Finite-Volume Diffusion Schemes for Svard's Eulerian Governing Equations

Emmett Padway

January 10th, 2024

NASA Langley Research Center

Outline

- Background/motivation

Outline

- Background/motivation
- Governing equations and discretization

Outline

- Background/motivation
- Governing equations and discretization
- Verification

Outline

- Background/motivation
- Governing equations and discretization
- Verification
- Results

Outline

- Background/motivation
- Governing equations and discretization
- Verification
- Results
- Conclusions and future work

Background/Motivation

- Magnus Svard put forth an argument for an alternative set of governing equations for fluid flow

Background/Motivation

- Magnus Svard put forth an argument for an alternative set of governing equations for fluid flow
- This derivation is done entirely using diffusion from an Eulerian perspective

Background/Motivation

- Magnus Svard put forth an argument for an alternative set of governing equations for fluid flow
- This derivation is done entirely using diffusion from an Eulerian perspective
- This set of governing equations allows for arguments of well-posedness and existence of solutions

Background/Motivation

- Magnus Svard put forth an argument for an alternative set of governing equations for fluid flow
- This derivation is done entirely using diffusion from an Eulerian perspective
- This set of governing equations allows for arguments of well-posedness and existence of solutions
- The result is that the viscous flux has only normal terms and it was believed this would lead to a simpler viscous dissipation matrix and hyperbolic diffusion system.

Background/Motivation

- Magnus Svard put forth an argument for an alternative set of governing equations for fluid flow
- This derivation is done entirely using diffusion from an Eulerian perspective
- This set of governing equations allows for arguments of well-posedness and existence of solutions
- The result is that the viscous flux has only normal terms and it was believed this would lead to a simpler viscous dissipation matrix and hyperbolic diffusion system.
- In this work, I compare a typical edge-based finite-volume diffusion discretization and a hyperbolic diffusion discretization

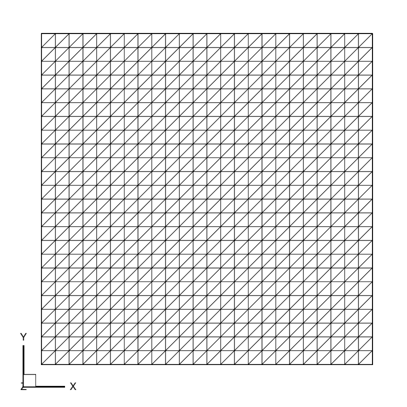
Governing Equations and Discretization

Governing Equations and Discretization

We are considering a 2D unsteady hyperbolic system defined as:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_x}{\partial x} + \frac{\partial \mathbf{f}_y}{\partial y} = \mathbf{s}(x, y, t), \tag{1}$$

solved on a representative 2D grid.

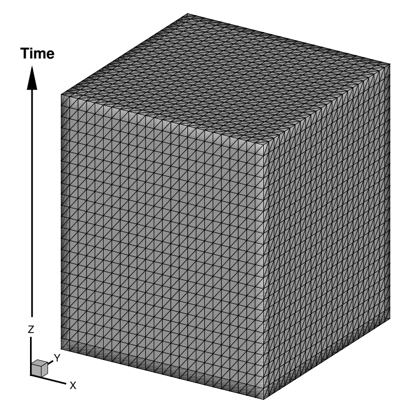


Governing Equations and Discretization

We then transform them to the space-time as follows:

$$\frac{\partial \mathbf{u}}{\partial z} + \frac{\partial \mathbf{f}_x}{\partial x} + \frac{\partial \mathbf{f}_y}{\partial y} = \mathbf{s}(x, y, z), \qquad (2)$$

solved on the extruded space-time grid below.



Governing Equations and Discretization: Navier Stokes

Compressible NS equations:

$$\begin{aligned}
\partial_t \rho + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v}) &= 0 \\
\partial_t (\rho \mathbf{v}) + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v} \otimes \mathbf{v}) &= -\nabla_{\mathbf{x}} p + \nabla_{\mathbf{x}} \cdot \tau \\
\partial_t (E) + \nabla_{\mathbf{x}} \cdot (E\mathbf{v} + p\mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\tau \mathbf{v}) - \nabla_{\mathbf{x}} \cdot \mathbf{q}
\end{aligned} \tag{3}$$

where t is the physical time.

The viscous stress tensor $\boldsymbol{\tau}$ is given by

$$\tau = -\frac{2}{3}\mu(\nabla_{\mathbf{x}} \cdot \mathbf{v})\mathbf{I} + \mu\left(\nabla_{\mathbf{x}}\mathbf{v} + (\nabla_{\mathbf{x}}\mathbf{v})^T\right). \tag{4}$$

