Spectral Lagrangian methods (in particular for SW)

L. Micalizzi[1,2], S. Tokareva[2], M. Ricchiuto[3], R. Abgrall[1]

[1]Institut für Mathematik,
Universität Zürich

[2]Theoretical Division,
Los Alamos National Laboratory

[3]Team CARDAMOM,
Inria Bordeaux sud-ouest

Outline

- Governing equations
- Scheme and motivation
- Numerics
  - Sod
  - Smooth periodic
  - Well-balancing
  - Supercritical smooth

Section 1

Governing equations

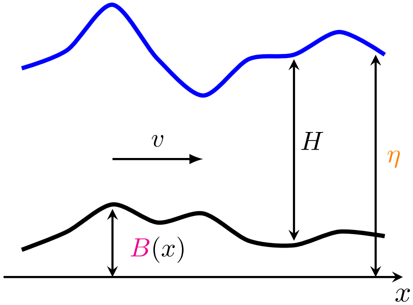
Shallow water equations, Eulerian

$$\frac{\partial}{\partial t}\boldsymbol{u} + div_{\boldsymbol{x}}\boldsymbol{F}(\boldsymbol{u}) = \boldsymbol{S}(\boldsymbol{x},\boldsymbol{u}), \quad (\boldsymbol{x},t) \in \Omega \times [0,T]$$

$$\boldsymbol{u} := \begin{pmatrix} H \\ H\boldsymbol{v} \end{pmatrix}$$

$$\boldsymbol{F}(\boldsymbol{u}) := \begin{pmatrix} H\boldsymbol{v} \\ H\boldsymbol{v}\otimes\boldsymbol{v} + g\frac{H^2}{2}\mathbb{I} \end{pmatrix}$$

$$\boldsymbol{S}(\boldsymbol{x},\boldsymbol{u}) := -\begin{pmatrix} 0 \\ gH\nabla_{\boldsymbol{x}}B(x) \end{pmatrix}$$



$$\eta := H + B$$

Shallow water equations, simplification of Euler (with gravity)

$$\frac{\partial}{\partial t}\boldsymbol{u} + div_{\boldsymbol{x}}\boldsymbol{F}(\boldsymbol{u}) = \boldsymbol{S}(\boldsymbol{x},\boldsymbol{u}), \quad (\boldsymbol{x},t) \in \Omega \times [0,T]$$

$$\boldsymbol{u} := \begin{pmatrix} H \\ H\boldsymbol{v} \end{pmatrix}$$

$$\boldsymbol{F}(\boldsymbol{u}) := \begin{pmatrix} H\boldsymbol{v} \\ H\boldsymbol{v}\otimes\boldsymbol{v} + g\frac{H^2}{2}\mathbb{I} \end{pmatrix}$$

$$\boldsymbol{S}(\boldsymbol{x},\boldsymbol{u}) := -\begin{pmatrix} 0 \\ gH\nabla_{\boldsymbol{x}}B(x) \end{pmatrix}$$

$$\rho \sim H$$

$$p \sim g\frac{H^2}{2}$$

$$\phi(x) \sim B(x)$$

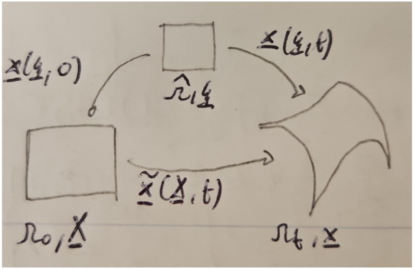
Neglecting the energy equation

Section 2

Scheme and motivation

Shallow water equations, Lagrangian (sorry for low quality)

$$\Omega_0 \ni \boldsymbol{X} \longrightarrow \boldsymbol{x} = \widetilde{\boldsymbol{x}}(\boldsymbol{X}, t) \in \Omega_t$$
$$\widehat{\Omega} \ni \boldsymbol{\xi} \longrightarrow \boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t) \in \Omega_t$$
$$\boldsymbol{X} = \boldsymbol{x}(\boldsymbol{\xi}, 0)$$



