Coordinate free numerics

Andrzej Pownuk
Department of Mathematical Science
University of Texas at El Paso
http://andrzej.pownuk.com

September 8, 2006

The physical significance of tensors

Laws of physics do not depend on our choice of coordinate system. In particular, if a law of physics is true in one coordinate system then it is automatically true in every other coordinate system.

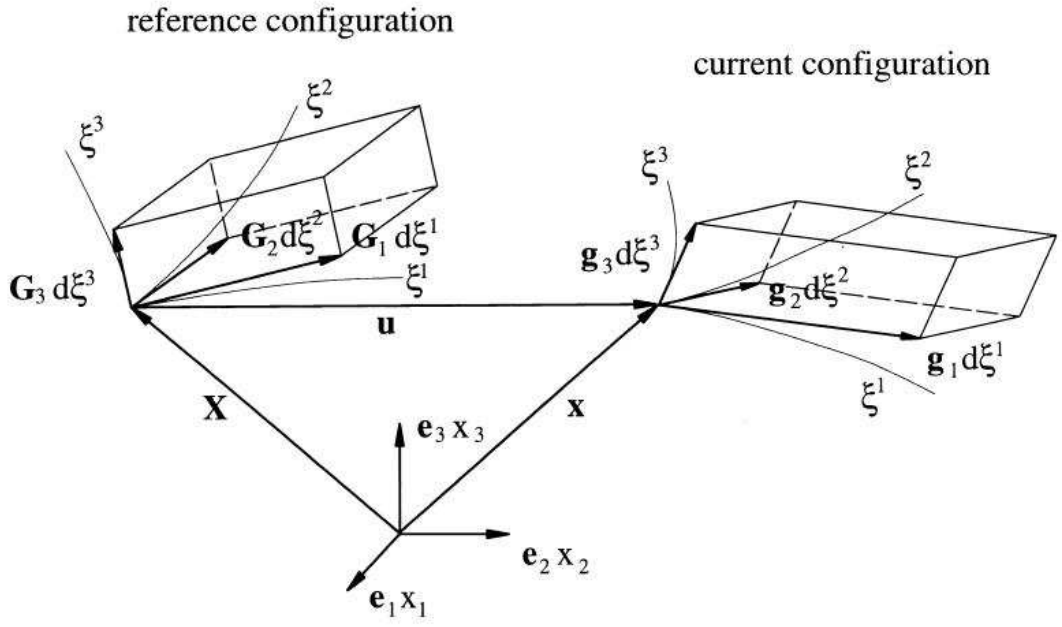

Figure: Curvilinear coordinates and convective base vectors

Definition of coordinate free formulation

Definition

The problem is **coordinate free** (independent of coordinate system) if it need not be changed when moving from one coordinate system to another.

In other words the formulation is coordinate free if it is true in all possible coordinate systems.

Is it possible to define a vector without coordinate system?

Methods of solutions of problems with vector parameters:
-analytical methods.
-graphical methods.

Second Newton's low of dynamics in curvilinear coordinate system

The vector form of Second Newton's low of dynamics is the following:

$$m\mathbf{a} = \mathbf{f} \tag{1}$$

where $\mathbf{a}$ is an acceleration and $\mathbf{f}$ is a force. Let assume that we have general curvilinear coordinate system (x_i). Then the natural basis vector of TM_x space in $\mathbb{R}^3$ is given by the following vectors:

$$\mathbf{g}_i = \frac{\partial \mathbf{r}}{\partial x_i} \tag{2}$$

All vectors from the equation (1) can be expressed as a linear combination of basis vectors (2).

$$\begin{aligned} \mathbf{a} &= \sum_i a^i \mathbf{g}_i \\ \mathbf{f} &= \sum_i f^i \mathbf{g}_i \end{aligned} \tag{3}$$

The numbers a^i, f^i are contravariant coordinates of the vectors **a** and **f**. The velocity can be defined as a time derivative of the vector **r**.

$$\mathbf{v} = \frac{d\mathbf{r}}{dt} = \sum_i \frac{\partial \mathbf{r}}{\partial x_i} \frac{dx_i}{dt} = \sum_i \mathbf{g}_i v^i = \sum_i \mathbf{g}_i \dot{x}_i \tag{4}$$

where $\mathbf{g}_i = \frac{\partial \mathbf{r}}{\partial x_i}$ and $v^i = \frac{dx_i}{dt} = \dot{x}_i$. Acceleration can be defined using second order time derivative.

$$a^i = \ddot{x}_i + \sum_j \sum_k \Gamma^i_{jk} \dot{x}_j \dot{x}_k \tag{5}$$

Now it is possible to write the equation (2) in arbitrary coordinate system.

$$m \left(\ddot{x}_i + \sum_j \sum_k \Gamma^i_{jk} \dot{x}_j \dot{x}_k \right) = f^i \tag{6}$$

Now we can formulate the initial value problem in a coordinate free way

$$\begin{cases} m\mathbf{a} = \mathbf{f} \\ \mathbf{r}(0) = \mathbf{r}_0 \\ \mathbf{v}(0) = \mathbf{v}_0 \end{cases} \tag{7}$$

or equivalently in general curvilinear coordinate system.

$$\begin{cases} m\left(\ddot{x}_i + \sum_j \sum_k \Gamma^i_{jk} \dot{x}_j \dot{x}_k\right) = f^i \\ r^i(0) = r^i_0 \\ v^i(0) = v^i_0 \end{cases} \tag{8}$$

SAGA - Scientific Computing with Algebraic and Generative Abstractions

Supervisor :**Magne Haveraaen**
Department of Informatics
University of Bergen
Hyteknologisenteret
N-5020 BERGEN, Norway
WWW: **http://www.ii.uib.no/saga/**

The project supported by:

- The Research Council of Norway (NFR)
- European Union (EU) through ESPRIT-IV Long Term Research

Figure: University of Bergen

- Algebraic structures
  - Many Sorted Algebras
  - The concept of the signature
  - Theory, models, etc.
  - Different logics
  - ...
  - Category theory

Algebraic Software Methodologies for PDEs

- using algebraic techniques for software structuring purposes gives more mature software structures which are less prone to redesigns than conventionally developed systems, providing a general framework for solvers of most kinds of PDEs, with
- high degree of reuse even within the library itself, so that advanced concepts are defined by combinations of basic modules, which provides the ability that
- numerical solution strategy may be replaced with another just by substituting a few modules and without changing the system structure, prototype code development times are reduced since central numerical modules are reused rather than readapted, and
- only a few minor modules need to be replaced when porting to and between HPC architectures.

Industrial Case Studies

Two industrial areas have been chosen for the practical tests of the SAGA methodologies:

- seismic modelling,
- computational fluid dynamics.