Governing Equations and Discretization: Hyperbolic Navier Stokes

Compressible HNS equations:

$$
\begin{aligned}
\partial_\tau \rho + \partial_t \rho + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\nu_r \mathbf{r}) \\
\partial_\tau (\rho \mathbf{v}) + \partial_t (\rho \mathbf{v}) + \operatorname{div}(\rho \mathbf{v} \otimes \mathbf{v}) &= -\nabla_{\mathbf{x}} p + \nabla_{\mathbf{x}} \cdot (\mu_v \widetilde{\tau}) \\
\partial_\tau (\rho E) + \partial_t (\rho E) + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v} H) &= \nabla_{\mathbf{x}} \cdot (\mu_v \widetilde{\tau} \mathbf{v}) + \nabla_{\mathbf{x}} \cdot \left(\frac{\mu_h}{\gamma(\gamma - 1)} \mathbf{h} \right) \\
T_r \partial_\tau \mathbf{r} &= \nabla_{\mathbf{x}} \rho - \mathbf{r} \\
T_v \partial_\tau \mathbf{g} &= \nabla_{\mathbf{x}} \mathbf{v} - \mathbf{g} \\
T_h \partial_\tau \mathbf{h} &= \nabla_{\mathbf{x}} T - \mathbf{h}
\end{aligned}
\tag{5}
$$

where τ is a pseudotime variable, and $\widetilde{\boldsymbol{\tau}} = -\frac{1}{2} tr(\mathbf{g})\mathbf{I} + \frac{3}{4}\left(\mathbf{g} + \mathbf{g}^T\right)$,

$$
T_r = \frac{L_r^2}{\nu_r},\ T_v = \frac{L_v^2}{\nu_v},\ T_h = \frac{L_h^2}{\nu_h},\ \nu_r = V_{min}^{4/3} M_\infty, \nu_v = \frac{4\mu}{3\rho},\ \nu_h = \frac{\gamma\mu}{\rho Pr}, \tag{6}
$$

Governing Equations and Discretization: Hyperbolic Navier Stokes

Compressible HNS equations:

$$
\begin{aligned}
\partial_\tau \rho + \partial_t \rho + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\nu_r \mathbf{r}) \\
\partial_\tau (\rho \mathbf{v}) + \partial_t (\rho \mathbf{v}) + \mathrm{div}(\rho \mathbf{v} \otimes \mathbf{v}) &= -\nabla_{\mathbf{x}} p + \nabla_{\mathbf{x}} \cdot (\mu_v \widetilde{\tau}) \\
\partial_\tau (\rho E) + \partial_t (\rho E) + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v} H) &= \nabla_{\mathbf{x}} \cdot (\mu_v \widetilde{\tau} \mathbf{v}) + \nabla_{\mathbf{x}} \cdot \left(\frac{\mu_h}{\gamma(\gamma - 1)} \mathbf{h} \right) \\
T_r \partial_\tau \mathbf{r} &= \nabla_{\mathbf{x}} \rho - \mathbf{r} \\
T_v \partial_\tau \mathbf{g} &= \nabla_{\mathbf{x}} \mathbf{v} - \mathbf{g} \\
T_h \partial_\tau \mathbf{h} &= \nabla_{\mathbf{x}} T - \mathbf{h}
\end{aligned}
\tag{7}
$$

where τ is a pseudotime variable, and $\widetilde{\boldsymbol{\tau}} = -\frac{1}{2} tr(\mathbf{g})\mathbf{I} + \frac{3}{4}\left(\mathbf{g} + \mathbf{g}^T\right)$,

$$
T_r = \frac{L_r^2}{\nu_r},\ T_v = \frac{L_v^2}{\nu_v},\ T_h = \frac{L_h^2}{\nu_h},\ \nu_r = V_{min}^{4/3} M_\infty, \nu_v = \frac{4\mu}{3\rho},\ \nu_h = \frac{\gamma\mu}{\rho Pr}, \tag{8}
$$

Governing Equations and Discretization: Svard-Eulerian equations

Svard's Eulerian equations:

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\nu \nabla_{\mathbf{x}} \rho), \\
\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v} \otimes \mathbf{v}) + \nabla_{\mathbf{x}} p &= \nabla_{\mathbf{x}} \cdot (\nu \nabla_{\mathbf{x}} \rho \mathbf{v}), \\
\frac{\partial E}{\partial t} + \nabla_{\mathbf{x}} \cdot (E \mathbf{v} + p \mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\nu \nabla_{\mathbf{x}} E) + \nabla_{\mathbf{x}} \cdot (\kappa \nabla_{\mathbf{x}} T), \\
p &= \rho R T, \\
\nu &= \alpha \frac{\mu}{\rho} + \beta(\rho, T).
\end{aligned} \tag{9}
$$

Governing Equations and Discretization: Svard-Eulerian equations

Svard's Eulerian equations:

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\nu \nabla_{\mathbf{x}} \rho), \\
\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v} \otimes \mathbf{v}) + \nabla_{\mathbf{x}} p &= \nabla_{\mathbf{x}} \cdot (\nu \nabla_{\mathbf{x}} \rho \mathbf{v}), \\
\frac{\partial E}{\partial t} + \nabla_{\mathbf{x}} \cdot (E \mathbf{v} + p \mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\nu \nabla_{\mathbf{x}} E), \\
p &= \rho R T, \\
\nu &= \alpha \frac{\mu}{\rho} + \beta(\rho, T).
\end{aligned}
\tag{10}
$$

