MATH 676

–

Finite element methods in scientific computing

Wolfgang Bangerth, Texas A&M University

Lecture 31:

First-order hyperbolic systems

First-order hyperbolic systems

A general first order hyperbolic system has the form

$$\frac{\partial u}{\partial t} + \nabla \cdot F(u) = f \qquad u(x,0) = u_0(x)$$

with certain conditions on *F(u)*.

The simplest time-dependent version is

$$\frac{\partial u}{\partial t} + \nabla \cdot (\vec{w} u) = f \qquad u(x,0) = u_0(x)$$

which is often written as

$$\frac{\partial u}{\partial t} + \vec{w} \cdot \nabla u = f \qquad u(x,0) = u_0(x)$$

if *w* is divergence-free, e.g. an incompressible flow field.

First-order hyperbolic systems

One can rewrite

$$\frac{\partial u}{\partial t} + \vec{w}\cdot\nabla u = f \qquad u(x,0)=u_0(x)$$

as

$$\begin{pmatrix} 1 \\ w_1 \\ \vdots \\ w_n \end{pmatrix} \cdot \begin{pmatrix} \partial_t \\ \partial_{x_1} \\ \vdots \\ \partial_{x_n} \end{pmatrix} u = f \qquad u(x,0)=u_0(x)$$

In other words, the simplest first-order hyperbolic system is in fact the stationary equation

$$\vec{w}\cdot\nabla u = f \qquad + \text{boundary conditions}$$

where we have just re-interpreted the coordinates and w.

How to interpret boundary conditions

The simplest transport problem in conservative formulation:

$$\nabla\cdot(\vec{w}\,u) = f \qquad \text{+ boundary conditions}$$

Here, the solution is transported along the direction given by the vector field $\vec{w}$. Imagine: Pollutant transport by wind!

Thus, we transport things *into* the domain where

$$\Gamma_{\text{in}} = \{x\in\partial\Omega:\ \vec{n}\cdot\vec{w}<0\}$$

Boundary conditions for *u* can only be posed on this *inflow boundary*.

Discretizing the transport equation

Weak formulation:

Start with the conservative formulation:

$$\nabla\cdot(\vec{w}u) = f \qquad\qquad u = g \quad \text{on } \Gamma_{\text{in}}$$

Multiply with a test function:

$$\left(\varphi, \nabla\cdot(\vec{w}u)\right) = \left(\varphi, f\right)$$

Integrate by parts:

$$-\left(\nabla\varphi, \vec{w}u\right)_\Omega + \left(\varphi, \vec{n}\cdot\vec{w}u\right)_{\partial\Omega} = \left(\varphi, f\right)$$

Then use boundary conditions:

$$-\left(\nabla\varphi, \vec{w}u\right)_\Omega + \left(\varphi, \vec{n}\cdot\vec{w}u\right)_{\partial\Omega\setminus\Gamma_{\text{in}}} = \left(\varphi, f\right) - \left(\varphi, \vec{n}\cdot\vec{w}g\right)_{\Gamma_{\text{in}}}$$

A naïve discretization

A naïve approach:

Using the weak formulation:

$$-\left(\nabla\varphi,\vec{w}u\right)_{\Omega}+\left(\varphi,\vec{n}\cdot\vec{w}u\right)_{\partial\Omega\setminus\Gamma_{\text{in}}}=\left(\varphi,f\right)-\left(\varphi,\vec{n}\cdot\vec{w}g\right)_{\Gamma_{\text{in}}}\qquad\forall\varphi$$

Restrict solution u and the test functions to finite element spaces (e.g. Lagrange elements):

$$-\left(\nabla\varphi_h,\vec{w}u_h\right)_{\Omega}+\left(\varphi_h,\vec{n}\cdot\vec{w}u_h\right)_{\partial\Omega\setminus\Gamma_{\text{in}}}=\left(\varphi_h,f\right)-\left(\varphi_h,\vec{n}\cdot\vec{w}g\right)_{\Gamma_{\text{in}}}\qquad\forall\varphi_h$$

