Predictive Engineering and Computational Sciences

Verification, Validation and Uncertainty Quantification in Direct Numerical Simulation

Nicholas Malaya

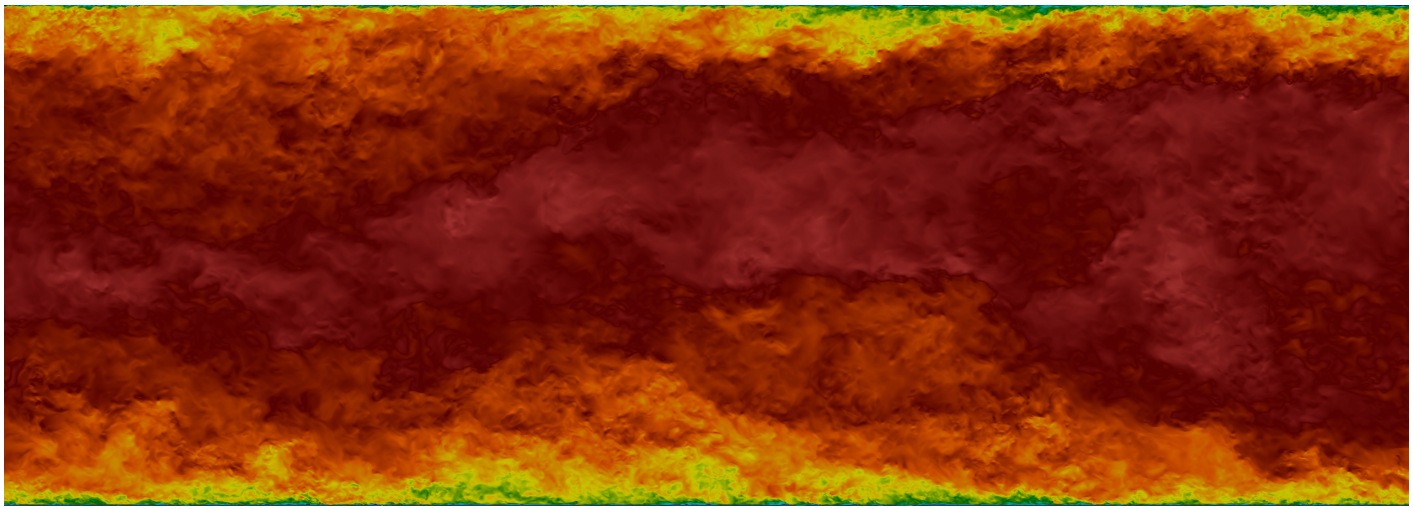
Department of Mechanical Engineering
Institute for Computational Engineering and Sciences
The University of Texas at Austin

May 12th, 2016

Direct Numerical Simulations

Physics Simulation

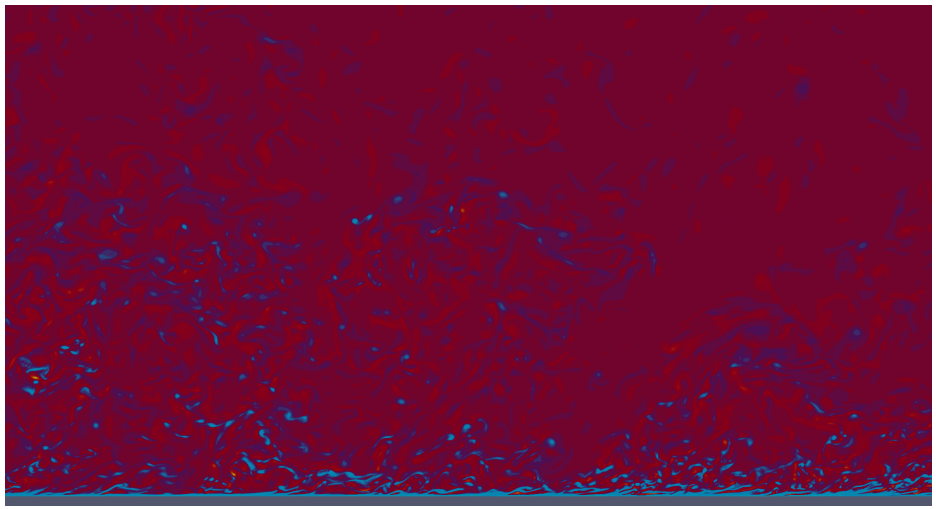
- 225 Billion Degrees of Freedom
- 524,288 Processing Cores
- 275 Million Compute Hours
- Indistinguishable from physical experiments
- **Are we confident in our predictions?**



Direct Numerical Simulations

It's not you, it's me.

- "We have released Mira for use. However, service is degraded. Several racks (1024 nodes each) remain offline, and the system will have a high risk of hardware errors."
- "A new efix was installed on Mira and Cetus which contains fixes and performance enhancements for the BG/Q MPI and PAMI packages. All users are strongly advised to recompile and relink their code."
- **Are we STILL confident in our predictions?**



Estimating Uncertainties in DNS

Direct Numerical Simulation (DNS) as a Computational Laboratory

- Resolve all relevant physical scales (large and small)
- DNS data is widely used by the turbulence community
- Errors and uncertainties are seldom quantified

Typical practice

- Mesh resolution and simulation time guided by previous experience
- Results evaluated by expert practitioners looking for:
  - Non-stationary behavior, insufficient sampling
  - Non-converged quantities, insufficient resolution
- Almost entirely qualitative

Error Decomposition

$$\langle q_h \rangle_N = \langle q \rangle + e_h + \epsilon_{h,N}$$

q_h	Instantaneous value of QoI at resolution h
$\langle q_h \rangle_N$	Computed mean at resolution h with sampling duration N
$\langle q \rangle$	"True" mean value of QoI ($h \to 0$, $N \to \infty$)
e_h	Discretization error at resolution h ($N \to \infty$)
$\epsilon_{h,N}$	Sampling error at resolution h for sampling duration N

Require methods to estimate both e_h and $\epsilon_{h,N}$

Estimating Statistical Error, $\epsilon_{h,N}$

CLT, Regained

- How to characterize the statistical error?
  - $\epsilon_{h,N} \xrightarrow{d} \mathcal{N}(0, \sigma^2/N)$ in limit of large (independent) sample size
  - Calculate "decorrelation time": N_0
  - "effective sample size": $N_{\text{effective}} = N/N_0$
  - Regain CLT convergence: $\frac{1}{\sqrt{N_{\text{effective}}}}$

Computing the Autocorrelation

- Computing N_0 requires the autocorrelation function, ρ
  - Direct calculations of ρ from $O(100)$ samples is noisy
- Instead, ρ estimated using maximum entropy autoregressive model fitting as in Trenberth (1984)

Estimating Discretization Error, e_h

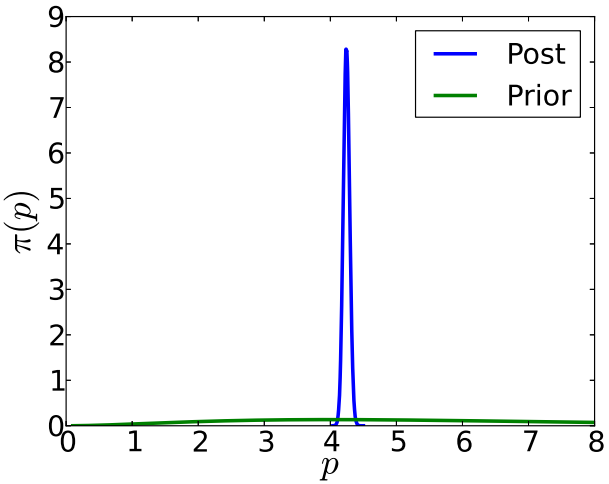
Deterministic (Standard Richardson Extrapolation)

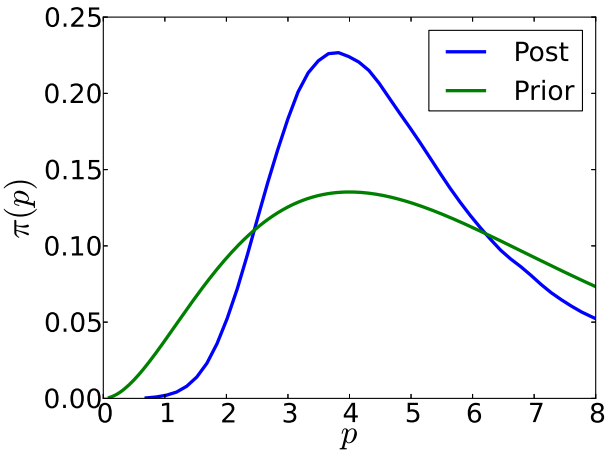
- Assume: $q_h = q + Ch^p$
- Given q_h for 3 distinct h, solve for q, C, and p
- Breaks down when q_h contaminated by sampling error

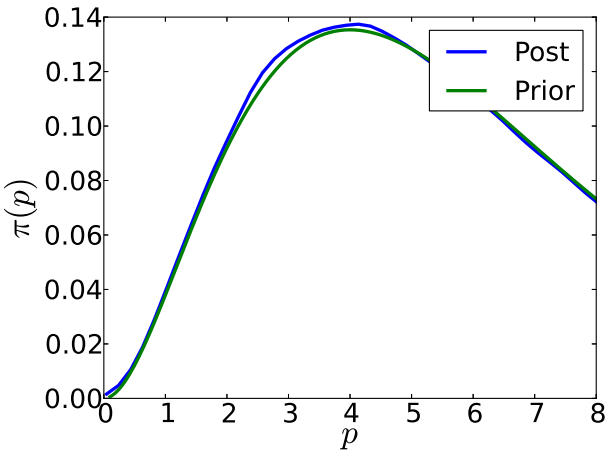
Stochastic (Bayesian Richardson Extrapolation)

- Recall our model: $\langle q_h \rangle_N = \langle q \rangle + e_h + \epsilon_{h,N}$
- Assume: $e_h = Ch^p$
  - Other models may be appropriate depending on the discretization
- Objective: *Infer* $\langle q \rangle$, C, and p under uncertainty imposed by $\epsilon_{h,N}$
- We choose a Bayesian approach to the inverse problem
- Prior depends on knowledge of the problem; we use “broad” priors

Verification of Methodology: Lorenz Equations







Increasing statistical uncertainty

- Using RK4 $\Rightarrow$ $p = 4$
- Only recover true p with small statistical uncertainty
- With large statistical uncertainty, posterior same as prior

Numerical Methods

- Fourier–Galerkin in streamwise (x) and spanwise (z)
- B-spline collocation in wall-normal (8th order B-splines)
- SMR91 hybrid implicit/explicit time scheme[4] (Mixed 2^{nd} and 3^{rd} order.)