Governing Equations and Discretization: Hyperbolic Eulerian Flow

Compressible HEF equations:

$$
\begin{aligned}
\partial_\tau \rho + \partial_t \rho + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\nu \mathbf{r}) \\
\partial_\tau (\rho \mathbf{v}) + \partial_t (\rho \mathbf{v}) + \operatorname{div}(\rho \mathbf{v} \otimes \mathbf{v}) &= -\nabla_{\mathbf{x}} p + \nabla_{\mathbf{x}} \cdot \nu \left(\mathbf{vr} + \rho \mathbf{g}\right) \\
\partial_\tau (\rho E) + \partial_t (\rho E) + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v} H) &= \nabla_{\mathbf{x}} \cdot (\nu \mathbf{k}) \\
T_r \partial_\tau \mathbf{r} &= \nabla_{\mathbf{x}} \rho - \mathbf{r} \\
T_v \partial_\tau \mathbf{g} &= \nabla_{\mathbf{x}} \mathbf{v} - \mathbf{g} \\
T_h \partial_\tau \mathbf{k} &= \nabla_{\mathbf{x}} E - \mathbf{k}
\end{aligned}
\tag{11}
$$

where τ is a pseudotime variable and

$$
T_r = T_v = T_h = \frac{L_h^2}{\nu} \tag{12}
$$

Governing Equations and Discretization: Hyperbolic Eulerian Flow

Compressible HEF equations:

$$
\begin{aligned}
\partial_\tau \rho + \partial_t \rho + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v}) &= \nabla_{\mathbf{x}} \cdot (\nu \mathbf{r}) \\
\partial_\tau (\rho \mathbf{v}) + \partial_t (\rho \mathbf{v}) + \operatorname{div}(\rho \mathbf{v} \otimes \mathbf{v}) &= -\nabla_{\mathbf{x}} p + \nabla_{\mathbf{x}} \cdot \nu \left(\mathbf{v}\mathbf{r} + \rho \mathbf{g} \right) \\
\partial_\tau (\rho E) + \partial_t (\rho E) + \nabla_{\mathbf{x}} \cdot (\rho \mathbf{v} H) &= \nabla_{\mathbf{x}} \cdot (\nu \mathbf{k}) \\
T_r \partial_\tau \mathbf{r} &= \nabla_{\mathbf{x}} \rho - \mathbf{r} \\
T_v \partial_\tau \mathbf{g} &= \nabla_{\mathbf{x}} \mathbf{v} - \mathbf{g} \\
T_h \partial_\tau \mathbf{k} &= \nabla_{\mathbf{x}} E - \mathbf{k}
\end{aligned}
\tag{13}
$$

where τ is a pseudotime variable and

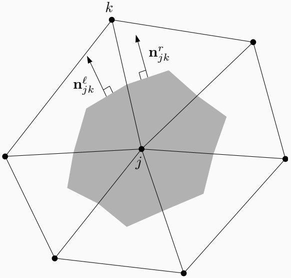
$$
T_r = T_v = T_h = \frac{L_h^2}{\nu} \tag{14}
$$

Governing Equations and Discretization: Residual

We discretize the governing equations on a tetrahedral grid where the residual at node j is defined as:

$$\sum_{k \in \{k_j\}} \mathbf{\Phi}_{jk} A_{jk} = s(x_j, y_j, z_j), \tag{15}$$

where Φ_{jk} is the numerical flux and A_{jk} is the directed area vector on the edge that connects nodes j and k.



Stencil for the edge-based (EB) discretization, showing the directed area vector on edge jk

We extend this to be 2^{nd}-order accurate using the U-MUSCL scheme

$$\mathbf{w}_L = \kappa\frac{\mathbf{w}_j + \mathbf{w}_k}{2} + (1-\kappa)\left[\mathbf{w}_j + \nabla\mathbf{w}_j^{LSQ}\cdot(\mathbf{x}_k - \mathbf{x}_j)\right], \quad (16)$$

$$\mathbf{w}_R = \kappa\frac{\mathbf{w}_j + \mathbf{w}_k}{2} + (1-\kappa)\left[\mathbf{w}_k - \nabla\mathbf{w}_k^{LSQ}\cdot(\mathbf{x}_k - \mathbf{x}_j)\right], \quad (17)$$

with $\kappa = \frac{1}{2}$.

Governing Equations and Discretization: Fluxes

The numerical flux is defined as:

$$\mathbf{\Phi}_{jk} \quad = \quad \mathbf{\Phi}_{jk}^{inv}|(\hat{\hat{n}}_x, \hat{\hat{n}}_y)| + \mathbf{\Phi}_{jk}^{vis}|(\hat{\hat{n}}_x, \hat{\hat{n}}_y)| + \mathbf{\Phi}_{jk}^{time}|\hat{\hat{n}}_t|, \tag{18}$$

where the norm $n_{jk} = (n_x, n_y, n_t)$ is normalized in space-time as $\hat{\mathbf{n}}_{jk} = \mathbf{n}_{jk}/|\mathbf{n}_{jk}| = (\hat{n}_x, \hat{n}_y, \hat{n}_t)$.

Governing Equations and Discretization: Fluxes

The numerical flux is defined as:

$$\mathbf{\Phi}_{jk} \quad = \quad \mathbf{\Phi}_{jk}^{inv}|(\hat{\hat{n}}_x, \hat{\hat{n}}_y)| + \mathbf{\Phi}_{jk}^{vis}|(\hat{\hat{n}}_x, \hat{\hat{n}}_y)| + \mathbf{\Phi}_{jk}^{time}|\hat{\hat{n}}_t|, \tag{18}$$

where the norm $n_{jk} = (n_x, n_y, n_t)$ is normalized in space-time as $\hat{\mathbf{n}}_{jk} = \mathbf{n}_{jk}/|\mathbf{n}_{jk}| = (\hat{n}_x, \hat{n}_y, \hat{n}_t)$.