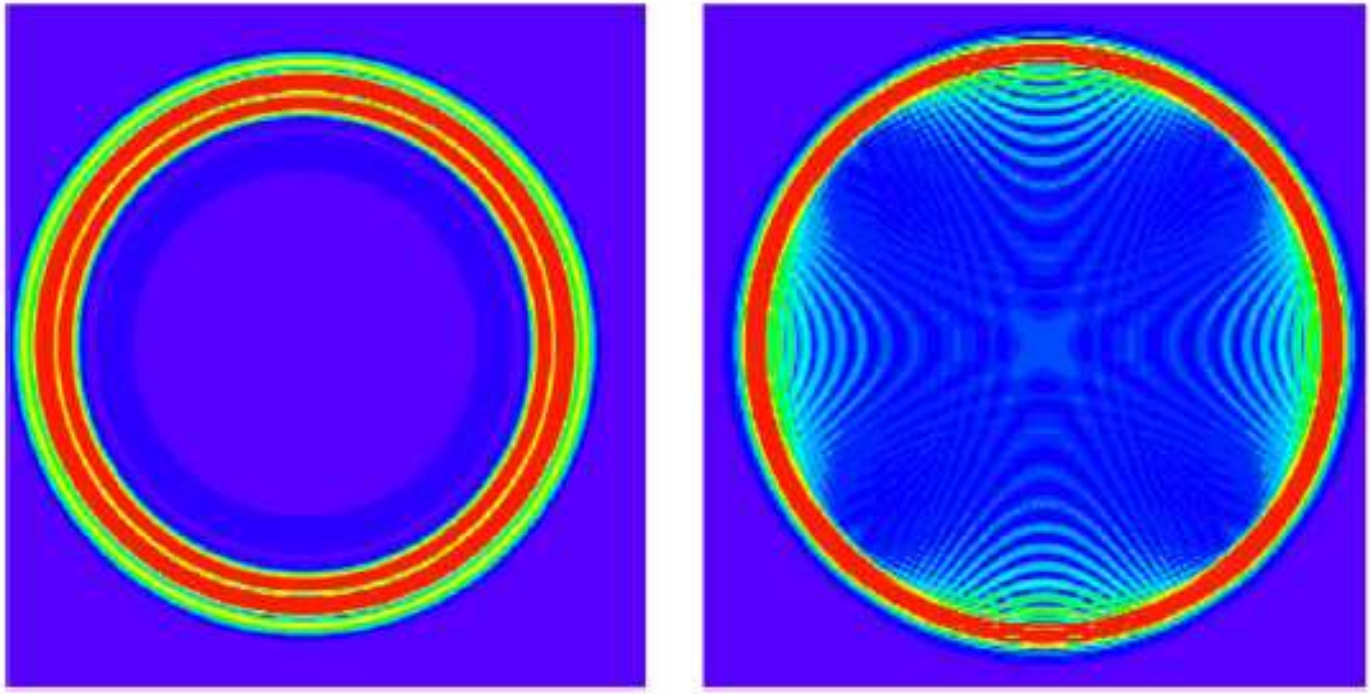

Fig. 1. In the left snapshot the source has frequency 30 Hz, in the right 100 Hz. Numerical dispersion is clearly visible inside the wavefront in the right snapshot. Also notice that the 100 Hz wavefront defines a slightly larger circle than the 30 Hz wavefront. This is also a numerical dispersion phenomenon. Homogeneous medium with 1×1 km grid and 5 m resolution. Snapshots are taken after 175 ms (0.5 ms resolution). Wave source of type Ricker (zero) is placed at the centre.

Figure: Solution of wave equation

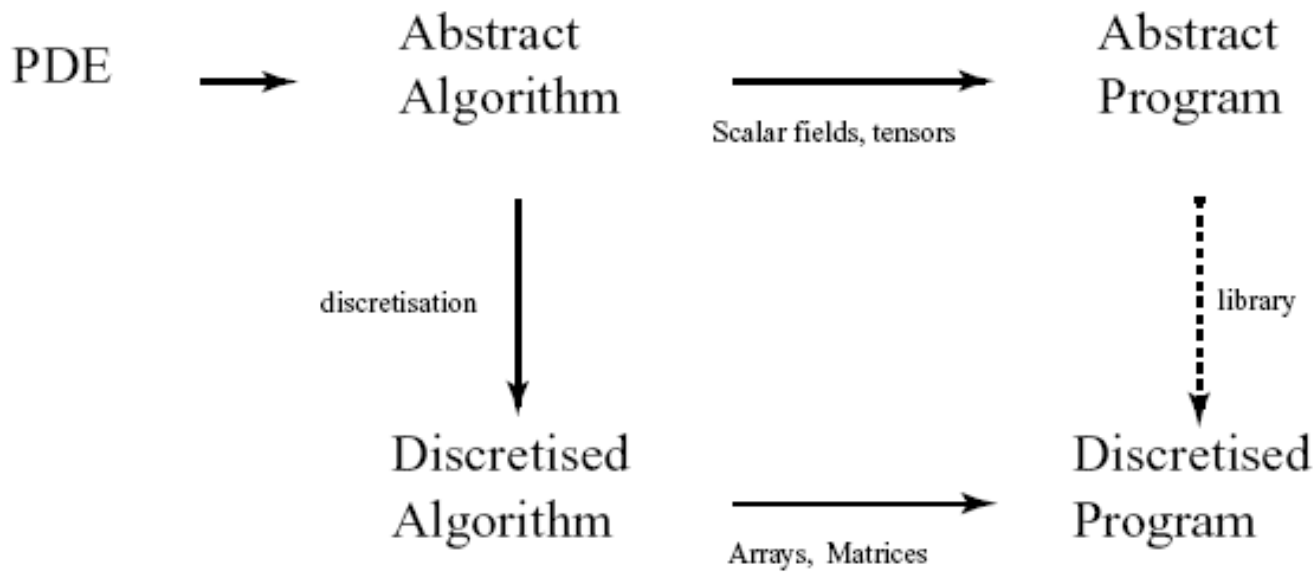

Fig. 2. Coordinate free versus conventional methodology.

$$\rho \frac{\partial^2 \vec{u}}{\partial t^2} = \nabla \cdot \boldsymbol{\sigma} + \vec{f}(t),$$

$$\boldsymbol{\sigma} = \Lambda(e),$$

$$e = \mathcal{L}_u(g).$$

```
void advance(
  DiffTensor<ScalarField> &un,        // u at timestep n
  DiffTensor<ScalarField> &unm1,      // u at timestep n-1
  const DiffTensor<ScalarField> &g,   // coordinate system
  const DiffTensor<ScalarField> &Lambda,
  const DiffTensor<ScalarField> &rho,// seismic properties
  const int &stoptime)                // stoptime
{
  DiffTensor<ScalarField> unp1;        // u at timestep n+1
  DiffTensor<ScalarField> e, sigma, a;// intermediate values
  for (int t=0; t < stoptime; t++) {
    e = Lie(un,g);
    sigma = apply(Lambda,e);
    a = div(sigma)/rho + fsource(t);
    unp1 = 2*un-unm1+a*(dt*dt);
    unm1 = un;
    un = unp1;
  }
}
```

$$\rho_{11}\frac{\partial^2 \boldsymbol{u}}{\partial t^2}+\rho_{12}\frac{\partial^2 \boldsymbol{U}}{\partial t^2}=\nabla\cdot\sigma+(1-q)f(t)+b@\left(\frac{\partial \boldsymbol{U}}{\partial t}-\frac{\partial \boldsymbol{u}}{\partial t}\right),$$

$$\rho_{12}\frac{\partial^2 \boldsymbol{u}}{\partial t^2}+\rho_{22}\frac{\partial^2 \boldsymbol{U}}{\partial t^2}=\nabla s\cdot q f(t)-b@\left(\frac{\partial \boldsymbol{U}}{\partial t}-\frac{\partial \boldsymbol{u}}{\partial t}\right),$$

