Numerical Investigation of Synthetic, Buoyancy-Induced Columnar Vortices

Nicholas Malaya

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

July 8th, 2016

Acknowledgment: This material is based upon work supported by the Department of Energy [ARPA-E] under Award Number [DE-FOA-0000670]

The dust devil phenomenon

Turbulent, unstable motions

- Incident solar energy absorbed into ground
- Ground heats air by conduction
- Heated air expands, lowering density, driven upward by force of buoyancy
- Arizona in Summer ($\Delta T = 30$ K)
- $Ra \approx 10^9 - 10^{11} \rightarrow$ unstably stratified fluid
- Velocities can exceed 33 m/s.

How much energy is present in one of these objects?

Energy Estimate [Sinclair]

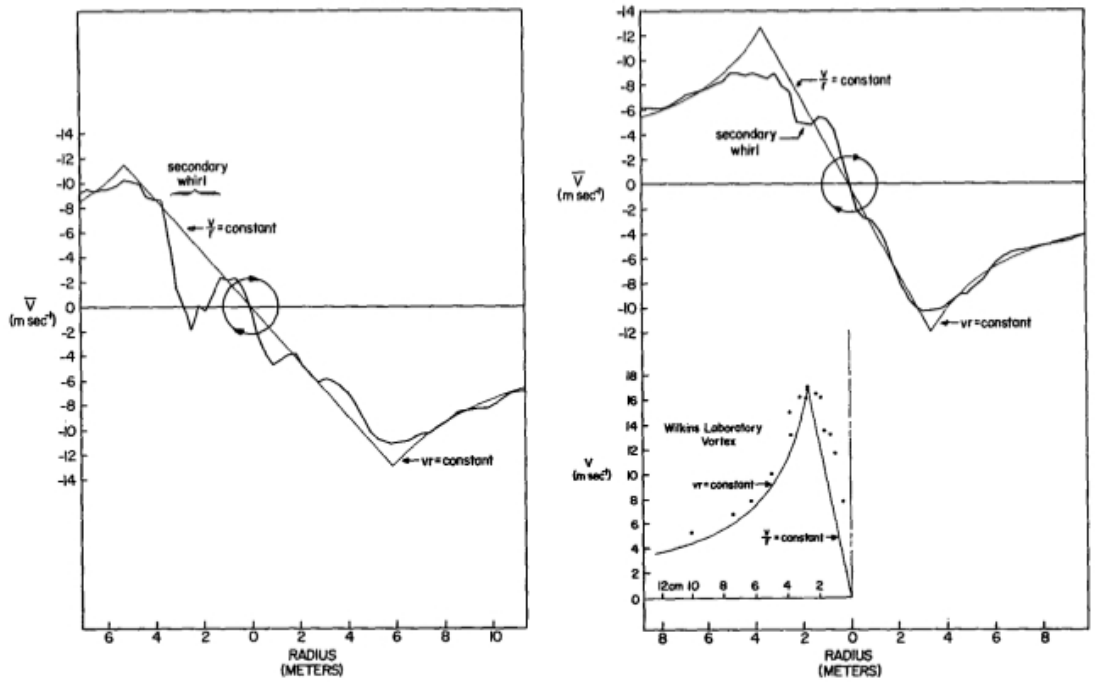
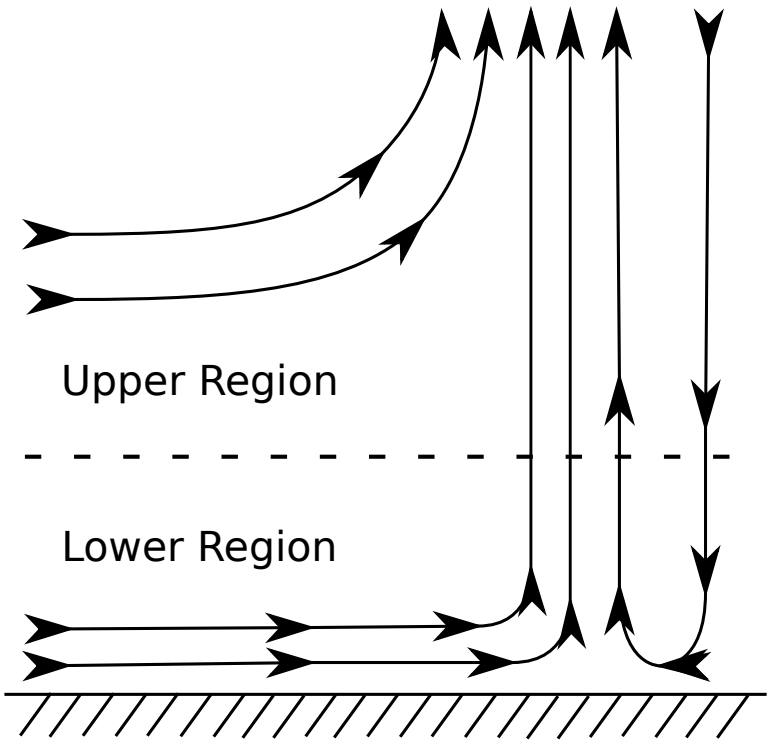


FIG. 11. Radial distribution of the mean tangential velocity at two levels, $z=7$ ft (a) and $z=31$ ft (b), for D-D #1 with superimposed Rankine combined profiles.

- Estimate energy by integrating Sinclair profile
- R $\approx$ 6 meters, $V_\theta \approx 11$ m/s, $V_z \approx 10$ m/s
- Energy flux through horizontal plane $= \frac{1}{2}\rho \int V_z(V_z^2 + V_\theta^2)\, dA$
- $P \approx 24$ kW

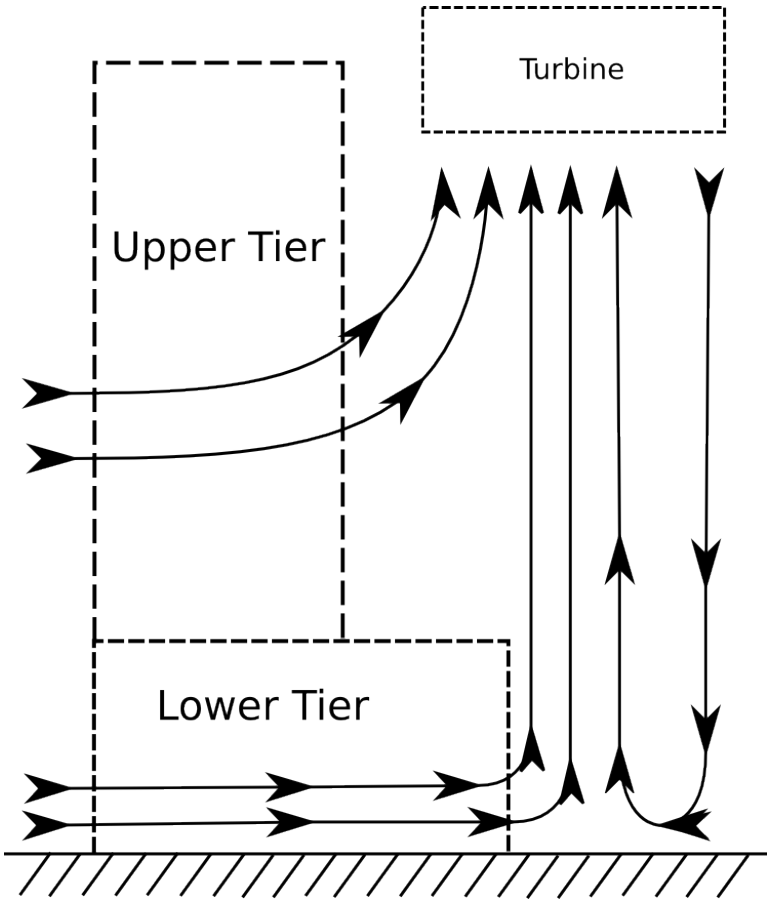
Dust Devil Structure [Sinclair,Kanak,Renno]



Dust devil structure

- Driven by buoyancy of hot ground
- Near surface: radial inflow spinning region (core)
- Downward flow in hot, low-pressure center (“eye”)
- Fluid from higher region entrained (inviscid potential flow region)
- Characterized by strong rotation
  - What generates vorticity?
    - vortex stretching
    - vortex tilting

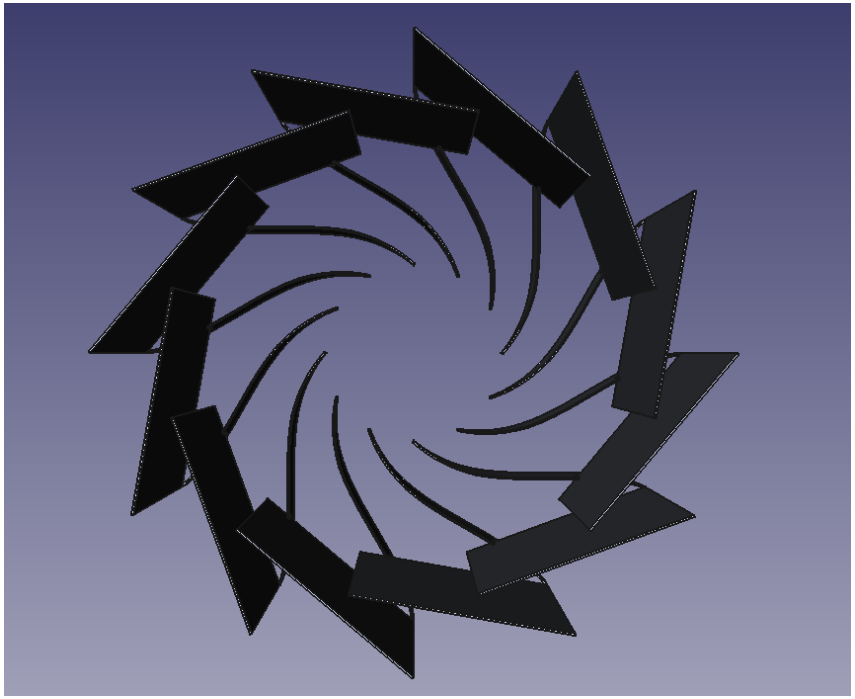
Synthetic Dust Devils

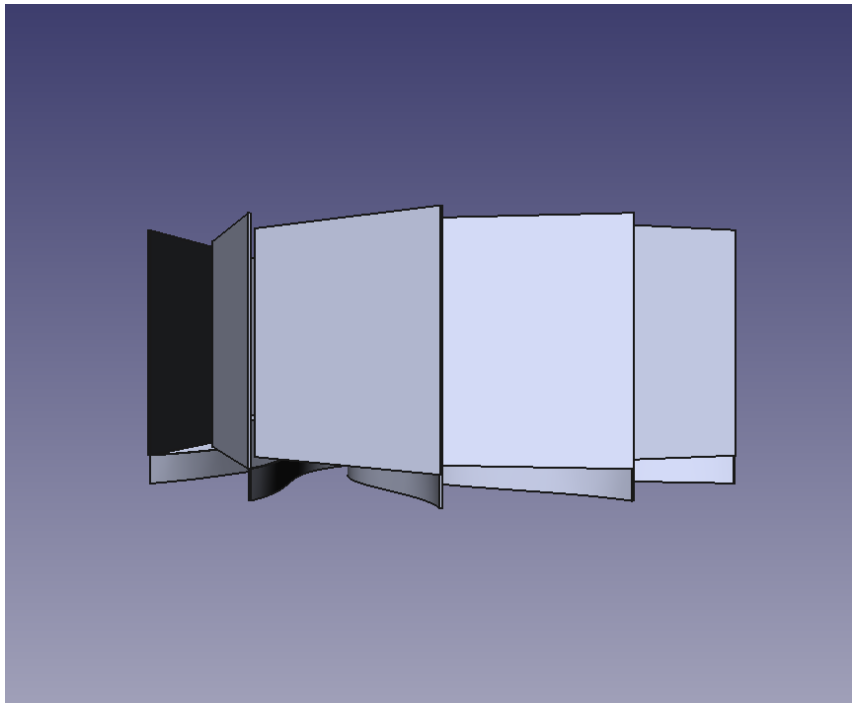


Motivation

- Engineer synthetic vortices for energy generation
  - Two tier configuration mimics natural phenomenon
  - Turbine at upper tier vanes to extract energy
- Difficult to explore space experimentally
  - Field tests in Arizona (GT)
  - expensive to alter configuration
  - difficult to measure everything

Example configuration





Turning Vanes

- Several (typically 12) control surfaces used to turn the flow
- Top tier significantly taller than the bottom
- A cone may be added to the top to contract the flow

Modeling Objectives

System Configuration

- Probe and optimize configuration with CFD
  - Vanes
  - Cone (and solid surfaces)
  - Turbine
- Inform design for 2016 field test
  - Predict energy output
  - Sensitivities to design
  - Sensititivies to conditions

Scenario Space – Not much prior work!