We use a $\hat{\hat{}}$ notation to indicate normalization in space or time alone. For space this is:

$$\hat{\hat{\mathbf{n}}}_{xy} = \frac{\hat{\mathbf{n}}_{xy}}{|\hat{\mathbf{n}}_{xy}|} = \frac{(\hat{n}_x, \hat{n}_y)}{|(\hat{n}_x, \hat{n}_y)|} = \frac{(\hat{n}_x, \hat{n}_y)}{\sqrt{\hat{n}_x^2 + \hat{n}_y^2}}, \tag{19}$$

Governing Equations and Discretization: Fluxes

The numerical flux is defined as:

$$\boldsymbol{\Phi}_{jk} \quad = \quad \boldsymbol{\Phi}_{jk}^{inv}|(\hat{\hat{n}}_x, \hat{\hat{n}}_y)| + \boldsymbol{\Phi}_{jk}^{vis}|(\hat{\hat{n}}_x, \hat{\hat{n}}_y)| + \boldsymbol{\Phi}_{jk}^{time}|\hat{\hat{n}}_t|, \tag{18}$$

where the norm $n_{jk} = (n_x, n_y, n_t)$ is normalized in space-time as $\hat{\mathbf{n}}_{jk} = \mathbf{n}_{jk}/|\mathbf{n}_{jk}| = (\hat{n}_x, \hat{n}_y, \hat{n}_t)$.

We use a $\hat{\hat{}}$ notation to indicate normalization in space or time alone. For space this is:

$$\hat{\hat{\mathbf{n}}}_{xy} = \frac{\hat{\mathbf{n}}_{xy}}{|\hat{\mathbf{n}}_{xy}|} = \frac{(\hat{n}_x, \hat{n}_y)}{|(\hat{n}_x, \hat{n}_y)|} = \frac{(\hat{n}_x, \hat{n}_y)}{\sqrt{\hat{n}_x^2 + \hat{n}_y^2}}, \tag{19}$$

and for time it is:

$$\hat{\hat{n}}_t = \frac{\hat{n}_t}{|\hat{n}_t|}. \tag{20}$$

Governing Equations and Discretization: Fluxes

We use Roe's flux for the invisicid flux

$$\boldsymbol{\Phi}_{jk}^{inv} = \frac{1}{2}\left[\mathbf{f}^{inv}(\mathbf{w}_L) + \mathbf{f}^{inv}(\mathbf{w}_R)\right] - \frac{1}{2}\left|\mathbf{A}^{inv}(\mathbf{w}_L, \mathbf{w}_R)\right|(\mathbf{u}_R - \mathbf{u}_L)\,, \qquad (21)$$

Governing Equations and Discretization: Fluxes

We use Roe's flux for the invisicid flux

$$\boldsymbol{\Phi}_{jk}^{inv} = \frac{1}{2}\left[\mathbf{f}^{inv}(\mathbf{w}_L) + \mathbf{f}^{inv}(\mathbf{w}_R)\right] - \frac{1}{2}\left|\mathbf{A}^{inv}(\mathbf{w}_L, \mathbf{w}_R)\right|(\mathbf{u}_R - \mathbf{u}_L), \qquad (21)$$

an upwind temporal flux

$$\boldsymbol{\Phi}_{jk}^{time} = \frac{1}{2}\left[\mathbf{u}_L + \mathbf{u}_R\right]\hat{\hat{n}}_t - \frac{|\hat{\hat{n}}_t|}{2}(\mathbf{u}_R - \mathbf{u}_L), \qquad (22)$$

Governing Equations and Discretization: Fluxes

We use Roe's flux for the invisicid flux

$$\mathbf{\Phi}_{jk}^{inv} = \frac{1}{2}\left[\mathbf{f}^{inv}(\mathbf{w}_L) + \mathbf{f}^{inv}(\mathbf{w}_R)\right] - \frac{1}{2}\left|\mathbf{A}^{inv}(\mathbf{w}_L, \mathbf{w}_R)\right|(\mathbf{u}_R - \mathbf{u}_L)\,, \tag{21}$$

an upwind temporal flux

$$\mathbf{\Phi}_{jk}^{time} = \frac{1}{2}\left[\mathbf{u}_L + \mathbf{u}_R\right]\hat{\hat{n}}_t - \frac{|\hat{\hat{n}}_t|}{2}(\mathbf{u}_R - \mathbf{u}_L)\,, \tag{22}$$

and use an upwind flux for the viscous flux making it hyperbolic

$$\mathbf{\Phi}_{jk}^{vis} = \frac{1}{2}\left[\mathbf{f}^{vis}(\mathbf{w}_L) + \mathbf{f}^{vis}(\mathbf{w}_R)\right] - \frac{1}{2}\left|\mathbf{A}^{vis}(\mathbf{w}_L, \mathbf{w}_R)\right|(\mathbf{u}_R - \mathbf{u}_L) \tag{23}$$

We use the HNS20G formulation for both the NS and EF equation viscous fluxes, for NS this is:

$$\phi_{jk}^{vis} = \begin{bmatrix} 0 \\ -\mu_v \left[\widetilde{\tau}_{xx}\hat{\hat{n}}_x + \widetilde{\tau}_{yx}\hat{\hat{n}}_y\right] \\ -\mu_v \left[\widetilde{\tau}_{xy}\hat{\hat{n}}_x + \widetilde{\tau}_{yy}\hat{\hat{n}}_y\right] \\ -\mu_v \left[\left(\vec{\widetilde{\tau}}_x \vec{u} + \frac{\mu_h h_x}{\gamma(\gamma-1)}\right)\hat{\hat{n}}_x + \left(\vec{\widetilde{\tau}}_y \vec{u} + \frac{\mu_h h_y}{\gamma(\gamma-1)}\right)\hat{\hat{n}}_y\right] \\ -\rho\hat{\hat{n}}_x \\ -\rho\hat{\hat{n}}_y \\ -u\hat{\hat{n}}_x \\ -u\hat{\hat{n}}_y \\ -v\hat{\hat{n}}_x \\ -v\hat{\hat{n}}_y \\ -T\hat{\hat{n}}_x \\ -T\hat{\hat{n}}_y \end{bmatrix} \tag{24}$$