Shallow water equations, Lagrangian

- **Motion**

$$\frac{d}{dt}\boldsymbol{x}(\boldsymbol{\xi},t) = \boldsymbol{v}(\boldsymbol{x}(\boldsymbol{\xi},t),t)$$

- **Water height/density (SMC)**

$$H(\boldsymbol{x}(\boldsymbol{\xi},t),t) = \frac{\widehat{H}(\boldsymbol{\xi})}{det\boldsymbol{J}(\boldsymbol{\xi},t)}, \quad \boldsymbol{J}(\boldsymbol{\xi},t) = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi},t)$$

- **Velocity**

$$\begin{aligned} H(\boldsymbol{x},t)\frac{d}{dt}\boldsymbol{v}(\boldsymbol{x},t) &= -\nabla_{\boldsymbol{x}} p - gH\nabla_{\boldsymbol{x}} B \\ &= -\nabla_{\boldsymbol{x}}\left(\frac{gH^2}{2}\right) - gH\nabla_{\boldsymbol{x}} B \end{aligned}$$

$$\frac{d}{dt}\boldsymbol{v}(\boldsymbol{x},t) = -g\nabla_{\boldsymbol{x}}(H+B)$$

Shallow water equations, FEM discretization[1]

$\widehat{\varphi}_i \in \mathbb{P}_{M+1}$ continuous
$\widehat{\psi}_i \in \mathbb{P}_M$ discontinuous

$$\varphi_i(\boldsymbol{x}(\boldsymbol{\xi}, t), t) = \widehat{\varphi}_i(\boldsymbol{\xi})$$
$$\psi_i(\boldsymbol{x}(\boldsymbol{\xi}, t), t) = \widehat{\psi}_i(\boldsymbol{\xi})$$

$$\boldsymbol{x}_h(\boldsymbol{\xi}, t) = \sum_i \boldsymbol{x}_i(t)\widehat{\varphi}_i(\boldsymbol{\xi})$$

$$\boldsymbol{v}_h(\boldsymbol{\xi}, t) = \sum_i \boldsymbol{v}_i(t)\widehat{\varphi}_i(\boldsymbol{\xi})$$

$$H_h(\boldsymbol{\xi}, t) = \sum_i H_i(t)\widehat{\psi}_i(\boldsymbol{\xi})$$

$$\boldsymbol{v}_h(\boldsymbol{x}, t) = \sum_i \boldsymbol{v}_i(t)\widehat{\varphi}_i(\boldsymbol{\xi}(\boldsymbol{x}, t))$$

$$H_h(\boldsymbol{x}, t) = \sum_i H_i(t)\widehat{\psi}_i(\boldsymbol{\xi}(\boldsymbol{x}, t))$$

[1] Dobrev, Kolev, Rieben, High-Order Curvilinear Finite Element Methods for Lagrangian Hydrodynamics, 2012

- **Motion**

$$\frac{d}{dt}\boldsymbol{x}_i(t) = \boldsymbol{v}_i(t)$$

- **Water height/density (SMC) $\Rightarrow$ PP**

$$H_i(t) = \frac{\widehat{H}_i}{det\boldsymbol{J}(\boldsymbol{\xi}_i, t)}, \quad \boldsymbol{J}(\boldsymbol{\xi}, t) = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi}, t)$$

- **Velocity**

$$\sum_{K \in K_i} \sum_{\boldsymbol{x}_j \in K} \left(\int_K \varphi_i \varphi_j d\boldsymbol{x} \right) \frac{d}{dt}\boldsymbol{v}_j(t)$$
$$= -g \sum_{K \in K_i} \left[\int_K \varphi_i \nabla_{\boldsymbol{x}}(H_h + B_h) d\boldsymbol{x} \right] - \sum_{K \in K_i} \boldsymbol{CT}_i^K - \boldsymbol{ST}_i$$

