The Shape of CFDs to Come by Brian Spalding The Past

OpenFoam OCTOBER 2016

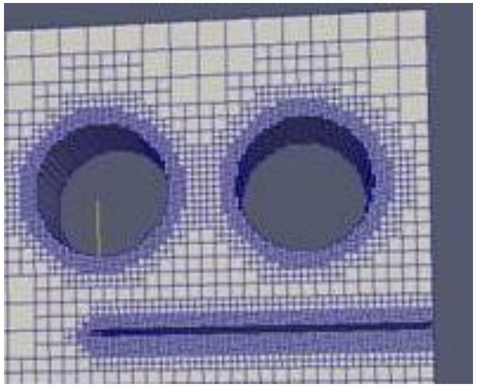
The Shape of CFDs to Come

Early 1970s: open-source was the rule.

Patankar's 2D parabolic-flow code was published in a book.
Runchal's & Wolfshtein's 2D elliptic-flow code likewise.
Subsequent 3D parabolic and elliptic Imperial College codes were widely loaned or otherwise distributed.

Mid and late 1970s: semi- then fully-closed source.
Industry recognised CFD's promise, sought assistance.
CHAM first provided client-specific codes; then, for QA and lower cost, variants of one General-Purpose code.
Hence, 1981, **part-open-**source PHOENICS; whereafter closed-source Fluent, Star-CD, Fire, FIDAP, *etc*., *etc.*.

Up to the Present



What's good:

CFD is now widely used.

New techniques have been developed, *e.g.* **unstructured grids.**

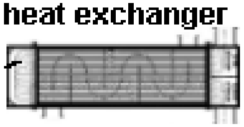
Others are vanishing, *e.g.* **finite-elements for fluids.**

Open-source practices have been revived (Open-Foam).

What's bad:

Costly 'brute-force' (**one-grid-for-all**) methods prevail.

The most-widely-used codes have the fewest advanced features.

Much of 

industry still uses **pre-CFD** methods.

What CFD can & **cannot** do is **not widely understood**.

The Future:
Will general-purpose codes survive?

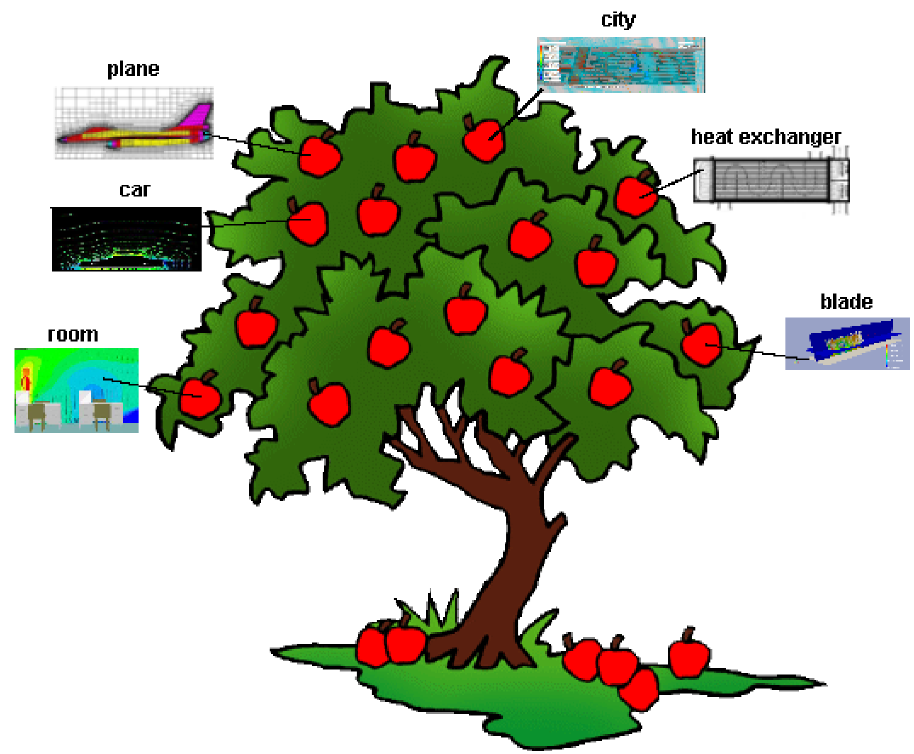
The Shape of CFDs to Come

Yes, but mainly out of sight.
Instead, **CFD-apps** will occupy attention.

CFD-apps apply CFD to **classes** of equipment, *i.e.* Simulation Scenarios, *via* **application-specific menus**.

App **users** need know even less about CFD than apple-eaters about arboriculture.

Apps and apples can be equally healthy if the tree-roots are well nourished…. by the underlying CFD code.



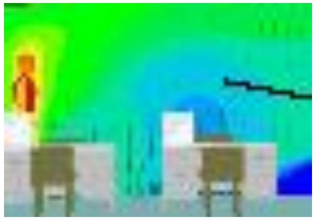
What is a CFD-app?
And why will they prevail?

