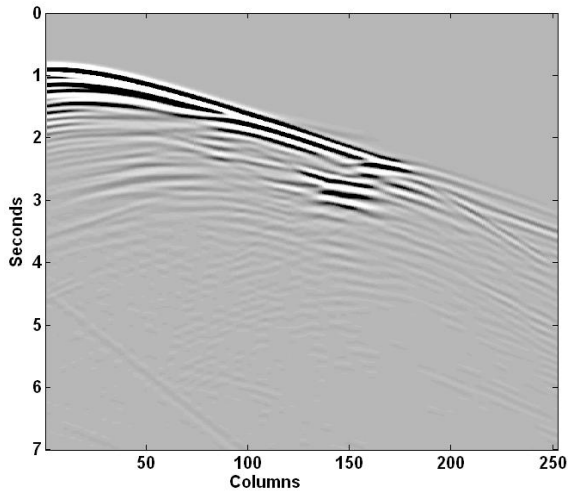
Salt dome model - SEAM Consortium





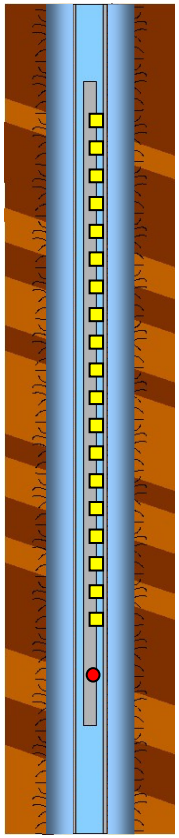


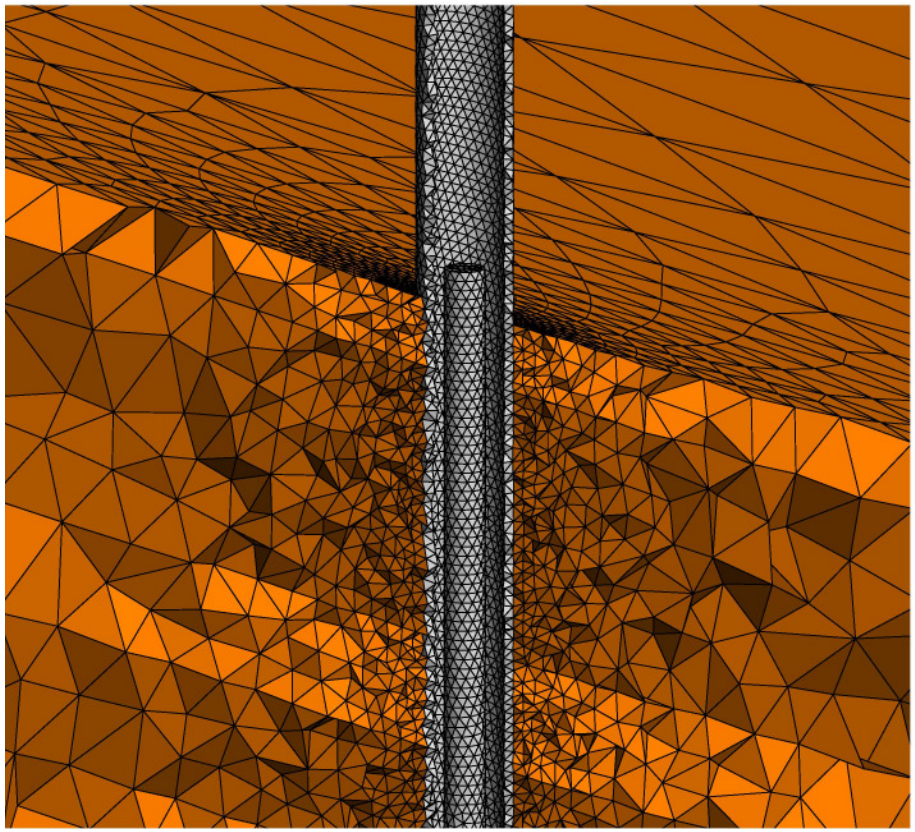


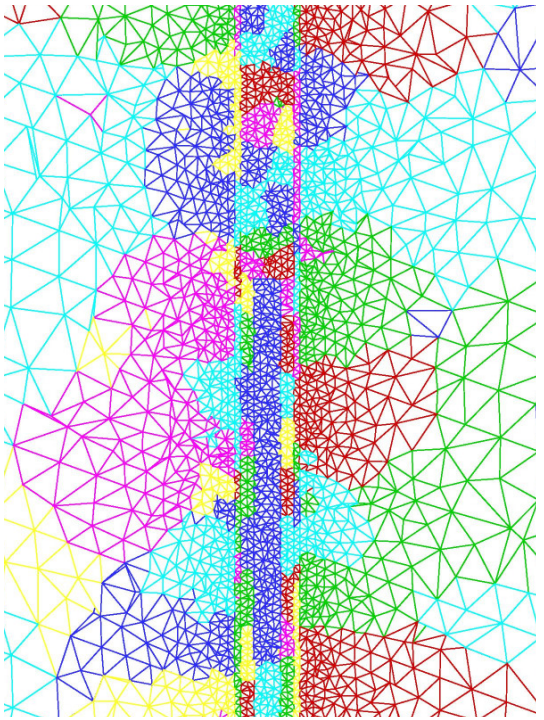


Characteristics of the SeisSol simulation software:

• use of tetrahedral and hexahedral meshes to approximate complex 3D model geometries



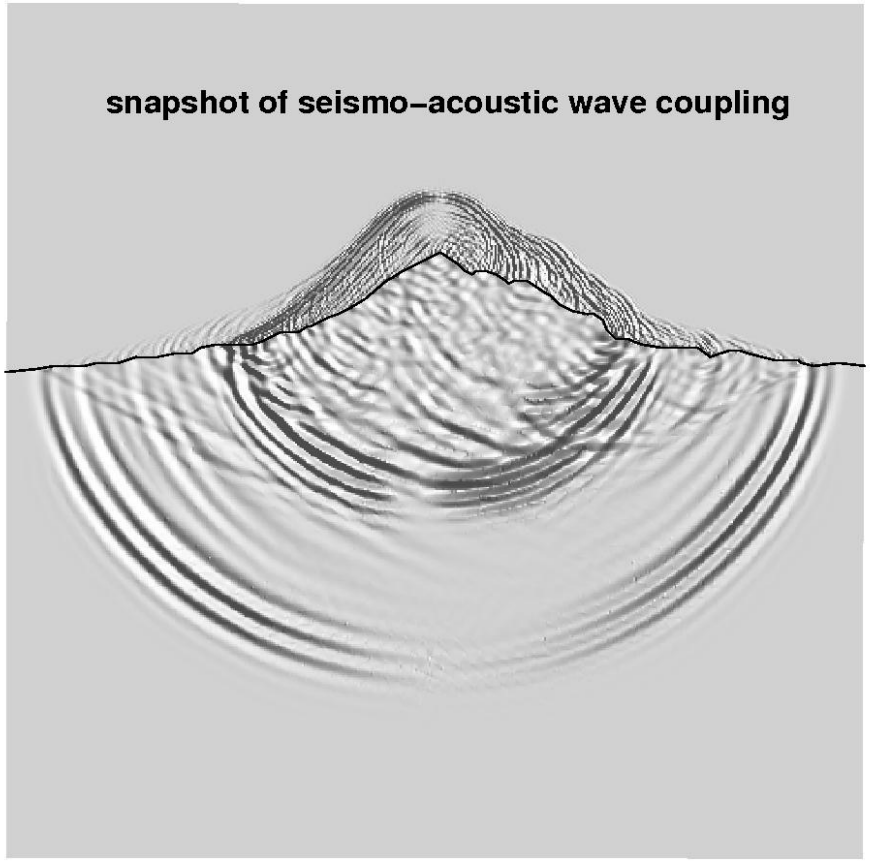


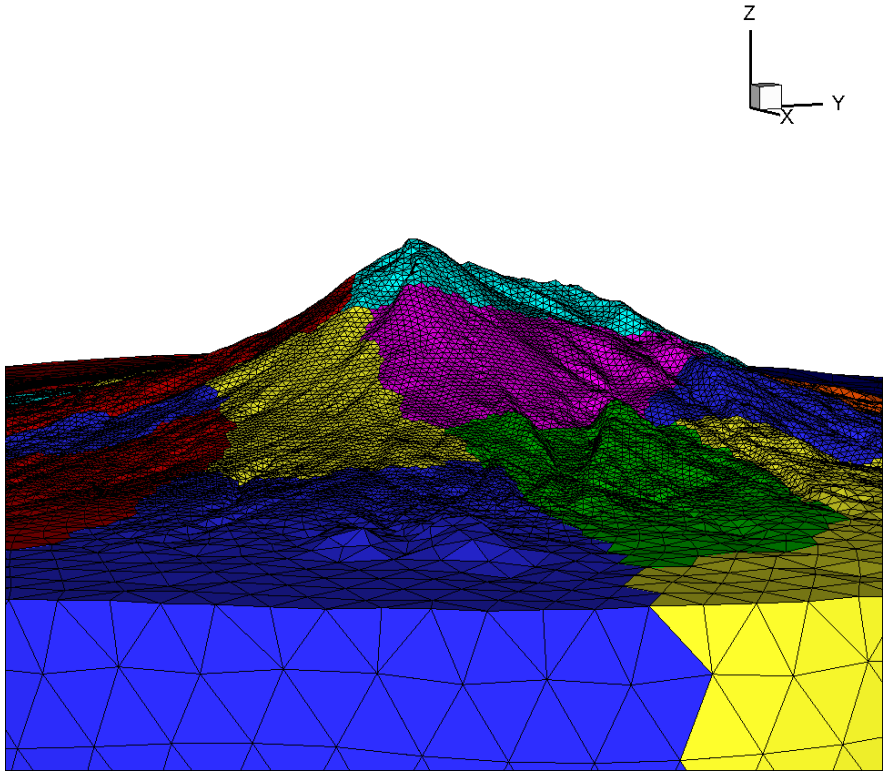


Sonic logging for oil industry – model includes steel casing!

Characteristics of the SeisSol simulation software:

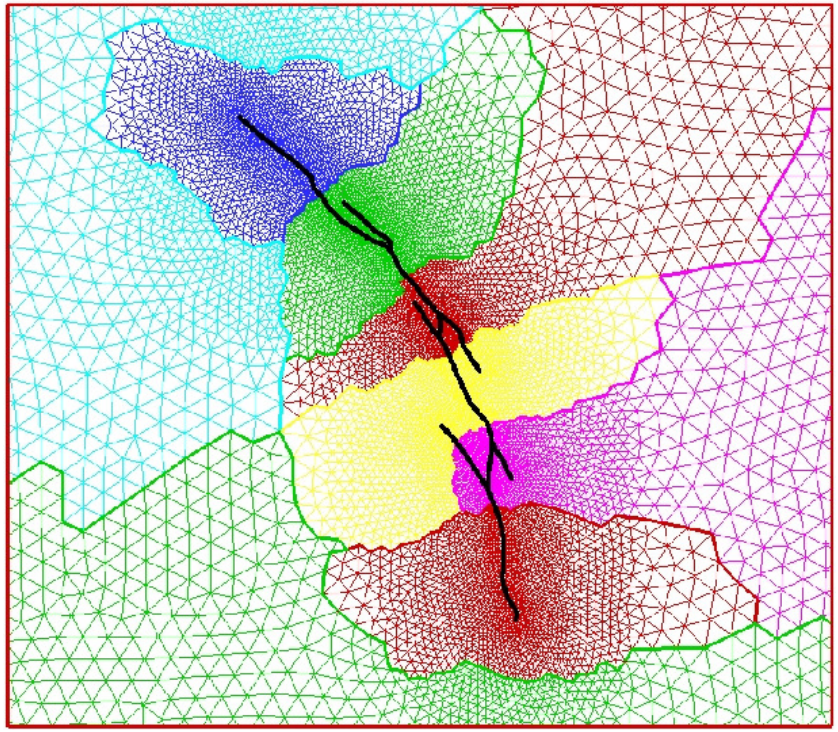
• use of tetrahedral and hexahedral meshes to approximate complex 3D model geometries

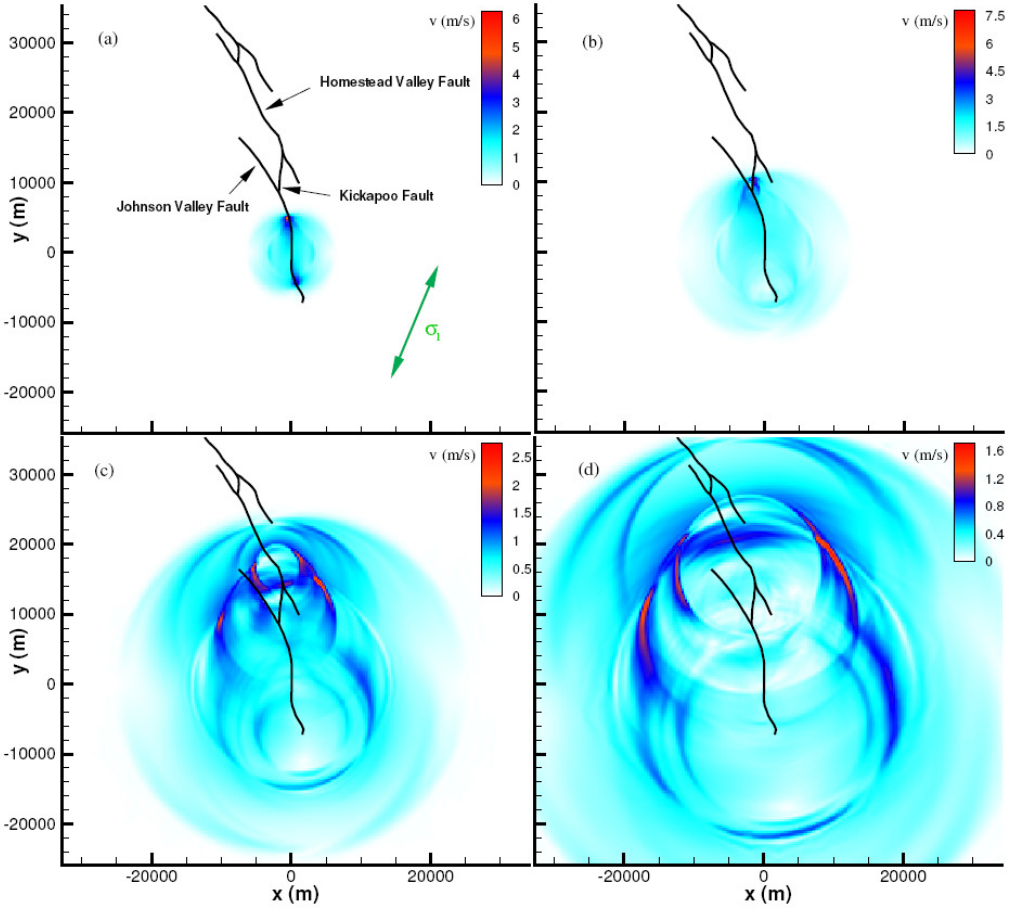




Characteristics of the SeisSol simulation software:

• use of tetrahedral and hexahedral meshes to approximate complex 3D model geometries



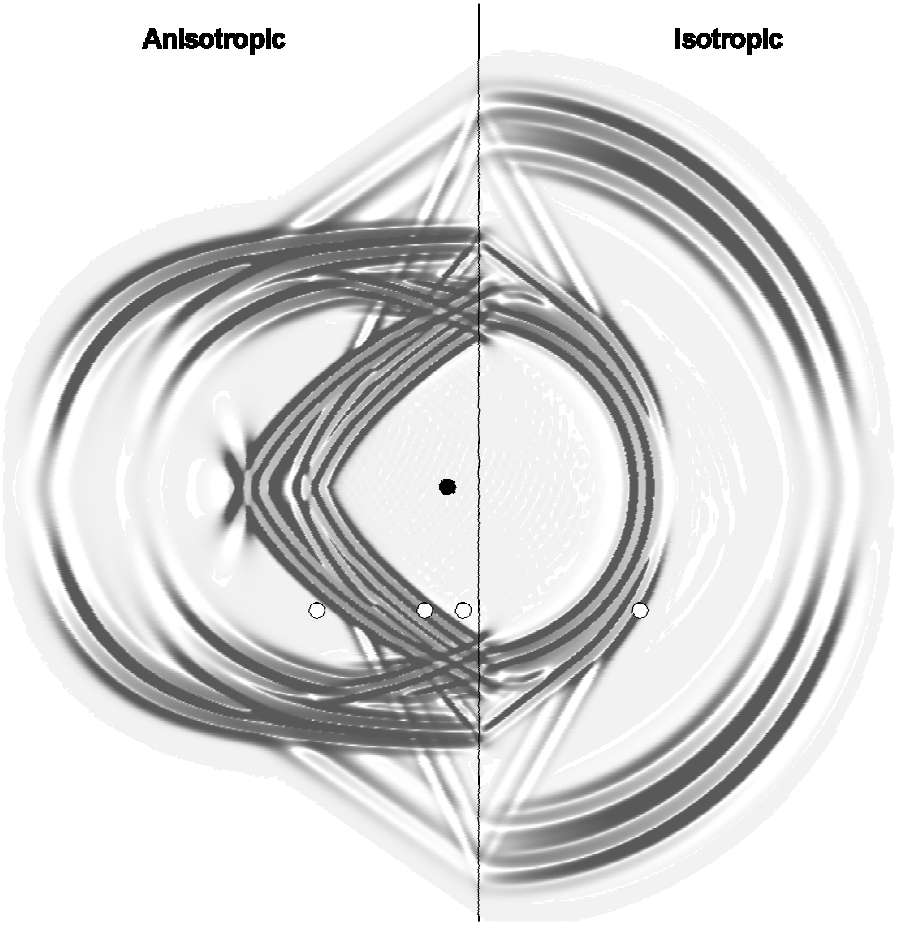


Dynamic rupture simulation of the Landers Earthquake

Characteristics of the SeisSol simulation software:

• use of tetrahedral and hexahedral meshes to approximate complex 3D model geometries

• use of acoustic, elastic, viscoelastic, poroelastic, and anisotropic material to approximate realistic geological subsurface properties

