Efficient computation of Jacobian matrices for entropy stable summation-by-parts schemes

Jesse Chan, Christina Taylor

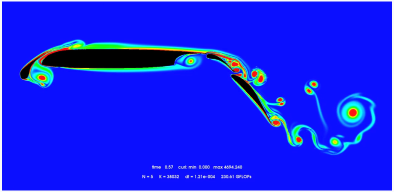
SIAM TX-LA Meeting

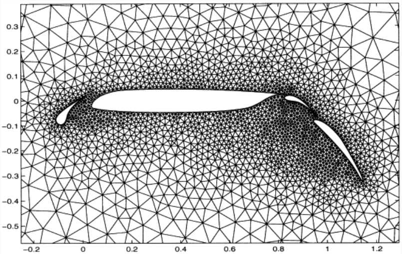
October 17, 2020

Department of Computational and Applied Mathematics

High order finite element methods for hyperbolic PDEs

- Aerodynamics applications: acoustics, vorticular flows, turbulence, shocks.
- Goal: high accuracy simulations on unstructured meshes.
- Discontinuous Galerkin (DG) methods: geometric flexibility, high order accuracy.

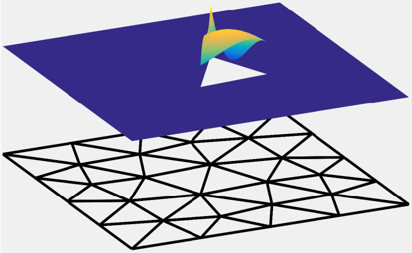




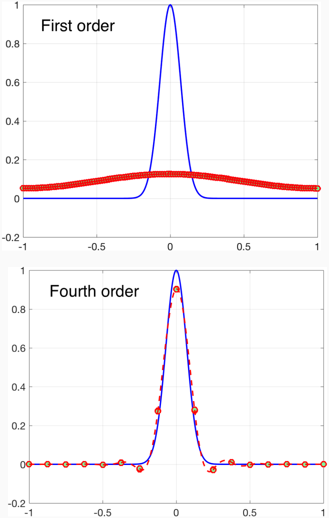
Mesh from Slawig 2001.

High order finite element methods for hyperbolic PDEs

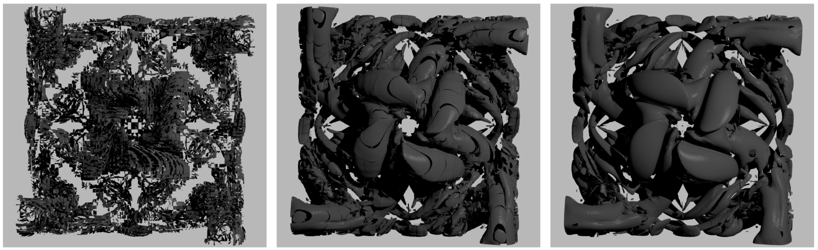
- Aerodynamics applications: acoustics, vorticular flows, turbulence, shocks.
- Goal: high accuracy simulations on unstructured meshes.
- Discontinuous Galerkin (DG) methods: geometric flexibility, high order accuracy.

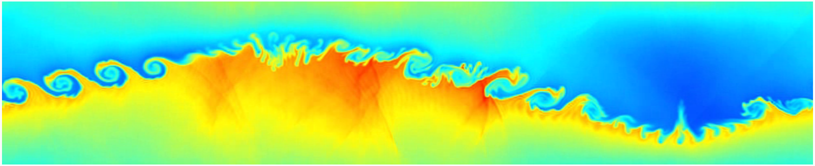


Why high order accuracy?



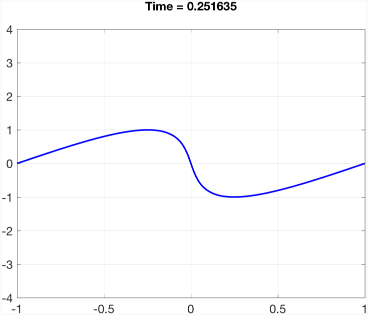
Why high order accuracy?



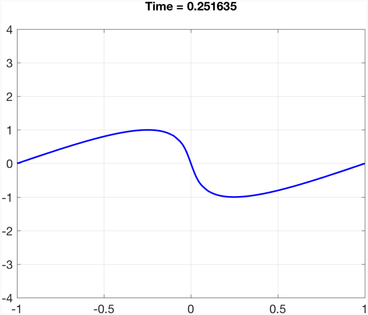


2nd, 4th, and 16th order Taylor-Green (top), 8th order Kelvin-Helmholtz (bottom).
Vorticular structures and acoustic waves are both sensitive to numerical dissipation.
Results from Beck and Gassner (2013) and Per-Olof Persson's website.

Why *not* high order DG methods?