OpenFoam OCTOBER 2016

The Shape of CFDs to Come

1. Why? General-purpose CFD codes simulate **many classes** of scenario; users need **just one**.

To **particularize** a general-purpose code requires **specialist skills** which users **can ill afford** to learn.



2. What? A CFD-app is a **one-scenario-class** user-interface. Its **creators** provide **the particularization.**

CFD-apps ask only for inputs that users know about in **application-specific** language *e.g.* 'air-change/hour'.

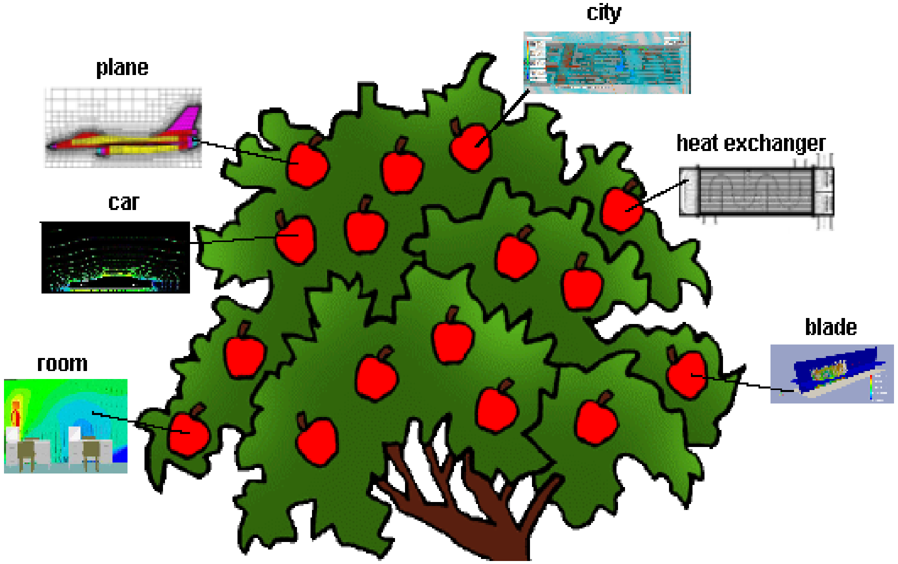
CFD-apps create grids **without user intervention**; and **set numerical parameters** likewise.

CFD-apps supply **results-displaying macros**; and **automatically** write results-interpreting **reports**.

The three CFD-app communities:
1. Users 2. Creators
3. CFD-service providers

1. **Users**: choose the app; state their requirements; receive the results; enjoy the benefits; pay the money; call the tune.

2. **App creators:** are the major innovators; they speak two languages: users' and CFD's; and receive most of the money.



3. **CFD-service providers** (ANSYS, ESI, CHAM, *etc):* supply simulation features which creators call for; crunch numbers at minimum cost; are paid in proportion to use.

What's the difference from today?
1. From the user's view-point

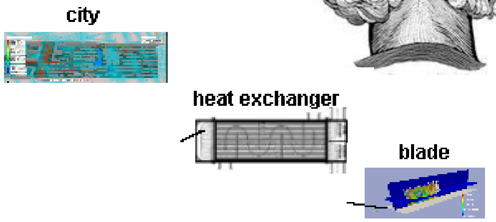
Now: Users choose **one service provider;**
pay significant money, get **more than they need**, which they may suppose to be **the whole** of CFD; but which is very often **less** than they need.

They must **themselves** create the **grids;** and make other **numerical settings**, optimal or not; run the code; display and interpret the results.

Future: Users choose **the currently-needed app;**
pay less; and for **no more** than they need;
do have access to the **whole of CFD**;
rely on settings **made by the app-creator** for **that** application; but may use **different creators** for for other applications.

What's the difference?
2. From the app-creator's view-point

1. His important middle-man role (*ie* facing both ways) is acknowledged and rewarded.



2. Of the two, his special-application knowhow is the more important.

3. But he must also know **which** general CFD code has the **special features** that his particular app needs, *eg:*

- Parabolic-solution.
- Data input *via* formulae,
- PARSOL (*i.e.* objects embedded in structured grids),
- sub-divided Cartesian grids,
- simultaneous solid-stress-solving option, *etc.*

No service provider's code possesses them all.

What's the difference?
3. From the CFD-service-provider's view-point

1. What a relief! To **concentrate** on what he knows best, no longer trying to be all things to all men;

2. And a pleasure, to know that users of **any** apps may use **his** services …. so long as they are **easily accessed; and competitively priced.**

