Computational modelling of moving and deforming boundary flow problems

Presented by: K.S. Yeo
Department of Mechanical Engineering, National University of Singapore

2013 Fall Progress in Mathematical and Computational Studies on Science and Engineering Problems

November 11-13, 2013

308, Mathematics Research Centre, National Taiwan University

Overview of presentation

- ❑ Motivation
- ❑ The SVD-GFD method on hybrid Cartesian-meshfree grid system
- ❑ Aspects of nodal administration
- ❑ Test results for stationary and moving boundary flow problems
- ❑ Coupled Fluid-body interaction
- ❑ Test results for fluid-body coupled flow problems
- ❑ Applications to swimming and flying
- ❑ Recent development - A high-order SVD-GFD scheme with compact support

The SVD-GFD method on hybrid Cartesian-meshfree grid system*

Primary Motivation for development: applications to *biological-related self-propulsion / locomotion / flight* problems.

Key features of such problems are:

- *Complex geometry*
- *Multiple bodies*
- *Large boundary motion* (due to *shape changes* and *whole-body* motion) *and*
- *Flow-body/structure interactions*
- *Incompressible flows*
- *Largely external flows*

Slow motion Bee in flight (UltraSlo HD)3 - Youtube

The method that we set up is designed with these in mind.
After reviewing leading approaches, such FE, FV, Overset grid and IBM, we decided to start anew.

*Chew C.S., Yeo K.S., Shu C. (2006) J. Comp. Phys. **218**, 510-548.
Ang S.J., Yeo K.S., Chew C.S., Shu C. (2008) Intl. J. Num. Meth. Engrg. **76**, 1892-1929.

Hybrid Cartesian-meshfree grid system

The **hybrid grid** comprises:

- Cartesian background grid
- Meshfree nodes around the body.

By 'meshfree' is meant the absence of presumed connectivity between nodes.

For moving boundary/body:

Meshfree nodes convect with the body or boundary.

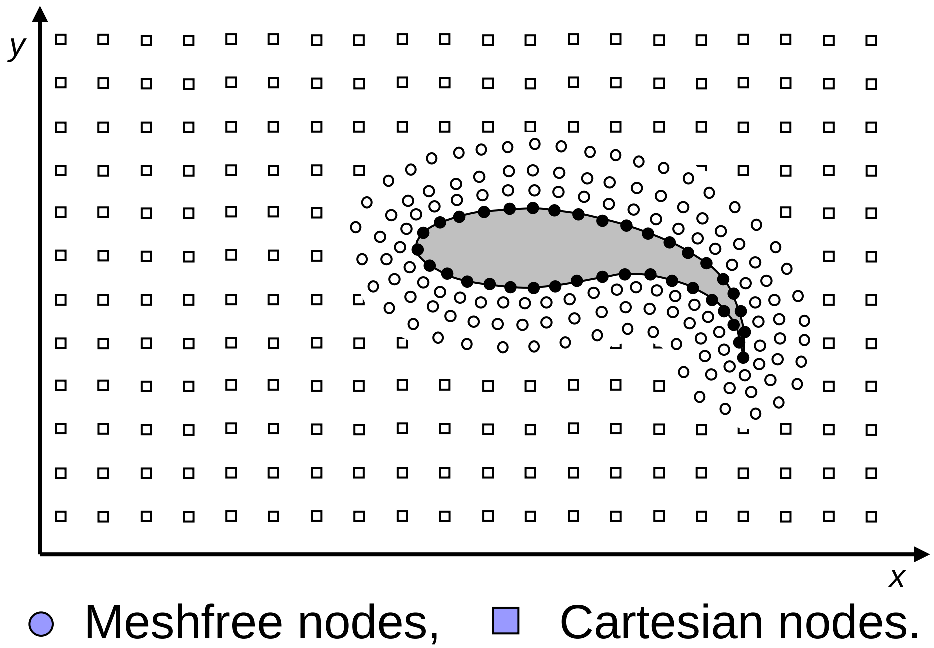


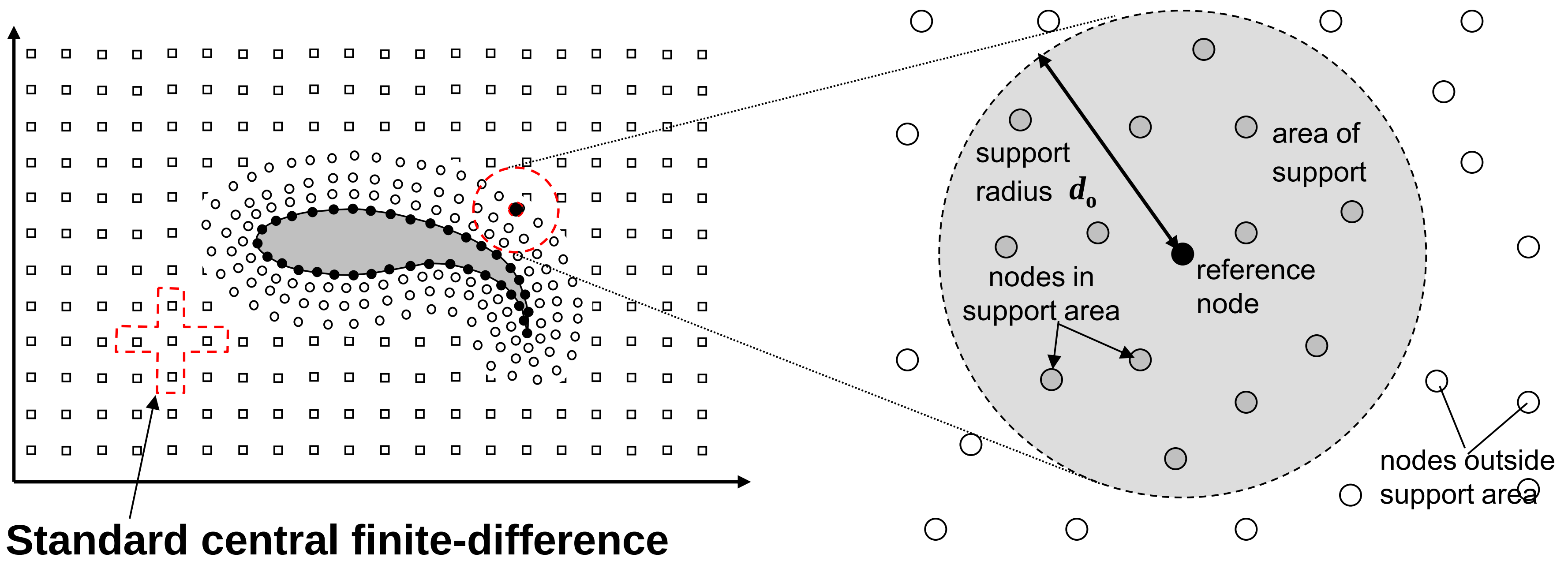
Figure 1. Hybrid Grid

Discretization approximations on the hybrid grid

Finite difference approximations are applied to governing equations:

- **Standard central finite-difference** on Cartesian-arranged nodes.
- **Generalized finite-difference** on other nodes.

Figure 2. Spatial discretization templates



Generalized finite-difference (GFD) approximation of derivatives – on a set of nodes with a certain radius.

The **generalized finite-difference** (GFD) approximation of spatial derivatives on a set of meshfree nodes is based on:

- **Taylor expansions** and
- **Least square (LS) approximation**.

For a function $f(\mathbf{x})$ of the 3D coordinate variable $\mathbf{x}$, the value of the function at $\mathbf{x}_1 = \mathbf{x}_0 + \Delta\mathbf{x}_1$ is given by:

$$f(\mathbf{x}_1) = f(\mathbf{x}_0) + \sum_{1 \le j_1 + j_2 + j_3 \le m-1} \frac{\Delta x_1^{j_1} \Delta y_1^{j_2} \Delta z_1^{j_3}}{j_1!\, j_2!\, j_3!} \left[\partial_x^{j_1} \partial_y^{j_2} \partial_z^{j_3} f \right]_{\mathbf{x}_0} + O(|\Delta\mathbf{x}_1|^m) \qquad (1)$$

where ∂_x, ∂_y and ∂_z denote the partial derivatives.

Applying (1) to ***n*** *support* nodes around the reference node $\mathbf{x}_0$ yields a system of linear equations, which relates the **derivatives** $\partial\mathbf{f}(\mathbf{x}_0)$ at the reference node to the **values of** f at the support nodes (m set to 4).

$$\Delta\mathbf{f}_{n\times 1} = [S]_{n\times 19}\,\partial\mathbf{f}_{19\times 1} \qquad (n \text{ support nodes; } m = 4) \qquad (2)$$

where $\Delta\mathbf{f}_{n\times 1} = (f_1 - f_0, f_2 - f_0, \cdots, f_n - f_0)^T$

$[S]_{n\times 19}$ is the configuration matrix of selected meshfree nodes

$$\partial\mathbf{f}_{19\times 1} = (\partial_x, \partial_y, \partial_z, \partial_x^2, \partial_x\partial_y, \cdots, \partial_y^3, \partial_z^3)^T f \,|_{\mathbf{x}_0}$$

We want to **obtain the derivatives** $\partial\mathbf{f}_{19\times 1}$ in terms of the **data** $\Delta\mathbf{f}_{n\times 1}$ at the ***n* nodes**.

To close linear system:

***n*=19** support nodes are needed for 3D problems and

***n*=9** support nodes for 2D problems.

However, the linear systems thus obtained based on ***n*=19 nodes** (3D) and ***n*=9 nodes** (2D) tend to be **poorly-conditioned** due to **nodal irregularity**.

Weighted least square approximation

Usually ***n*>19 (3D)** and ***n*>9 (2D)** nodes are required for good numerical conditioning in (2) (about 1.5 to 2.5 times).

The resulting **over-determined** system of equations is then solved via a least square method, where the Euclidean l_2-norm of residual error vector

$$\|\mathbf{r}\|_2^2 = \mathbf{r}^T\mathbf{r}, \quad \mathbf{r} := \Delta\mathbf{f}_{n\times 1} - [S]_{n\times 19}\,\partial\mathbf{f}_{19\times 1} \qquad (3)$$

is minimized with respect to the solution $\partial\mathbf{f}_{19\times 1}$ (**optimal solution**).

The **weighted error vector** $\mathbf{r}^T[W_n]\mathbf{r}$ is usually used, where $[W_n]$ is a diagonal matrix that give greater importance to components of error at nodes closer to the reference node.

The **conventional least square** process (also known as the Normal Equation method) then leads to closed system of linear equations for $\partial\mathbf{f}_{19\times 1}$:

$$[S]_{n\times 19}^T [W_n][S]_{n\times 19}\,\partial\mathbf{f}_{19\times 1} = [S]_{n\times 19}^T [W_n]\,\Delta\mathbf{f}_{n\times 1} \qquad (4)$$

Singular Value Decomposition (SVD)

The **conventional least square** loses accuracy and may be unstable to rounding-off error according to Trefethen & Bau*

The **singular value decomposition** (SVD) method for least square approximation is used here to find the derivatives $\partial \mathbf{f}_{19\times1}$,

$$\partial \mathbf{f}_{19\times1} = \underbrace{\left[W_n S\right]^{+}_{n\times19}}_{\text{pseudo-inverse}} \left[W_n\right] \Delta \mathbf{f}_{n\times1} \qquad (5)$$

Singular value decomposition

where $[W]_n$ is a distance-based ($|\mathbf{x}_i - \mathbf{x}_0|$) error weighting matrix.

For a non-singular (square) matrix $[A]$, the pseudo-inverse $[A]^{+} = [A]^{-1}$.

The **singular value decomposition** (SVD) method is used extensively in large scale data-mining and signal-processing applications to optimally to extract significant information.

*Trefethen L.N. & Bau D. (1997) Numerical Linear Algebra, SIAM Philadelphia, PA.

The **advantages** of SVD over Normal Equation for LS approximation are:

- Greater numerical stability and accuracy.
- SVD can be found for any matrix.
- In highly ill-conditioned situation, a *regularization* process can be carried out to remove 'noise' associated with contributions from small singular eigenvectors, which are frequently associated with noise in data.
- Can solve under-determined *under-rank* problems ($n<19$ in 3D cases), where there is a continuum of solutions, SVD picks the solution with smallest norm.

Note: On a standard Cartesian grid using just $n=4 << 9$ nodes (2D case) , GFD with SVD could accurately recover the result of standard FD.

Thus SVD gives the GFD scheme on hybrid Cartesian-meshfree grid a high degree of ***numerical stability*** and ***robustness*** in applications.

Flow equations

Flow equations modelled are the **incompressible Navier-Stokes equations** in **arbitrary Lagrangian-Eulerian (ALE)** form:

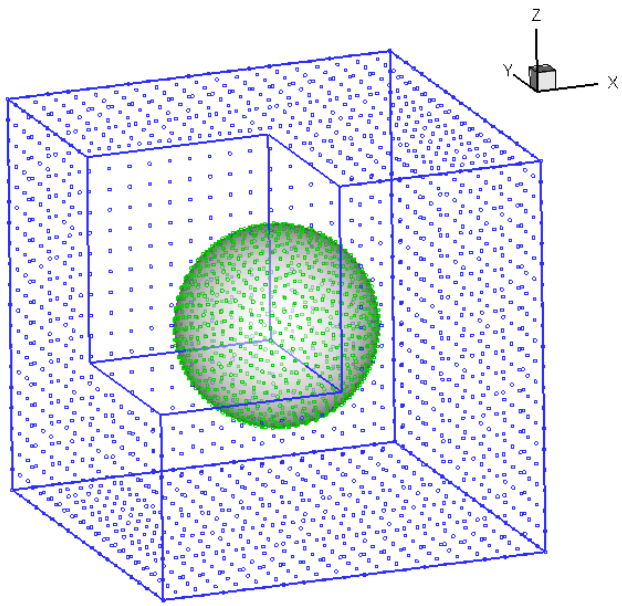
$$\partial_t \mathbf{u} + (\mathbf{u} - \mathbf{u}_c) \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{\mathrm{Re}} \nabla^2 \mathbf{u} \qquad (6)$$

$$\nabla \cdot \mathbf{u} = 0 \qquad (7)$$

where $\mathbf{u}_c$ is **convection velocity** for moving nodes.

- Both the meshfree nodes and Cartesian background nodes can be convected.

- The latter is particularly useful if we want to follow the motion of the swimmer or flyer over distances that are much longer than its length.



Hybrid Cartesian cum meshfree grid

Fractional-step method

The NS equation is discretized in time by the second-order *Trapezoidal rule* or *Crank-Nicolson* (CN) scheme:

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^{n}}{\Delta t}=-\frac{1}{2}\left[(\mathbf{u}\cdot\nabla\mathbf{u})^{n}+(\mathbf{u}\cdot\nabla\mathbf{u})^{n+1}\right]+\frac{\upsilon}{2}\left[\nabla^{2}\mathbf{u}^{n}+\nabla^{2}\mathbf{u}^{n+1}\right]-\frac{1}{2}(\nabla p^{n}+\nabla p^{n+1}) \tag{8}$$

It is solved by a *fractional-step projection* procedure:

$$\frac{\mathbf{u}^{*}-\mathbf{u}^{n}}{\Delta t}=-\frac{1}{2}\left[(\mathbf{u}\cdot\nabla\mathbf{u})^{n}+(\mathbf{u}\cdot\nabla\mathbf{u})^{n+1}\right]+\frac{\upsilon}{2}\left[\nabla^{2}\mathbf{u}^{n}+\nabla^{2}\mathbf{u}^{n+1}\right]-\frac{1}{2}\nabla p^{n} \tag{9}$$

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^{*}}{\Delta t}=-\frac{1}{2}\nabla p^{n+1} \tag{10}$$

where $\nabla^{2}p^{n+1}=\dfrac{2}{\Delta t}\nabla\cdot\mathbf{u}^{*}$ subject to the boundary condition (11)

$$\mathbf{n}\cdot\nabla p^{n+1}\Big|_{\partial\Omega}=\frac{2}{\Delta t}\left(\mathbf{u}*\big|_{\partial\Omega}-\mathbf{u}^{n+1}\big|_{\partial\Omega}\right)\cdot\mathbf{n}=\left(-\frac{\partial\mathbf{u}}{\partial t}\bigg|_{\partial\Omega}^{n+1}-(\mathbf{u}\cdot\nabla\mathbf{u})_{\partial\Omega}^{n+1}+\upsilon\nabla^{2}\mathbf{u}\big|_{\partial\Omega}^{n+1}\right)\cdot\mathbf{n}\,. \tag{12}$$

Aspects of nodal administration

❑ Initial set-up of problem at time t=0:

- Solid immersed body/nodes and its enveloping cloud of meshfree may be setup by established CAD and FE-based codes.
- These will also enable us to identify all the Cartesian nodes that are overlapped by the body (and nodal cloud) – these are denoted as Type-1.