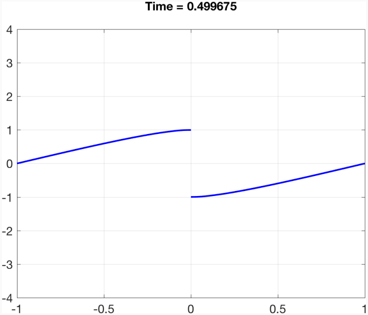
(a) Exact solution



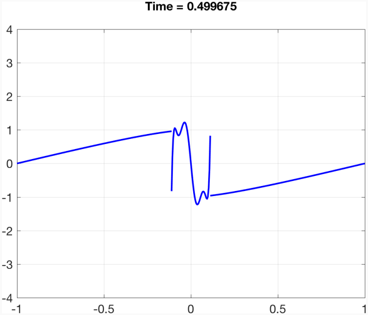
(b) 8th order DG

High order methods blow up for under-resolved solutions of nonlinear conservation laws (e.g., shocks and turbulence).

Why *not* high order DG methods?



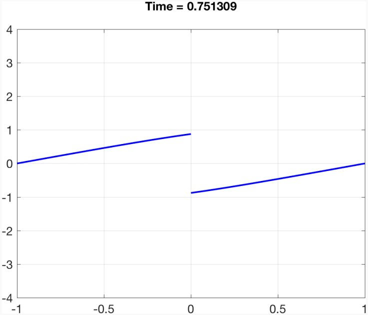
(a) Exact solution



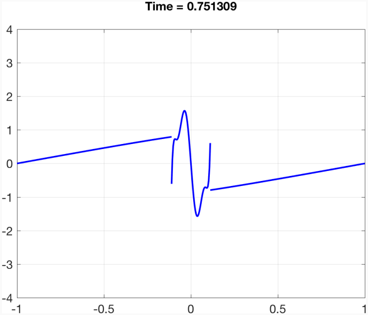
(b) 8th order DG

High order methods blow up for under-resolved solutions of nonlinear conservation laws (e.g., shocks and turbulence).

Why *not* high order DG methods?



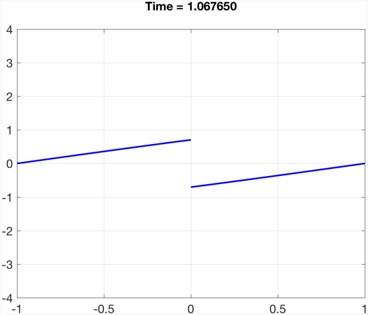
(a) Exact solution



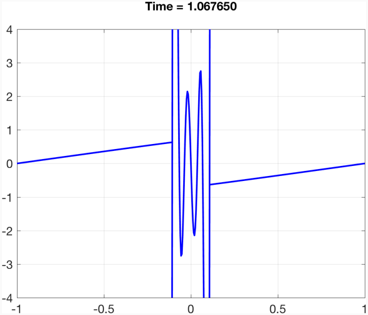
(b) 8th order DG

High order methods blow up for under-resolved solutions of nonlinear conservation laws (e.g., shocks and turbulence).

Why *not* high order DG methods?



(a) Exact solution



(b) 8th order DG

High order methods blow up for under-resolved solutions of nonlinear conservation laws (e.g., shocks and turbulence).

Continuous stability $\not\Longrightarrow$ discrete stability

- Entropy stability: generalizes energy stability to nonlinear conservation laws (shallow water, compressible Euler + Navier-Stokes).

$$\frac{\partial \boldsymbol{u}}{\partial t} + \frac{\partial \boldsymbol{f}(\boldsymbol{u})}{\partial x} = 0.$$

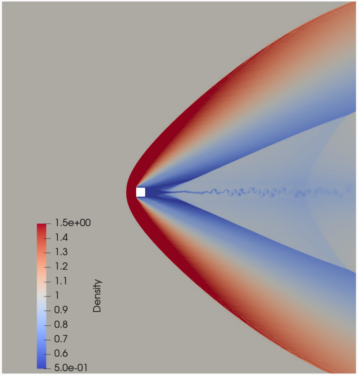
- Continuous entropy inequality: given a convex entropy function $S(\boldsymbol{u})$ and "entropy potential" $\psi(\boldsymbol{u})$, test with $\boldsymbol{v}(\boldsymbol{u})$

$$\int_\Omega \boldsymbol{v}^T \left(\frac{\partial \boldsymbol{u}}{\partial t} + \frac{\partial \boldsymbol{f}(\boldsymbol{u})}{\partial x}\right) = 0, \qquad \boxed{\boldsymbol{v}(\boldsymbol{u}) = \frac{\partial S}{\partial \boldsymbol{u}}}$$

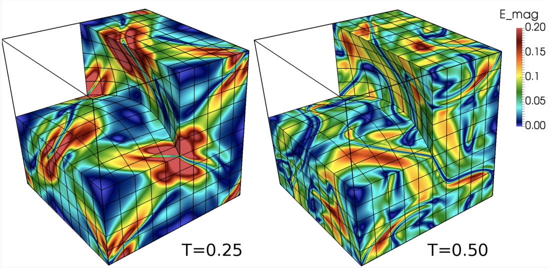
$$\Longrightarrow \int_\Omega \frac{\partial S(\boldsymbol{u})}{\partial t} + \left(\boldsymbol{v}^T \boldsymbol{f}(\boldsymbol{u}) - \psi(\boldsymbol{u})\right)\Big|_{-1}^{1} \leq 0.$$

- Proof of entropy inequality relies on chain rule, integration by parts.