Shallow water equations, discretized equations

$$\sum_{K\in K_i}\sum_{\boldsymbol{x}_j\in K}\left(\int_K \varphi_i\varphi_j d\boldsymbol{x}\right)\frac{d}{dt}\boldsymbol{v}_j(t)$$
$$= -g\sum_{K\in K_i}\left[\int_K \varphi_i \nabla_{\boldsymbol{x}}(H_h+B_h)d\boldsymbol{x}\right] - \sum_{K\in K_i}\boldsymbol{CT}_i^K - \boldsymbol{ST}_i$$

$$\boldsymbol{CT}_i^K = g\int_{\partial K}\varphi_i(\eta^* - \eta|_K)\boldsymbol{\nu} d\boldsymbol{\sigma}$$

$$\boldsymbol{ST}_i^{CIP} := \sum_f \alpha_f \left[\!\left[\frac{\partial}{\partial x}\varphi_i\right]\!\right]\left[\!\left[\frac{\partial}{\partial x}v\right]\!\right], \quad \boldsymbol{ST}_i^{LxF} := \sum_{K\in K_i}\alpha_K(\boldsymbol{v}_i - \overline{\boldsymbol{v}}_K)$$

Last but not least an ODE integrator

Problem: mass matrix

$$\sum_{K \in K_i} \sum_{\boldsymbol{x}_j \in K} \left(\int_K \varphi_i \varphi_j d\boldsymbol{x} \right) \frac{d}{dt} \boldsymbol{v}_j(t)$$

$$= -g \sum_{K \in K_i} \left[\int_K \varphi_i \nabla_{\boldsymbol{x}} (H_h + B_h) d\boldsymbol{x} \right] - \sum_{K \in K_i} \boldsymbol{CT}_i^K - \boldsymbol{ST}_i$$

$$\mathcal{M} \frac{d}{dt} \boldsymbol{v} = \boldsymbol{r}$$

Solutions

- LO mass lumping
- DeC Remi[2] (problems for order 4 on)
- Spectral methods $\varphi_i \sim \boldsymbol{x}_i$ (1D GLB, 2D tensor product or Cubature)

[2] Abgrall, High order schemes for hyperbolic problems using globally continuous approximation and avoiding mass matrices, 2017

Price to pay: time-dependent mass matrix

$$\int_{K(t)} \varphi_i(\boldsymbol{x},t)\varphi_j(\boldsymbol{x},t)d\boldsymbol{x} = \int_{\widehat{K}} \widehat{\varphi}_i(\boldsymbol{\xi})\widehat{\varphi}_j(\boldsymbol{\xi})det\boldsymbol{J}(\boldsymbol{\xi},t)d\boldsymbol{\xi}$$
$$\approx \delta_{i,j}\widehat{\omega}_i det\boldsymbol{J}(\boldsymbol{\xi}_i,t)$$

$$\int_{K(t)} H_h(\boldsymbol{x},t)\varphi_i(\boldsymbol{x},t)\varphi_j(\boldsymbol{x},t)d\boldsymbol{x} = \int_{\widehat{K}} H_h(\boldsymbol{\xi},t)\widehat{\varphi}_i(\boldsymbol{\xi})\widehat{\varphi}_j(\boldsymbol{\xi})det\boldsymbol{J}(\boldsymbol{\xi},t)d\boldsymbol{\xi}$$
$$\approx \delta_{i,j}\widehat{\omega}_i H_h(\boldsymbol{\xi}_i,t)det\boldsymbol{J}(\boldsymbol{\xi}_i,t)$$

However, only a point evaluation

Advantages

$$\frac{d}{dt}\boldsymbol{x} = \boldsymbol{v}, \quad \frac{d}{dt}\boldsymbol{v} = \boldsymbol{r}, \quad \text{SMC}$$

- Truly arbitrary high order (no problems as for Bernstein and β-limiting)
- Extendible to Euler
- Extendible to multi-D (quads, PGL, tensor products, Cubature)
- Whatever time integration method.