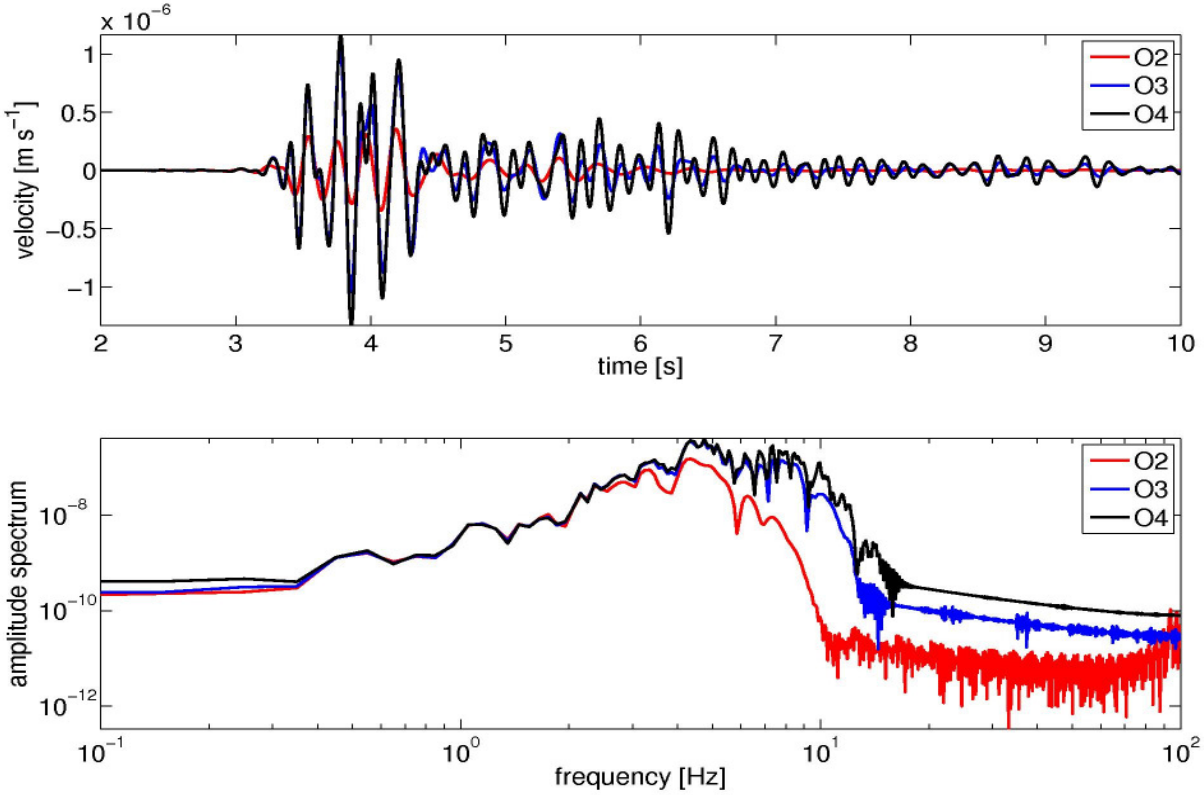


Characteristics of the SeisSol simulation software:

• use of tetrahedral and hexahedral meshes to approximate complex 3D model geometries

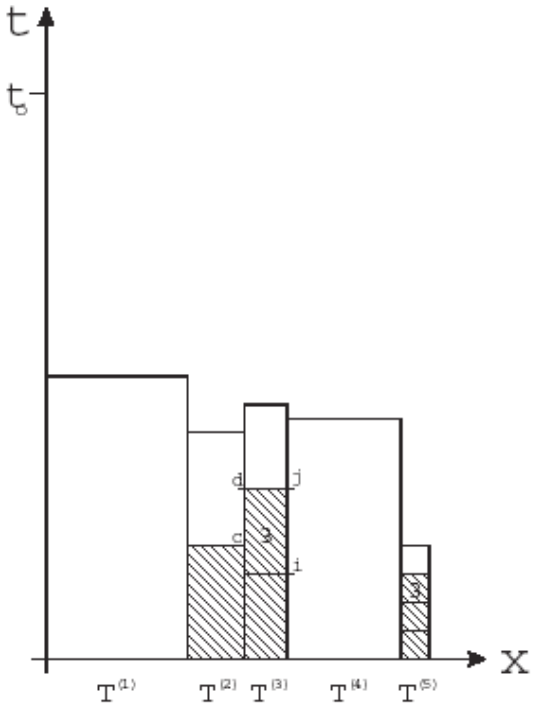
• use of acoustic, elastic, viscoelastic, poroelastic, and anisotropic material to approximate realistic geological subsurface properties

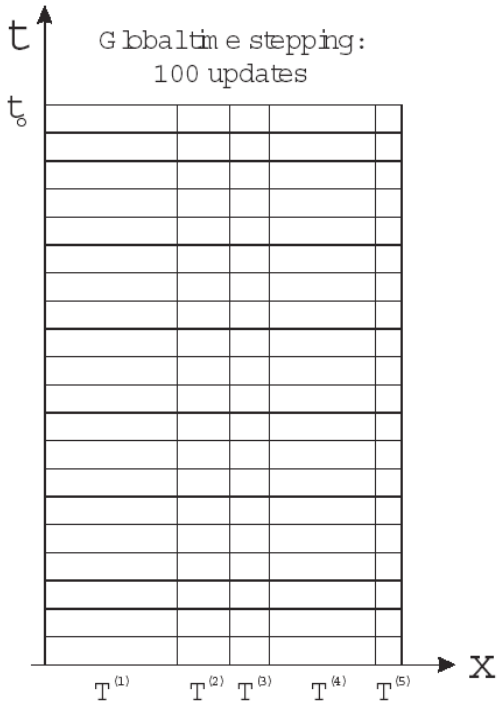
• use of arbitrarily high approximation order in time and space to produce reliable and sufficiently accurate synthetic seismograms or other seismological data set

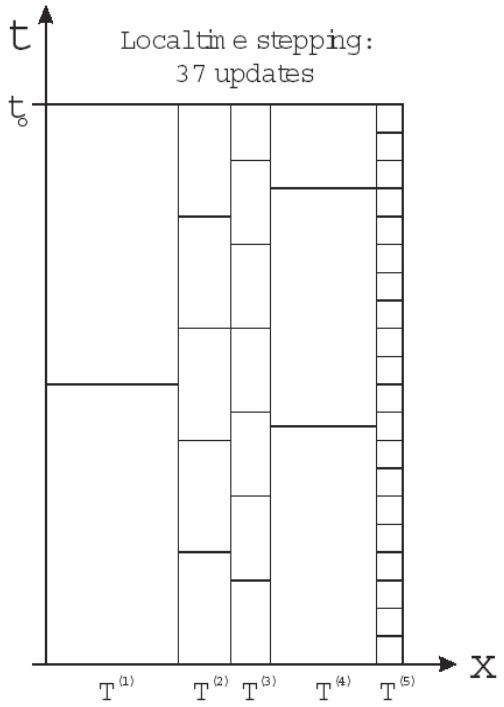


Characteristics of the SeisSol simulation software:

- use of tetrahedral and hexahedral meshes to approximate complex 3D model geometries
- use of acoustic, elastic, viscoelastic, poroelastic, and anisotropic material to approximate realistic geological subsurface properties
- use of arbitrarily high approximation order in time and space to produce reliable and sufficiently accurate synthetic seismograms or other seismological data set
- use of explicit local time step algorithm, such that each element is running its own optimal time step length to reduce computation time



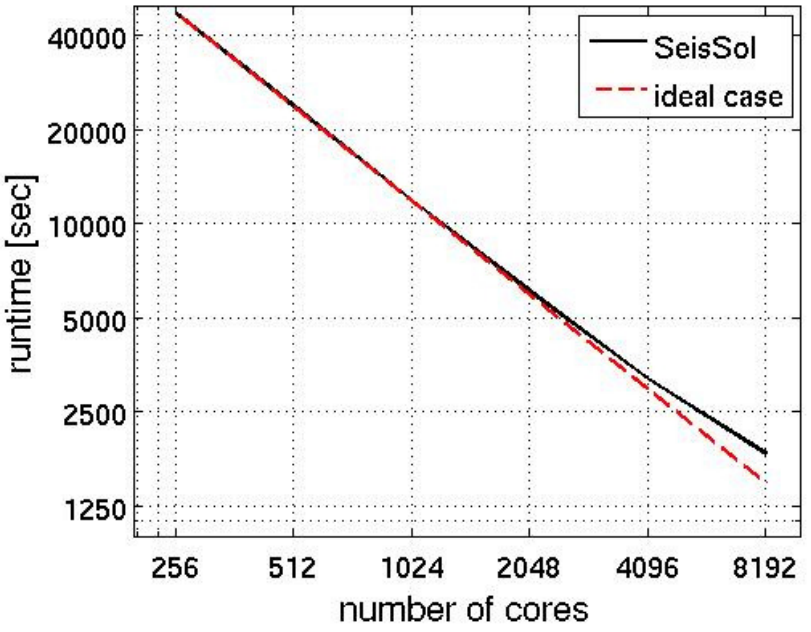


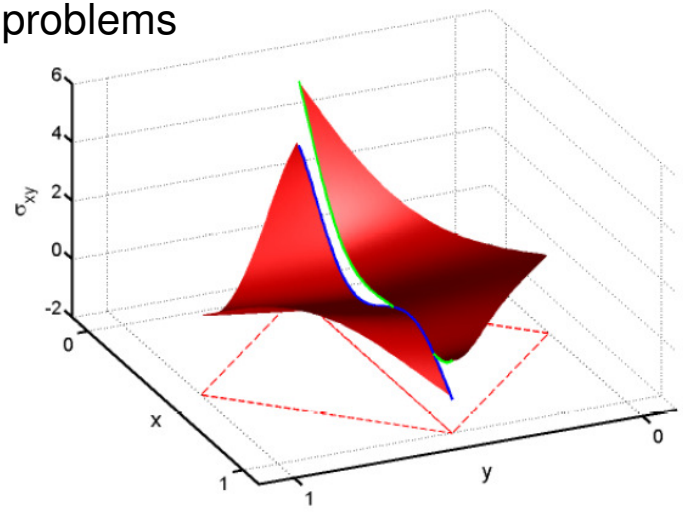


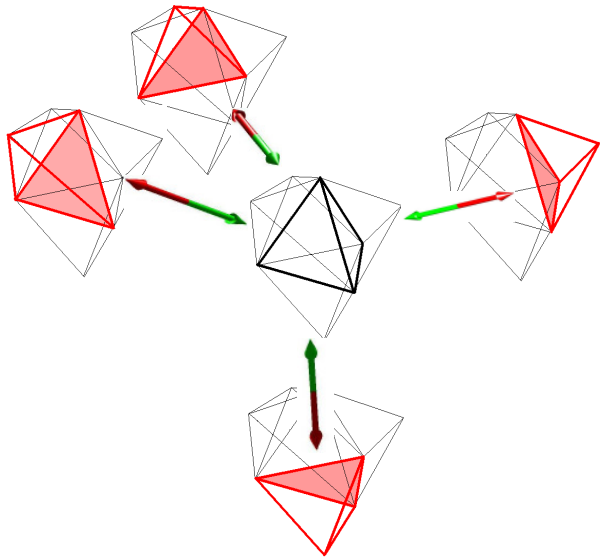
Characteristics of the SeisSol simulation software:

It is based on a numerical approach using the Discontinuous Galerkin Finite Element Method which exhibits the desirable properties of:

- high-order orthogonal polynomial basis functions on tetrahedrons and hexahedrons
- locality of the computations, as only directly neighboring elements are required to exchange data, which leads to small communication times for parallel calculations
- concept of numerical fluxes from the Finite Volume framework that allows us to consider different physical properties occuring in wave propagation problems







Elastic wave equation in 2D

System of **5 equations for 5 unknowns** $\sigma_{xx}\,,\sigma_{yy}\,,\sigma_{xy}\,,u\,,v$

$$
\begin{aligned}
\frac{\partial}{\partial t}\sigma_{xx} - (\lambda+2\mu)\frac{\partial}{\partial x}u - \lambda\frac{\partial}{\partial y}v &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{yy} - \lambda\frac{\partial}{\partial x}u - (\lambda+2\mu)\frac{\partial}{\partial y}v &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{xy} - \mu(\frac{\partial}{\partial x}v + \frac{\partial}{\partial y}u) &= 0\,,\\
\rho\frac{\partial}{\partial t}u - \frac{\partial}{\partial x}\sigma_{xx} - \frac{\partial}{\partial y}\sigma_{xy} &= f_x\,,\\
\rho\frac{\partial}{\partial t}v - \frac{\partial}{\partial x}\sigma_{xy} - \frac{\partial}{\partial y}\sigma_{yy} &= f_y\,,
\end{aligned}
$$

Seismic wave equations in 3D

System of **9 equations for 9 unknowns** $\sigma_{xx}\,,\sigma_{yy}\,,\sigma_{zz}\,,\sigma_{xy}\,,\sigma_{xz}\,,\sigma_{yz}\,,u\,,v\,,w$

$$
\begin{aligned}
\frac{\partial}{\partial t}\sigma_{xx} - (\lambda+2\mu)\frac{\partial}{\partial x}u - \lambda\frac{\partial}{\partial y}v - \lambda\frac{\partial}{\partial z}w &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{yy} - \lambda\frac{\partial}{\partial x}u - (\lambda+2\mu)\frac{\partial}{\partial y}v - \lambda\frac{\partial}{\partial z}w &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{zz} - \lambda\frac{\partial}{\partial x}u - \lambda\frac{\partial}{\partial y}v - (\lambda+2\mu)\frac{\partial}{\partial z}w &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{xy} - \mu(\frac{\partial}{\partial x}v + \frac{\partial}{\partial y}u) &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{xz} - \mu(\frac{\partial}{\partial z}u + \frac{\partial}{\partial x}w) &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{yz} - \mu(\frac{\partial}{\partial z}v + \frac{\partial}{\partial y}w) &= 0\,,\\
\rho\frac{\partial}{\partial t}u - \frac{\partial}{\partial x}\sigma_{xx} - \frac{\partial}{\partial y}\sigma_{xy} - \frac{\partial}{\partial z}\sigma_{xz} &= f_x\,,\\
\rho\frac{\partial}{\partial t}v - \frac{\partial}{\partial x}\sigma_{xy} - \frac{\partial}{\partial y}\sigma_{yy} - \frac{\partial}{\partial z}\sigma_{yz} &= f_y\,,\\
\rho\frac{\partial}{\partial t}w - \frac{\partial}{\partial x}\sigma_{xz} - \frac{\partial}{\partial y}\sigma_{yz} - \frac{\partial}{\partial z}\sigma_{zz} &= f_z\,,
\end{aligned}
$$

Discretization

What numerical methods do you know or have you heard of ?

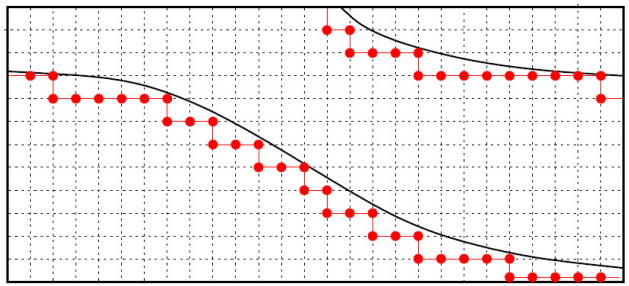
Finite Differences (FD), Finite Elements (FE), Finite Volumes (FV), etc.

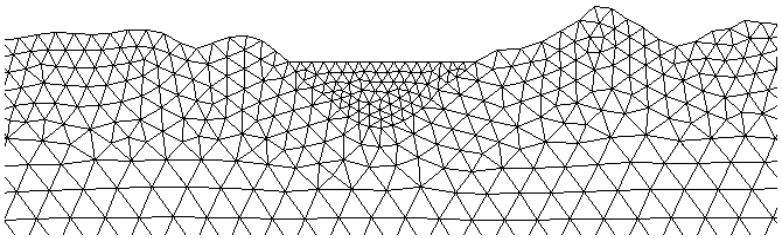
What do numerical method usually require ?

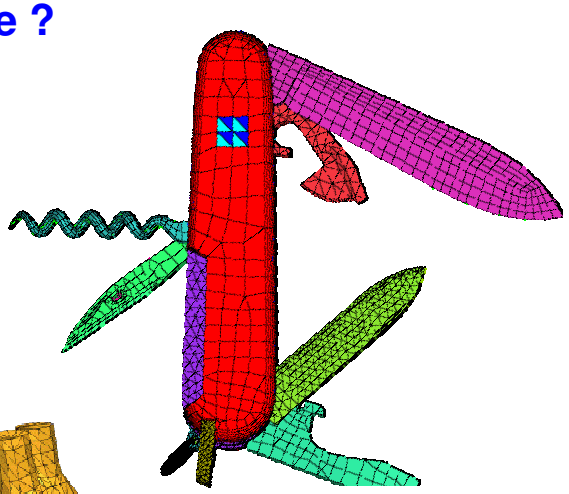
points or particles, regular grids, unstructured meshes, etc.

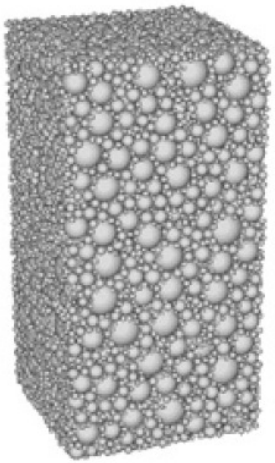
What is an *"unstructured mesh"* ?

