Using Peano's space-tree traversal for explicit discontinuous Galerkin methods

Dominic Etienne Charrier

November 30, 2015

1 Introduction

We show how to use Peano's core functionality, the space-tree traversal, and Peano's patch toolbox to solve partial differential equations of the evolution type (parabolic, hyperbolic) using explicit discontinuous Galerkin (DG) methods.

To this end, we discuss a linear advection problem. The governing equations are discretized according the method of lines. For the discretization in time, we employ an explicit Euler time marching scheme. We discretize the governing equations in space using an upwind DG formulation. While this is a rather simple setup, it already covers most of the steps that have to be performed for a wide range of linear hyperbolic problems and their corresponding explicit DG discretizations.

1.1 Preliminaries

The (classical) divergence $\boldsymbol{\nabla}\cdot\boldsymbol{T}$ (a vector-valued function) of a sufficiently regular second-order tensor-valued function $\boldsymbol{T}$ is defined according to:

$$(\boldsymbol{\nabla}\cdot\boldsymbol{T})_i = \sum_{j=1}^{d} \frac{\partial}{\partial x_j} T_{ij}, \qquad i = 1,\ldots,N_{\text{var}} \tag{1.1}$$

where N_{var} denotes the number of variables and d the space dimension.

1.2 Governing equation

Let $\Omega \subset \mathbb{R}^d$, $d = 2,3$, denote a rectangular domain with boundary $\partial\Omega$. We want to model the time evolution of a material concentration u inside of a flow field $\boldsymbol{b}$ by means of the following conservation law:

$$\frac{\partial u}{\partial t} + \boldsymbol{\nabla}\cdot\boldsymbol{F}(u) = 0 \qquad \text{in } \Omega\times(0,T], \tag{1.2a}$$

$$u(\boldsymbol{x},0) = u^0(\boldsymbol{x}) \qquad \text{in } \Omega\times\{0\}, \tag{1.2b}$$

with u^0 denoting a sufficiently regular initial condition, T denoting the simulation time, and

$$\boldsymbol{F}(u) = \boldsymbol{b}\,u \in \mathbb{R}^{N_{\text{var}}\times d} \tag{1.2c}$$

denoting the second-order tensor-valued physical flux ($N_{\text{var}} = 1$). Please note that $\boldsymbol{F}(u)$ and $\boldsymbol{b}$ are here defined as row vectors. Here and below, $\boldsymbol{b}$ is assumed to be sufficiently

regular. To complete the problem formulation, the material concentration is further subject to sufficiently regular inflow boundary conditions, i.e.,

$$u(\boldsymbol{x}, t) = j(\boldsymbol{x}, t), \qquad (\boldsymbol{x}, t) \in \partial\Omega_- \times (0, T]. \tag{1.2d}$$

2 An explicit one-step discontinuous Galerkin method for linear advection

2.1 Computational mesh and trace operators

Let $\mathcal{T}_h$ be a structured Cartesian mesh on $\Omega \subset \mathbb{R}$. We refer to the disjoint open sets $K \in \mathcal{T}_h$ as elements. We denote by h_K the diameter of K. We store the local quantities h_K in the vector $\underline{\mathrm{h}} = \{h_K\}_{K\in\mathcal{T}_h}$, and set $h_{\max} = \max_{K\in\mathcal{T}_h} h_K$. Finally, $\boldsymbol{n}$ denotes the outward normal unit vector to the element boundary ∂K. We will denote the number of elements by N_K.

An interior face of $\mathcal{T}_h$ is the $d-1$ dimensional intersection $\partial K^+ \cap \partial K^-$, where K^+ and K^- are two adjacent elements of $\mathcal{T}_h$. Similarly, a boundary face of $\mathcal{T}_h$ is the $d-1$ dimensional intersection $\partial K \cap \partial\Omega$ which consists of entire faces of ∂K. We denote by Γ_h the union of all interior faces of $\mathcal{T}_h$.

Here and in the following, we refer generically to a "face" even in the case $d = 2$. Now, let $e \in \Gamma_h$ be an interior face shared by the elements K^+ and K^-. We denote by φ a scalar-valued function (in $H^1(\mathcal{T}_h)$). We observe that the traces of φ are defined twice on e. Let $\varphi^\pm$ denote the trace of φ on e from the interior of $K^\pm$; we define a jump operator $[\![\varphi\boldsymbol{n}]\!]$ by

$$[\![\varphi\boldsymbol{n}]\!] = \varphi^+\boldsymbol{n}^+ + \varphi^-\boldsymbol{n}^-. \tag{2.1}$$

2.2 Discretization of the linear advection problem

We discretize (1.2a) – (1.2d) following the method of lines procedure. Thus, we make the ansatz

$$u_h(\boldsymbol{x}, t) = \sum_m u_m(t)\,\phi_i(\boldsymbol{x}), \tag{2.2}$$

with time-dependent coefficients u_i, and basis functions $\boldsymbol{\phi}_i$ that depend solely on $\boldsymbol{x}$ and belong to the discrete space

$$\mathcal{V}_h = \left\{v_h \in L^2(\Omega) : v_h|_K \in Q_N(K),\, \forall K \in \mathcal{T}_h\right\} \tag{2.3}$$

with N denoting the order of approximation, and $Q_N(K)$ denoting the space of polynomials of degree at most N. Note that the above ansatz uses indices in terms of a global basis. It is however often more useful to introduce a local basis function index. This

index together with the index of the element the basis function "lives in" can then be mapped to a global index if necessary. Thus, we make the following ansatz instead:

$$u_h(\boldsymbol{x},t) = \sum_{c=1}^{N_K} \sum_{i=1}^{N_{\text{basis}}} u_{c,m}(t)\, \phi_{c,m}(\boldsymbol{x}), \tag{2.4}$$