Entropy stable methods recover a discrete entropy inequality.



(a) Compressible Navier-Stokes (Ma=1.5, Re=10000, degree 3)



(b) Resistive MHD (3D Orszag-Tang)

No limiters, filters, or artificial viscosity beyond DG "upwinded" fluxes.

Bohm et al. (2019). *An entropy stable nodal DG method for the resistive MHD equations. Part I.*

Talk outline

1. Entropy stable nodal summation-by-parts (SBP) schemes
2. Jacobian matrices for "flux differencing" formulations
3. Numerical experiments

Talk outline

1 Entropy stable nodal summation-by-parts (SBP) schemes

2 Jacobian matrices for "flux differencing" formulations

3 Numerical experiments

Ingredients for entropy stable methods

- Summation-by-parts (SBP) differentiation matrices $\mathbf{Q}_i$ with boundary matrix $\mathbf{B}_i$

$$\mathbf{Q}_i + \mathbf{Q}_i^T = \mathbf{B}_i.$$

- Tadmor-type entropy conservative flux

$$\begin{aligned} \boldsymbol{f}_S(\boldsymbol{u},\boldsymbol{u}) &= \boldsymbol{f}(\boldsymbol{u}), && \text{(consistency)}\\ \boldsymbol{f}_S(\boldsymbol{u},\boldsymbol{v}) &= \boldsymbol{f}_S(\boldsymbol{v},\boldsymbol{u}), && \text{(symmetry)}\\ (\boldsymbol{v}_L - \boldsymbol{v}_R)^T \boldsymbol{f}_S(\boldsymbol{u}_L,\boldsymbol{u}_R) &= \psi_L - \psi_R, && \text{(conservation).} \end{aligned}$$

Flux differencing formulations

- Main idea: "flux differencing" discretization

$$\mathbf{M}\frac{\mathrm{d}\mathbf{u}}{\mathrm{dt}} + \sum_{i=1}^{d}\left(\left(\mathbf{Q}_i - \mathbf{Q}_i^T\right)\circ\mathbf{F}_i\right)\mathbf{1} + \mathbf{B}_i\mathbf{f}^* = \mathbf{0}.$$

$$\left(\mathbf{F}_i\right)_{jk} = \boldsymbol{f}_{i,S}\left(\mathbf{u}_j, \mathbf{u}_k\right)$$

- Can prove discrete entropy conservation

$$\mathbf{1}^T\mathbf{M}\frac{\mathrm{d}S(\mathbf{u})}{\mathrm{dt}} = 0, \qquad \text{for appropriate BCs}$$

- Add numerical or physical dissipation for discrete entropy inequality

$$\mathbf{1}^T\mathbf{M}\frac{\mathrm{d}S(\mathbf{u})}{\mathrm{dt}} + \underbrace{\mathbf{v}^T\left(\mathbf{K}\circ\mathbf{D}\right)\mathbf{1}}_{\geq 0} = 0$$

Example 1: entropy conservative finite volume methods

- Usual formulation

$$\frac{\mathrm{d}\mathbf{u}_i}{\mathrm{dt}} + \frac{\boldsymbol{f}_S(\mathbf{u}_{i+1}, \mathbf{u}_i) - \boldsymbol{f}_S(\mathbf{u}_i, \mathbf{u}_{i-1})}{h} = \mathbf{0}$$

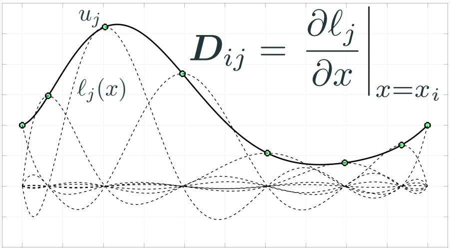
- Mass matrix $\mathbf{M} = h\mathbf{I}$, differentiation matrix $\mathbf{Q}$

$$\mathbf{Q} = \frac{1}{2}\begin{bmatrix} -1 & 1 & & \\ -1 & & \ddots & \\ & \ddots & & 1 \\ & & -1 & 1 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} -1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}.$$

- Let $\mathbf{f}^* = [\mathbf{f}_L, 0, \ldots, 0, \mathbf{f}_R]$ impose BCs. Equivalent to

$$\mathbf{M}\frac{\mathrm{d}\mathbf{u}}{\mathrm{dt}} + \left(\left(\mathbf{Q} - \mathbf{Q}^T\right) \circ \mathbf{F}\right)\mathbf{1} + \mathbf{B}\mathbf{f}^* = \mathbf{0}.$$