- Atmospheric model
- Impact of buoyancy
- Effect of wind

Dynamical Equations

The equations describing fluid flow with natural convection are,

$$\frac{\partial u}{\partial t} + u \cdot \nabla u = -\frac{1}{\rho}\nabla P + \nu \nabla^2 u - g\frac{T - T_0}{T_0} \tag{1}$$

$$\nabla \cdot u = 0 \tag{2}$$

$$\rho c_p \frac{\partial T}{\partial t} + u \cdot \nabla T = \nabla \cdot (K \nabla T) \tag{3}$$

- Incompressible N-S + Boussinesq buoyancy
- Temperature variation small compared to mean temperature
- Low mach number

However, $Ra \approx 10^9 - 10^{11}$ – Need turbulence model!

Atmospheric Viscosity Model

Introduction to Monin-Obukhov

- Under stationary, homogeneous conditions turbulent quantity $\bar{f}$,

$$\bar{f} = f(z, \frac{g}{T_0}, \rho_0, U_\infty, q, \nu_l, K_l) \tag{4}$$

- 5 parameters - 4 dimensions = 1 dimension-less group:

$$\xi = -\frac{\kappa \frac{g}{T_0} \frac{q}{c_p \rho_0} z}{U_\infty^3} \tag{5}$$

- Interpret as a length scale,

$$\xi = \frac{z}{L_{M-O}}; \quad L_{M-O} = -\frac{{U_\infty}^3}{\kappa \frac{g}{T_0} \frac{q}{c_p \rho_0}} \tag{6}$$

- L_{M-O} is height where shear production $\approx$ buoyancy

Atmospheric Viscosity Model

M-O, cont.

By similarity, mean turbulent quantities are functions of the non-dimensional group,

$$\frac{\bar{f}}{f_{MO}} = \phi(\frac{z}{L_{M-O}}) \tag{7}$$

e.g.

$$\bar{u}(z) = \frac{U_\infty}{\kappa}\phi_u(\frac{z}{L_{M-O}}) \quad \text{and} \quad \frac{\partial \bar{u}(z)}{\partial z} = \frac{U_\infty}{\kappa L_{M-O}}\varphi_u(\frac{z}{L_{M-O}}) \tag{8}$$

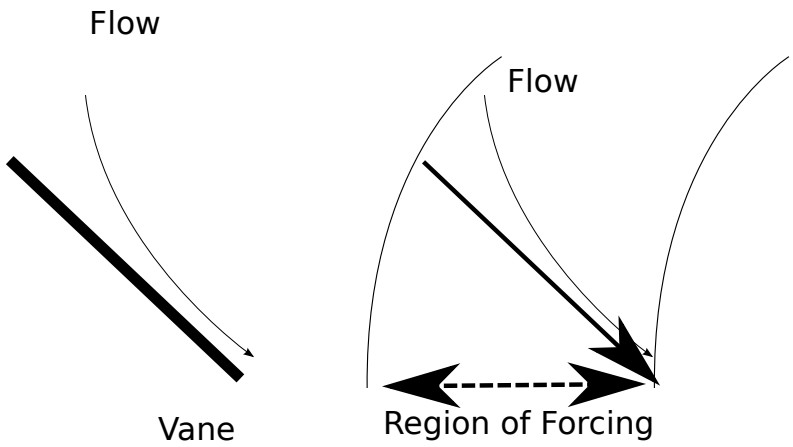
Our interest is in the eddy viscosity, $u'v' = \nu_T \frac{\partial \bar{u}}{\partial z}$ where $\xi \to -\infty$,

$$\nu_T = \frac{1}{C_{\nu_T}} \left(\frac{q}{c_p \rho_0} \frac{g}{T_0} \right)^{\frac{1}{3}} z^{\frac{4}{3}} \text{ for } z \gg L_{M-O}. \tag{9}$$

Turning Vane Representation

Virtual Vanes

- Requirements:
  - Very general formulation
  - No re-meshing
- Volumetric forcing:
  - Vane surface not represented
  - Similar to actuator-disk



Formulation

- Unit normal to vane surface,

$$\mathbf{n}(\mathbf{x}) = \sin\left(\phi(r)\right)\hat{\mathbf{r}} + \cos\left(\phi(r)\right)\hat{\theta} \tag{10}$$

- Apply $\mathbf{f_v}$ to force flow parallel to vanes,

$$\mathbf{f}_v = -\frac{1}{\ell_v}|\mathbf{u}|(\mathbf{u}\cdot\mathbf{n})\mathbf{n} \tag{11}$$

Numerics

Formulation

- Using Galerkin FEM discretization (N-S in weak form)
- Stabilized with SUPG stabilization *al la* Hughes/Becker+Braack
  - Adjoint stabilization scheme (τ)
  - Circumvents Babuska-Brezzi (we are using linear elements)
- Time discretization via backward Euler (if not steady)
- Newton method used to solve resulting non-linear implicit system

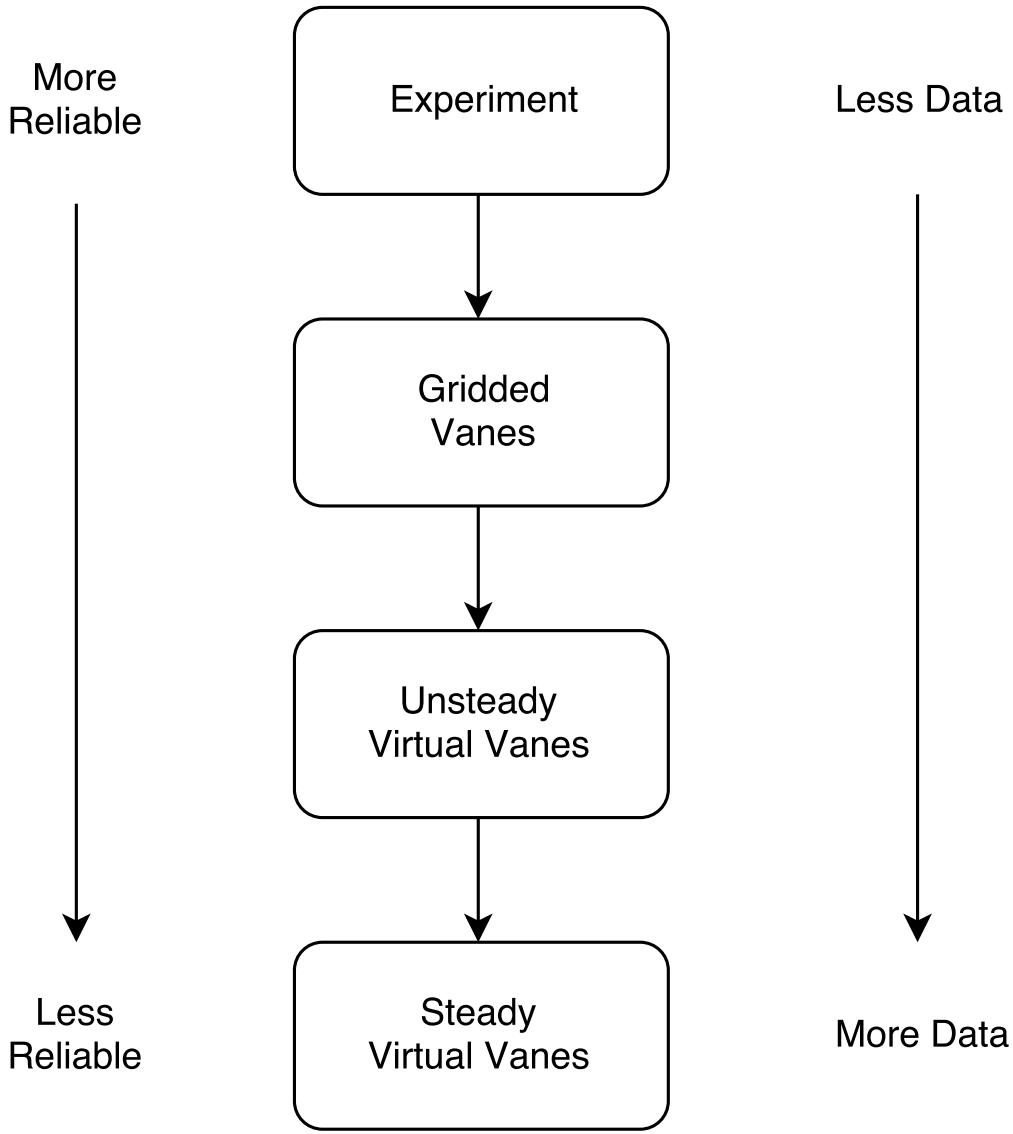
Software

- Using GRINS(+Libmesh) library developed by Bauman and Stogner

Hardware

- Most of the runs performed on LS4,LS5, Stampede at TACC
- 264-528 processing cores
- Several million degrees of freedom for each run, locally $O(10^4)$

Modeling Hierarchy



Steady vs. Transient Virtual Vanes

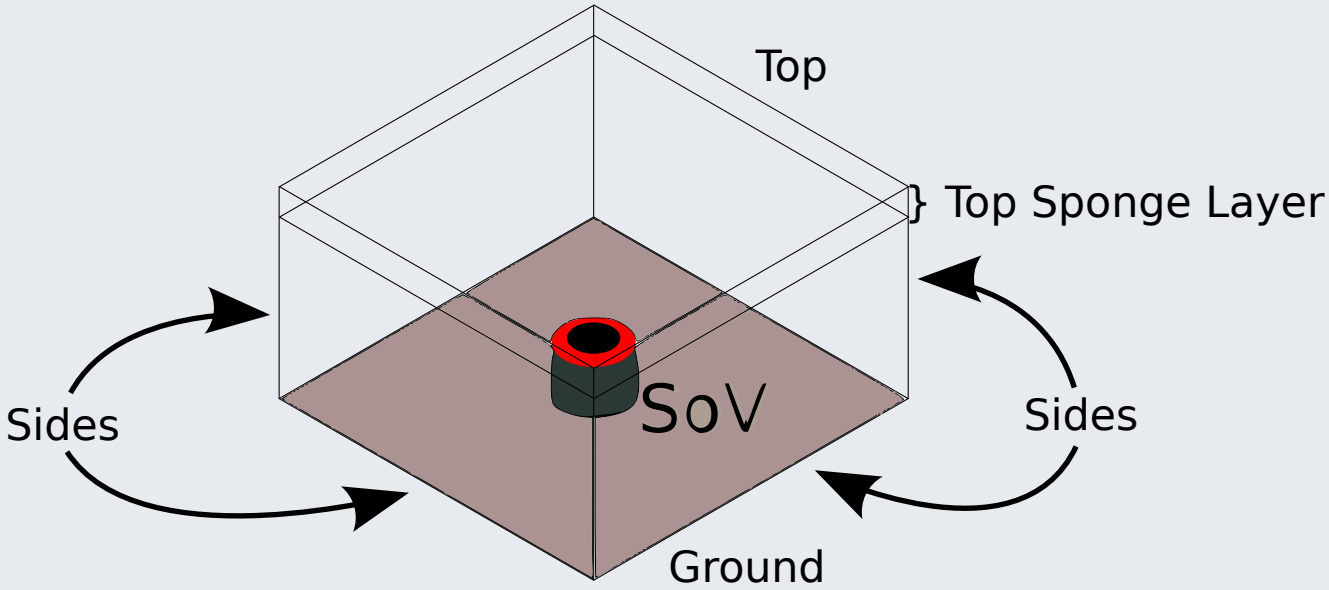
Unsteady VV

- Transient, like V-LES
  - captures dynamics of plume and wake
- Solutions temporally averaged
- Run-time ≈ 12 hours