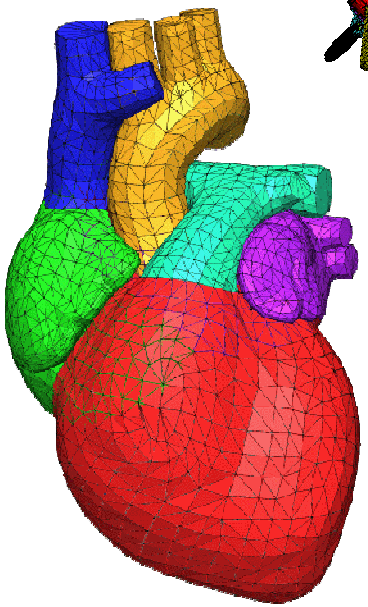
A mesh where element indices do not show a logical structure

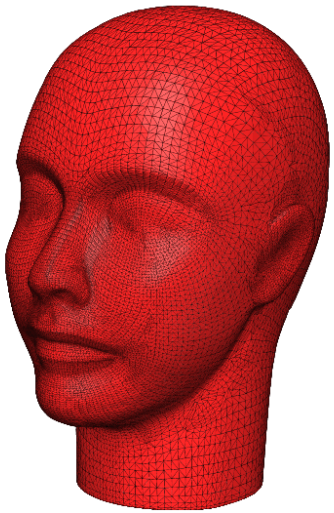








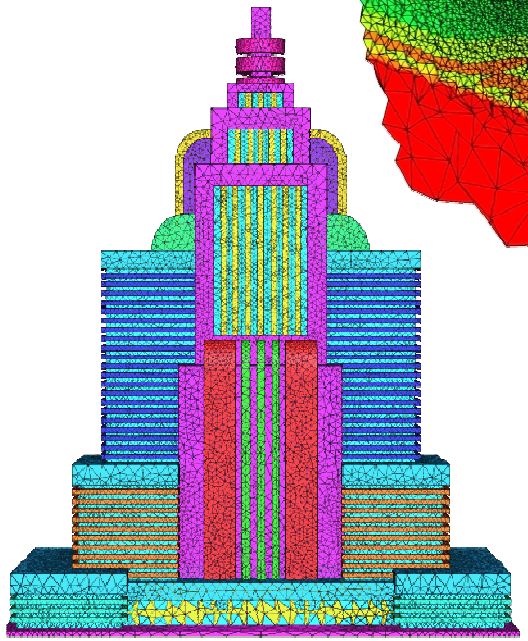


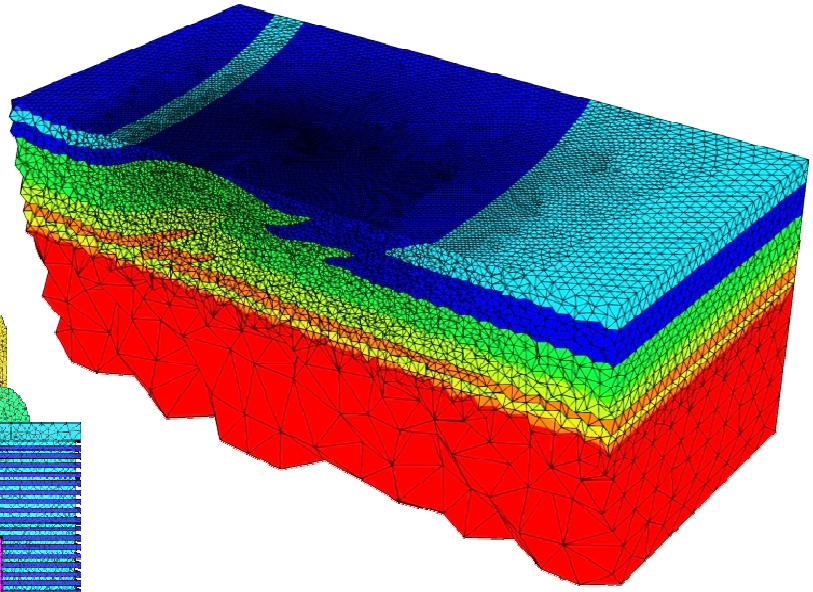


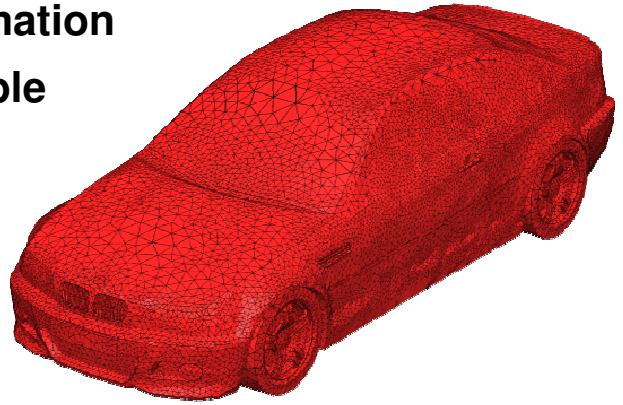
Discretization

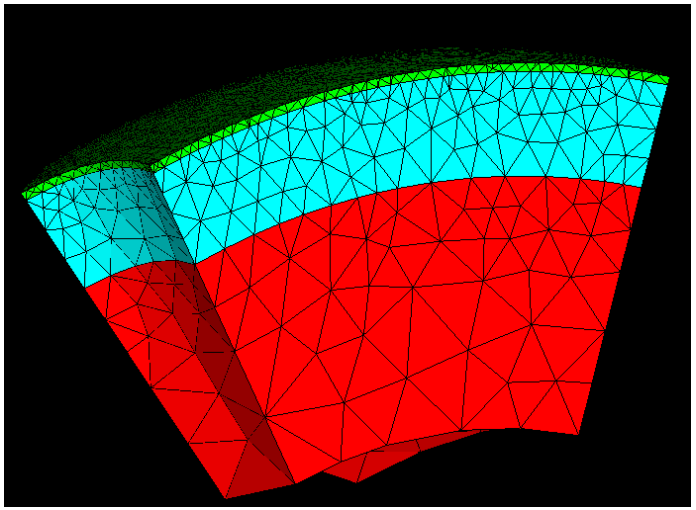
The discretization subdivides a continuous physical model into a set of points, elements, control volumes, cells, etc.

- ➔ **numerical methods approximate functions discretely on these points or elements**
- ➔ **the finer the mesh, the more accurate the approximation**
- ➔ **unstructured meshes are geometrically more flexible**





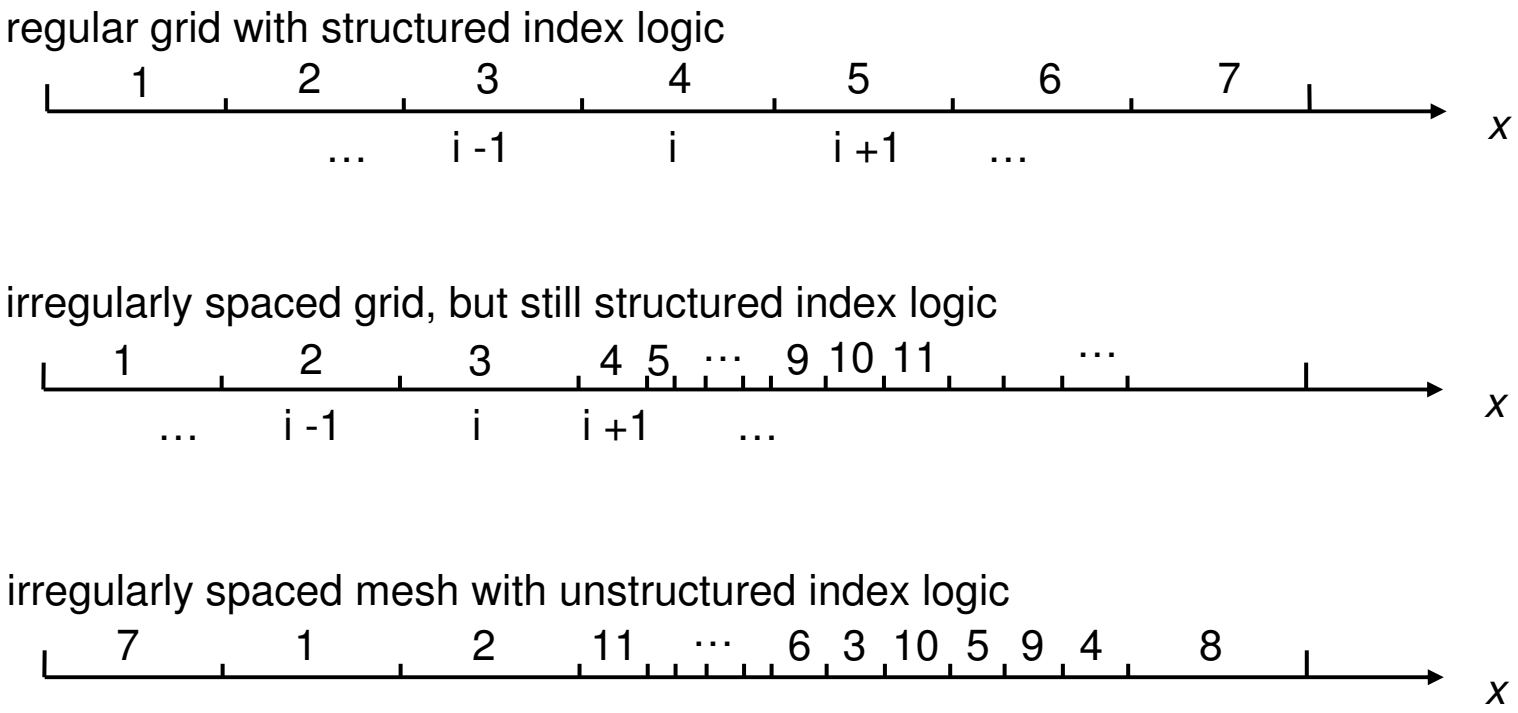




Discretization of a physical model in 1D

Discretization (= the process of Mesh Generation) is very simple in 1D

➔ a line just has to be subdivided into a set of intervals



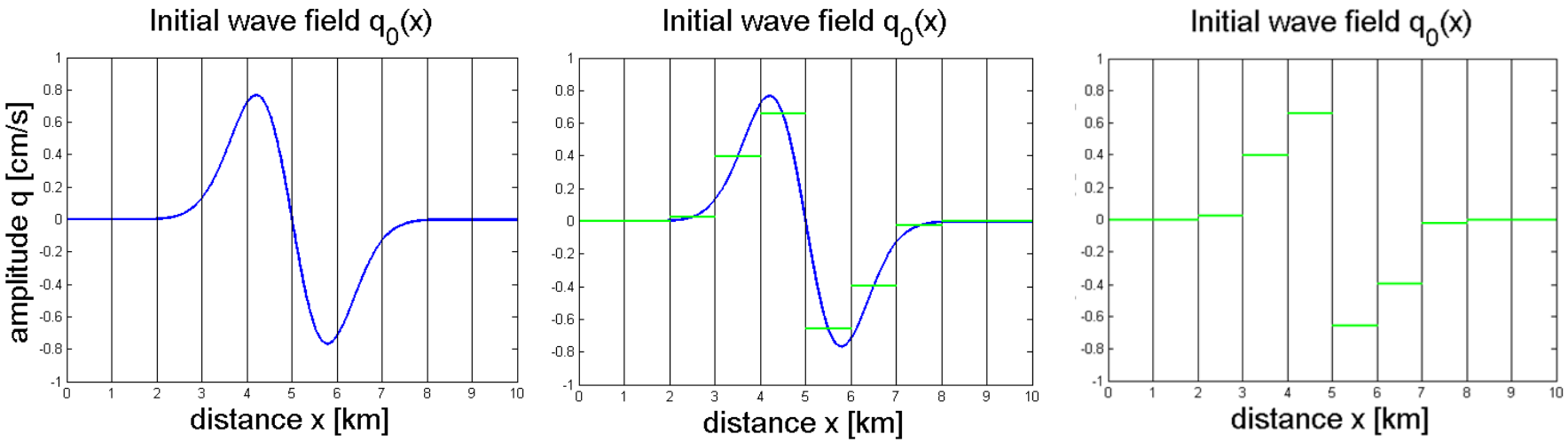
Polynomial Approximation of Functions

The approximation can be done by different basis functions, however, in many cases polynomial functions are used for the approximation.

$$P(x) = c_0 + c_1 x + c_2 x^2 + c_2 x^3 + \ldots,$$

monomial basis, usually is never used

The highest exponent occurring defines the degree of the polynomial !

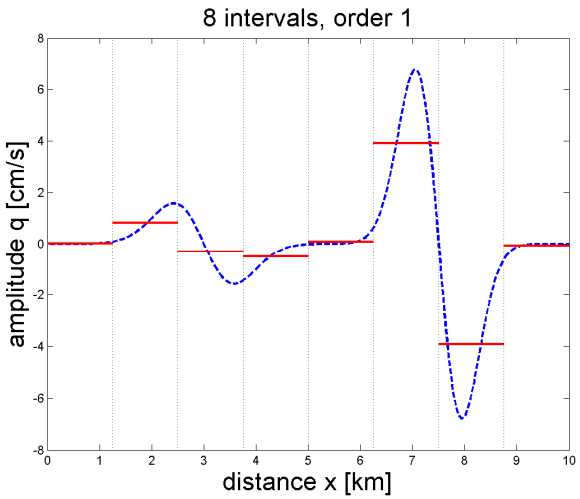


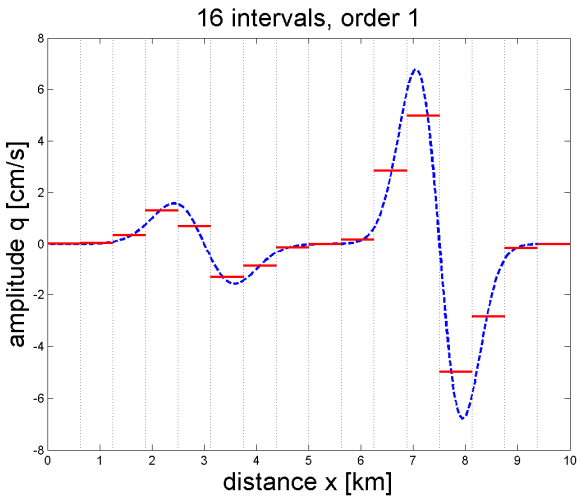
the most simple case is a **piecewise constant approximation**, i.e. the chosen polynomial is a **polynomial of degree 0**.

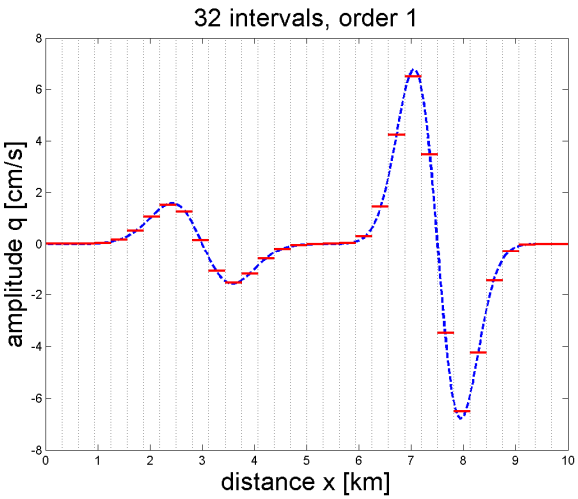
In the above case the approximation criterion is that the **constant is equal to the function value at the center of the interval**:

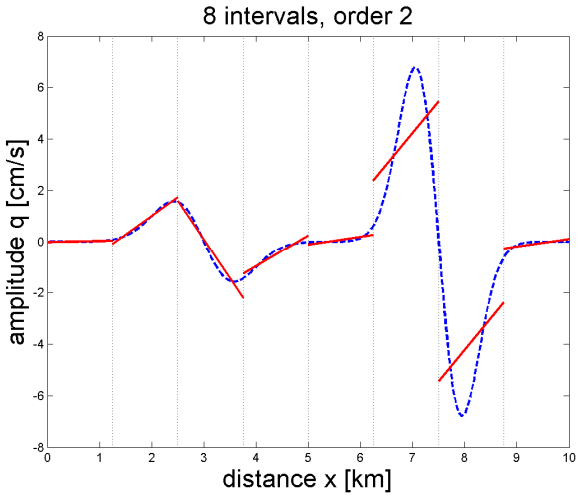
$$q_0(x_{c_I}) = c_0$$

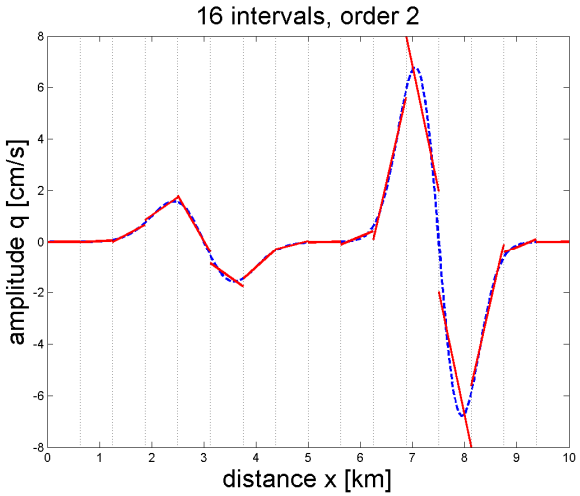
Improved Numerical Approximation by Higher Order



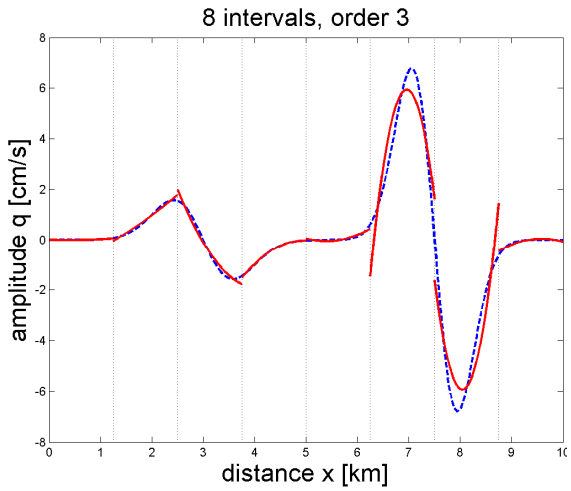


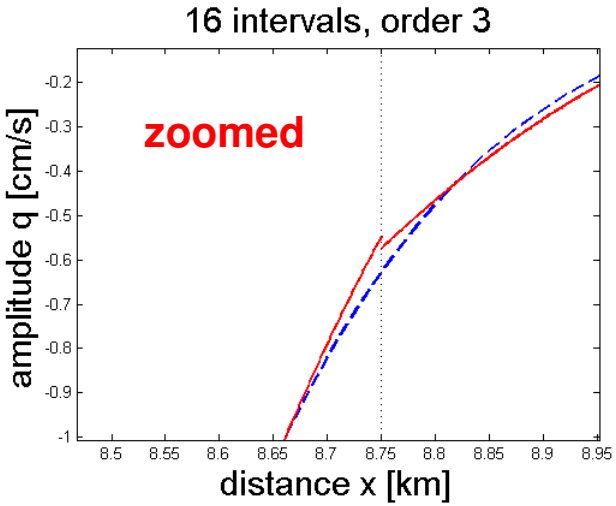


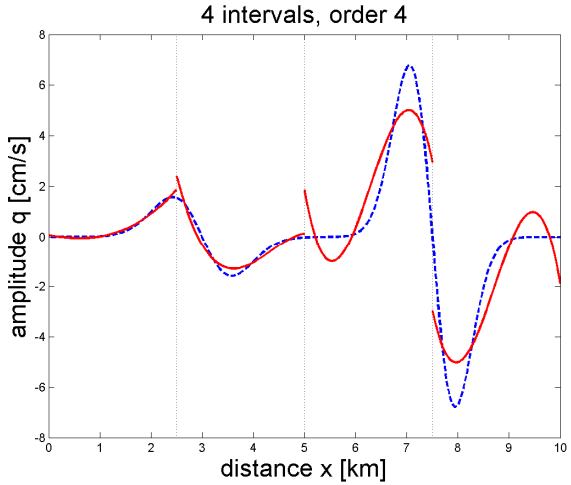


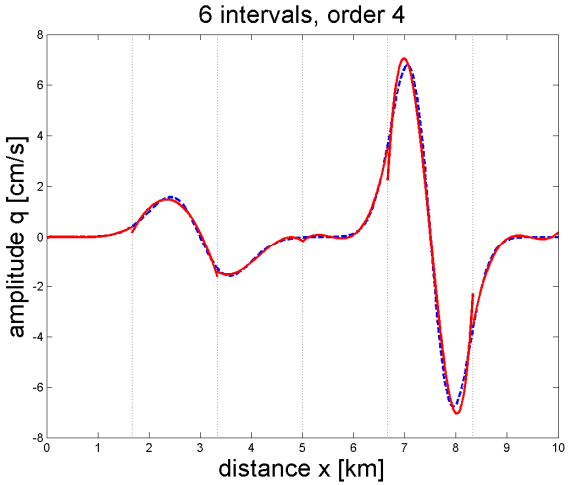


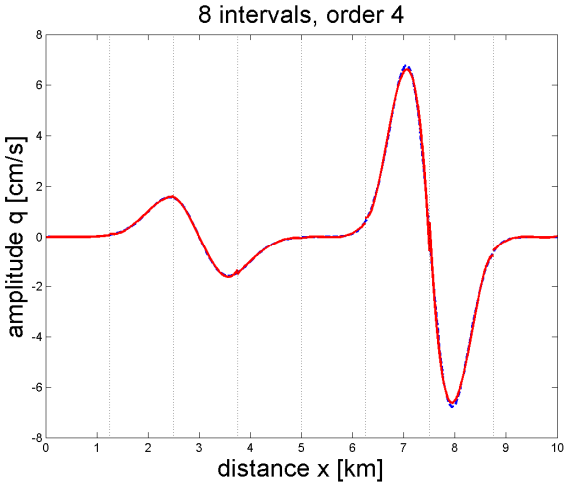
Improved Numerical Approximation by Higher Order