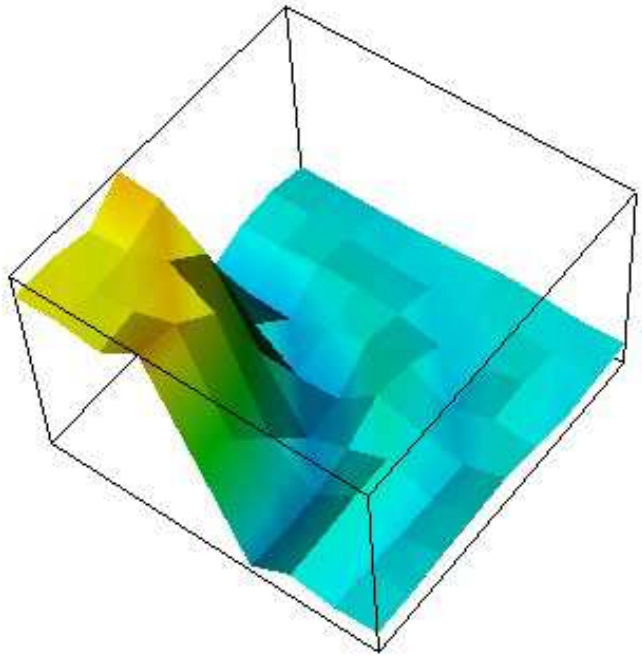
Problems:

- The discrete solution will not converge
- The discrete solution is dominated by oscillations
- The oscillations get worse under mesh refinement.

A naïve discretization

Example (step-12):

Solution computed using a continuous finite element.

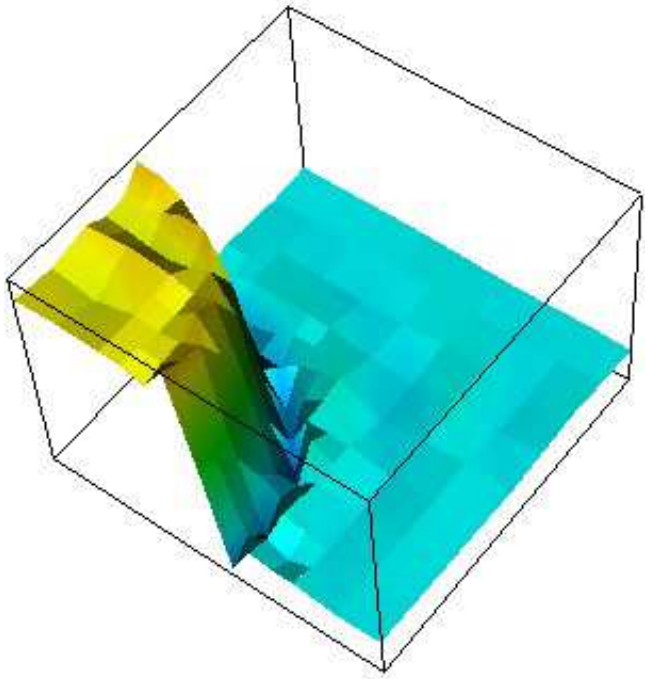


Coarse mesh

A naïve discretization

Example (step-12):

Solution computed using a continuous finite element.

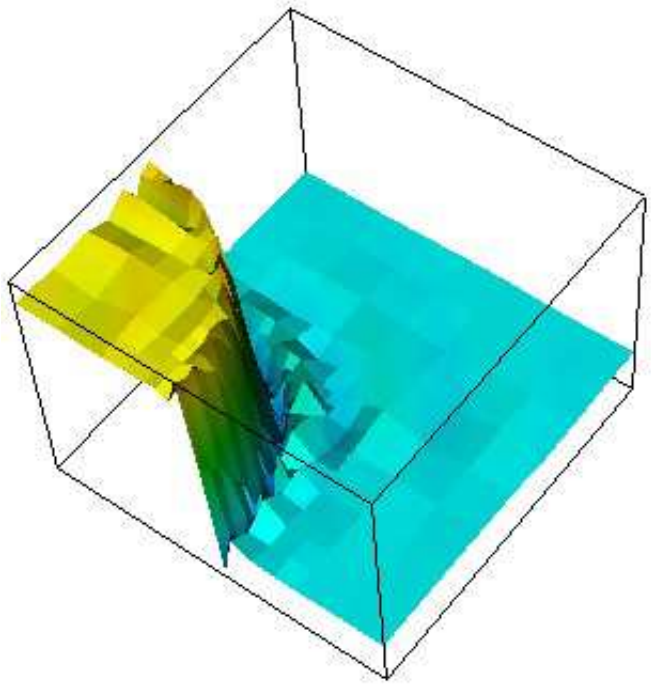


One adaptive refinement step.