Steady VV

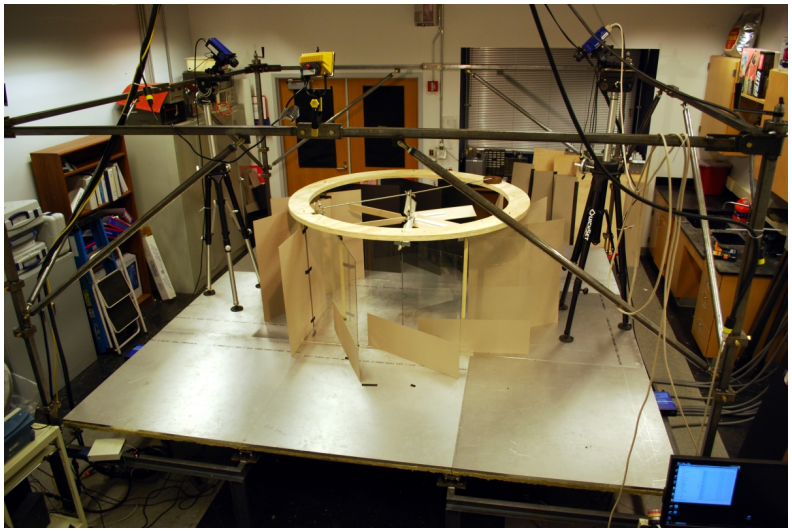
- Steady, like RANS (no $\frac{\partial()}{\partial t}$)
  - no dynamics
- Run-time ≈ 2 minutes
- Critical design tool

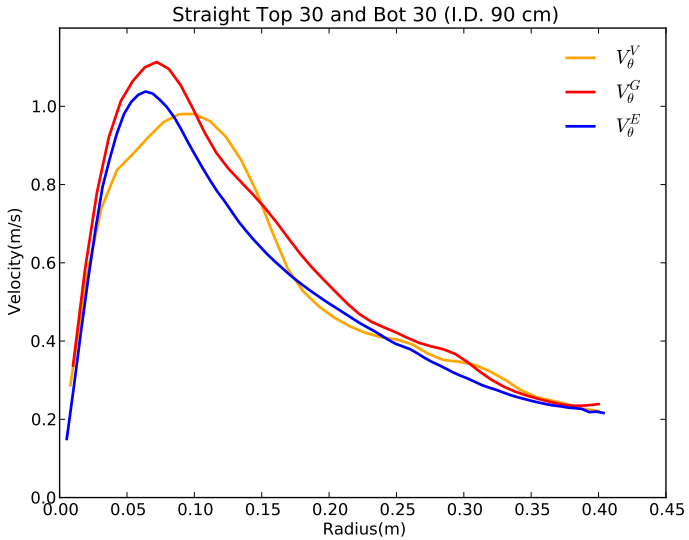
No Wind (Thermal-Only)



- Domain extents scaled by system diameter
- Sides are periodic
- Sponge layers are finite thickness high viscosity boundaries
- Top has special mixed B.C.

Tabletop Experiment Validation





Laboratory Comparison of Azimuthal Velocity

- Slightly different B.C.s (inflow)
- Straight, single-tier, thirty degree vanes
- Red: Gridded, Blue: Experiment, Gold: Virtual
- Gridded consistent with experimental data
- Virtual vane solution more diffuse, incorrect peak predict

Thermal-Only Virtual Vanes

Figure: Bottom Vanes

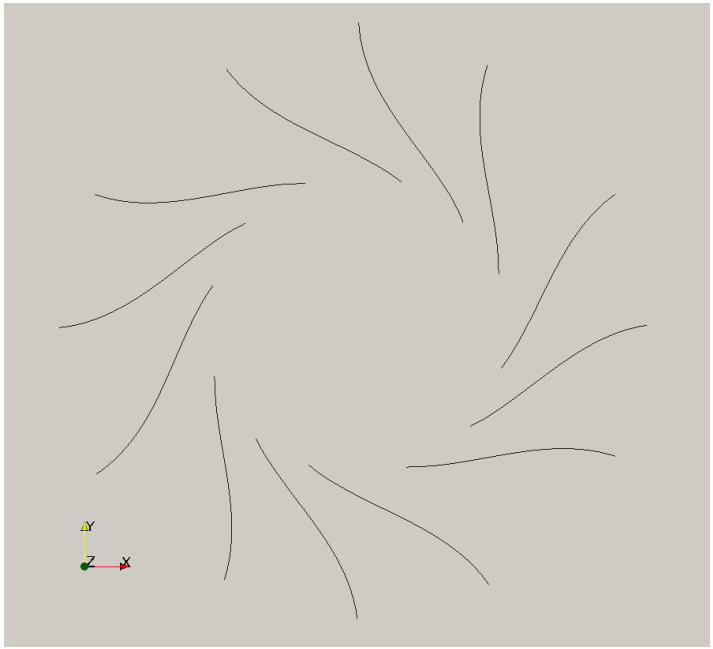


Figure: Top Vanes

Two Tier Configuration

- Curves drawn by advancing particle through forcing field
- No cone
- The top and bottom tier final angles are $70°$ and $85°$, respectively

Probing the Solution: Vertical Slices

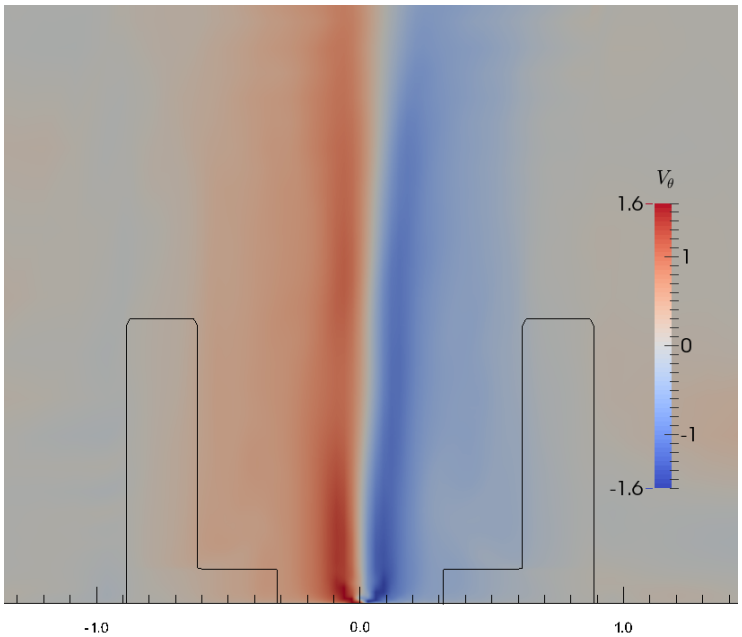


Figure: Azimuthal velocity

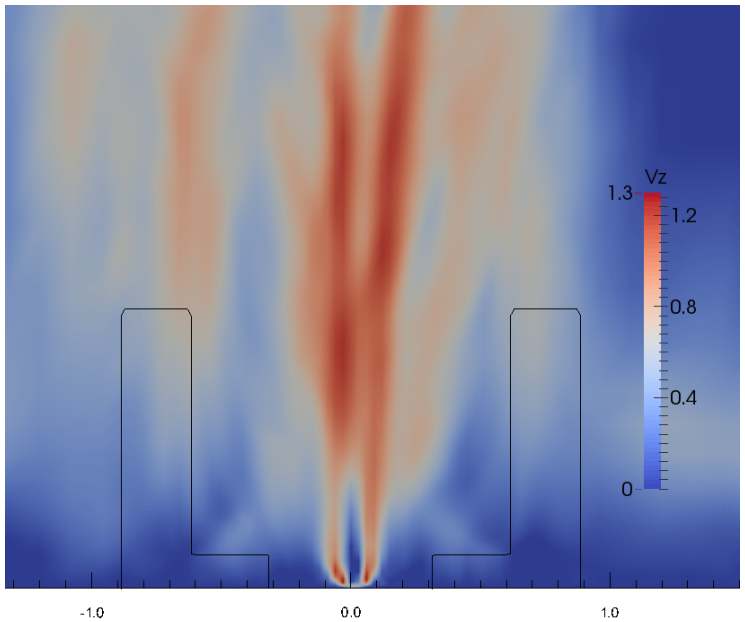
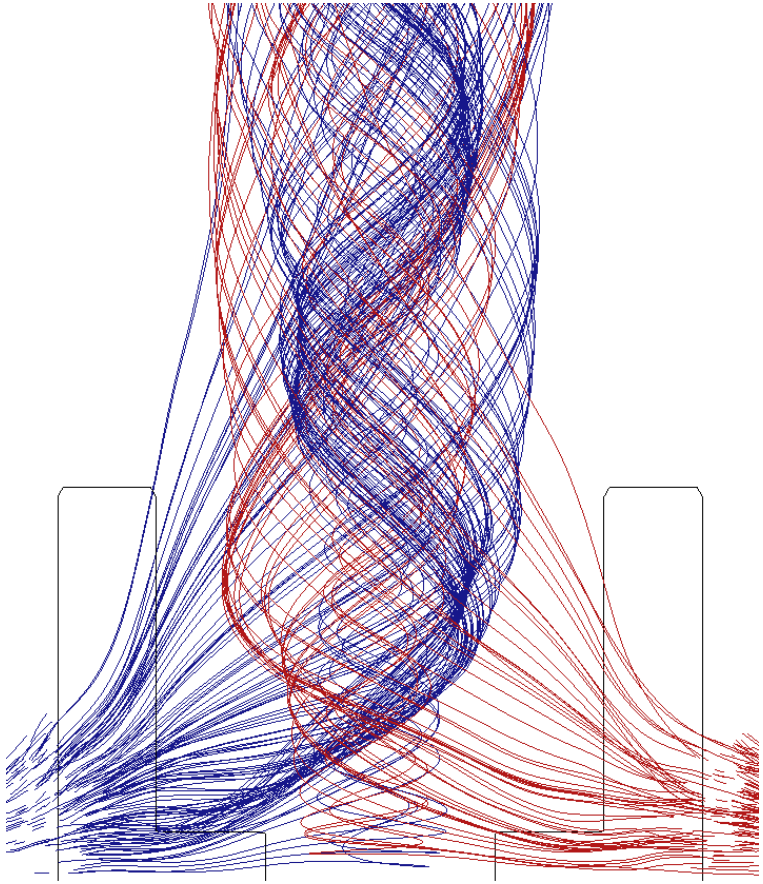


Figure: Vertical velocity

Velocity Observations

- Slices are temporally averaged virtual vane cases
- Virtual vane forcing region outlined in black
- Velocity highest in tight region near center

Solution structure



Particle Paths

- “Seed” particles outside of device
- RK4 to advance the particles through the velocity field

