

Introduction to Computational Methods

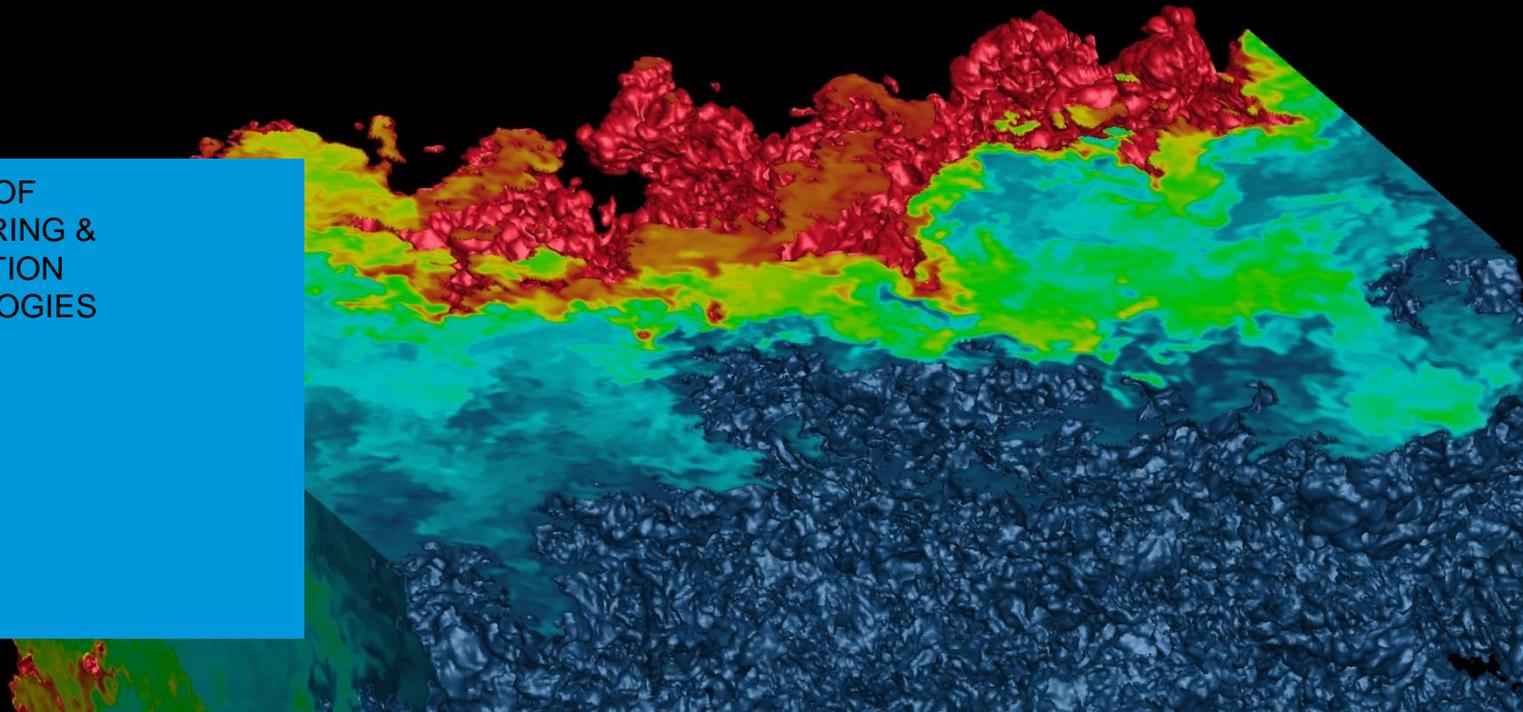
Ben Thornber (ben.thornber@sydney.edu.au)

Acknowledgement : Some slides from Validation and Verification lectures by Dr. Vladimir Titarev

FACULTY OF
ENGINEERING &
INFORMATION
TECHNOLOGIES



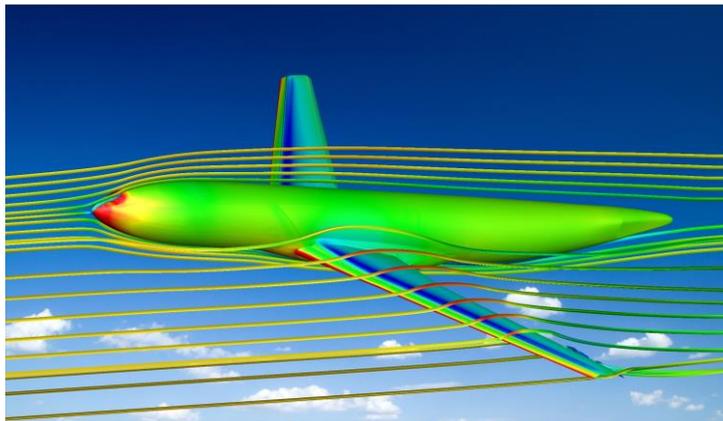
THE UNIVERSITY OF
SYDNEY

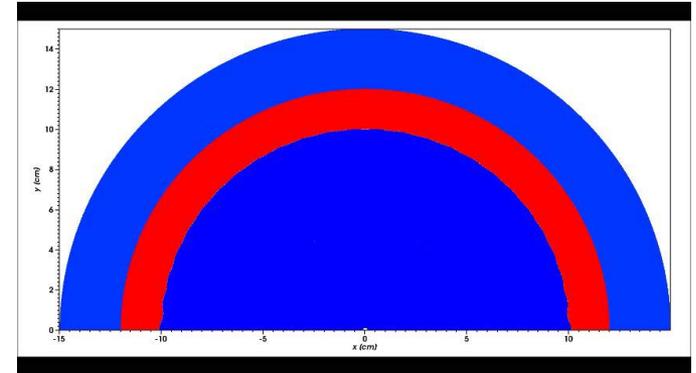
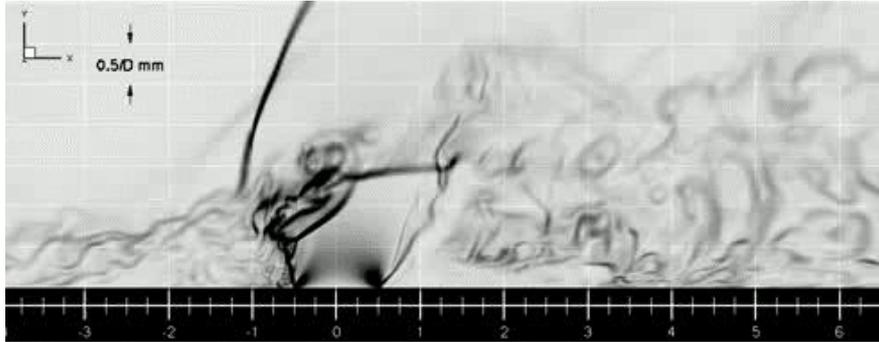


Cover the key elements of ensuring that you can demonstrate a credible computational analysis:

- › **Code Verification**
- › **Calculation Verification**
- › **Validation**

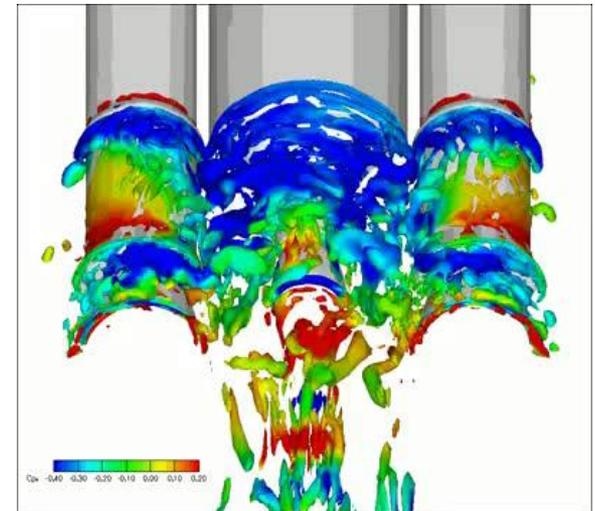
We'll have some examples from research within the compressible flows group.