$$\sigma=\Lambda@e+Q\otimes\tau,$$

$$s=Q@e+R\otimes\tau,$$

$$e=\mathcal{L}_{\boldsymbol{u}}(g),\quad \tau=\nabla\cdot\boldsymbol{U}.$$

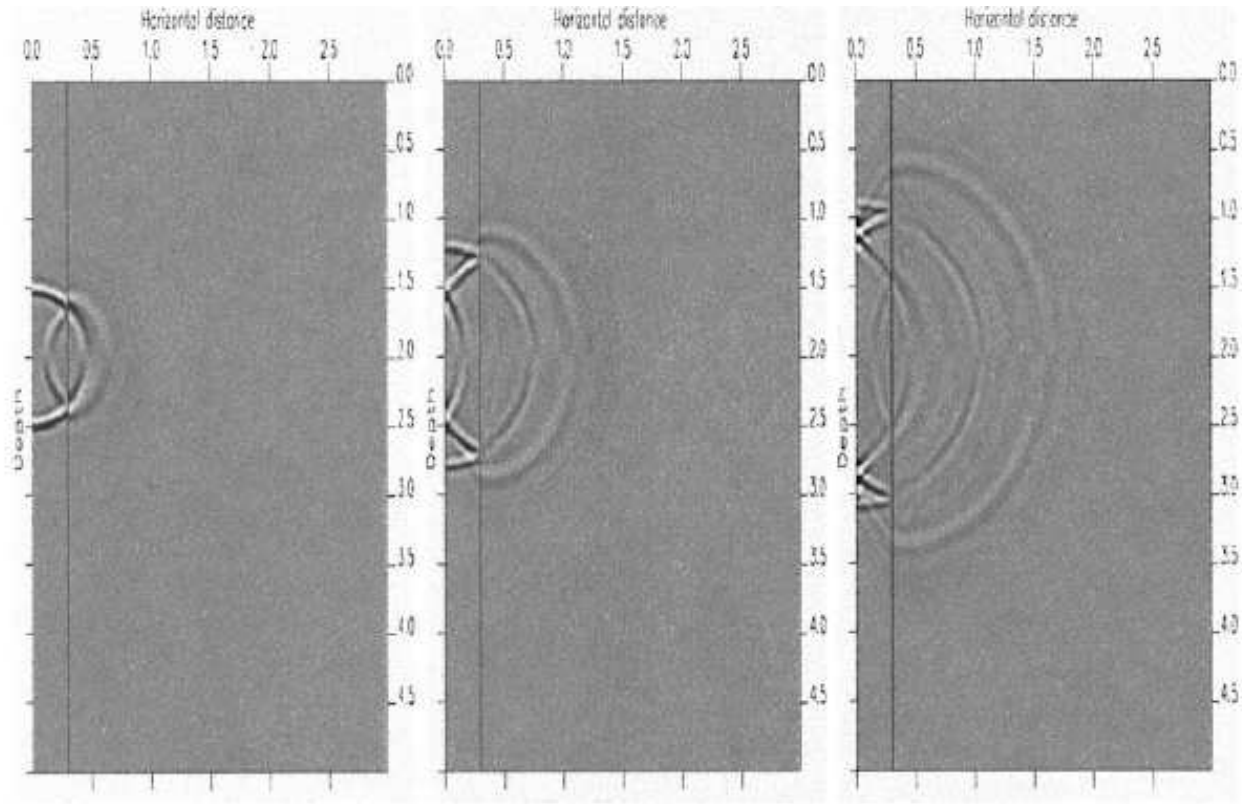

Fig. 3. Snapshots of wave propagation in borehole model at $t = 0.4$, 0.6 and 0.8 msec, radial (u_r) component (top) and vertical (u_z) component (bottom). Horizontal and vertical axis are in meters. The borehole wall is located at 0.30 meters and the source is located at the borehole center, vertically 2.0 m below the top of the model. P–P and P–S are the transmitted P and S waves, whereas Rp and Rs denote the refracted P and S waves, respectively.

SAGA Technologies

Sapphire: For the quick prototyping of mathematical models we have developed an algebraic programming language and a compiler that translates recursive functions into non-recursive, imperative code. This allows us to code the recursive equations of the mathematical formulation of a solver directly as recursive functions and compile this for both sequential and parallel HPC computers

SAGA Technologies

Sophus: This is a software library written in C++ and carefully designed to mimic the abstract structure of the PDE mathematics. By requiring strict adherence to specified interfaces, we have been able to achieve that different implementations of the same mathematical concepts are basically interchangeable.

SAGA Technologies

- Sophus library components can be presented as different layers of abstractions.
  - Application layer: solvers for PDEs such as the seismic simulator and the coating problem.
  - Tensor layer: handles coordinate systems, matrices and vectors and general differentiation operators.
  - Scalar field layer: numerical discretisations such as finite differences and finite elements with partial derivatives.
  - Mesh layer: implements grids for sequential and parallel HPC machines.

SAGA Technologies

CodeBoost: This is a software transformation system being developed to address the gap between well formed code (from a software engineering point of view) and efficient code (from a run-time point of view).

Data structures which are related to tensors - notes from SOPHUS seminars

Signature for **tensor** has the following form:

$$T < R, k > \tag{9}$$

where R-real numbers, k-dimension of the $\mathbb{R}^k$ space which is used to the definition of manifold.
More extended definition of tensors has the following form:

$$T < R, k, v, c > \tag{10}$$

where v is a number of vectors and c is a numbers of covectors. Traditionally tensor space on the manifold M in the point x is denoted by

$$T_c^v(M) = V \otimes ... \otimes V \otimes V^* \otimes ... \otimes V^* \tag{11}$$

Operations on tensors

In the tensor class there are implemented basic arithmetic operations.
Addition is an operation of tensor field.

$$+ : T_c^v \times T_c^v \longrightarrow T_c^v \tag{12}$$

More extended version of that operation is defined in the following way:

$$+ : T < R, k, v, c >, T < R, k, v, c > \longrightarrow T < R, k, v, c > \tag{13}$$

Semantic of that operation can be defined using components:

$$\begin{aligned} \mathbf{A} + \mathbf{B} &= A^{j_1 \dots j_v}_{i_1 \dots i_c} \mathbf{g}_{j_1} \otimes \dots \otimes \mathbf{g}^{i_c} + B^{j_1 \dots j_v}_{i_1 \dots i_c} \mathbf{g}_{j_1} \otimes \dots \otimes \mathbf{g}^{i_c} = \\ &= (A^{j_1 \dots j_v}_{i_1 \dots i_c} + B^{j_1 \dots j_v}_{i_1 \dots i_c}) \mathbf{g}_{j_1} \otimes \dots \otimes \mathbf{g}^{i_c} = \\ &= C^{j_1 \dots j_v}_{i_1 \dots i_c} \mathbf{g}_{j_1} \otimes \dots \otimes \mathbf{g}^{i_c} = \mathbf{C} \end{aligned} \tag{14}$$

If we neglect the basis vectors we will get

$$\mathbf{A} + \mathbf{B} = A^{j_1 \dots j_v}_{i_1 \dots i_c} + B^{j_1 \dots j_v}_{i_1 \dots i_c} = C^{j_1 \dots j_v}_{i_1 \dots i_c} = \mathbf{C} \tag{15}$$

Multiplication by number "*"

Multiplication is an operation of tensor field.

$$\begin{aligned} * : T_c^v \times R &\longrightarrow T_c^v \\ * : R \times T_c^v &\longrightarrow T_c^v \end{aligned} \tag{16}$$

More extended version of that operation is defined in the following way:

$$\begin{aligned} * : T < R, k, v, c >, R &\longrightarrow T < R, k, v, c > \\ * : R, T < R, k, v, c > &\longrightarrow T < R, k, v, c > \end{aligned} \tag{17}$$

Semantic of that operation can be defined using components:

$$\begin{aligned} \alpha * \mathbf{A} = \mathbf{A} * \alpha = \alpha * A_{i_1 \dots i_c}^{j_1 \dots j_v} \mathbf{g}_{j_1} \otimes \dots \otimes \mathbf{g}^{i_c} = \\ = C_{i_1 \dots i_c}^{j_1 \dots j_v} \mathbf{g}_{j_1} \otimes \dots \otimes \mathbf{g}^{i_c} = \mathbf{C} \end{aligned} \tag{18}$$

If we neglect the basis vectors we will get

$$\alpha * \mathbf{A} = \mathbf{A} * \alpha = \alpha * A_{i_1 \dots i_c}^{j_1 \dots j_v} = C_{i_1 \dots i_c}^{j_1 \dots j_v} = \mathbf{C} \tag{19}$$

FEM discretization of tensor fields

If we divide M into finite elements $M = \bigcup_{e=1} M_e$. Each finite element contain some nodes $x_{ei} \in M_e$. The values of the tensor fields inside element can be approximated using shape functions

$$\mathbf{T}(x) = \sum_{i=1} N_{ei}(x)\mathbf{T}_h(x_{ei}) \tag{20}$$

Due to continuity of tensor field it is possible to create global list of nodes.

$$x_{ei} = \sum_e \sum_j U_{eij} x_j \tag{21}$$

$$\mathbf{T}_h(x_{ei}) = \sum_e \sum_j U_{eij} \mathbf{T}_h(x_j) \tag{22}$$

If we substitute the equation (22) to the equation (20) then we will get:

$$\mathbf{T}_h(x) = \sum_i N_{ei}(x) \sum_e \sum_j U_{eij} \mathbf{T}_h(x_j) \tag{23}$$

FEM discretization of tensor fields

We can rewrite the equation(20) to the following form:

$$\mathbf{T}_h(x) = \sum_j \left(\sum_i \sum_e N_{ei}(x) U_{eij} \right) \mathbf{T}_h(x_j) \tag{24}$$