Example 2: high order spectral collocation



$$\boldsymbol{B} = \begin{bmatrix} -1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}$$

- SBP operators at Lobatto quadrature points: lumped mass matrix $\mathbf{M}$, weak differentiation matrix $\mathbf{Q} = \mathbf{M}\mathbf{D}$.
- Entropy conservative formulation: same algebraic structure

$$\mathbf{M}\frac{\mathrm{d}\mathbf{u}}{\mathrm{dt}} + \left(\left(\mathbf{Q} - \mathbf{Q}^T\right) \circ \mathbf{F}\right) \mathbf{1} + \mathbf{B}\mathbf{f}^* = \mathbf{0}.$$

Example 3: high order modal DG formulations

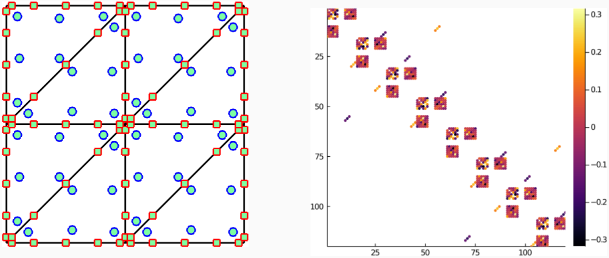


Figure: Global SBP-DG matrix for $N = 2$ on a periodic mesh.

- $\mathbf{V}_h, \mathbf{V}_f$ matrices interpolate basis coefficients to quadrature.
- Formulation with global SBP matrix $\mathbf{Q}_i$.

$$\mathbf{M}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} + \sum_{i=1}^{d} \mathbf{V}_h^T \left(\left(\mathbf{Q}_i - \mathbf{Q}_i^T\right) \circ \mathbf{F}_i\right) \mathbf{1} + \mathbf{V}_f^T \mathbf{B}\mathbf{f}^* = \mathbf{0}.$$

Example 3: high order modal DG formulations, cont.

- "Modal" formulation uses L^2 projection of entropy variables $\boldsymbol{v}(\boldsymbol{u}_h)$

$$\Pi_N \boldsymbol{v}\left(\boldsymbol{u}_h\right), \qquad \Pi_N = L^2 \text{ projection onto degree } N \text{ polynomials}$$

- Flux matrix $\mathbf{F}_i$ must be computed in terms of the *entropy-projected conservative variables* $\widetilde{\mathbf{u}}$

$$\left(\mathbf{F}_i\right)_{jk} = \boldsymbol{f}_{i,S}\left(\widetilde{\mathbf{u}}_j, \widetilde{\mathbf{u}}_k\right)$$
$$\widetilde{\mathbf{u}} \approx \boldsymbol{u}\left(\Pi_N \boldsymbol{v}\left(\boldsymbol{u}_h\right)\right).$$

- Enables near-arbitrary combinations of basis and quadrature.

Example 3: high order modal DG formulations, cont.

- "Modal" formulation uses L^2 projection of entropy variables $\boldsymbol{v}(\boldsymbol{u}_h)$

$$\Pi_N \boldsymbol{v}\left(\boldsymbol{u}_h\right), \qquad \Pi_N = L^2 \text{ projection onto degree } N \text{ polynomials}$$

- Flux matrix $\mathbf{F}_i$ must be computed in terms of the *entropy-projected conservative variables* $\widetilde{\mathbf{u}}$

$$\left(\mathbf{F}_i\right)_{jk} = \boldsymbol{f}_{i,S}\left(\widetilde{\mathbf{u}}_j, \widetilde{\mathbf{u}}_k\right)$$
$$\widetilde{\mathbf{u}} \approx \boldsymbol{u}\left(\Pi_N \boldsymbol{v}\left(\boldsymbol{u}_h\right)\right).$$

- Enables near-arbitrary combinations of basis and quadrature.

Example 3: high order modal DG formulations, cont.

- "Modal" formulation uses L^2 projection of entropy variables $\boldsymbol{v}(\boldsymbol{u}_h)$

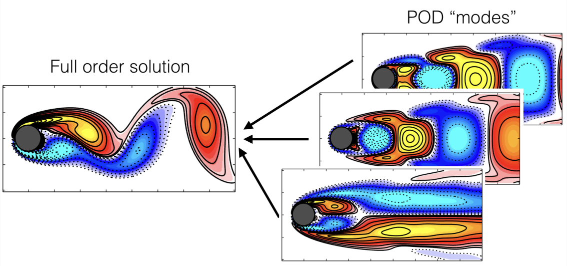
$$\Pi_N \boldsymbol{v}\left(\boldsymbol{u}_h\right), \qquad \Pi_N = L^2 \text{ projection onto degree } N \text{ polynomials}$$