Furthermore, for any boundary face $e \in \partial\Omega_+$ of the mesh, we introduce inflow ("$\downarrow$") and outflow ("$\uparrow$") parts

$$e^{\downarrow} = \{\boldsymbol{x} \in e : (\boldsymbol{b}\cdot\boldsymbol{n})(\boldsymbol{x}) < 0\}, \qquad e^{\uparrow} = \{\boldsymbol{x} \in e : (\boldsymbol{b}\cdot\boldsymbol{n})(\boldsymbol{x}) \geq 0\}. \tag{2.5}$$

With respect to the previous definitions, the semidiscrete upwind DG formulation of problem (1.2a) – (1.2d) consists of finding a sufficient regular $u_h \in \mathcal{V}_h$ such that

$$\begin{aligned}\sum_{K\in\mathcal{T}_h} \int_K \frac{\partial}{\partial t} u_h\, v_h \,\mathrm{d}\boldsymbol{x} = \sum_{K\in\mathcal{T}_h} \int_K \boldsymbol{F}(u_h)\cdot\nabla v_h \,\mathrm{d}\boldsymbol{x} - \sum_{e\in\Gamma_h} \int_e \boldsymbol{G}(u_h^+, u_h^-)\cdot[\![v_h \boldsymbol{n}]\!]\,\mathrm{d}s \\ - \sum_{e\in\partial\Omega} \int_{e^{\uparrow}} \boldsymbol{F}(u_h)\cdot\boldsymbol{n}\, v_h \,\mathrm{d}s - \sum_{e\in\partial\Omega} \int_{e^{\downarrow}} (\boldsymbol{F}(j)\cdot\boldsymbol{n})\, v_h \,\mathrm{d}s, \qquad \forall v_h \in \mathcal{V}_h.\end{aligned} \tag{2.6}$$

The above equations are the result of elementwise partial integration and a consequent summation over all elements. To complete the DG formulation, we have to define unique values on the interior faces, the so-called numerical fluxes here denoted by $\boldsymbol{G}^*(u_h^+, u_h^-)$. Above, we have decided to use so-called upwind fluxes, i.e., we choose,

$$\boldsymbol{G}(u_h^+, u_h^-) = u_h^{\uparrow}\,\boldsymbol{b}. \tag{2.7}$$

where we have defined the upstream/upwind value ("$\uparrow$") of u_h according to

$$u_h^{\uparrow}(\boldsymbol{x}) = \begin{cases} u_h^+ & (\boldsymbol{b}\cdot\boldsymbol{n}^+)(\boldsymbol{x}) \geq 0, \\ u_h^- & (\boldsymbol{b}\cdot\boldsymbol{n}^-)(\boldsymbol{x}) > 0, \end{cases} \tag{2.8}$$

with $\boldsymbol{x} \in e$ and $u_h^{\pm} = u_h|_{K^{\pm}}$.

Note that the choice of the fluxes has effects on both, the stability and the order of convergence of a DG method Note further that for the linear advection equation, many numerical fluxes, e.g., Lax-Friedrichs, HLLE, etc., reduce to the above upwind flux. One particluar exception is the central flux which leads to an unstable method; cf. [2] for a source of all of the statements above.

After the choice of the fluxes, we need to choose a basis of the space $\mathcal{V}_h$ to complete the spatial discretization. We give examples for such a basis in the next sections.

If we choose a basis, the above discrete variational problem is equivalent to the following system of ordinary differential equations:

$$\underline{\mathrm{M}}\,\frac{\mathrm{d}}{\mathrm{d}t}\underline{\mathrm{u}}(t) = (\underline{\mathrm{F}} + \underline{\mathrm{G}} + \underline{\mathrm{O}})\,\underline{\mathrm{u}}(t) + \underline{\mathrm{I}}(t), \qquad \text{almost everywhere in } (0,T), \tag{2.9}$$

s.t. the initial conditions

$$\underline{\mathrm{u}}(0) = \underline{\mathrm{u}}^0, \tag{2.10}$$

with the vector $\underline{\mathrm{u}}(t)$ containing the time-dependent coefficients to the time-independent basis functions. Note that only the operation $\underline{\mathrm{G}}\,\underline{\mathrm{u}}$ is non-local and requires communication between processes. Note further that the mass operator $\underline{\mathrm{M}}$ is in general block-diagonal or even diagonal if an $L^2(\Omega)$-orthogonal basis is used. We will further refer to $\underline{\mathrm{F}}$ as the volume flux operator, to $\underline{\mathrm{O}}$ as the outflow operator, to $\underline{\mathrm{G}}$ as the normal flux operator, and to $\underline{\mathrm{I}}$ as the inflow vector. We will clarify in the next sections which bilinear forms appearing in the above variational problem are mapped to which operator. Note that we use the term operator since our implementation is matrix-free, i.e., we will not assemble any global matrices.

We discretize 2.9 in time using an explicit (one-step) forward Euler method. This results in the update scheme below:

$$\underline{\mathrm{u}}^{l+1} = \underline{\mathrm{u}}^l + \Delta t\, \underline{\mathrm{M}}^{-1} \left((\underline{\mathrm{F}} + \underline{\mathrm{O}})\, \underline{\mathrm{u}}^l + \underline{\mathrm{I}}(t^l) + \underline{\mathrm{G}}\, \underline{\mathrm{u}}^l \right), \tag{2.11}$$

with $l = 0, \ldots, l_{max}$ denoting the time step, t^l denoting a discrete point in time, and Δt denoting the time step size. The maximum time step depends on the simulation time according to $l_{max} = T/\Delta t$.