Probing the Solution: Vertical Slices

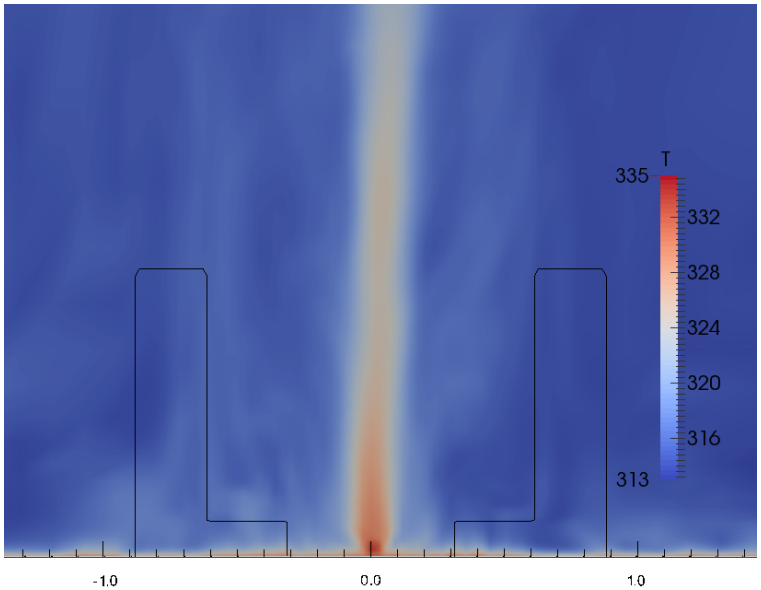


Figure: Temperature

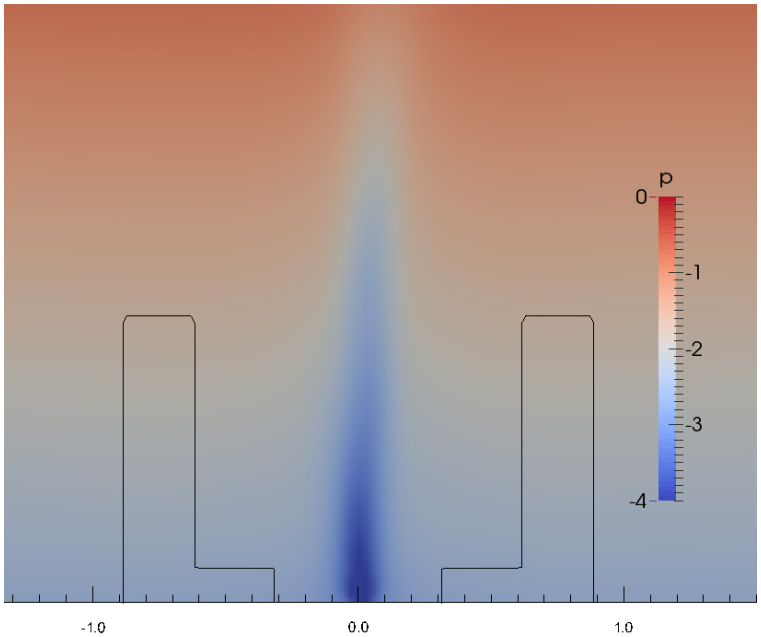


Figure: Pressure

Pressure and Temperature Observations

- Tight velocity core contains hot, low pressure “eye”

Probing the Solution: Horizontal Slices

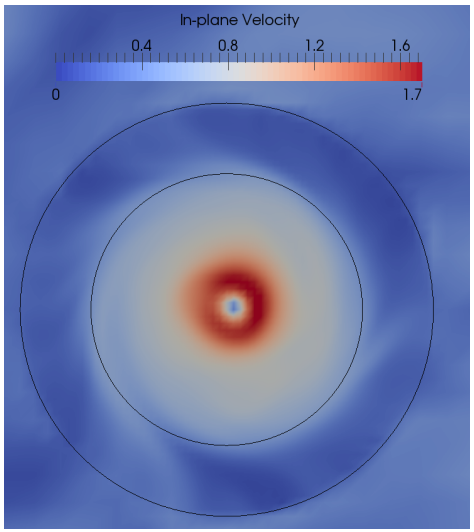


Figure: In-plane Velocity

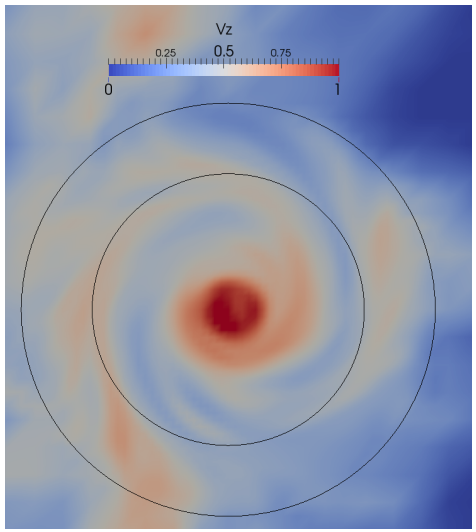


Figure: Vertical Velocity

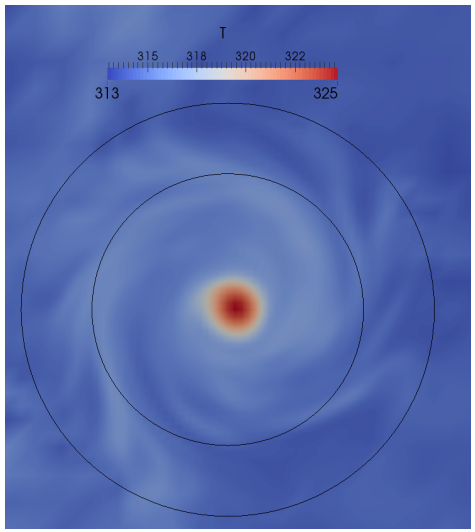
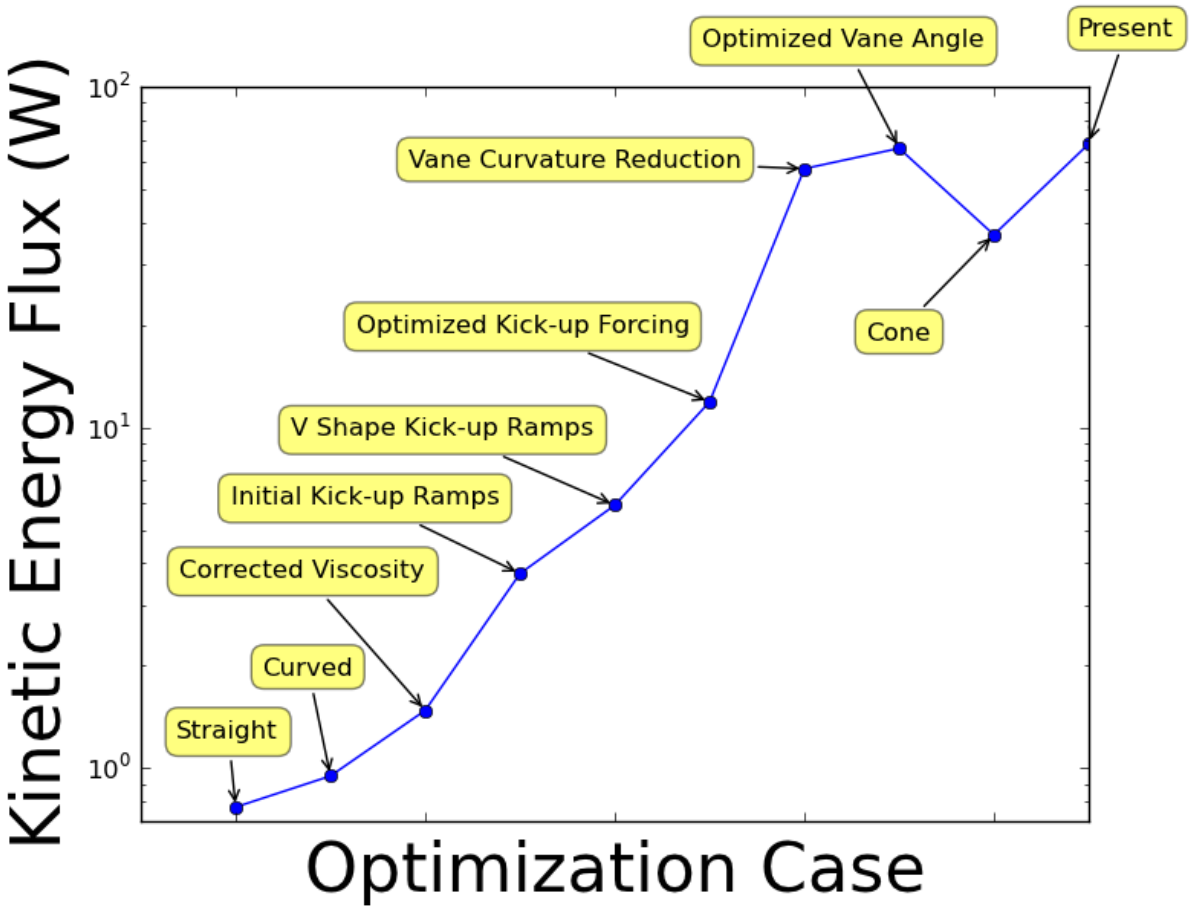


Figure: Temperature

Observations

- Slice taken at the height of the 2nd tier of vanes

Optimizing the Thermal-Only Configuration



Kinetic energy flux,

$$\dot{E} = -\frac{\rho}{2} \int V_z (V_\theta^2 + V_z^2) dA$$

- 78X increase in flux!
- Performed by exploration of design space
- flux < 100 Watts

Estimate of Energy Scaling

- Medium size dust devil: 3m = D, $U = 5$ m/s, $\Delta T = 30$ K, 3 meters tall
- Assume a turbulent boundary layer ($\delta = 10$ cm),

$$u(z) = U \text{ min } \left(\left(\frac{z}{\delta} \right)^7, 1 \right)$$

Boussinesq potential energy flux over the upstream flow [Renno]:

$$E_p = \int u(z)(\rho(z) - \rho_\infty) g z \, dA$$

$$E_p = g \pi R \beta \rho_0 U \Delta T \left[\frac{z_{\text{max}}^2}{2} - \frac{7\delta^2}{18} \right]$$

$$= 103 \text{ Watts}$$

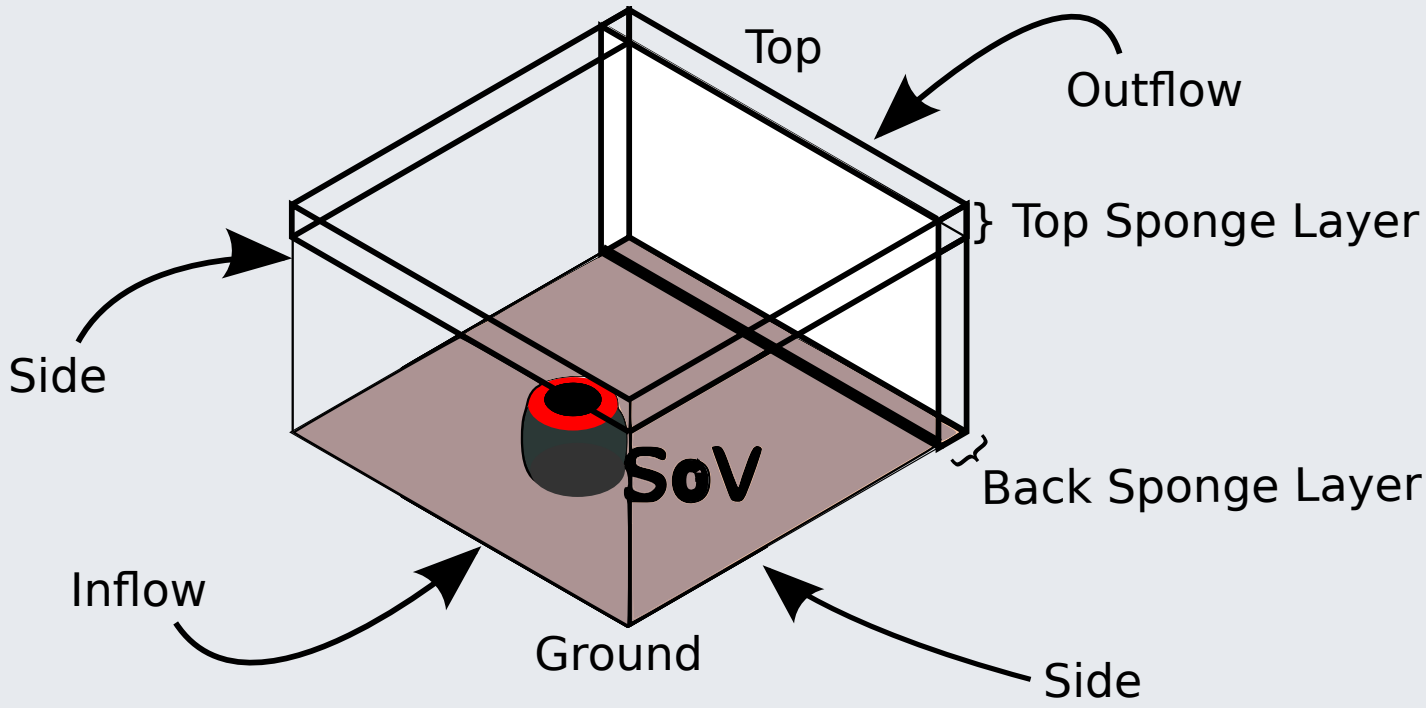
The KE flux: surface integral over upstream face of dust devil:

$$\text{KE} = \int \frac{\vec{V}^2}{2} \rho \vec{V} \cdot \hat{n} \, dA$$

$$\text{KE} = R \rho U^3 \left[z_{\text{max}} - \frac{10}{11} \delta \right]$$

$$= 1144 \text{ Watts}$$

Wind Cases



- Specified Inflow (Dirichlet)
- Top and Outflow have special mixed B.C.
- Sponge layers on Top and Outflow
- Natural (do nothing) boundaries on the side (Neumann)

Simulation Geometry and Boundary Conditions, cont.