- Flux matrix $\mathbf{F}_i$ must be computed in terms of the *entropy-projected conservative variables* $\widetilde{\mathbf{u}}$

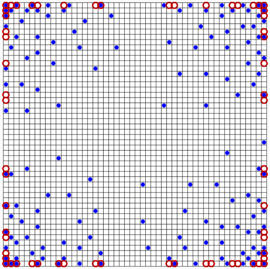
$$\left(\mathbf{F}_i\right)_{jk} = \boldsymbol{f}_{i,S}\left(\widetilde{\mathbf{u}}_j, \widetilde{\mathbf{u}}_k\right)$$
$$\widetilde{\mathbf{u}} \approx \boldsymbol{u}\left(\Pi_N \boldsymbol{v}\left(\boldsymbol{u}_h\right)\right).$$

- Enables near-arbitrary combinations of basis and quadrature.

Example 4: entropy stable reduced order modeling



(a) Figure adapted from Brunton, Proctor, Kutz (2016)

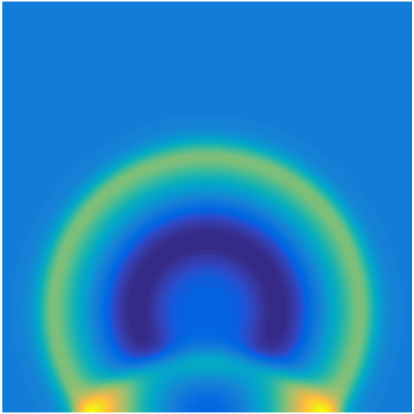


(b) Hyper-reduction

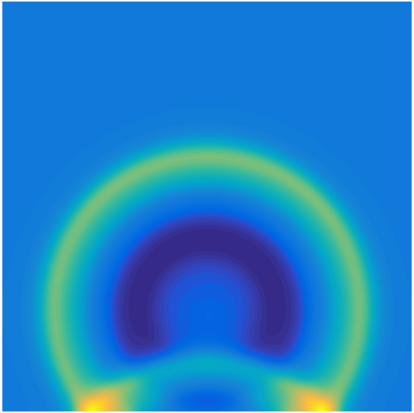
- Principal orthogonal decomposition (POD) $\Longleftrightarrow$ basis functions
- Sampling/weighting hyper-reduction $\Longleftrightarrow$ quadrature
- Treat as single-element modal DG scheme.

Chan (2020), *Entropy stable reduced order modeling of nonlinear conservation laws.*

Example 4: entropy stable reduced order modeling, cont.



(a) Density, full order model

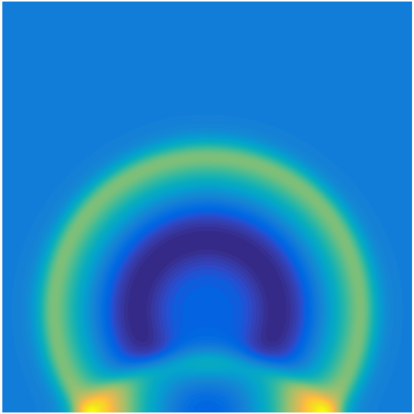


(b) Reduced order model

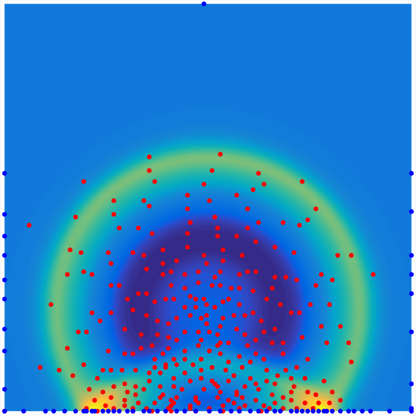
Full order model with 10000 points, ROM with 25 modes, 306 points.

Chan (2020), *Entropy stable reduced order modeling of nonlinear conservation laws.*

Example 4: entropy stable reduced order modeling, cont.



(a) Density, full order model



(b) ROM w/reduced quad. points

Full order model with 10000 points, ROM with 25 modes, 306 points.

Chan (2020), *Entropy stable reduced order modeling of nonlinear conservation laws.*

Talk outline

1 Entropy stable nodal summation-by-parts (SBP) schemes

2 Jacobian matrices for "flux differencing" formulations

3 Numerical experiments

Current methods for computing Jacobian matrices

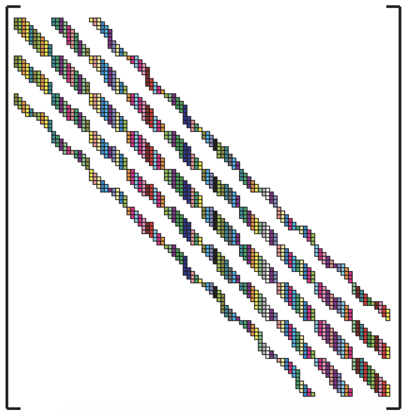


Figure from Gebremedhin, Manne, Pothen (2005), *What color is your Jacobian? Graph coloring for computing derivatives.*