A naïve discretization

Example (step-12):

Solution computed using a continuous finite element.

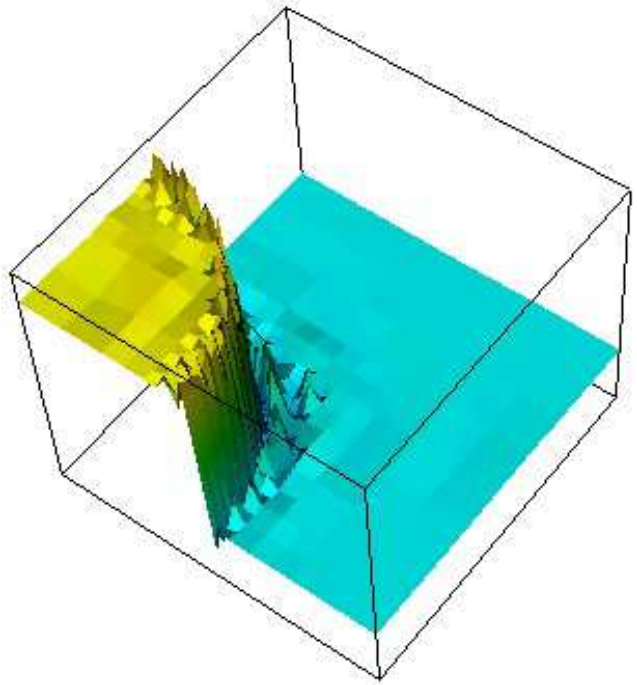


Two adaptive refinement steps.

A naïve discretization

Example (step-12):

Solution computed using a continuous finite element.

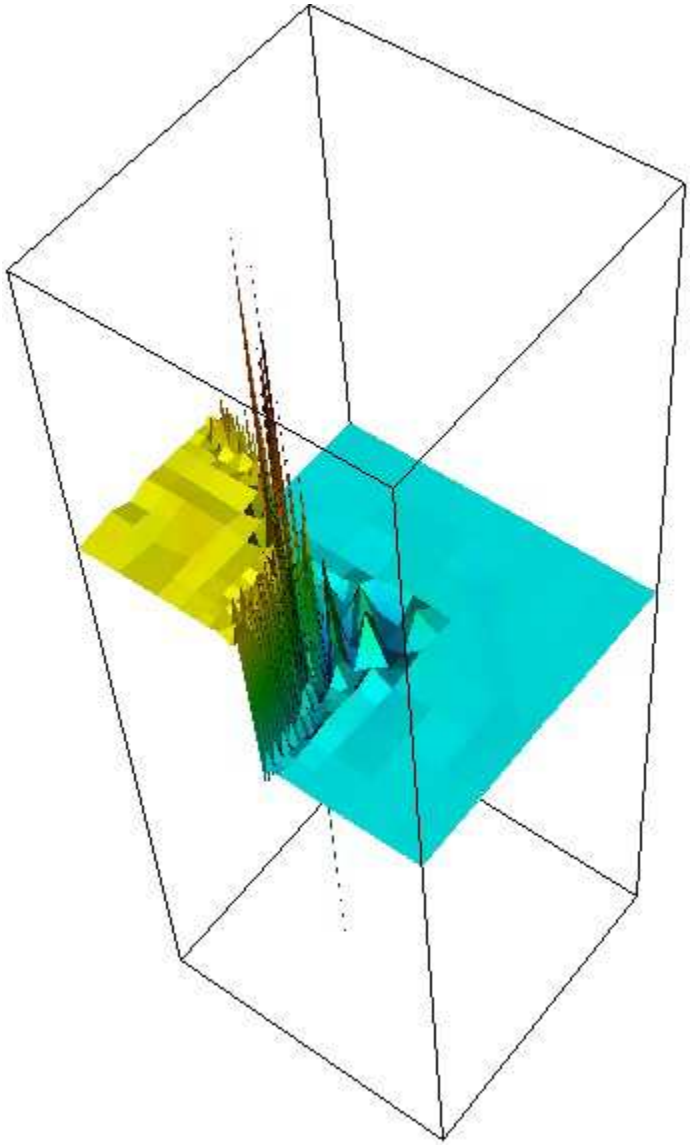


Three adaptive refinement steps.