Name	N_x	N_z	N_y	Δx^+	Δz^+	Δy^+_{wall}	Δy^+_{CL}	TU_b/L_x	$\Delta t U_b/\delta$
Coarsest	96	96	64	24.3	12.2	0.44	9.14	2651.0	0.02
Coarse	136	136	90	17.2	8.6	0.26	6.46	273.5	0.01414
Nominal	192	192	128	12.2	6.1	0.16	4.53	2145.3	0.01
Finest	384	384	256	6.1	3.0	0.07	2.26	709.3	0.005

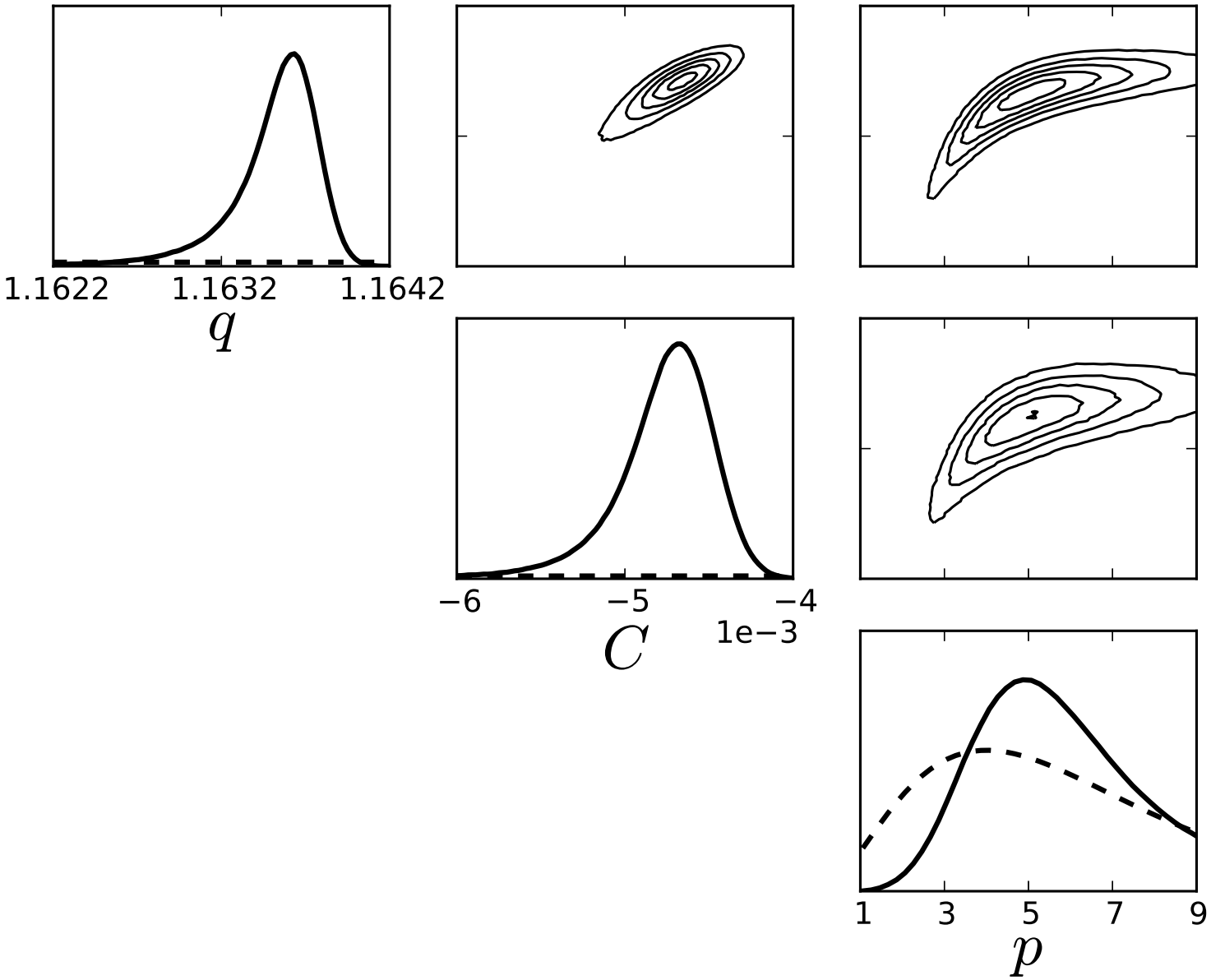
Case Details

- $\text{Re}_\tau \approx 180$, $\text{Re}_{\text{bulk}} = U\delta/\nu = 2925$
- Box size[5]: $L_x/\delta = 4\pi$, $L_z/\delta = 2\pi$
- Δt always $<$ CFL limit

[4] Spalart, Moser, and Rogers, *Journal of Computational Physics*, 1991.

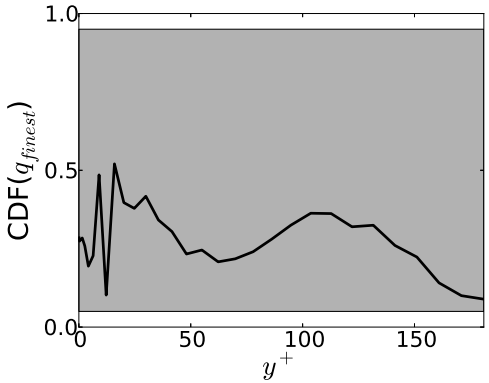
[5] Kim, Moin, and Moser, *Journal of Fluid Mechanics*, 1987.

Results of the Inverse Problem for U_{CL}

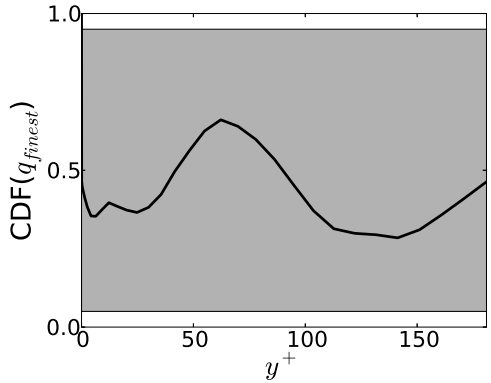


Model Validation

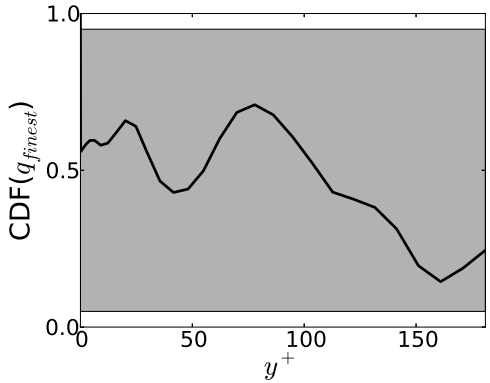
Valid



(a) $\langle u \rangle$

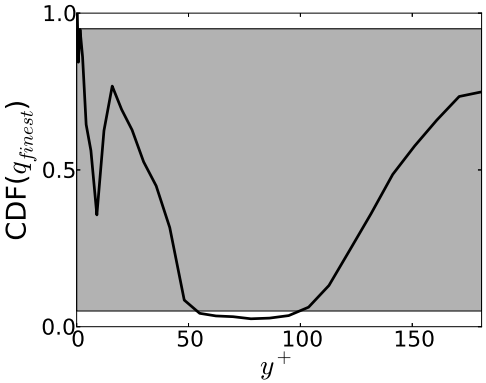


(b) $\langle v'v' \rangle$

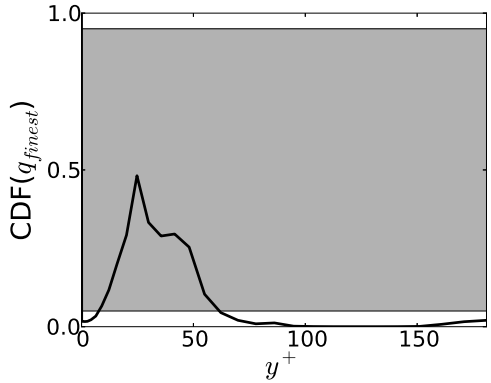


(c) $\langle w'w' \rangle$

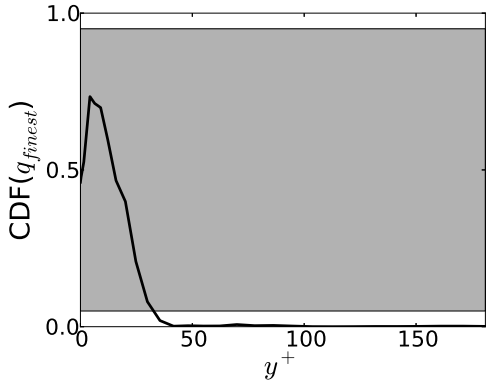
Invalid



(d) $\langle u'u' \rangle$



(e) $\langle \omega'_y \omega'_y \rangle$



(f) $\langle \omega'_z \omega'_z \rangle$

- Solid line is the computed value of the CDF at the observed value.
- Grey shows the 90% credibility interval.