- **Develop new numerical methods and governing equations**
- **Very high order accurate methods**
- **Steady and Unsteady Turbulence Modelling**
- **Multiple compressible species**

Algorithms developed are used in institutions / CFD codes worldwide



- › Our previous examples require:
 - A mathematical model of the physical system (reduction)
 - A numerical implementation of that mathematical model
 - Verification that the numerical implementation is correct
 - Validation that the mathematical model represents reality
 - Understanding of the computational error
 - › You must be able to demonstrate all of the above for your results to be taken as publishable
-

Code Verification answers the question "are we solving the equations correctly?"

- estimates the magnitude of the error in the computational implementation of the mathematical model.
- compares the numerical methods used in the code to exact analytical results.
- tests for computer programming errors.

Validation answers the questions "are we solving the correct equations?"

- assumes that the numerical solution of the chosen physical model is sufficiently accurate
 - estimates the magnitude of the difference between the results of the
 - computational simulation and physical reality.
 - compares the computed results with experimental results.
-

- › Mathematics or Engineering/Physics?
 - › Mathematics is a tool of science which exists by itself and it is `true' regardless of any correspondence to the natural world.
 - Verification is seen to be essentially and strictly an activity in mathematics, namely the numerical analysis.
 - Validation is essentially and strictly an activity in science and engineering: physics, chemistry, fluid dynamics etc.
 - › Example of a conceptual modelling assumption would be assuming incompressibility for a simulation of flow over a supersonic aircraft
 - If a user incorrectly applies the code the results will be wrong but this does not mean that Verification fails.
 - In this case the lack of agreement with experiment is not a code error, but a poor choice of governing equations thus it fails the Validation.
-

2. Code Verification

You have developed the following mathematical model of a fluid flow:

$$\frac{\partial z_k \rho_k}{\partial t} + \nabla \cdot (z_k \rho_k \mathbf{u}) = \nabla \cdot (\rho D_{12} \nabla Y_k)$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \cdot \mathbf{u}) = -\nabla \cdot \mathbf{S}$$

$$\frac{\partial \rho e}{\partial t} + \nabla \cdot (\rho e \mathbf{u}) = -\nabla \cdot (\mathbf{S} \cdot \mathbf{u} + \mathbf{q} + \mathbf{q}_d)$$

$$\frac{\partial z_k}{\partial t} + \mathbf{u} \cdot \nabla z_k = \nabla \cdot (D_{12} \nabla z_k) - \mathcal{M} D_{12} \nabla z_1 \cdot \nabla z_k + D_{12} \nabla z_k \cdot \frac{\nabla N}{N}$$

You then define an algorithm to solve the equations. How do you verify the code?

We want to ensure that the following errors are either as expected or fully understood:

1. discretization errors
2. programming errors (mistakes)
3. Computer round-off errors

Programming errors (mistakes):

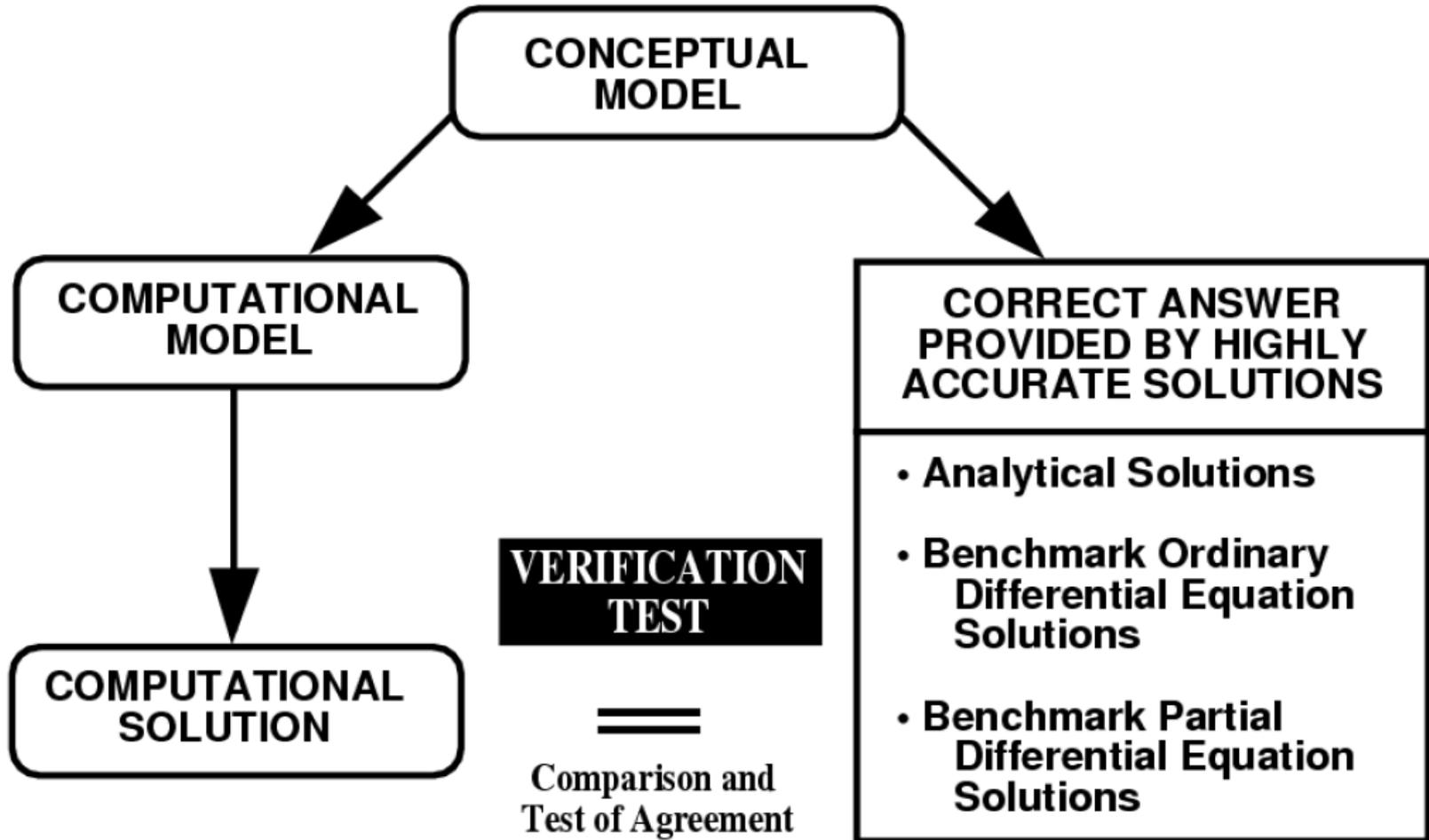
- these can be detected by grid convergence studies for problems with exact solutions.
- For non-analytical problems see method of manufactured solutions.