Governing Equations and Discretization: Fluxes

In contrast, the HEF flux is:

$$\phi_{jk}^{vis} = \begin{bmatrix} -\nu(r_x n) \\ -\nu\left[\rho(u_x n) + (r_x n)u\right] \\ -\nu\left[\rho(v_x n) + (r_x n)v\right] \\ -\nu(E_x n) \\ -\rho\hat{\hat{n}}_x \\ -\rho\hat{\hat{n}}_y \\ -u\hat{\hat{n}}_x \\ -u\hat{\hat{n}}_y \\ -v\hat{\hat{n}}_x \\ -v\hat{\hat{n}}_y \\ -E\hat{\hat{n}}_x \\ -E\hat{\hat{n}}_y \end{bmatrix} \tag{25}$$

Where we define the following variables $u_x n = u_x\hat{\hat{n}}_x + u_y\hat{\hat{n}}_y$, $v_x n = v_x\hat{\hat{n}}_x + v_y\hat{\hat{n}}_y$, $r_x n = r_x\hat{\hat{n}}_x + r_y\hat{\hat{n}}_y$, $E_x n = E_x\hat{\hat{n}}_x + E_y\hat{\hat{n}}_y$.

Governing Equations and Discretization: Dissipation

By diagonalizing the matrix and using the local preconditioning approach we get a dissipation matrix of the form:

$$|PA| = \begin{bmatrix} a\nu & 0 & 0 \\ 0 & \frac{\nu nx^2}{a\nu} & \frac{\nu nxny}{a\nu} \\ 0 & \frac{\nu nxnxy}{a\nu} & \frac{\nu ny^2}{a\nu} \\ \dots & \dots & \dots \end{bmatrix} \tag{26}$$

where $a\nu = \sqrt{\frac{\nu}{T_h}}$. Note that this (reduced) matrix is block diagonal. Additionally, it lacks the additional dissipation vectors HNS20G requires to maintain strong coupling for design-order accurate computation of the velocity gradients.

Verification

Verification: Exponential Solution

We used the method of manufactured solutions (MMS) with an exponential solution to calculate the truncation and discretization errors. The solution is:

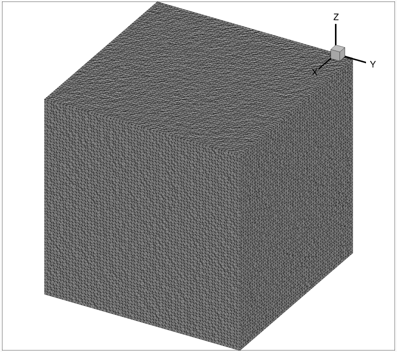
$$a = a_0 + a_{scale} exp(a_x x + a_y y + a_t t), \tag{27}$$

and the parameters to define it are below.

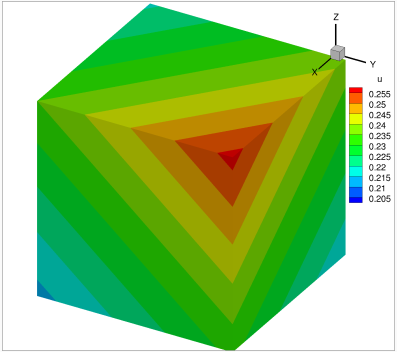
variable	a_0	a_{scale}	a_x	a_y	a_t
ρ	.5	1.0	0.525	0.550	0.575
u	.1	0.1	0.125	0.150	0.175
v	.2	0.2	0.225	0.250	0.275
p	.4	0.714	0.425	0.450	0.475

Parameters used to define the exact solution.

Verification: Exponential Solution



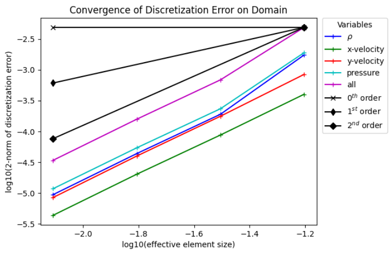
(a) Coarsest perturbed mesh



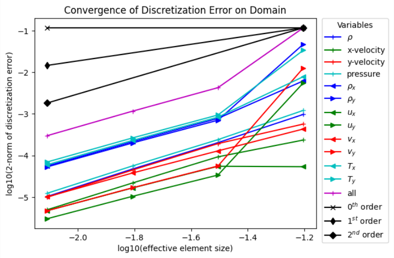
(b) Solution on coarsest mesh

Example mesh and solution.

Verification: Exponential Solution Discretization Error



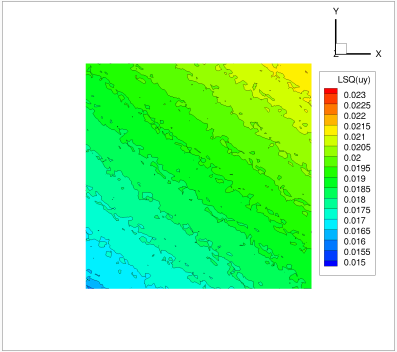
(a) Alpha damping discretization error convergence



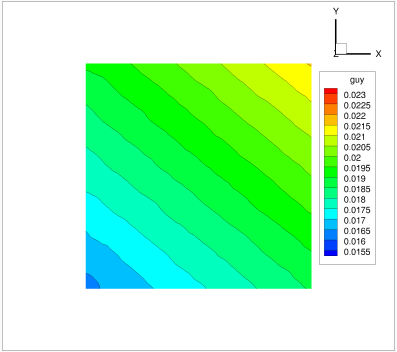
(b) HEF discretization error convergence

Discretization error convergence.