To ensure stability of the scheme, we have to limit the size of time step. For global time stepping schemes as ours, the upper bound for admissible time steps is usually estimated according to:

$$\Delta t < \mathrm{CFL} \max_{K \in \mathcal{T}_h} \left\{ \frac{\varrho_K}{\max_{\boldsymbol{x} \in K} |\boldsymbol{b}|} \right\}, \tag{2.12}$$

with $\mathrm{CFL} = (2\,N + 1)^{-1}$ (cf. e.g. [1]), where N denotes the polynomial order of the spatial discretization. Here, ρ_K denotes the smallest circle that can be inscribed inside the grid cell K.

Remark 2.1 For cellwise constant DG methods on Cartesian grids, the CFL condition can be interpreted the following way: *During a time step, information is only allowed to flow to the direct neighbors of a cell.*

Remark 2.2 Note that the CFL criterion imposes a severe restriction on the time step size if we want to use higher order spatial approximations ($N > 0$).

2.3 The local discrete operators

2.3.1 Preliminaries

We store all cells of the mesh in an ordered list $\{K_c\}_{1 \leq c \leq N_K}$.

Recall that we construct a global basis for $\mathcal{V}_h$ using local basis functions with compact support on each cell K_c, i.e., the basis functions attain non-zero values only in one particular cell K_c and are identical to zero elsewhere.

We denote by $\{\phi_m^c\}_{1\leq m\leq N_{\text{basis}}}$ the local basis in K_c, with N_{basis} denoting the size of the local basis. Note that for an approximation of order N, the size of the local basis is computed according to:

$$N_{\text{basis}} = 1 \times (N+1)^d. \tag{2.13}$$

There are many different ways to choose the basis functions. Usual choices are nodal basis functions (Lagrange polynomials) or modal basis functions (Legendre polynomials). The latter and certain nodal basis functions where the nodes are located at Gauss-Legendre quadrature points have a useful property: They are orthogonal in the $L^2(\Omega)$ sense. Choosing such a basis often results in very sparse operators, e.g., the mass operator is completely diagonal in this case.

We will further define $u_h^c = u_h|_{K_c}$. To derive the elements of the operators introduced in the previous section, we can apply the following replacement rule:

$$u_h^c \to \phi_n^c \qquad \text{specifies a column,} \tag{2.14}$$

$$v_h^c \to \phi_m^c \qquad \text{specifies a row,} \tag{2.15}$$

where $u_h \in \mathcal{V}_h$ refers to the ansatz function and $v_h \in \mathcal{V}_h$ to a test function. Lastly, if two cells share a face, we will always indicate the neighbor cell by the index b, and the "current"cell by the index c.

2.3.2 Mass operator

Thanks to the compact support of the local basis functions, we only have to compute the local block matrix $\underline{\mathrm{M}}^{c,c} \in \mathbb{R}^{N_{\text{basis}}\times N_{\text{basis}}}$ for each cell K_c, where

$$\mathrm{M}_{m,n}^{c,c} = \int_{K_c} \phi_m^c \, \phi_n^c \, \mathrm{d}\boldsymbol{x}, \qquad m,n = 1,\dots,N_{\text{basis}}. \tag{2.16}$$

The mass operator is thus in general block-diagonal.

2.3.3 Volume flux operator

Thanks to the compact support of the local basis functions, we only have to compute the local block matrix $\underline{\mathrm{F}}^{c,c} \in \mathbb{R}^{N_{\text{basis}}\times N_{\text{basis}}}$ for each cell K_c, where

$$\mathrm{F}_{m,n}^{c,c} = \sum_{K\in\mathcal{T}_h} \int_K \boldsymbol{F}(\phi_n^c)\cdot\nabla\phi_m^c \, \mathrm{d}\boldsymbol{x} \qquad m,n = 1,\dots,N_{\text{basis}}. \tag{2.17}$$

The volume flux operator is thus in general block-diagonal.

2.3.4 Outflow operator

Let us denote the outflow part of a face from the perspective of cell K_c by

$$e^{c,\uparrow} := \{\boldsymbol{x} \in e : (\boldsymbol{b} \cdot \boldsymbol{n}^c)(\boldsymbol{x}) > 0\}. \tag{2.18}$$

Let $e \subset \partial\Omega \cap \partial K_c$, where ∂K_c denotes the boundary of cell K_c. Thanks to the compact support of the local basis functions, we only have to compute the local matrix $\underline{\mathrm{O}}^{c,c} \in \mathbb{R}^{N_{\mathrm{basis}} \times N_{\mathrm{basis}}}$ for each cell K_c, where

$$\mathrm{O}^{c,c}_{m,n} = \int_{e^{c,\uparrow}} -(\boldsymbol{F}(\phi^c_n) \cdot \boldsymbol{n})\, \phi^c_m \,\mathrm{d}\boldsymbol{x}, \qquad m,n = 1,\dots,N_{\mathrm{basis}}. \tag{2.19}$$

The outflow operator is thus in general block-diagonal.

2.3.5 Inflow vector

Let us denote the inflow part of a face from the perspective of cell K_c by

$$e^{c,\downarrow} := \{\boldsymbol{x} \in e : (\boldsymbol{b} \cdot \boldsymbol{n}^c)(\boldsymbol{x}) < 0\} \tag{2.20}$$

Thanks to the compact support of the local basis functions, we only have to compute the local vector $\underline{\mathrm{I}}^c \in \mathbb{R}^{N_{\mathrm{basis}} \times N_{\mathrm{basis}}}$ for each cell K_c, where

$$\mathrm{I}^c_m = \int_{e^{c,\downarrow}} j(t^n)\, \phi^c_m \,\mathrm{d}\boldsymbol{x}, \qquad m,n = 1,\dots,N_{\mathrm{basis}}. \tag{2.21}$$

Remark 2.3 *Compact support* should sound familiar at this point.