A naïve discretization

Example (step-12):

Solution computed using a continuous finite element.

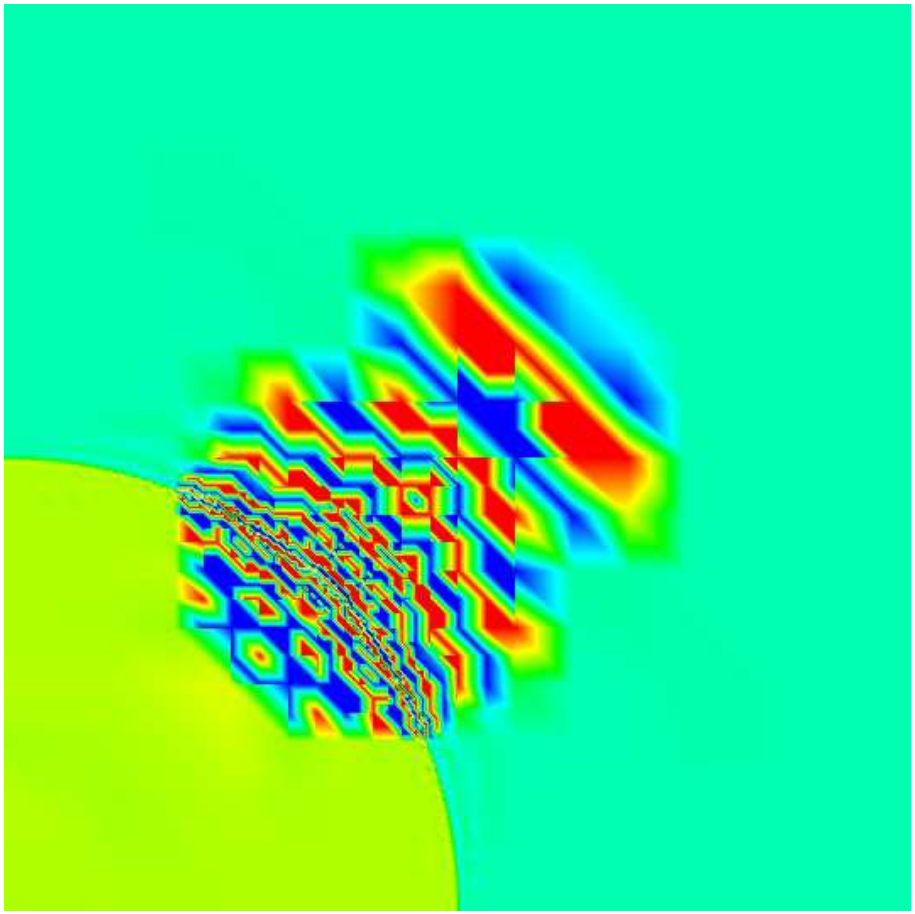


Four adaptive refinement steps.

A naïve discretization

Example (step-12):

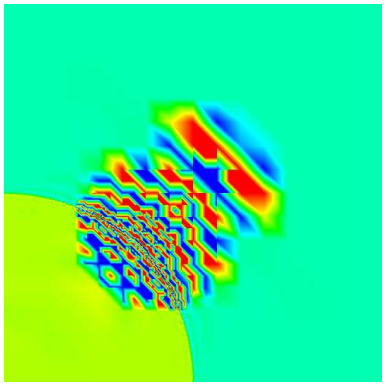
Solution computed using a continuous finite element.



Five adaptive refinement steps. Solution can't be plotted in 3d any more!

Solutions of hyperbolic equations

What are the causes for this?