Verification: Exponential Solution



(a) $LSQ(u_x)$ on a 32^3-node grid.



(b) $g_{ux}(= u_x)$ on a 32^3-node grid.

Gradient calculation on arbitrary perturbed mesh

Results

Results: Boundary Layer Solution

- We used the FUN3D sketch to solution framework (combining EGADS, *refine*, and ruby scripts).

Results: Boundary Layer Solution

- We used the FUN3D sketch to solution framework (combining EGADS, *refine*, and ruby scripts).
- This entailed 10 adaptation cycles with mesh size doubling every other iteration.

Results: Boundary Layer Solution

- We used the FUN3D sketch to solution framework (combining EGADS, *refine*, and ruby scripts).
- This entailed 10 adaptation cycles with mesh size doubling every other iteration.
- The Mach Hessian metric computed by *refine* off the alpha-damping viscous discretization

Results: Boundary Layer Solution

- We used the FUN3D sketch to solution framework (combining EGADS, *refine*, and ruby scripts).
- This entailed 10 adaptation cycles with mesh size doubling every other iteration.
- The Mach Hessian metric computed by *refine* off the alpha-damping viscous discretization
- The final HEF results shown are from solving the hyperbolic equations on the final adapted mesh

Results: Boundary Layer Solution

The exact solution is defined by:

$$\mathbf{w}_{exact} = \begin{bmatrix} 1.0 \\ 1.0 \\ 0.1 \\ 1.0 \end{bmatrix} - \begin{bmatrix} 0.1 \\ 1.0 \\ 0.1 \\ 0.1 \end{bmatrix} \exp(-\eta), \quad \eta = y\sqrt{\frac{R(z)}{x - 0.2}}, \tag{28}$$

Results: Boundary Layer Solution

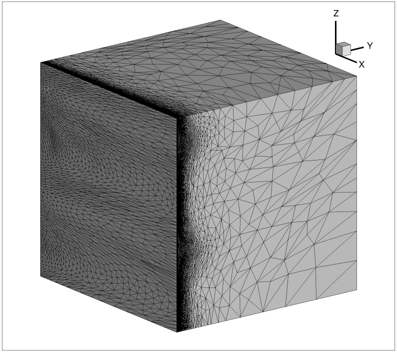
The exact solution is defined by:

$$\mathbf{w}_{exact} = \begin{bmatrix} 1.0 \\ 1.0 \\ 0.1 \\ 1.0 \end{bmatrix} - \begin{bmatrix} 0.1 \\ 1.0 \\ 0.1 \\ 0.1 \end{bmatrix} \exp(-\eta), \quad \eta = y\sqrt{\frac{R(z)}{x-0.2}}, \tag{28}$$

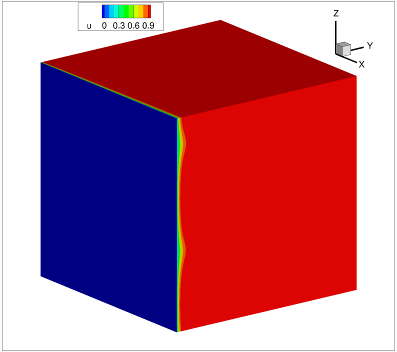
where $R(z)$ is a time-dependent parameter defined by

$$R(z) = 10^5 \left[1 + 0.75 \sin(4\pi z)\right]. \tag{29}$$

Results: Boundary Layer Solution - Mesh and Solution



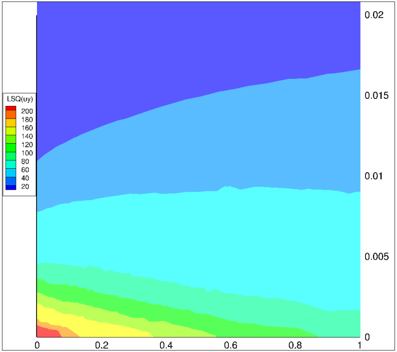
(a) Grid.



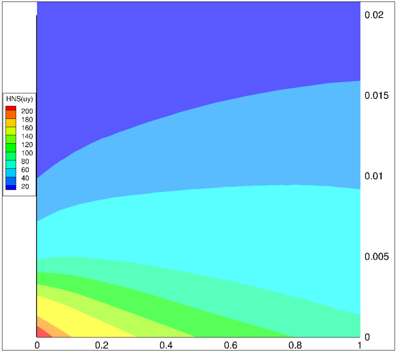
(b) X-velocity contours.

Grid and solution for MMS boundary layer test case.

Results: Boundary Layer Solution - Gradient Comparison



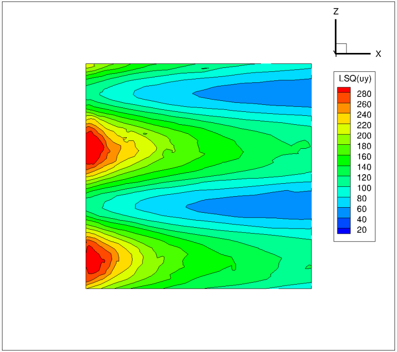
(a) Alpha: $LSQ(u_y)$ at $z = 0.619$.