2.4 The non-local discrete normal flux operator

Let the face $e \in \Gamma_h$ be part of the boundary of both cells K_c and K_b, we have to compute the block matrices $\underline{\mathrm{G}}^{c,c}_e \in \mathbb{R}^{N_{\mathrm{basis}} \times N_{\mathrm{basis}}}$, $\underline{\mathrm{G}}^{c,b} \in \mathbb{R}^{N_{\mathrm{basis}} \times N_{\mathrm{basis}}}$, $\underline{\mathrm{G}}^{b,c} \in \mathbb{R}^{N_{\mathrm{basis}} \times N_{\mathrm{basis}}}$, and $\underline{\mathrm{G}}^{b,b} \in \mathbb{R}^{N_{\mathrm{basis}} \times N_{\mathrm{basis}}}$ according to:

$$G^{c,c}_{e;m,n} = \int_e -\boldsymbol{G}^{\uparrow}(\phi^c_n, \boldsymbol{n}^c) \cdot \boldsymbol{n}^c\, \phi^c_m \,\mathrm{d}\boldsymbol{x}, \qquad m,n = 1,\dots,N_{\mathrm{basis}}, \tag{2.22}$$

$$G^{c,b}_{m,n} = \int_e \boldsymbol{G}^{\uparrow}(\phi^b_n, \boldsymbol{n}^b) \cdot \boldsymbol{n}^b\, \phi^c_m \,\mathrm{d}\boldsymbol{x}, \qquad m,n = 1,\dots,N_{\mathrm{basis}}, \tag{2.23}$$

and

$$\tag{2.24}$$

$$G^{b,b}_{e;m,n} = \int_e -\boldsymbol{G}^{\uparrow}(\phi^b_n, \boldsymbol{n}^b) \cdot \boldsymbol{n}^b\, \phi^b_m \,\mathrm{d}\boldsymbol{x}, \qquad m,n = 1,\dots,N_{\mathrm{basis}}, \tag{2.25}$$

$$G^{b,c}_{m,n} = \int_e \boldsymbol{G}^{\uparrow}(\phi^c_n, \boldsymbol{n}^c) \cdot \boldsymbol{n}^c\, \phi^b_m \,\mathrm{d}\boldsymbol{x}, \qquad m,n = 1,\dots,N_{\mathrm{basis}}. \tag{2.26}$$

Above, we have introduced

$$\boldsymbol{G}^{\uparrow}(\phi_n^+, \boldsymbol{n}) = \begin{cases} \boldsymbol{b}\,\phi_n^+ & \text{for } (\boldsymbol{b}\cdot\boldsymbol{n})(\boldsymbol{x}) \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{2.27}$$

Note that we write once per face, i.e., maximum $2\,d$ times, to the block $\underline{\mathrm{G}}^{c,c}$:

$$\underline{\mathrm{G}}^{c,c} = \sum_{\substack{e\subset\partial K \\ e\in\Gamma_h}} \underline{\mathrm{G}}_e^{c,c} = \sum_{i=1}^{2\,d} \underline{\mathrm{G}}_{e_i}^{c,c}. \tag{2.28}$$

On Cartesian grids, we can combine the contributions made by pairs of opposite neighbors of the current cell K_c by defining so-called fluctuations. Let us indicate the left, right, front, and back neighbor of cell K_c by $\mathrm{left}(c)$, $\mathrm{right}(c)$, $\mathrm{front}(c)$, and $\mathrm{back}(c)$, respectively If we write the neighbor contributions down in terms of fluctuations, we obtain:

$$\begin{aligned}
\mathrm{D}_{x;m}^{c}(u_n^{\mathrm{left}(c)}, u_n^{\mathrm{right}(c)}) &= \mathrm{G}_{m,n}^{c,\mathrm{right}(c)}\, u_n^{\mathrm{right}(c)} + \mathrm{G}_{m,n}^{c,\mathrm{left}(c)}\, u_n^{\mathrm{left}(c)} && (2.29)\\
&= \left(-\int_e \boldsymbol{G}^{\uparrow}(\phi_n^{\mathrm{right}(c)}, -\boldsymbol{n}_x)\cdot\boldsymbol{n}_x\,\phi_m^c\,\mathrm{d}\boldsymbol{x} \right) u_n^{\mathrm{right}(c)} && (2.30)\\
&\quad - \left(-\int_e \boldsymbol{G}^{\uparrow}(\phi_n^{\mathrm{left}(c)}, \boldsymbol{n}_x)\cdot\boldsymbol{n}_x\,\phi_m^c\,\mathrm{d}\boldsymbol{x} \right) u_n^{\mathrm{left}(c)}, && (2.31)\\
\mathrm{D}_{y;m}^{c}(u_n^{\mathrm{front}(c)}, u_n^{\mathrm{back}(c)}) &= \mathrm{G}_{m,n}^{c,\mathrm{back}(c)}\, u_n^{\mathrm{back}(c)} + \mathrm{G}_{m,n}^{c,\mathrm{front}(c)}\, u_n^{\mathrm{front}(c)} && (2.32)\\
&= \left(-\int_e \boldsymbol{G}^{\uparrow}(\phi_n^{\mathrm{back}(c)}, -\boldsymbol{n}_y)\cdot\boldsymbol{n}_y\,\phi_m^c\,\mathrm{d}\boldsymbol{x} \right) u_n^{\mathrm{back}(c)} && (2.33)\\
&\quad - \left(-\int_e \boldsymbol{G}^{\uparrow}(\phi_n^{\mathrm{front}(c)}, \boldsymbol{n}_y)\cdot\boldsymbol{n}_y\,\phi_m^c\,\mathrm{d}\boldsymbol{x} \right) u_n^{\mathrm{front}(c)}, && (2.34)
\end{aligned}$$

with $\boldsymbol{n}_x = (1,0)^{\mathrm{T}}$, and $\boldsymbol{n}_y = (0,1)^{\mathrm{T}}$.