Section 3

Numerics

Sod

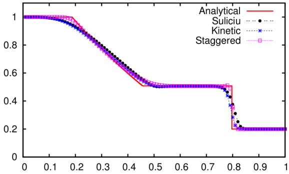
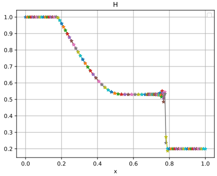
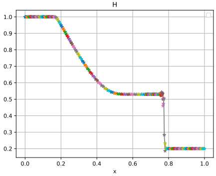


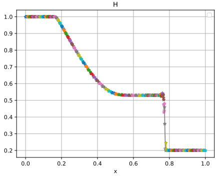
Figure: Reference



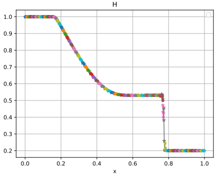
(a) order 2



(b) order 3



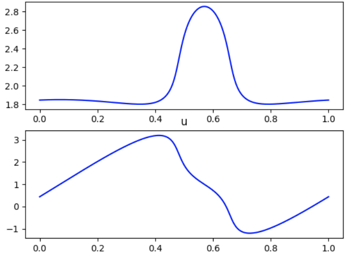
(c) order 4



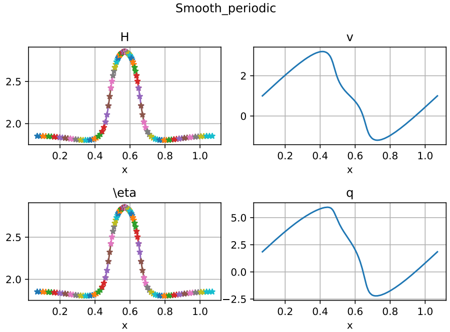
(d) order 5

Numerics, Convergence

$$\begin{cases} H = 2 + cos(2\pi x) \\ v = 1 \end{cases}$$



(a) Eulerian



(b) Lagrangian

Numerics, Convergence[3] $(2p)$

ORDER 2 P1P2

N	error x	Order x	Error v	Order v
50	2.850e-04	0.000	1.305e-02	0.000
100	7.166e-05	1.992	3.021e-03	2.111
200	1.798e-05	1.995	7.579e-04	1.995
400	4.504e-06	1.997	1.901e-04	1.996
800	1.127e-06	1.998	4.756e-05	1.999

ORDER 3 P2P3

N	error x	Order x	Error v	Order v
50	1.197e-06	0.000	9.768e-04	0.000
100	1.497e-07	2.999	4.086e-05	4.579
200	9.487e-09	3.980	4.666e-06	3.131
400	6.040e-10	3.973	3.973e-07	3.554
800	3.811e-11	3.986	2.878e-08	3.787

ORDER 4 P3P4 (final plateau probably due to the fact that H hit machine precision)

N	error x	Order x	Error v	Order v
50	2.696e-07	0.000	8.121e-05	0.000
100	7.205e-09	5.226	4.995e-06	4.023
200	7.689e-11	6.550	6.990e-08	6.159
400	1.060e-12	6.181	1.087e-09	6.007
800	1.739e-14	5.930	1.606e-10	2.758

ORDER 5 P4P5

N	error x	Order x	Error v	Order v
50	2.973e-08	0.000	3.642e-05	0.000
100	2.038e-10	7.188	2.641e-07	7.108
200	9.721e-13	7.712	1.196e-09	7.786

[3] Mantri, Oeffner, Ricchiuto, Fully well balanced entropy controlled DGSEM, 2022

Numerics, well-balancing

In particular (in Eulerian) $\frac{\partial}{\partial t}\boldsymbol{u} \equiv 0 \;\Leftrightarrow\; \frac{\partial}{\partial x}\boldsymbol{F}(\boldsymbol{u}) = \boldsymbol{S}(x, \boldsymbol{u})$