Conclusions

Summary

- Estimator of statistical error developed
- Used as input for Bayesian estimator of discretization error
- When model not invalidated, results estimate DNS uncertainties

More information available:

- paper: “Estimating uncertainties in statistics computed from direct numerical simulation”
- DOI: 10.1063/1.4866813
- Data available at: turbulence.ices.utexas.edu

Weaknesses

- Requires several runs for simulations (e.g. expensive)
- Not run with variable Δt, yet all DNS run in this mode

Shifting gears– What is verification

Reality

$\Downarrow$ (Validation)

Mathematical Model

$$\frac{d^2x(t)}{dt^2} = \frac{F}{M}$$

$\Downarrow$ (Verification)

Numerical Representation

$$\frac{d^2x}{dt^2} = f''(t) = \frac{f(t+h)-2f(t)+f(t-h)}{h^2}$$

Methods of Verification

Code Verification

- Code verification: Ensuring that the code used in the simulation correctly implements the intended numerical discretization of the model.

Method of Exact Solutions

- Numerically solve the governing equations for which the solution can be determined analytically.

Method of Manufactured Solutions

- Often, analytical solutions either:
  - Do not exist (Navier-Stokes)
  - Do not fully exercise equations (e.g. a symmetric solution, nonlinearities)
- Alleviate this using Method of Manufactured Solutions (MMS)
  - Simply put, we “create” our own solutions

Manufactured solution to Laplace's Equation

Laplace's Equation:

$$\nabla^2 \phi = 0$$

In two dimensions:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0$$

"Manufacture" a solution, with two constants:

$$\phi(x, y) = (Ly - y)^2 (Ly + y)^2 + (Lx - x)^2 (Lx + x)^2$$

Calculating the Source Term

We insert our manufactured solution back into the governing equations:

$$\frac{\partial^2((Lx-x)^2(Lx+x)^2)}{\partial x^2} + \frac{\partial^2((Ly-y)^2(Ly+y)^2)}{\partial y^2} = 0$$

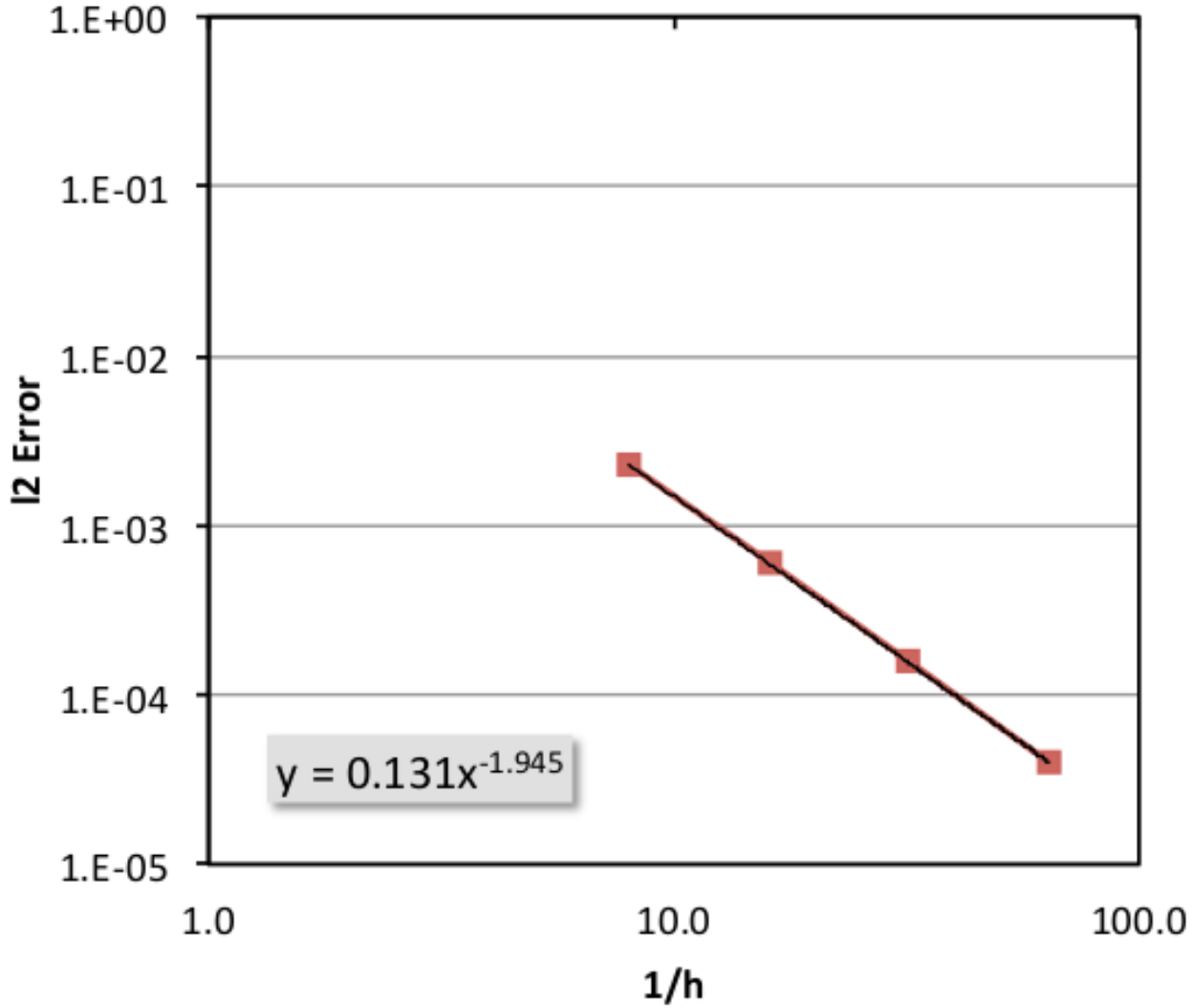
$$\begin{aligned}
=&2(Lx-x)^2 - 8(Lx-x)(Lx+x) + 2(Lx+x)^2 \\
&+2(Ly-y)^2 - 8(Ly-y)(Ly+y) + 2(Ly+y)^2 \\
&\neq 0
\end{aligned}$$

This does not satisfy Laplace's Equation!

To balance the equation, add the residual to the RHS as a source term.

Example Results: What we're hoping for

2nd Order Central Finite-difference Scheme



A Real Example

MMS Creation Process

- Start by “manufacturing” a suitable closed-form exact solution
- For example, the 10 parameter trigonometric solution of the form: (Roy, 2002)

$$\hat{u}(x,y,z,t) = \hat{u}_0 + \hat{u}_x\, f_s\left(\frac{a_{\hat{u}x}\pi x}{L}\right) + \hat{u}_y\, f_s\left(\frac{a_{\hat{u}y}\pi y}{L}\right) + \\ + \hat{u}_z\, f_s\left(\frac{a_{\hat{u}z}\pi z}{L}\right) + \hat{u}_t\, f_s\left(\frac{a_{\hat{u}_t}\pi t}{L}\right)$$

- Apply this solution to equations of interest, solve for source terms (residual)

Accomplished using symbolic manipulation SymPy, Maple, Mathematica, Macsyma, etc.