❑ For GFD, the *n* nearest nodes to the reference node are normally selected as the support nodes. *n* is typically 1.5 to 2.5 times the minimum number.

❑ To facilitate search for near neighbours, each meshfree and boundary node *B* at a location $\mathbf{r}_B = (x, y, z)_B$ is assigned an 3-index:

$$(i, j, k)_B = \left(\mathrm{int}(x_B / \Delta x), \mathrm{int}(y_B / \Delta y), \mathrm{int}(z_B / \Delta z)\right) \qquad (13)$$

corresponding to the Cartesian background cell $(\Delta x, \Delta y, \Delta z)$ which contains it.

❑ A list of non-empty Cartesian cells, and its meshfree nodal content is maintained. The number of non-empty cells is generally very small since the meshfree/body nodes typically range over a very small fraction of the total computational domain in most problems.

Nodal administration (Contd.)

- **3 categories of Cartesian nodes** are identified for **stationary boundary** problems:

Type 1: Nodes overlapped by body or nodal cloud – they DO NOT participate in the flow computation.

Type 2: Nodes that DO NOT have meshfree/body nodes in its rectangular Δ-neighbourhood and are thus treated by standard FD.

Type 3: Nodes that HAVE one or more meshfree/body nodes in its rectangular Δ-neighbourhood and are thus treated by SVD-GFD.

- For **moving and deforming boundary** problems:

Type 4 Cartesian nodes: **Fresh nodes** that are uncovered by the moving boundary.

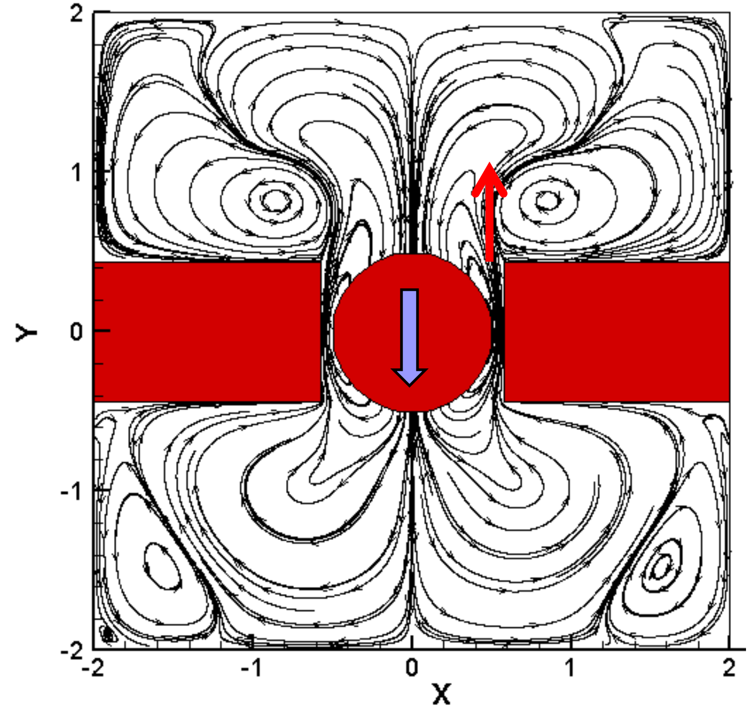
The **Nodal Type** of the Cartesian nodes in the vicinity of the body changes due to boundary motion and needs to be tracked and updated constantly.

- SVD-GFD is slightly more expensive than LS-GFD which is more costly standard FD. However, Meshfree and Cartesian nodes that need GFD treatment typically constitute a very small % of total nodal population for external flow problems: **< 5% for 2D problems and < 2% for 3D problems**.

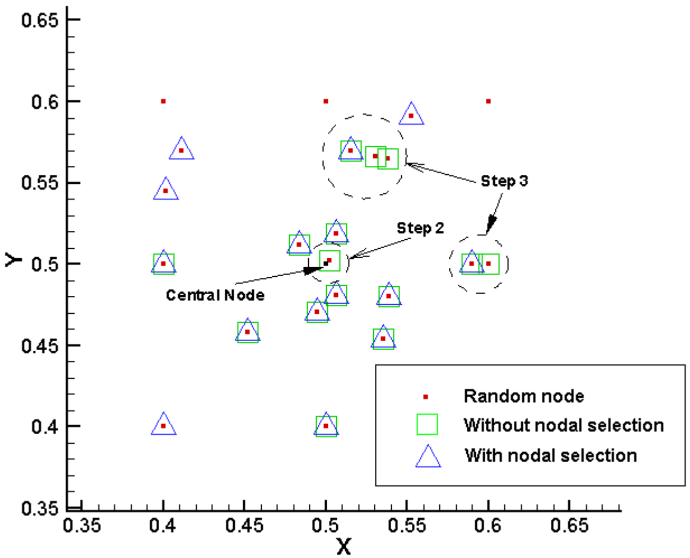
Close interaction in multi-body problems

A common cause of numerical ill-conditioning in GFD is **extremely close clustering** among some nodes.

When two bodies come into close contact, multiple close clustering could occur due to merging of meshfree nodal clouds



Oscillating cylinder in a narrow slot. Gap=0.075D.



A **nodal selection scheme** to de-cluster the support nodes is then necessary. **De-clustering** is done by simply discounting the unwanted support nodes. A scheme is described in Ang *et al*. (2008)*.

*Ang S.J., Yeo K.S., Chew C.S., Shu C. (2008) Intl. J. Num. Meth. Engrg. **76**, 1892-1929.

Some positive features of the numerical scheme:

- Complex body geometry.

Comparable to mesh-based finite-element and finite-volume methods. Better than over-set/chimera grid methods, which use local coordinate systems around immersed bodies.

- Relatively simple data structure; avoids re-meshing of mesh-based methods.

Primarily *positional data* of meshfree nodes. Nodes can be deleted or added with ease. Minimal interpolation is involved except in creation of new node.

- Simple and precise implementation of boundary conditions on body.

Compared to Immersed-boundary methods typically, where boundary conditions are enforced via distributed nodal forces.

- Good boundary and boundary layer resolution.

Compared to Immersed-boundary methods typically; where boundary smearing, leakages and feature resolution may be a problem. Cartesian cut-cell methods may be difficult to implement in 3D.

- Generalized Conservation Law (GCL) condition

Deforming mesh-based schemes must also satisfy GCL to ensure that changes in volume with time is taken into account in the numerical conservation equations.

Some weaknesses:

- Lack advantage of full Cartesian grid as compared to IBM etc.

Harder to program for parallelization. Currently parallelized for SMP and CUDA-based GPU systems.

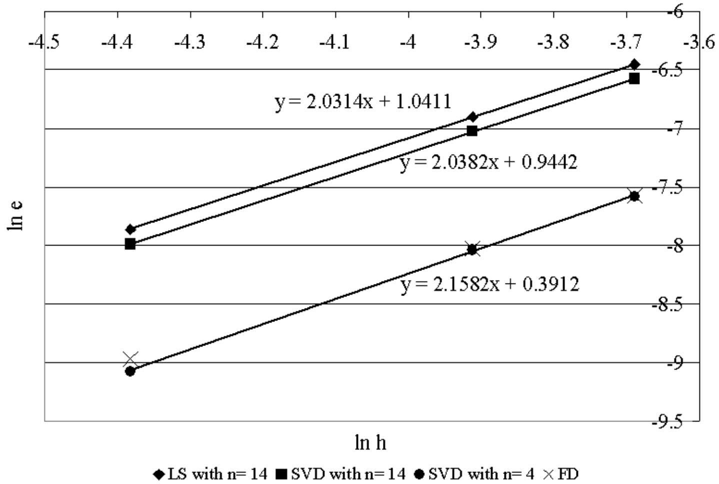
- Not conservative (in the sense of finite volume).

This is true for most methods whose implementation are not flux-based. For incompressible flows, conservation errors are bounded by discretization errors, which become small as grid system is refined.

Stationary boundary problems

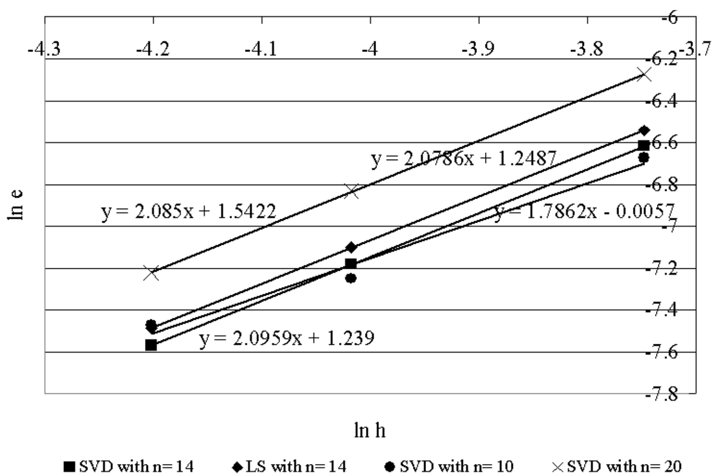
Test results – 2D Poisson problems*

Convergence of numerical solutions on Cartesian grids.



- SVD-GFD gives slightly improved errors on Cartesian grids (n=14) when compared to LS-GFD.
- SVD-GFD with just 4 support nodes (under rank by 5) gives similar errors with standard FD.

Convergence of numerical solutions on Randomized grids: 1800, 3088, 4465.

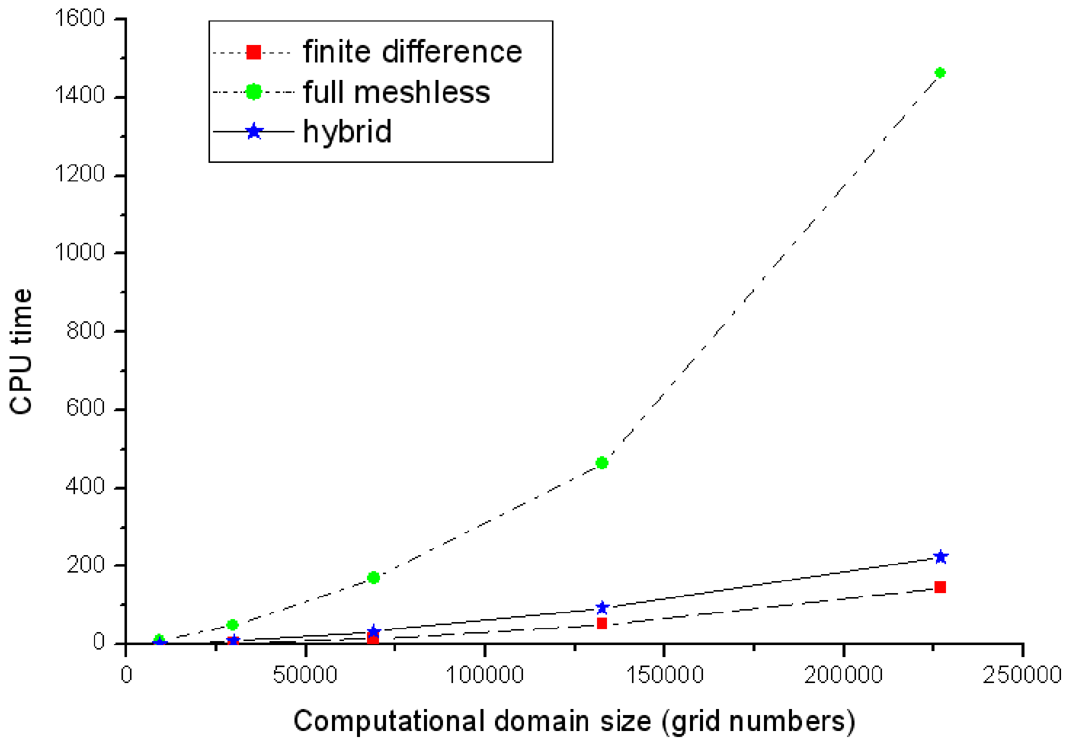


- SVD-GFD gives slightly improved errors on Randomized grids (n=14).
- Absolute errors increase with support node number n. Use smallest n compatible with stability.
- The order of grid convergence is 2. Regularization at low n can reduce convergence order.

*Ang S.J., Yeo K.S., Chew C.S., Shu C. (2008) Intl. J. Num. Meth. Engrg. **76**, 1892-1929

Test results – 3D Poisson problems*

Comparison of CPU times for 3D Poisson problem as a function of grid number: (1) Cartesian grid with standard FD, (2) Full meshfree grid with SVD-GFD and (3) Hybrid Cartesian-meshfree (12.5%) grid.



*Wang X.Y.,Yeo K.S., Chew C.S., Khoo B.C. (2008) Comp. & Fluids **37**, 733-746

- Cost of SVD-GFD can be kept reasonably low when the meshfree nodes are a small percentage of the total nodal population.

- In most external flow problems: meshfree/total nodes is < 5% for 2D problems and <2-3% in 3D problems.

Test results: Decaying Vortex (2D)*

Solved in a square computational domain:

$$(x,y) \in [-0.5, 0.5] \times [-0.5, 0.5]$$

with Re=10 and a small $\Delta t = 10^{-5}$ on:

- Cartesian grid
- Randomized grid (Dirichlet pressure BC)
- Randomized grid (Neumann pressure BC)

Exact solution:

$$u(t,x,y) = -[\cos(\pi x)\sin(\pi y)]e^{-2\pi^2 t/\mathrm{Re}},$$

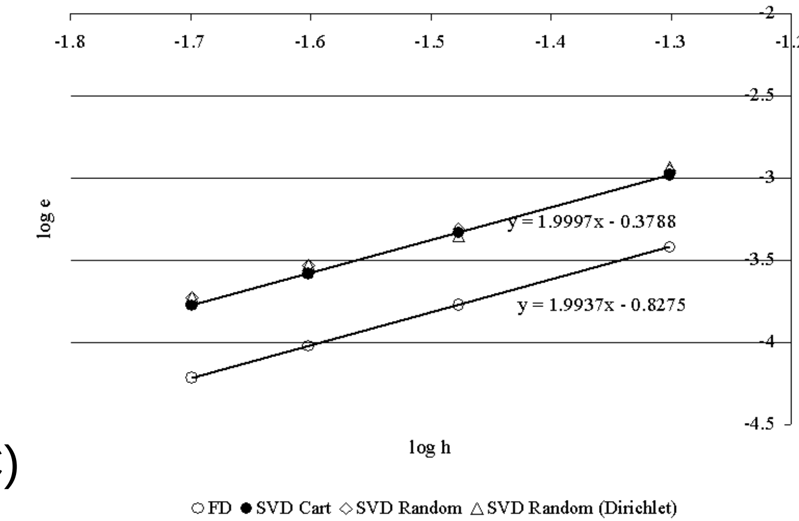
$$v(t,x,y) = [\sin(\pi x)\cos(\pi y)]e^{-2\pi^2 t/\mathrm{Re}},$$

$$p(t,x,y) = -\frac{1}{4}[\cos(2\pi x) + \cos(2\pi y)]e^{-4\pi^2 t/\mathrm{Re}}.$$

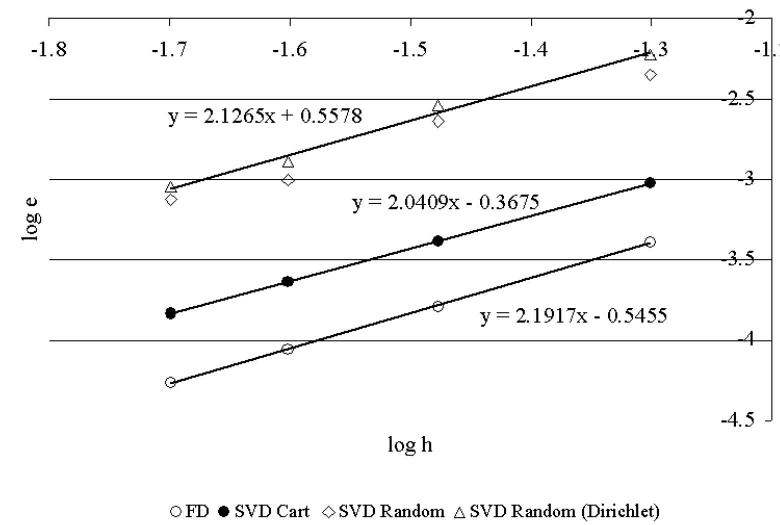
Numerical solutions second-order in space for both velocity and pressure.

*Ang S.J., Yeo K.S., Chew C.S., Shu C. (2008) Intl. J. Num. Meth. Engrg. **76**, 1892-1929

Spatial accuracy of u and v



Spatial accuracy of pressure.