Specified Inflow

- 7th order turbulent boundary layer,

$$u_{\text{in}}(z) = U \ \text{min} \left(\left(\frac{z}{\delta} \right)^7, 1 \right)$$

- The thermal inflow is then,

$$T_{\text{in}}(z) = \Delta T \left(1 - \ \text{min} \left(\left(\frac{z}{\delta} \right)^7, 1 \right) \right) + T_0 - 2z/3.$$

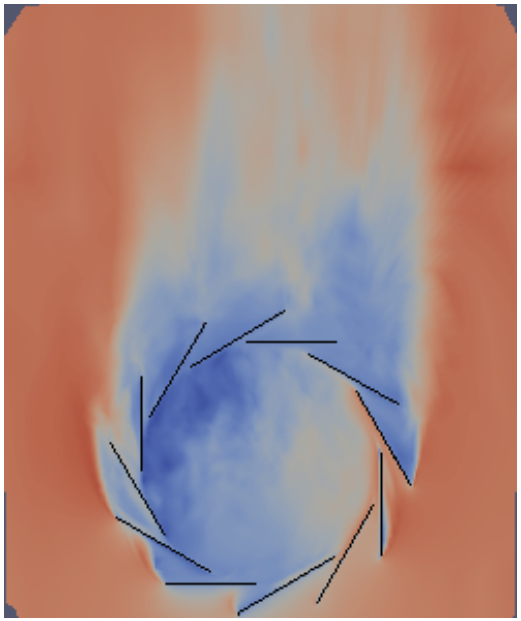
Ground

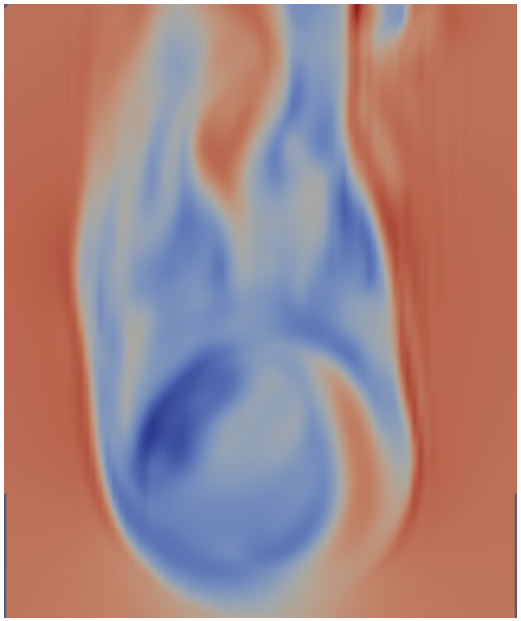
- modeled with a Dirichlet boundary condition such that,

$$\vec{u} = 0 \quad \text{on } \Gamma_G$$
$$T = T_g$$

Cold Wind Tunnel Validation

Streamwise Velocity

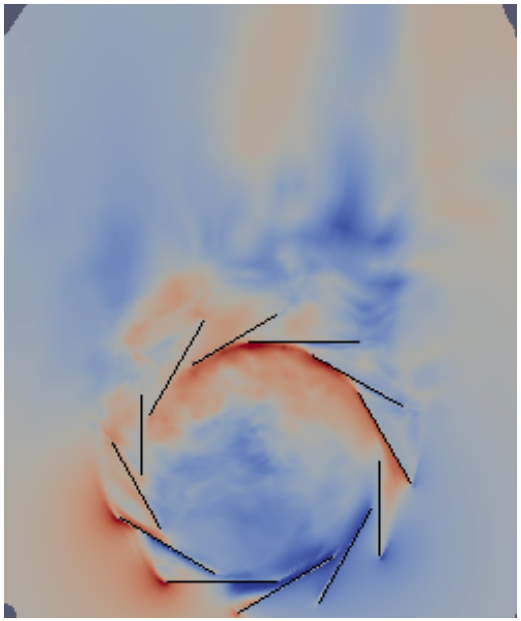


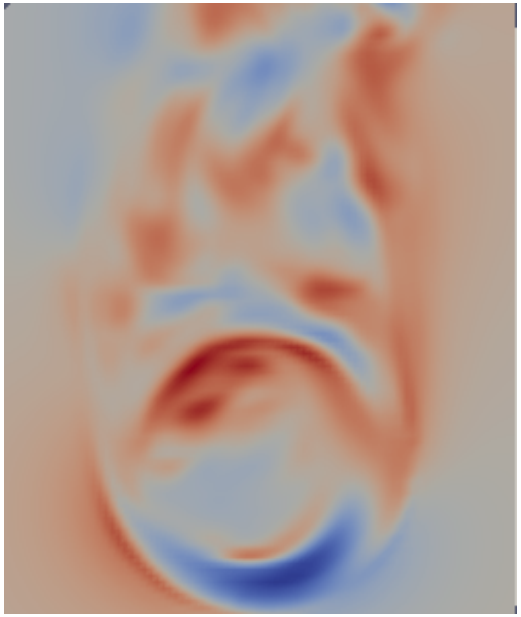


Horizontal slice (midway through the $60°$ vanes)

- Constant velocity wind, no temperature gradient
- Gridded vanes on left, Virtual vanes on right
- Flow penetrates vane region where aligned
- Results qualitatively agree with wind tunnel observations

Wind Tunnel: Spanwise Velocity





Horizontal slice (midway through the $60°$ vanes)

- Gridded Vanes on left, Virtual Vanes on right
- Not particularly accurate in the wake region
- Virtual vanes reproduce direction and magnitude of spanwise velocity
- Flow does not penetrate out back of device

Simulation of August Field Test

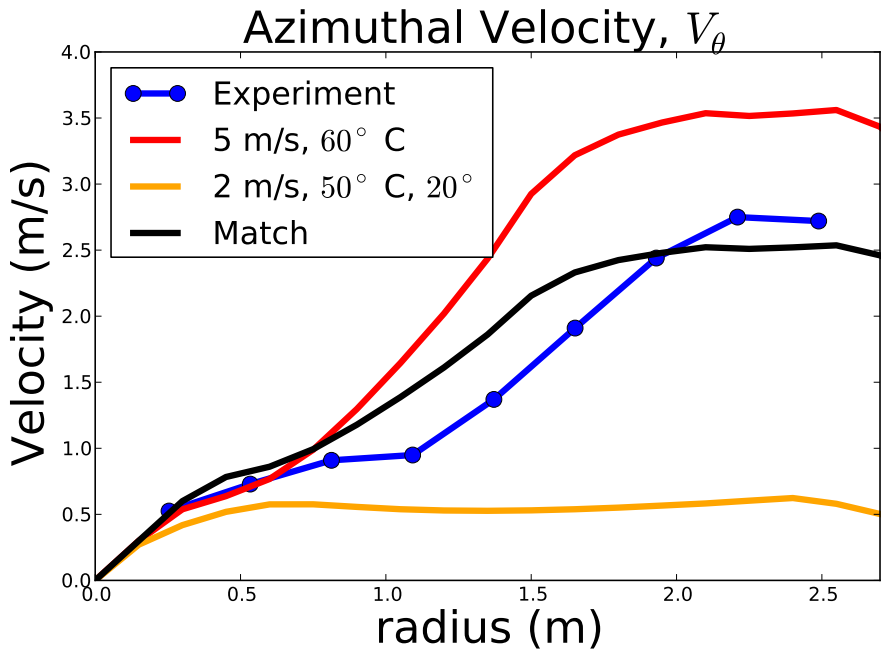
Model Problem

- Emulate field conditions in Arizona, August 2015
- Hot ground, ambient winds
- Lower tier, upper tier, and cone

Field Validation Study

Scenario parameter uncertainty

- Wind speed: 2-5 m/s
- Wind direction($\approx 20°$)
- $T_s = 50 - 60°$ C ($\approx 121 - 140°$ F)
- No boundary layer temperature data (DAQ equipment malfunctioned)



- CFD broadly consistent with field observations
  - Experimental data extremely limited
- Kinetic energy fluxes match to within 10%