Computer round-off errors:

- Normally these ones are not a problem
 - However, these are an issue for very high order methods because they can preclude convergence on fine meshes
 - Problems which have small perturbations of large numbers
-



2. Code Verification



Oberkampf 2002a

Define an algorithm to compute the first derivative at second order accuracy:

$$\frac{\partial y}{\partial x} \approx \frac{y(x + \Delta x) - y(x - \Delta x)}{2\Delta x}$$

To verify the implementation, we assume $y = \cos(x)$ and run our algorithm for $x = \frac{\pi}{4}$, choosing $\Delta x = 2^\circ, 4^\circ$ and 8°

Δx (deg)	Δx (rad)	CD	Error	Convergence rate
8	0.139626	-0.706532529	0.0005743	
4	0.069813	-0.706963192	0.0001436	1.999736316
2	0.034907	-0.707070882	3.59E-05	1.999934079

Convergence rate defined as $\log\left(\frac{Error_2}{Error_1}\right) / \log\left(\frac{\Delta x_2}{\Delta x_1}\right)$

The previous slide is fine for a single point estimation

Normally we need an error measure for a solution as an array

Now we use 'norms'

$$L^p = \left(\Delta x \sum_{i=1}^{npts} |f_i - f(x_i)|^p \right)^{1/p}$$

Where f_i is the numerical approximation and $f(x_i)$ is the exact solution.

Here we usually look at $p=1,2$ and ∞ . What do these represent qualitatively?

Once you have the norms, you can look at verify the actual convergence compared to expected

We expect an error proportional to Δx^n , i.e. $L \sim C\Delta x^n$

Run two computations, one with mesh size Δx_A , the other with mesh size Δx_B .

Gain two errors, L_A^1 and L_B^1

We can say:

$$L_A^1 \sim C\Delta x_A^n$$
$$L_B^1 \sim C\Delta x_B^n$$

With rearranging to gain 'n':

$$n = \log\left(\frac{L_A^1}{L_B^1}\right) / \log\left(\frac{\Delta x_A}{\Delta x_B}\right)$$

The measured order of convergence must equal that expected. Otherwise you have an error.

Note that for Implicit schemes you must ensure that the error is not impacted by the approximate matrix inversion which most of these methods employ

How to conduct a grid convergence study:

- Define grids to be used in the study.
 - Typically want at least 3 different grid resolutions where each subsequent grid has 2x the number of points in each direction i/j/k.
 - Commonly use grids with 2^k number of points i.e. $N=32,64,128,256,\dots$
 - Calculate expected order of convergence.
 - Based on order of accuracy of numerical approximations in the algorithm.
 - Also need an analytical solution.
 - For hyperbolic PDEs commonly use problems based on linear advection.
 - Quantify error on each grid using norms. Typically L1, L2, L ∞ are used.
 - Calculate order of convergence to the analytical solution as shown on previous slide.
-

Example:

- Governing equations: 2D Euler equations.

$$\frac{\partial}{\partial t} \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho v \\ (E + p)u \end{pmatrix} + \frac{\partial}{\partial y} \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (E + p)v \end{pmatrix} = \mathbf{0}$$

- Numerical method: 2nd order Godunov-type finite volume method.
- Initial condition: Isentropic vortex.

$$u = 1 - \frac{\varepsilon}{2\pi} e^{\frac{1}{2}(1-r^2)} y, \quad v = 1 + \frac{\varepsilon}{2\pi} e^{\frac{1}{2}(1-r^2)} x,$$

$$T = 1 - \frac{(\gamma - 1)\varepsilon^2}{8\gamma\pi^2} e^{(1-r^2)}, \quad \frac{p}{\rho^\gamma} = 1,$$

- Analytical solution: Initial condition propagates with constant velocity.
-

Example:

N	L1	L2	Linf	O(L1)	O(L2)	O(Linf)
32x32	9.60E-01	4.76E-01	3.75E-01	-	-	-
64x64	5.84E-01	3.29E-001	3.13E-001	0.72	0.54	0.26
128x128	1.46E-01	7.80E-002	7.00E-002	2.00	2.08	2.16
256x256	3.88E-002	2.08E-002	1.63E-02	1.91	1.90	2.10
512x512	9.90E-003	5.28E-003	4.08E-03	1.97	1.98	2.00
1024x1024	2.48E-003	1.32E-003	1.04E-03	1.99	2.00	1.98

- Calculated order of convergence is sufficiently close to the expected order of accuracy.
 - This indicates that our numerical method has been implemented successfully.
-

Some other advice:

- If no analytical solution is available:
 - Verify against results from another previously verified code.
 - Method of manufactured solutions.
 - Unsteady problems may require careful initialisation.
 - e.g. finite volume solvers require cell averaged values of the initial condition.
 - Try to avoid analytical solutions with discontinuities.
 - Can affect order of convergence.
 - Slope/flux limiters used to prevent oscillatory behaviour in high-order methods can also affect convergence.
-

3. Verification of the Calculation

It is distinctly different from Verification of the Code.

- › the use of a Verified Code is not enough.
- › A code may be rigorously verified to be (say) 2nd order accurate, but when applied to a new problem, this fact provides **no estimate** of accuracy or confidence interval.
- › It is still necessary to band the numerical order for the individual calculation

In other words, you want an error estimation for your given calculation

3. Richardson Extrapolation

- › We assume the discrete solution f to have a series representation in grid spacing h of

$$f(h) = f_{exact} + g_1 h + g_2 h^2 + g_3 h^3 + \dots$$

- › The functions g_i are defined in the continuum and do not depend on any discretization.

e.g. For a 2nd-order method we have $g_1 = 0$

- › The key idea is to combine two separate discrete solutions from two uniform grids: $f(h_1)$ (fine grid) and $f(h_2)$ (coarse grid), to eliminate the leading order error terms in the assumed error expansion, i.e.
 - solve for g_2 at the grid points
 - substitute this in to the equation for f
 - finally obtain a more accurate estimate of f_{exact} .
-

3. Richardson Extrapolation

- › The result is given in the original 1927 paper for h^2 expansion:

$$f_{exact} = f_1 + \frac{f_1 - f_2}{r^2 - 1}$$

Where $r = h_2/h_1$.