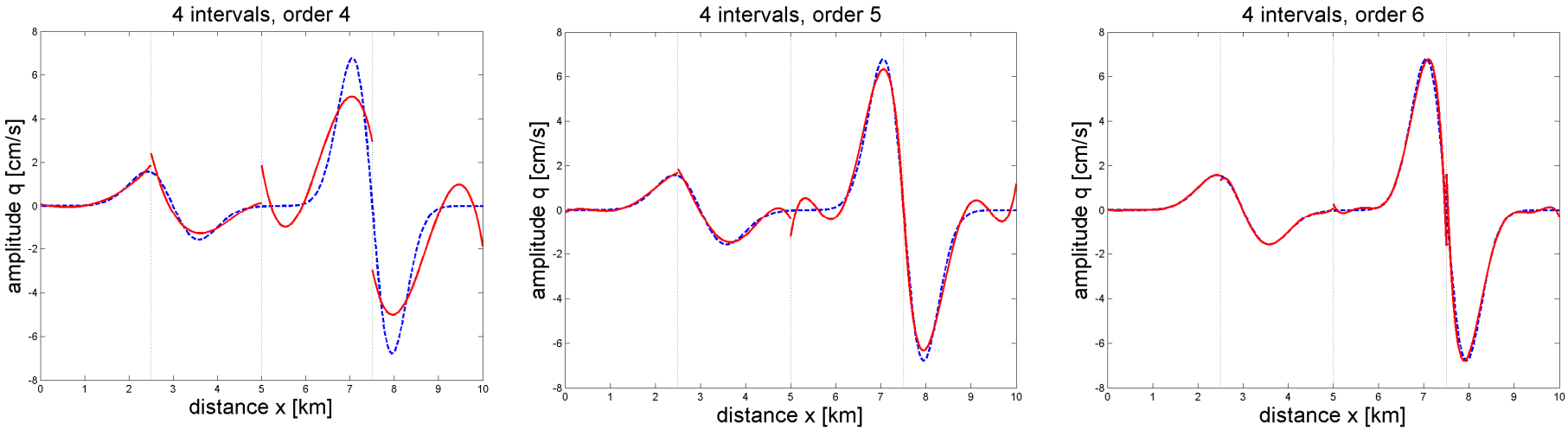








Improved Numerical Approximation by Higher Order

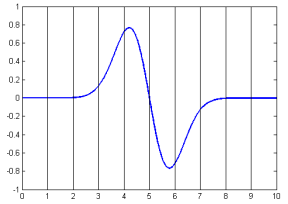


➔ **mesh refinement** and increase of the approximation **order improve** the approximation

➔ **which approach is more efficient** is often a **difficult question to answer** and has to be studied in detail, however, in DG **coarse meshes and high oders pay off**!

➔ However, **first-order methods usually are very diffusive and of no use for practical problems**, where some reasonable accuracy is required !!!

Polynomial Approximation in the DG framework



Inside each **interval I (=element)** of our discretization, we approximate the solution ***q(x,t)*** numerically by using the **reference interval I_R and a linear combination of**

- **time-independent polynomial functions** $\phi_l(\xi)$ in 1D: Legendre Polynomials P_n) as a set of basis functions up to **maximum degree *N***.
- **time-dependent degrees of freedom (DOF)** $\hat{q}_l(t)$

such that we get

$$q(\xi,t) \approx \hat{q}_l(t)\cdot\phi_l(\xi) = \sum_{l=0}^{N} \hat{q}_l(t)\cdot\phi_l(\xi)$$

$$= \hat{q}_0(t)\cdot\phi_0(\xi) + \hat{q}_1(t)\cdot\phi_1(\xi) + \cdots$$

"Einstein summation convention"

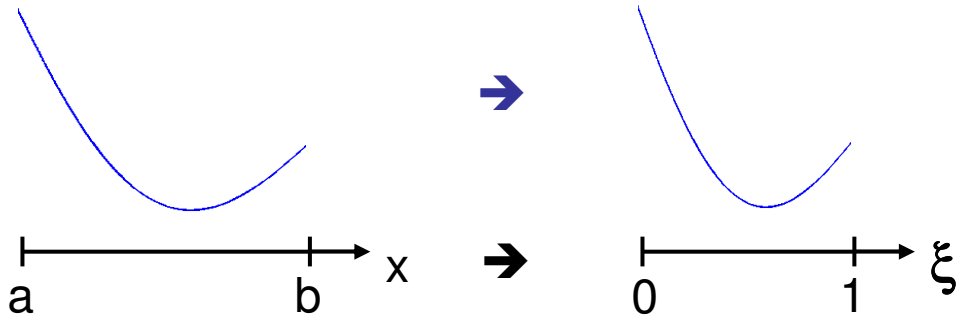
Note that the **physical space in I** is denoted by ***x*** whereas the **reference space in the unit interval I_R** is denoted by ξ

Polynomial Approximation in the DG framework

$$q(\xi,t) \approx \hat{q}_l(t)\cdot\phi_l(\xi) = \sum_{l=0}^{N} \hat{q}_l(t)\cdot\phi_l(\xi)$$
$$= \hat{q}_0(t)\cdot\phi_0(\xi) + \hat{q}_1(t)\cdot\phi_1(\xi) + \cdots$$

This approximation is one of the **key ingredients for the extension** of the **Discontinuous Galerkin Finite Element method to high-order accuracy.**

The **mapping into a reference interval** (reference element), where the basis functions are defined **leads to an elegant treatment of complicated discretizations** of physical models with deformed or variable-size element.

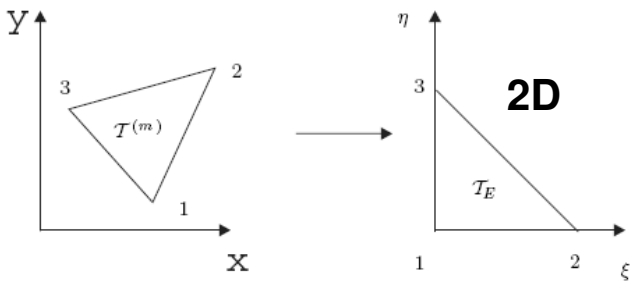


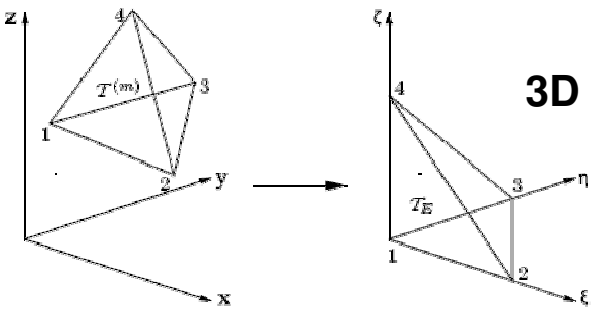
Polynomial Approximation in the DG framework

$$q(\xi,t) \approx \hat{q}_l(t)\cdot\phi_l(\xi) = \sum_{l=0}^{N}\hat{q}_l(t)\cdot\phi_l(\xi)$$
$$= \hat{q}_0(t)\cdot\phi_0(\xi)+\hat{q}_1(t)\cdot\phi_1(\xi)+\cdots$$

This approximation is one of the **key ingredients for the extension** of the **Discontinuous Galerkin Finite Element method to high-order accuracy.**

The **mapping into a reference interval** (reference element), where the basis functions are defined **leads to an elegant treatment of complicated discretizations** of physical models with deformed or variable-size element.





Approximation Order and Numerical Convergence

To **get correct high-order approximations is not easy** and implementation has to be carried out with caution.

➔ to **check if the approximation method works correctly** and reaches the expected order of accuracy, one has to **compute the errors of a numerical solution compared to an exact solution** on a series of refined meshes

➔ **convergence test**

The **approximation error E** can be written as a function of the **mesh spacing h** and the **order *O*** as:

$$E = h^O + C$$

Determining the **errors E_1 and E_2 for two different mesh spacings h_1 and h_2** gives

$$\begin{aligned} E_1 &= h_1^O + C \\ E_2 &= h_2^O + C \end{aligned} \quad \rightarrow \quad \begin{aligned} \log E_1 &= O \log h_1 + \log C \\ \log E_2 &= O \log h_2 + \log C \end{aligned}$$

and the order *O* can be computed as

$$O = \frac{\log\left(\frac{E_1}{E_2}\right)}{\log\left(\frac{h_1}{h_2}\right)}$$

Convergence Test and Error Norms

The **functions of the errors E_1 and E_2** with respect to the **mesh spacings h_1 and h_2** have the **form of a straight line on a logarithmic scale.**

$$\log E_1 = O \log h_1 + \log C$$
$$\log E_2 = O \log h_2 + \log C$$

Question:
What is *O* representing ?

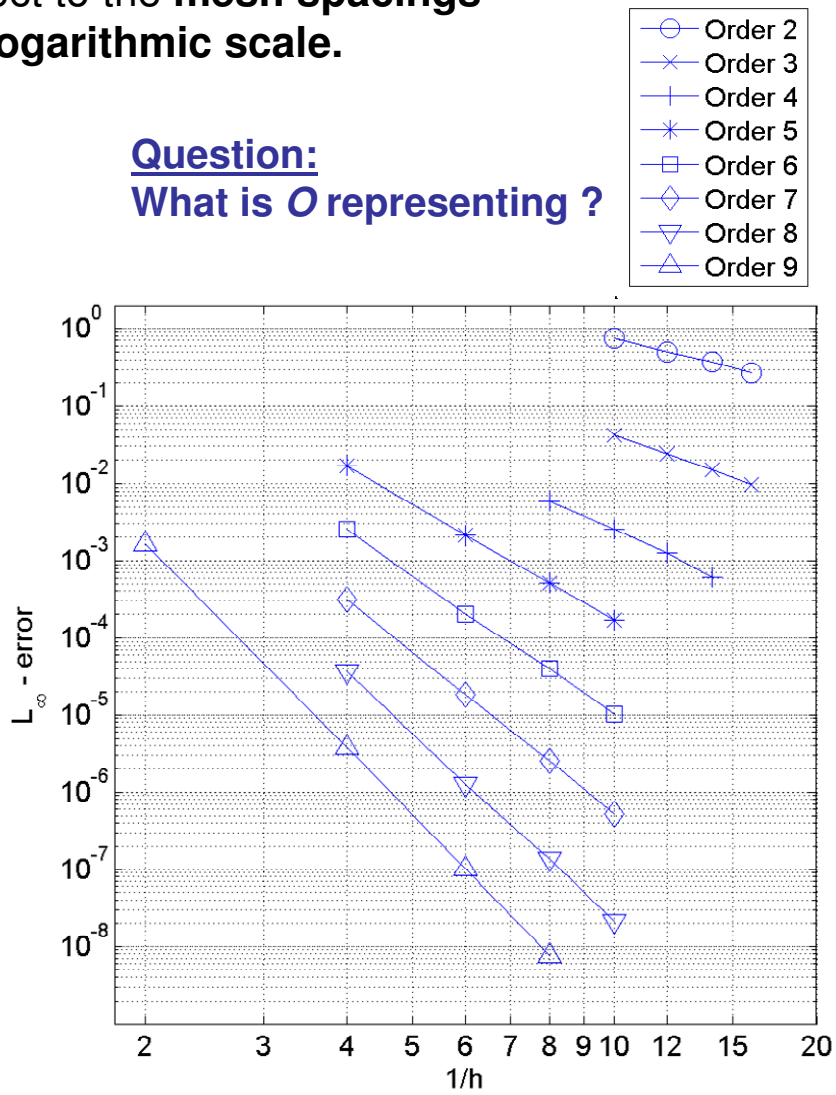
The **order O can then be interpreted as the slope** of the error lines.

The **discrete error at all points *i*** between the **exact q^e** and the **numerical q^n** solution can be measured in different norms, e.g.

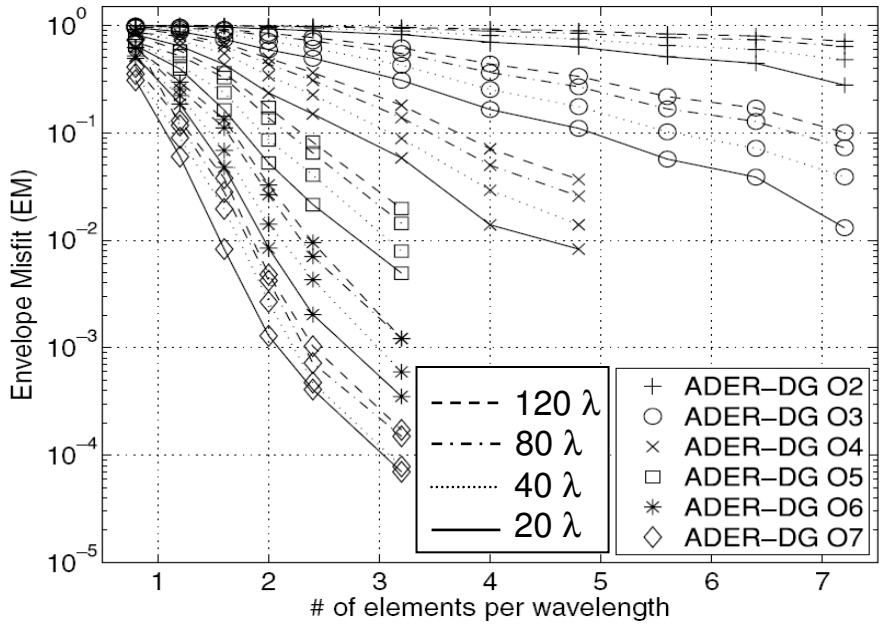
$$\|E\|_p = \left(h \sum_{i=-\infty}^{\infty} |q_i^e - q_i^n|^p \right)^{\frac{1}{p}}$$

often the ***p=2,*** or the **maximum norm** are used

$$\|E\|_\infty = \max_{-\infty<i<\infty} |q_i^e - q_i^n|$$



Approximation Order and Numerical Convergence

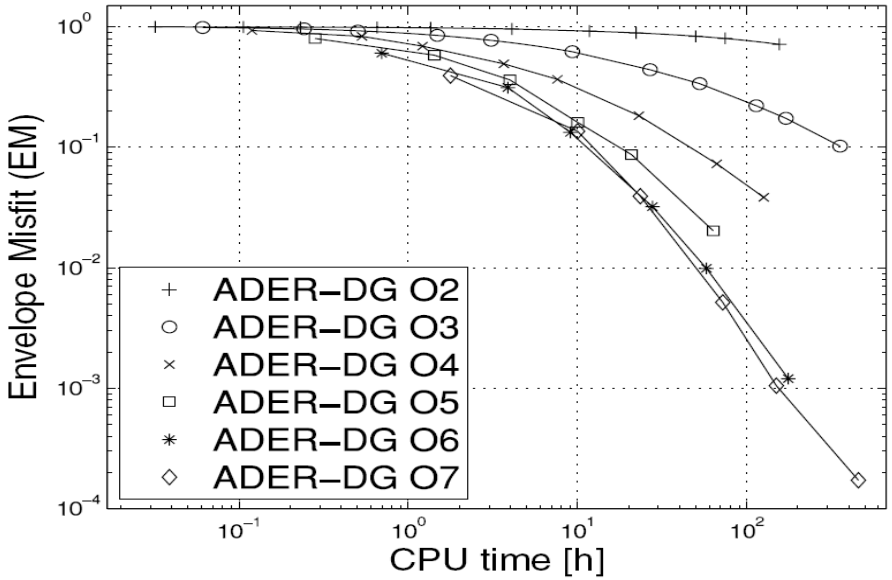


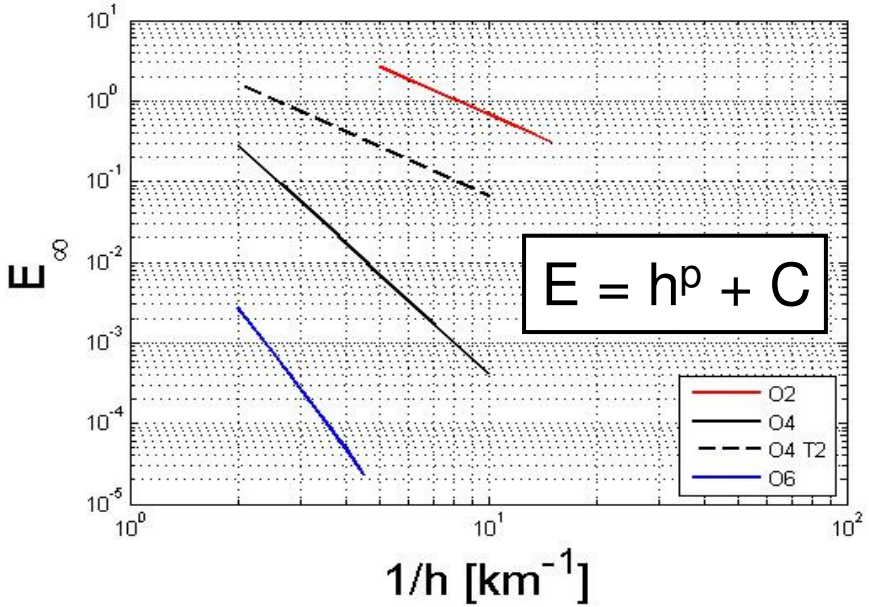
Accuracy depence on:

- spatial sampling (mesh spacing)
- distance of propagated wavelength
- approximation order

2 Elements per wavelength at O5

high order pays off !