Horizontal images at height of the vanes

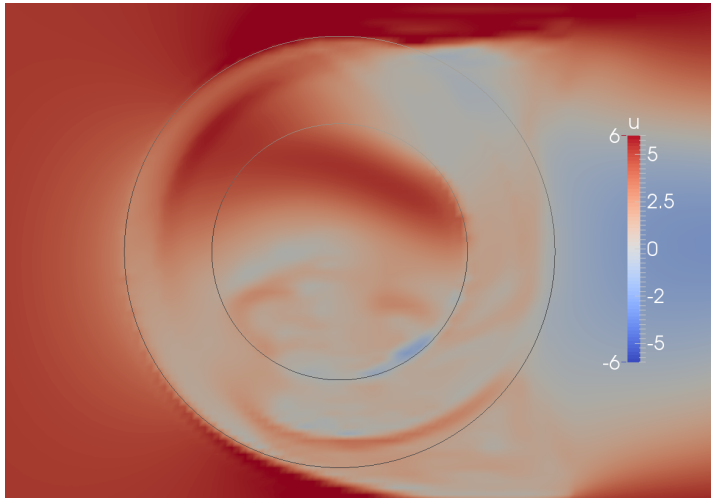


Figure: Streamwise Velocity

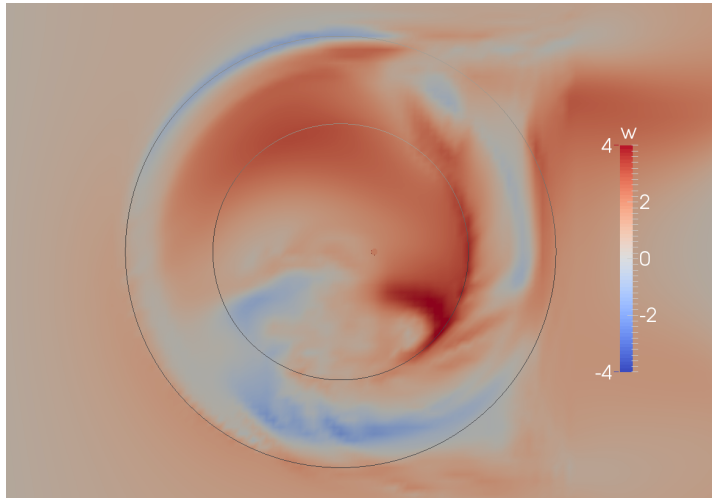
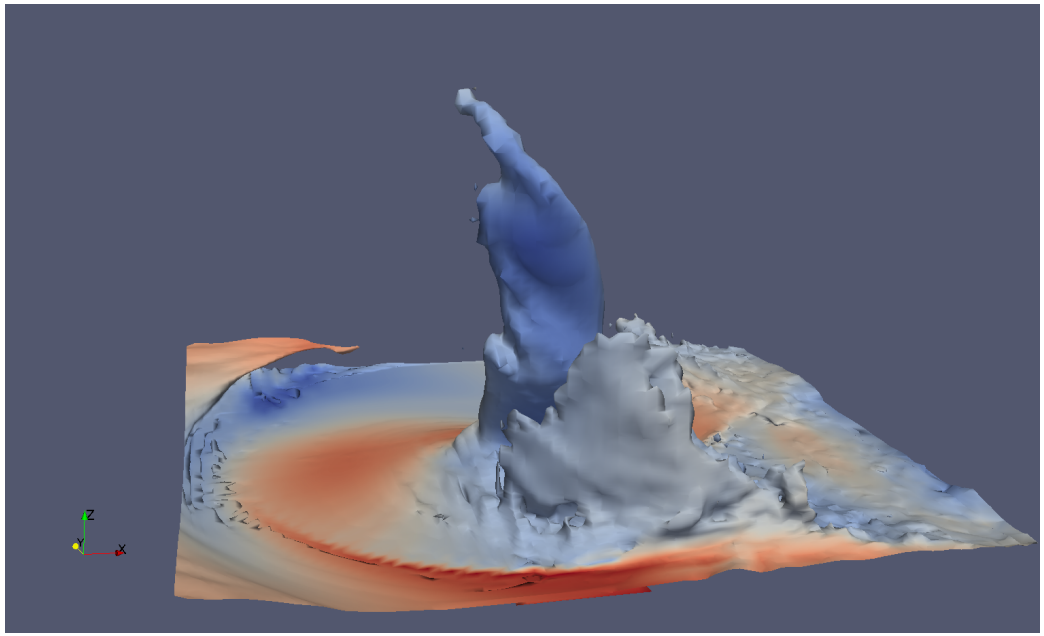


Figure: Vertical Velocity

Observations

- Freestream velocity is 3 m/s
- $\Delta T = 32$ K
- Some rotation visible in center

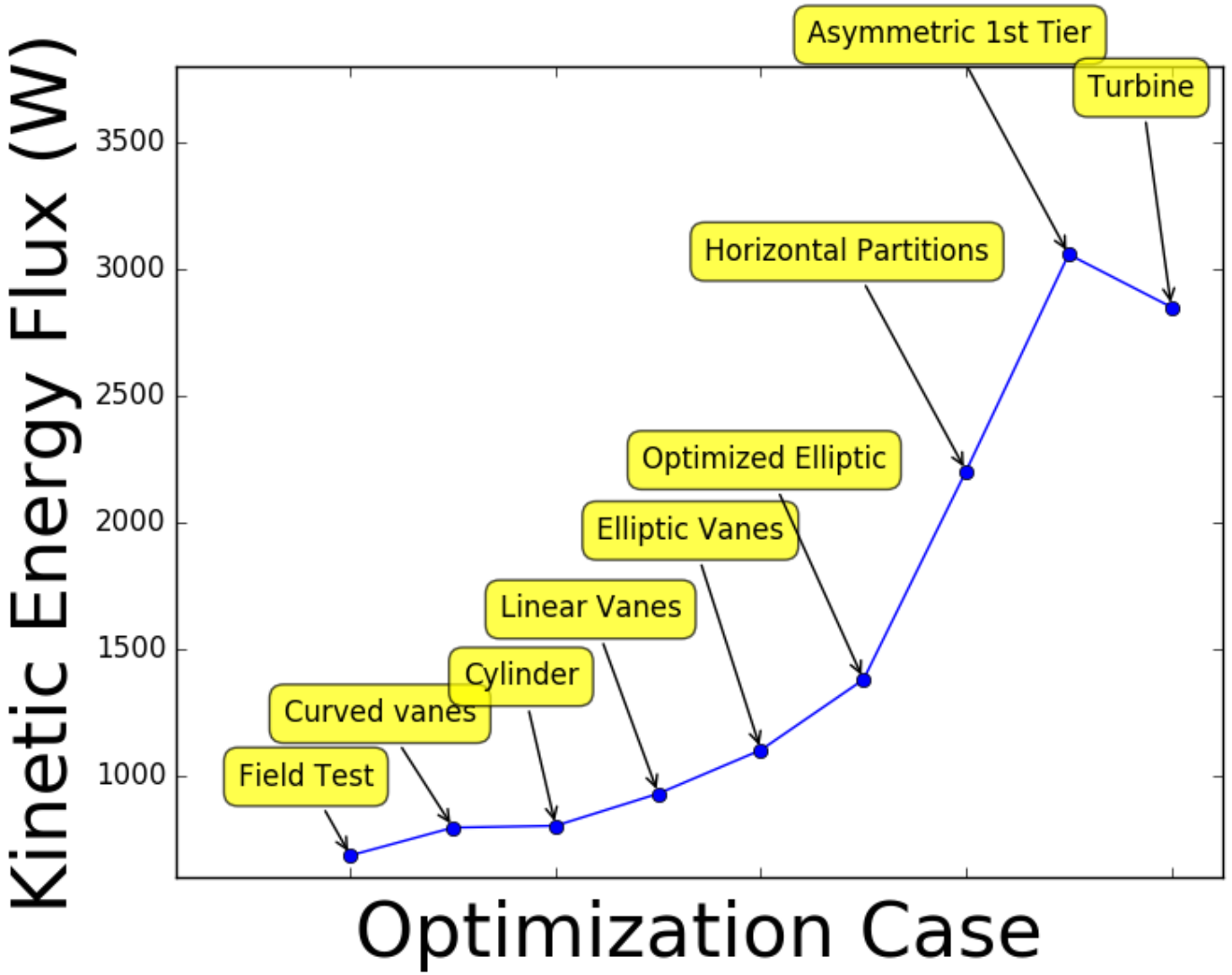
Temperature Plume



Temperature isocontour

- Potential temperature, $\tau(x, y, z) = T(x, y, z) - T_{\text{in}}(z)$
- Iso-contour corresponds to $\tau = 3K$ [Sinclair]
- Colored by streamwise velocity
- 600 Watts of Kinetic Energy Flux

Optimization of the apparatus for Field



Present Field Configuration

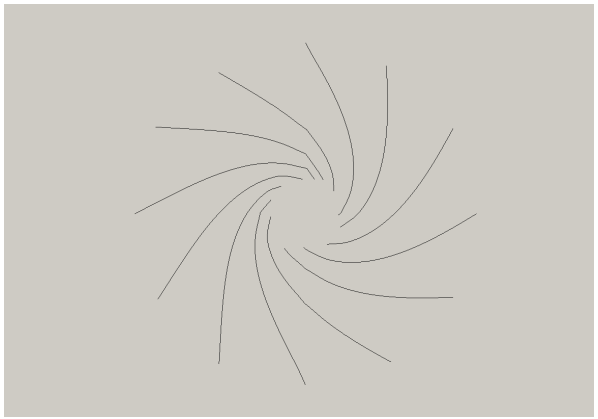


Figure: Bottom Vanes

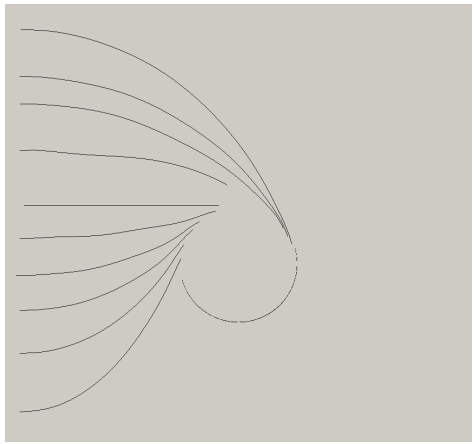


Figure: Top Vanes

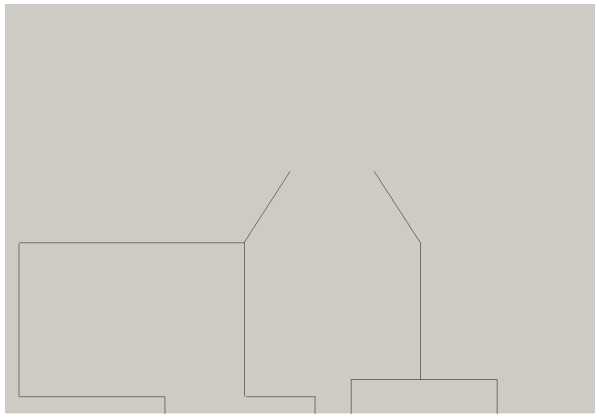


Figure: Vertical Slice

Two Tier Configuration

- Highly asymmetric second tier
- Vanes arranged to align with incoming wind
- Bottom tier asymmetric in height and angle

Present Field Results

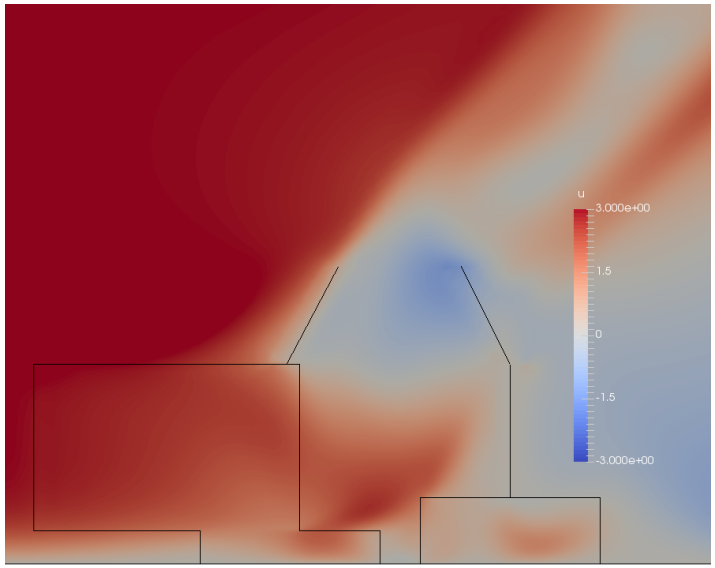


Figure: Streamwise velocity

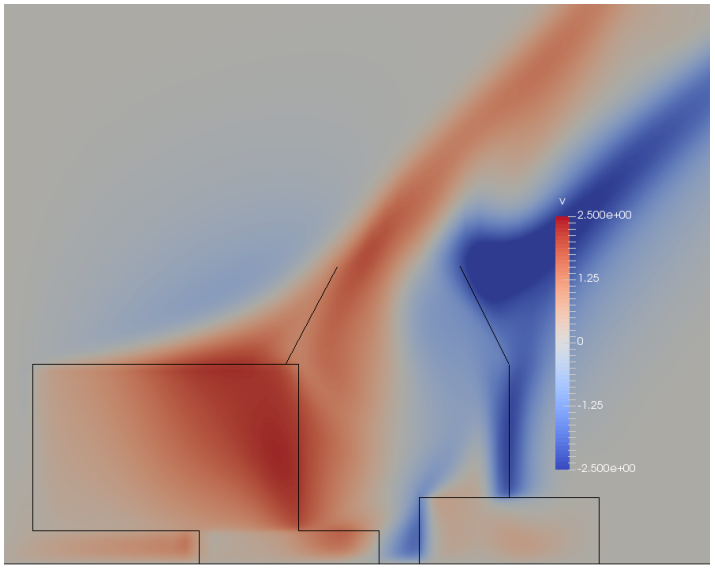


Figure: out of plane velocity

Observations

- Configuration space explored with steady Virtual Vanes
- ≈ 3 kW KE flux

Conclusions

Summary

- Developed modeling approach for domain of interest
- Simulation results consistent with available measurements
- Solution structure is cyclonic, dust devil-like

Future Work

- Final optimization of apparatus
- Turbine drag polar optimization
- Field test summer 2016
- Assessment of technological feasibility
- Connection to natural phenomena (Rankine Vortex?)