Basically, because the transport equation itself is not stable:

- Hyperbolic equations typically have infinitely many solutions
- Only one of those is physically correct
- The *physically correct solution* is the limit solution of equations perturbed by adding a diffusion term

Solutions of hyperbolic equations

Example: Consider the 1d eikonal equation, a hyperbolic problem:

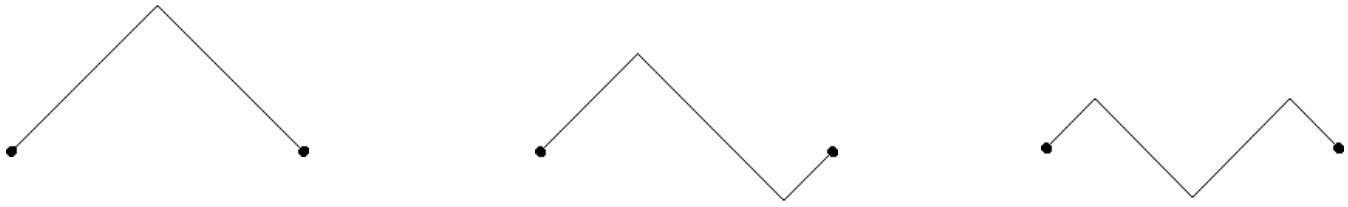
$$|u'(x)| = 1, \qquad u(0)=u(1)=0$$

Its solution is the *distance function* from the boundary.

But, in fact, any continuous function with

$$u'(x)=\pm 1$$

is a solution. For example:

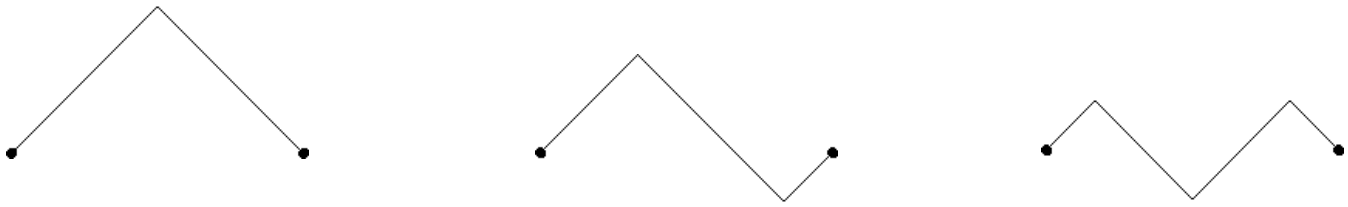


Solutions of hyperbolic equations

Example: Consider the 1d eikonal equation, a hyperbolic problem:

$$|u'(x)| = 1, \qquad u(0)=u(1)=0$$

Question: Which of these is the solution we *want*?



Answer 1: From physical considerations, we almost always want a solution that satisfies a certain *entropy condition* (e.g., the *Rankine-Hugoniot* conditions).

Answer 2: Entropy solution = viscosity limit.

Solutions of hyperbolic equations

Example: Consider the 1d eikonal equation, a hyperbolic problem:

$$|u'(x)| = 1, \qquad u(0)=u(1)=0$$

Answer 2: The solution we are looking for is the viscosity limit, i.e., the limit of the solutions of

$$|u_\varepsilon'(x)| - \varepsilon \Delta u_\varepsilon = 1, \qquad u_\varepsilon(0)=u_\varepsilon(1)=0$$

as $\varepsilon \to 0$.

"Theorem" (Lax, Oleinik, Kruzhkov, Godunov, late 1950s-1970): The viscosity solution is unique. It also satisfies the physical entropy conditions.

Solutions of hyperbolic equations

Example: Consider the 1d eikonal equation, a hyperbolic problem:

$$|u'(x)| = 1, \qquad u(0)=u(1)=0$$

Answer 2: The solution we are looking for is the viscosity limit, i.e., the limit of the solutions of

$$|u_\varepsilon'(x)| - \varepsilon \Delta u_\varepsilon = 1, \qquad u_\varepsilon(0)=u_\varepsilon(1)=0$$

as $\varepsilon \to 0$.

Next question: Which of the many solutions does the finite element solution converge to?

Answer: This is difficult to establish for any given scheme!

What to do?