Test results: A decaying flow – temporal accuracy*

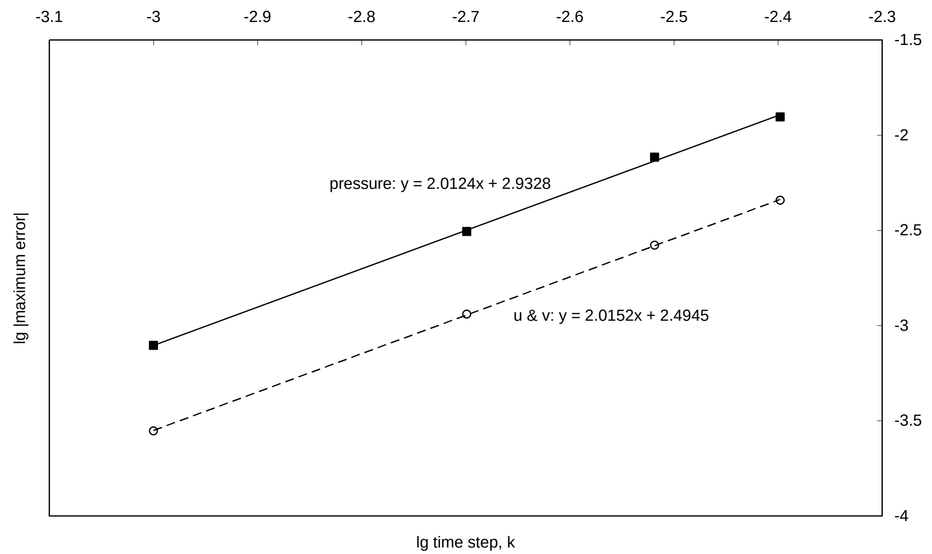
Decay of an initial flow field in a square box with 81x81 grid points at Re=1000:

$$u(x,y,0) = (1-\cos 2\pi x)\sin 2\pi y,$$
$$v(x,y,0) = (\cos 2\pi y - 1)\sin 2\pi x,$$
$$p(x,y,0) = 0,$$

Comparison with Reference solution at t=0.1 with $\Delta t=10^{-5}$:

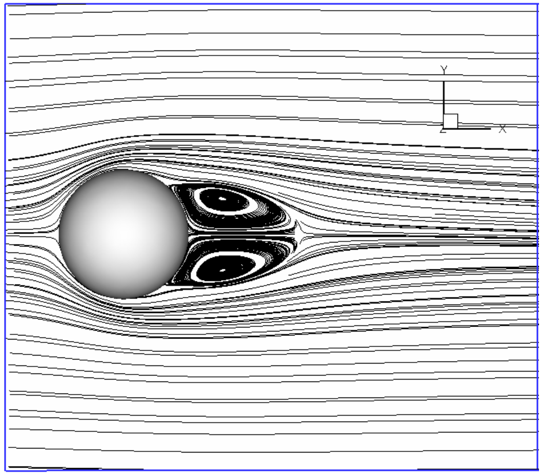
Numerical solutions second-order in time for both velocity and pressure.

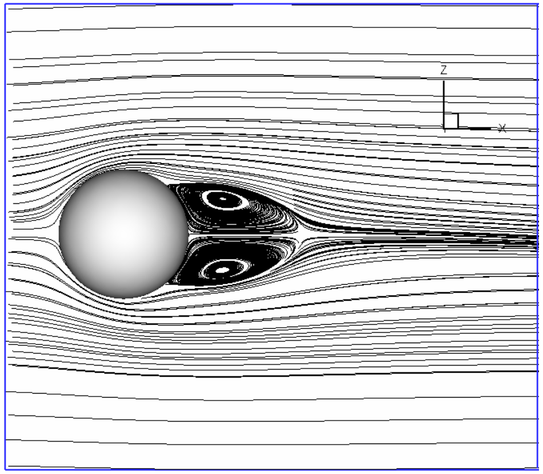
Temporal accuracy of velocity and pressure relative to reference solution.



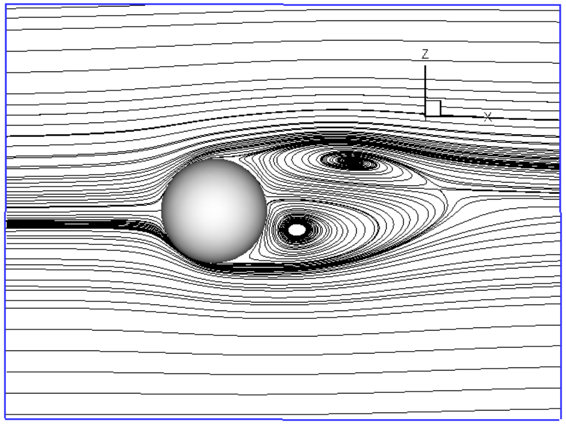
*Chew C.S., Yeo K.S., Shu C. (2006) J. Comp. Phys. **218**, 510-548.

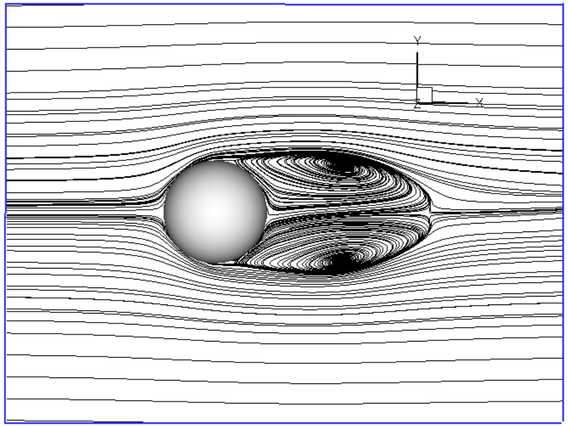
Test results: Stationary boundary problem – Flow past a sphere





Re=100: Steady axisymmetric flow with attached vortex.





Re=250: Quasi-steady non-axisymmetric flow with tilted vortex wake. In transition region between 210 – 270 (from steady axisymmetric to unsteady shedding).

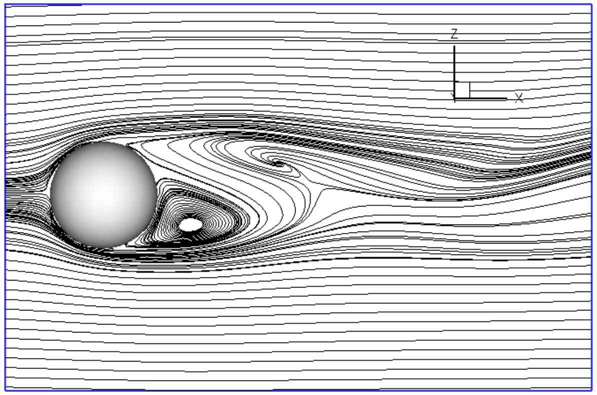
Discretization: 3854 nodes on sphere with 5 layers of meshfree grid.

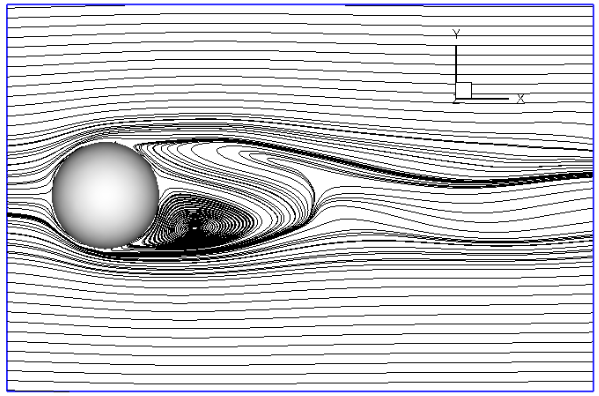
Domain 20,10,10, Cartesian Δ=0.025.

Meshfree nodes <1%.

There are 3 flow regimes.

Flow past sphere (Contd.)





Note: The meshfree grid used here is non-symmetric with respect to flow direction.

Re=300: Unsteady vortex shedding with Strouhal no. $St = Df_s/U$ = 0.132 (Johnson et al. St=0.137, Tomboulides et al. St=0.136).

Comparison of separation length and force coefficients for flows past a sphere

Reynolds no.	Results from	X_s	C_d	$\|C_l\|$
100	Current method	$0.86D$	1.108	0.006
	T.A. Johnson [25]	$0.88D$	1.112	–
	A. Gilmanov et al. [27]	$0.894D$	1.153	–
	S. Lee [26]	$0.98D$	1.096	–
	Clift et al. (exp) [28]	$0.89D$	1.087	–
250	Current method	$1.63D$	0.746	0.064
	T.A. Johnson [25]	$1.617D$	–	0.062
	A. Gilmanov et al. [27]	$1.62D$	–	–
300	Current method	–	$0.690 \pm 3.8e{-}3$	$0.071 \pm 1.9e{-}2$
	T.A. Johnson [25]	–	$0.656 \pm 3.5e{-}3$	$0.069 \pm 1.6e{-}2$
	P. Ploumhas et al. [29]	–	0.657	$0.061 \pm 1.3e\text{-}2$
	Tomboulides et al. [24]	–	$0.671 \pm 2.8e{-}3$	–

Moving boundary problems

Test results: 2D moving nodal patch in a **decaying vortex***

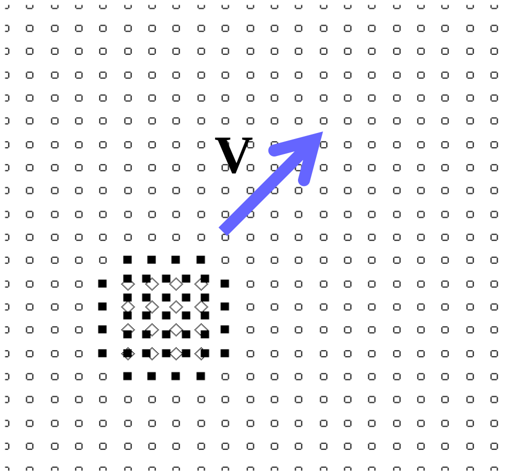
This case tests the errors generated by a moving patch of meshfree nodes (with **no physical body**) in a decaying vortex flow with Re=100.

The nodal patch covers over the space and its nodes partake in flow computation via **ALE LS-GFD**.

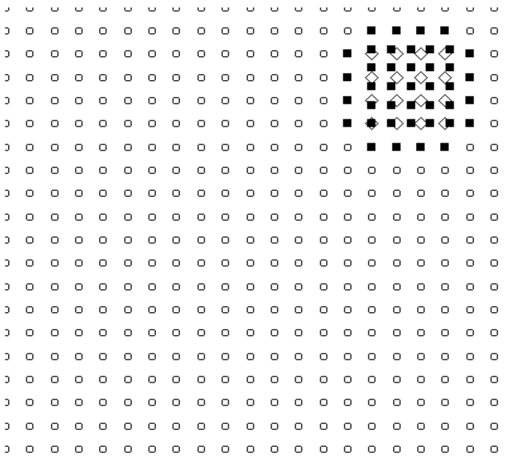
Nodal patch has nodal interval of 0.8h of background grid interval h.

The following are examined:

- Errors of ALE nodal convection solution relative exact solution.
- Errors of solutions with nodal patch versus solution with NO patch.



time t=0.0



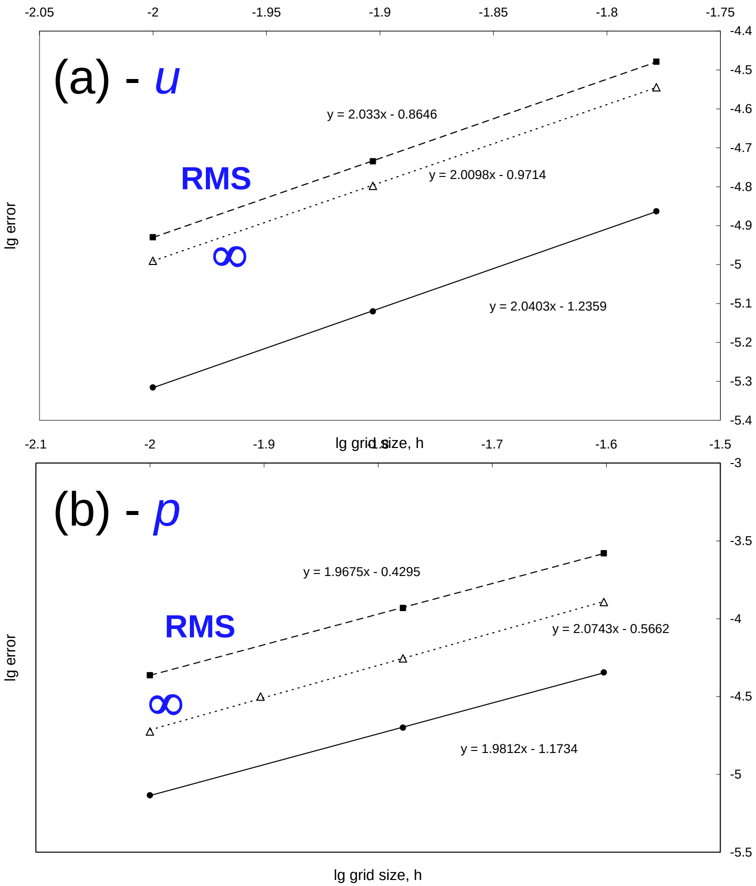
time t=1.0

$$\mathbf{V} = 0.25\sin(\pi t/3)(1,1)^T$$

*Chew C.S., Yeo K.S., Shu C. (2006) J. Comp. Phys. **218**, 510-548.

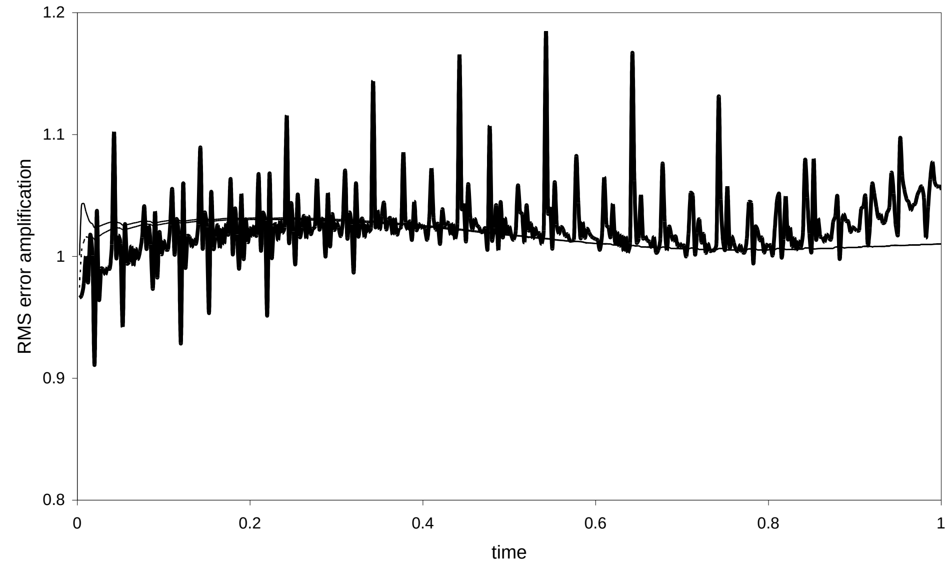
2D moving nodal patch (Contd.)

Spatial errors (RMS, ∞, L2 norm) under LS-GFD with ALE nodal convection at Re=100: (a) u and (b) p.



- ALE-GFD results maintain second-order spatial accuracy.

***Relative* RMS errors for ALE solution (u,v,p) with nodal patch versus FD solution with no patch.**



- Error amplification peaks are associated with *fresh node* creation.
- Amplification of (u,v) errors due to pure nodal convection is very small.
- Amplification factor of p errors due to pure nodal convection <1.2.

Test results: Single moving body
ALE-LS-GFD versus Moving-frame simulation*

In **moving-frame simulation**, one attaches a computational frame to the moving body and apply a moving frame formulation of NS equations (as many has done). The key advantage here is that the grid is fixed.

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \frac{1}{\mathrm{Re}}\nabla^2 \mathbf{u} + \frac{\partial \mathbf{u}_{frame}}{\partial t}$$

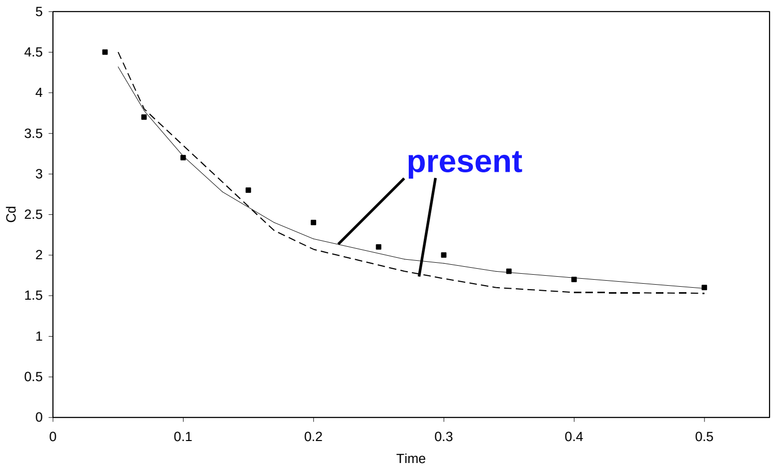
If we can recreate exactly equivalent boundary conditions, this will provide a **consistency** test of the two formulations.