Now own can define global shape function in the following way:

$$N_j(x) = \sum_i \sum_e U_{eij} N_{ei}(x) \tag{25}$$

FEM discretization of tensor fields

$$\mathbf{T}_h(x) = \sum_i N_{ei}(x) \sum_e \sum_j U_{eij} \sum_k \mathbf{g}_k(x_j)\, T^{jk} \tag{26}$$

$$\mathbf{T}_h(x) = \sum_j \sum_k \left(\sum_i \sum_e N_{ei}(x) U_{eij} \mathbf{g}_k(x_j) \right) T^{jk} \tag{27}$$

$$\mathbf{T}_h(x) = \sum_j \sum_k \mathbf{N}_{jk}(x)\, T^{jk} \tag{28}$$

where

$$\mathbf{N}_{jk}(x) = \sum_i \sum_e N_{ei}(x) U_{eij} \mathbf{g}_k(x_j) \tag{29}$$

Variational equations in the space of tensors and its solution

Many BVP can be equivalently expressed as some variational equations.

$$\forall \mathbf{v} \in V \quad a(\mathbf{t}, \mathbf{v}) = l(\mathbf{v}) \tag{30}$$

Approximate solution of the equation (30) can be found in the approximation space V_h from the following equation.

$$\forall \mathbf{v_h} \in V_h \quad a(\mathbf{t_h}, \mathbf{v_h}) = l(\mathbf{v_h}) \tag{31}$$

If we introduce some coordinate system $\{x_i\}$ then we will get also some natural basis vectors of $T_x(M)$ tangent vector space. Using these basis vectors

$$\mathbf{g}_i = \frac{\partial \mathbf{r}}{\partial x_i} \tag{32}$$

Variational equations in the space of tensors and its solution

Each vector (or tensor) $\mathbf{t}(\mathbf{x}) \in T_x(M)$ can be described using contravariant t^i or covariant t_i coordinates and basis $\mathbf{g}_i$ in the following way

$$\mathbf{t} = \sum_i t^i \mathbf{g}_i = \sum_i t_i \mathbf{g}^i \tag{33}$$

The vectors $\mathbf{g}^i$ are basis of dual tangent vector space $T_x^*(M)$. Using FEM discretization one can get

$$\mathbf{t}_h(\mathbf{x}) = \sum_i \sum_j \mathbf{N}_{ij}(\mathbf{x}) t^{ij} \tag{34}$$

$$\mathbf{v}_h(\mathbf{x}) = \sum_k \sum_l \mathbf{N}_{kl}(\mathbf{x}) v^{kl} \tag{35}$$

where

$$\mathbf{t}_h(\mathbf{x}_j) = \sum_k t^{jk} \mathbf{g}_k(\mathbf{x}_j) \tag{36}$$

Variational equations in the space of tensors and its solution

If we substitute the equation (34) and (35) to the equation (30) we will get the following system of linear equations:

$$\sum_{ij} K_{ijkl} u^{ij} = Q_{kl} \tag{37}$$

In this system it is possible to take into account the homogenous and non-homogenous Dirichlet boundary conditions.
Some interesting results about finite element method on manifold are in the paper:

M. Holst, Adaptive numerical treatment of elliptic systems on manifolds. Advances in Computational Mathematics, 15: 139-191, 2001.

Example of coordinate free differential equation

Let us consider differential equation

$$\frac{du}{dx} = f(x) \tag{38}$$

We can rewrite this equation to the following "coordinate free" form

$$\nabla_X u = f \tag{39}$$

where X is a unit vector field which is parallel to the manifold M. From the definition of directional derivative we can calculate that

$$u(x(t)) - u(x(t_0)) = \int\limits_{\gamma(t_0,t)} \nabla_X u dr = \int\limits_{\gamma(t_0,t)} f dr \tag{40}$$

Then finally we have:

$$u(x(t)) = \int\limits_{\gamma(t_0,t)} f dr + u(x(t_0)) \tag{41}$$

Example of coordinate free differential equation

In coordinate description we can rewrite the equation (39) in the following form:

$$\frac{\partial u}{\partial x_i} X^i = f \tag{42}$$

In one dimensional problems we have

$$\frac{du}{dx} X(x) = f \tag{43}$$

The function $X(x)$ is known in each coordinate system and it is equal to .

$$X(x) = \frac{1}{\sqrt{g}} \tag{44}$$

then the equation (39) now has the form

$$\frac{1}{\sqrt{g}} \frac{du}{dx} = f \tag{45}$$

Finite difference method

Let us consider the following boundary value problem

$$\begin{cases} \frac{\partial u}{\partial x} = c\frac{\partial u}{\partial t} \\ u(x,t)|_{t=0} = \hat{u}_{t_0}(x) \\ u(x,t)|_{x=0} = \hat{u}_0(t) \\ u(x,t)|_{x=L} = \hat{u}_L(t) \end{cases} \tag{46}$$

In order to solve this problem FDM can be applied. Each partial derivative will be replaced by appropriate finite difference.

Finite difference method

In the internal points.

$$\frac{\partial u(x_i, t)}{\partial x} = \frac{u_{i+1}^t - u_{i-1}^t}{2\Delta x} \tag{47}$$

In the boundary points. Left boundary

$$\frac{\partial u(x_i, t)}{\partial x} = \frac{u_{i+1}^t - u_i^t}{\Delta x} \tag{48}$$

Right boundary

$$\frac{\partial u(x_i, t)}{\partial x} = \frac{u_i^t - u_{i-1}^t}{\Delta x} \tag{49}$$

The time derivative can be approximated for example using the following method:

$$\frac{\partial u(x_i, t)}{\partial t} = \frac{u_i^{t+\Delta t} - u_i^t}{\Delta t} \tag{50}$$

FDM

The whole algorithm has the following steps.
1) Initial boundary condition ($t = t_0$)

$$u_i^{t_0} = \hat{u}_{t_0}(x_i) \tag{51}$$

2) Update the time.

$$t = t + \Delta t \tag{52}$$

3) Apply boundary conditions.

$$u_0^t = \hat{u}_0(t), \quad u_n^t = \hat{u}_L(t) \tag{53}$$

4) Calculate the solution in the internal points.

$$u_i^{t+\Delta t} = u_i^t + \frac{u_{i+1}^t - u_{i-1}^t}{2\Delta x} \frac{\Delta t}{c} \tag{54}$$

5) If $t < t_{max}$ goto step 2.
6) End of calculations.

FDM - coordinate free

In order to make this example coordinate free we write the equation (46) in coordinate free way.

$$\begin{cases} L(u) = c\frac{\partial u}{\partial t} \\ u(x,t)|_{t=0} = \hat{u}_{t_0}(x) \\ u(x,t)|_{x=A} = \hat{u}_0(t) \\ u(x,t)|_{x=B} = \hat{u}_L(t) \end{cases} \tag{55}$$

FDM - coordinate free

The solution in our case may have the following form:
1) Initial boundary condition ($t = t_0$)

$$u_i^{t_0} = \hat{u}_{t_0}(x_i) \tag{56}$$

2) Update the time.

$$t = t + \Delta t \tag{57}$$

3) Apply boundary conditions.

$$u_A^t = \hat{u}_0(t), \quad u_B^t = \hat{u}_L(t) \tag{58}$$

4) Calculate the solution in the internal points.

$$u^{t+\Delta t} = u^t + L(u^t)\frac{\Delta t}{c} \tag{59}$$

5) If $t < t_{max}$ goto step 2.
6) End of calculations.

Review of differential operators on manifold

Not all operations are implemented in Sophus.