- Matrix-free approach if only computing Jacobian-vector products

$$\mathbf{J}(\mathbf{u})\Delta\mathbf{u} \approx \frac{\mathbf{f}(\mathbf{u}+\Delta\mathbf{u}) - \mathbf{f}(\mathbf{u})}{\|\Delta\mathbf{u}\|}.$$

- Compute entries using finite differences or automatic differentiation (AD)
- Graph coloring reduces function evaluations and AD costs, but only for sparse matrices

Cost of computing dense Jacobian blocks

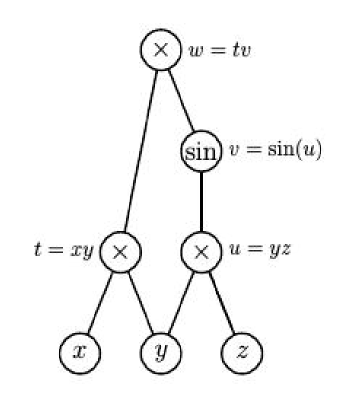


Image from Austin (2017), *How to Differentiate with a Computer.*

- Graph coloring AD expensive for dense matrix blocks (e.g., high order DG methods, reduced order models).
- Problem: cost of AD scales with size of input and output dimension.
- Can reduce costs by only applying AD only to nonlinear flux $\boldsymbol{f}_S(\boldsymbol{u}_L, \boldsymbol{u}_R)$.

Jacobian matrices for flux differencing

Hadamard product structure of flux differencing yields simple Jacobians.

Theorem

Assume $\mathbf{Q} = \pm\mathbf{Q}^T$*. Consider a scalar "collocation" discretization*

$$\mathbf{r}(\mathbf{u}) = (\mathbf{Q} \circ \mathbf{F})\,\mathbf{1}, \qquad \mathbf{F}_{ij} = f_S(\mathbf{u}_i, \mathbf{u}_j).$$

The Jacobian matrix is then

$$\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}\mathbf{u}} = (\mathbf{Q} \circ \partial\mathbf{F}_R) \pm \operatorname{diag}\left(\mathbf{1}^T\left(\mathbf{Q} \circ \partial\mathbf{F}_R\right)\right),$$

$$(\partial\mathbf{F}_R)_{ij} = \left.\frac{\partial f_S(u_L, u_R)}{\partial u_R}\right|_{\mathbf{u}_i,\mathbf{u}_j}.$$

Observations about flux differencing Jacobian formulas

Separates "template" discretization matrix Q and flux contributions.

$$\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}\mathbf{u}} = (\mathbf{Q} \circ \partial \mathbf{F}_R) \pm \operatorname{diag}\left(\mathbf{1}^T (\mathbf{Q} \circ \partial \mathbf{F}_R)\right),$$

$$(\partial \mathbf{F}_R)_{ij} = \left.\frac{\partial f_S(u_L, u_R)}{\partial u_R}\right|_{\mathbf{u}_i, \mathbf{u}_j}.$$

Option 1: compute derivatives $\frac{\partial f_S(u_L,u_R)}{\partial u_R}$ analytically

$$f_S(u_L, u_R) = \frac{1}{6}\left(u_L^2 + u_L u_R + u_R^2\right)$$

$$\frac{\partial f_S(u_L, u_R)}{\partial u_R} = \frac{1}{6}\left(u_L + 2u_R\right).$$

Observations about flux differencing Jacobian formulas

Separates "template" discretization matrix Q and flux contributions.

$$\frac{d\mathbf{r}}{d\mathbf{u}} = \left(\mathbf{Q} \circ \partial \mathbf{F}_R\right) \pm \operatorname{diag}\left(\mathbf{1}^T\left(\mathbf{Q} \circ \partial \mathbf{F}_R\right)\right),$$

$$\left(\partial \mathbf{F}_R\right)_{ij} = \left.\frac{\partial f_S(u_L, u_R)}{\partial u_R}\right|_{\mathbf{u}_i, \mathbf{u}_j}.$$

Option 2: use AD for $\frac{\partial f_S(u_L,u_R)}{\partial u_R}$. Efficient: $O(1)$ inputs/outputs. In Julia:

```
using ForwardDiff
f(uL,uR) = (1/6)*(uL^2 + uL*uR + uR^2)
dF(uL,uR) = ForwardDiff.derivative(uR->f(uL,uR),uR)
```