Three cases are given below:

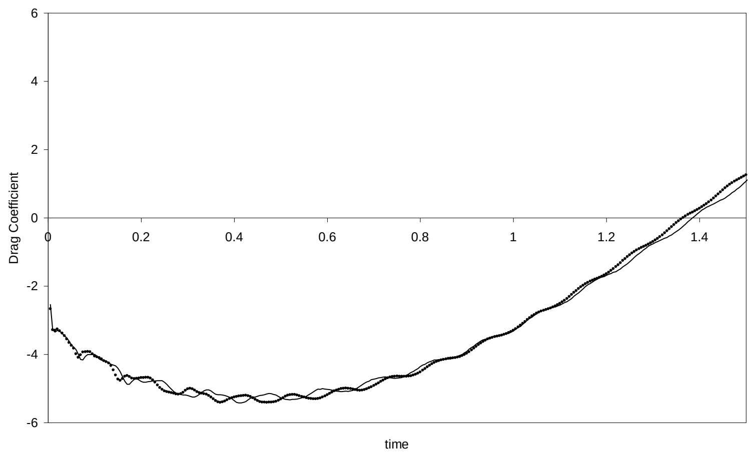
- Impulsively started cylinder (very abrupt starting condition)
- Sinusoidally started cylinder
- Sinusoidally oscillating cylinder

*Chew C.S., Yeo K.S., Shu C. (2006) J. Comp. Phys. **218**, 510-548.

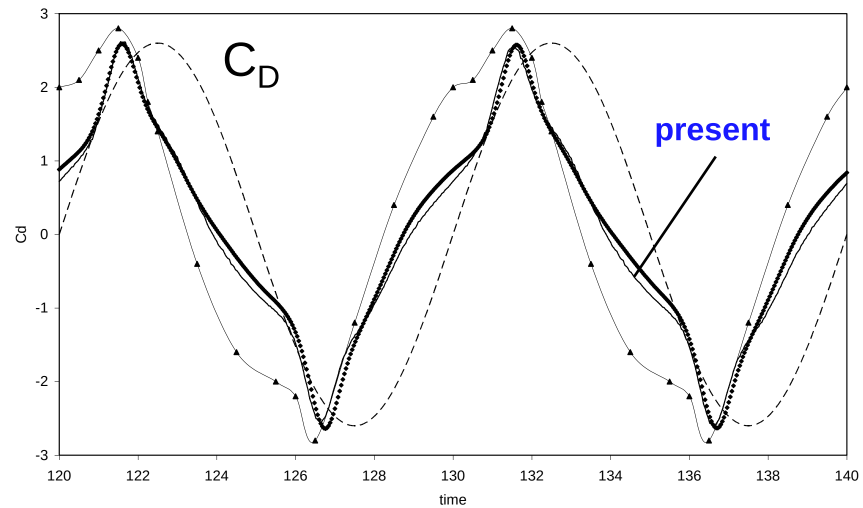
Present ALE-LS-GFD versus Moving-frame (Contd.)

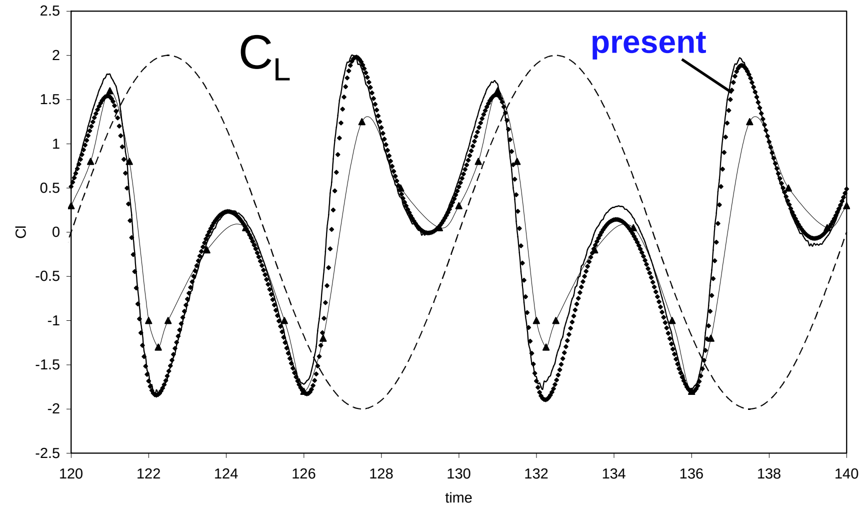


Drag on Impulsively started cylinder at Re=100. ALS-GFD case: u $\rightarrow$ 1.0 in time 0.02.



Drag on Sinusoidally started cylinder at Re=20.



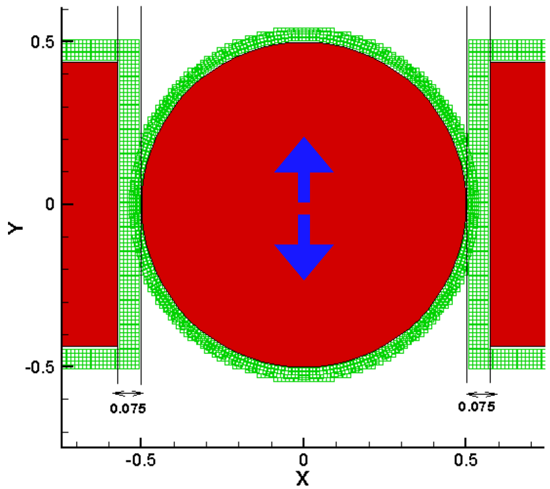


C_D and C_L for longitudinally oscillating cylinder at Re=200 and KC=$U_{max}T/D$=10.

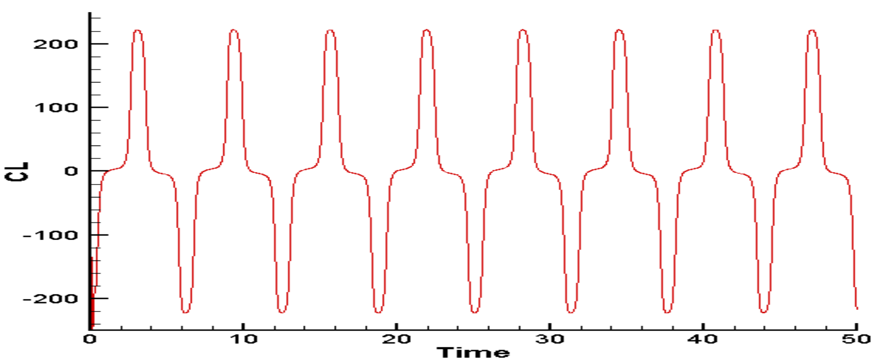
Example applications: Moving bodies in close interaction with Nodal Selection

Oscillating cylinder through slot in a box

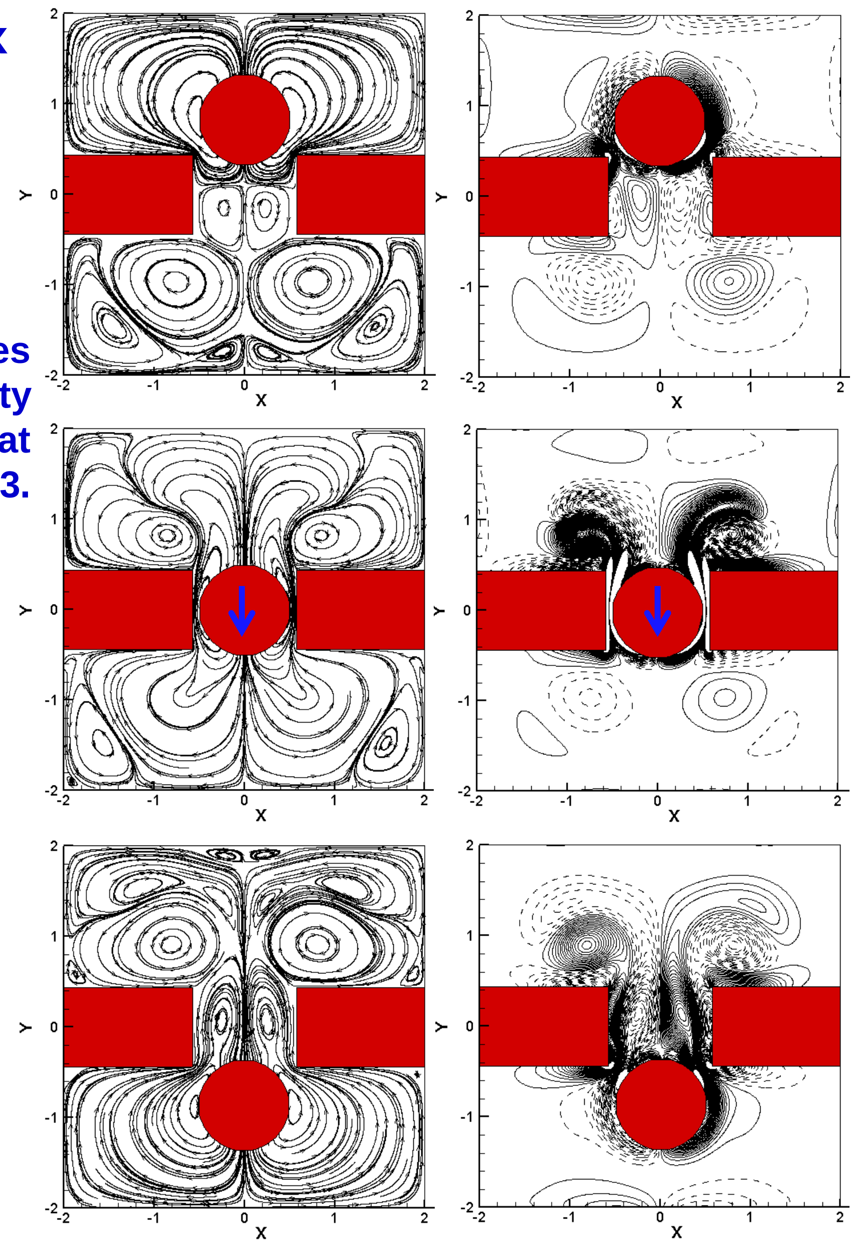
Overlapping Meshfree grids. Nodal selection scheme used. Gap:0.075D; Re=50.



Vertical lift coefficient

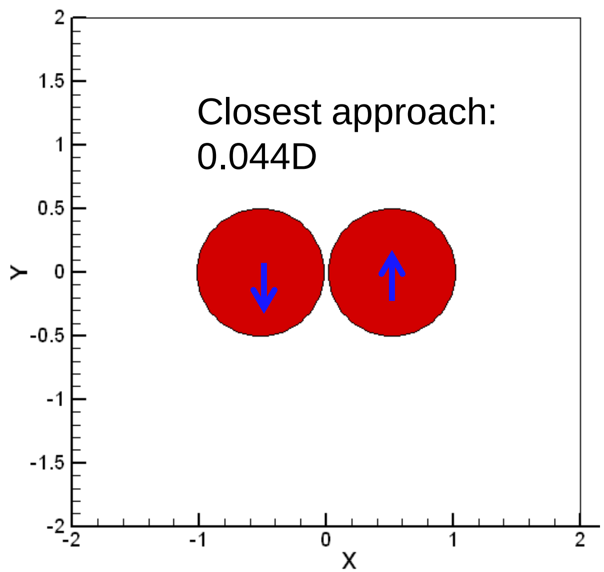


Stream traces and Vorticity contours at t=21,22,23.



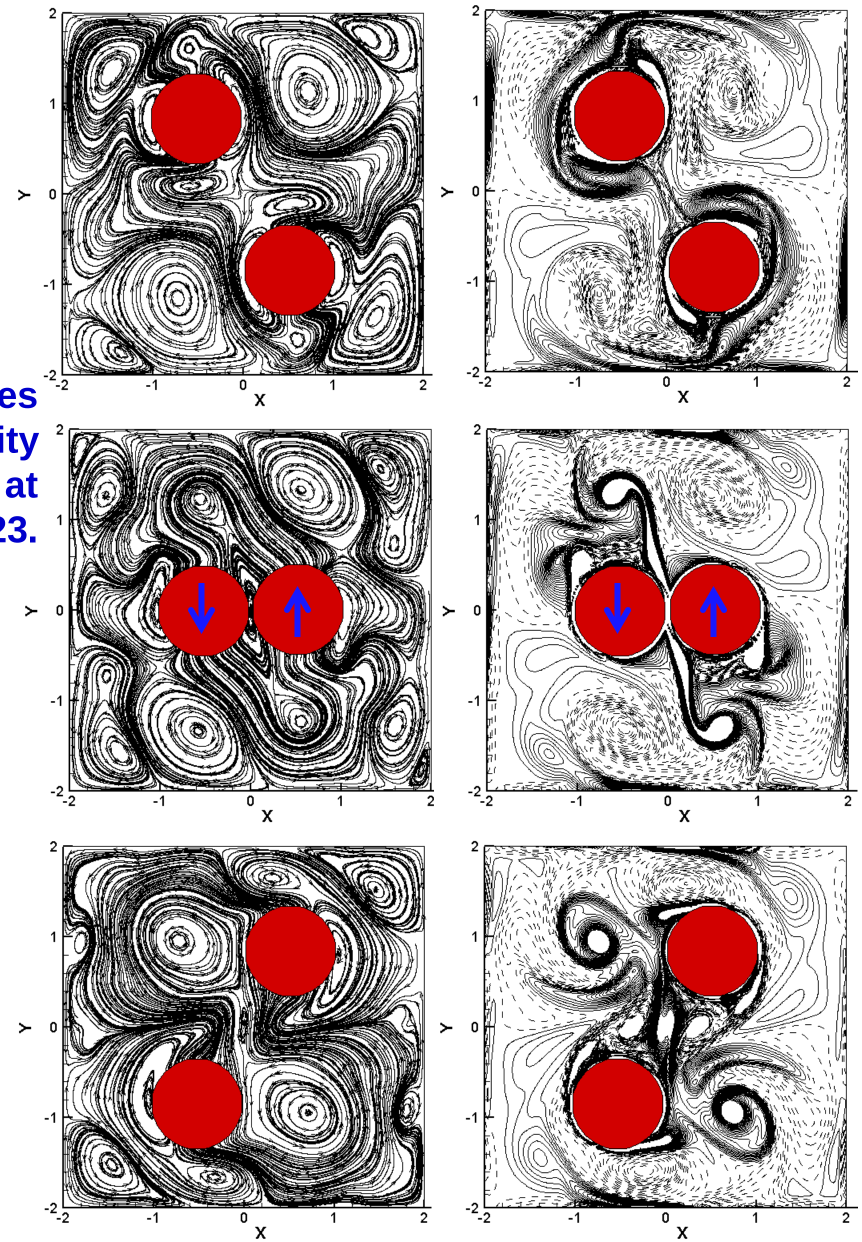
Moving bodies in close interaction – Contd.

Two side-by-side cylinders oscillating in anti-phase in a box (with Nodal Selection)



Gap at closest approach = 0.044D,
$Re = 2U_{max} D/\upsilon = 1000$.

Stream traces and Vorticity contours at t=21,22,23.



Test results: 2D tandem flapping-wing pair

In phase stroking

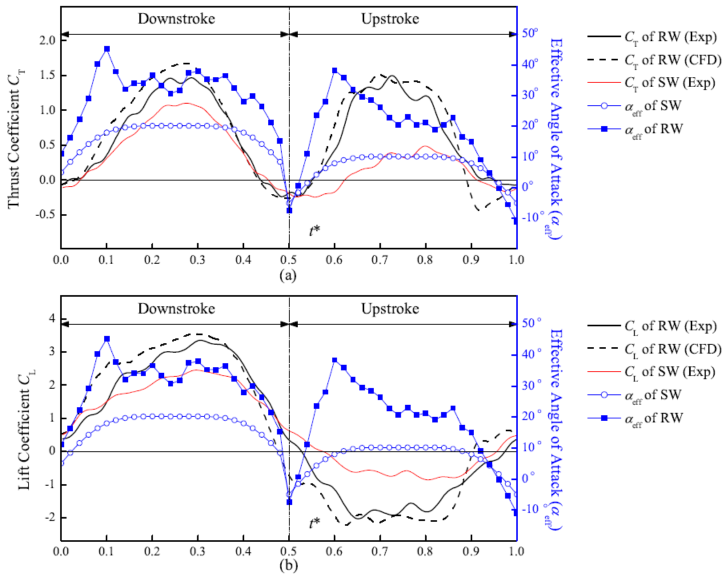


Figure 6.8: Experimentally and numerically obtained transient thrust and lift coefficients (C_T and C_L) of the rear wing in the case of in-phase stroking ($\varphi = 0°$). The effective angles of attack (α_{eff}) of the rear wing and the single wing are also included.