2.5 Cellwise constant basis functions (in space)

In the case of cellwise constant basis functions, we can make the following simplifications. Let $K_c \in \mathcal{T}_h$; the mass operator is diagonal and has one element per cell:

$$\mathrm{M}^{c,c} = \int_{K_c} 1\,\mathrm{d}\boldsymbol{x} = |K_c| \Leftrightarrow (\mathrm{M}^{-1})^{c,c} = 1/|K_c|, \tag{2.35}$$

The volume fluxes are identical to zero, i.e.,

$$\underline{\mathrm{F}} = 0, \tag{2.36}$$

since the gradient of the cellwise constant basis functions is identical to zero. The outflow operator is diagonal and has one element per cell:

$$\mathrm{O}^{c,c} = \int_{e^{c,+}} \boldsymbol{F}(1)\cdot\boldsymbol{n}\;\mathrm{d}\boldsymbol{x} = \int_{e^{c,+}} \boldsymbol{b}\cdot\boldsymbol{n}\;\mathrm{d}\boldsymbol{x}, \tag{2.37}$$

and the inflow vector has one element in each cell:

$$\mathrm{I}^c = \int_{e^{c,\downarrow}} j(t^n)\,\mathrm{d}\boldsymbol{x}, \tag{2.38}$$

We have the following contributions to the normal flux operator per face:

$$\mathrm{G}_e^{c,c} = \int_e -\boldsymbol{G}^{\uparrow}(1,\boldsymbol{n}^c)\,\boldsymbol{n}^c\,\mathrm{d}\boldsymbol{x}, \tag{2.39}$$

$$\mathrm{G}^{c,b} = \int_e \boldsymbol{G}^{\uparrow}(1,\boldsymbol{n}^b)\,\boldsymbol{n}^b\,\mathrm{d}\boldsymbol{x}, \tag{2.40}$$

and

$$\tag{2.41}$$

$$\mathrm{G}_e^{b,b} = \int_e -\boldsymbol{G}^{\uparrow}(1,\boldsymbol{n}^b)\,\boldsymbol{n}^b\,\mathrm{d}\boldsymbol{x}, \tag{2.42}$$

$$\mathrm{G}^{b,c} = \int_e \boldsymbol{G}^{\uparrow}(1,\boldsymbol{n}^c)\,\boldsymbol{n}^c\,\mathrm{d}\boldsymbol{x}. \tag{2.43}$$

The fluctuations simplify to

$$\mathrm{D}_x^c(u^{\mathrm{left}(c)}, u^{\mathrm{right}(c)}) = \mathrm{G}^{c,\mathrm{right}(c)}\,u^{\mathrm{right}(c)} + \mathrm{G}^{c,\mathrm{left}(c)}\,u^{\mathrm{left}(c)} \tag{2.44}$$

$$= \left(-\int_e \boldsymbol{G}^{\uparrow}(1,-\boldsymbol{n}_x)\cdot\boldsymbol{n}_x\,\mathrm{d}\boldsymbol{x}\right) u^{\mathrm{right}(c)} \tag{2.45}$$

$$-\left(-\int_e \boldsymbol{G}^{\uparrow}(1,\boldsymbol{n}_x)\cdot\boldsymbol{n}_x\,\mathrm{d}\boldsymbol{x}\right) u^{\mathrm{left}(c)}, \tag{2.46}$$

$$\mathrm{D}_y^c(u^{\mathrm{front}(c)}, u^{\mathrm{back}(c)}) = \mathrm{G}^{c,\mathrm{back}(c)}\,u^{\mathrm{back}(c)} + \mathrm{G}^{c,\mathrm{front}(c)}\,u^{\mathrm{front}(c)} \tag{2.47}$$

$$= \left(-\int_e \boldsymbol{G}^{\uparrow}(1,-\boldsymbol{n}_y)\cdot\boldsymbol{n}_y\,\mathrm{d}\boldsymbol{x}\right) u_n^{\mathrm{back}(c)} \tag{2.48}$$

$$-\left(-\int_e \boldsymbol{G}^{\uparrow}(1,\boldsymbol{n}_y)\cdot\boldsymbol{n}_y\,\mathrm{d}\boldsymbol{x}\right) u_n^{\mathrm{front}(c)}, \tag{2.49}$$

2.5.1 Cell updates

In each time step, we thus update the cell according to:

$$u^{c;n+1} = u^{c;n} + \Delta u^{c;n} \tag{2.50}$$

with

$$\Delta u^{c;n} = \Delta t\,|K|^{-1}\left(\mathrm{O}^{\mathrm{c,c}}u^{c;n} + \sum_{i=1}^{2\,d}\mathrm{G}_{e_i}^{c,c}u^{c;n}\right) \tag{2.51}$$

$$+\,\Delta t\,|K|^{-1}\,\mathrm{D}_x^c(u^{\mathrm{left}(c);n}, u^{\mathrm{right}(c);n}) \tag{2.52}$$

$$+\,\Delta t\,|K|^{-1}\,\mathrm{D}_y^c(u^{\mathrm{front}(c);n}, u^{\mathrm{back}(c);n}). \tag{2.53}$$

2.5.2 Cell updates

2.6 Mappings

In this section, we introduce mappings between the mesh elements $K \in \mathcal{T}_h$ and a reference element. These mappings allow us to treat integrals over arbitrarily shaped mesh elements in an uniform manner. To ease the presentation, we only consider the two-dimensional case.

Let us define the *reference element* $\hat{K} = [-1, 1]^2$ and let us denote by $\hat{\boldsymbol{x}} = (\hat{x}, \hat{y})^{\mathrm{T}}$ a point belonging to $\hat{K}$. Let $K \in \mathcal{T}_h$ denote a quadrilateral element and let its corners $\boldsymbol{P}_m$, $m = 1, 2, 3, 4$, be ordered as shown in figure 2.1.