Maple MMS: 3D Navier-Stokes Energy Term

$$
\begin{aligned}
Qe = &- \frac{a_{px}\pi p_x}{L}\frac{\gamma}{\gamma-1}\sin\left(\frac{a_{px}\pi x}{L}\right)\left[u_0 + u_x\sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y\cos\left(\frac{a_{uy}\pi y}{L}\right) + u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right] + \\
&+ \frac{a_{py}\pi p_y}{L}\frac{\gamma}{\gamma-1}\cos\left(\frac{a_{py}\pi y}{L}\right)\left[v_0 + v_x\cos\left(\frac{a_{vx}\pi x}{L}\right) + v_y\sin\left(\frac{a_{vy}\pi y}{L}\right) + v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right] + \\
&- \frac{a_{pz}\pi p_z}{L}\frac{\gamma}{\gamma-1}\sin\left(\frac{a_{pz}\pi z}{L}\right)\left[w_0 + w_x\sin\left(\frac{a_{wx}\pi x}{L}\right) + w_y\sin\left(\frac{a_{wy}\pi y}{L}\right) + w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right] + \\
&+ \frac{a_{\rho x}\pi\rho_x}{2L}\cos\left(\frac{a_{\rho x}\pi x}{L}\right)\left[u_0 + u_x\sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y\cos\left(\frac{a_{uy}\pi y}{L}\right) + u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]\left(\left[u_0 + u_x\sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y\cos\left(\frac{a_{uy}\pi y}{L}\right) + u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]^2 + \right. \\
&\left. + \left[w_0 + w_x\sin\left(\frac{a_{wx}\pi x}{L}\right) + w_y\sin\left(\frac{a_{wy}\pi y}{L}\right) + w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]^2 + \left[v_0 + v_x\cos\left(\frac{a_{vx}\pi x}{L}\right) + v_y\sin\left(\frac{a_{vy}\pi y}{L}\right) + v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]^2\right) + \\
&- \frac{a_{\rho y}\pi\rho_y}{2L}\sin\left(\frac{a_{\rho y}\pi y}{L}\right)\left[v_0 + v_x\cos\left(\frac{a_{vx}\pi x}{L}\right) + v_y\sin\left(\frac{a_{vy}\pi y}{L}\right) + v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]\left(\left[u_0 + u_x\sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y\cos\left(\frac{a_{uy}\pi y}{L}\right) + u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]^2 + \right. \\
&\left. + \left[w_0 + w_x\sin\left(\frac{a_{wx}\pi x}{L}\right) + w_y\sin\left(\frac{a_{wy}\pi y}{L}\right) + w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]^2 + \left[v_0 + v_x\cos\left(\frac{a_{vx}\pi x}{L}\right) + v_y\sin\left(\frac{a_{vy}\pi y}{L}\right) + v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]^2\right) + \\
&+ \frac{a_{\rho z}\pi\rho_z}{2L}\cos\left(\frac{a_{\rho z}\pi z}{L}\right)\left[w_0 + w_x\sin\left(\frac{a_{wx}\pi x}{L}\right) + w_y\sin\left(\frac{a_{wy}\pi y}{L}\right) + w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]\left(\left[u_0 + u_x\sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y\cos\left(\frac{a_{uy}\pi y}{L}\right) + u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]^2 + \right. \\
&\left. + \left[w_0 + w_x\sin\left(\frac{a_{wx}\pi x}{L}\right) + w_y\sin\left(\frac{a_{wy}\pi y}{L}\right) + w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]^2 + \left[v_0 + v_x\cos\left(\frac{a_{vx}\pi x}{L}\right) + v_y\sin\left(\frac{a_{vy}\pi y}{L}\right) + v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]^2\right) + \\
&+ \frac{a_{ux}\pi u_x}{2L}\cos\left(\frac{a_{ux}\pi x}{L}\right)\left\{\left(\left[w_0 + w_x\sin\left(\frac{a_{wx}\pi x}{L}\right) + w_y\sin\left(\frac{a_{wy}\pi y}{L}\right) + w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]^2 + \left[v_0 + v_x\cos\left(\frac{a_{vx}\pi x}{L}\right) + v_y\sin\left(\frac{a_{vy}\pi y}{L}\right) + v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]^2 + \right. \\
&\left. +3\left[u_0 + u_x\sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y\cos\left(\frac{a_{uy}\pi y}{L}\right) + u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]^2\right)\left[\rho_0 + \rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right) + \rho_y\cos\left(\frac{a_{\rho y}\pi y}{L}\right) + \rho_z\sin\left(\frac{a_{\rho z}\pi z}{L}\right)\right] + \\
&\left. + \left[p_0 + p_x\cos\left(\frac{a_{px}\pi x}{L}\right) + p_y\sin\left(\frac{a_{py}\pi y}{L}\right) + p_z\cos\left(\frac{a_{pz}\pi z}{L}\right)\right]\frac{2\gamma}{(\gamma-1)}\right\} + \\
&- \frac{a_{uy}\pi u_y}{L}\sin\left(\frac{a_{uy}\pi y}{L}\right)\left[v_0 + v_x\cos\left(\frac{a_{vx}\pi x}{L}\right) + v_y\sin\left(\frac{a_{vy}\pi y}{L}\right) + v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]\left[\rho_0 + \rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right) + \rho_y\cos\left(\frac{a_{\rho y}\pi y}{L}\right) + \rho_z\sin\left(\frac{a_{\rho z}\pi z}{L}\right)\right]\cdot \\
&\cdot\left[u_0 + u_x\sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y\cos\left(\frac{a_{uy}\pi y}{L}\right) + u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right] + \\
&- \frac{a_{uz}\pi u_z}{L}\sin\left(\frac{a_{uz}\pi z}{L}\right)\left[w_0 + w_x\sin\left(\frac{a_{wx}\pi x}{L}\right) + w_y\sin\left(\frac{a_{wy}\pi y}{L}\right) + w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]\left[\rho_0 + \rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right) + \rho_y\cos\left(\frac{a_{\rho y}\pi y}{L}\right) + \rho_z\sin\left(\frac{a_{\rho z}\pi z}{L}\right)\right]\cdot \\
&\cdot\left[u_0 + u_x\sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y\cos\left(\frac{a_{uy}\pi y}{L}\right) + u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right] +
\end{aligned}
$$

But wait, there's more!

$$
-\frac{a_{px}^2\pi^2 k\rho_x\left[2\cos\left(\frac{a_{px}\pi x}{L}\right)^2\rho_x+\sin\left(\frac{a_{px}\pi x}{L}\right)\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]\right]\left[p_0+p_x\cos\left(\frac{a_{px}\pi x}{L}\right)+p_y\sin\left(\frac{a_{py}\pi y}{L}\right)+p_z\cos\left(\frac{a_{pz}\pi z}{L}\right)\right]}{RL^2\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^3}+
$$

$$
-\frac{a_{py}^2\pi^2 k\rho_y\left[2\sin\left(\frac{a_{py}\pi y}{L}\right)^2\rho_y+\cos\left(\frac{a_{py}\pi y}{L}\right)\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]\right]\left[p_0+p_x\cos\left(\frac{a_{px}\pi x}{L}\right)+p_y\sin\left(\frac{a_{py}\pi y}{L}\right)+p_z\cos\left(\frac{a_{pz}\pi z}{L}\right)\right]}{RL^2\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^3}+
$$

$$
-\frac{a_{pz}^2\pi^2 k\rho_z\left[2\cos\left(\frac{a_{pz}\pi z}{L}\right)^2\rho_z+\sin\left(\frac{a_{pz}\pi z}{L}\right)\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]\right]\left[p_0+p_x\cos\left(\frac{a_{px}\pi x}{L}\right)+p_y\sin\left(\frac{a_{py}\pi y}{L}\right)+p_z\cos\left(\frac{a_{pz}\pi z}{L}\right)\right]}{RL^2\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^3}+
$$

$$
-\frac{4}{3}\frac{a_{ux}^2\pi^2\mu u_x}{L^2}\left[\cos\left(\frac{a_{ux}\pi x}{L}\right)^2u_x-\sin\left(\frac{a_{ux}\pi x}{L}\right)\left[u_0+u_x\sin\left(\frac{a_{ux}\pi x}{L}\right)+u_y\cos\left(\frac{a_{uy}\pi y}{L}\right)+u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{a_{uy}^2\pi^2\mu u_y}{L^2}\left[\sin\left(\frac{a_{uy}\pi y}{L}\right)^2u_y-\cos\left(\frac{a_{uy}\pi y}{L}\right)\left[u_0+u_x\sin\left(\frac{a_{ux}\pi x}{L}\right)+u_y\cos\left(\frac{a_{uy}\pi y}{L}\right)+u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{a_{uz}^2\pi^2\mu u_z}{L^2}\left[\sin\left(\frac{a_{uz}\pi z}{L}\right)^2u_z-\cos\left(\frac{a_{uz}\pi z}{L}\right)\left[u_0+u_x\sin\left(\frac{a_{ux}\pi x}{L}\right)+u_y\cos\left(\frac{a_{uy}\pi y}{L}\right)+u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{a_{vx}^2\pi^2\mu v_x}{L^2}\left[\sin\left(\frac{a_{vx}\pi x}{L}\right)^2v_x-\cos\left(\frac{a_{vx}\pi x}{L}\right)\left[v_0+v_x\cos\left(\frac{a_{vx}\pi x}{L}\right)+v_y\sin\left(\frac{a_{vy}\pi y}{L}\right)+v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{4}{3}\frac{a_{vy}^2\pi^2\mu v_y}{L^2}\left[\cos\left(\frac{a_{vy}\pi y}{L}\right)^2v_y-\sin\left(\frac{a_{vy}\pi y}{L}\right)\left[v_0+v_x\cos\left(\frac{a_{vx}\pi x}{L}\right)+v_y\sin\left(\frac{a_{vy}\pi y}{L}\right)+v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{a_{vz}^2\pi^2\mu v_z}{L^2}\left[\cos\left(\frac{a_{vz}\pi z}{L}\right)^2v_z-\sin\left(\frac{a_{vz}\pi z}{L}\right)\left[v_0+v_x\cos\left(\frac{a_{vx}\pi x}{L}\right)+v_y\sin\left(\frac{a_{vy}\pi y}{L}\right)+v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{a_{wx}^2\pi^2\mu w_x}{L^2}\left[\cos\left(\frac{a_{wx}\pi x}{L}\right)^2w_x-\sin\left(\frac{a_{wx}\pi x}{L}\right)\left[w_0+w_x\sin\left(\frac{a_{wx}\pi x}{L}\right)+w_y\sin\left(\frac{a_{wy}\pi y}{L}\right)+w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{a_{wy}^2\pi^2\mu w_y}{L^2}\left[\cos\left(\frac{a_{wy}\pi y}{L}\right)^2w_y-\sin\left(\frac{a_{wy}\pi y}{L}\right)\left[w_0+w_x\sin\left(\frac{a_{wx}\pi x}{L}\right)+w_y\sin\left(\frac{a_{wy}\pi y}{L}\right)+w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{4}{3}\frac{a_{wz}^2\pi^2\mu w_z}{L^2}\left[\sin\left(\frac{a_{wz}\pi z}{L}\right)^2w_z-\cos\left(\frac{a_{wz}\pi z}{L}\right)\left[w_0+w_x\sin\left(\frac{a_{wx}\pi x}{L}\right)+w_y\sin\left(\frac{a_{wy}\pi y}{L}\right)+w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]\right]+
$$