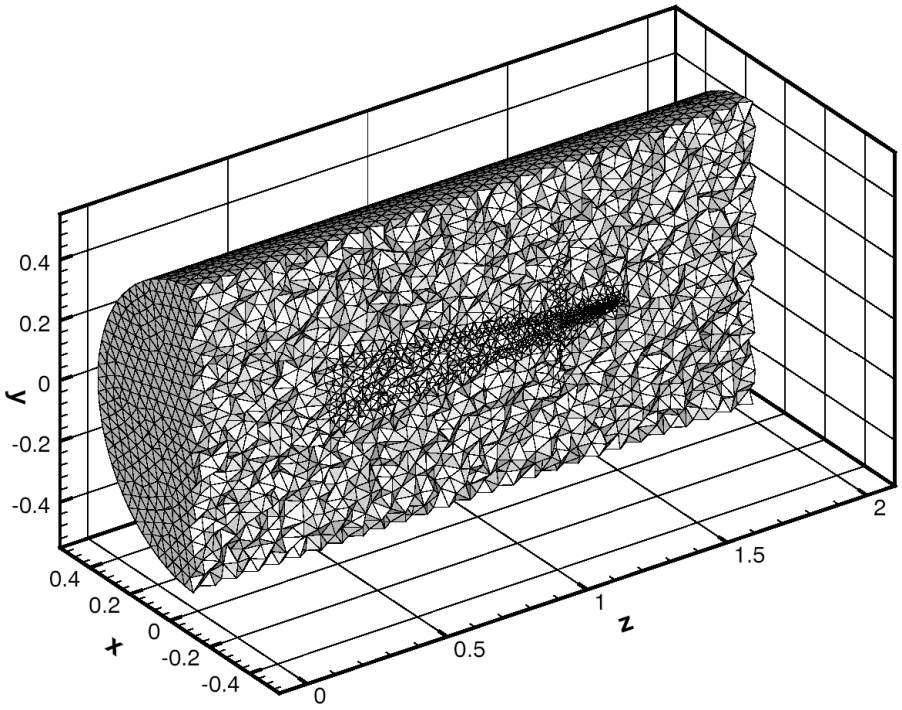


Time step length

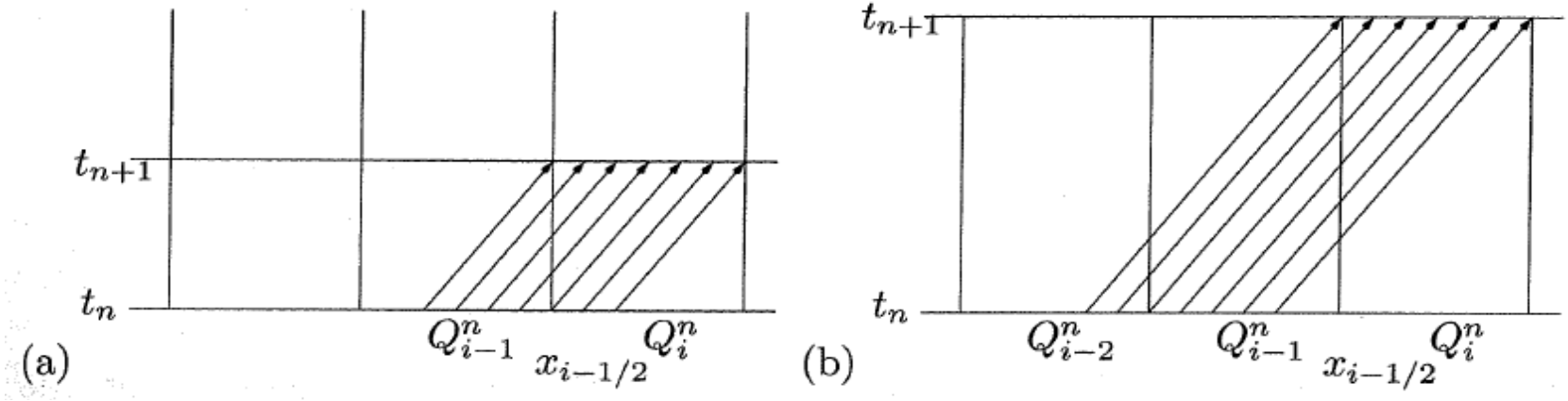
Every **tetrahedron element** (*m*) has its own time step

$$\Delta t^{(m)} < \frac{1}{2N+1} \cdot \frac{l_{min}^{(m)}}{a_{max}^{(m)}},$$

with l_{min} : radius of insphere
a_{max} : highest wave velocity
N : polynomial degree



The condition in (a) is fulfilled and is violated in (b).

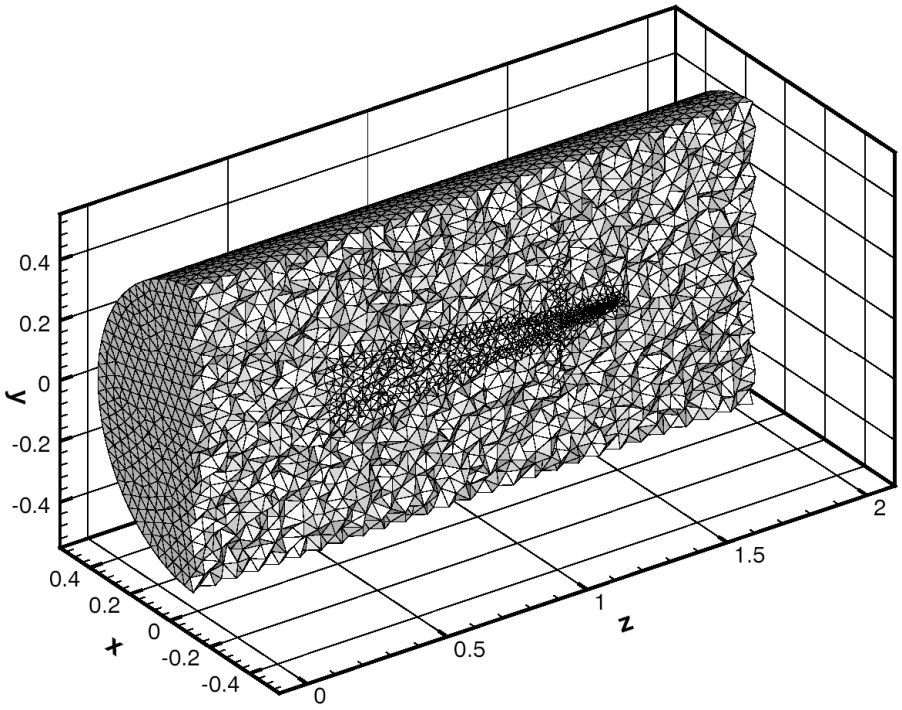


Time step length

Every **tetrahedron element** (*m*) has its own time step

$$\Delta t^{(m)} < \frac{1}{2N+1} \cdot \frac{l_{min}^{(m)}}{a_{max}^{(m)}},$$

with l_{min} : radius of insphere
a_{max} : highest wave velocity
N : polynomial degree



Time step criteria is implemented. However, for stability reasons a smaller time step has to be chosen: CFL-number

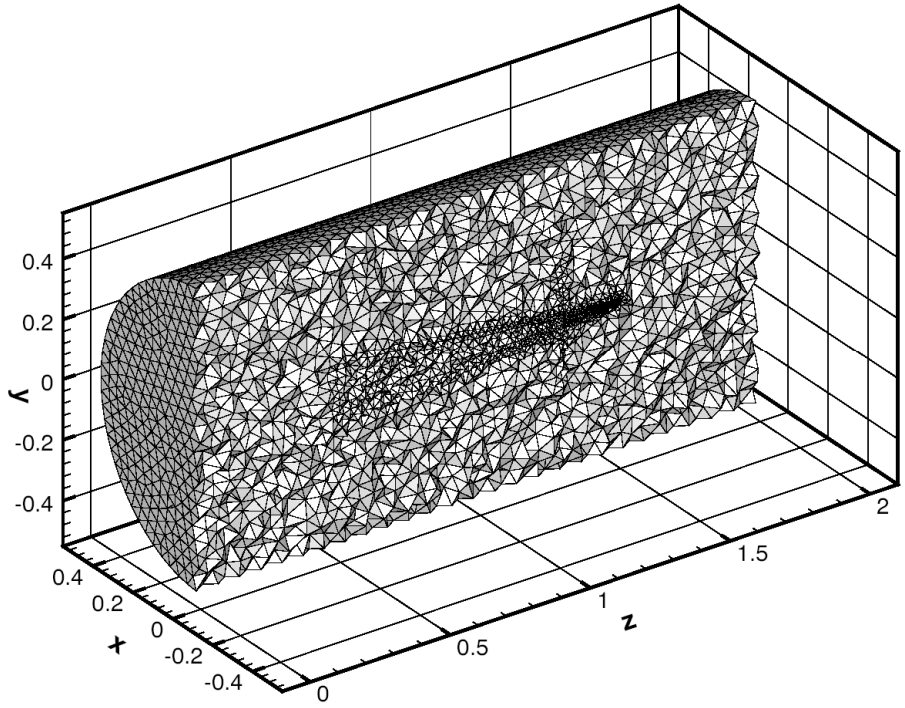
Rule of thumb: O3 CFL-number = 0.5

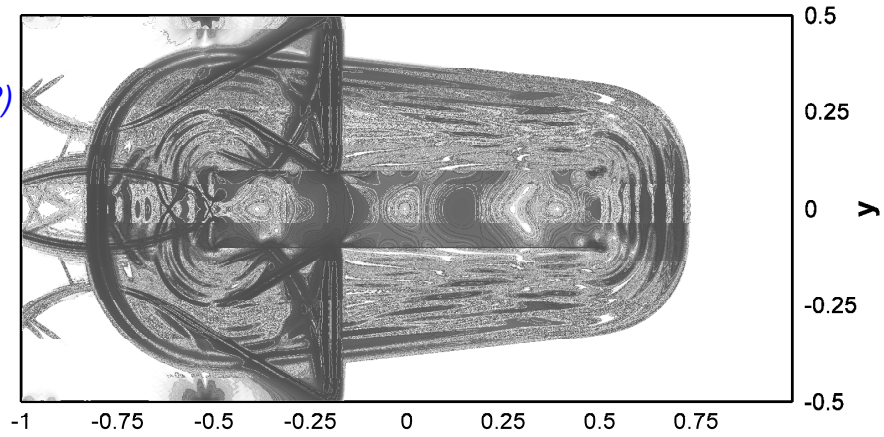
Empirical: for higher-order smaller numbers, but as high as possible!

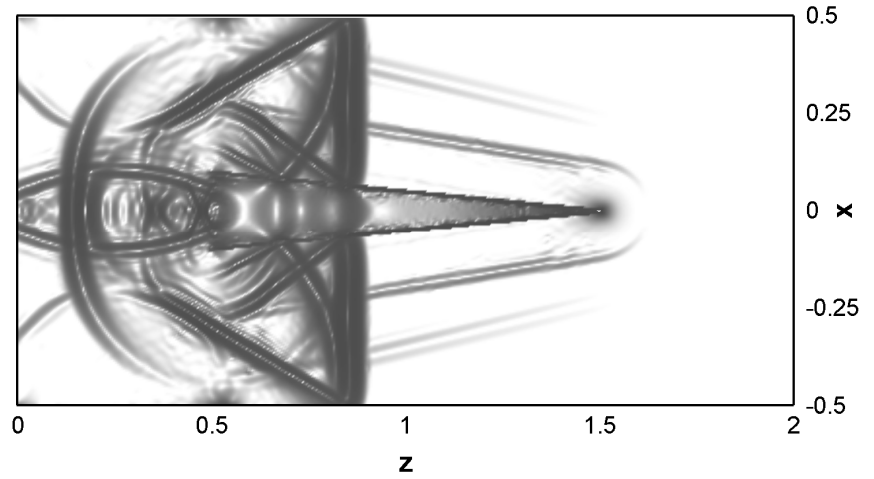
ADER-DG enables the use of local time stepping!

Example: Stiff inclusion

modified after LeVeque 2002)







Number of element-updates:

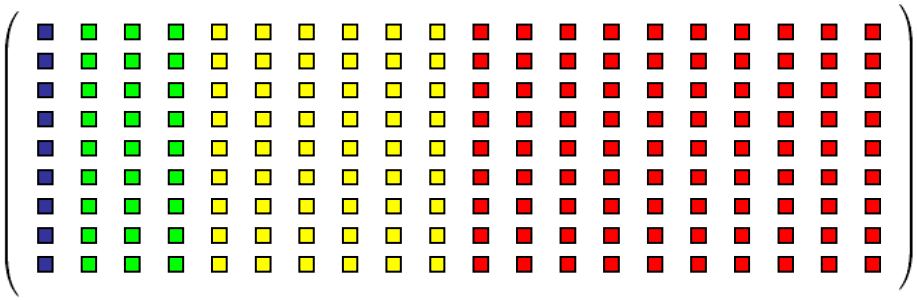
- 72 *10^9 for order 6 with **global** time step
- 95 *10^7 for order 6 with **local** time step

Speed-up: ~100 !

Hierarchy of Matrices – high order is not for free

Matrices in the Discontinuous Galerkin scheme depend on the approximation order O, i.e. on the degree N of the polynomials.
e.g. matrix of degrees of freedom:

$$\hat{Q}_{pl}^{(m)} =$$



O = 1, i.e. N = 0

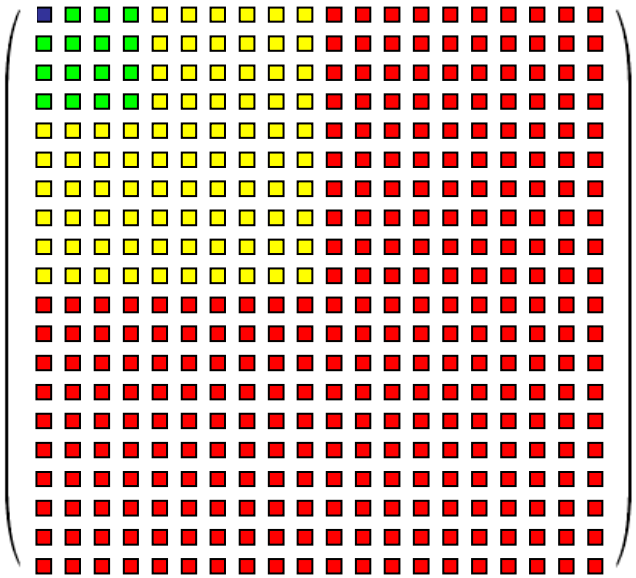
O = 2, i.e. N = 1

O = 3, i.e. N = 2

O = 4, i.e. N = 3

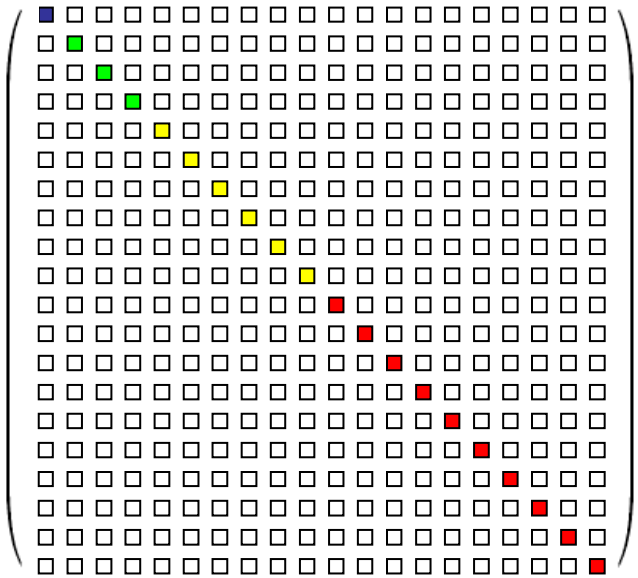
e.g. stiffness and flux matrices

$$K_{kl}^{\xi} =$$



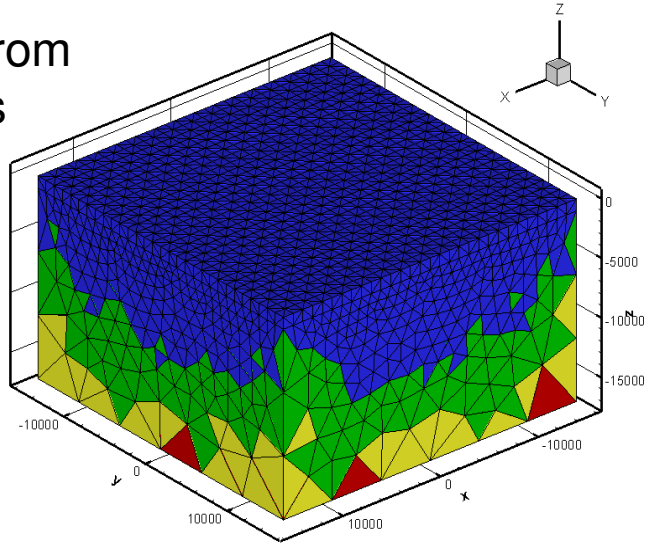
e.g. mass matrix

$$M_{kl} =$$



Pτ-Adaptivität

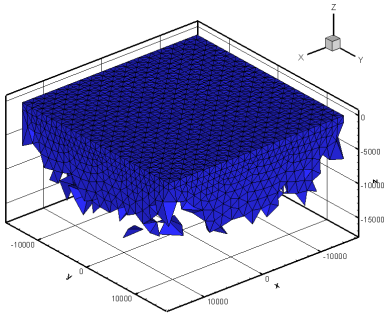
- Adaptation of the approximation order, e.g. ranging from **O4** to **O7**, according to the **insphere radius**, which is responsible for **time step criterion**



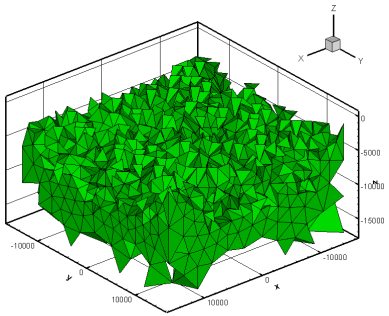
→ 2 x larger time step with respect to pure **O7**

→ only 28% of number of degrees of freedom with respect to pure **O7**: 400 000 instead of 1.4 million

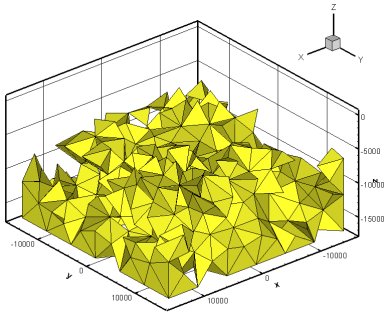
→ 6 x faster than pure **O7** simulation, for a similar accuracy of the results



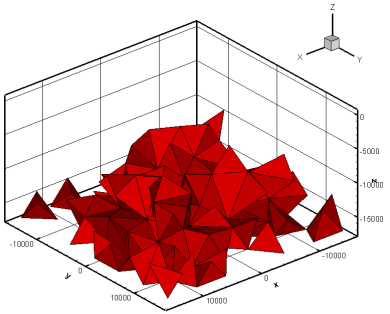
O4



O5



O6



O7

Parallel computing

How many computers do I have to use?

• Depends on order of accuracy and system architecture (memory, CPU frequency, I/O speed, inter-nodal communication speed)

• and has to be figured out individually (at least 2 simulations to recognize a trend – better or worse efficiency?).

More is better?

Not always! More nodes increase the communication time and, thus, decrease the time for real computations.