Out-of-phase stroking

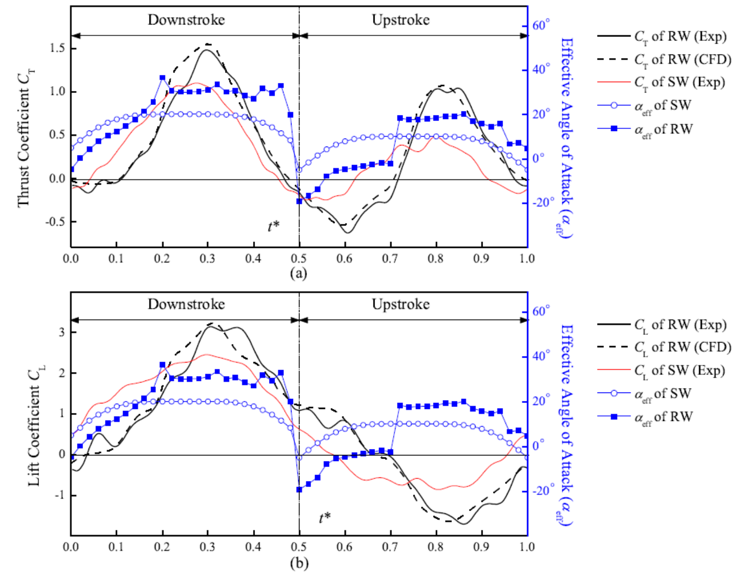
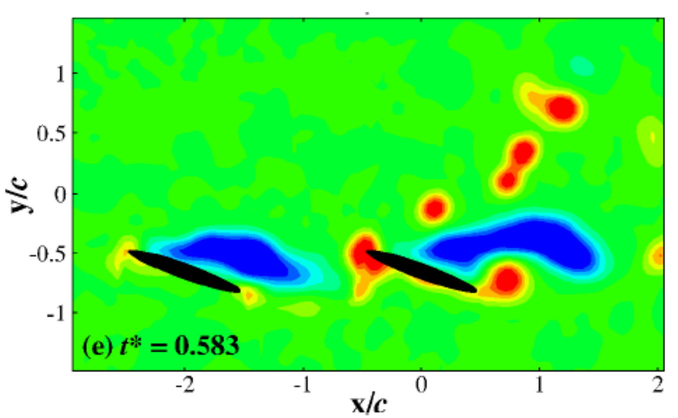
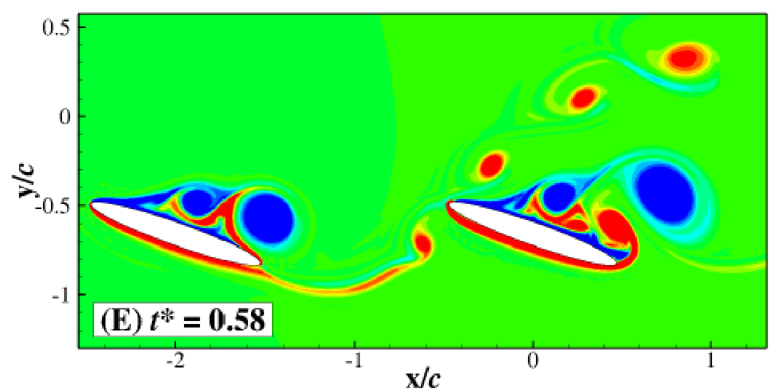


Figure 6.17: Experimentally and numerically obtained transient thrust and lift coefficients (C_T and C_L) of the rear wing in the case of $\varphi = 90°$. The effective angles of attack (α_{eff}) of the rear wing and the single wing are also included .

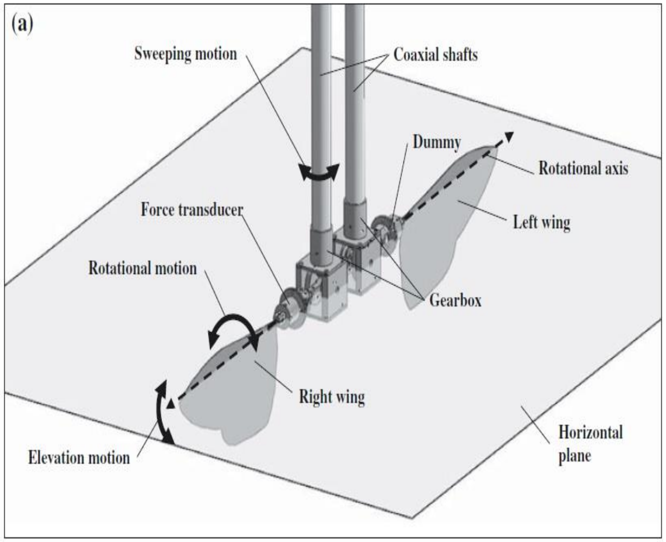


Vorticity field (Expt)

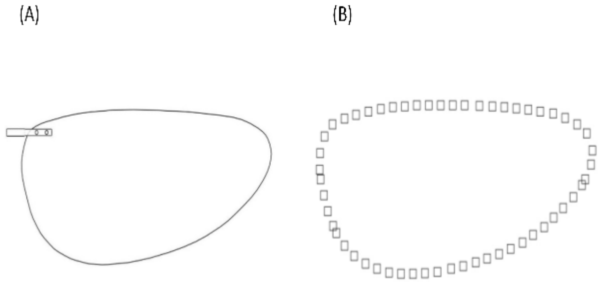


Vorticity field (Computational)

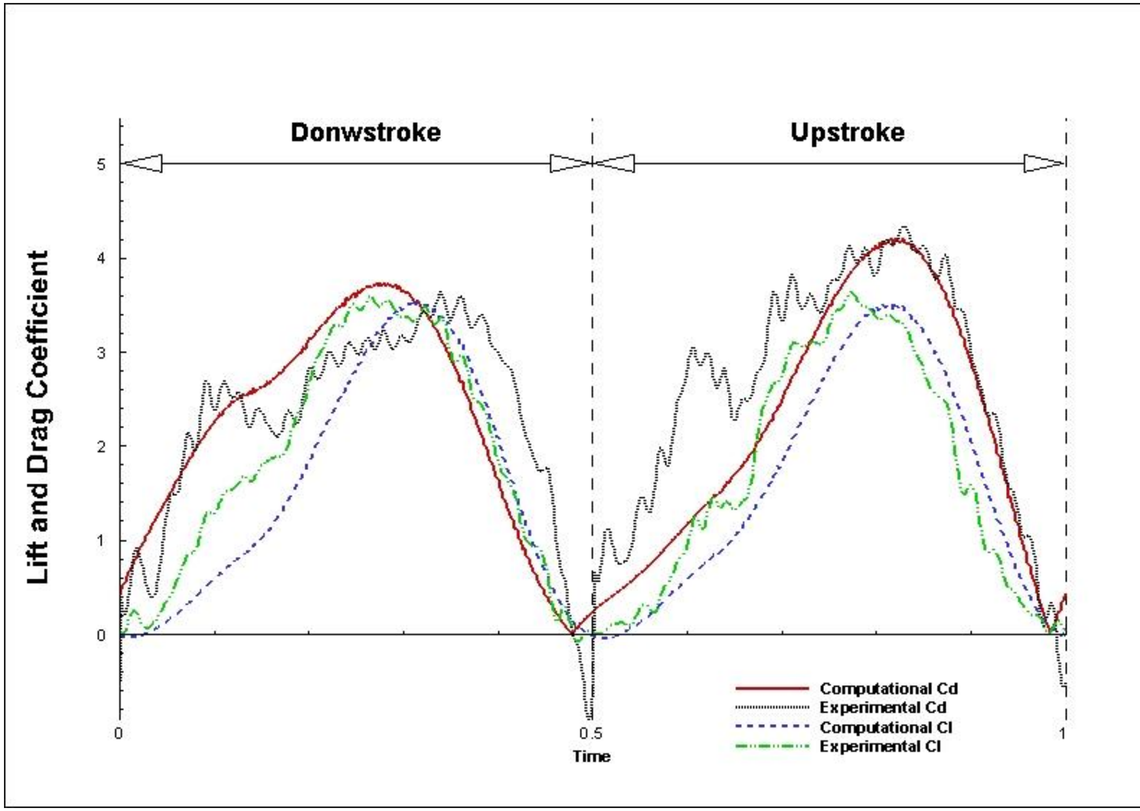
Test results: 3D moving boundary – flapping wings – comparison with experimental results*



Schematics of test rig.



Plan form of fruit fly wing



Comparison for lift and drag for fruit fly type wing at Re =150.

*Lua, K. B., Lai, K. C., Lim, T. T., & Yeo, K. S. (2010). *EXPERIMENTS IN FLUIDS*, **49**(6), 1263–1291.

Coupled Fluid-body interaction

Coupled Fluid-body interaction

Dynamics of the body is governed by **Newton's Laws**:

$$\frac{d\mathbf{P}}{dt}(t) = \mathbf{F}(t) + \mathbf{F}_{gr} \qquad \frac{d\mathbf{L}_C}{dt}(t) = \boldsymbol{\tau}(t)$$

(Conservation of Linear and Angular Momentum about CM at $\boldsymbol{X_C}$.) (14-15)

$$\mathbf{P} = \sum_i m_i \mathbf{v}_i = M\mathbf{V}_C \qquad \mathbf{L}_C = \sum_i \mathbf{x}_{i/C} \wedge m_i \mathbf{v}_{i/C} = [I_C] \cdot \boldsymbol{\omega}_C$$

(16-17)

where

$$\mathbf{F}(t) = \int_{\Gamma(t)} \left([\sigma_f] \cdot \mathbf{n} \right) \, dS$$

(Total fluid force) (18)

$$\boldsymbol{\tau}(t) = \int_{\Gamma(t)} \left(\mathbf{x}(t) - \mathbf{X}_C(t) \right) \wedge \left([\sigma_f] \cdot \mathbf{n} \right) dS$$

(Moment of fluid forces about CM at $\boldsymbol{X_C(t)}$) (19)

$$[\sigma_f] = -p\mathbf{I} + \frac{1}{\mathrm{Re}} \left[\nabla\mathbf{u} + (\nabla\mathbf{u})^T \right]$$

(Newtonian fluid stress tensor) (20)

where $\Gamma\left(t, \mathbf{X}_C(t), \boldsymbol{\Theta}(t)\right)$ denotes the current configuration of the body.

Coupled Fluid-body interaction (Contd.)

These are supplemented by the kinematic equations:

$$\frac{d}{dt}\mathbf{X}_C = \mathbf{V}_C \qquad \text{(Translation of CM)} \qquad (21)$$

$$\frac{d}{dt}\mathbf{\Theta}_C = \boldsymbol{\omega}^C \wedge \mathbf{\Theta}_C = [\![\boldsymbol{\omega}^C]\!]\mathbf{\Theta}_C \qquad \text{(Body frame rotation about CM)} \qquad (22)$$

where $\mathbf{\Theta}_C = (\mathbf{i}_1^C, \mathbf{i}_2^C, \mathbf{i}_3^C)$ is the orientation matrix of body frame at CM.

Equations (14,15,21,22) are integrated together with the flow equations.

The trapezoidal rule can again be applied to the integration of these equations:

$$\frac{\mathbf{Y}^{n+1} - \mathbf{Y}^n}{\Delta t} = \frac{1}{2}\left(\zeta^{n+1} + \zeta^n\right) \qquad (23)$$

where $\mathbf{Y} = \left(\mathbf{X}_C, \mathbf{\Theta}_C, \mathbf{P}, \mathbf{L}\right)$ and $\zeta = \left(\mathbf{V}_C, [\![\boldsymbol{\omega}^C]\!]\mathbf{\Theta}_C, \mathbf{F} + \mathbf{F}_{gr}, \boldsymbol{\tau}\right)$

Algorithms for Fluid-body Interaction

The dynamically-coupled equations of fluid and body at time t is solved by a time-iterative procedure that iterates on $\mathbf{X}_C(t)$ and $\mathbf{\Theta}_C(t)$ to determine the configuration $\Gamma(t, \mathbf{X}_C(t), \mathbf{\Theta}_C(t))$ of the body. The algorithm follows:

Step 1: Assume that current flow field solution and body solution $\mathbf{Y}^n$ at time level *(n)* is known.

Step 2: Assume that i-th approximation to the body solution $\mathbf{Y}^{n+1,i}$ at time level n+1 is known ($i = 0$ refers to initial guess at time level $n+1$).

Step 3: March the flow equations to determine the fluid force $\mathbf{F}^{n+1,i+1}$ and torque $\boldsymbol{\tau}^{n+1,i+1}$ and evaluate $\boldsymbol{\zeta}^{n+1,i+1}$.

Step 4: Evaluate $\mathbf{Y}^{n+1,i+1}$ by (23).

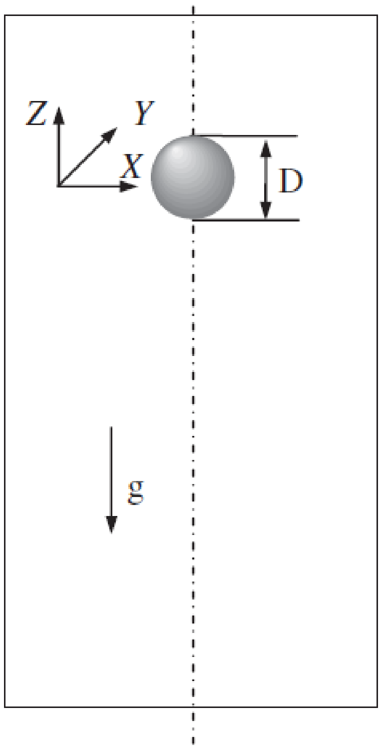
Step 5: Check for convergence: If

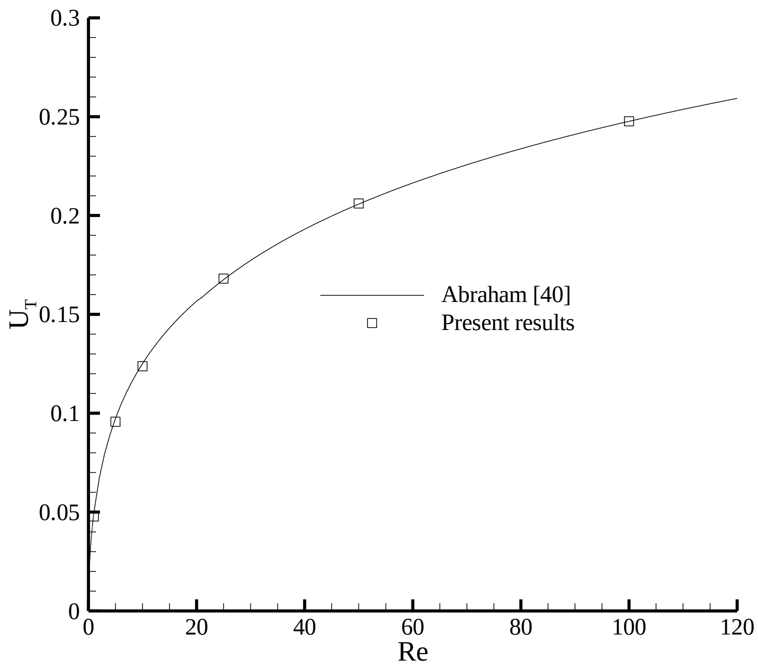
$\|\mathbf{Y}^{n+1,i+1} - \mathbf{Y}^{n+1,i}\| > \varepsilon$ go to Step 2 with $\mathbf{Y}^{n+1,i+1}$.

$\|\mathbf{Y}^{n+1,i+1} - \mathbf{Y}^{n+1,i}\| < \varepsilon$ go to Step 1 with $\mathbf{Y}^{n+1}$.

The above amounts to a fixed-point iterative scheme: $\Gamma(t) = S(F(\Gamma(t)))$ where S and F denote the body dynamic solver and flow solver respectively.

Test results: Falling spheres in an infinite domain at terminal state*





Comparison of terminal velocity of a free falling sphere at different Re.

[40] Abraham F. *Physics of fluids* 1970; **13**: 2194-2195. (Empirical)

Sphere with ρ_s/ρ=1.05. Surface nodes 2966 plus 4 layers of meshfree nodes. Computational domain: 131x131x381 nodes, central region 2Dx2Dx12D with Δ=0.04D.