To numerically approximate the entropy solution of

$$\nabla \cdot F(u) = 0, \qquad u = g \text{ on } \Gamma_{\text{in}}$$

there are essentially two approaches:

- Standard discretizations for a modified equation:
  - artificial viscosity
  - streamline upwind Petrov-Galerkin (SUPG)
- Special discretizations for these kinds of equations:
  - Finite volumes
  - Discontinuous Galerkin methods

Note 1: All of these methods add some diffusion, explicitly or implicitly.

Note 2: Thus, they approximate the viscosity solution!

What to do?

Let us look at some of these methods using the simplest model case, the *transport equation*:

$$\nabla\cdot\left(\vec{w}\,u\right) = 0, \qquad u=g \text{ on } \Gamma_{\text{in}}$$

Artificial viscosity

To numerically approximate the entropy solution of

$$\nabla\cdot(\vec{w}\,u) \;=\; 0, \qquad\qquad u=g \text{ on } \Gamma_{\text{in}}$$

the *artificial viscosity method* applies a standard discretization to the equation

$$\nabla\cdot(\vec{w}\,u)-\nu\,\Delta\,u \;=\; 0, \qquad\qquad u=g \text{ on } \Gamma_{\text{in}}$$

where $\nu(x) \;=\; \beta \;\|\vec{w}(x)\|\; h(x)$

Note 1: If β is large enough, then this discretization is stable.

Note 2: Since $\nu(x)\to 0$ as $h(x)\to 0$ this method imitates the viscosity limit.

Residual-based artificial viscosity

An alternative to the standard choice

$$\nu(x) = \beta \ \|\vec{w}(x)\| \ h(x)$$

is to limit adding viscosity where necessary, e.g. as in

$$\nu(x) = \beta \ \|\vec{w}(x)\| \ h(x) \ \min\left\{1, \frac{\|r(x)\|}{c_R}\right\}$$

where the residual *r(x)* indicates how well u_h satisfies the equation (c_R is a normalization constant).

Note: This is what we use in step-31, step-32, step-43.

Streamline upwind Petrov-Galerkin (SUPG)

To numerically approximate the entropy solution of

$$\nabla\cdot(\vec{w}\,u) = 0, \qquad u=g \text{ on } \Gamma_{\text{in}}$$

the SUPG method chooses test functions different from the trial functions. Namely:

$$\left(\varphi+ \tau\nabla\cdot(\vec{w}\,\varphi), \nabla\cdot(\vec{w}\,u)\right) = 0 \qquad \forall\varphi$$

Then do integration by parts as always.

Note 1: As before, we will use $\tau(x)\to 0$ as $h(x)\to 0$

Streamline upwind Petrov-Galerkin (SUPG)

To numerically approximate the entropy solution of

$$\nabla\cdot(\vec{w}u) = 0, \qquad u=g \text{ on } \Gamma_{\text{in}}$$

the SUPG method chooses test functions different from the trial functions. Namely:

$$\left(\varphi+ \tau\nabla\cdot(\vec{w}\varphi), \nabla\cdot(\vec{w}u)\right) = 0 \qquad \forall\varphi$$

Note 2: The strong form of this is

$$\nabla\cdot(\vec{w}u)-\tau[\vec{w}\cdot\nabla]\nabla\cdot(\vec{w}u) = 0$$

which in turn equals

$$\nabla\cdot(\vec{w}u)-\tau\nabla\cdot[\vec{w}\otimes\vec{w}]\nabla u = 0$$

That is, we only add diffusion in streamline direction!

Artificial viscosity vs. SUPG

In general:

- SUPG only adds viscosity in the direction where necessary
- SUPG is difficult to generalize to the time-dependent case

- Artificial viscosity is indiscriminate, but can be made much better using residual-based information