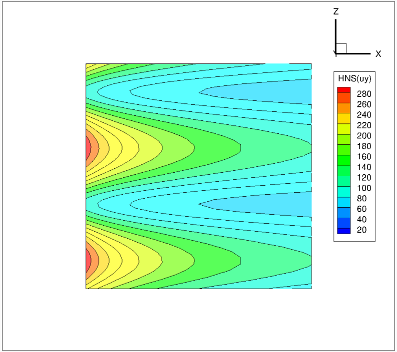
(b) HNS: g_{uy} at $z = 0.619$.

Wall normal gradient contours on XY plane for $z = 0.619$

Results: Boundary Layer Solution - Gradient Comparison



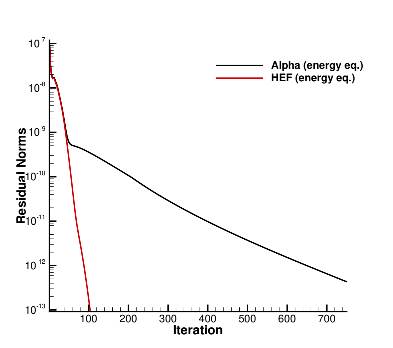
(a) Alpha: $LSQ(u_y)$ at $y = 0$.



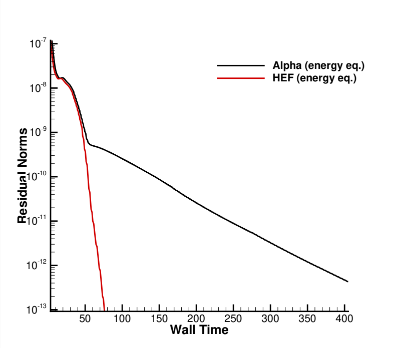
(b) HNS: g_{uy} at $y = 0$.

Wall normal gradient contours on XZ plane for $y = 0.0$

Results: Boundary Layer Solution - Time to Solution Comparison



(a) Iterations to solution



(b) CPU time to soluion

Comparison of CPU time and iterations to convergence for boundary layer MMS.

Results: Infinite Cylinder

- Infinite cylinder in $M = 0.2$, $Re = 200$ crossflow.
- Once again, used the FUN3D sketch to solution framework.

Results: Infinite Cylinder

- Infinite cylinder in $M = 0.2$, $Re = 200$ crossflow.
- Once again, used the FUN3D sketch to solution framework.
- This entailed 18 adaptation cycles with mesh size doubling every four iterations.

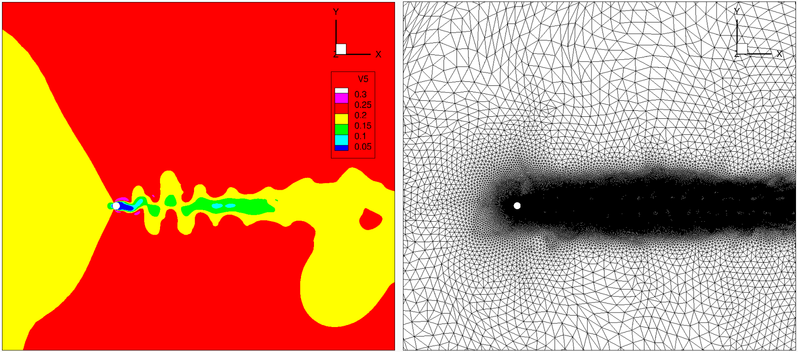
Results: Infinite Cylinder

- Infinite cylinder in $M = 0.2$, $Re = 200$ crossflow.
- Once again, used the FUN3D sketch to solution framework.
- This entailed 18 adaptation cycles with mesh size doubling every four iterations.
- The Mach Hessian metric computed by *refine*

Results: Infinite Cylinder

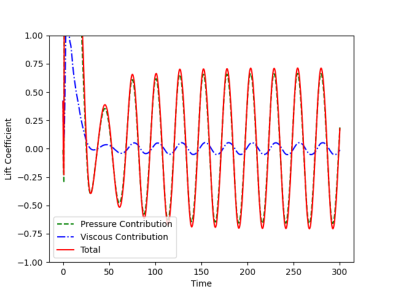
- Infinite cylinder in $M = 0.2$, $Re = 200$ crossflow.
- Once again, used the FUN3D sketch to solution framework.
- This entailed 18 adaptation cycles with mesh size doubling every four iterations.
- The Mach Hessian metric computed by *refine*
- We look at the 13^{th} mesh ($26.5M$ nodes) and compare the results of HEF discretization to those of the alpha-damping one.

Results: Infinite Cylinder - Solution Evolution

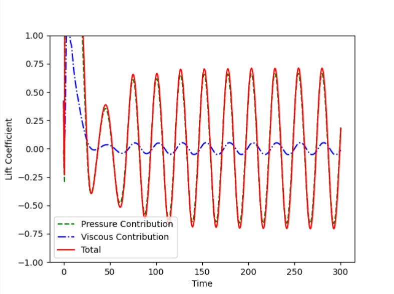


Solution for 13^{th} spatiotemporal triangulation

Results: Infinite Cylinder - Comparison of Lift Coefficient



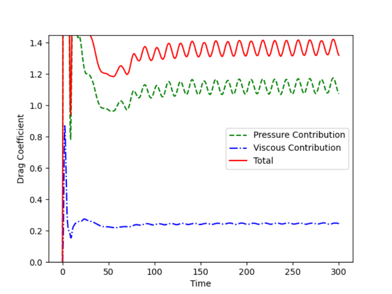
(a) Alpha damping.