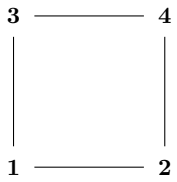

Figure 2.1: Node ordering

For simplicity, we assume that the quadrilateral K has straight edges. Then, we can express K according to

$$K = \boldsymbol{F}_K(\hat{K}),$$

where we define the mapping $\boldsymbol{F}_K : \hat{K} \to K$ by

$$\boldsymbol{x} = \boldsymbol{F}_K(\hat{\boldsymbol{x}}) = \begin{pmatrix} F_{K,x}(\hat{\boldsymbol{x}}) \\ F_{K,y}(\hat{\boldsymbol{x}}) \end{pmatrix} = \sum_{m=1}^{4} \boldsymbol{P}_m \, s_m(\hat{\boldsymbol{x}}), \qquad \hat{\boldsymbol{x}} \in \hat{K}, \boldsymbol{x} \in K. \tag{2.54}$$

The *shape functions* $s_m(\hat{\boldsymbol{x}})$, $m = 1, 2, 3, 4$, are defined according to

$$s_1(\hat{\boldsymbol{x}}) = \frac{1}{4}(1 - \hat{x})\,(1 - \hat{y}), \qquad s_3(\hat{\boldsymbol{x}}) = \frac{1}{4}(1 - \hat{x})\,(1 + \hat{y}), \tag{2.55}$$

$$s_2(\hat{\boldsymbol{x}}) = \frac{1}{4}(1 + \hat{x})\,(1 - \hat{y}), \qquad s_4(\hat{\boldsymbol{x}}) = \frac{1}{4}(1 + \hat{x})\,(1 + \hat{y}), \tag{2.56}$$

and have the property

$$s_m(\hat{\boldsymbol{P}}_n) = \begin{cases} 1, & m = n, \\ 0, & m \neq n, \end{cases} \tag{2.57}$$

where $\hat{\boldsymbol{P}}_1 = (-1, -1)^{\mathrm{T}}$, $\hat{\boldsymbol{P}}_2 = (1, -1)^{\mathrm{T}}$, $\hat{\boldsymbol{P}}_3 = (-1, 1)^{\mathrm{T}}$, and $\hat{\boldsymbol{P}}_4 = (1, 1)^{\mathrm{T}}$ denote the corner points of the reference element. Note that the indexing of these corner points is in compliance with the node ordering depicted in figure 2.1.

Note that in case of Cartesian elements, we can simplify the above mapping according to:

$$\boldsymbol{x} = \boldsymbol{F}_K(\hat{\boldsymbol{x}}) = \boldsymbol{P}_{\text{center}} + \frac{1}{2}\begin{pmatrix} \Delta x & 0 \\ 0 & \Delta y \end{pmatrix}\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix}, \tag{2.58}$$

where Δx and Δy denote the dimensions of the element in the respective coordinate direction. Note that $\boldsymbol{P}_{\text{center}}$ is the center of the element.

Assuming it exists, we denote the inverse mapping by $\boldsymbol{F}_K^{-1} : K \to \hat{K}$, which in general cannot be constructed by polynomial shape functions. Next, let us define the Jacobian matrix

$$\mathrm{D}\boldsymbol{F}_K(\hat{\boldsymbol{x}}) = \begin{pmatrix} \frac{\partial F_{K,x}}{\partial \hat{x}} & \frac{\partial F_{K,x}}{\partial \hat{y}} \\ \frac{\partial F_{K,y}}{\partial \hat{x}} & \frac{\partial F_{K,y}}{\partial \hat{y}} \end{pmatrix}(\hat{\boldsymbol{x}}),$$

and its determinant

$$J_K(\hat{\boldsymbol{x}}) = \det\left(\mathrm{D}\boldsymbol{F}_K\right)(\hat{\boldsymbol{x}}) = \begin{vmatrix} \frac{\partial F_{K,x}}{\partial \hat{x}} & \frac{\partial F_{K,x}}{\partial \hat{y}} \\ \frac{\partial F_{K,y}}{\partial \hat{x}} & \frac{\partial F_{K,y}}{\partial \hat{y}} \end{vmatrix}(\hat{\boldsymbol{x}}).$$

Assume that $\boldsymbol{F}_K$ is invertible, we accordingly define $\mathrm{D}\boldsymbol{F}_K^{-1}(\boldsymbol{x})$ and it follows from the inverse function theorem that

$$\mathrm{D}\boldsymbol{F}_K^{-1}(\boldsymbol{x}) = \left(\mathrm{D}\boldsymbol{F}_K\right)^{-1}(\hat{\boldsymbol{x}}),$$

for $\hat{\boldsymbol{x}} \in \hat{K}$ such that $\boldsymbol{x} = \boldsymbol{F}_K(\hat{\boldsymbol{x}}) \in K$. We note that if the element K is affine, it holds that

$$\mathrm{D}\boldsymbol{F}_K(\hat{\boldsymbol{x}}) = \mathrm{D}\boldsymbol{F}_K = \text{const.} \qquad \Leftrightarrow \qquad J_K(\hat{\boldsymbol{x}}) = J_K = \text{const.},$$

and the mapping has the form

$$\boldsymbol{F}_K(\hat{\boldsymbol{x}}) = \mathrm{D}\boldsymbol{F}_K \cdot \hat{\boldsymbol{x}} + \boldsymbol{P}_1.$$

In this case, there exists an inverse mapping if $\mathrm{D}\boldsymbol{F}_K$ is regular, i.e. $J_K \neq 0$.