		Re 50	*Re* 100
Falling sphere	Present results (in body frame)	0.396	0.867
Stationary sphere	Present results	0.402	0.864
	Taneda *et al.* [41]	0.475	0.903
	Johnson and Patel [42]	0.407	0.884
	Tomboulides and Orszag [43]	0.398	0.871

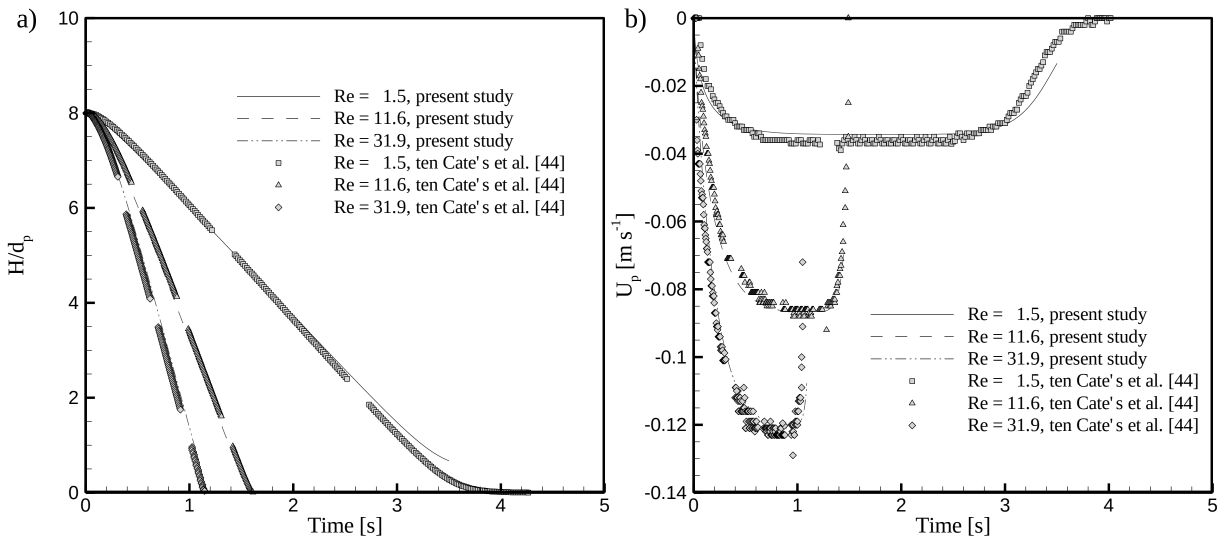
Length of the re-circulating wakes from present FSI-based and stationary sphere simulations.

*Yu P. Yeo K.S., Shyam Sundar D.,Ang S.J. 2011 Intl J. Num Meth. Engrg. **88**, 385-408.

Test results: Sedimentation of a sphere in bounded domain

Physical Domain size:
0.1x0.1x0.16m

Computational domain:
101x101x248 nodes with central region 2Dx2Dx9D with Δ=0.04D.
Sphere as before.



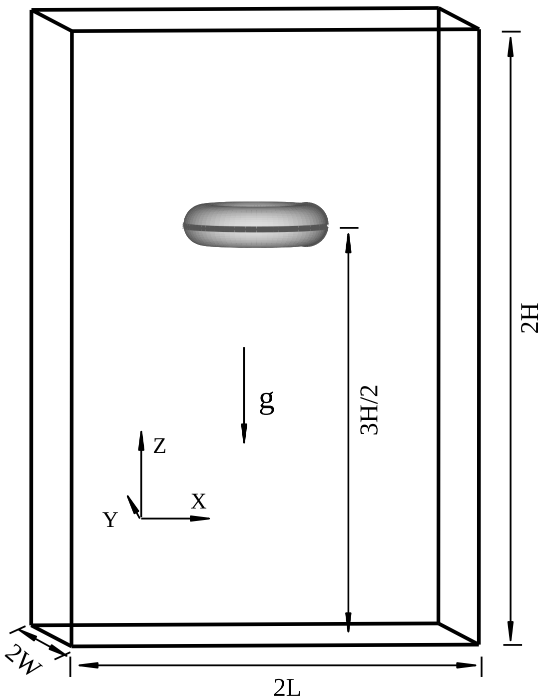
(a) Trajectory to the bottom of box.
(b) Sedimentation velocity with time.

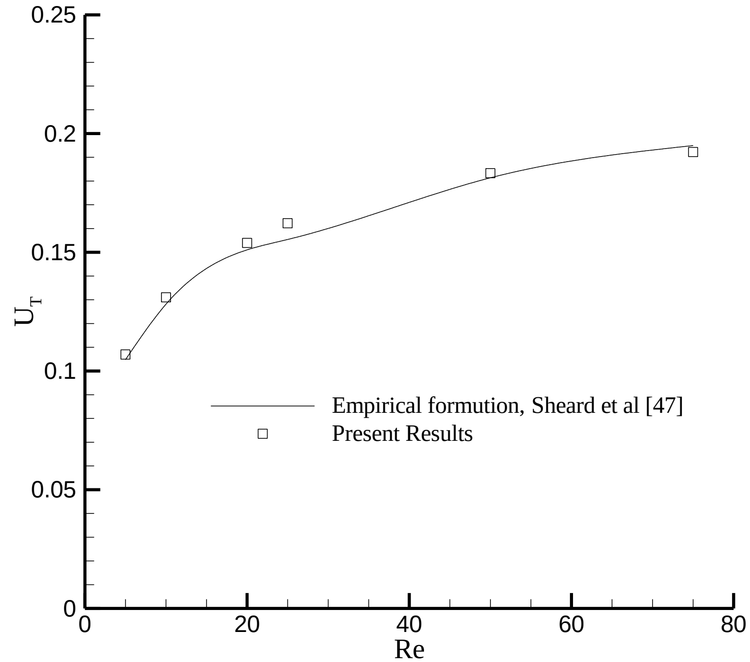
Table III. Comparison of the normalized maximum sedimentation velocity u_{max}/u_T, where u_T is the theoretical steady-state velocity of a freely falling sphere in an infinite medium.

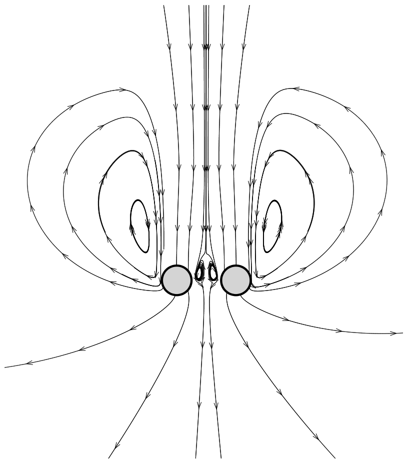
Case no.	ρ_f (kg/m^3)	μ_f (N s/m^2)	u_T (m/s)	Re	u_{max}/u_T Present	Exp., ten Cate [44]	Num., ten Cate [44]
1	970	373	0.038	1.5	0.904	0.947	0.894
2	965	212	0.060	4.1	0.955	0.953	0.950
3	962	113	0.091	11.6	0.957	0.959	0.955
4	960	58	0.128	31.9	0.945	0.955	0.947

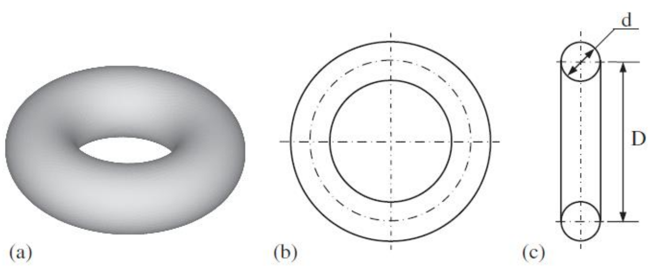
Agreement with experiments better at higher Re.

Test results: Falling toroidal ring in an infinite fluid domain









Terminal velocity for a free falling torus ring at different Re.

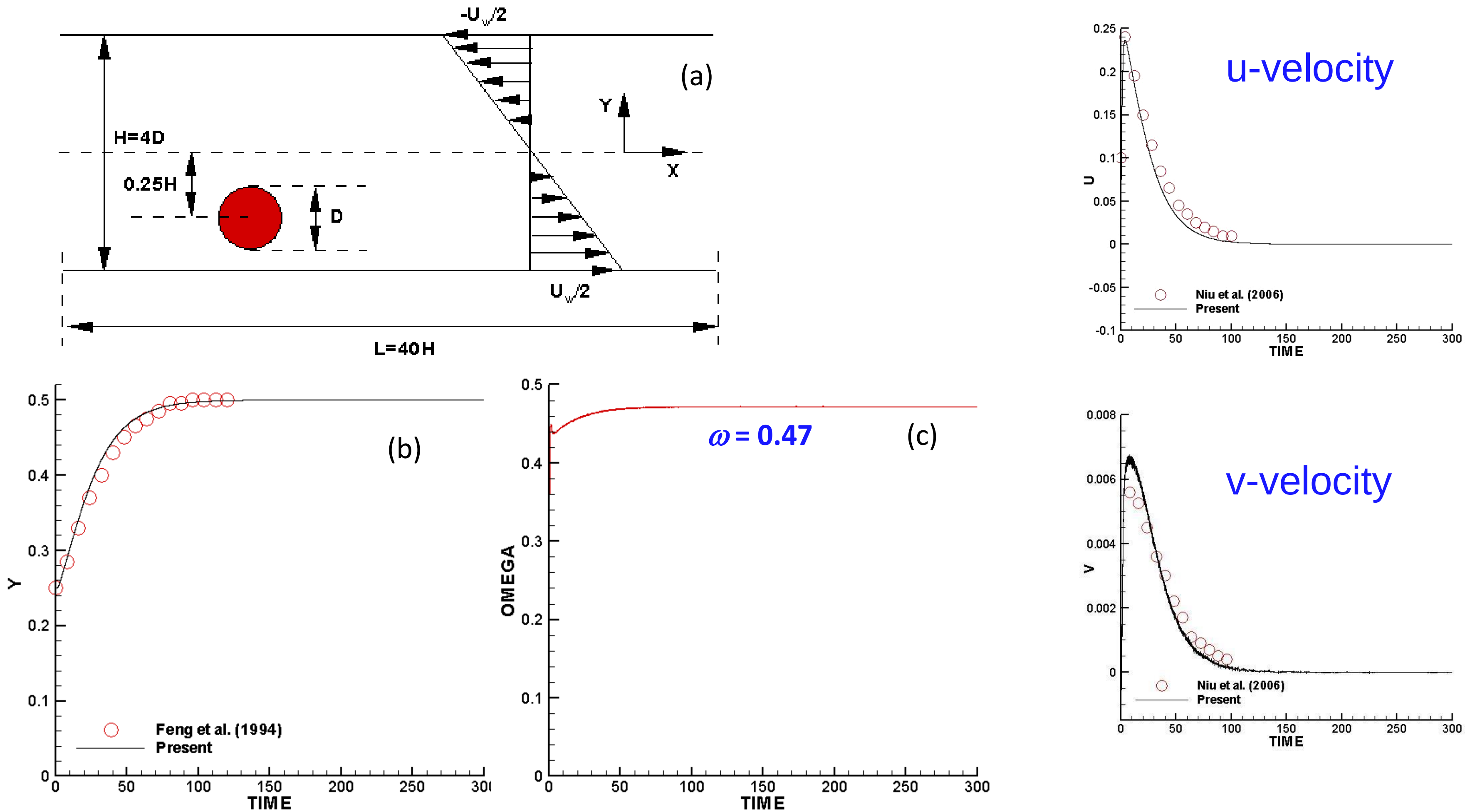
[47] Sheard GJ, Hourigan K, Thompson MC. *J Fluid Mech.* 2005; **526**: 257-275.

Flow in global frame at Re=50.

Computational domain: L=W=15d, H=30d. 181x181x681 nodes. Central region 6dx6dx6d with Δ=0.06d.

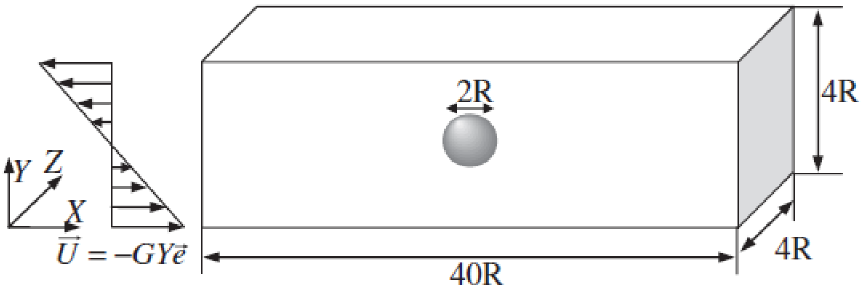
Toroidal surface 5420 nodes.

Test results: A freely rotating cylinder (neutrally buoyant) in simple shear flow

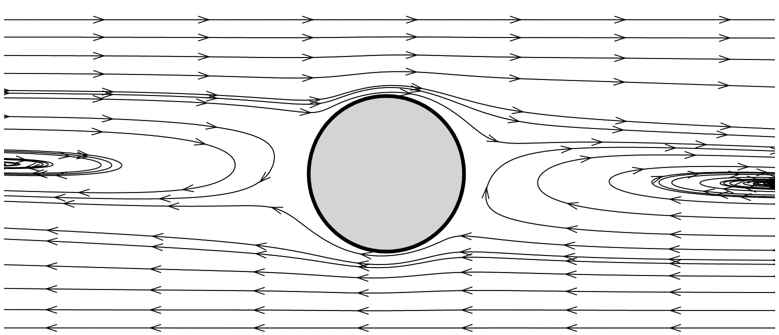


(a) Schematics of computational domain showing the initial position of the neutrally buoyant cylinder in a shear flow. (b) The *y*-displacement of the cylinder with time. (c) The angular velocity ω of the cylinder with time tending to ω = 0.47. Reynolds number is $\mathrm{Re} = U_w H / \upsilon = 40.$

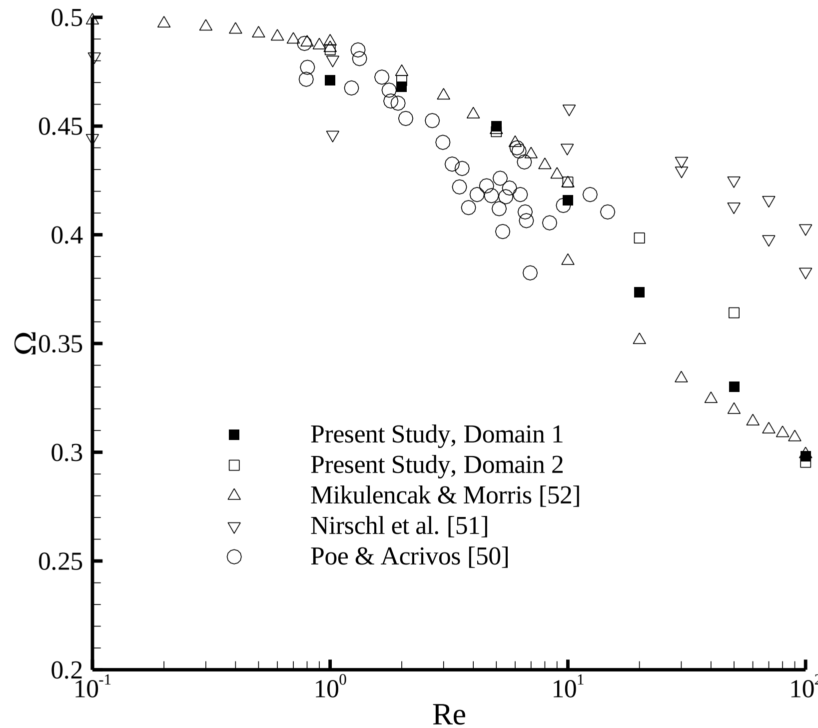
Test results: A freely rotating sphere (neutrally buoyant) in simple shear flow



Computational domain.
Domain 1: 211x81x81
Domain 2: 235x155x155. (40Rx12Rx12R)
Central region $(4R)^3$ with Δ=0.05R.
Sphere 7501 nodes.



Flow field around a freely rotating sphere in simple shear flow at Re=50.



Rotation rates of freely suspended spheres in a simple shear flow.

Sample applications : Free swimming and basic start manoeuvres*

Carangiform fish

Laterally compressed body.
BCF propulsion provided mainly by the tail / caudal fin.

Cyclic swimming (3D/2D)→

Guided Swimming → :

- Swimming straight
- Swimming to a target point

Swimming manoeuvres:

- Sharp turn during swimming →
- C-start →
- S-start →

Batoid (ray-type) fish

Vertically compressed body.
MFP propulsion provided mainly by enlarged side / pectoral fins.

Level swimming →

*Yeo K.S., Ang S.J., Shu C. *Computers & Fluids* 39 (2010) 403–430.