- › The most common use of the method is with grid doubling or halving (this is essentially the same). With $r = 2$ we get

$$f_{exact} \approx \frac{4}{3}f_1 - \frac{1}{3}f_2$$

- › This is 3rd-order accurate in the general case and becomes 4th order accurate if $g_3 = 0$, as is the case with e.g. the use of centred differences
 - › The concept of the ‘Grid Convergence Index’ generalizes this to non-integer refinement levels and is the ASME default for reporting grid convergence studies.
 - › In FEA h (space) and p (order) refinement studies are both employed.
-

3. Example in FEA

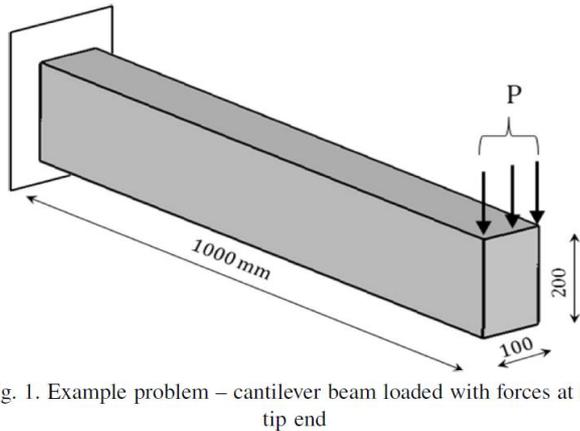


Fig. 1. Example problem – cantilever beam loaded with forces at the tip end

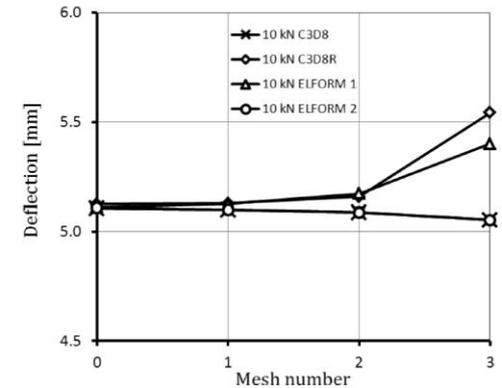
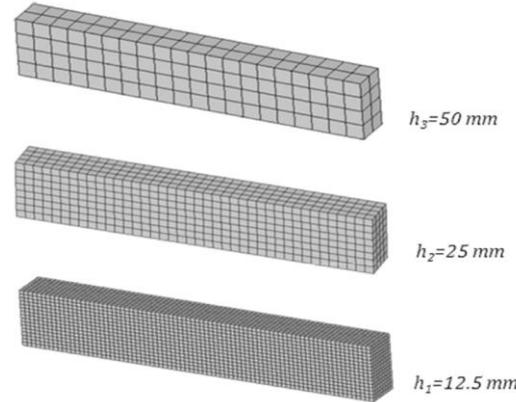


Fig. 3. Results for load $P = 10$ kN

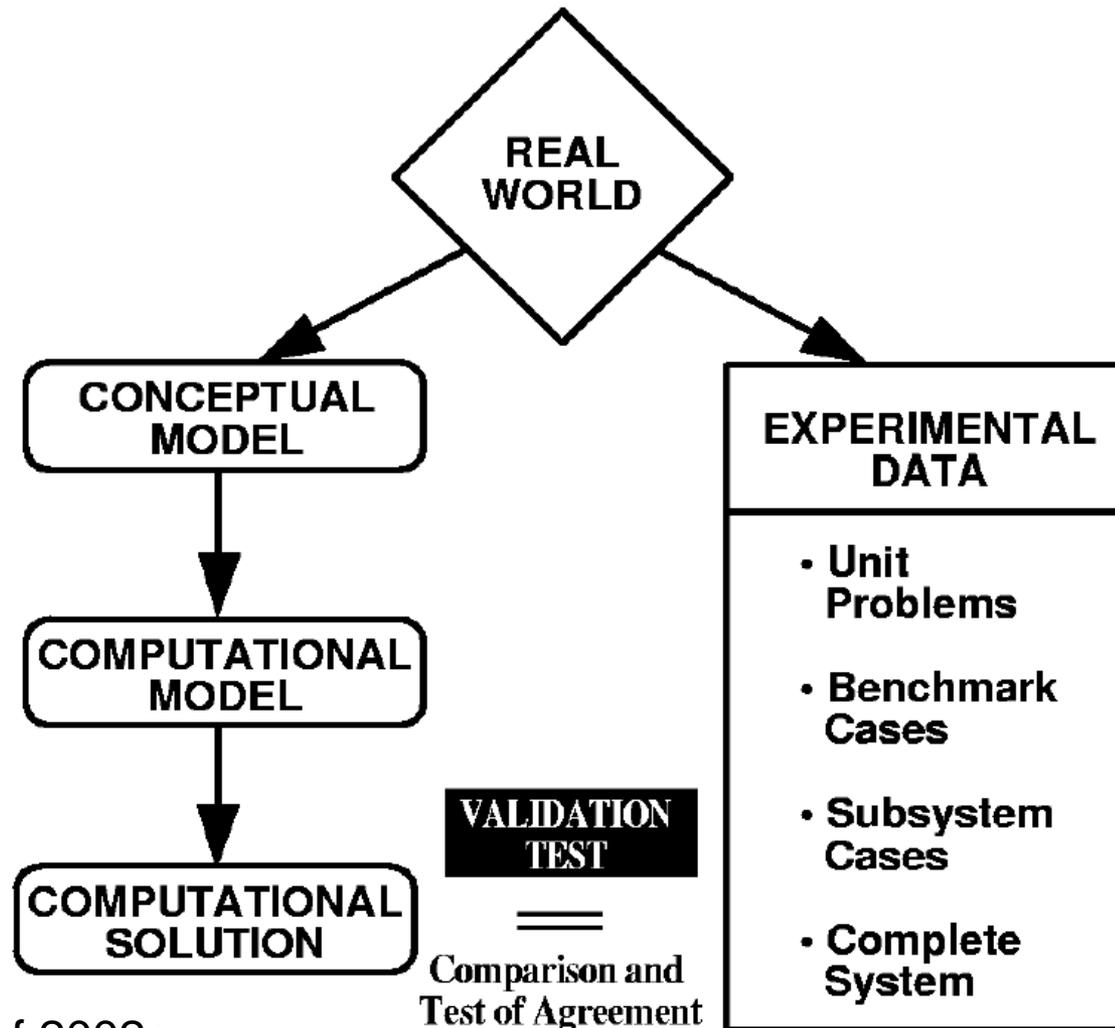
Table 2
Calculation of grid convergence index

Load [kN]	FE element formulation	Order of convergence p (5)	Asymptotic solution $f_{h=0}$ (10)	GCI_{12} (15) [%]	GCI_{23} (15) [%]	$GCI_{23}/r^p GCI_{12}$ (16)
10	C3D8	1.431	5.107	0.177	0.478	1.002
	C3D8R	3.656	5.125	0.064	0.803	0.994
	ELFORM 1	2.242	5.111	0.315	1.477	0.991
	ELFORM 2	1.479	5.107	0.167	0.467	1.002

Validation is the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model.