According to an invertible mapping $\boldsymbol{F}_K$, we can further derive the following identity. Let $\hat{f}$ be a sufficiently regular function on the reference element $\hat{K}$, and let f be sufficiently regular on K and such that

$$f(\boldsymbol{x}) = (\hat{f} \circ \boldsymbol{F}_K^{-1})(\boldsymbol{x}) = \hat{f}(\hat{\boldsymbol{x}}), \qquad \hat{\boldsymbol{x}} \in \hat{K},\ \boldsymbol{x} = \boldsymbol{F}_K(\hat{\boldsymbol{x}}) \in K. \tag{2.59}$$

Then, it holds that

$$\nabla f(\boldsymbol{x}) = \left(\mathrm{D}\boldsymbol{F}_K^{-\mathrm{T}} \cdot \hat{\nabla}\, \hat{f}\right)(\hat{\boldsymbol{x}}),$$

where

$$\hat{\nabla} = \left(\frac{\partial}{\partial \hat{x}}, \frac{\partial}{\partial \hat{y}}\right)^{\mathrm{T}}$$

denotes the gradient with respect to the reference coordinates. This follows from (2.59) and using the chain rule. A similar identity can be derived for vector-valued functions. We can further express volume integrals over the element K with respect to the reference coordinates and the invertible mapping $\boldsymbol{F}_K$. Let $\hat{f}$ and f be defined as in (2.59), we have

$$\int_K f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_{\hat{K}} \hat{f}(\hat{\boldsymbol{x}})\,|J_K(\hat{\boldsymbol{x}})|\,\mathrm{d}\hat{\boldsymbol{x}} = \int_{-1}^{1}\int_{-1}^{1} \hat{f}(\hat{x},\hat{y})\,|J_K(\hat{x},\hat{y})|\,\mathrm{d}\hat{\boldsymbol{x}}\,\mathrm{d}\hat{y}.$$

Further, if we define the faces of the reference element according to

$$\begin{aligned} \hat{e}_1 &= \{\hat{\boldsymbol{x}} \in \hat{K} : \hat{x} = -1\}, \qquad & \hat{e}_3 &= \{\hat{\boldsymbol{x}} \in \hat{K} : \hat{y} = -1\}, \\ \hat{e}_2 &= \{\hat{\boldsymbol{x}} \in \hat{K} : \hat{x} = +1\}, \qquad & \hat{e}_4 &= \{\hat{\boldsymbol{x}} \in \hat{K} : \hat{y} = +1\}, \end{aligned}$$

we can define the four faces belonging to the boundary of K by

$$e_1 = \boldsymbol{F}_K(\hat{e}_1), \qquad e_2 = \boldsymbol{F}_K(\hat{e}_2), \qquad e_3 = \boldsymbol{F}_K(\hat{e}_3), \qquad e_4 = \boldsymbol{F}_K(\hat{e}_4).$$

Now we can express integrals over faces belonging to the boundary of K with respect to the reference coordinates and the mapping $\boldsymbol{F}_K$. Let $\hat{f}$ and f be defined as in (2.59), we have

$$\begin{aligned} \int_{e_1} f(\boldsymbol{x})\,\mathrm{d}s &= \int_{-1}^{1} \hat{f}(-1,\hat{y}) \left|\frac{d\boldsymbol{F}_K(-1,\hat{y})}{\mathrm{d}\hat{y}}\right| \mathrm{d}\hat{y}, \\ \int_{e_2} f(\boldsymbol{x})\,\mathrm{d}s &= \int_{-1}^{1} \hat{f}(+1,\hat{y}) \left|\frac{d\boldsymbol{F}_K(+1,\hat{y})}{\mathrm{d}\hat{y}}\right| \mathrm{d}\hat{y}, \\ \int_{e_3} f(\boldsymbol{x})\,\mathrm{d}s &= \int_{-1}^{1} \hat{f}(\hat{x},-1) \left|\frac{d\boldsymbol{F}_K(\hat{x},-1)}{\mathrm{d}\hat{x}}\right| \mathrm{d}\hat{x}, \\ \int_{e_4} f(\boldsymbol{x})\,\mathrm{d}s &= \int_{-1}^{1} \hat{f}(\hat{x},+1) \left|\frac{d\boldsymbol{F}_K(\hat{x},-1)}{\mathrm{d}\hat{x}}\right| \mathrm{d}\hat{x}. \end{aligned}$$

Please note that all these concepts extend to the 3-d case and it is also possible to use higher order shape functions instead of the linear Lagrange polynomials that we used in (2.55) and (2.56); by linear we mean linear in each reference coordinate direction.

3 Implementation of the cell-wise constant discretization

3.1 The four phases of DG methods

We have decided to use a DG method for the spatial discretization and an explicit scheme for the temporal discretization. For this kind of methods, the update process that is performed in each time step can be divided into the following four phases:

1. Compute cell volume data (cell-local)
2. Extrapolate face data from cell volume data (cell-local)
3. Exchange face data (non-local)
4. Apply cell updates (cell-local)

Phase one includes the computation of volume fluxes, right-hand side vectors (and of the space-time predictors in case of the ADER-DG method with local time stepping). In the second phase, we compute the contribution of each cell to the (numerical) normal fluxes that will be exchanged in phase three of the scheme. Only phase three requires communication between processors if the code is executed in a parallel environment and should thus be considered separately from the other phases. In this phase, the normal fluxes are computed according to the local contributions of neighboring cells.

Finally, note that phase one and two can be performed in the same grid traversal.

3.2 A more cache-aware ordering of the four phases

The ordering of the update phases introduced in the previous section has one disadvantage: We have to traverse the whole space-tree three times, i.e., we have to load the data of each cell at least three times into the cache of the processor "owning" that part of the grid the cell belongs to. If we initialize the cell updates to be zero initially, we can however use the following slightly modified ordering of the four phases:

1. Apply cell updates (cell-local)
2. Compute cell volume data (cell-local)
3. Extrapolate face data from cell volume data (cell-local)
4. Exchange face data (non-local)

The above ordering requires only two grid traversals since we can combine the first three phases and process them in one single traversal.