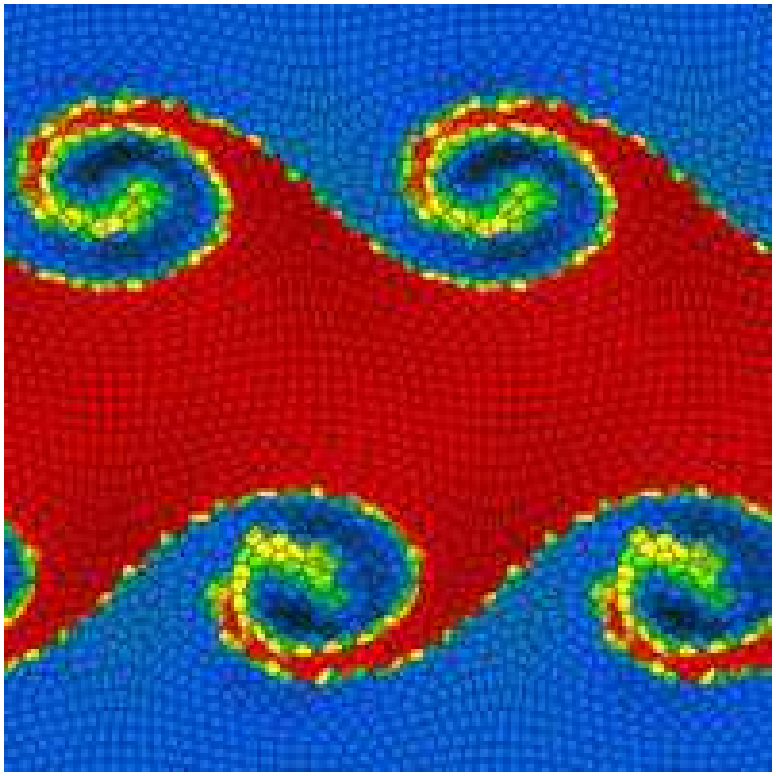
- Both methods require tuning dimension-less parameters

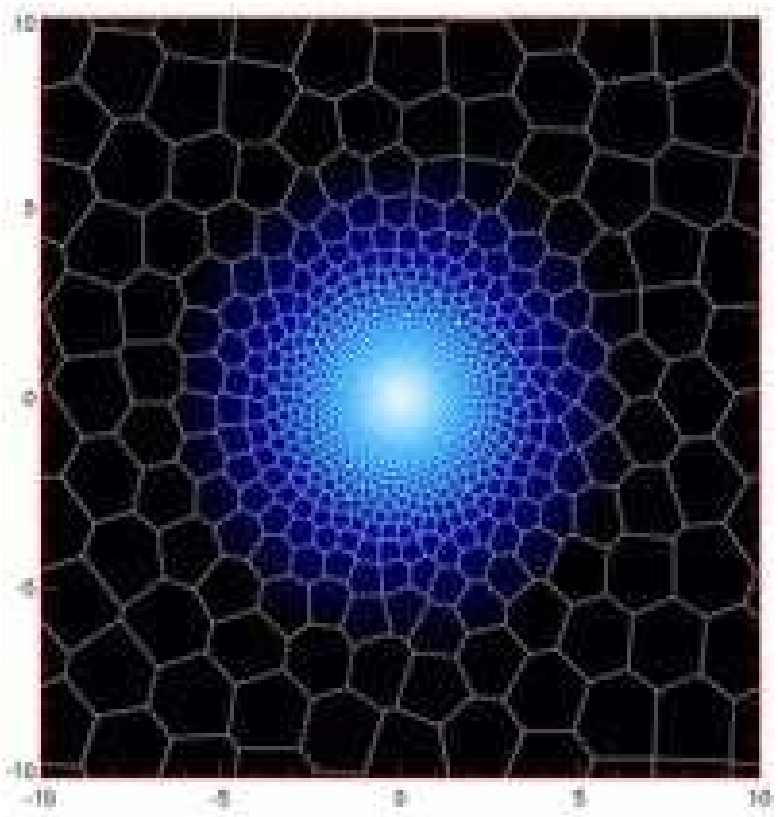
- Both methods readily generalize to higher order methods.

Finite volumes

Idea:

- Subdivide the domain into cells K, not necessarily triangles or squares





(Pictures: Volker Springel, MPA Garching, http://www.mpa-garching.mpg.de/~volker/arepo)

Finite volumes

Idea:

- Subdivide the domain into cells K, not necessarily triangles or squares
- Integrate the equation on every cell:

$$\nabla\cdot(\vec{w}\,u) = f, \qquad u=g \text{ on } \Gamma_{\text{in}}$$

yields

$$\int_K \nabla\cdot(\vec{w}\,u) = \int_K f$$

- Integrate by parts on every cell:

$$\int_{\partial K} \vec{n}\cdot\vec{w}\,u = \int_K f$$

Finite volumes

Idea:

- Use the integral form

$$\int_{\partial K} \vec{n}\cdot\vec{w}\,u \;=\; \int_K f$$

- Then sum over all cells:

$$\sum_{e\in\text{interior edges}} \int_e \vec{n}\cdot\vec{w}\,[u] + \sum_{e\in\text{exterior inflow edges}} \int_e \vec{n}\cdot\vec{w}\,(g-u) \;=\; \sum_K \int_K f$$

where n is an arbitrarily chosen normal to every edge and

$$[u(x)] = \lim_{\epsilon\to 0}\,[u(x+\epsilon n) - u(x-\epsilon n)]$$

is the “jump” of the solution across the edge.

Finite volumes

Idea:

- Now assume that the solution is constant in every cell
- Then:

$$\sum_{e\in\text{interior edges of } K} \int_e \vec{n}\cdot\vec{w}[u] + \sum_{e\in\text{exterior inflow edges of } K} \int_e \vec{n}\cdot\vec{w}(g-u) = \int_K f$$

- This is a linear system for the value on cell K that couples it to the neighboring cells and the boundary values

Finite volumes

In general:

- Finite volume schemes have shown to be exceptionally stable
- Very flexible with regard to cell shapes (arbitrary polygons)
- Typically not very accurate and frequently very diffusive

- Exceptionally difficult to generalize to higher order:
  - No systematic way such as the polynomial degree in finite element methods
  - Need flux limiters, slope limiters, ...

Discontinuous Galerkin methods

Weak formulation:

Start with conservative formulation:

$$\nabla\cdot(\vec{w}\,u) = f \qquad\qquad u = g \quad \text{on } \Gamma_{\text{in}}$$

Get weak formulation:

$$-(\varphi,\nabla\cdot[\vec{w}\,u])_{\Omega} = (\varphi,f)-(\varphi,\vec{n}\cdot\vec{w}\,g)_{\Gamma_{\text{in}}}$$

We want to integrate by parts, but we can only do that if the integrand is smooth.

In the discontinuous Galerkin method (DG), take trial and test functions that are

- Piecewise polynomial on cells
- Discontinuous between cells

Discontinuous Galerkin methods

Thus: Integrating this:

$$-\left(\varphi, \nabla\cdot[\vec{w}\,u]\right)_\Omega \;=\; \left(\varphi, f\right) - \left(\varphi, \vec{n}\cdot\vec{w}\,g\right)_{\Gamma_{\text{in}}}$$

by parts can only be done on every cell individually:

$$\sum_K -\left(\nabla\varphi, \vec{w}\,u\right)_K + \left(\varphi, \vec{n}\cdot\vec{w}\,u\right)_{\partial K\setminus\Gamma_{\text{in}}} \;=\; \left(\varphi, f\right) - \sum_K \left(\varphi, \vec{n}\cdot\vec{w}\,g\right)_{\partial K\cap\Gamma_{\text{in}}}$$

We can re-arrange edge terms to obtain

$$\sum_K -\left(\nabla\varphi, \vec{w}\,u\right)_K + \frac{1}{2}\left(\varphi, \vec{n}\cdot\vec{w}\,[u]\right)_{\partial K\setminus\Gamma_{\text{in}}} \;=\; \left(\varphi, f\right) - \sum_K \left(\varphi, \vec{n}\cdot\vec{w}\,g\right)_{\partial K\cap\Gamma_{\text{in}}}$$

Note: We then have to decide what value the test function should have on edges (where it is discontinuous). This leads to lots of technicalities.

Discontinuous Galerkin methods

In general:

- Lowest order DG methods are often equal to the finite volume method
- This provides a convenient way to analyze the FVM

- Provides a systematic way of producing schemes of higher order

Limitations for numerical schemes

For any scheme there holds “Godunov's Theorem”:

***Any* linear scheme that does not produce oscillations can be at most first-order accurate!**

- If we insist on methods that are non-oscillatory, then just using higher-order elements is not sufficient
- We need to use a nonlinear scheme even for linear equations

Note: It doesn't matter whether we want to use FEM, FDM, FVM or any other method. They all fall under the theorem!

MATH 676

–

Finite element methods in scientific computing

Wolfgang Bangerth, Texas A&M University