3. To encourage use he documents his **code-feature-activation protocols** and publicises them **widely.**

4. And of course makes his software available *via* the 'cloud' on a **pay-by-use** basis.

PART 2. Examples of apps:
1. The liquid-ring pump

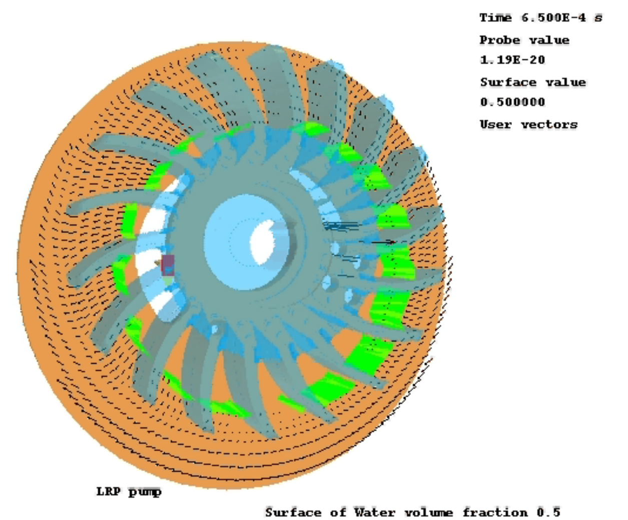
OpenFoam OCTOBER 2016

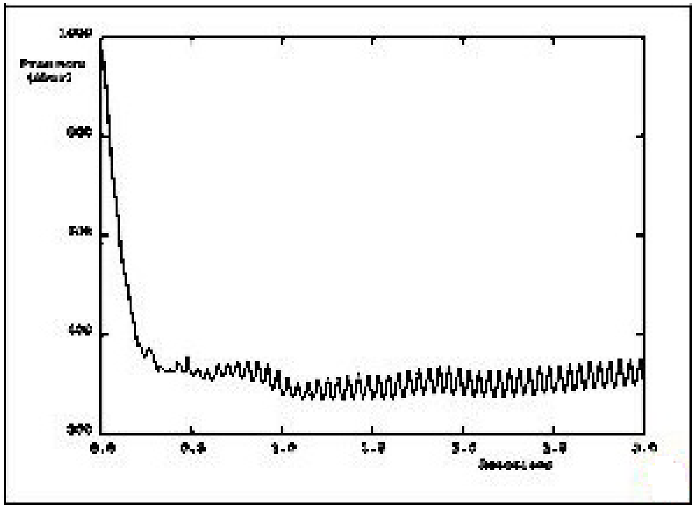
The Shape of CFDs to Come

The liquid-ring pump is used for extracting air from power-station condensers.

Its flow is:

- Transient,
- 2-phase free-surface,
- Caused by eccentric rotor blades
- which pass inlet and outlet ports in an end plate.

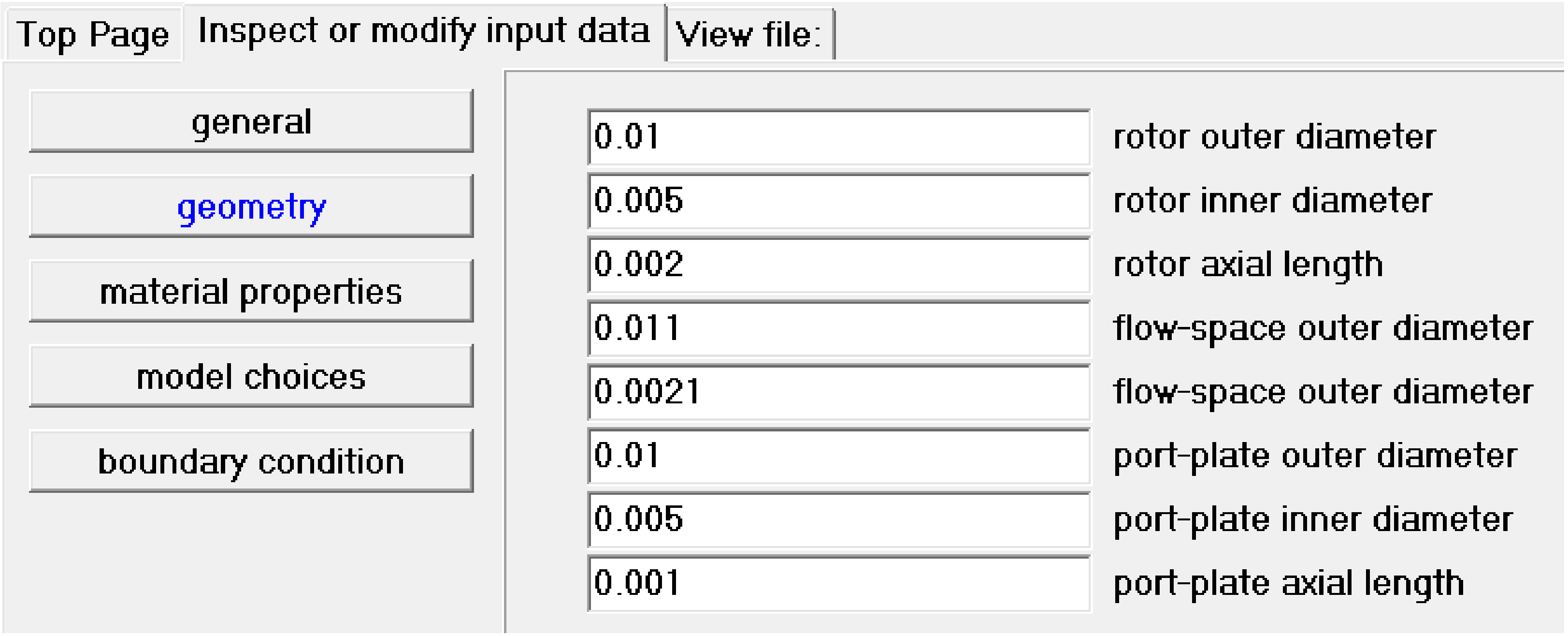




From the CFD view-point the computation results in a cyclically steady-state solution.