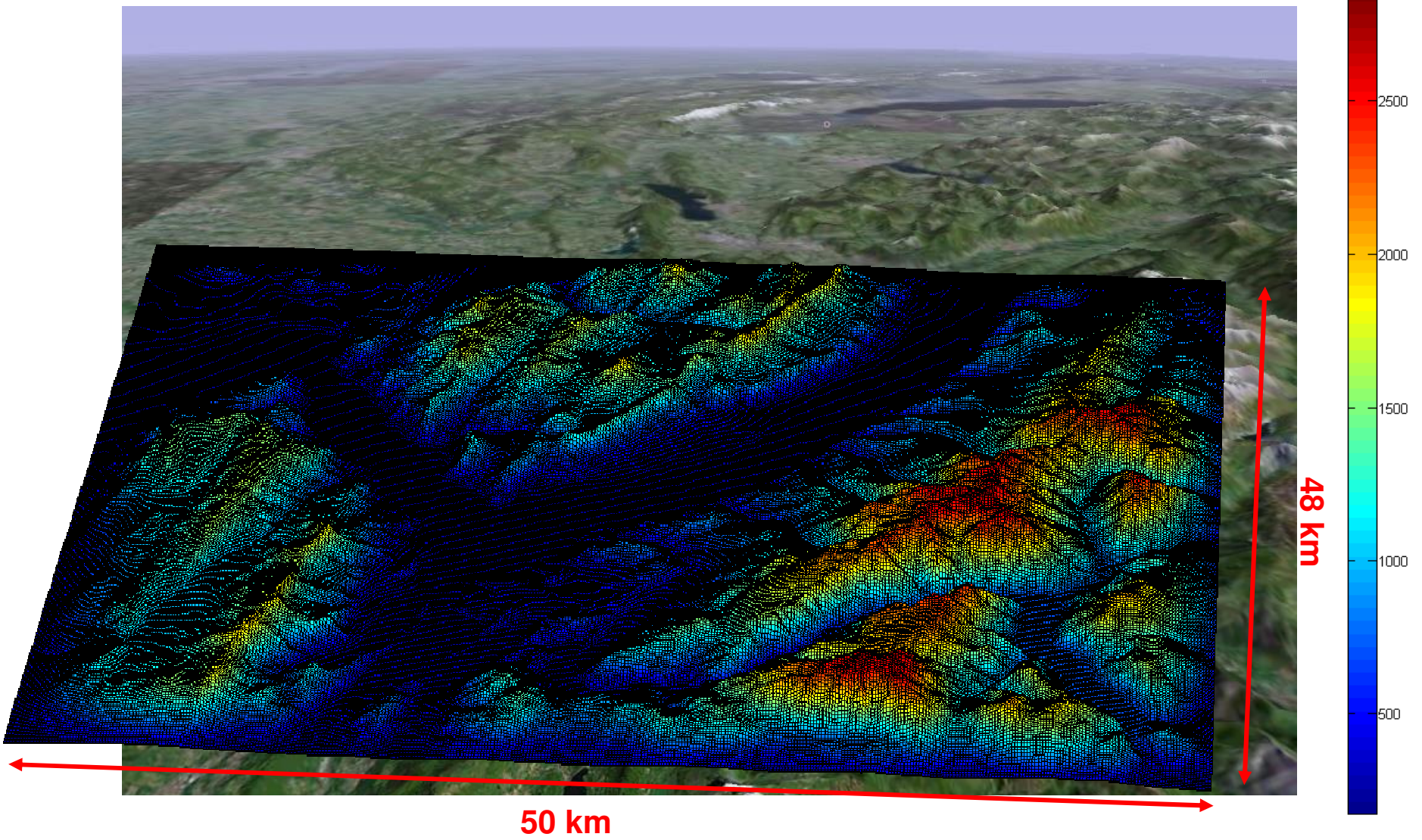
Experience:

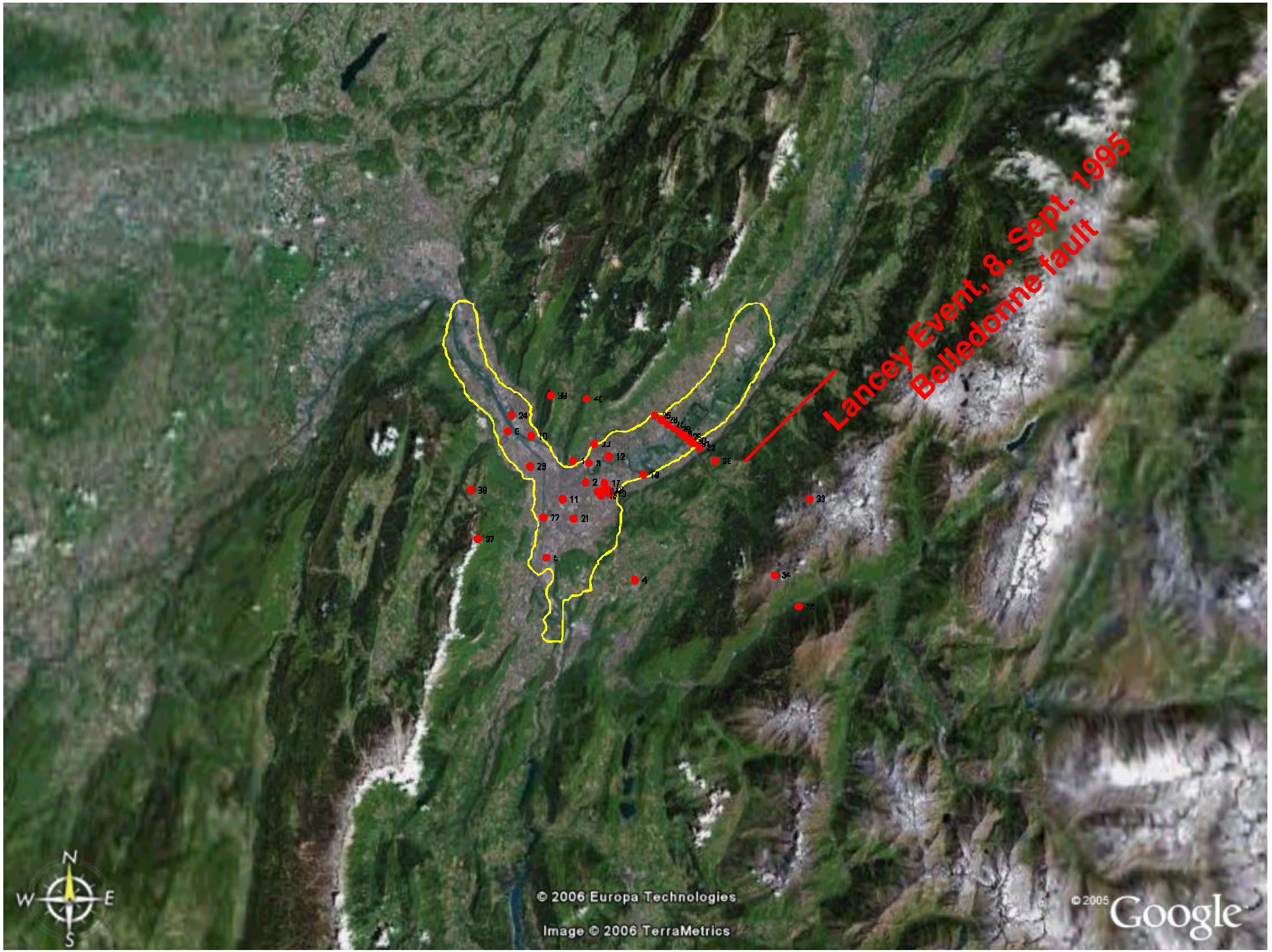
• standard cluster: ca. 5,000-10,000 elements per core with O3

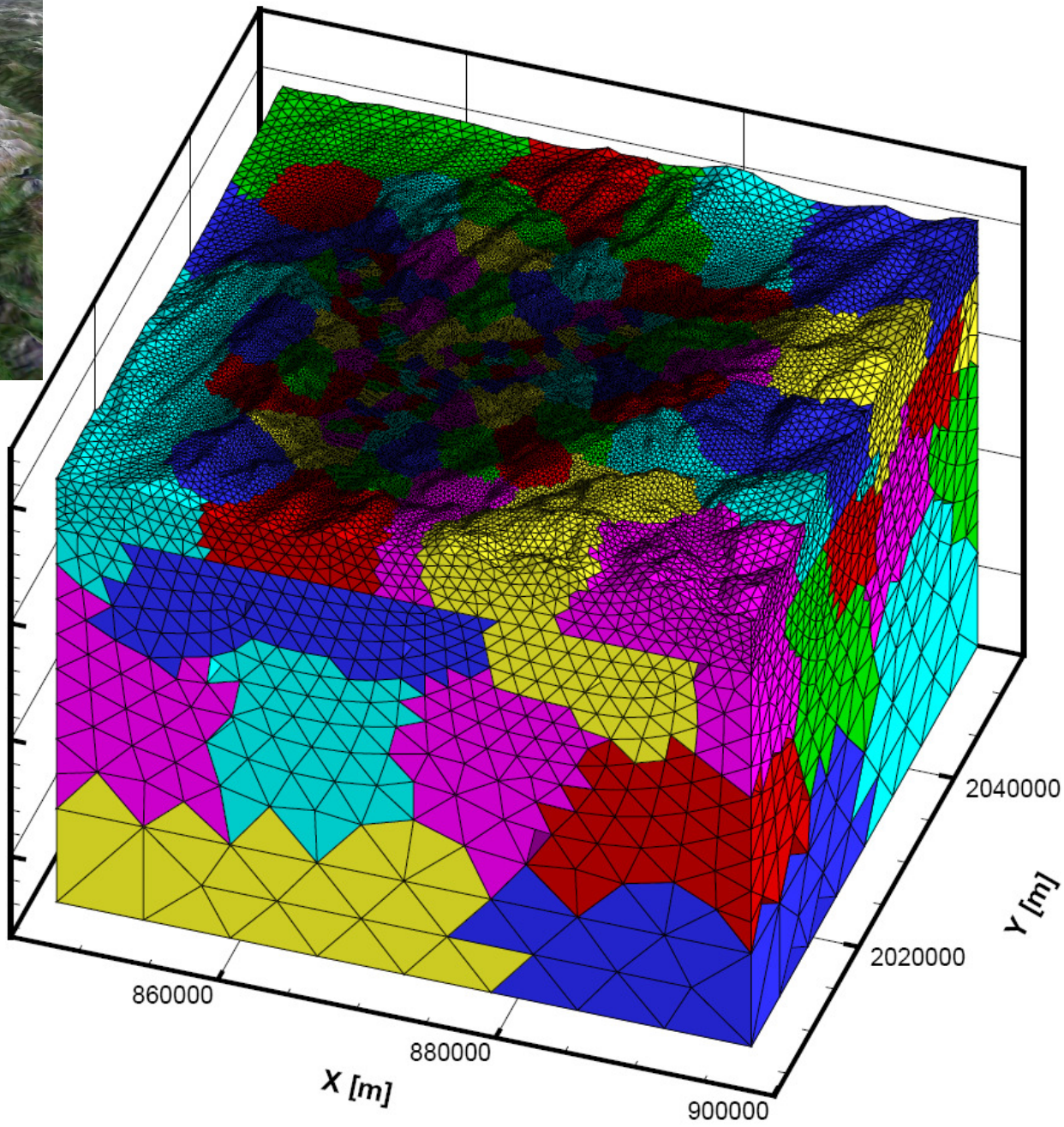
to less than 1,000 elements per core with O7

• IBM BlueGene: a quarter of these numbers

Seismic risk of Grenoble




Lancey Event, 8. Sept. 1995
Belledonne fault
© 2006 Europa Technologies
Image © 2006 TerraMetrics
© 2005 Google



Problem:

Local time stepping

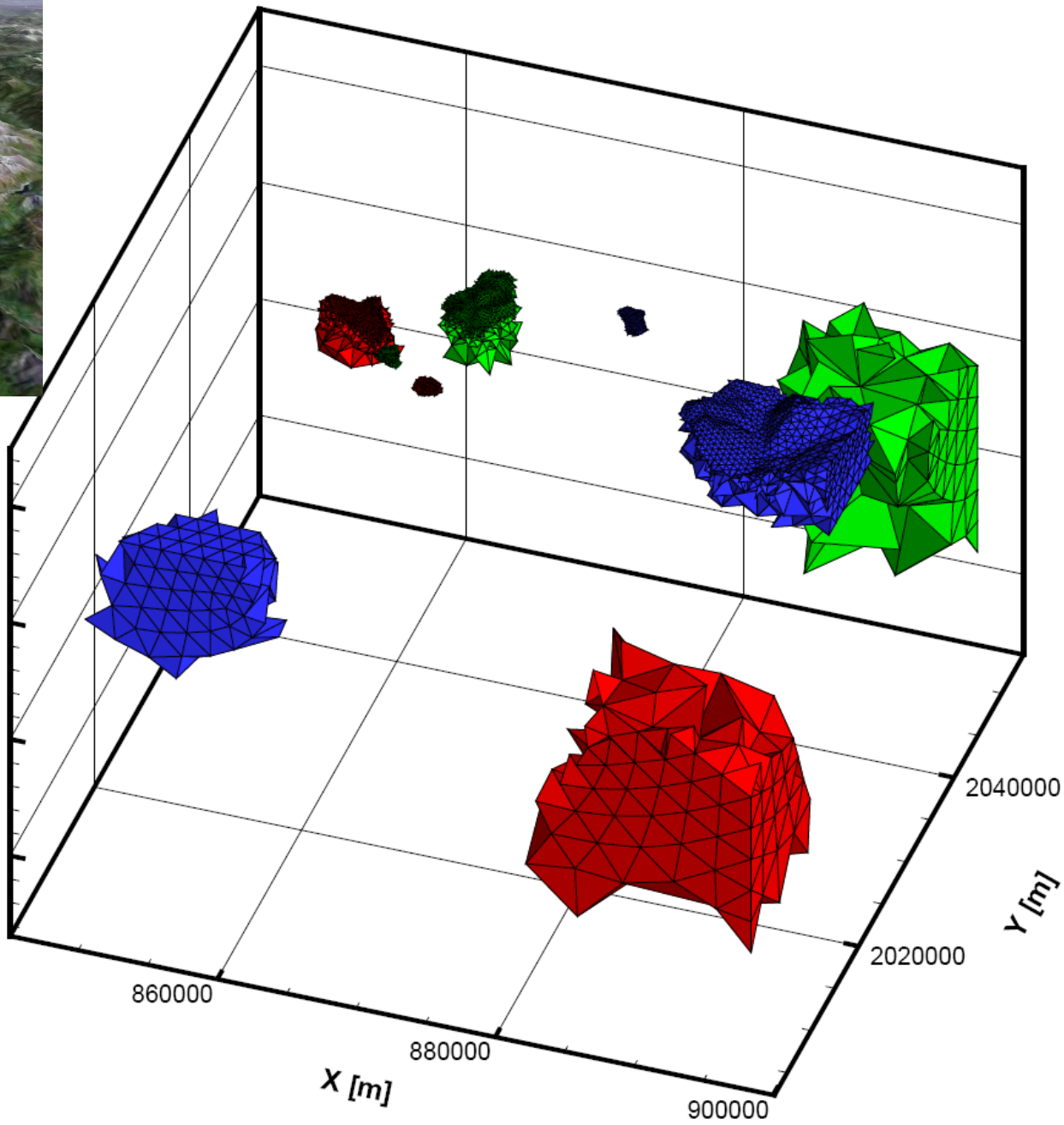
+

Realistic model

➔ computing time varies strongly in space and time

➔ **inhomogeneous load balancing** caused by asynchronous element updates

➔ Group elements (options in matlab scripts available)



The Derivation of the DG Scheme

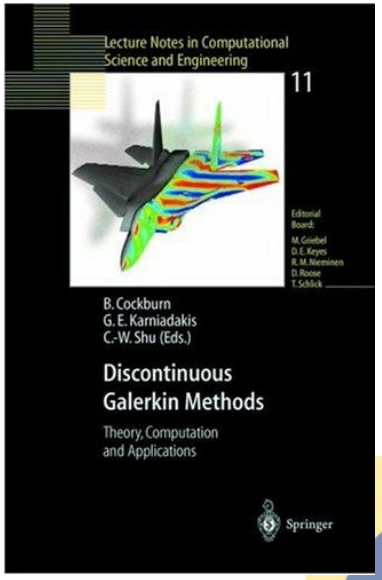
Part II

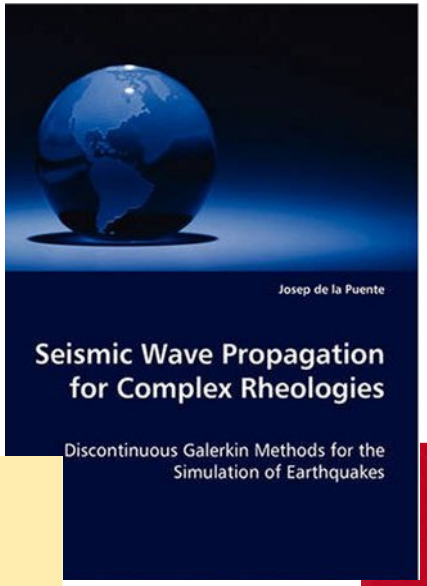
Content of the lecture

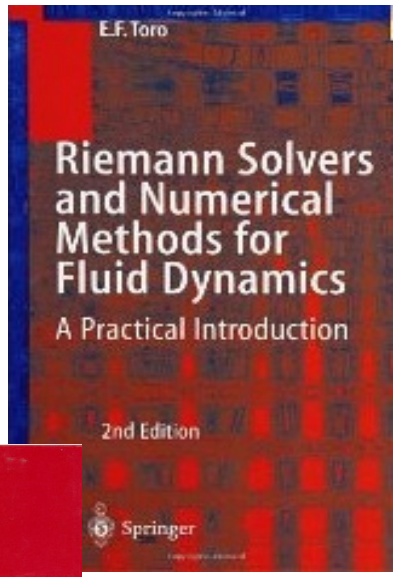
- elastic wave equation
- spatial discretization: grids and meshes
- numerical approximation with polynomials
- Riemann problem & numerical flux
- ADER time integration

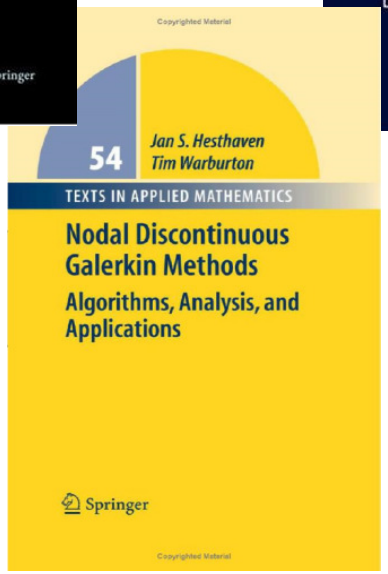
Background to Numerical Scheme:

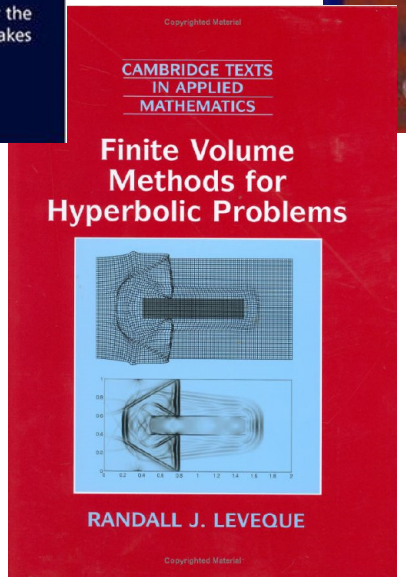
SeisSol is based on the Discontinuous Galerkin (DG) Finite Element Method.
It uses a particular Time Integration approach (ADER) developed by Prof. Toro and V.Titarev.