Shifting Gears– What is Scientific Software 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:

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

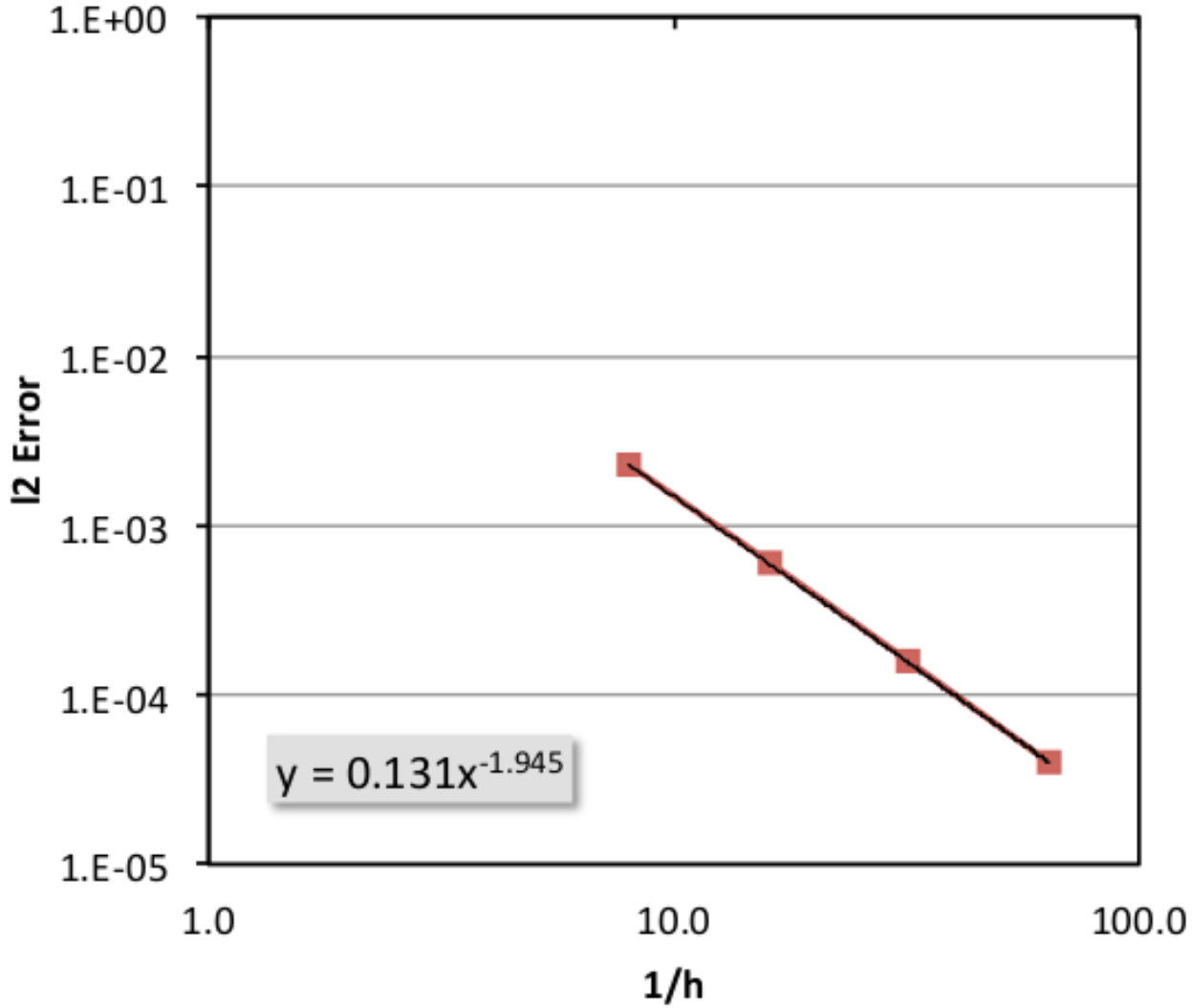
$$\begin{aligned}
=&2(Lx-x)^2 - 8(Lx-x)(Lx+x) + 2(Lx+x)^2 \\
&+2(Ly-y)^2 - 8(Ly-y)(Ly+y) + 2(Ly+y)^2 \\
\neq&0
\end{aligned}$$

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



Useful for Detecting Subtle Bugs

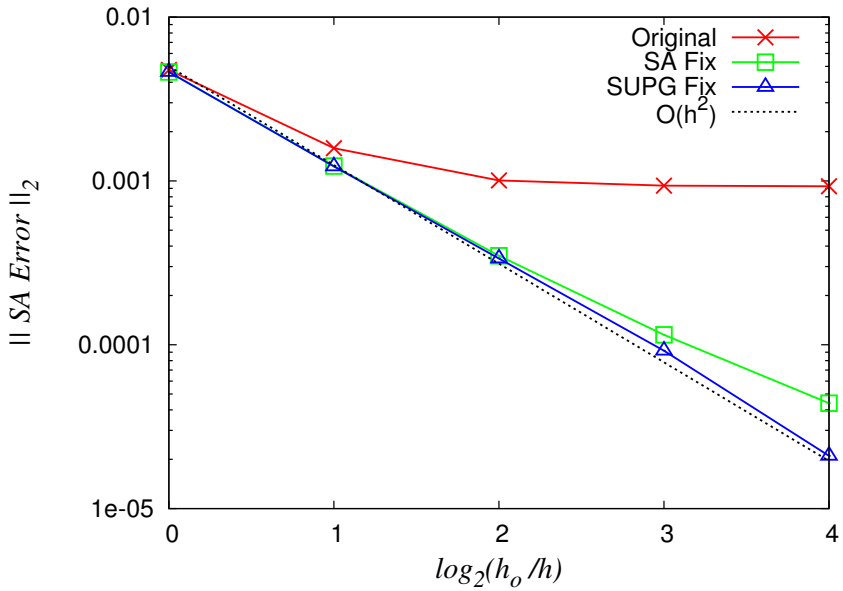
Verification of FIN-S

- FANS, Spalart-Allmaras
- Derivative:

$$\frac{d(sa)}{dx} = \frac{1}{\rho} * \left(\frac{d(\rho * sa)}{dx} - sa \frac{d\rho}{dx} \right)$$

- In code:

$$\frac{d(sa)}{dx} = \frac{1}{\rho} * \frac{d(\rho * sa)}{dx} - sa \frac{d\rho}{dx}$$



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!

$$
\begin{aligned}
&-\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]+
\end{aligned}
$$

$$
\begin{aligned}
&-\frac{2a_{px}a_{px}\pi^2 kp_x\rho_x\cos\left(\frac{a_{px}\pi x}{L}\right)\sin\left(\frac{a_{px}\pi x}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^2}+\\
&-\frac{2a_{py}a_{py}\pi^2 kp_y\rho_y\sin\left(\frac{a_{py}\pi y}{L}\right)\cos\left(\frac{a_{py}\pi y}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^2}+\\
&-\frac{2a_{pz}a_{pz}\pi^2 kp_z\rho_z\cos\left(\frac{a_{pz}\pi z}{L}\right)\sin\left(\frac{a_{pz}\pi z}{L}\right)}{L^2R\left[\rho_0+\rho_x\sin\left(\frac{a_{px}\pi x}{L}\right)+\rho_y\cos\left(\frac{a_{py}\pi y}{L}\right)+\rho_z\sin\left(\frac{a_{pz}\pi z}{L}\right)\right]^2}+\\
&+\frac{4}{3}\frac{a_{ux}a_{vy}\pi^2\mu u_x v_y}{L^2}\cos\left(\frac{a_{ux}\pi x}{L}\right)\cos\left(\frac{a_{vy}\pi y}{L}\right)+\\
&-\frac{4}{3}\frac{a_{ux}a_{wz}\pi^2\mu u_x w_z}{L^2}\cos\left(\frac{a_{ux}\pi x}{L}\right)\sin\left(\frac{a_{wz}\pi z}{L}\right)+\\
&-\frac{2a_{uy}a_{vx}\pi^2\mu u_y v_x}{L^2}\sin\left(\frac{a_{uy}\pi y}{L}\right)\sin\left(\frac{a_{vx}\pi x}{L}\right)+\\
&+\frac{2a_{uz}a_{wx}\pi^2\mu u_z w_x}{L^2}\cos\left(\frac{a_{wx}\pi x}{L}\right)\sin\left(\frac{a_{uz}\pi z}{L}\right)+\\
&-\frac{4}{3}\frac{a_{vy}a_{wz}\pi^2\mu v_y w_z}{L^2}\cos\left(\frac{a_{vy}\pi y}{L}\right)\sin\left(\frac{a_{wz}\pi z}{L}\right)+\\
&-\frac{2a_{vz}a_{wy}\pi^2\mu v_z w_y}{L^2}\cos\left(\frac{a_{vz}\pi z}{L}\right)\cos\left(\frac{a_{wy}\pi y}{L}\right).
\end{aligned}
$$

$$
\begin{aligned}
&-\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_{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]\cdot\\
&\quad\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.\\
&\quad\left.\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_{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.\\
&\quad\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_{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]\cdot\\
&\quad\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_{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]\cdot\\
&\quad\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_{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]\cdot\\
&\quad\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.\right.\\
&\quad\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\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.\\
&\quad\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_{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]}+\\
&+\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_{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]}+\\
&+\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_{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]}+
\end{aligned}
$$