Examples of apps:
1. The liquid-ring pump, slide 2

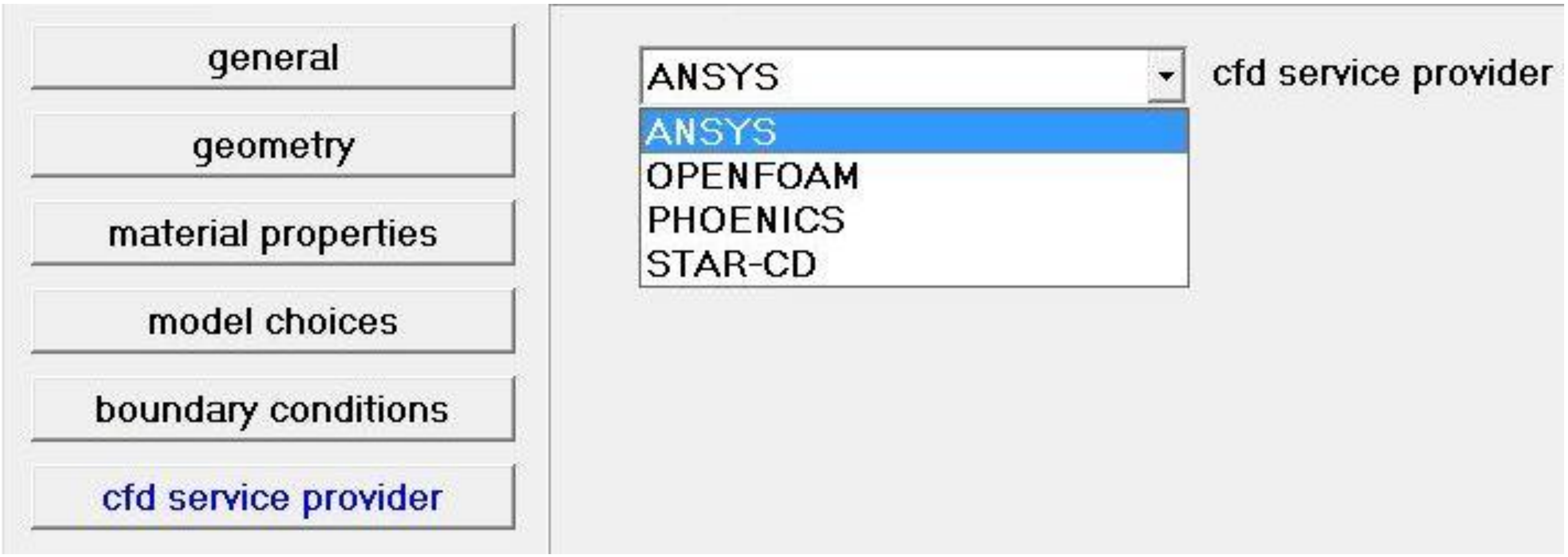
No liquid-ring-pump designer could set up the CFD computation for himself; but he **could** use an app menu, of the straight-forward style shown here:



These are the defaults. He can replace any white-box content with numbers, or with **algebraic expressions**.

Examples of apps:
1. The liquid-ring pump; slide 3

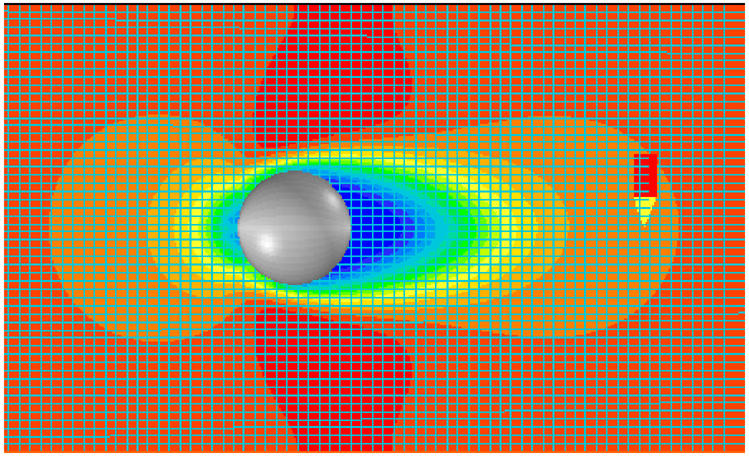
In the future, users will be able to compare the outputs of different CFD-service-providers *via* menus like this:

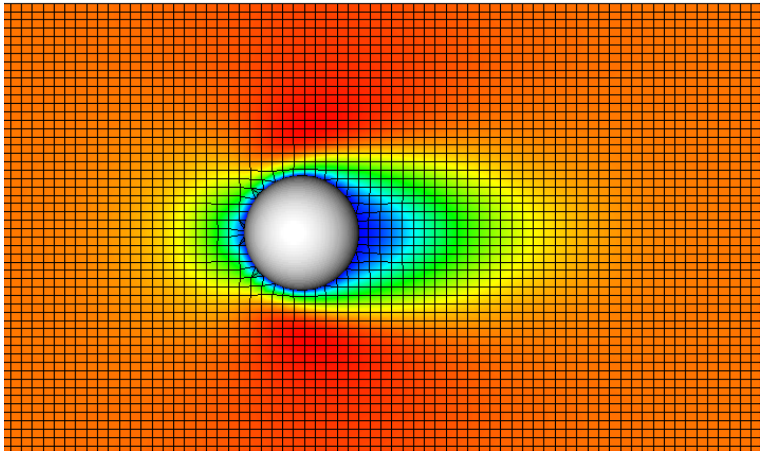


The app-creator formulates the menu; and designs the app to convert inputs into CFD-code-acceptable form; but o**nly** for codes which **can** accept it.

Examples of apps:
2. Virtual Wind Tunnel

Here is such a comparison between PHOENICS (left) and OpenFoam (right) for a Virtual Wind Tunnel app.





Diagrams above are of absolute-velocity contours for identical grids, boundary conditions & 20 outer iterations

PHOENICS used PARSOL; OpenFoam SnappyHex.
Stagnation-point pressures were 0.55 and 0.60 .
Study of the printed results allows detailed comparison.

What it takes to create an app: Input-, solver- & human-side needs

Input side:

1. A **parameterized** (or –**izable)** input language.
PHOENICS has PIL. Do all codes have equivalents?

2. An **automatic**-menu-making editor, in order to: save labour, avoid mistakes and ensure uniformity.

Solver side:

1. Acceptance of formulae as **character strings** interpreted as how-to-compute instructions (**In-Form**).

2. Ability to embed objects with facet-vertex-defined surfaces in Cartesian, polar or BFC grids (**PARSOL**).

Human side:

The intention to serve the app-users interests even when it entails explaining: No, **CFD can't yet do that**.

PART 3. Other CFDs for the future. Cherished Fallacies Demolished: Cherished Fallacy No. 1

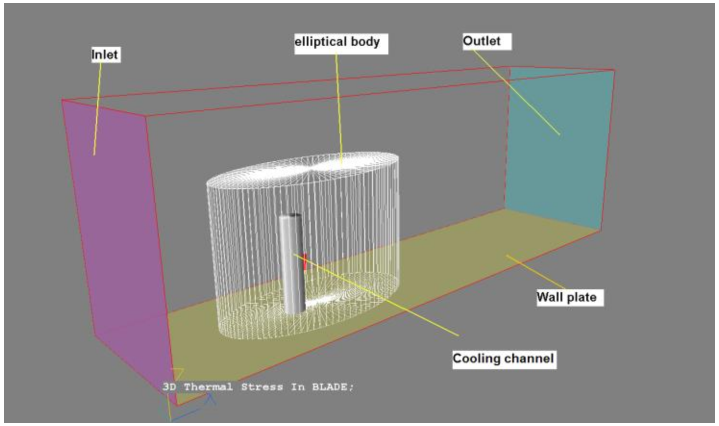
OpenFoam OCTOBER 2016

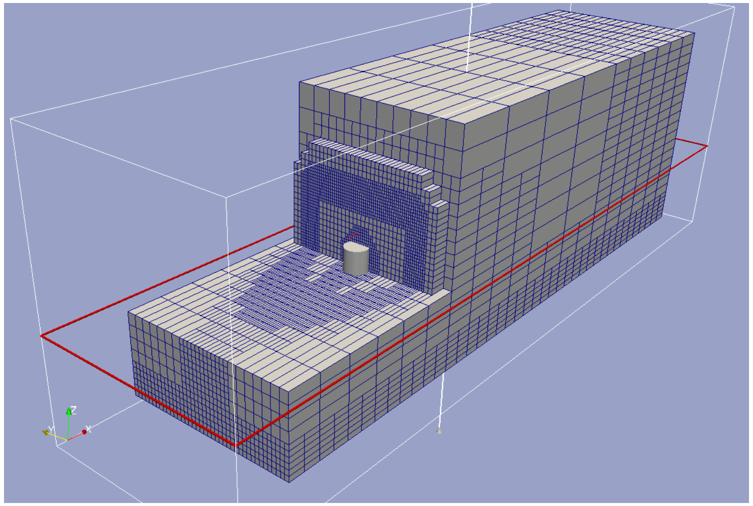
The Shape of CFDs to Come

"FVM for fluids OK; but **FEM** is essential for **solids**".

'Everyone knows' this; but **it is simply not true;** as has been demonstrated many times; and now here.

Here is a centrally-air-cooled object, held in a hot gas stream. It incurs both thermal and mechanical stresses.