Elastic wave equation in 2D

System of **5 equations for 5 unknowns** $\sigma_{xx}\,,\sigma_{yy}\,,\sigma_{xy}\,,u\,,v$

$$
\begin{aligned}
\frac{\partial}{\partial t}\sigma_{xx} - (\lambda+2\mu)\frac{\partial}{\partial x}u - \lambda\frac{\partial}{\partial y}v &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{yy} - \lambda\frac{\partial}{\partial x}u - (\lambda+2\mu)\frac{\partial}{\partial y}v &= 0\,,\\
\frac{\partial}{\partial t}\sigma_{xy} - \mu(\frac{\partial}{\partial x}v + \frac{\partial}{\partial y}u) &= 0\,,\\
\rho\frac{\partial}{\partial t}u - \frac{\partial}{\partial x}\sigma_{xx} - \frac{\partial}{\partial y}\sigma_{xy} &= f_x\,,\\
\rho\frac{\partial}{\partial t}v - \frac{\partial}{\partial x}\sigma_{xy} - \frac{\partial}{\partial y}\sigma_{yy} &= f_y\,,
\end{aligned}
$$

Simplification of the elastic wave equation to 1D

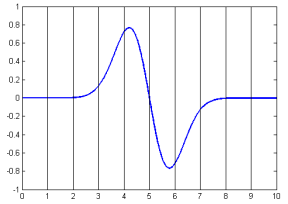
System of **2 equations for 2 unknowns** $\sigma_{xx}\,,\, u$

$$\begin{aligned} \frac{\partial}{\partial t}\sigma_{xx} - (\lambda + 2\mu)\frac{\partial}{\partial x}u &= 0\,, \\ \rho\frac{\partial}{\partial t}u - \frac{\partial}{\partial x}\sigma_{xx} &= f_x\,, \end{aligned}$$

We will start with the simple model equation of **scalar linear advection** in 1-D:

$$q_t + a\, q_x = 0$$

Polynomial Approximation in the DG framework



Inside each **interval I (=element)** of our discretization, we approximate the solution ***q(x,t)*** numerically by using the **reference interval I_R and a linear combination of**

- **time-independent polynomial functions** $\phi_l(\xi)$ in 1D: Legendre Polynomials P_n) as a set of basis functions up to **maximum degree *N***.
- **time-dependent degrees of freedom (DOF)** $\hat{q}_l(t)$

such that we get

$$q(\xi,t) \approx \hat{q}_l(t)\cdot\phi_l(\xi) = \sum_{l=0}^{N} \hat{q}_l(t)\cdot\phi_l(\xi)$$

$$= \hat{q}_0(t)\cdot\phi_0(\xi) + \hat{q}_1(t)\cdot\phi_1(\xi) + \cdots$$

"Einstein summation convention"

Note that the **physical space in I** is denoted by ***x*** whereas the **reference space in the unit interval I_R** is denoted by ξ

Legendre Polynomials – basis functions

Some **important properties of the Legendre Polynomials** are:

➔ defined on the interval **-1 ≤ x ≤ 1**

➔ they are normalized

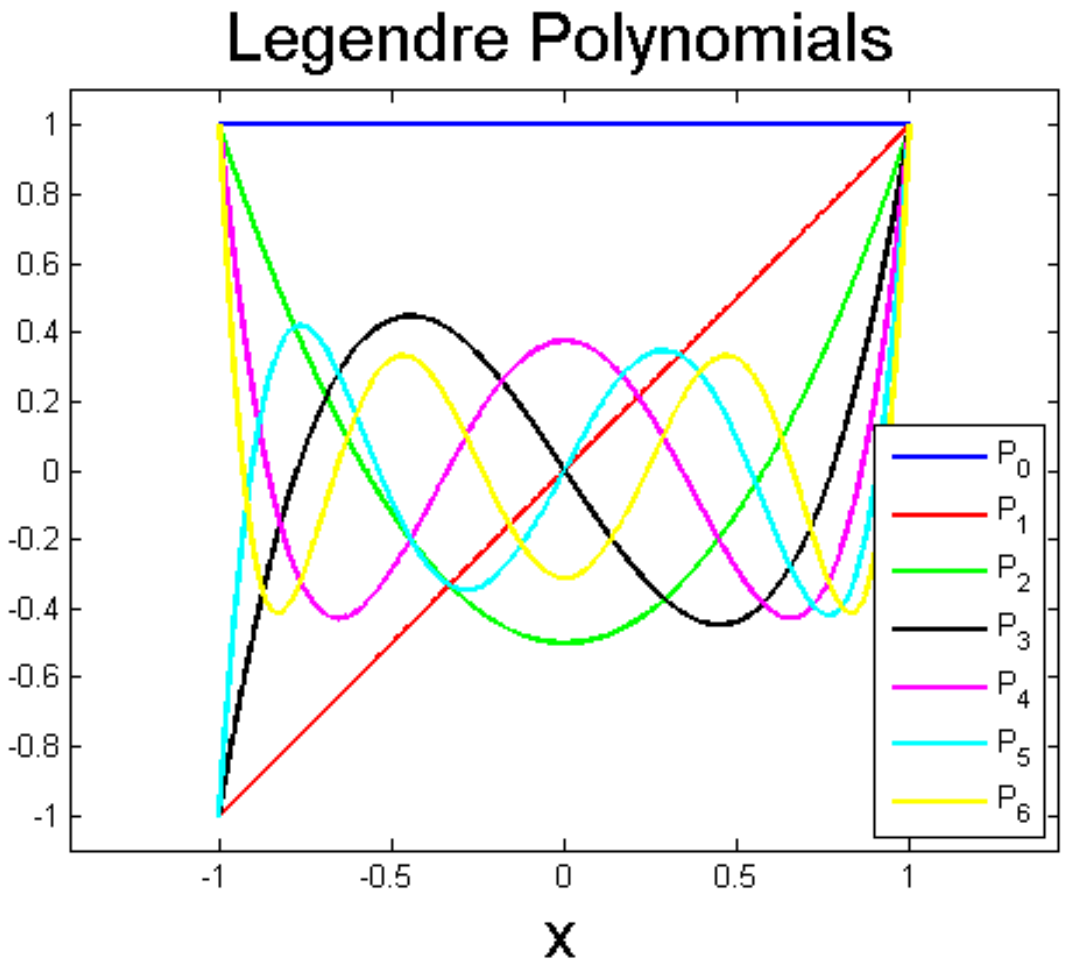
$$P_n(1) = 1$$

➔ Legendre Polynomials are symmetric or antisymmetric:

$$P_n(-x) = (-1)^n P_n(x)$$

➔ they form an orthogonal basis

n	$P_n(x)$
0	1
1	x
2	$\frac{1}{2}(3x^2-1)$
3	$\frac{1}{2}(5x^3-3x)$
4	$\frac{1}{8}(35x^4-30x^2+3)$
5	$\frac{1}{8}(63x^5-70x^3+15x)$
6	$\frac{1}{16}(231x^6-315x^4+105x^2-5)$



The Derivation of the DG Scheme

We will start with the simple model equation of **scalar linear advection** in 1-D:

$$q_t + a\, q_x = 0$$

The DG-method solves the **weak formulation** of the problem, given by

$$\int_t \int_x \phi(q_t + a\, q_x)\, dxdt = 0$$

This is similar to the **integral form of hyperbolic conservations laws** introduced for the finite volume schemes, where smooth function $\phi = 1$.

The Derivation of the DG Scheme

An intelligent choice of the smooth functions ϕ are the **Legendre Polynomials**.

If we integrate over one time step $\Delta t = t^{n+1} - t^n$ and one element (=interval I), we get

$$\int_{t^n}^{t^{n+1}} \int_I \phi_k \frac{\partial q}{\partial t}\, dx\, dt + \int_{t^n}^{t^{n+1}} \int_I \phi_k \left(a \frac{\partial q}{\partial x} \right) dx\, dt = 0$$

Here the index k specifies which basis function ϕ_k, i.e. Legendre Polynomials of degree k, we are using. The range of k, i.e. how many basis functions we are using, depends on the order of **approximation O** and therefore on the **polynomial degree N**.

Question 1: How does the approximation order O depend on the polynomial degree N?

The approximation order O depends on the degree N of the Legendre Polynomials used to approximate the function q(x,t) like **O = N+1**.

The Derivation of the DG Scheme

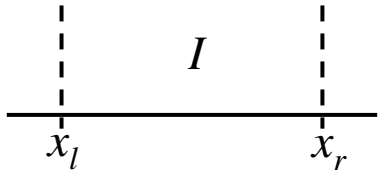
Question:
How does integration by parts work ?

$$\int_a^b f(x)g'(x)\,dx = [f(x)g(x)]_a^b - \int_a^b f'(x)g(x)\,dx$$

Starting from the previous **integral equation**

$$\int_{t^n}^{t^{n+1}} \int_I \phi_k \frac{\partial q}{\partial t}\,dx\,dt + \int_{t^n}^{t^{n+1}} \int_I \phi_k \left(a\frac{\partial q}{\partial x}\right) dx\,dt = 0$$

and **integrating the second term by parts** leads to



$$\boxed{\int_{t^n}^{t^{n+1}} \int_I \phi_k \frac{\partial q}{\partial t}\,dx\,dt} + \boxed{\int_{t^n}^{t^{n+1}} [a\phi_k q]_{x_l}^{x_r}\,dt} - \boxed{\int_{t^n}^{t^{n+1}} \int_I \left(a\frac{\partial \phi_k}{\partial x} q\right) dx\,dt} = 0$$

update term | **flux term** | **stiffness term or volume integral**

In the following we will go through the approximation and detailed form of these three terms one by one!

The Update Term

$$\int_{t^n}^{t^{n+1}} \int_I \phi_k \frac{\partial q}{\partial t}\, dx\, dt$$

First we have to **map the interval I from the physical space** into the **reference interval I_R = [0,1].**

$$\int_{t^n}^{t^{n+1}} \int_I \phi_k \frac{\partial q}{\partial t}\, dx\, dt = \int_{t^n}^{t^{n+1}} \int_{I_R} \phi_k \frac{\partial q}{\partial t} |I|\, d\xi\, dt$$

This transformation includes the multiplication with the size $|I|$ of the interval I.

Now, use the polynomial approximation of q on the reference interval.

$$q(\xi,t) \approx \hat{q}_l(t) \cdot \phi_l(\xi) \quad = \quad \sum_{l=0}^{N} \hat{q}_l(t) \cdot \phi_l(\xi) \quad = \quad \hat{q}_0(t) \cdot \phi_0(\xi) + \hat{q}_1(t) \cdot \phi_1(\xi) + \cdots$$

This finally lead to the update term

$$\int_{t^n}^{t^{n+1}} \int_{I_R} \phi_k \phi_l \frac{\partial \hat{q}_l}{\partial t} |I|\, d\xi\, dt = |I| \int_{I_R} \phi_k \phi_l\, d\xi \int_{t^n}^{t^{n+1}} \frac{\partial \hat{q}_l}{\partial t}\, dt$$

as the basis functions are time-independent and the DOF space-independent !

Question:
Why can we separate the integrals this way?

The Update Term

$$|I| \int_{I_R} \phi_k \phi_l \, d\xi \int_{t^n}^{t^{n+1}} \frac{\partial \hat{q}_l}{\partial t} \, dt$$

The **orthogonality** of the basis functions ϕ leads to the following property of the **space integral over the product of Legendre Polynomials on the reference interval [0,1]:**

$$\int_0^1 \phi_k \phi_l \, d\xi = \frac{1}{2l+1} \delta_{kl}$$

Therefore, we can construct the **mass matrix** $\boldsymbol{M_{kl}}$ for k,l = 0,…,N that has **only entries on its diagonal**

$$M_{kl} = \int_0^1 \phi_k \phi_l \, d\xi$$

Question: **How does the mass matrix for a scheme of order 3 look like?**

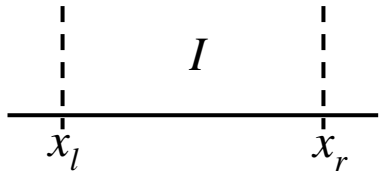
The **mass matrix for a scheme of order 3** is as the polynomial degree **N = 2** ➔ k,l = 0,1,2

$$M_{kl} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{5} \end{pmatrix}$$

The Flux Term

Mapping the interval I from the physical space into the **reference interval I_R = [0,1]** leads to x_r **= 1** and x_l **= 0.**

$$\int_{t^n}^{t^{n+1}} \left[a\phi_k q\right]_{x_l}^{x_r} dt$$



The approximation of q on the reference interval $q(\xi,t) \approx \hat{q}_l(t) \cdot \phi_l(\xi)$ leads to

$$\begin{aligned}
& \int_{t^n}^{t^{n+1}} \left[a\phi_k q\right]_{x_l}^{x_r} dt \\
= & \int_{t^n}^{t^{n+1}} \left[a\phi_k \phi_l \hat{q}_l\right]_{x_l}^{x_r} dt \\
= & \int_{t^n}^{t^{n+1}} \left(a\phi_k(x_r)\phi_l(x_r)\hat{q}_l(x_r) - a\phi_k(x_l)\phi_l(x_l)\hat{q}_l(x_l)\right) dt \\
= & \; a\int_{t^n}^{t^{n+1}} \left(\phi_k(1)\phi_l(1)\hat{q}_l^i - \phi_k(0)\phi_l(1)\hat{q}_l^{i-1}\right) dt \\
= & \; a\left(\phi_k(1)\phi_l(1)\int_{t^n}^{t^{n+1}} \hat{q}_l^i \, dt - \phi_k(0)\phi_l(1)\int_{t^n}^{t^{n+1}} \hat{q}_l^{i-1} \, dt\right)
\end{aligned}$$

The Flux Term

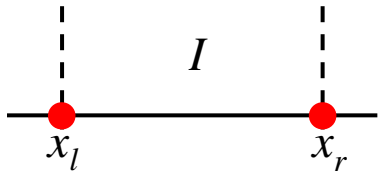
$$a\left(\phi_k(1)\phi_l(1)\int_{t^n}^{t^{n+1}} \hat{q}_l^i\, dt - \phi_k(0)\phi_l(1)\int_{t^n}^{t^{n+1}} \hat{q}_l^{i-1}\, dt\right)$$

Again, only the **DOF are time-dependent** and remain under the time integral.

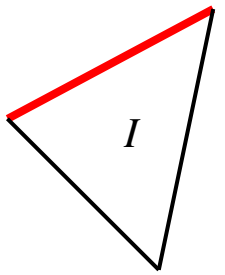
For completeness we construct the **flux matrices** F^R_{kl} and F^L_{kl} for k,l = 0,…,N even if the entries are simple.

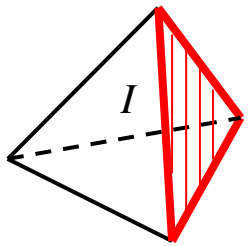
$$F^R_{kl} = \phi_k(1)\phi_l(1)\, d\xi = 1$$

$$F^L_{kl} = \phi_k(0)\phi_l(1)\, d\xi = (-1)^k$$



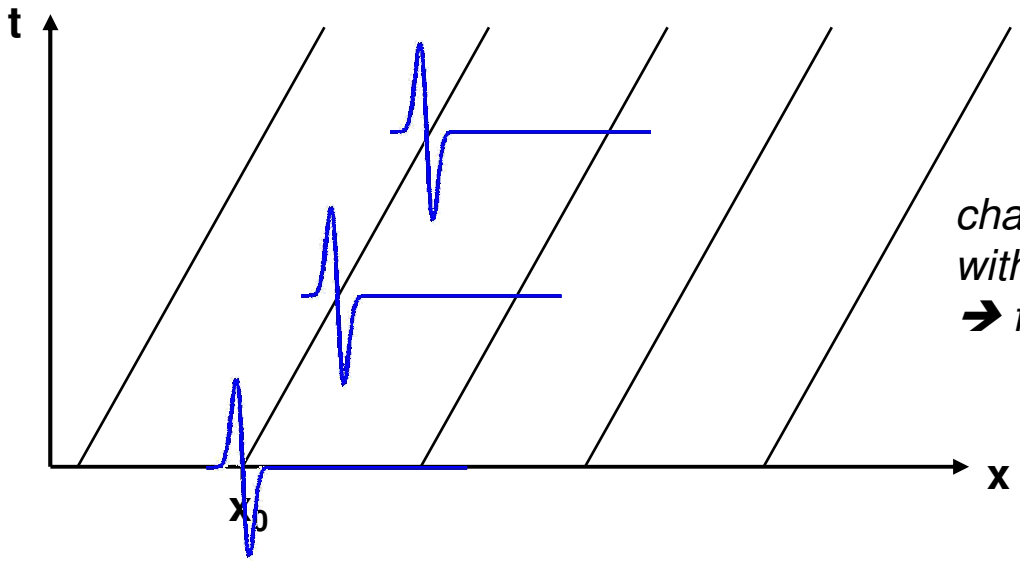
Note: This simple form of the flux matrices will change for multi-dimensional problems, as the boundaries of elements become edges in 2-D or faces in 3-D where the integrals over products of basis functions have to be computed.





General Solution using Characteristics

The **characteristics** or **characteristic curves** of the scalar linear advection equation are **curves in the x-t plane along which *x – at* is constant**.



$$q_t + a\, q_x = 0,$$

*characteristic curves **x = x(t)***
*with slope **1/a***
➔ *for a = const. they are all parallel !*

➔ values of ***q*** simply advect with constant speed ***a = dx / dt***

➔ a particular characteristic passing through x_0 is given by $x(t) = x_0 + at$

➔ the general solution is $$q(x,t) = q_0(x_0) = q_0(x - at),$$

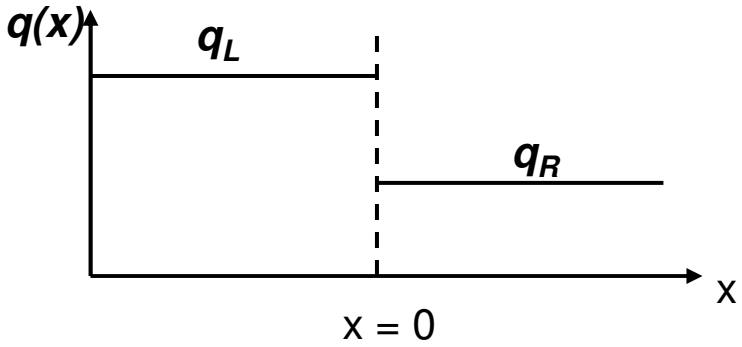
The initial condition translates with speed a, but the shape remains unchanged !