Base vectors of tangent and cotangent space

We can describe manifold M using the following map

$$\psi_{cartesian} \circ \phi^{-1} : R^n \supset U \ni y \rightarrow \psi_{cartesian} \circ \phi^{-1}(y) \in R^n \quad (60)$$

We will denote that function as $\chi = \psi_{cartesian} \circ \phi^{-1}$. Coordinate base of the space $T_0^1(M)$ contain the following vectors:

$$\frac{\partial r}{\partial y_i} = g_i = \frac{\partial \chi(y)}{\partial y_i} = \frac{\partial}{\partial y_i}(\psi_{cartesian} \circ \phi^{-1}(y)) \quad (61)$$

In cartesian coordinates we have

$$\left(\frac{\partial r}{\partial y_i}\right)^k = (g_i)^k = \frac{\partial \chi^k(y)}{\partial y_i} = \frac{\partial}{\partial y_i}(\psi^k_{cartesian} \circ \phi^{-1}(y)) \quad (62)$$

where $k = 1, ..., m$, $i = 1, ..., n$. We can write also that formula in a simpler way

$$\frac{\partial r}{\partial y_i} = g_i = \sum_k \frac{\partial x^k_{cartesian}(y)}{\partial y_i} e_k \quad (63)$$

Base of dual space

$$\frac{\partial}{\partial x_i} \cdot dx^j = g_i \cdot g^j = \delta_{ij} \tag{64}$$

If $m = n$ then the map $x = \chi(y) = x(y)$ is invertible and we can calculate the function $y = \chi^{-1}(x) = y(x)$ then

$$dy^i = g^i = \sum_k \frac{\partial x_i}{\partial x^k_{cartesian}} e_k \tag{65}$$

If we know the basis vectors of the spaces $T^1_0(M)$ then each vector $X \in T^1_0(M)$ can be described using contravariant components X^i.

$$X = \sum_i X^i g_i \tag{66}$$

If $Y \in T^0_1(M)$ then we can describe the vector Y using covariant components.

$$Y = \sum_i Y_i g^i \tag{67}$$

Metric tensor

Definition

A **weak pseudo-Riemannian metric** on a manifold M is defined to be a tensor field $g \in T_2^0(M)$ that is symmetric and weakly nondegenerate, that is, such that at each $m \in M, g(m)(v_m, w_m) = 0$ for all $w_m \in T_m(M)$ implies $v_m = 0$. A **strong pseudo-Riemannian metric** is a 2-tensor field that, in addition is strongly nondegenerate for all $m \in M$; that is, the map $v_m \to g(m)(v_m, .)$ is an isomorphism of $T_m(M)$ onto $T_m^*(M)$. A weak (resp., strong) pseudo-Riemannian metric is called weak (resp., strong) Riemannian if in addition $g(m)(v_m, v_m) > 0$ for all $v_m \in T_m(M),\ v_m \neq 0$.

Any Hilbert space is a Riemannian manifold with a constant metric equal to the inner product.

Metric tensor

If we have manifold M, coordinates of metric tensor can be defined using base vectors of the space $T_0^1(M)$ and the inner product

$$g_{ij} = g_i \cdot g_j = g(g_i, g_j) \tag{68}$$

In that case metric tensor is a tensor of type $T_2^0(M)$.

$$g = \sum_{ij} g_{ij} g^i \otimes g^j \tag{69}$$

It is also possible to define a metric tensor in the following way

$$g^{ij} = g^i \cdot g^j \tag{70}$$

In that case $g \in T_0^2(M)$

$$g = \sum_{ij} g^{ij} g_i \otimes g_j \tag{71}$$

The metric tensor of the type $T_1^1(M)$ is has the following components.

$$g = \sum_{ij} \delta_j^i g_i \otimes g^j = \sum_{ij} \delta_i^j g^i \otimes g_j \tag{72}$$

Metric tensor in cylindrical coordinates

In cylindrical coordinates metric tensor has the following covariant coordinates ($g = \sum_{ij} g_{ij} g^i \otimes g^j$)

$$[g_{ij}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & r^2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{73}$$

Contravariant coordinates ($g = \sum_{ij} g^{ij} g_i \otimes g_j$)are the following.

$$[g^{ij}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{r^2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{74}$$

Lowering and rising index

The following theorem come from Riesz.

Theorem

For every continuous linear functional f on a Hilbert space $\mathcal{H}$, there is a unique $u \in \mathcal{H}$ such that $f(x) = (x, u)$ for all $x \in \mathcal{H}$.
Note: (x, u) denotes the inner product between x and u.

As we can see in the spaces with inner product there is a map from the vector space V and dual space V^*. In other words for all $\alpha \in V^*$ there is $v = g^\sharp(\alpha) \in V$ such that

$$\forall x \in V \ \alpha(x) = g(v, x) = g(g^\sharp(\alpha), x) = g^\flat(v)(x) \qquad (75)$$

Lowering and rising index

Each vector v can have coordinates in the base $g_1, ..., g_n$ or in the base $g^1, ..., g^n$.

$$v = \sum_i v_i g^i = \sum_i v^i g_i \tag{76}$$

where

$$\begin{aligned} v_i &= \sum_j v^j g_{ij} \\ v^i &= \sum_j v_j g^{ij} \end{aligned} \tag{77}$$

Differential forms

Let us consider a vector space V. Differential k-forms are tensor fields of type $(0, k)$ that are completely antisymmetric.
If we introduce coordinate system then basis vectors of the space $\Omega^k(M)$ are 1-form $dx^1, ..., dx^m$. Each member of the space $\Omega^k(M)$ can be expressed in coordinates in the following way:

$$\omega = \sum_i \omega_i dx^i \tag{78}$$

where $\omega_i = \omega\left(\frac{\partial}{\partial x^i}\right)$.

Integration of differential forms

Let us consider differential form $\omega \in \Omega^k(M)$, some subset $U \subset M$ and

$$h : R^n \supset V \ni t \to h(t) = x \in U \subset M \tag{79}$$

is orientation-preserving diffeomorphism. Integral from the differential form ω over the set $U \subset M$ is defined as integral in the space R^n

$$\int_V h^*\omega = \int_U \omega \tag{80}$$

For example if

$$f : R^1 \supset (t_1, t_2) \ni t \to f(t) = (x(t), y(t), z(t)) \in U \subset R^3 \tag{81}$$

is parametric description of sum curve $\gamma \subset R^3$ then an integral from one form $\omega = Pdx + Qdy + Rdz$ over the curve γ is equal to

$$\int_\gamma \omega = \int_{(t_1,t_2)} f^*\omega = \int_{t_1}^{t_2} \left(P\frac{dx}{dt} + Q\frac{dy}{dt} + R\frac{dz}{dt} \right) dt \tag{82}$$

Hodge star

Let V be a n-dimensional (n finite) vector space with inner product g. The **Hodge star operator** (denoted by $*$) is a linear operator mapping p-forms on M to $(n-p)$-forms, i.e.,

$$* : \Omega^p(M^n) \longrightarrow \Omega^{n-p}(M^n). \tag{83}$$

In local coordinates $\{x^1, \ldots, x^n\}$, where $g = g_{ij} dx^i \otimes dx^j$, the $*$-operator is defined as the linear operator that maps the basis elements of $\Omega^p(M^n)$ as

$$\begin{gathered} *(dx^{i_1} \wedge \cdots \wedge dx^{i_p}) = \\ = \frac{\sqrt{|g|}}{(n-p)!} g^{i_1 l_1} \ldots g^{i_p l_p} \varepsilon_{l_1 \cdots l_p \, l_{p+1} \cdots l_n} dx^{l_{p+1}} \wedge \cdots \wedge dx^{l_n}. \end{gathered} \tag{84}$$

Here, $|g| = \det g_{ij}$, and ε is the Levi-Civita permutation symbol. This operator may be defined in a coordinate-free manner by the condition

$$u \wedge *v = g(u, v)\ \mathbf{Vol}(g) =< u, v >\ \mathbf{Vol}(g) \tag{85}$$

Interior product

Let us consider a vector $x \in T_0^1(M)$, convector $\alpha \in T_1^0(M)$ and a tensor $t \in T_c^v(M)$. The tensor t is equivalent to the following multilinear function

$$t(c_1, ..., c_v, v_1, ..., v_c) = \sum_{i_1,...,i_v,j_1,...,j_c} t^{i_1,...,i_v}_{j_1,...,j_c} c_{i_1} ... c_{i_v} v^{j_1} ... v^{j_c} \qquad (86)$$

Interior product of the tensor t and the vector x can be defined as the following mapping:

$$\begin{aligned} i_x t(c_1, ..., c_v, v_1, ..., v_{c-1}) &= t(c_1, ..., c_v, v_1, ..., v_{c-1}, x) = \\ &= \sum_{i_1,...,i_v,j_1,...,j_c} t^{i_1,...,i_v}_{j_1,...,j_c} c_{i_1} ... c_{i_v} v^{j_1} ... v^{j_{c-1}} x^{j_c} = \\ &= \sum_{i_1,...,i_v,j_1,...,j_{c-1}} \left(\sum_k t^{i_1,...,i_v}_{j_1,...,j_{c-1},k} x^k \right) c_{i_1} ... c_{i_v} v^{j_1} ... v^{j_{c-1}} = \\ &= \sum_{i_1,...,i_v,j_1,...,j_{c-1}} (i_x t)^{i_1,...,i_v}_{j_1,...,j_{c-1}} c_{i_1} ... c_{i_v} v^{j_1} ... v^{j_{c-1}} \end{aligned} \qquad (87)$$