Here is the unstructured grid on which all solid-stress-and-strain and hydrodynamic variables are computed **at the same time.**

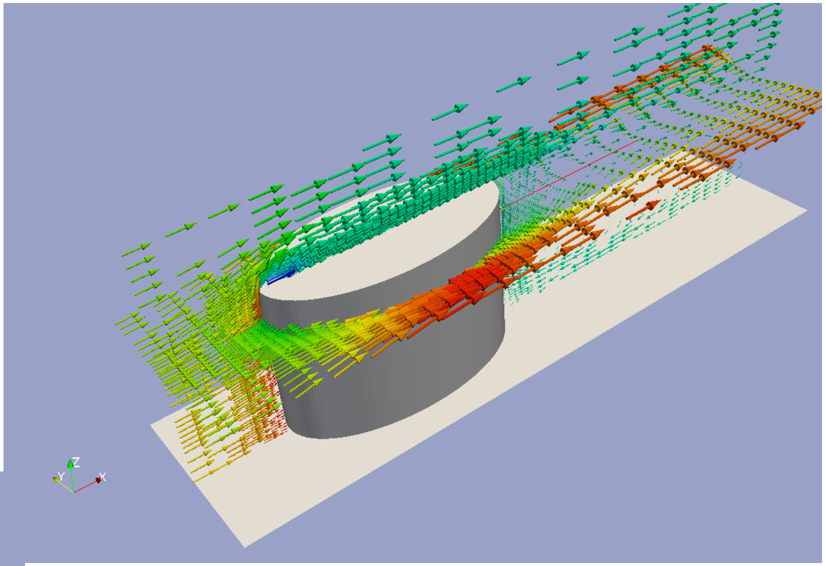
“FVM for fluids OK; but **FEM** is essential for **solids????**”. **Cherished Fallacy No.1** continued

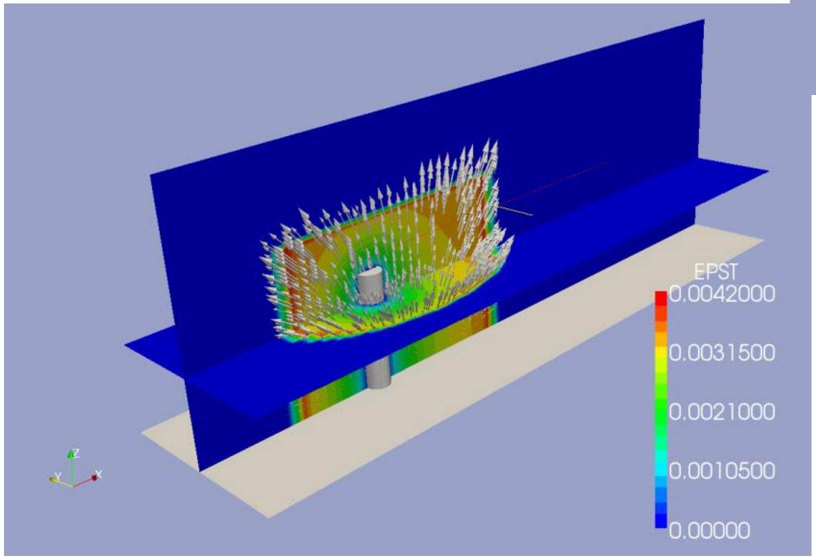
OpenFoam OCTOBER 2016

The Shape of CFDs to Come

Now for the calculations:

Here are the velocity vectors, which any CFD code can compute. No surprises here.





But here vectors are solid **displacements;** contours are thermal expansions.

Can your CFD code do it?

If not, use the right app.

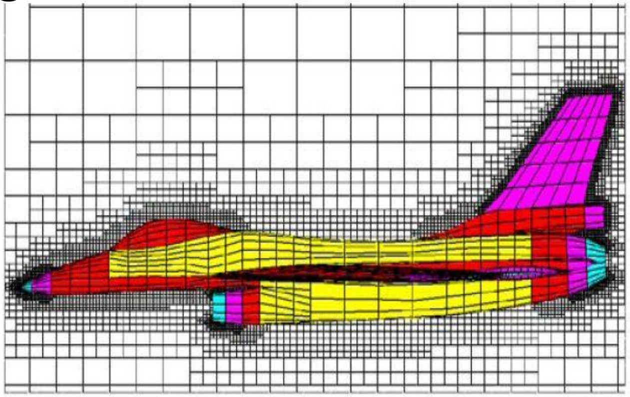
Other CFDs for the future; Cherished Fallacies Demolished: Cherished Fallacy No. 2

"**Simultaneous** is better than sequential ... always."
Not true, *eg* for calculating **aircraft lift and drag**.

Pre-CFD engineers iterated between:
(1) 2D boundary-layer theory near the surface and
(2) Source-sink potential-flow theory elsewhere.

Optimal **modern CFD-based apps** iterate between:
(1) Parabolic solution on fine 3D grids near surfaces, &
(2) Elliptic solution on coarser 3D grids elsewhere.

Codes lacking a parabolic option,
solve elliptically **everywhere**;
on grids always **expensive** but
never fine enough near surface.