The fundamental strategy of Validation involves

- quantification of the numerical error in the computational solution,
 - estimation of the experimental uncertainty,
 - comparison between the computational results and experimental data.
 - We do not assume that the experimental measurements are more accurate than the computational results.
 - the estimation process for error and uncertainty must occur on both sides of the coin: mathematical physics and experiment.
-



The building-block approach:

- divide the complex engineering system of interest into a three or more progressively simpler tiers;
- these are: subsystem cases, benchmark cases, and unit problems.
- Results are compared at multiple degrees of physics coupling and geometric complexity.

The approach is clearly constructive in that it recognizes that

- there is a hierarchy of complexity in systems and simulations
 - the quantity and accuracy of information that is obtained from experiments varies radically over the range of tiers.
-

Validation hierarchy construction should:

- Carefully disassemble the complete system
- Identify experiments that are attainable and practical
- Identify experiments where validation quality characterization and measurement data can be obtained
- The top of the hierarchy focuses on the application of interest
- The bottom of the hierarchy focuses on separate-effects physics

A good hierarchical tier construction should accomplish two tasks:

- to carefully disassemble the complete system into tiers so that each lower-level tier has one less level of physical complexity
 - selection of individual validation experiments in a tier that are practically attainable and able to produce validation-quality data
-

4. Validation hierarchy example

A complex, multidisciplinary system: an air-launched, air-breathing, hypersonic cruise missile.

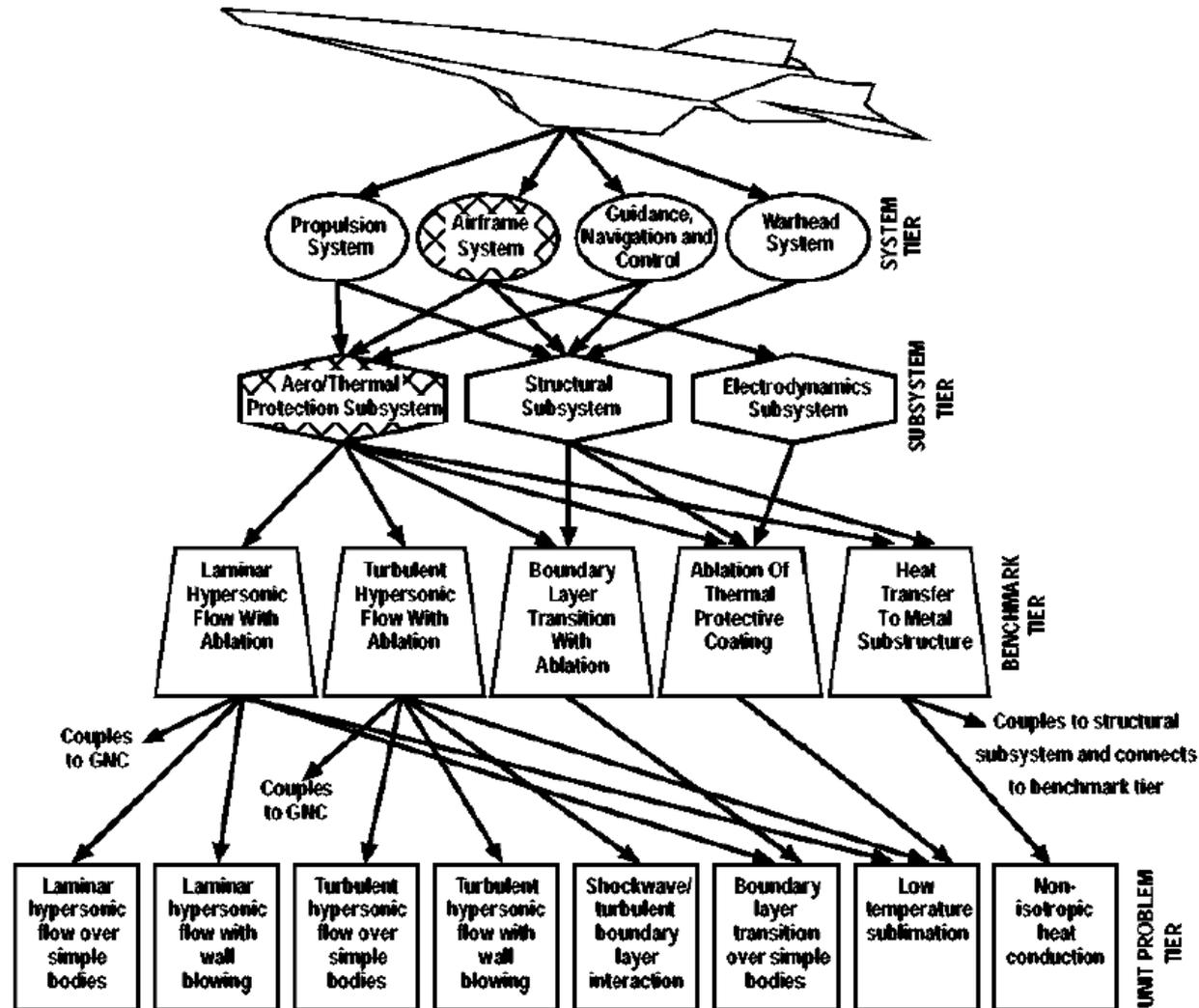
We assume that the missile has an autonomous guidance, navigation and control (GNC) system, an on-board optical target seeker, and a warhead.

We refer to the missile as a complete system and to the following as systems: propulsion, airframe, GNC and warhead.

The launch aircraft is not included because his location would be at the next higher level, i.e. above cruise missile.

We note that the structure shown is not unique.

4. Validation hierarchy example



4. Validation experiments at each Tier level

The aero/thermal subsystem simulation. It would contain

- the actual thermal protective coating over the metal skin of the missile,
- the actual metallic skin of the vehicle,
- much of the substructure under the skin of the vehicle,
- all of the functional/lifting surfaces,
- and the internal flow path of the propulsion system.

It would not contain any other hardware inside the vehicle unless some particular heat conduction is crucial

The validation experiment for the structural dynamics code would contain every piece of the hardware from the missile because very part of the structure is mechanically coupled

It would not contain

- warhead and functional propulsion system
 - mass-mockups may be used instead
-

4. Validation hierarchy example

A complex, multidisciplinary system: an air-launched, air-breathing, hypersonic cruise missile.

We assume that the missile has an autonomous guidance, navigation and control (GNC) system, an on-board optical target seeker, and a warhead.

We refer to the missile as a complete system and to the following as systems: propulsion, airframe, GNC and warhead.

The launch aircraft is not included because his location would be at the next higher level, i.e. above cruise missile.

We note that the structure shown is not unique.

5. General Advice

Before undertaking computations using techniques outside the direct scope of your PhD discuss it with an expert within that domain. Clarify clearly if what you are undertaking is 'difficult'

Start with a small computation to check that your results are approximately correct before using half a supercomputer. Also try to avoid using a supercomputer if you can reduce your problem by e.g. symmetry

For all computations the first step should be verification, unless you can be certain it has already been done.