Sample applications: Flapping-wing flight with active control

Hovering flight →

Rectilinear flight of Fruit fly :

- Forward to hovering →
- Reverse →

High-order SVD-GFD scheme with compact support

Recent development - High-order SVD-GFD scheme with compact support*

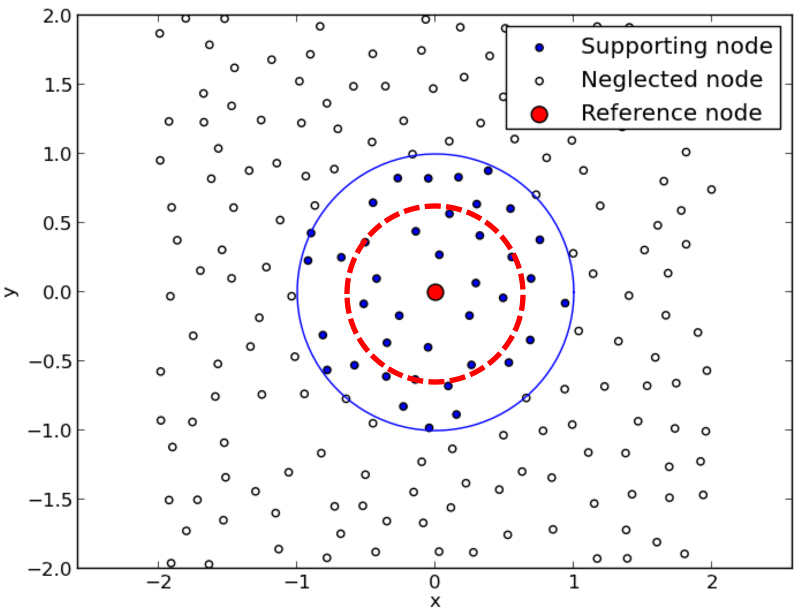
This represents a generalization of the SVD-GFD scheme.

Conventional meshfree schemes utilize the lowest-order solution data to reconstruct the solution and its derivatives in the neighborhood of the reference node. The present method increases the usage of available information at the nodes by interpolating over vector sets of nodal data.

This results in a solution reconstruction procedure that is both *high-order* and yet *compact* (small number of support nodes) in space.

High-order schemes are useful when the solution is known to be smooth.

More details of the scheme will be given when the work is published.



*Shyam Sundar D., Yeo K.S. (2013) A very high order meshless method with compact support. *Submitted for publication*.

Sample test results: Diffusion equation and Poisson equation in a square domain

Meshfree grid generated from a Cartesian square grid Δx by 0.3Δx random perturbation.

Degree of interpolation function $\mathscr{I}$: $p^I = p^Q + 2$

The number of support nodes is set at $N_s = 9$ in all cases.

❑ A Diffusion equation

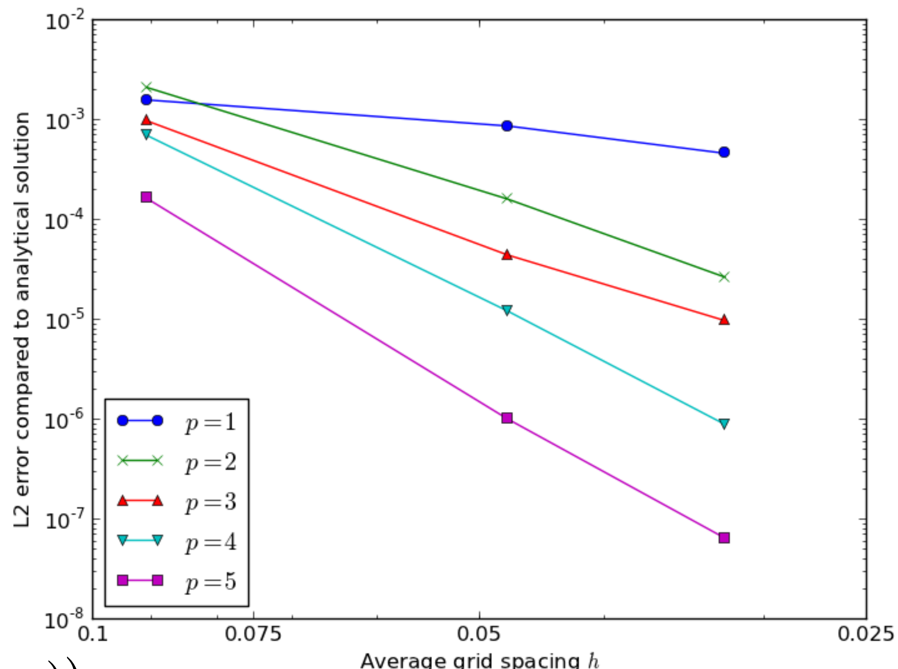
$$\frac{\partial Q}{\partial t} + D\nabla^2 Q = 0$$

In a square domain 1x1, with initial condition

$$Q(x, y, 0) = \sin(2\pi x) + \sin(2\pi y)$$

and exact solution

$$Q(x, y, t) = \exp(-4D\pi^2 t)(\sin(2\pi x) + \sin(2\pi y))$$



Solution convergence at time t=1

Sample test results (Contd.)

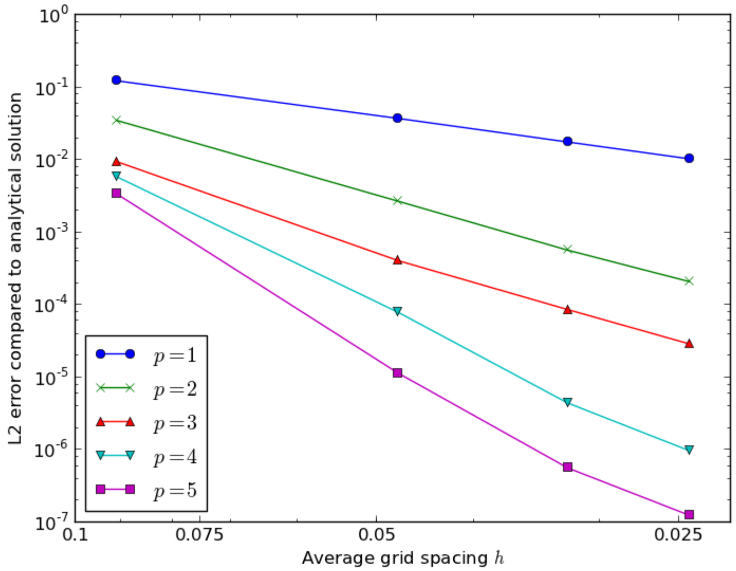
Poisson equation

$$\nabla^2 Q = -4\pi^2\left(x^2 + y^2\right)\sin(2\pi xy)$$

In a square domain 1x1 with exact solution:

$$Q(x, y) = \sin(2\pi xy).$$

Solved by Jacobi preconditioned GMRES is employed to converge the solution to 10^{-12}.



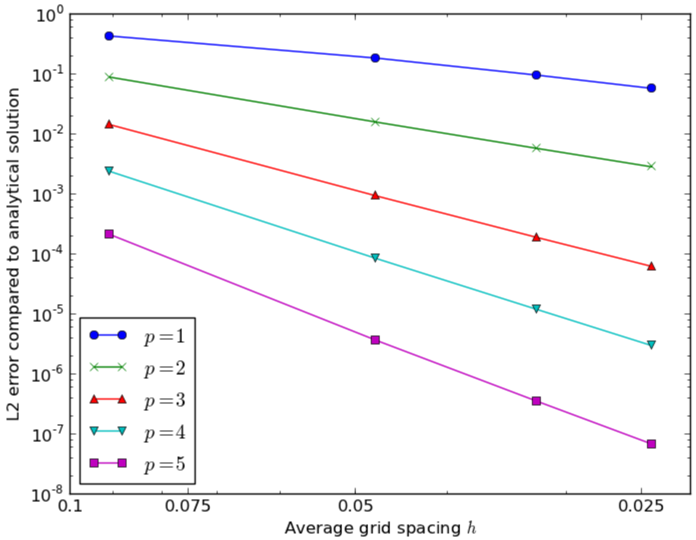
❑ A linear advection equation

$$\frac{\partial Q}{\partial t} + \vec{a} \cdot \nabla Q = 0$$

in a periodic square domain with initial condition:

$$Q = \sin\left[2\pi(x+y)\right]$$

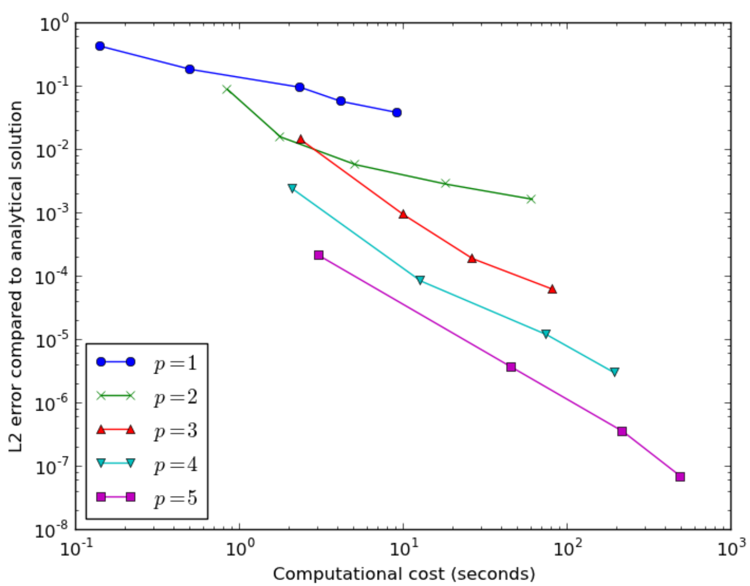
Time integration by Strong Stability Preserving Runge-Kutta (SSPRK).



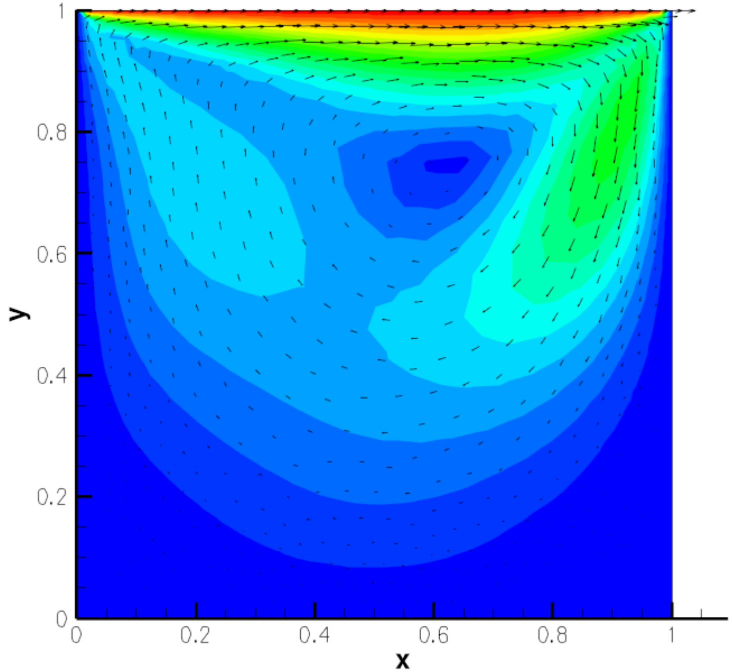
Computation for a linear advection equation to t=5.0.

For a given level of solution accuracy the high order schemes can yield a lower computational cost.

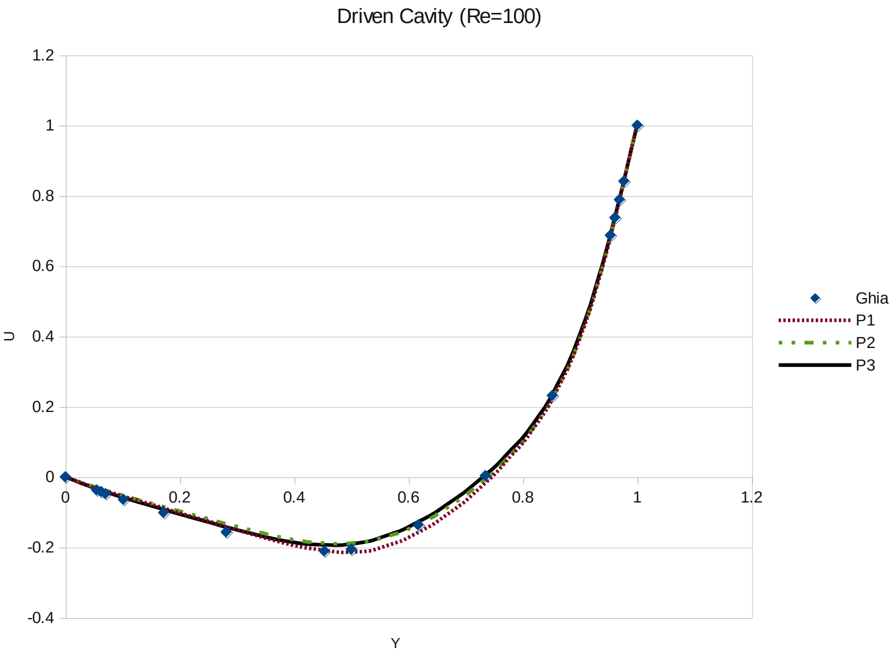
Eg. For solution accuracy of 10^{-4}, the 4^{th} order scheme costs 47.5 seconds while a 6^{th} order scheme costs only 4.8 seconds.



A preliminary application to incompressible flow problems



Driven Cavity flow (Re=100)
Meshfree grid points = 848.



U-velocity along x=0.5. Meshfree solution with 848 grid nodes versus Ghia et al. (129^2 = 16,641 nodes).

The scheme is still being developed for nonlinear problems such as fluid flow problems.

End

Thank you for your attention

Singular value decomposition (SVD)

- The SVD of a $m \times n$ matrix A is given by

$$A = U \Sigma V^T = \sum_{i=1}^{r} u_i \sigma_i v_i^T; \quad U = (u_1, \cdots, u_m), \quad V = (v_1, \cdots, v_n)$$

U and V are $m \times m$ and $n \times n$ *orthogonal* matrices whose columns are eigenvectors of AA^T and A^TA respectively.

Σ is a $m \times n$ diagonal matrix of rank r, whose r diagonal values are the square roots of non-zero eigenvalues of AA^T and A^TA.

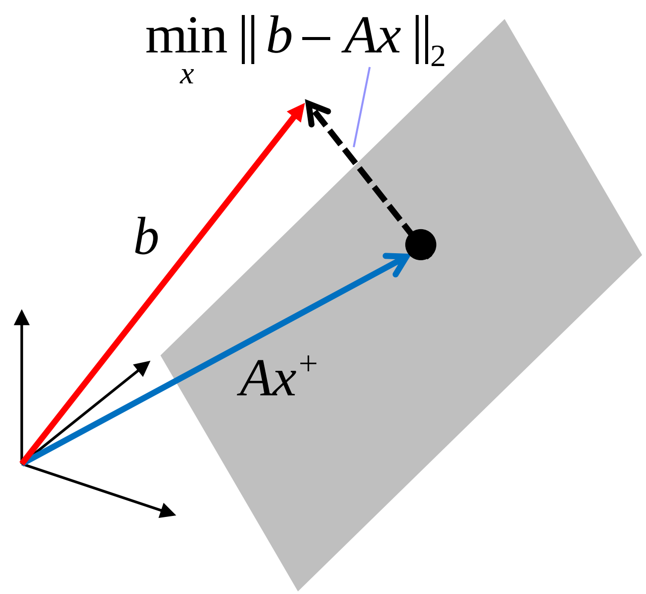
- The least square solution of

$$Ax = b$$

is *minimum length* (L_2-norm) solution

$$x^+ = A^+ b \quad \text{where} \quad A^+ = V \Sigma^+ U^T;$$

$$\Sigma^+ = Diag_{(n \times m)}\{1/\sigma_1, \cdots, 1/\sigma_r\}$$



Return

An artificial dissipation model

An second-order artificial dissipation term of the form* may be added to the momentum (NS) equations:

$$\left(d^2\mathbf{C}\right)\cdot\nabla^2\mathbf{u}$$

where $d=\sqrt{\Delta x\Delta y}$ and the dissipation coefficient $\mathbf{C}=(C_x,C_y)$ are tuned by the local pressure gradient ∇p in accordance with Caughey (1988)**:

$$C_x=\min\left\{\alpha_x\partial p/\partial x,\beta_x\right\},\quad C_y=\min\left\{\alpha_y\partial p/\partial y,\beta_y\right\}$$

The parameter β sets the maximum damping level. Typical values are $\alpha \approx 1$ and $\beta \approx 10$.

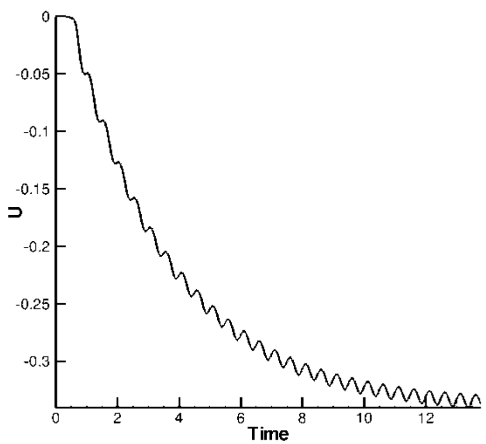
The scheme is easily used in irregular grid system, compared to the usual upwind schemes. The damping becomes negligible where the pressure gradient ∇p is small.