- Refine a lot the mesh $\Rightarrow$ Longer computational time
- Well-balancing

$$\boldsymbol{u}_{eq} := \begin{pmatrix} \eta - B \\ 0 \end{pmatrix}$$

$$\eta = H + B \equiv const$$

Figure: Lake of Zürich at rest

Lake at rest is exactly preserved

$$\frac{d}{dt}\boldsymbol{x} = \boldsymbol{v}, \quad \frac{d}{dt}\boldsymbol{v} = \boldsymbol{r}, \quad \text{SMC}$$

$$H(\boldsymbol{x}(\boldsymbol{\xi}, t), t) = \frac{\widehat{H}(\boldsymbol{\xi})}{det\boldsymbol{J}(\boldsymbol{\xi}, t)}, \quad \boldsymbol{J}(\boldsymbol{\xi}, t) = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi}, t)$$

$$\sum_{K \in K_i} \sum_{\boldsymbol{x}_j \in K} \left(\int_K \varphi_i \varphi_j d\boldsymbol{x} \right) \frac{d}{dt}\boldsymbol{v}_j(t)$$
$$= -g \sum_{K \in K_i} \left[\int_K \varphi_i \nabla_{\boldsymbol{x}} (H_h + B_h) d\boldsymbol{x} \right] - \sum_{K \in K_i} \boldsymbol{CT}_i^K - \boldsymbol{ST}_i$$

Numerics, well-balancing

$$\sum_{K\in K_i}\sum_{\boldsymbol{x}_j\in K}\left(\int_K \varphi_i\varphi_j d\boldsymbol{x}\right)\frac{d}{dt}\boldsymbol{v}_j(t)$$
$$= -g\sum_{K\in K_i}\left[\int_K \varphi_i \nabla_{\boldsymbol{x}}(H_h+B_h)d\boldsymbol{x}\right] - \sum_{K\in K_i}\boldsymbol{CT}_i^K - \boldsymbol{ST}_i$$

$$\boldsymbol{CT}_i^K = g\int_{\partial K}\varphi_i(\eta^* - \eta|_K)\boldsymbol{\nu}d\boldsymbol{\sigma}$$

$$\boldsymbol{ST}_i^{CIP} := \sum_f \alpha_f \left[\!\left[\frac{\partial}{\partial x}\varphi_i\right]\!\right]\left[\!\left[\frac{\partial}{\partial x}v\right]\!\right], \quad \boldsymbol{ST}_i^{LxF} := \sum_{K\in K_i}\alpha_K(\boldsymbol{v}_i - \overline{\boldsymbol{v}}_K)$$

Numerics, well-balancing

Non-smooth bathymetry, only C^0

$$\eta(x) = \begin{cases} \eta_{eq} + A \exp\left(1 - \frac{1}{1-\left(\frac{x-6}{0.5}\right)^2}\right) & 5.5 < x < 6.5 \\ \eta_{eq} & otherwise \end{cases}$$

with $A = 5 \cdot 10^{-5}$.

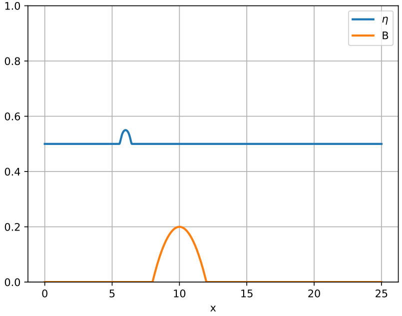
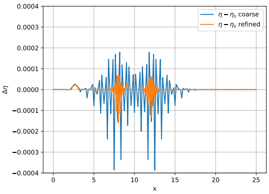