3.3 Lookup of the (first) cell description index of the neighbors

The multilinkedcell toolbox supplies us with a enumeration of cells around the current cell. In 2-d, we are supplied with the enumeration depicted in figure 3.1. In 3-d, we are supplied with the enumeration depicted in figure 3.2. Note that the enumerations also incldue the current cell.

For DG methods, the neighbors diagonal to the current cell are not of relevance. The relevant neighbors are the cells that share a complete face with the current cell. The first CellDescription index of each relevant neighbor is stored inside of an array, i.e.,

$$\underline{\mathrm{N}}_2 = \{\mathrm{Index}(3), \mathrm{Index}(5), \mathrm{Index}(1), \mathrm{Index}(7)\} \tag{3.1}$$

$$\underline{\mathrm{N}}_3 = \{\mathrm{Index}(12), \mathrm{Index}(14), \mathrm{Index}(10), \mathrm{Index}(16), \mathrm{Index}(4), \mathrm{Index}(22)\}, \tag{3.2}$$

where $\underline{\mathrm{N}}_2$ is the corresponding array for the 2-d case and $\underline{\mathrm{N}}_3$ for the 3-d case.

6	**7**	8
3	**$\underline{4}$**	**5**
0	**1**	2

Figure 3.1: 2-d neighbor enumeration. The current cell (in the center) is underlined.

3.4 Grid traversal

Peano does not store all cells as objects in a large data structure but does only initialize $1+2^d$ cell objects. One object is necessary for the storage of the parent cell (coarse grid cell), the remaining are used to store its children (fineGridCell).

During the traversal, parameters like position and dimensions of the a parent cell and its children are computed/loaded into this data structure. This idea is named the *flyweight pattern*; cf. [**?**]. Note that the root cell and its vertices are on level one (1) of the grid, its children belong to level two (2).

Note further that after refinement, the final "grid" is defined by the fine grid vertices/cells not by the coarse grid vertices/cells in our implementation!

Once:

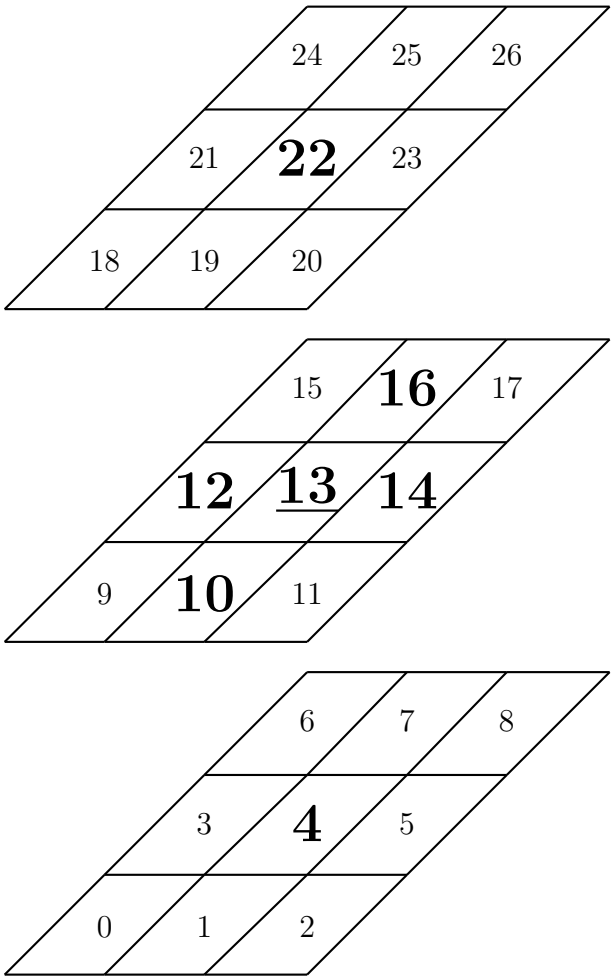

Figure 3.2: 3-d neighbor enumeration. The current cell (in the center) is underlined.

1. InitCells: beginIteration(): allocate CellDescription and data heap data memory
2. InitCellData: initalize Update and Solution

Every traversal:

1. UpdateCellData:
2. ComputeCellData:
3. ExtrapolateFaceData:
4. ExchangeFaceData:

3.5 Next steps

1. AMR: Introduce parent CellDescriptions that do not necessarily hold data but refer to the CellDescription indices of their child processes
2. Plotting for high-order methods
3. User templates for plotting so that we do not have to generate them everytime.

4 Examples

4.1 Step-12

We consider a problem setting similar to the one in The step-12 tutorial program: Given is a domain $\Omega = (0,1)^2$; inside the domain and on its boundary acts a circular counter-clockwise flow field

$$\boldsymbol{b} = \frac{1}{\sqrt{x^2+y^2}}(-y, x) \tag{4.1}$$

We have the following inflow boundary conditions:

$$j(t) = \begin{cases} 1 & \text{for } \boldsymbol{x} \in [0, 0.5] \times \{0\} \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

Note that according to the definition of the flow field, we only have one inflow boundary, the boundary at $y = 0$.

Bibliography

[1] B. Cockburn, G. E. Karniadakis, and C.-W. Shu, *Discontinuous Galerkin Methods. Lect. Notes Comput. Sci. Eng.*, Springer, 2000.

[2] R. Hartmann, *Numerical analysis of higher order discontinuous Galerkin finite element methods*, in VKI LS 2008-08: CFD - ADIGMA course on very high order discretization methods, Oct. 13-17, 2008, H. Deconinck, ed., Von Karman Institute for Fluid Dynamics, Rhode Saint Genèse, Belgium, 2008.