Fluxes can be complicated to differentiate analytically

- Entropy conservative fluxes for 1D compressible Euler

$$
\begin{aligned}
f_S^1(\boldsymbol{u}_L, \boldsymbol{u}_R) &= \{\{\rho\}\}^{\log} \{\{u\}\} \\
f_S^2(\boldsymbol{u}_L, \boldsymbol{u}_R) &= \frac{\{\{\rho\}\}}{2\,\{\{\beta\}\}} + \{\{u\}\}\, f_S^1 \\
f_S^3(\boldsymbol{u}_L, \boldsymbol{u}_R) &= f_S^1 \left(\frac{1}{2(\gamma-1)\,\{\{\beta\}\}^{\log}} - \frac{1}{2}\{\{u^2\}\} \right) + \{\{u\}\}\, f_S^2,
\end{aligned}
$$

- Fluxes involve logarithmic mean $\{\{u\}\}^{\log} = \frac{u_L - u_R}{\log u_L - \log u_R}$ and “inverse temperature” $\beta = \frac{\rho}{2p}$.
- Specialized evaluation of $\{\{u\}\}^{\log}$ using γ-expansions.

Chandreshekar (2013), *KEP and entropy stable FV schemes for comp. Euler and NS equations.*

Winters et al. (2020), *ES numerical approximations for the isothermal and polytropic Euler equations.*

Extensions: systems of nonlinear conservation laws

- Assume n fields; nonlinear term is now

$$\mathbf{r}(\mathbf{u}) = ((\mathbf{I}_n \otimes \mathbf{Q}) \circ \mathbf{F})\,\mathbf{1} = \begin{bmatrix} (\mathbf{Q} \circ \mathbf{F}_1)\,\mathbf{1} \\ \vdots \\ (\mathbf{Q} \circ \mathbf{F}_n)\,\mathbf{1} \end{bmatrix}, \qquad (\mathbf{F}_\ell)_{ij} = (\boldsymbol{f}_S(\mathbf{u}_i, \mathbf{u}_j))_\ell\,.$$

- Jacobian matrix involves Jacobian of $\boldsymbol{f}_S$

$$\frac{\partial \mathbf{f}}{\partial \mathbf{u}} = \begin{bmatrix} \mathbf{F}_{1,\mathbf{u}_1} & \dots & \mathbf{F}_{1,\mathbf{u}_n} \\ \vdots & \ddots & \vdots \\ \mathbf{F}_{n,\mathbf{u}_1} & \dots & \mathbf{F}_{n,\mathbf{u}_n} \end{bmatrix}, \qquad \partial \mathbf{F}_{i,\mathbf{u}_j} \Longleftrightarrow \frac{\partial\,(\boldsymbol{f}_S)_i}{\partial \boldsymbol{u}_{R,j}}$$

$$\mathbf{F}_{i,\mathbf{u}_j} = \left(\mathbf{Q} \circ \partial \mathbf{F}_{i,\mathbf{u}_j}\right) \pm \operatorname{diag}\left(\mathbf{1}^T\left(\mathbf{Q} \circ \partial \mathbf{F}_{i,\mathbf{u}_j}\right)\right)$$

Extensions: dissipative terms

- Define anti-symmetric entropy dissipative flux (e.g., Lax-Friedrichs)

$$\boldsymbol{d}_S(\boldsymbol{u}_L, \boldsymbol{u}_R) = -\boldsymbol{d}_S(\boldsymbol{u}_R, \boldsymbol{u}_L)$$
$$(\boldsymbol{v}_L - \boldsymbol{v}_R)^T \boldsymbol{d}_S(\boldsymbol{u}_L, \boldsymbol{u}_R) \geq 0.$$

- Dissipation matrix $\mathbf{K}$ (symmetric, non-negative entries)

$$\mathbf{d}(\mathbf{u}) = (\mathbf{K} \circ \mathbf{D})\,\mathbf{1}, \qquad \mathbf{D}_{ij} = \boldsymbol{d}_S(\mathbf{u}_i, \mathbf{u}_j).$$

- Jacobian of $\mathbf{d}(\mathbf{u})$ is similar to previous formulas

$$\frac{\partial \mathbf{d}}{\partial \mathbf{u}} = -(\mathbf{K} \circ \partial \mathbf{D}_R^T) + \operatorname{diag}\left(\left(\mathbf{K} \circ \partial \mathbf{D}_R^T\right)\mathbf{1}\right).$$

Extensions: modal DG methods (entropy projection)

$$\frac{\partial \mathbf{r}}{\partial \widehat{\mathbf{u}}} = \underbrace{\boxed{\mathbf{V}_h^T}}_{(N_p \times N_{\text{total}})} \underbrace{\boxed{\mathbf{Q} \circ \partial \mathbf{F}}}_{(N_{\text{total}} \times N_{\text{total}})} \underbrace{\boxed{\left.\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{v}}\right|_{\mathbf{V}_h \mathbf{P} \mathbf{v}_q}}}_{(N_{\text{total}} \times N_{\text{total}})} \underbrace{\boxed{\mathbf{V}_h \mathbf{P}}}_{(N_{\text{total}} \times N_q)} \underbrace{\boxed{\left.\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{u}}\right|_{\mathbf{V}\widehat{\mathbf{u}}}}}_{(N_q \times N_q)} \underbrace{\boxed{\mathbf{V}}}_{(N_q \times N_p)}$$

- Suppose degrees of freedom are N_p "modal" coefficients $\widehat{\mathbf{u}}$.
- Fluxes use entropy projected conservative variables $\boldsymbol{u}\left(\Pi_N \boldsymbol{v}(\boldsymbol{u}_h)\right)$.
- Jacobian requires projection/interpolation matrices and Jacobians of transformations between conservative and entropy variables.