$$
-\frac{2a_{px}a_{\rho x}\pi^2 kp_x\rho_x\cos\left(\frac{a_{\rho x}\pi x}{L}\right)\sin\left(\frac{a_{px}\pi x}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^2}+
$$

$$
-\frac{2a_{py}a_{\rho y}\pi^2 kp_y\rho_y\sin\left(\frac{a_{\rho y}\pi y}{L}\right)\cos\left(\frac{a_{py}\pi y}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^2}+
$$

$$
-\frac{2a_{pz}a_{\rho z}\pi^2 kp_z\rho_z\cos\left(\frac{a_{\rho z}\pi z}{L}\right)\sin\left(\frac{a_{pz}\pi z}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^2}+
$$

$$
+\frac{4}{3}\frac{a_{ux}a_{vy}\pi^2\mu u_x v_y}{L^2}\cos\left(\frac{a_{ux}\pi x}{L}\right)\cos\left(\frac{a_{vy}\pi y}{L}\right)+
$$

$$
-\frac{4}{3}\frac{a_{ux}a_{ws}\pi^2\mu u_x w_z}{L^2}\cos\left(\frac{a_{wx}\pi x}{L}\right)\sin\left(\frac{a_{wz}\pi z}{L}\right)+
$$

$$
-\frac{2a_{uy}a_{vx}\pi^2\mu u_y v_x}{L^2}\sin\left(\frac{a_{uy}\pi y}{L}\right)\sin\left(\frac{a_{vx}\pi x}{L}\right)+
$$

$$
+\frac{2a_{uz}a_{wx}\pi^2\mu u_z w_x}{L^2}\cos\left(\frac{a_{wx}\pi x}{L}\right)\sin\left(\frac{a_{uz}\pi z}{L}\right)+
$$

$$
-\frac{4}{3}\frac{a_{vy}a_{wz}\pi^2\mu v_y w_z}{L^2}\cos\left(\frac{a_{vy}\pi y}{L}\right)\sin\left(\frac{a_{wz}\pi z}{L}\right)+
$$

$$
-\frac{2a_{vz}a_{wy}\pi^2\mu v_z w_y}{L^2}\cos\left(\frac{a_{vz}\pi z}{L}\right)\cos\left(\frac{a_{wy}\pi y}{L}\right).
$$

$$
-\frac{a_{vx}\pi v_x}{L}\sin\left(\frac{a_{vx}\pi x}{L}\right)\left[u_0+u_x\sin\left(\frac{a_{ux}\pi x}{L}\right)+u_y\cos\left(\frac{a_{uy}\pi y}{L}\right)+u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]\cdot
$$

$$
\cdot\left[v_0+v_x\cos\left(\frac{a_{vx}\pi x}{L}\right)+v_y\sin\left(\frac{a_{vy}\pi y}{L}\right)+v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]+
$$

$$
+\frac{a_{vy}\pi v_y}{2L}\cos\left(\frac{a_{vy}\pi y}{L}\right)\left\{\left(\left[u_0+u_x\sin\left(\frac{a_{ux}\pi x}{L}\right)+u_y\cos\left(\frac{a_{uy}\pi y}{L}\right)+u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]^2+\left[w_0+w_x\sin\left(\frac{a_{wx}\pi x}{L}\right)+w_y\sin\left(\frac{a_{wy}\pi y}{L}\right)+w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]^2+\right.\right.
$$

$$
\left.+3\left[v_0+v_x\cos\left(\frac{a_{vx}\pi x}{L}\right)+v_y\sin\left(\frac{a_{vy}\pi y}{L}\right)+v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]^2\right)\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]+
$$

$$
\left.+\left[p_0+p_x\cos\left(\frac{a_{px}\pi x}{L}\right)+p_y\sin\left(\frac{a_{py}\pi y}{L}\right)+p_z\cos\left(\frac{a_{pz}\pi z}{L}\right)\right]\frac{2\gamma}{\gamma-1}\right\}+
$$

$$
+\frac{a_{vz}\pi v_z}{L}\cos\left(\frac{a_{vz}\pi z}{L}\right)\left[w_0+w_x\sin\left(\frac{a_{wx}\pi x}{L}\right)+w_y\sin\left(\frac{a_{wy}\pi y}{L}\right)+w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]\cdot
$$

$$
\cdot\left[v_0+v_x\cos\left(\frac{a_{vx}\pi x}{L}\right)+v_y\sin\left(\frac{a_{vy}\pi y}{L}\right)+v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]+
$$

$$
+\frac{a_{wx}\pi w_x}{L}\cos\left(\frac{a_{wx}\pi x}{L}\right)\left[u_0+u_x\sin\left(\frac{a_{ux}\pi x}{L}\right)+u_y\cos\left(\frac{a_{uy}\pi y}{L}\right)+u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]\cdot
$$

$$
\cdot\left[w_0+w_x\sin\left(\frac{a_{wx}\pi x}{L}\right)+w_y\sin\left(\frac{a_{wy}\pi y}{L}\right)+w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]+
$$

$$
+\frac{a_{wy}\pi w_y}{L}\cos\left(\frac{a_{wy}\pi y}{L}\right)\left[v_0+v_x\cos\left(\frac{a_{vx}\pi x}{L}\right)+v_y\sin\left(\frac{a_{vy}\pi y}{L}\right)+v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]\cdot
$$

$$
\cdot\left[w_0+w_x\sin\left(\frac{a_{wx}\pi x}{L}\right)+w_y\sin\left(\frac{a_{wy}\pi y}{L}\right)+w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]+
$$

$$
-\frac{a_{wz}\pi w_z}{2L}\sin\left(\frac{a_{wz}\pi z}{L}\right)\left\{\left(3\left[w_0+w_x\sin\left(\frac{a_{wx}\pi x}{L}\right)+w_y\sin\left(\frac{a_{wy}\pi y}{L}\right)+w_z\cos\left(\frac{a_{wz}\pi z}{L}\right)\right]^2+\left[v_0+v_x\cos\left(\frac{a_{vx}\pi x}{L}\right)+v_y\sin\left(\frac{a_{vy}\pi y}{L}\right)+v_z\sin\left(\frac{a_{vz}\pi z}{L}\right)\right]^2+\right.
$$

$$
\left.+\left[u_0+u_x\sin\left(\frac{a_{ux}\pi x}{L}\right)+u_y\cos\left(\frac{a_{uy}\pi y}{L}\right)+u_z\cos\left(\frac{a_{uz}\pi z}{L}\right)\right]^2\right)\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]+
$$

$$
\left.+\left[p_0+p_x\cos\left(\frac{a_{px}\pi x}{L}\right)+p_y\sin\left(\frac{a_{py}\pi y}{L}\right)+p_z\cos\left(\frac{a_{pz}\pi z}{L}\right)\right]\frac{2\gamma}{(\gamma-1)}\right\}+
$$

$$
+\frac{a_{px}^2\pi^2 kp_x\cos\left(\frac{a_{px}\pi x}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]}+
$$

$$
+\frac{a_{py}^2\pi^2 kp_y\sin\left(\frac{a_{py}\pi y}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]}+
$$

$$
+\frac{a_{pz}^2\pi^2 kp_z\cos\left(\frac{a_{pz}\pi z}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{\rho x}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]}+
$$