In coordinates:

$$(i_x t)^{i_1,...,i_v}_{j_1,...,j_{c-1}} = \sum_k t^{i_1,...,i_v}_{j_1,...,j_{c-1},k} x^k \qquad (88)$$

Inner product of two tensor in Hilbert spaces

Definition

Let H_1 and H_2 be two Hilbert spaces with inner products $\cdot_1$ and $\cdot_2$, respectively. **Inner product** of two tensors $\phi_1 \otimes \phi_2$, $\psi_1 \otimes \psi_2$ where $\phi_1, \psi_1 \in H_1$, $\phi_2, \psi_2 \in H_2$ can be defined in the following way:

$$(\phi_1 \otimes \phi_2) \cdot (\psi_1 \otimes \psi_2) = (\phi_1 \cdot \psi_1)(\phi_2 \cdot \psi_2) \tag{89}$$

Contraction

Let us consider a tensor $t \in T_c^v(M)$ the **contraction** of the $k-th$ contravariant and the $l-th$ covariant index or for short the (k,l) contraction, is the family of linear maps

$$C_l^k : T_c^v(M) \ni t \longrightarrow C_l^k(t) \in T_{c-1}^{v-1}(M) \tag{90}$$

where the contracted tensor has the following components.

$$C_l^k(t)^{i_1,\ldots,i_{k-1},\hat{i}_k,i_{k+1},\ldots,i_v}_{j_1,\ldots,j_{l-1},\hat{j}_l,j_{l+1},\ldots,j_v} = \sum_p t^{i_1,\ldots,i_{k-1},p,i_{k+1},\ldots,i_v}_{j_1,\ldots,j_{l-1},p,j_{l+1},\ldots,j_v} \tag{91}$$

An important particular example of contraction is **trace operator**. If $t \in T_1^1(M)$ then

$$tr(t) = C_1^1(t) = \sum_i t_i^i \tag{92}$$

Christoffel symbols

If one would like to calculate partial derivative with respect to the curvilinear coordinates we will get **Christoffel symbols**

$$\frac{\partial g_i}{\partial x^j} = \sum_k \Gamma_{ij}^k g_k \tag{93}$$

$$\frac{\partial g_i}{\partial x^j} = \sum_k [k, ij] g^k \tag{94}$$

The symbol

$$[k, ij] = \frac{\partial g_i}{\partial x^j} \cdot g^k \tag{95}$$

is **Christoffel symbol of the first kind**.
The symbol

$$\Gamma_{ij}^k = \left\{ \begin{array}{c} k \\ ij \end{array} \right\} = \frac{\partial g_i}{\partial x^j} \cdot g_k \tag{96}$$

is **Christoffel symbol of the second kind**.

Christoffel symbol

$$\left(\frac{\partial g_j}{\partial x_k}\right)^i = \frac{1}{2} g^{im} \left(\frac{\partial g_{mj}}{\partial x_k} - \frac{\partial g_{kj}}{\partial x_m} + \frac{\partial g_{mk}}{\partial x_j}\right) = \Gamma^i_{jk} \qquad (97)$$

Covariant derivative (Covariant derivative of scalar filed)

Let us consider scalar filed $f : M \to R$ and the vector field X and the flow F_t^X which is generated by the vector field X.

$$\nabla_X f = \frac{d}{dt} f \circ F_t^X \quad for \quad t = 0 \tag{98}$$

As we can see covariant derivative of scalar filed is equal to Lie derivative and in coordinates can be expressed as

$$\nabla_X f(p) = \sum_i \frac{\partial f(p)}{\partial x^i} X^i(p) \tag{99}$$

where $p \in M$.

Covariant derivative of vectors

The **covariant derivative** of the vector filed u in the direction of the vector v can be calculated in the following way:

$$\nabla u_v = \sum_k v^i \left(\sum_i \frac{\partial u^k}{\partial x^i} + \sum_{ij} \Gamma^k_{ij} u^j \right) g_k = \sum_k v^i u^k_{;i} g_k \qquad (100)$$

Covariant derivative can be viewed as an operator

$$\nabla : T^1_0(M) \times T^1_0(M) \in (X, Y) \longrightarrow \nabla_X Y \in T^1_0(M) \qquad (101)$$

or

$$\nabla : T^1_0(M) \ni Y \longrightarrow \nabla Y \in T^1_1(M) \qquad (102)$$

in that case $\nabla Y : T^1_0(M) \ni X \longrightarrow \nabla_X Y \in T^1_0(M)$ then ∇Y is a linear map on $T^1_0(M)$ i.e. it is a tensor filed $T^1_1(M)$.

Covariant derivative of tensors fields

If we have tensor filed $t \in T_c^v(M)$ then the covariant derivative of that tensor filed in the direction of the base vector $\frac{\partial}{\partial x^c}$ can be defined in the following way:

$$\begin{aligned}(\nabla_{\frac{\partial}{\partial x^c}} t)^{i_1,\ldots,i_v}_{j_1,\ldots,j_c} = t^{i_1,\ldots,i_v}_{j_1,\ldots,j_c;c} = \\ = \frac{\partial t^{i_1,\ldots,i_v}_{j_1,\ldots,j_c}}{\partial x^c} + \\ + \sum_{m=1}^{v} \sum_{d} \Gamma^{i_m}_{dc} t^{i_1,\ldots,i_{m-1},d,i_{m+1},\ldots,i_v}_{j_1,\ldots,j_c} - \\ - \sum_{n=1}^{c} \sum_{p} \Gamma^{p}_{j_n c} t^{i_1,\ldots,i_v}_{j_1,\ldots,j_{n-1},p,j_{n+1},\ldots,j_c}\end{aligned} \tag{103}$$

As we can see covariant derivative is a function

$$\nabla_{(.)}(.) : T_0^1(M) \times T_c^v(M) \ni t \longrightarrow \nabla_v t \in T_c^v(M) \tag{104}$$

Lie derivative (Lie derivative of scalar filed)

If we have a function $f : M \longrightarrow R$ and the vector filed $X = \sum_i X^i g_i$, then we can define the Lie derivative of the function f in the direction of the vector filed X in the following way:

$$\mathcal{L}_X f(p) = df[X(p)] = \sum_i \frac{\partial f(p)}{\partial x^i} X^i(p) \in R \tag{105}$$

where df is exterior derivative of the 0-form f.
The following set of relation holds:

$$\begin{array}{l} \frac{d}{dt} F_t^* f(p) = \frac{d}{dt} f(F_t(p)) = df(F_t(p)) \frac{d}{dt} F_t(p) = \\ = df(F_t(p)) X(F_t(p)) = \mathfrak{L}_X f(F_t(p)) = F_t^* \mathfrak{L}_X f \end{array} \tag{106}$$

As we can see Lie derivative is an operation from the space $T_0^0(M)$ to the space $T_0^0(M)$

$$\mathcal{L} : T_0^1(M) \times T_0^0(M) \ni (X, v) \longrightarrow \mathcal{L}_X(v) = \sum_i \frac{\partial v^i}{\partial x^i} X^i \in T_0^0(M) \tag{107}$$

Lie derivative of vector filed

If we have two vector field $X, Y \in T_0^1(M)$, then

$$\mathcal{L}_X Y = [X, Y] = (X\nabla)Y - (Y\nabla)X = \sum_{ij} \left(X^i \frac{\partial Y^j}{\partial x^i} - Y^i \frac{\partial X^j}{\partial x^i} \right) g_j \tag{108}$$

If the vector filed X has the flow Fl_t i.e.