Steady state problems:

- Check choice of physical model/governing equations
 - Check the required levels of convergence if the solution is implicit and utilises approximate matrix inversions
 - Ensure results are grid converged and quantify error (Richardson/GCI)
 - Validate against appropriate experiments or, if available, analytical solutions.
 - Be critically aware of the experiments/real world. Was it actually steady, or are you comparing to time-averaged data? What is the modelling which allowed you to represent it as steady state? What are the key assumptions underlying that model?
 - Extract and plot experimental bounds along with your numerical results.
-

Unsteady problems:

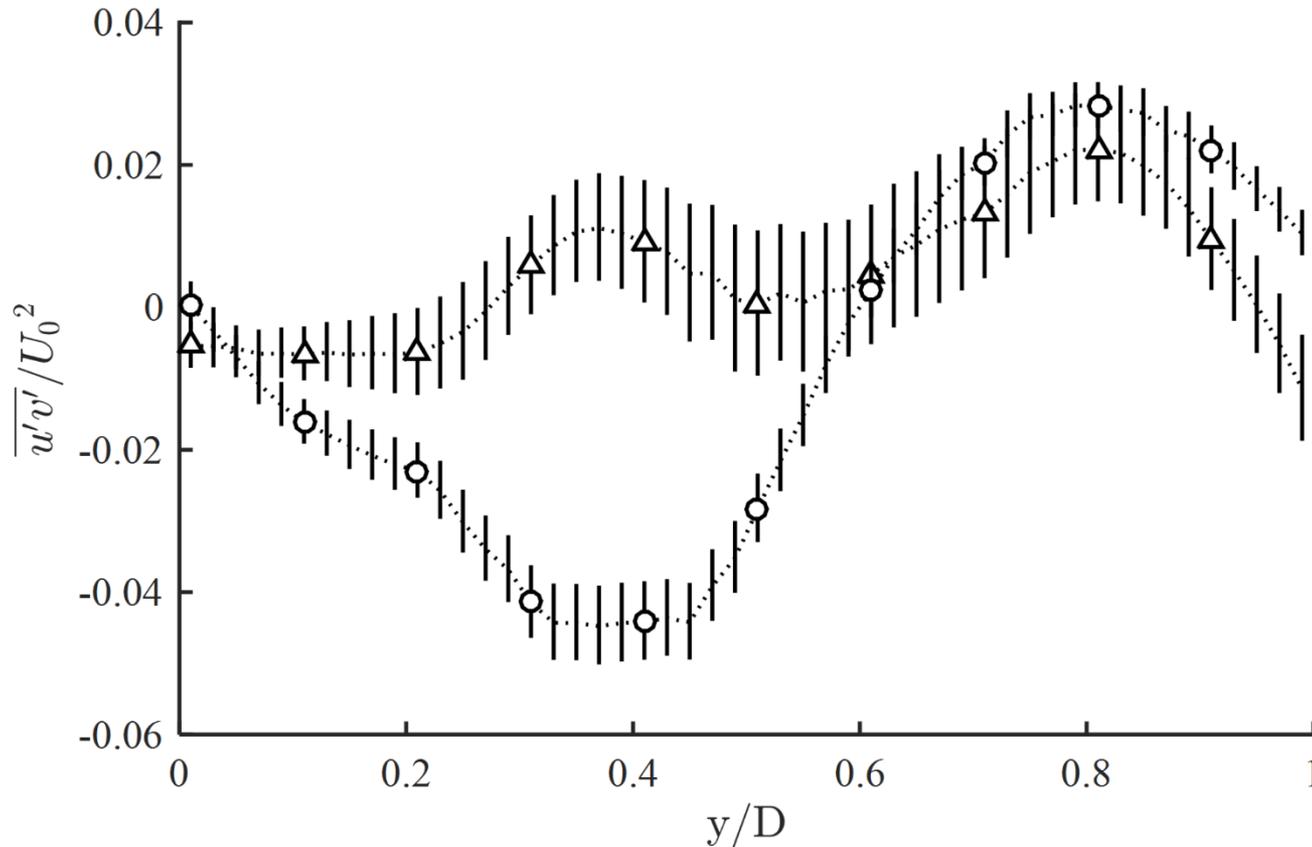
- Check choice of physical model/governing equations
 - Check the required levels of convergence within each time iteration if the solution is implicit and utilises approximate matrix inversions
 - Check the expected sampling time required to gain statistically converged solutions
 - Determine a sampling rate which will capture the expected physical fluctuations
 - Your computation will gain one realisation of physical reality. It is important to understand whether that is sufficient to compare with experiment. It is not unusual for cases to require tens or hundreds of independent realisations to gain an accurate average.
 - Be critically aware of the approximations in the experiments/real world, particularly boundary conditions
 - Extract and plot experimental bounds along with your numerical results with numerical errors clearly highlighted.
-

Daniel Linton: Representing Uncertainty in CFD

- There is a tendency to overlook uncertainty when presenting computational results
 - Unlike experimental work there is no uncertainty associated with instrumentation, but the uncertainty due to incomplete convergence can and should be quantified
 - Running an unsteady simulation until complete statistical convergence can be infeasible - particularly when attempting to resolve turbulence
 - Confidence intervals are one way of representing the uncertainty in some parameter of a distribution
 - See, for example, Kreyszig's Advanced Engineering Mathematics for instructions and conditions associated with calculating confidence intervals
 - As an example, $\text{CONF}_{0.99}\{(\bar{U} - \Delta U) < \bar{U} < (\bar{U} + \Delta U)\}$ indicates a probability of 99% that the true mean velocity is within $\pm\Delta U$ of the mean of the sample set
-