C-code output

```
RHO = rho_0 + rho_x * sin(a_rhox * PI * x / L) + rho_y * cos(a_rhoy * PI * y / L) + rho_z * sin(a_rhoz * PI * z / L) + rho_t * sin(a_rhot * PI * t / L);
 U = u_0 + u_x * sin(a_ux * PI * x / L) + u_y * cos(a_uy * PI * y / L) + u_z * cos(a_uz * PI * z / L) + u_t * cos(a_ut * PI * t / L);
 V = v_0 + v_x * cos(a_vx * PI * x / L) + v_y * sin(a_vy * PI * y / L) + v_z * sin(a_vz * PI * z / L) + v_t * sin(a_vt * PI * t / L);
 W = w_0 + w_x * sin(a_wx * PI * x / L) + w_y * sin(a_wy * PI * y / L) + w_z * cos(a_wz * PI * z / L) + w_t * cos(a_wt * PI * t / L);
 P = p_0 + p_x * cos(a_px * PI * x / L) + p_y * sin(a_py * PI * y / L) + p_z * cos(a_pz * PI * z / L) + p_t * cos(a_pt * PI * t / L);
 M1 = A_mu * pow(P / R / RHO, 0.3e1 / 0.2e1);
 M2 = P / R / RHO + B_mu;
 MU = M1 / M2;
 Q_e = (0.3e1 * a_ux * u_x * cos(a_ux * PI * x / L) + a_vy * v_y * cos(a_vy * PI * y / L) - a_wz * w_z * sin(a_wz * PI * z / L)) * PI * RHO * U * U / L / 0.2e1 + (a_ux * u_x * cos(a_ux * PI
* x / L) + 0.3e1 * a_vy * v_y * cos(a_vy * PI * y / L) - a_wz * w_z * sin(a_wz * PI * z / L)) * PI * RHO * V * V / L / 0.2e1 + (a_ux * u_x * cos(a_ux * PI * x / L) + a_vy * v_y * cos(a_vy *
PI * y / L) - 0.3e1 * a_wz * w_z * sin(a_wz * PI * z / L)) * PI * RHO * W * W / L / 0.2e1 + (0.4e1 * a_ux * a_ux * u_x * sin(a_ux * PI * x / L) + 0.3e1 * a_uy * a_uy * u_y * cos(a_uy * PI * y
/ L) + 0.3e1 * a_uz * a_uz * u_z * cos(a_uz * PI * z / L)) * MU * PI * PI * U * pow(L, -0.2e1) / 0.3e1 + (0.3e1 * a_vx * a_vx * v_x * cos(a_vx * PI * x / L) + 0.4e1 * a_vy * a_vy * v_y *
sin(a_vy * PI * y / L) + 0.3e1 * a_vz * a_vz * v_z * sin(a_vz * PI * z / L)) * MU * PI * PI * V * pow(L, -0.2e1) / 0.3e1 + (0.3e1 * a_wx * a_wx * w_x * sin(a_wx * PI * x / L) + 0.3e1 * a_wy *
a_wy * w_y * sin(a_wy * PI * y / L) + 0.4e1 * a_wz * a_wz * w_z * cos(a_wz * PI * z / L)) * MU * PI * PI * W * pow(L, -0.2e1) / 0.3e1 + (a_ux * u_x * cos(a_ux * PI * x / L) + a_vy * v_y *
cos(a_vy * PI * y / L) - a_wz * w_z * sin(a_wz * PI * z / L)) * PI * P / (Gamma - 0.1e1) / L - (0.2e1 * a_rhox * a_px * rho_x * p_x * cos(a_rhox * PI * x / L) * sin(a_px * PI * x / L) + 0.2e1
* a_rhoy * a_py * rho_y * p_y * sin(a_rhoy * PI * y / L) * cos(a_py * PI * y / L) + 0.2e1 * a_rhoz * a_pz * rho_z * p_z * cos(a_rhoz * PI * z / L) * sin(a_pz * PI * z / L)) * PI * PI * k *
pow(L, -0.2e1) / R * pow(RHO, -0.2e1) + (U * U + V * V + W * W) * a_rhot * PI * rho_t * cos(a_rhot * PI * t / L) / L / 0.2e1 - a_pt * PI * p_t * sin(a_pt * PI * t / L) / (Gamma - 0.1e1) / L -
(a_uy * u_y * sin(a_uy * PI * y / L) + a_vx * v_x * sin(a_vx * PI * x / L)) * PI * RHO * U * V / L - (a_uz * u_z * sin(a_uz * PI * z / L) - a_wx * w_x * cos(a_wx * PI * x / L)) * PI * RHO * U
* W / L + (a_vz * v_z * cos(a_vz * PI * z / L) + a_wy * w_y * cos(a_wy * PI * y / L)) * PI * RHO * V * W / L + (a_px * a_px * p_x * cos(a_px * PI * x / L) + a_py * a_py * p_y * sin(a_py * PI
* y / L) + a_pz * a_pz * p_z * cos(a_pz * PI * z / L)) * PI * PI * k * pow(L, -0.2e1) / R / RHO - (0.4e1 * a_ux * a_ux * u_x * u_x * pow(cos(a_ux * PI * x / L), 0.2e1) - 0.4e1 * a_ux * a_vy *
u_x * v_y * cos(a_ux * PI * x / L) * cos(a_vy * PI * y / L) + 0.4e1 * a_ux * a_wz * u_x * w_z * cos(a_ux * PI * x / L) * sin(a_wz * PI * z / L) + 0.3e1 * a_uy * a_uy * u_y * u_y *
pow(sin(a_uy * PI * y / L), 0.2e1) + 0.6e1 * a_uy * a_vx * u_y * v_x * sin(a_uy * PI * y / L) * sin(a_vx * PI * x / L) + 0.3e1 * a_uz * a_uz * u_z * u_z * pow(sin(a_uz * PI * z / L), 0.2e1) -
0.6e1 * a_uz * a_wx * u_z * w_x * sin(a_uz * PI * z / L) * cos(a_wx * PI * x / L) + 0.3e1 * a_vx * a_vx * v_x * v_x * pow(sin(a_vx * PI * x / L), 0.2e1) + 0.4e1 * a_vy * a_vy * v_y * v_y *
pow(cos(a_vy * PI * y / L), 0.2e1) + 0.4e1 * a_vy * a_wz * v_y * w_z * cos(a_vy * PI * y / L) * sin(a_wz * PI * z / L) + 0.3e1 * a_vz * a_vz * v_z * v_z * pow(cos(a_vz * PI * z / L), 0.2e1) +
0.6e1 * a_vz * a_wy * v_z * w_y * cos(a_vz * PI * z / L) * cos(a_wy * PI * y / L) + 0.3e1 * a_wx * a_wx * w_x * w_x * pow(cos(a_wx * PI * x / L), 0.2e1) + 0.3e1 * a_wy * a_wy * w_y * w_y *
pow(cos(a_wy * PI * y / L), 0.2e1) + 0.4e1 * a_wz * a_wz * w_z * w_z * pow(sin(a_wz * PI * z / L), 0.2e1)) * MU * PI * PI * pow(L, -0.2e1) / 0.3e1 + (a_ux * u_x * cos(a_ux * PI * x / L) +
a_vy * v_y * cos(a_vy * PI * y / L) - a_wz * w_z * sin(a_wz * PI * z / L)) * PI * P / L - a_ut * PI * u_t * RHO * U * sin(a_ut * PI * t / L) / L + a_vt * PI * v_t * RHO * V * cos(a_vt * PI *
t / L) / L - a_wt * PI * w_t * RHO * W * sin(a_wt * PI * t / L) / L + (U * U + V * V + W * W) * a_rhox * PI * rho_x * U * cos(a_rhox * PI * x / L) / L / 0.2e1 - (U * U + V * V + W * W) *
a_rhoy * PI * rho_y * V * sin(a_rhoy * PI * y / L) / L / 0.2e1 + (U * U + V * V + W * W) * a_rhoz * PI * rho_z * W * cos(a_rhoz * PI * z / L) / L / 0.2e1 - (a_rhox * a_rhox * rho_x *
sin(a_rhox * PI * x / L) + a_rhoy * a_rhoy * rho_y * cos(a_rhoy * PI * y / L) + a_rhoz * a_rhoz * rho_z * sin(a_rhoz * PI * z / L)) * PI * PI * k * P * pow(L, -0.2e1) / R * pow(RHO, -0.2e1) -
(0.2e1 * a_rhox * a_rhox * rho_x * rho_x * pow(cos(a_rhox * PI * x / L), 0.2e1) + 0.2e1 * a_rhoy * a_rhoy * rho_y * rho_y * pow(sin(a_rhoy * PI * y / L), 0.2e1) + 0.2e1 * a_rhoz * a_rhoz *
rho_z * rho_z * pow(cos(a_rhoz * PI * z / L), 0.2e1)) * PI * PI * k * P * pow(L, -0.2e1) / R * pow(RHO, -0.3e1) - (0.3e1 * a_rhox * a_uz * rho_x * u_z * cos(a_rhox * PI * x / L) * sin(a_uz *
PI * z / L) - 0.3e1 * a_rhox * a_wx * rho_x * w_x * cos(a_rhox * PI * x / L) * cos(a_wx * PI * x / L) + 0.3e1 * a_rhoy * a_vz * rho_y * v_z * sin(a_rhoy * PI * y / L) * cos(a_vz * PI * z / L)
+ 0.3e1 * a_rhoy * a_wy * rho_y * w_y * sin(a_rhoy * PI * y / L) * cos(a_wy * PI * y / L) + 0.2e1 * a_rhoz * a_ux * rho_z * u_x * cos(a_rhoz * PI * z / L) * cos(a_ux * PI * x / L) + 0.2e1 *
a_rhoz * a_vy * rho_z * v_y * cos(a_rhoz * PI * z / L) * cos(a_vy * PI * y / L) + 0.4e1 * a_rhoz * a_wz * rho_z * w_z * cos(a_rhoz * PI * z / L) * sin(a_wz * PI * z / L)) * MU * (0.3e1 * B_mu
* R * RHO + P) * PI * PI * W / (B_mu * R * RHO + P) * pow(L, -0.2e1) / RHO / 0.6e1 - a_pz * PI * p_z * Gamma * W * sin(a_pz * PI * z / L) / (Gamma - 0.1e1) / L - (0.3e1 * a_px * a_uy * p_x *
u_y * sin(a_px * PI * x / L) * sin(a_uy * PI * y / L) + 0.3e1 * a_px * a_vx * p_x * v_x * sin(a_px * PI * x / L) * sin(a_vx * PI * x / L) - 0.2e1 * a_py * a_ux * p_y * u_x * cos(a_py * PI * y
/ L) * cos(a_ux * PI * x / L) + 0.4e1 * a_py * a_vy * p_y * v_y * cos(a_py * PI * y / L) * cos(a_vy * PI * y / L) + 0.2e1 * a_py * a_wz * p_y * w_z * cos(a_py * PI * y / L) * sin(a_wz * PI *
z / L) - 0.3e1 * a_pz * a_vz * p_z * v_z * sin(a_pz * PI * z / L) * cos(a_vz * PI * z / L) - 0.3e1 * a_pz * a_wy * p_z * w_y * sin(a_pz * PI * z / L) * cos(a_wy * PI * y / L)) * (0.3e1 * B_mu
* R * RHO + P) * MU * PI * PI * V / (B_mu * R * RHO + P) * pow(L, -0.2e1) / P / 0.6e1 + a_py * PI * p_y * Gamma * V * cos(a_py * PI * y / L) / (Gamma - 0.1e1) / L - (0.3e1 * a_rhox * a_uy *
rho_x * u_y * cos(a_rhox * PI * x / L) * sin(a_uy * PI * y / L) + 0.3e1 * a_rhox * a_vx * rho_x * v_x * cos(a_rhox * PI * x / L) * sin(a_vx * PI * x / L) - 0.2e1 * a_rhoy * a_ux * rho_y * u_x
* sin(a_rhoy * PI * y / L) * cos(a_ux * PI * x / L) + 0.4e1 * a_rhoy * a_vy * rho_y * v_y * sin(a_rhoy * PI * y / L) * cos(a_vy * PI * y / L) + 0.2e1 * a_rhoy * a_wz * rho_y * w_z *
sin(a_rhoy * PI * y / L) * sin(a_wz * PI * z / L) - 0.3e1 * a_rhoz * a_vz * rho_z * v_z * cos(a_rhoz * PI * z / L) * cos(a_vz * PI * z / L) - 0.3e1 * a_rhoz * a_wy * rho_z * w_y * cos(a_rhoz
* PI * z / L) * cos(a_wy * PI * y / L)) * MU * (0.3e1 * B_mu * R * RHO + P) * PI * PI * V / (B_mu * R * RHO + P) * pow(L, -0.2e1) / RHO / 0.6e1 + (0.4e1 * a_rhox * a_ux * rho_x * u_x *
cos(a_rhox * PI * x / L) * cos(a_ux * PI * x / L) - 0.2e1 * a_rhox * a_vy * rho_x * v_y * cos(a_rhox * PI * x / L) * cos(a_vy * PI * y / L) + 0.2e1 * a_rhox * a_wz * rho_x * w_z * cos(a_rhox
* PI * x / L) * sin(a_wz * PI * z / L) + 0.3e1 * a_rhoy * a_uy * rho_y * u_y * sin(a_rhoy * PI * y / L) * sin(a_uy * PI * y / L) + 0.3e1 * a_rhoy * a_vx * rho_y * v_x * sin(a_rhoy * PI * y /
L) * sin(a_vx * PI * x / L) - 0.3e1 * a_rhoz * a_uz * rho_z * u_z * cos(a_rhoz * PI * z / L) * sin(a_uz * PI * z / L) + 0.3e1 * a_rhoz * a_wx * rho_z * w_x * cos(a_rhoz * PI * z / L) *
cos(a_wx * PI * x / L)) * MU * (0.3e1 * B_mu * R * RHO + P) * PI * PI * U / (B_mu * R * RHO + P) * pow(L, -0.2e1) / P / 0.6e1 - (a_px * PI * p_x * Gamma * U * sin(a_px * PI * x / L) / (Gamma
- 0.1e1) / L + (0.4e1 * a_px * a_ux * p_x * u_x * sin(a_px * PI * x / L) * cos(a_ux * PI * x / L) - 0.2e1 * a_px * a_vy * p_x * v_y * sin(a_px * PI * x / L) * cos(a_vy * PI * y / L) + 0.2e1 *
a_px * a_wz * p_x * w_z * sin(a_px * PI * x / L) * sin(a_wz * PI * z / L) + 0.3e1 * a_py * a_uy * p_y * u_y * cos(a_py * PI * y / L) * sin(a_uy * PI * y / L) + 0.3e1 * a_py * a_vx * p_y * v_x
* cos(a_py * PI * y / L) * sin(a_vx * PI * x / L) - 0.3e1 * a_pz * a_uz * p_z * u_z * sin(a_pz * PI * z / L) * sin(a_uz * PI * z / L) + 0.3e1 * a_pz * a_wx * p_z * w_x * sin(a_pz * PI * z /
L) * cos(a_wx * PI * x / L)) * (0.3e1 * B_mu * R * RHO + P) * MU * PI * PI * U / (B_mu * R * RHO + P) * pow(L, -0.2e1) / P / 0.6e1 - (0.3e1 * a_px * a_uz * p_x * u_z * sin(a_px * PI * x / L)
* sin(a_uz * PI * z / L) - 0.3e1 * a_px * a_wx * p_x * w_x * sin(a_px * PI * x / L) * cos(a_wx * PI * x / L) + 0.3e1 * a_py * a_vz * p_y * v_z * cos(a_py * PI * y / L) * cos(a_vz * PI * z /
L) + 0.3e1 * a_py * a_wy * p_y * w_y * cos(a_py * PI * y / L) * cos(a_wy * PI * y / L) + 0.2e1 * a_pz * a_ux * p_z * u_x * sin(a_pz * PI * z / L) * cos(a_ux * PI * x / L) + 0.2e1 * a_pz *
a_vy * p_z * v_y * sin(a_pz * PI * z / L) * cos(a_vy * PI * y / L) + 0.4e1 * a_pz * a_wz * p_z * w_z * sin(a_pz * PI * z / L) * sin(a_wz * PI * z / L)) * (0.3e1 * B_mu * R * RHO + P) * MU *
PI * PI * W / (B_mu * R * RHO + P) * pow(L, -0.2e1) / P / 0.6e1;
```