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
==================
```

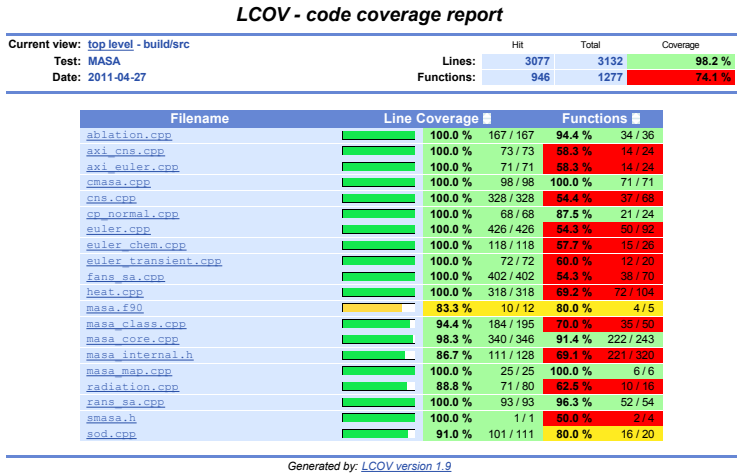
Software Library Snapshot

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



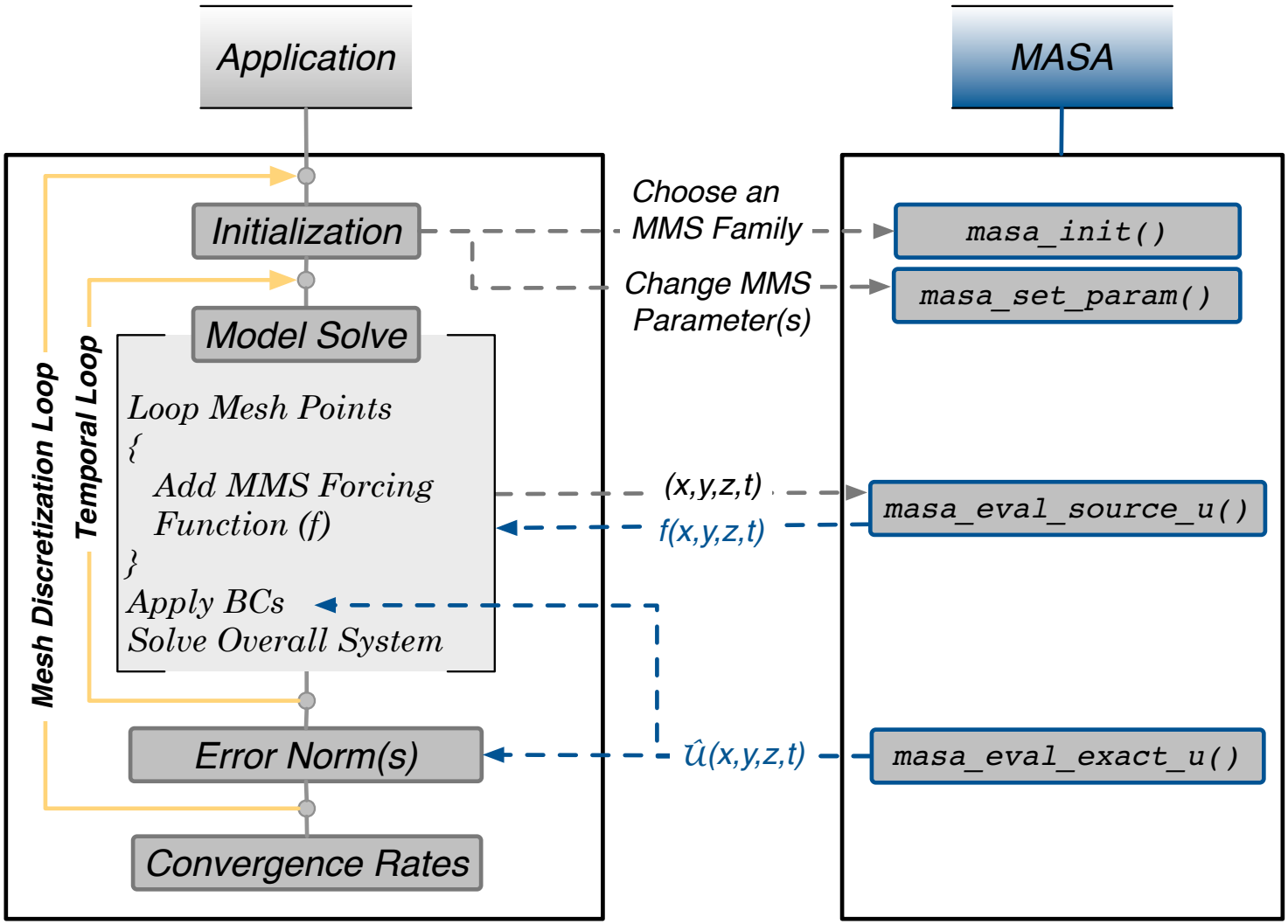
LCOV - code coverage report

			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.

Snapshot

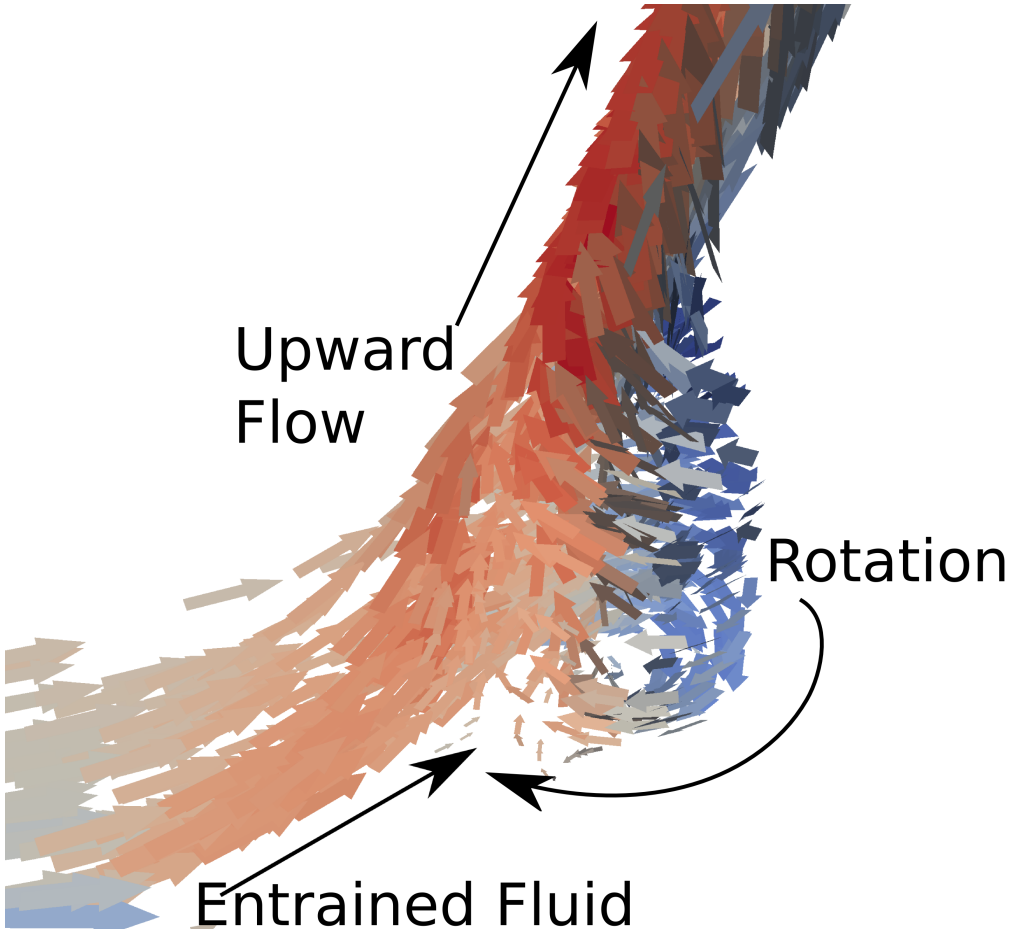
Release

- 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

MASA PDE Examples

Source Terms: Euler

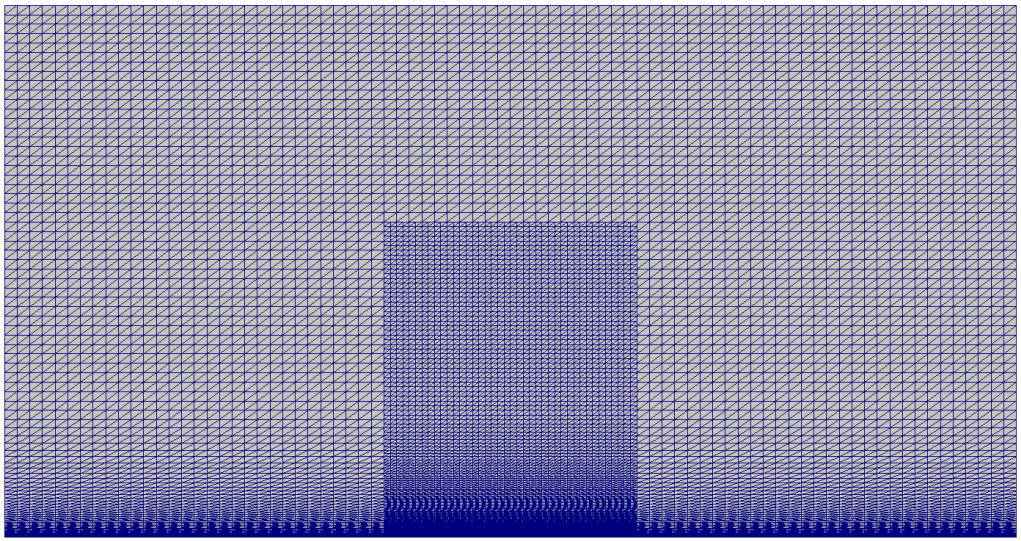
```
// Arbitrary manufactured solutions
U.template get<0>() = u_0 + u_x * sin(a_ux * PI * x / L)
                          + u_y * cos(a_uy * PI * y / L);
```

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

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

```
// 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));
```

Mesh



Discretization

- Single refinement in vane region
- $$\mathrm{Re}_{\mathrm{cell}} = \frac{\max(\Delta x, \Delta y)u}{\nu_T}$$
- Boundary layer mesh visible

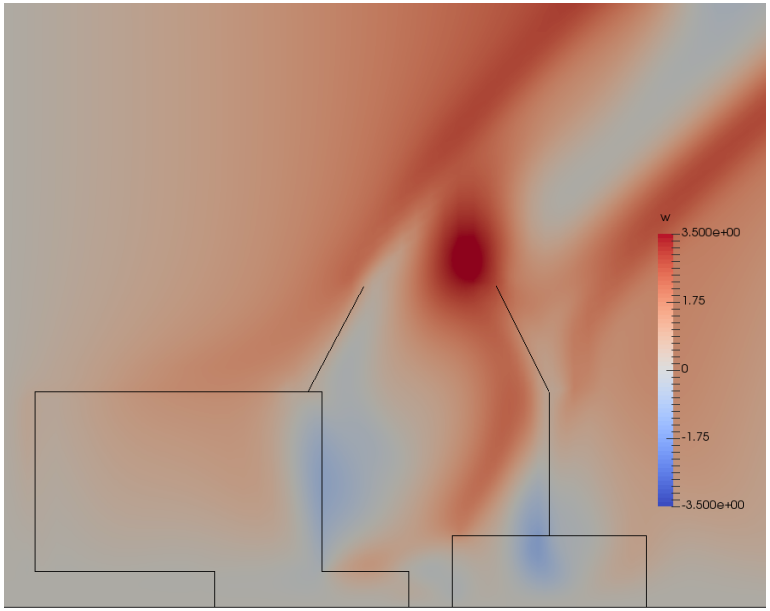
Turbine

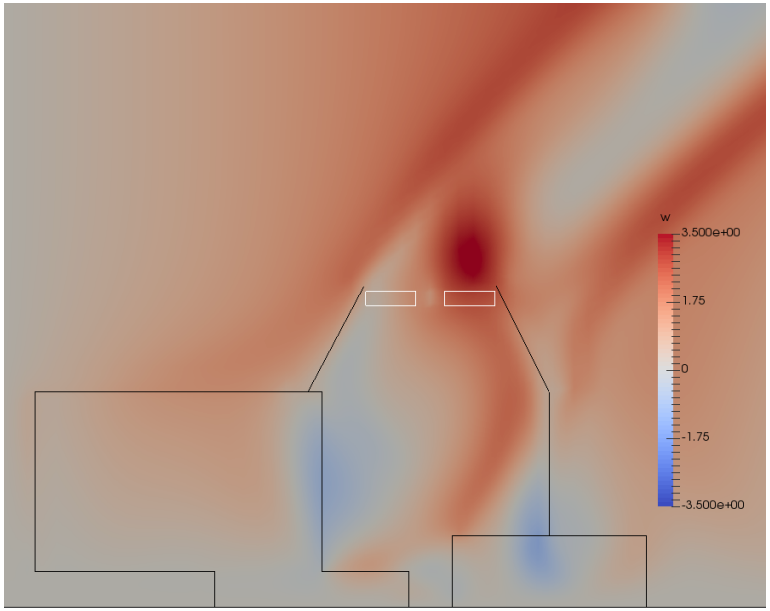
Model

- Actuator-disk
- Implementation similar to virtual vanes
- 90° circular-arc lift and drag polars (C_D, C_L)
- 7 parameters (shown below)

Name	Symbol	Comments
Initial Angle of Attack	α_0	At R_G
Final Angle of Attack	α_f	At L_B
Chord Length	C	Blade thickness
# of blades	A_B	
Center Gap	R_G	Gap in turbine
Base Velocity	ω	fixed tip velocity
Blade Radius	L_B	Length of blade
Height of turbine	H_B	Height of center of turbine

Vertical velocity field with and without turbine





Impact of the turbine (drawn in white)

- Modestly impacting flow, vortex structure largely intact
- Only extracting $\approx 30\%$ of energy
- Optimized drag polars would provide “best case” (see timeline)

Stabilization Outline

- Cast Navier Stokes + Boussinesq equations into weak form
- Prepare as operator $Lc = f$
- Calculate Fréchet derivative
- Separate into differential (P) and constant (Z) components, $L'[c] = P + Z$
- Choose stabilization operator such that $S = -P^*$
- Then stabilization has form, $a_h(c, \phi) = a(c, \phi) + \langle Lc, S\phi \rangle_\tau$