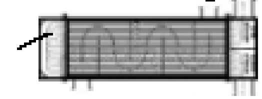


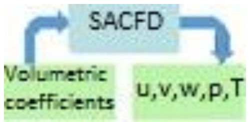
More about the “Simultaneous is better” fallacy

OpenFoam OCTOBER 2016

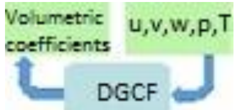
The Shape of CFDs to Come

Another example: the **heat-exchanger app** uses space-averaged CFD for the whole equipment; and detailed-geometry CFD for selected tube-bundle parts; which interact cyclically.

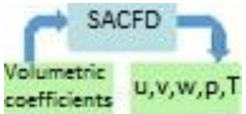




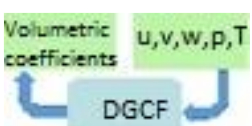
1. SACFD inputs **volumetric-coefficient formulae**, outputs mean velocities & temperatures at specific points.



2. DGCFD inputs **these velocities & temperatures** & outputs **improved volumetric-coefficient formulae**.



3. SACFD inputs these **formulae** & outputs **improved** mean **velocities & temperatures** at specific points.



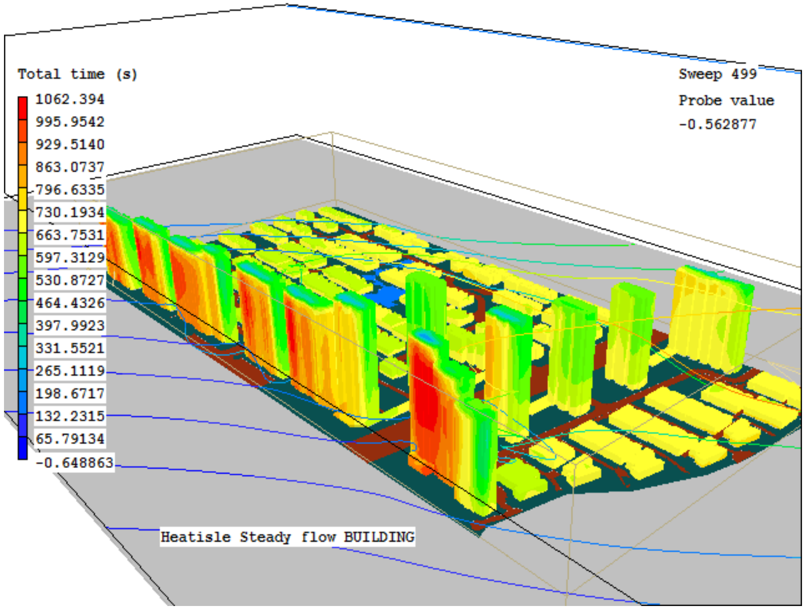
4. DGCFD inputs **improved velocities & temperatures** & outputs **further-improved** coefficient formulae.

5. And so on to **convergence**.

CFD-providers featuring **cyclically-sequential-run** capability will be chosen by creators of many apps.

Other CFDs for the future; Cherished Fallacies Demolished: Cherished Fallacy No. 3

"You have CAD; we add CFD" is often **a fallacious claim;** for which "that's what they want" is no excuse.



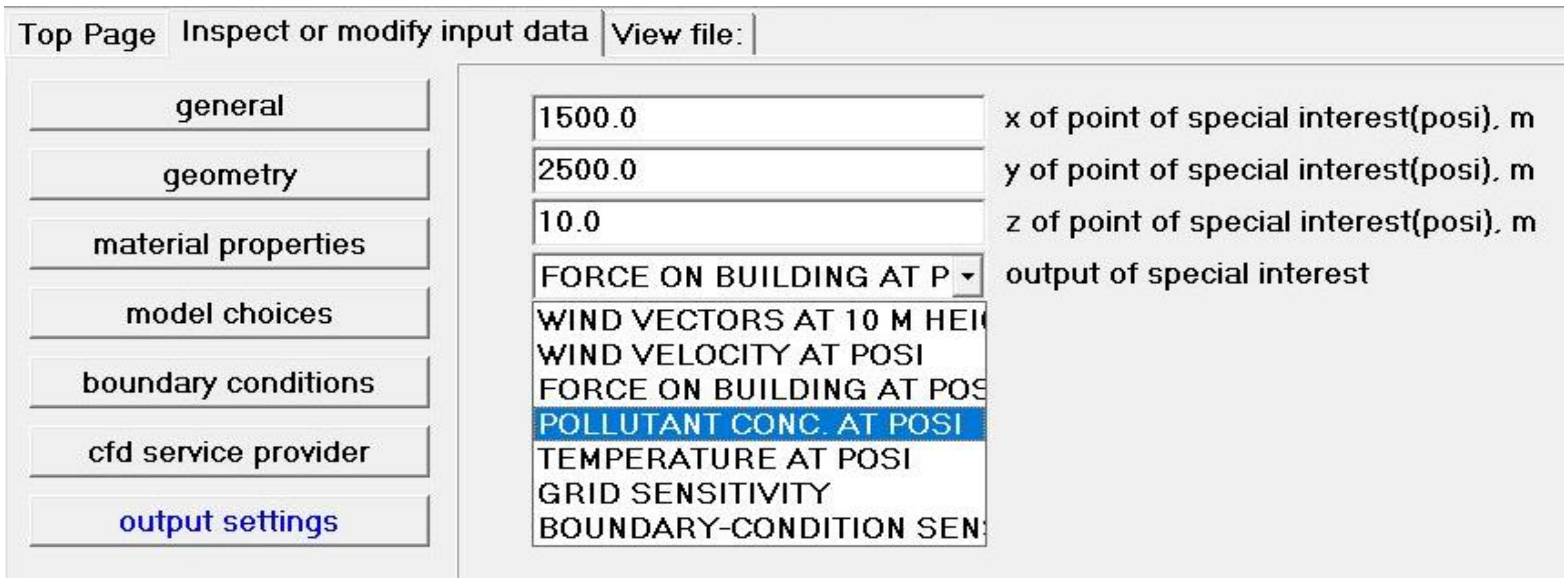
"Adding CFD" to a CAD-file-described scenario, given wind, pollutant, sun conditions, is (too?) easy.