Manufactured Analytical Solutions Abstractions Library

Goal: Provide a repository and standardized interface for MMS usage

High Priority:

- Extreme fidelity to generated MMS
- Portability
- Traceability
- Extensible

Low Priority:

- Speed/Performance

Verifying the “Verifier”

Precision is not negotiable.

MASA Testing

- Error target $<$ 1e-15
  - Absolute error on local machines
  - Relative error (other)
  - On all supported compiler sets
- -O0 not sufficient
  - -fp-model precise (Intel)
  - -fno-unsafe-math-optimizations (GNU)
  - -Kieee -Mnofpapprox (PGI)
- “make check”
  - Run by Buildbot every two hours

```
[nick@magus trunk]$ make check
----------------------------------------
Initializing MASA Tests
----------------------------------------
PASS: init.sh
PASS: misc
PASS: fail_cond
PASS: catch_exception
PASS: register
PASS: poly
PASS: uninit
PASS: vec
PASS: purge
PASS: heat_const_steady
PASS: euler1d

.           .
.           .
.           .

----------------------------------------
Finalizing MASA Tests
----------------------------------------
==================
All 65 tests passed
==================
```

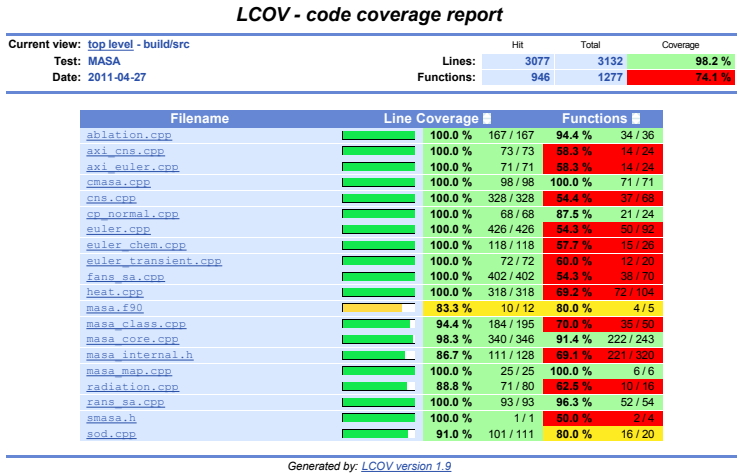
Software Library Snapshot

Software Environment

- Built with: Autotools, C++
- Supports Intel, GNU, Portland Group compilers
- C/C++/Fortran/Python interfaces
- Python interfaces generated with SWIG

Testing

- GIT: version control (github)
- TravisCI: automated testing
- GCOV: line coverage
  - 15,826 lines of code
  - 13,195 lines of testing
  - 98%+ line coverage