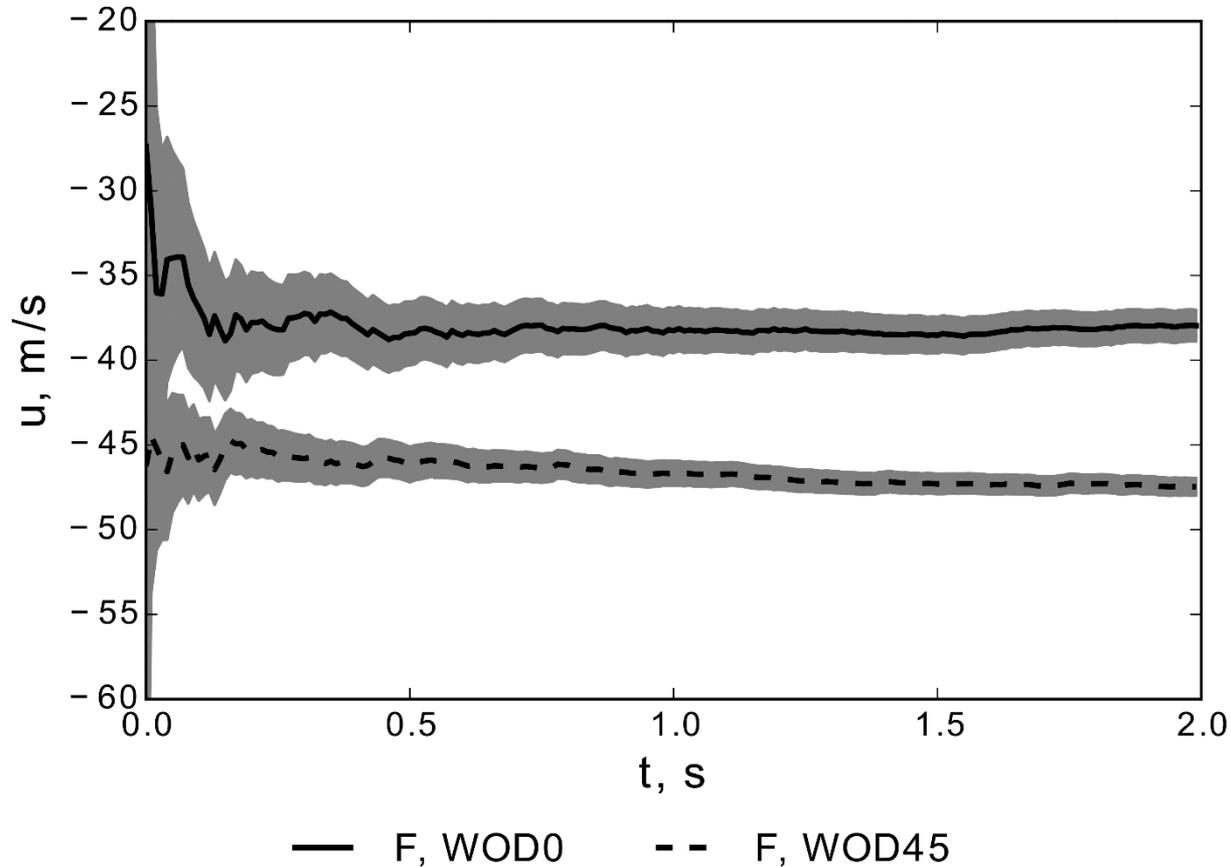


Representing Uncertainty in CFD



Mean of Reynolds stress component showing 99% confidence interval

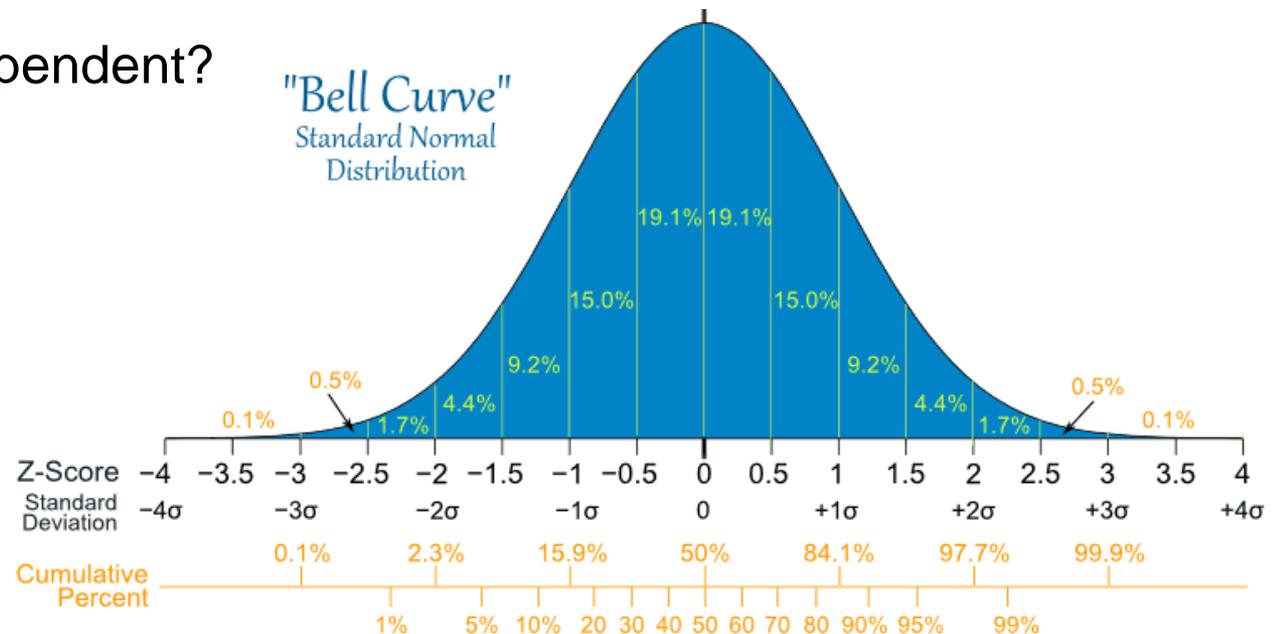
Representing Uncertainty in CFD

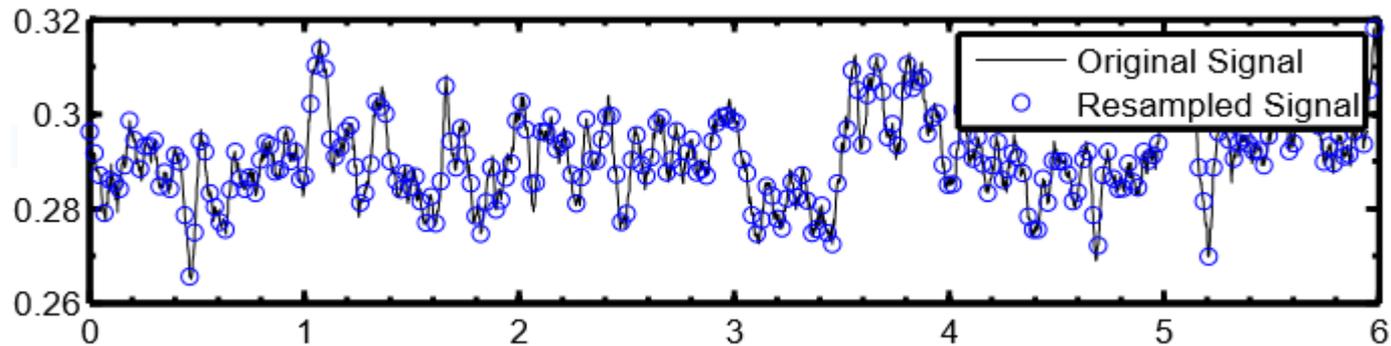


Convergence of mean velocity at sample point showing 95% confidence interval

Quantification of uncertainty in correlated data – Asiful

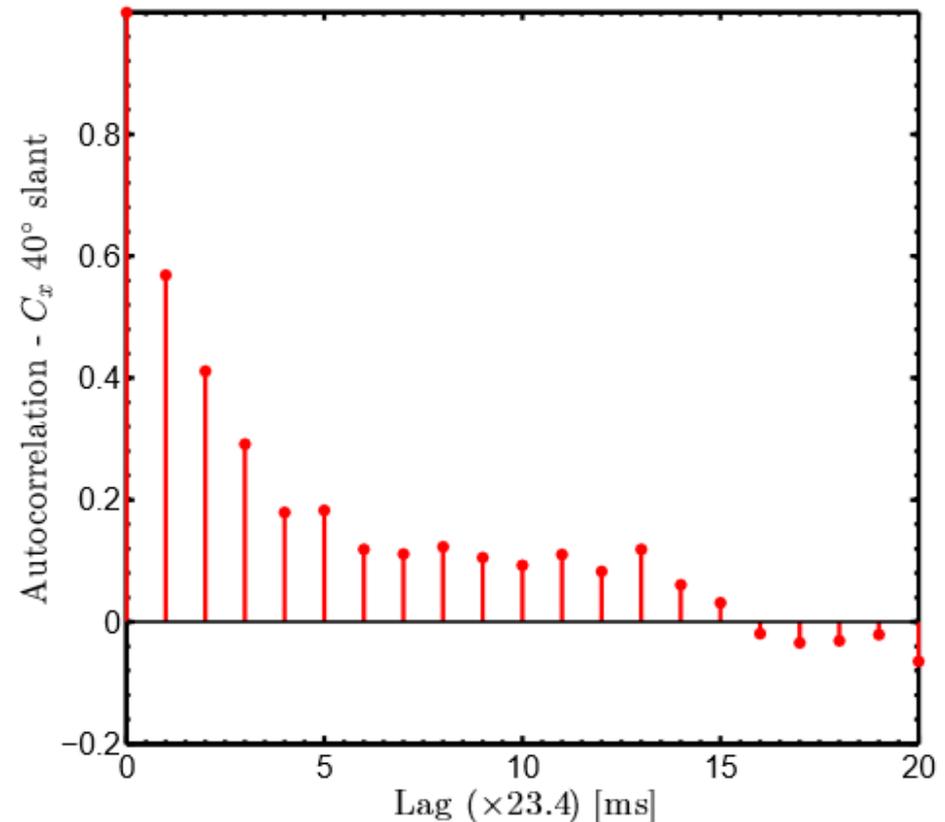
- › Need computational models to predict flows – mean and unsteady
- › What is our level of uncertainty?
- › Is it enough to invoke Standard Normal Distribution?
 - Central Limit Theorem
- › Is the data independent?



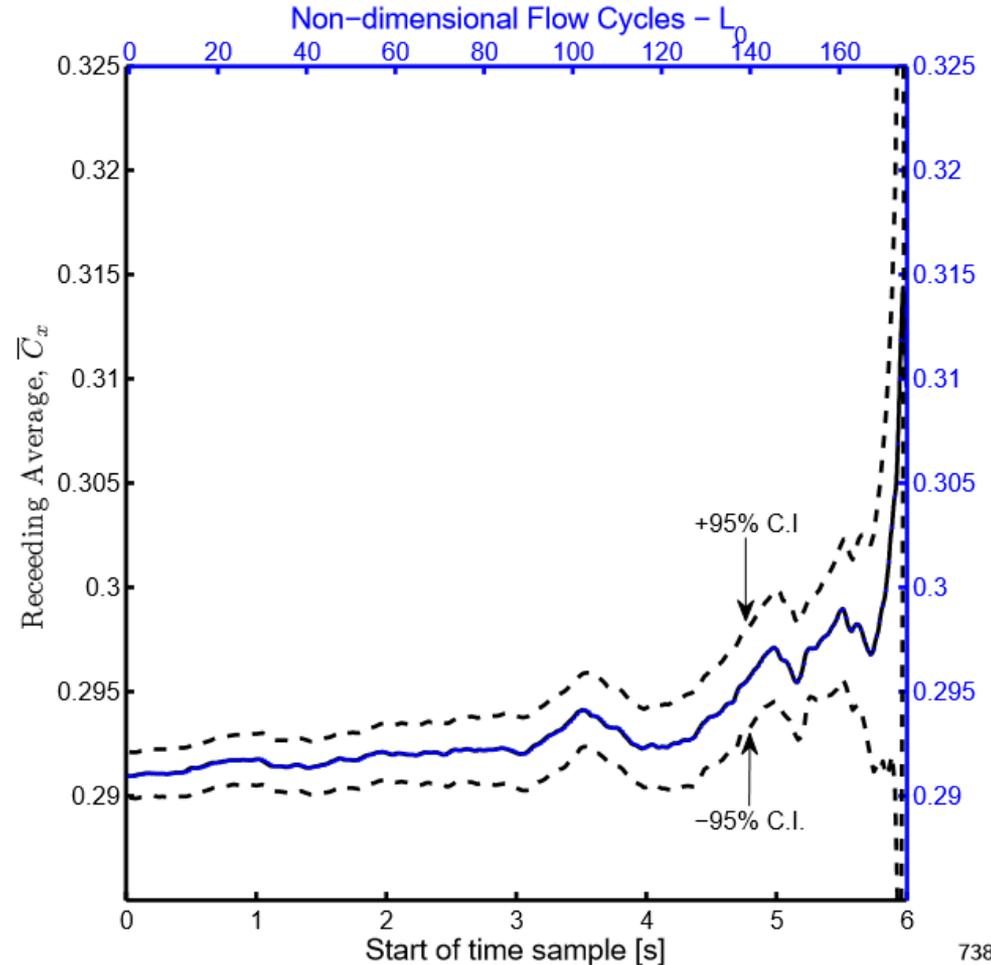


$$\bar{\phi} = \frac{1}{N} \sum_{i=1}^N \phi_i$$
$$S_N = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (\phi_i - \bar{\phi})^2}$$
$$A = t_{(0.05, n-1)}$$
$$\bar{\phi} \pm A \frac{S_N}{\sqrt{N}}$$

- › What are the integral length and time-scales present?
- › Autocorrelation – similarity of observations relative to time-lag between them.



- › How long does a simulation take to ‘converge’?
- › What is the minimum clock-time we need to reach acceptable mean?
- › How does this compare with confidence bounds assuming independent data?



- › Given you a brief introduction to some of the challenges computational methods
 - › Hopefully these will give you a sufficient base to use computation tools competently and avoid falling into the 'crap in, crap out' trap:
 1. Don't just run a simulation then believe it immediately. Follow the Verification and Validation procedure outlined. Compare against experiment, previous simulations (published) or theory - even at a coarse grid resolution.
 2. Don't start with the largest grid you can afford. This will only end up with you running a simulation for a week then finding out that it is rubbish. Start very small to check boundary conditions, then work up from there. Start simple and ideally in 1D.
 3. Be very self-critical. 1st order solutions are not accurate. Unconverged solutions are not accurate. Many physical models are gross simplifications
 4. Accepting unexplained results always ends in disaster. Questioning them either ends in improved understanding where it was missing, or a publication!
 5. Look at the literature - see what is already out there. This can save you weeks of work.
-

1. W.L. Oberkampf and T.G. Trucano. Verification and validation in computational fluid dynamics. In Sandia report SAND2002-0529, page 124, 2002.
 2. P. J. Roache. Quantification of uncertainty in computational fluid dynamics. Annu. Rev. Fluid. Mech. 1997. N. 29 pp. 123-160.
 3. P.J. Roache. Verification and validation in computational science and engineering. 1998.
-