Talk outline

1 Entropy stable nodal summation-by-parts (SBP) schemes

2 Jacobian matrices for "flux differencing" formulations

3 Numerical experiments

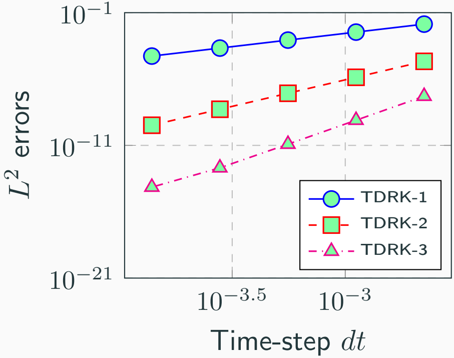
Computational timings

- Ratio of cost of flux evaluation to cost of AD in Julia:
  - For $1/(uL + uR)$: 7.132 μs / 13.593 μs
  - For logmean $\{\{u\}\}^{\log} = \frac{u_L - u_R}{\log u_L - \log u_R}$: 129.254 μs / 161.322 μs

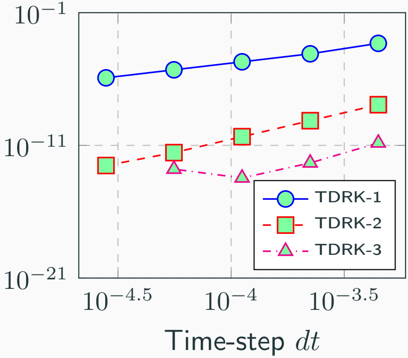
- Jacobian timings for $f_S(u_L, u_R) = \frac{1}{6}\left(u_L^2 + u_L u_R + u_R^2\right)$ and dense differentiation matrices $\mathbf{Q} \in \mathbb{R}^{N\times N}$.

	N = 10	N = 25	N = 50
Direct automatic differentiation	5.666	60.388	373.633
`FiniteDiff.jl`	1.429	17.324	125.894
Jacobian formula (analytic flux deriv.)	.209	1.005	3.249
Jacobian formula (AD flux deriv.)	.210	1.030	3.259
Evaluation of $\mathbf{f}(\mathbf{u})$ (for reference)	.120	.623	2.403

Application: two-derivative time-stepping methods



(a) Burgers' equation

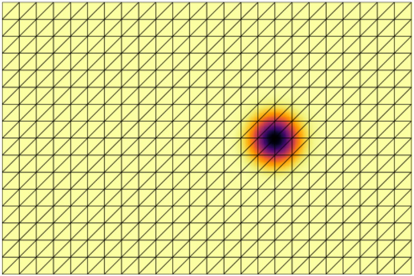


(b) Shallow water equations

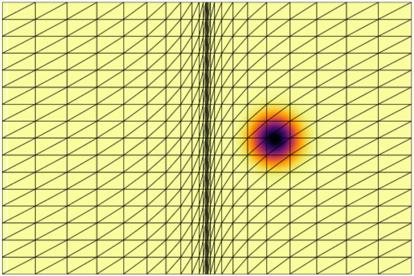
Two-derivative Runge-Kutta (TDRK) schemes: 2nd order example

$$\mathbf{u}^{k+1} = \mathbf{u}^k - \Delta t \mathbf{f}(\mathbf{u}^k) + \frac{\Delta t^2}{2}\mathbf{g}(\mathbf{u}^k), \quad \mathbf{g}(\mathbf{u}) = \frac{\mathrm{d}^2\mathbf{u}}{\mathrm{d}t^2} = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = -\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\mathbf{f}(\mathbf{u}).$$

Application: implicit midpoint method, compressible Euler



(a) Uniform mesh, L^2 error $.0901$



(b) Anisotropic mesh, L^2 error $.0935$

Figure: Solutions for a degree $N = 3$ modal DG method with $dt = .1$ on uniform and "squeezed" meshes.

Conclusions

- Simple Jacobian formulas for entropy stable flux differencing schemes.
- Concise and efficient Julia implementation `FluxDiffUtils.jl` (available on Github, will be registered soon).
- This work was supported by DMS-1719818 and DMS-CAREER-1943186.

Thank you! Questions?

Chan, Taylor (2020). *Efficient computation of Jacobian matrices for entropy stable summation-by-parts schemes.*