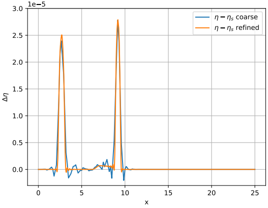
Figure: Perturbation amplified by 1000

Numerics, well-balancing

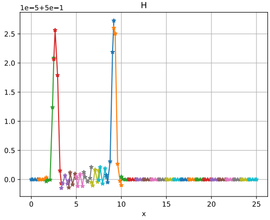
Order 5, 30 elements



(a) Eulerian, non WB

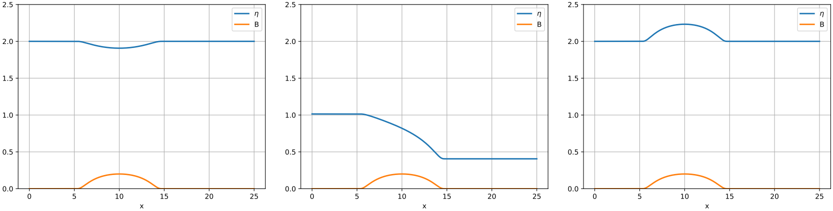


(b) Eulerian, WB



(c) Lagrangian

Numerics, smooth steady states



(a) Sub, $Hv \equiv 4.42$ (b) Trans, $Hv \equiv 1.53$ (c) Super, $Hv \equiv 24$

$$\boldsymbol{F}_x(\boldsymbol{u}) = \boldsymbol{S}(\boldsymbol{x}, \boldsymbol{u})$$

Numerics, supercritical (problem with order 3)

P1P2 -> order 2

N	L^1(v)	Order v	L^1(H)	Order H	L^1(q)	Order q	||v||_2	Order v	||H||_2	Order H
25	4.281e-02	0.000	1.837e-02	0.000	1.875e-01	0.000	3.024e-03	0.000	1.113e-03	0.000
50	1.021e-02	2.068	5.229e-03	1.813	4.787e-02	1.970	8.202e-04	1.882	3.348e-04	1.733
100	2.431e-03	2.070	9.872e-04	2.405	1.045e-02	2.196	1.884e-04	2.122	6.528e-05	2.359
200	4.775e-04	2.348	1.446e-04	2.771	1.928e-03	2.438	3.731e-05	2.336	1.091e-05	2.581
400	1.125e-04	2.086	2.981e-05	2.278	4.510e-04	2.096	8.973e-06	2.056	2.450e-06	2.155
800	2.793e-05	2.010	7.313e-06	2.027	1.126e-04	2.002	2.243e-06	2.000	6.048e-07	2.018

P2P3 -> order 3 (instead I get 2)

N	L^1(v)	Order v	L^1(H)	Order H	L^1(q)	Order q	||v||_2	Order v	||H||_2	Order H
25	9.342e-03	0.000	3.326e-03	0.000	3.702e-02	0.000	9.183e-04	0.000	3.003e-04	0.000
50	1.945e-03	2.264	7.998e-04	2.056	1.033e-02	1.841	1.574e-04	2.545	7.969e-05	1.914
100	4.239e-04	2.198	1.947e-04	2.038	2.180e-03	2.244	3.369e-05	2.224	1.590e-05	2.325
200	8.308e-05	2.351	3.091e-05	2.655	3.042e-04	2.841	6.592e-06	2.354	2.444e-06	2.702
400	1.833e-05	2.180	5.953e-06	2.376	4.774e-05	2.672	1.523e-06	2.114	4.756e-07	2.361
800	4.502e-06	2.026	1.455e-06	2.033	1.120e-05	2.092	3.758e-07	2.019	1.150e-07	2.048

P3P4 -> order 4

N	L^1(v)	Order v	L^1(H)	Order H	L^1(q)	Order q	||v||_2	Order v	||H||_2	Order H
25	3.877e-03	0.000	1.475e-03	0.000	1.573e-02	0.000	5.056e-04	0.000	2.086e-04	0.000
50	7.459e-04	2.378	3.024e-04	2.286	3.726e-03	2.078	6.793e-05	2.896	3.157e-05	2.724
100	9.525e-05	2.969	5.184e-05	2.544	6.887e-04	2.436	7.144e-06	3.249	4.526e-06	2.802
200	9.417e-06	3.338	4.700e-06	3.463	6.982e-05	3.302	6.768e-07	3.400	3.857e-07	3.553
400	3.955e-07	4.574	1.974e-07	4.573	2.982e-06	4.549	2.757e-08	4.618	1.556e-08	4.632