But should **results** be relied upon? With care! because, usually:

* grid-cells are far too big:
* prescribed wind and sun conditions are guesses;
* turbulence models are no more than approximations;
* whether convergence has been achieved is uncertain;
* unless supplied by an app, needed checks are absent.

Determining what predictions the user hopes to rely upon

An app needs to **inquire** about **what most interests its users,** perhaps *via* a whole set of menus such as:



It can then: **automatically** refine the grid in indicated regions; print out **sensitivities** of selected items to less-certain inputs; advise on **reliability** of CFD predictions.

More CFDs for the future; Cherished Fallacies Demolished: Cherished Falllacy No. 3

“We add CFD” means nothing but: “we have a very simplistic view of what constitutes CFD”.

Thus CFD **can** simulate forest fires, in **various** ways;

but it needs to activate a **population model** of turbulent reaction, if its predictions are to **help fire-fighters.**

The app creator needs to know of, and **use only,** the CFD-service providers that offer such models.

Other CFDs for the future; Cherished Fallacies Demolished: CheFalDem No. 4

OpenFoam OCTOBER 2016

The Shape of CFDs to Come

"All turbulence modellers **must** follow the Kolmogorov pattern, *viz* by solving equations for **statistical-averages** such as: *k*, ε, vorticity fluctuations, Reynolds stresses, *etc.*"

Another untruth; although most modellers do believe it**;** and modish variants such as **Large Eddy Simulation** (of which there are many) may create the **illusion** of novelty.

Such models perform **badly** when body forces act differently on, say, hotter and colder elements in the turbulent mixture, as *eg* in forest fires.

My view? **Population theory** is the best way forward.

The Shape of CFDs to Come
Some final questions

Will the **Cherished Fallacies** be **Demolished**? Not soon. But to recognise that serious **fallacies do exist** is to make a start.

Will indeed general-purpose-code vendors gain their incomes increasingly *via* the 'cloud' rather than licence sales?

And will a two-ways-facing **community of app creators** become the main contact between CFD-users on the one hand and traditional code vendors on the other? I truly believe so.

From where will this imagined army of Januses arise? Many, I hope, from the attendants at **this Conference**.

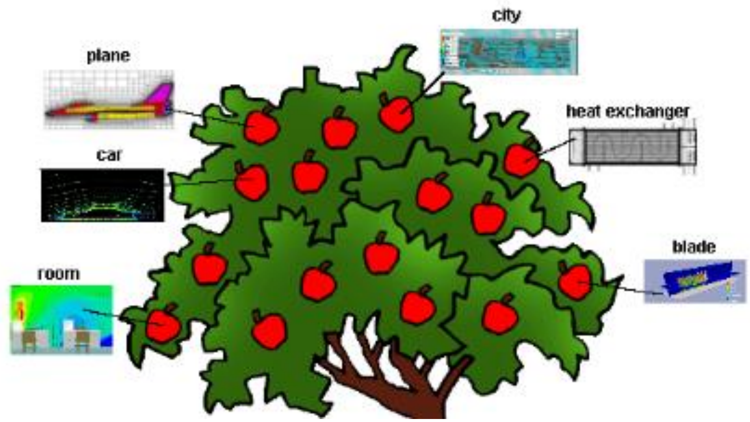
The Shape of CFDs to come: the CFD-App-Centred Vision

OpenFoam OCTOBER 2016

The Shape of CFDs to Come

Users:
Amy
Bill
George
Helga
Jeremy
Jill
Stephanie
Stephen
William
Zelda

App Creators



CFD-service providers:
ANSYS
CFX
OpenFoam
PHOENICS
StarCD
etc.
etc.

Applications: Heat exchanger, plane, room. liquid ring pump, city, forest fire, stress in pipe bend, turbine, virtual wind tunnel, furnace, stirred reactor, *etc, etc, etc.*

The End

Thank you for your attention.