The Riemann Problem – a special initial condition

A **hyperbolic equation together** with a special initial condition, i.e. **piecewise constant values with a single discontinuity** at one point, is called the **Riemann Problem**.

PDE: $q_t + a\, q_x = 0,$

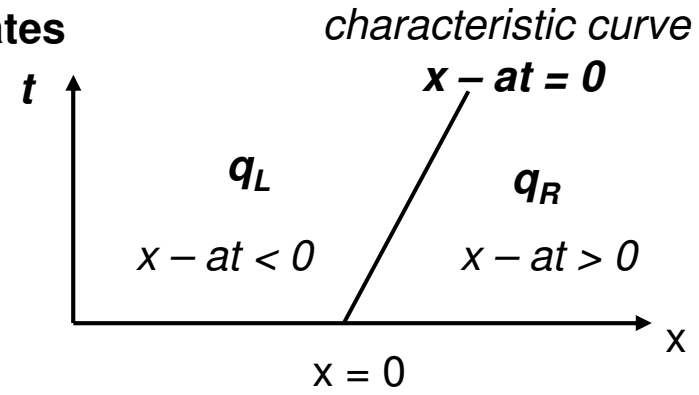
IC: $$q(x,0) = q_0(x) = \begin{cases} q_L & \text{if} \quad x < 0\,, \\ q_R & \text{if} \quad x > 0\,, \end{cases}$$



Using **characteristics** the **initial discontinuity propagates with speed *a*** by a distance ***x = at*** during time ***t***.

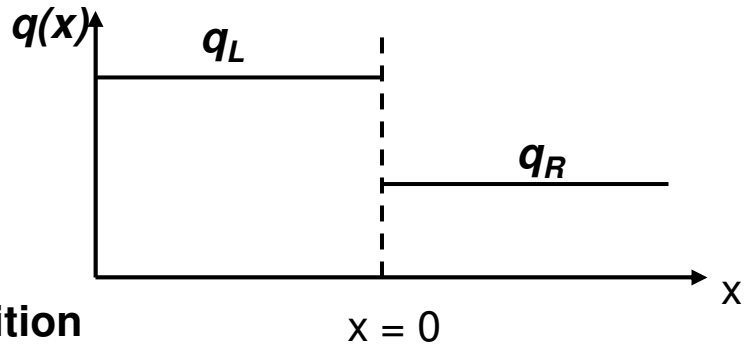
➔ this **characteristic separates the left and right states** and the **solution of the Riemann Problem** is

$$q(x,t) = q_0(x - at) = \begin{cases} q_L & \text{if} \quad x - at < 0\,, \\ q_R & \text{if} \quad x - at > 0\,, \end{cases}$$



➔ **this characteristic curve is the only one across which the solution changes !**

The Riemann Problem



➔ the Riemann Problem is the combination of a **PDE** and a **piecewise constant initial condition**

➔ **solving the Riemann Problem** gives **information** which is used to **compute numerical fluxes** and **update solutions** in time

➔ the **solution of a Riemann Problem** is typically a **similarity solution**, i.e. a function of x/t, and consists of **waves that propagate away from the original discontinuity** with constant wave speeds

➔ the **solution of a Riemann Problem** is easily solved in terms of **eigenvalues** and **eigenvectors** of the **system matrix A**

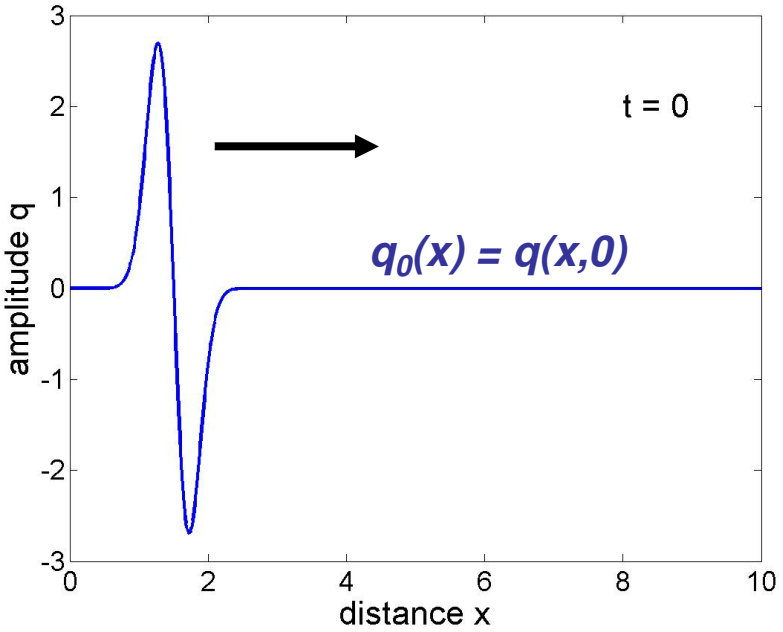
To simulate seismic wave propagation with the DG method, we need
➔ numerical fluxes between neighboring elements
➔ solutions of the Riemann Problem stated by the seismic wave equation

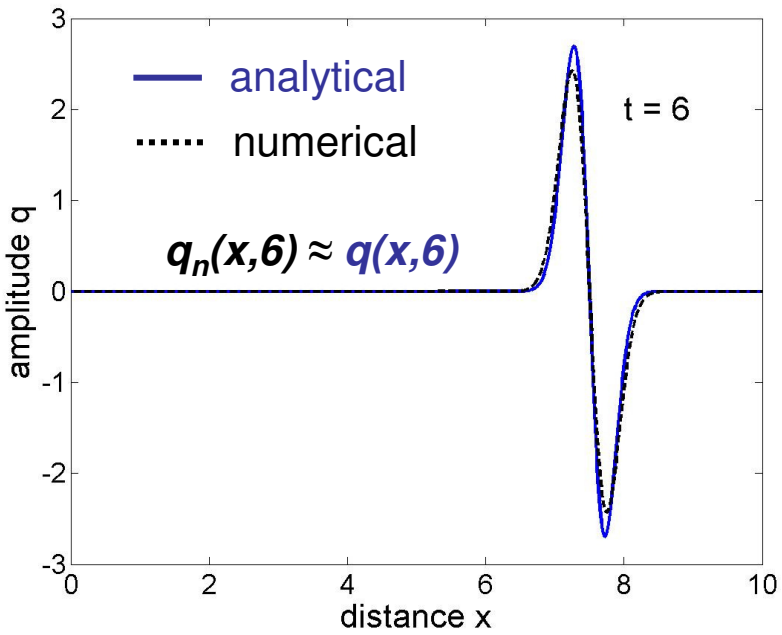
Linear Advection - analytical vs. numerical solutions

the scalar linear advection problem with constant propagation speed is simple
➔ an **analytical solution** q can be computed that **is exact**

for more difficult problems it is much harder or impossible to find such solutions
➔ a **numerical solution** q_n has to be computed that **is only approximate**

The **main goal** is the **development of numerical methods** that are
- **as accurate as possible** while computationally
- **as cheap as possible** !





The Stiffness Term

$$\int_{t^n}^{t^{n+1}} \int_I \left(a \frac{\partial \phi_k}{\partial x} q \right) dx\, dt$$

Mapping the interval I from the physical space into the **reference interval I_R = [0,1]** and considering the resulting change in derivatives $\frac{\partial}{\partial x} = \frac{1}{|I|} \frac{\partial}{\partial \xi}$ gives

$$\int_{t^n}^{t^{n+1}} \int_I \left(a \frac{\partial \phi_k}{\partial x} q \right) dx\, dt = \int_{t^n}^{t^{n+1}} \int_{I_R} \left(a \frac{1}{|I|} \frac{\partial \phi_k}{\partial \xi} q \right) |I|\, d\xi\, dt$$

The approximation of q on the reference interval $q(\xi, t) \approx \hat{q}_l(t) \cdot \phi_l(\xi)$ leads to

$$\int_{t^n}^{t^{n+1}} \int_{I_R} \left(a \frac{1}{|I|} \frac{\partial \phi_k}{\partial \xi} \phi_l \hat{q}_l \right) |I|\, d\xi\, dt = a \int_{I_R} \frac{\partial \phi_k}{\partial \xi} \phi_l\, d\xi \int_{t^n}^{t^{n+1}} \hat{q}_l\, dt$$

Question: Why can we separate the integrals here?

as again, only the **DOF are time-dependent** and remain under the time integral, while the **space integral over the product of basis functions with their space-derivatives** can be separated.

The Stiffness Term

Now, the **integral over the product of basis functions with their space derivatives** has to be computed explicitly.

$$a \int_{I_R} \frac{\partial \phi_k}{\partial \xi} \phi_l \, d\xi \int_{t^n}^{t^{n+1}} \hat{q}_l \, dt$$

Question: How would you solve this space integral?

➔ The space integral can be pre-calculated analytically or computed numerically using a Gaussian integration rule that is exact for the required polynomial degree obtained through the product in the integrand.

The Stiffness Term

$$a \int_{I_R} \frac{\partial \phi_k}{\partial \xi} \phi_l \, d\xi \int_{t^n}^{t^{n+1}} \hat{q}_l \, dt$$

The **stiffness matrix** K_{kl} for k,l = 0,…,N is given as

$$K_{kl} = \int_0^1 \frac{\partial \phi_k}{\partial \xi} \phi_l \, d\xi$$

and is **very sparse**, i.e. only a few entries are non-zero. This **property also remains for the multi-dimensional case**.

Question: **How do the entries of the first row for k=0 look like?**

The **stiffness matrix for a scheme of order 3** is as the polynomial degree **N = 2** ➔ k,l = 0,1,2

$$K_{kl} = \begin{pmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ 0 & 2 & 0 \end{pmatrix}$$

The DG-Scheme

Putting together **all three terms** again leads to the following formulation

$$|I|M_{kl} \int_{t^n}^{t^{n+1}} \frac{\partial \hat{q}_l}{\partial t}\, dt + aF^R_{kl} \int_{t^n}^{t^{n+1}} \hat{q}^i_l\, dt - aF^L_{kl} \int_{t^n}^{t^{n+1}} \hat{q}^{i-1}_l\, dt - aK_{kl} \int_{t^n}^{t^{n+1}} \hat{q}_l\, dt = 0$$

update term | **flux terms** | **stiffness term or volume integral**

Each term includes a **time integral over one time step $\Delta t = t^{n+1} - t^n$.**

In order to **obtain a numerical scheme** that provides the **same accuracy in time as in space** we need to approximate these time integrals in the flux and stiffness terms with the same accuracy as the chosen space accuracy.

➔ One choice is using a **Runge-Kutta** scheme for accurate time integration.

➔ Our choice is following the **ADER-approach** using high-order derivatives

The ADER Time Integration

The basis of high-order time integration is the **Taylor Series Expansion** in time!

$$q(x,t^{n+1}) = q(x,t^n + \Delta t) \; = \sum_{j=0}^{N} \frac{\Delta t^j}{j!} \frac{\partial^j}{\partial t^j} q(x,t^n)$$

First ingredient

The **ADER time integration** method follows the technique of Lax-Wendroff, where the **time derivative is replaced by the space derivative**, by using the governing PDE, i.e.

$$\frac{\partial}{\partial t} q(x,t) = -a \frac{\partial}{\partial x} q(x,t)$$

This can be extended to a **recursive formula for higher-order derivatives**.

$$\frac{\partial^{j+1}}{\partial t^{j+1}} q(x,t) = -a \frac{\partial}{\partial x} \left(\frac{\partial^j}{\partial t^j} q(x,t) \right)$$

This approach is often called the **Cauchy-Kowalewski Procedure** !

Second ingredient

$$q(\xi,t) \approx \hat{q}_l(t) \cdot \phi_l(\xi)$$

The Cauchy-Kowalewski Procedure

If we now formulate the **Taylor series expansion for the degrees of freedom** we get

$$\hat{q}_l(t^{n+1}) = \sum_{j=0}^{N} \frac{\Delta t^j}{j!} \frac{\partial^j}{\partial t^j} \hat{q}_l(t^n)$$

and finally, for the **time integration of the degrees of freedom** we have

$$\int_{t^n}^{t^{n+1}} \hat{q}_l(t)\, dt = \sum_{j=0}^{N} \frac{\Delta t^{j+1}}{(j+1)!} \frac{\partial^j}{\partial t^j} \hat{q}_l(t^n)$$

with

$$\frac{\partial^{j+1}}{\partial t^{j+1}} \hat{q}_l(t^n) = -\frac{a}{|I|} \frac{\int_{I_R} \phi_k \frac{\partial \phi_l}{\partial \xi}\, d\xi}{\int_{I_R} \phi_k \phi_l\, d\xi} \frac{\partial^j}{\partial t^j} \hat{q}_l(t^n)$$

So we can determine the required time integrals with the spatial order of accuracy !

$$|I| M_{kl} \int_{t^n}^{t^{n+1}} \frac{\partial \hat{q}_l}{\partial t}\, dt + a F^R_{kl} \int_{t^n}^{t^{n+1}} \hat{q}_l^i\, dt - a F^L_{kl} \int_{t^n}^{t^{n+1}} \hat{q}_l^{i-1}\, dt - a K_{kl} \int_{t^n}^{t^{n+1}} \hat{q}_l\, dt = 0$$

The Fully Discrete ADER-DG Scheme

We now have to find an **update formula of our degrees of freedom from time level t^n to t^{n+1}** using the formulation

$$|I|M_{kl}\int_{t^n}^{t^{n+1}}\frac{\partial \hat{q}_l}{\partial t}\,dt + aF_{kl}^R\int_{t^n}^{t^{n+1}}\hat{q}_l^i\,dt - aF_{kl}^L\int_{t^n}^{t^{n+1}}\hat{q}_l^{i-1}\,dt - aK_{kl}\int_{t^n}^{t^{n+1}}\hat{q}_l\,dt = 0$$

Thus we carry out the **integration of the update term** leading to

$$|I|M_{kl}\int_{t^n}^{t^{n+1}}\frac{\partial \hat{q}_l}{\partial t}\,dt = |I|M_{kl}\left(\hat{q}_l(t^{n+1}) - \hat{q}_l(t^n)\right)$$

If we solve the equation now for the **degrees of freedom at time level t^{n+1}** we get the **fully discrete update formula of the ADER-DG scheme** for element ***i***

$$\hat{q}_l^i(t^{n+1}) = \hat{q}_l^i(t^n) - \left(aF_{kl}^R\int_{t^n}^{t^{n+1}}\hat{q}_l^i(t^n)\,dt - aF_{kl}^L\int_{t^n}^{t^{n+1}}\hat{q}_l^{i-1}(t^n)\,dt - aK_{kl}\int_{t^n}^{t^{n+1}}\hat{q}_l^i(t^n)\,dt\right)/|I|M_{kl}$$

The order is determined by the number of degrees of freedom (the range of k,l)!

Matrix-vector formulation of the system in 2D

The system of the isotropic, elastic wave equation can be written as

$$\frac{\partial Q}{\partial t} + A\frac{\partial Q}{\partial x} + B\frac{\partial Q}{\partial y} = S,$$

with the **vector Q of unknowns**, the **Jacobian matrices A and B** and the **source term S**

$$A = \begin{pmatrix} 0 & 0 & 0 & -(\lambda+2\mu) & 0 \\ 0 & 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & 0 & -\mu \\ -\frac{1}{\rho} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{\rho} & 0 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 & 0 & 0 & 0 & -\lambda \\ 0 & 0 & 0 & 0 & -(\lambda+2\mu) \\ 0 & 0 & 0 & -\mu & 0 \\ 0 & 0 & -\frac{1}{\rho} & 0 & 0 \\ 0 & -\frac{1}{\rho} & 0 & 0 & 0 \end{pmatrix}$$

$$Q = \begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \\ u \\ v \end{pmatrix}, \quad S = \begin{pmatrix} 0 \\ 0 \\ 0 \\ f_x/\rho \\ f_y/\rho \end{pmatrix}$$

The Cauchy-Kowalewski Procedure

If we apply this methodology on our linear advection equation **in the reference space**, we get

$$q(\xi, t^{n+1}) = \sum_{j=0}^{N} \frac{\Delta t^j}{j!} \frac{\partial^j}{\partial t^j} q(\xi, t^n)$$

with

$$\frac{\partial^{j+1}}{\partial t^{j+1}} q(\xi, t) = -\frac{a}{|I|} \frac{\partial}{\partial \xi} \left(\frac{\partial^j}{\partial t^j} q(\xi, t) \right)$$

In weak formulation and expressing q by $q(\xi, t) \approx \hat{q}_l(t) \cdot \phi_l(\xi)$ leads to

$$\int_{I_R} \phi_k \frac{\partial^{j+1}}{\partial t^{j+1}} \phi_l \hat{q}_l \, d\xi = \int_{I_R} \phi_k \left(-\frac{a}{|I|} \frac{\partial}{\partial \xi} \frac{\partial^j}{\partial t^j} \phi_l \hat{q}_l \right) d\xi$$

$$\int_{I_R} \phi_k \phi_l \, d\xi \, \frac{\partial^{j+1}}{\partial t^{j+1}} \hat{q}_l = -\frac{a}{|I|} \int_{I_R} \phi_k \frac{\partial \phi_l}{\partial \xi} \, d\xi \, \frac{\partial^j}{\partial t^j} \hat{q}_l$$

$$\frac{\partial^{j+1}}{\partial t^{j+1}} \hat{q}_l = -\frac{a}{|I|} \frac{\int_{I_R} \phi_k \frac{\partial \phi_l}{\partial \xi} \, d\xi}{\int_{I_R} \phi_k \phi_l \, d\xi} \frac{\partial^j}{\partial t^j} \hat{q}_l$$

Calculation of the Degrees of feedom (DOF)

Now we have everything to **compute the *N+1* degrees of freedom** given as

$$\hat{q}_l(t) \quad = \quad \frac{\int_{I_R} q(\xi,t)\cdot\phi_l(\xi)\,d\xi}{\int_{I_R} \phi_l(\xi)\cdot\phi_l(\xi)\,d\xi}$$

Using the **orthogonality of the basis functions ϕ** gives

$$\hat{q}_l(t) \quad = \quad \frac{\int_{I_R} q(\xi,t)\cdot\phi_l(\xi)\,d\xi}{\frac{1}{2l+1}}$$

Applying **Gauss-Legendre numerical integration over discrete integration points ξ_i** finally leads to

$$\hat{q}_l(t) \quad = \quad (2l+1)\sum_{i=1}^{N+1} w_i\cdot q(\xi_i,t)\cdot\phi_l(\xi_i)$$

for $l = 0,\ldots,N$