$$\frac{d}{dt} Fl_t(t) = X \quad for \quad t = 0 \tag{109}$$

If we have the vector field Y then

$$\frac{d}{dt} F_t^{X*}(Y) = \mathcal{L}_X Y \quad for \quad t = 0 \tag{110}$$

It is possible to rewrite this expression in the following form:

$$\mathcal{L}_X Y(x) = \lim_{t \longrightarrow 0} \frac{(F_t^{X*} Y)(x) - F_0^{X*} Y(x)}{t} \quad for \quad t = 0 \tag{111}$$

Lie derivative of vector filed

As we can see Lie derivative of vector filed is a function from the space $T_0^1(M)$ to the space $T_0^1(M)$.
As we can see Lie derivative is an operation from the space $T_0^1(M)$ to the space $T_0^1(M)$

$$\mathcal{L} : T_0^1(M) \times T_0^1(M) \ni (X, Y) \longrightarrow \mathcal{L}_X Y \in T_0^1(M) \qquad (112)$$

Lie derivative of tensor filed

Lie derivative in coordinates can be defined in the following way:

$$
\begin{aligned}
(\mathcal{L}_X t)^{i_1 \dots i_v}_{j_1 \dots j_c} = \\
= \sum_k \frac{\partial t^{i_1 \dots i_v}_{j_1 \dots j_c}}{\partial x^k} X^k - \\
- \sum_{m=1}^{v} \sum_p \frac{\partial X^{i_m}}{\partial x^p} t^{i_1, \dots, i_{m-1}, p, i_{m+1}, \dots, i_v}_{j_1 \dots j_c} + \\
+ \sum_{n=1}^{c} \sum_q \frac{\partial X^q}{\partial x^{j_n}} t^{i_1 \dots i_v}_{j_1, \dots, j_{n-1}, q, j_{n+1}, \dots, j_c}
\end{aligned}
\tag{113}
$$

As we can see Lie derivative is the following function

$$
\mathcal{L} : T^1_0(M) \times T^v_c(M) \ni (X, t) \longrightarrow \mathcal{L}_X t \in T^v_c(M) \tag{114}
$$

Gradient (spatial gradient of scalar fields)

In cartesian coordinate system gradient is defined as the following vector:

$$grad(f_{cartesian}) = \nabla f_{cartesian} = \sum_i \frac{\partial f_{cartesian}}{\partial x_i} e_i \tag{115}$$

In curvilinear coordinate system

$$grad(f_{curvilinear}) = \nabla f_{curvilinear} = \sum_i \frac{\partial f_{curvilinear}}{\partial x_i} g^i \in T_1^0(M) \tag{116}$$

Where g^i is a vector of (dual) base in the space $T_1^0(M)$. As we can see gradient is a covariant vector. Using this definition the gradient is a function from $T_0^0(M)$ to $T_1^0(M)$

$$\nabla : T_0^0(M) \ni v \longrightarrow \nabla v = \sum_i \frac{\partial v}{\partial x^i} g^i \in T_1^0(M) \tag{117}$$

Gradient (spatial gradient of scalar fields)

In some books the gradient is defined as a dual vector to df and it is element of the space $T_0^1(M)$

$$(df)^\sharp = grad(f_{cartesian}) \in T_0^1(M) \tag{118}$$

$$[grad(f)]^\flat = df_{cartesian} \in T_1^0(M) \tag{119}$$

In general curvilinear coordinate systems we have:

$$\begin{gathered} grad(f_{curvilinear}) = \nabla f_{curvilinear} = \\ = \sum_{ij} g^{ij} \frac{\partial f}{\partial x_i} \frac{\partial}{\partial x_j} = \sum_{ij} g^{ij} \sqrt{g_{jj}} \frac{\partial f}{\partial x_i} \left(\frac{1}{\sqrt{g_{jj}}} \frac{\partial}{\partial x_j} \right) = \\ = \sum_{ij} g^{ij} \sqrt{g_{jj}} \frac{\partial f}{\partial x_i} e_j \in T_0^1(M) \end{gathered} \tag{120}$$

In orthogonal coordinate systems we have:

$$\begin{gathered} grad(f_{curvilinear}) = \nabla f_{curvilinear} = \\ = \sum_{ij} g^{ij} \frac{\partial f}{\partial x_i} \frac{\partial}{\partial x_j} = \sum_{ij} g^{ij} \sqrt{g_{jj}} \frac{\partial f}{\partial x_i} \left(\frac{1}{\sqrt{g_{jj}}} \frac{\partial}{\partial x_j} \right) = \\ = \sum_{i} g^{ii} \sqrt{g_{ii}} \frac{\partial f}{\partial x_i} e_i = \sum_{i} \frac{1}{g_{ii}} \sqrt{g_{ii}} \frac{\partial f}{\partial x_i} e_i = \sum_{i} \frac{1}{\sqrt{g_{ii}}} \frac{\partial f}{\partial x_i} e_i \in T_0^1(M) \end{gathered} \tag{121}$$

Gradient of vector fields

$$grad(v) = \frac{dv}{dx} = \sum_{ij} \left(\frac{dv^i}{dx^j} + \sum_k \Gamma^i_{kj} v^k \right) g_i \otimes g^j \tag{122}$$

It is possible to define gradient using covariant derivative:

$$grad(v) = \nabla v = \frac{dv}{dx} = \sum_{ij} v^i_{;j} g_i \otimes g^j \tag{123}$$

As we can see gradient of the vector filed $T^1_0(M)$ is a tensor filed $T^1_1(M)$.

$$\nabla : T^1_0(M) \ni v \longrightarrow \nabla v \in T^1_1(M) \tag{124}$$

Gradient of tensor fields

Gradient of tensor field $t \in T^v_c(M)$ is a tensor filed $\nabla t \in T^v_{c+1}(M)$

$$\nabla : T^v_c(M) \ni t \longrightarrow \nabla t \in T^v_{c+1}(M) \tag{125}$$

In coordinates this operation can be described in the following way:

$$\nabla t = \nabla \left(\sum_{i_1 \ldots i_c, j_1 \ldots j_v} t^{j_1 \ldots j_v}_{i_1 \ldots i_c} g_{j_1} \otimes \ldots \otimes g_{j_v} \otimes g^{j_1} \otimes \ldots \otimes g^{j_c} \right) =$$
$$= \sum_{i_1, \ldots, i_c, j_1, \ldots, j_v, k} \left(t^{j_1 \ldots j_v}_{i_1 \ldots i_c} \right)_{;k} g_{j_1} \otimes \ldots \otimes g_{j_v} \otimes g^{j_1} \otimes \ldots \otimes g^{j_c} \otimes g^k \tag{126}$$

Derivative of functions with tensor arguments

Let us consider the function $f : T_c^v(M) \ni t \to f(t) \in R$.
Differential of such function is defined in the following way:

$$df(t, dt) = \nabla f(t) \cdot dt \tag{127}$$

If $t \in T_c^v(M)$ then
The gradient of the function the the $f : T_c^v(M) \ni t \to f(t) \in R$
i.e.

$$\nabla f(t) = \sum_{i_1,\ldots,i_c,j_1,\ldots,j_v} \frac{\partial f(t)}{\partial t^{i_1,\ldots,i_c}_{j_1,\ldots,j_v}} g^{i_1} \otimes \ldots \otimes g^{i_v} \otimes g_{j_1} \otimes \ldots \otimes g_{j_c} \tag{128}$$

is a function $\nabla f : T_c^v(M) \ni t \to f(t) \in T_v^c(M)$.

$$\nabla f : T_c^v(M) \ni t \to \nabla f(t) \in T_v^c(M) \tag{129}$$

$$\nabla : C^1(T_c^v(M), R) \times T_c^v(M) \ni (f, t) \to \nabla f(t) \in T_v^c(M) \tag{130}$$

$$df : T_c^v(M) \times T_c^v(M) \ni (t, dt) \to \nabla f(t) \cdot dt \in R \tag{131}$$

$$\begin{gathered} d : C^1(T_c^v(M), R) \times T_c^v(M) \times T_c^v(M) \ni (t, dt) \to \\ \to df(r, dt) = \nabla f(t) \cdot dt \in R \end{gathered} \tag{132}$$