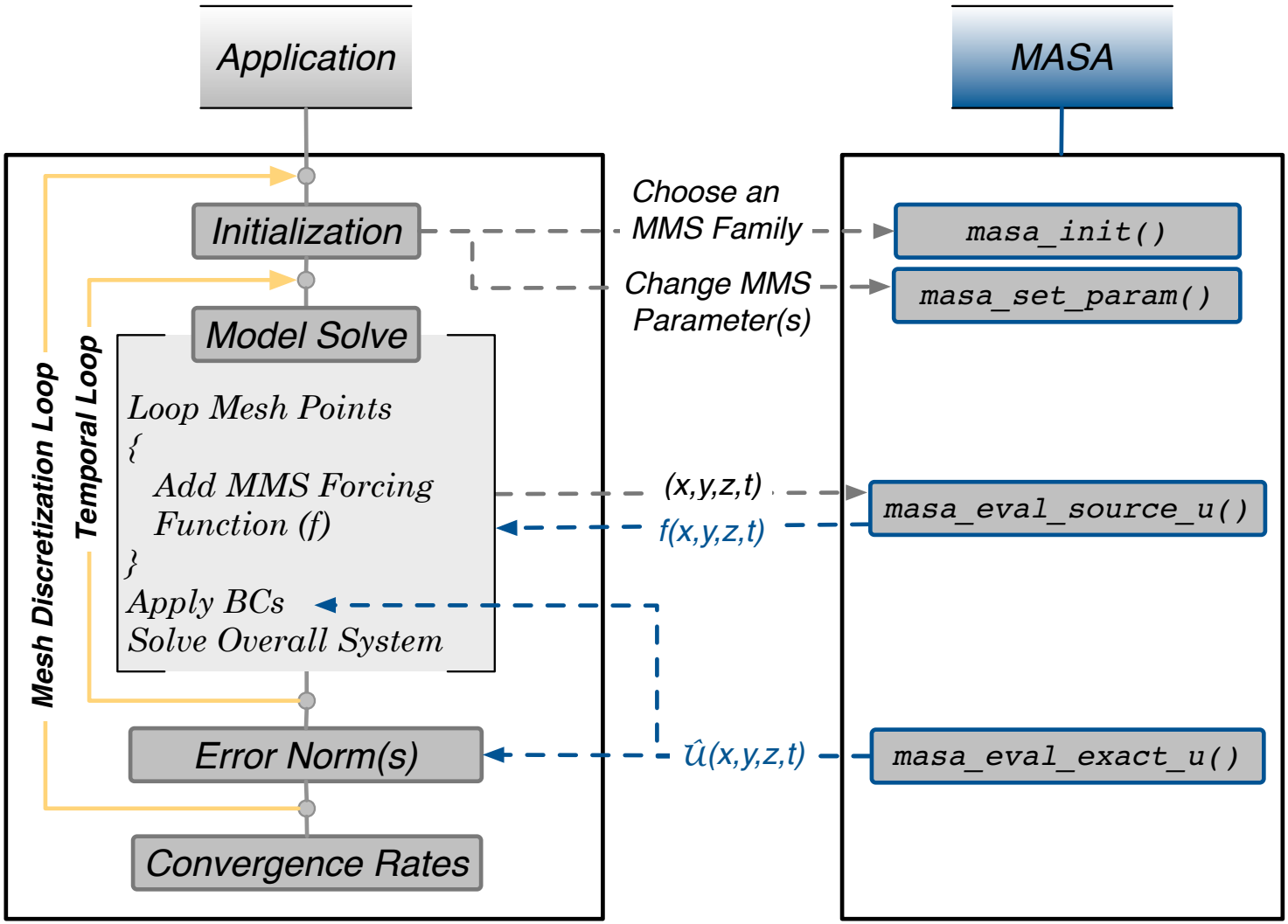
LCOV - code coverage report

			Hit	Total	Coverage
Current view:	top level - build/src	**Lines:**	3077	3132	98.2 %
Test:	MASA	**Functions:**	946	1277	74.1 %
Date:	2011-04-27				

Filename	Line Coverage		Functions	
ablation.cpp	100.0 %	167 / 167	94.4 %	34 / 36
axi_cns.cpp	100.0 %	73 / 73	58.3 %	14 / 24
axi_euler.cpp	100.0 %	71 / 71	58.3 %	14 / 24
cmasa.cpp	100.0 %	98 / 98	100.0 %	71 / 71
cns.cpp	100.0 %	328 / 328	54.4 %	37 / 68
cp_normal.cpp	100.0 %	68 / 68	87.5 %	21 / 24
euler.cpp	100.0 %	426 / 426	54.3 %	50 / 92
euler_chem.cpp	100.0 %	118 / 118	57.7 %	15 / 26
euler_transient.cpp	100.0 %	72 / 72	60.0 %	12 / 20
fans_sa.cpp	100.0 %	402 / 402	54.3 %	38 / 70
heat.cpp	100.0 %	318 / 318	69.2 %	72 / 104
masa.f90	83.3 %	10 / 12	80.0 %	4 / 5
masa_class.cpp	94.4 %	184 / 195	70.0 %	35 / 50
masa_core.cpp	98.3 %	340 / 346	91.4 %	222 / 243
masa_internal.h	86.7 %	111 / 128	69.1 %	221 / 320
masa_map.cpp	100.0 %	25 / 25	100.0 %	6 / 6
radiation.cpp	88.8 %	71 / 80	62.5 %	10 / 16
rans_sa.cpp	100.0 %	93 / 93	96.3 %	52 / 54
smasa.h	100.0 %	1 / 1	50.0 %	2 / 4
sod.cpp	91.0 %	101 / 111	80.0 %	16 / 20

Generated by: LCOV version 1.9

General Verification Approach Using MMS and MASA



C: What you need from MASA

```
#include <masa.h>

int main()
{
  err += masa_init("laplace example","laplace_2d");

  // grab / set parameter values
  Lx = masa_get_param("Lx");
  masa_set_param("Ly",42.0);

  for(int i=0;i<nx;i++)
    for(int j=0;j<nx;j++)
      {
        x=i*dx;
        y=j*dy;

        // source term
        ffield    = masa_eval_2d_source_f (x,y);

        // manufactured solution
        phi_field = masa_eval_2d_exact_phi(x,y);

       } // finished iterating over space
} //end program
```

Available Solutions in MASA 0.44.0

Equations	Dimensions	Time
Euler	1,2,3, axi	Transient, Steady
Non-linear heat conduction	1,2,3	Transient, Steady
Navier-Stokes	1,2,3, axi	Transient, Steady
N-S + Sutherland	3	Transient, Steady
N-S + ablation	1	Transient, Steady
Burgers	2	Transient, Steady
Sod Shock Tube	1	Transient
Euler + chemistry	1	Steady
RANS: Spalart-Allmaras	1	Steady
FANS: SA	2	Steady
FANS: SA + wall	2	Steady
Radiation	1	Steady
SMASA: Gaussian	1	Steady

Or users can import own solutions.

MASA PDE Examples

Source Terms: Euler

```
// Arbitrary manufactured solutions
U.template get<0>() = u_0 + u_x * sin(a_ux * PI * x / L)
                          + u_y * cos(a_uy * PI * y / L);
```

$$\nabla \cdot (\rho u) = 0$$

$$\nabla \cdot (eu) + p\nabla \cdot u = 0$$

```
// Mass, momentum and energy
Scalar    Q_rho    = raw_value(divergence(RHO*U));
RawArray  Q_rho_u  = raw_value(divergence(RHO*U.outerproduct(U)) +
                     P.derivatives());
 Scalar   Q_rho_e  = raw_value(divergence((RHO*ET+P)*U));
```

Snapshot

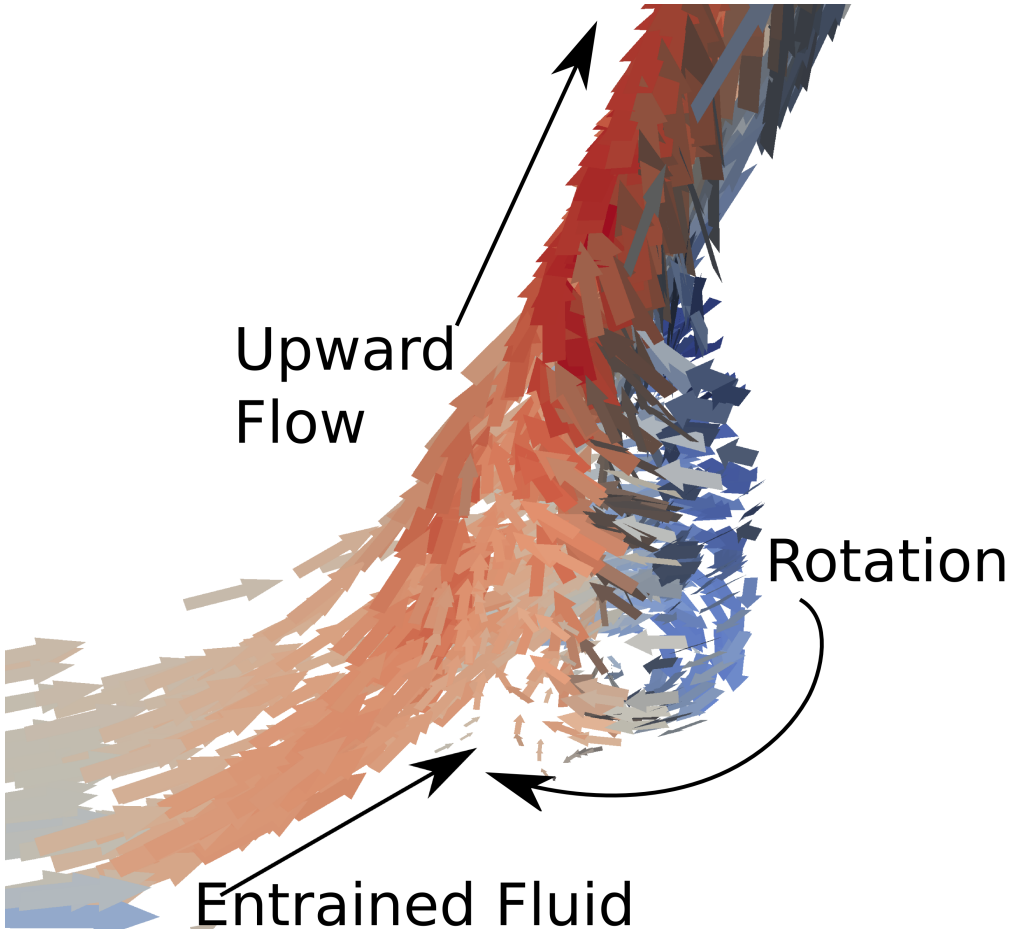
Release

- MASA 0.44.0 current release
- `https://github.com/manufactured-solutions/MASA`
- “Computational Physics Student Summer Workshop” at LANL (2011, 2012, 2013, 2014, 2016).

Publications

- “MASA: a library for verification using manufactured analytical solutions”
- A transient manufactured solution for the compressible Navier-Stokes equations with a power law viscosity
- Manufactured Solutions for the Favre-Averaged Navier-Stokes Equations with Eddy-Viscosity Turbulence Models
- ”A linear regression model for verification of linear problems using Bayesian calibration“ (in prep)

The End



Thank you!

nick@ices.utexas.edu