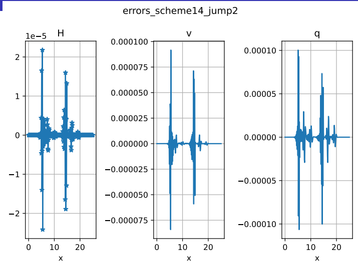
P4P5 -> order 5

N	L^1(v)	Order v	L^1(H)	Order H	L^1(q)	Order q	||v||_2	Order v	||H||_2	Order H
25	2.130e-03	0.000	8.068e-04	0.000	7.579e-03	0.000	2.959e-04	0.000	1.281e-04	0.000
50	3.021e-04	2.818	1.253e-04	2.687	1.390e-03	2.447	2.776e-05	3.414	2.241e-05	2.515
100	2.822e-05	3.420	1.237e-05	3.340	1.310e-04	3.407	2.418e-06	3.521	2.042e-06	3.456
200	1.202e-06	4.553	5.293e-07	4.547	6.777e-06	4.273	9.730e-08	4.635	7.038e-08	4.859

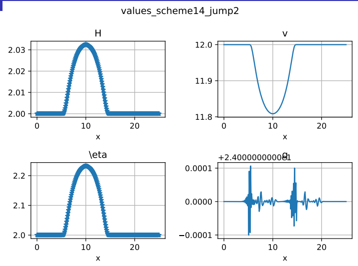
Numerics, debugging in progress

- Burman (no changes)
- SSPRK4 (no changes)
- one time-step (correct order)
- Exact v (OK)
- Exact H (Not OK, so the problem must be in the update of v)
- Checking the code

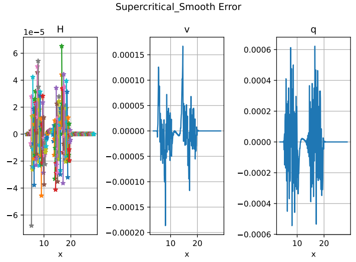
Numerics, shape of the error



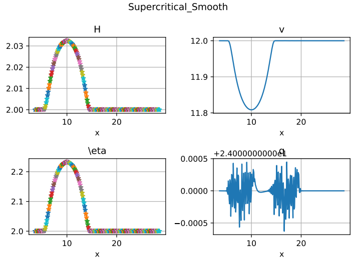
(a) Error Eulerian



(b) Values Eulerian

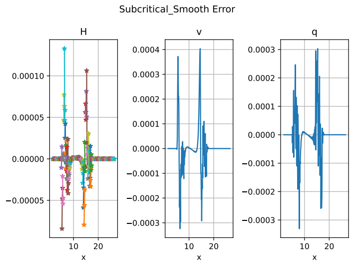


(c) Error Lagrangian

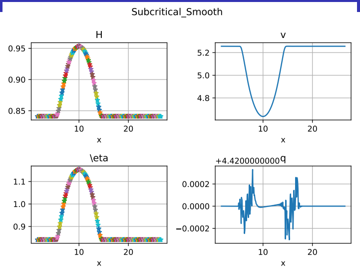


(d) Values Lagrangian

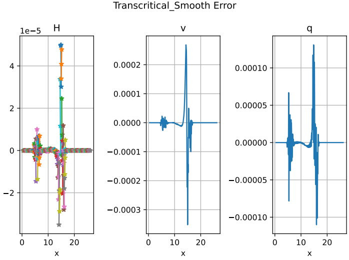
Numerics, shape of the error



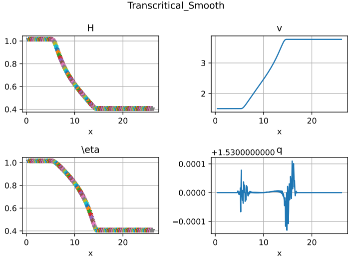
(a) Error Subcritical



(b) Values Subcritical



(c) Error Transcritical



(d) Values Transcritical