Derivative of functions with tensor arguments

If the function f is tensor-valued

$$f : T_c^v(M) \ni t \to f(t) \in T_q^p(M) \tag{133}$$

then the gradient is a function:

$$\nabla f : T_c^v(M) \ni t \to \nabla f(t) \in T_{q+v}^{p+c}(M) \tag{134}$$

$$\nabla : C^1(T_c^v(M), T_q^p(M)) \times T_c^v(M) \ni (f, t) \to \nabla f(t) \in T_{q+v}^{p+c}(M) \tag{135}$$

The differential is a function:

$$df : T_c^v(M) \times T_c^v(M) \ni (t, dt) \to \nabla f(t) \cdot dt \in T_q^p(M) \tag{136}$$

$$\begin{gathered} d : C^1(T_c^v(M), T_q^p(M)) \times T_c^v(M) \times T_c^v(M) \ni (t, dt) \to \\ \to df(r, dt) = \nabla f(t) \cdot dt \in T_q^p(M) \end{gathered} \tag{137}$$

Exterior derivative

Exterior derivative is a function

$$d : T_k^0(M) \supset \Omega^k(M) \ni \omega \longrightarrow d\omega \in \Omega^{k+1}(M) \subset T_{k+1}^0(M) \quad (138)$$

Let $U \subset M$, exterior derivative has the following properties.
1) If $\alpha \in \Omega^k(U)$, $\beta \in \Omega^l(U)$

$$d(\alpha \wedge \beta) = d\alpha \wedge \beta + (-1)^k \alpha \wedge d\beta \quad (139)$$

2) If $f_{curvilinear} \in \mathcal{F}^1(U) = \Omega^0(U)$ then:

$$df_{curvilinear} = \sum_i \frac{\partial f_{curvilinear}}{\partial x_i} dx^i \quad (140)$$

3) $d^2 = d \circ d = 0$

Exterior derivative

Let $\alpha \in \Omega^k(U)$. In coordinates we can express the form α in the following way:

$$\alpha = \sum_{i_1 < \ldots < i_k} \alpha_{i_1 \ldots i_k} dx^{i_1} \wedge \ldots \wedge dx^{i_k} \tag{141}$$

then the exterior derivative has the following form

$$d\alpha = \sum_{i < i_1 < \ldots < i_k} \frac{\partial \alpha_{i_1 \ldots i_k}}{\partial x^i} dx^i \wedge dx^{i_1} \wedge \ldots \wedge dx^{i_k} \tag{142}$$

The codifferential

The exterior derivative and the Hodge star operator enable us to introduce the following linear operator δ.

Theorem

The codifferential

$$\delta : \Omega^{k+1}(M) \rightarrow \Omega^{k}(M) \tag{143}$$

is defined by $\delta\alpha = 0$ *where* $\alpha \in \Omega^0(M) = T^0_0(M)$ *and on* $k+1$ *forms by* $\beta \in \Omega^{k+1}(M)$

$$\delta\beta = (-1)^{nk+1+Ind(g)} \star d \star \beta \tag{144}$$

The codifferential is the adjoint of the exterior derivative, in that

$$(\delta\alpha, \beta) = (\alpha, d\beta) \tag{145}$$

where

$$(\alpha, \beta) = \int_M \langle \alpha, \beta \rangle dV \tag{146}$$

Divergence

Divergence is a function $div : T_0^1(M) \longrightarrow T_0^0(M)$ and can be defined as

$$div(X) = \star(d(\star(X)^\flat)) \tag{147}$$

That formulation is valid only in the case when $\star\star\alpha = \alpha$. In general the formula for divergence is the following

$$div(X) = (-1)^{n(p-1)} \star (d(\star(X)^\flat)) \tag{148}$$

where n is a dimension of base space and $X \in T_0^p(M)$. In the index notation we will have:

$$div(X) = \nabla \cdot X = \sum_i X^i_{;i} \tag{149}$$

That definition can be also written in the following way:

$$div(X) = tr\,(\nabla X) = \nabla X : 1 \tag{150}$$

where $1 = \sum_{ij} \delta^i_j g_i \otimes g^j \in T_1^1(M)$ and ":" is double contraction operator.

Divergence

There is also the following definition of divergence operator.

$$div_g(X) = -\delta(X^\flat) \tag{151}$$

where δ is codifferential operator.
Divergence can be defined also as a formal scalar product of gradient and vector field

$$\begin{gathered} div(v) = \nabla \cdot v = \left(\sum_i (.)_{;i} g^i\right) \cdot \left(\sum_j v^j g_j\right) = \\ = \left(\sum_i (.)_{;i} g^i\right) \cdot \left(\sum_j v^j g_j\right) = \sum_{ij} v^j_{;i} g^i \cdot g_j = \\ = \sum_{ij} v^j_{;i} \delta_{ij} = \sum_i v^i_{;i} \end{gathered} \tag{152}$$

It can be also shown that the following relation holds

$$div(X) = \sum_i \frac{1}{\sqrt{g}} \frac{\partial}{\partial x^i} \left(\sqrt{g} X^i\right) \tag{153}$$

Divergence

If ω is volume vector form and X is a vector filed then divergence can be defined using Lie derivative

$$\mathfrak{L}_X\omega = div(X)\omega \tag{154}$$

Where $\mathfrak{L}_X\omega$ is a Lie derivative of the form ω in the direction of the vector field X.
Divergence of tensor field $t \in T_c^v(M)$ can be defined using the formula (150)

$$div(t) = tr\,(\nabla t) = \nabla X : 1 \tag{155}$$

Then divergence is clearly a function from $T_c^v(M)$ to $T_c^{v-1}(M)$

$$div : T_c^v(M) \ni t \longrightarrow div(t) \in T_c^{v-1}(M) \tag{156}$$

Curl

Curl operator can be defined in the following way

$$curl(X) = (*(d(X^\flat)))^\sharp \tag{157}$$

In coordinates curl can be expressed in the following way

$$curl(X) = \nabla \times X = \sum_{ijk} \varepsilon_{ijk} X_{k;j} g_i \tag{158}$$

Let's define third order permutation tensor

$$E^{(3)} = \sum_{ijk} \varepsilon^{ijk} e_i \otimes e_j \otimes e_k \tag{159}$$

where e_i are the basis vectors in the cartesian coordinate system.

$$curl(X) = E^{(3)} : (\nabla X)^T \tag{160}$$

As we can see the curl is an operation from the vector space $T_0^1(M)$ to the vector space $T_0^1(M)$

$$curl : T_0^1(M) \ni X \rightarrow curl(X) \in T_0^1(M) \tag{161}$$

Laplacian

The Laplacian in curvilinear coordinate system can be calculated in the following way:

$$\Delta f_{curvilinear} = \frac{1}{\sqrt{g}} \sum_{ij} \frac{\partial}{\partial x^i} \left(g^{ij} \sqrt{g} \frac{\partial f_{curvilinear}}{\partial x_j} \right) \tag{162}$$

Laplacian can be defined superposition of *div* and *grad* $= \nabla$. If $u \in T_0^0(M)$.

$$\Delta u = div(\nabla u) \tag{163}$$

As we can see Laplacian is a function from $T_0^0(M)$ to $T_0^0(M)$.

$$\Delta : T_0^0(M) \ni u \longrightarrow \Delta u \in T_0^0(M) \tag{164}$$

It is possible to define Laplacian as

$$\Delta f = \star d \star df \tag{165}$$

Laplacian

It is also possible to define Laplace-deRham operator.

$$\Delta : \Omega^k(M) \rightarrow \Omega^k(M) \tag{166}$$

where

$$\Delta = d\delta + \delta d \tag{167}$$

For example if $f \in \Omega^0(M) = T_0^0(M)$.

$$\Delta f = d\delta f + \delta df = \delta df = -div(grad(f)) = -\nabla^2 f \tag{168}$$

where ∇^2 is Laplace-Beltrami operator.

Conclusions

- Using algebraic techniques for software structuring purposes gives more mature software structures which are less prone to redesigns than conventionally developed systems, providing a general framework for solvers of most kinds of PDEs, with
- a high degree of reuse even within the library itself, so that advanced concepts are defined by combinations of basic modules, which provides the ability that
- a numerical solution strategy may be replaced with another just by substituting a few modules and without changing the system structure,
- prototype code development times are reduced since central numerical modules are reused rather than readapted, and only a few minor modules need to be replaced when porting to and between HPC architectures.

Figure: Thank you