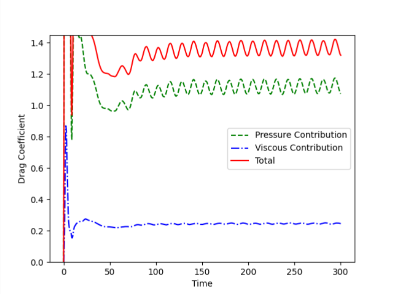
(b) HEF.

Lift coefficient vs. time

Results: Infinite Cylinder - Comparison of Drag Coefficient



(a) Alpha damping.



(b) HEF.

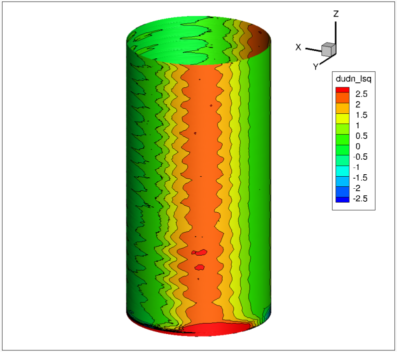
Drag coefficient vs. time

Results: Infinite Cylinder - Comparison of Engineering Coefficients

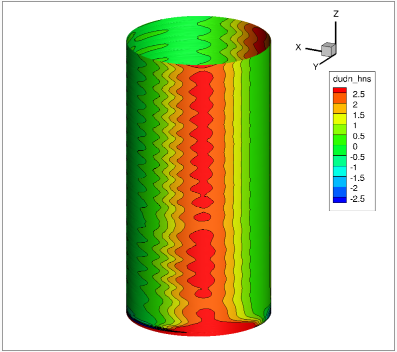
	St	$c_{L_{rms}}$	$c_{D_{avg}}$
reference	0.1957	0.4244	1.3365
NS values	0.1961	0.4941	1.3376
EF values	0.1964	0.4991	1.3699
HEF values	0.1969	0.4986	1.3697

Comparison of engineering quantities to reference values.

Results: Infinite Cylinder - Comparison of Normal Gradient of x-velocity



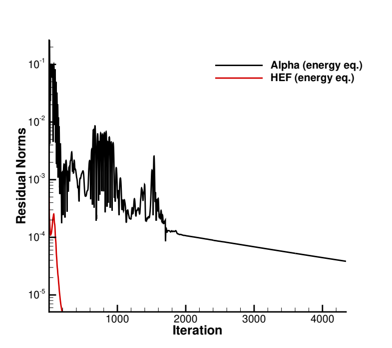
(a) LSQ gradient.



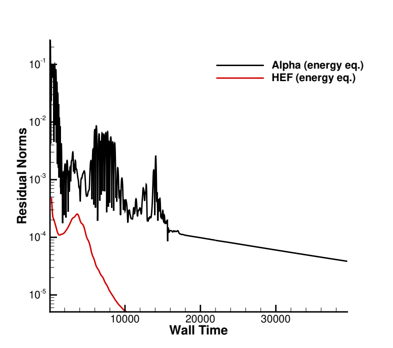
(b) HEF gradient.

Normal gradient on cylinder surface.

Results: Infinite Cylinder - Comparison of Time to Solution



(a) Iterations to solution



(b) CPU time to soluion

Comparison of CPU time and iterations to convergence for boundary layer MMS.

Conclusion and Future Work

Conclusions and Future Work: Conclusions

- Implemented Svard's Eulerian governing equations and showed computations for highly skewed/anisotropic meshes

Conclusions and Future Work: Conclusions

- Implemented Svard's Eulerian governing equations and showed computations for highly skewed/anisotropic meshes
- Verified these results with MMS and compared with NS discretizations

Conclusions and Future Work: Conclusions

- Implemented Svard's Eulerian governing equations and showed computations for highly skewed/anisotropic meshes
- Verified these results with MMS and compared with NS discretizations
- Demonstrated that the HEF solver is faster in time to solution than the alpha-damping solver and has better accuracy in the gradients

Conclusions and Future Work: Conclusions

- Implemented Svard's Eulerian governing equations and showed computations for highly skewed/anisotropic meshes
- Verified these results with MMS and compared with NS discretizations
- Demonstrated that the HEF solver is faster in time to solution than the alpha-damping solver and has better accuracy in the gradients
- The HEF discretization has a simpler viscous dissipation matrix

Conclusions and Future Work: Future Work

- Implement the equations in 3 physical dimensions and benchmark on realistic configurations.

Conclusions and Future Work: Future Work

- Implement the equations in 3 physical dimensions and benchmark on realistic configurations.
- High-order finite-volume/difference schemes with one flux per edge due to the lack of tangent terms in the viscous fluxes.

Our thanks to:

Acknowledgements

Our thanks to:

1. **Kyle Anderson for suggesting Svard's paper**

Acknowledgements

Our thanks to:

1. **Kyle Anderson for suggesting Svard's paper**
2. **Hiro Nishikawa for his assistance in deriving the viscous dissipation matrix**

Acknowledgements

Our thanks to:

1. **Kyle Anderson for suggesting Svard's paper**
2. **Hiro Nishikawa for his assistance in deriving the viscous dissipation matrix**
3. **Army Research Office for funding the portion of this work that was done on the HNS equation**

Thank you!