The scheme helps to stabilize computation in regions of strong flow separation, such as those encountered around sharp edges.

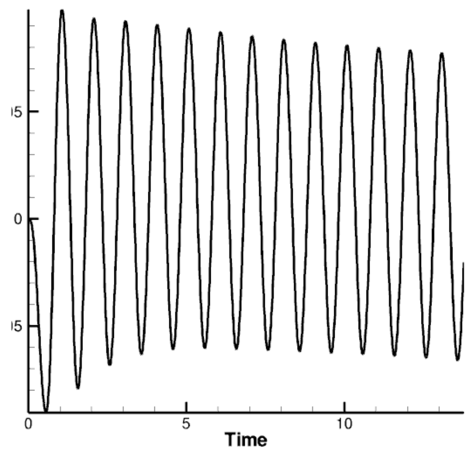
*Mendez B, Velazquez A. *Comput Methods Appl Mech Eng* (2004) **193**:825–48.
Caughey DA. *AIAA J* 1(988) **26:841–51.

Return

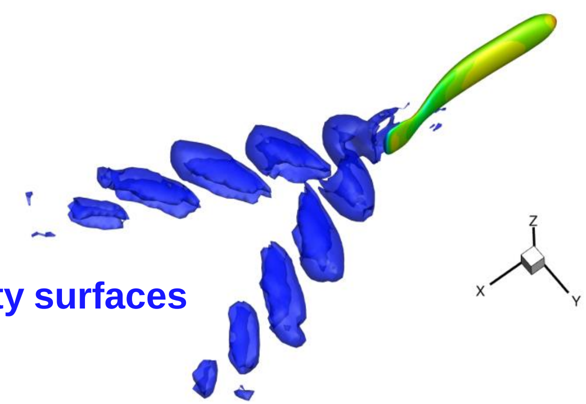
Cyclic Swimming*: $Re_f = 5000$



Swimming speed



Yaw angle



Iso-vorticity surfaces

(Animation)

(Animation)

*Yu P. Yeo K.S., Shyam Sundar D.,Ang S.J.
Intl J. Num Meth. Engrg. (2011) **88**, 385-408.

Return

Steady Cyclic Swimming in 2D

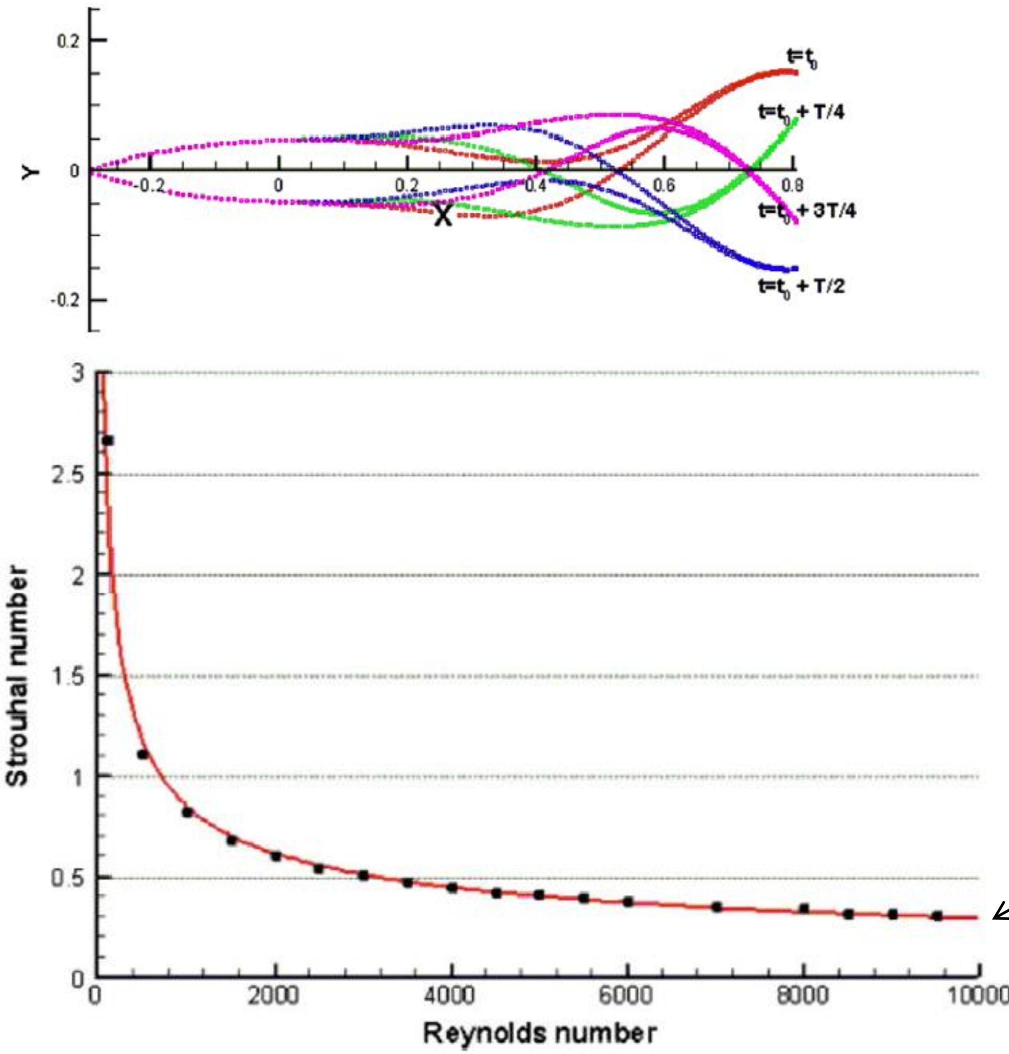
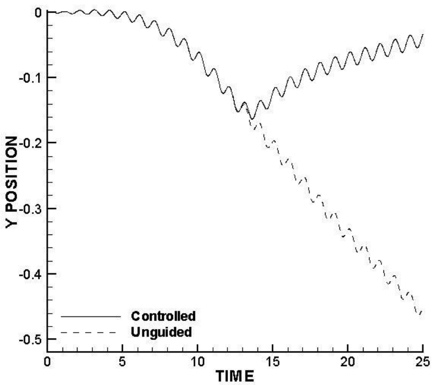


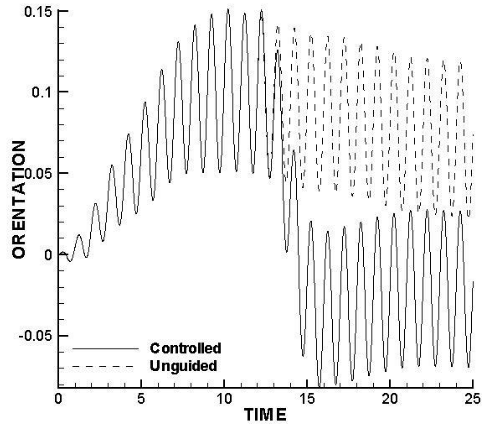
Fig. 21. Strouhal number St versus Reynolds number Re_f for steady swimming of the simulated fish.

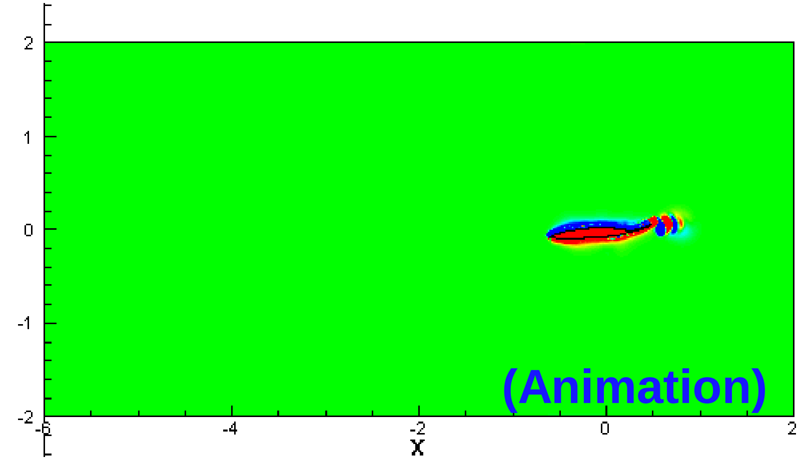
$\mathrm{St} = fA/U$ tends towards to 0.25<**0.3**<0.35 the optimal range for oscillating fins (Triantafyllou et al.1993)

Return

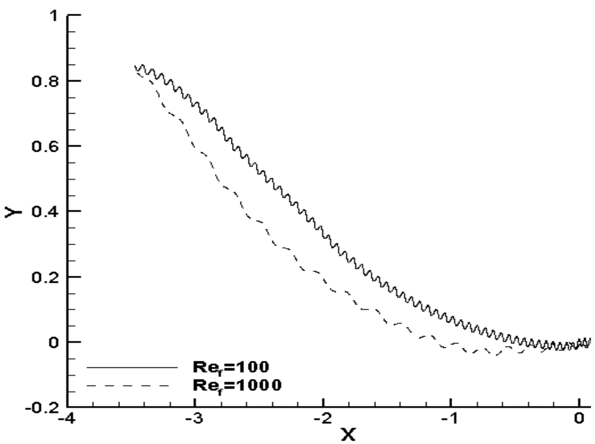
Guided Swimming*

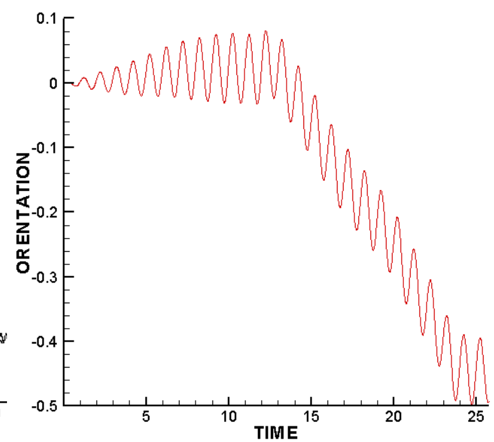


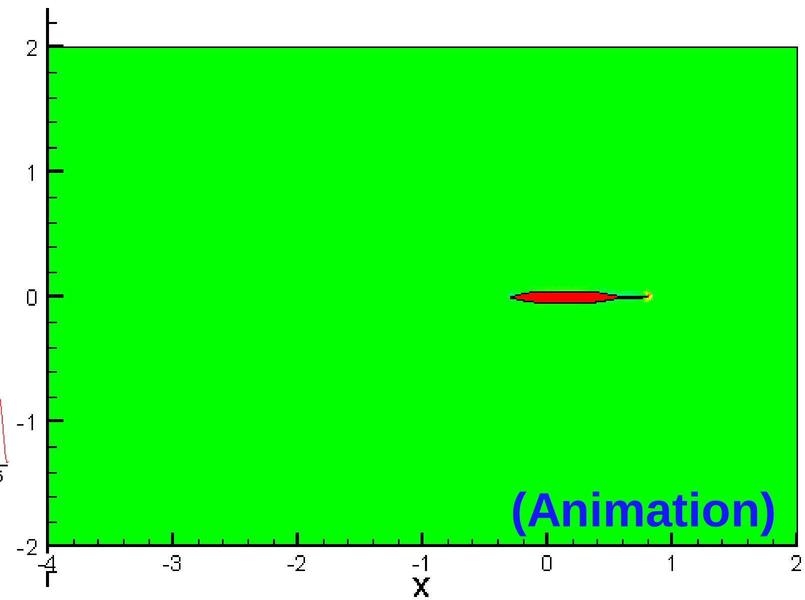




Swimming straight: Trajectory and Orientation of guided and unguided fish ($\mathrm{Re}_f = 2000$).







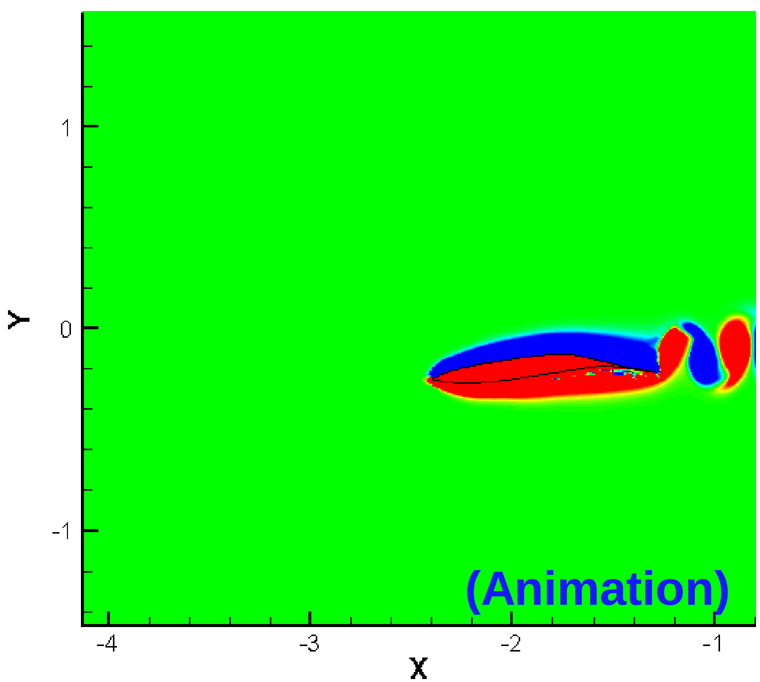
Swimming to target point (0,1): Trajectory and Orientation of guided and unguided fish ($\mathrm{Re}_f = 1000$).

*Yeo K.S., Ang S.J., Shu C. *Computers & Fluids* 39 (2010) 403–430.

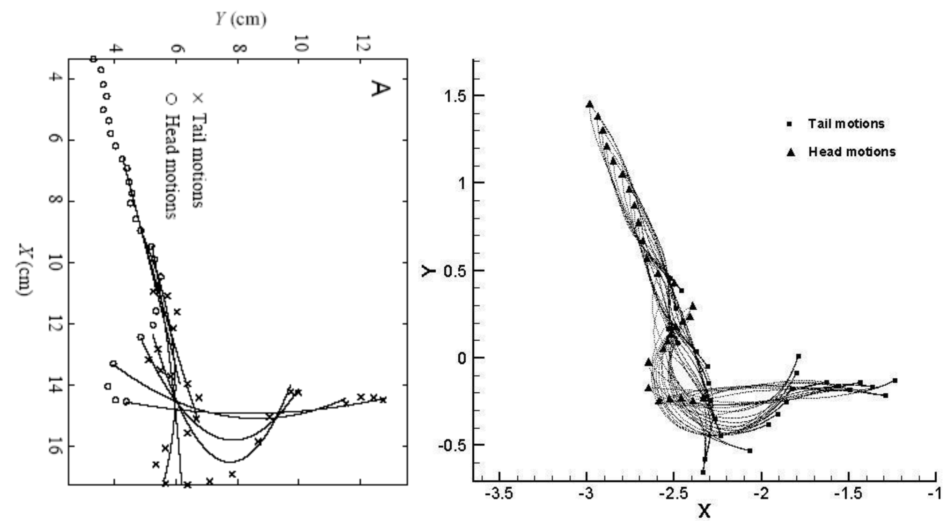
Return

Swimming manoeuvres:

Sharp turn during swimming at Re_f =2000.



The typical BCF fish is able to make a turn in a volume that is the order of its length.



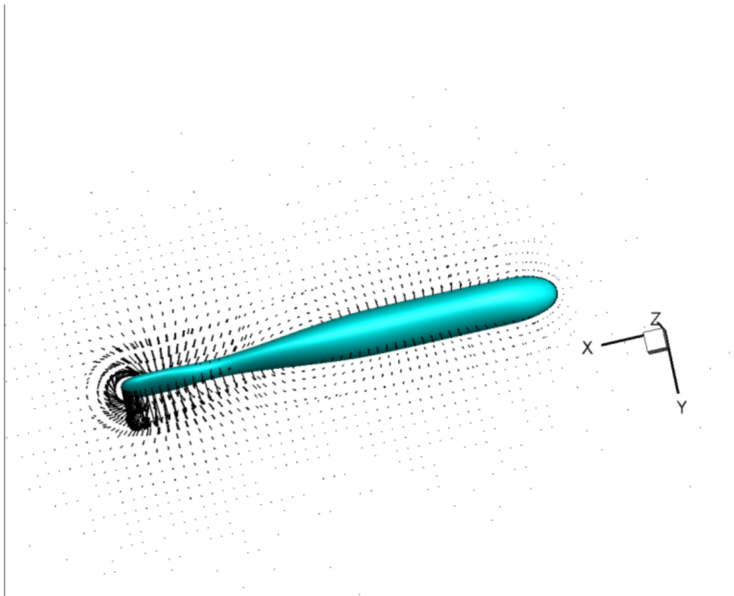
Trajectory of fish turning: (left) live fish from Wolfgang et al. [36] turning through 72-75 deg., (right) present simulation of fish turning through 70 deg. at Re_f =2000.

Return

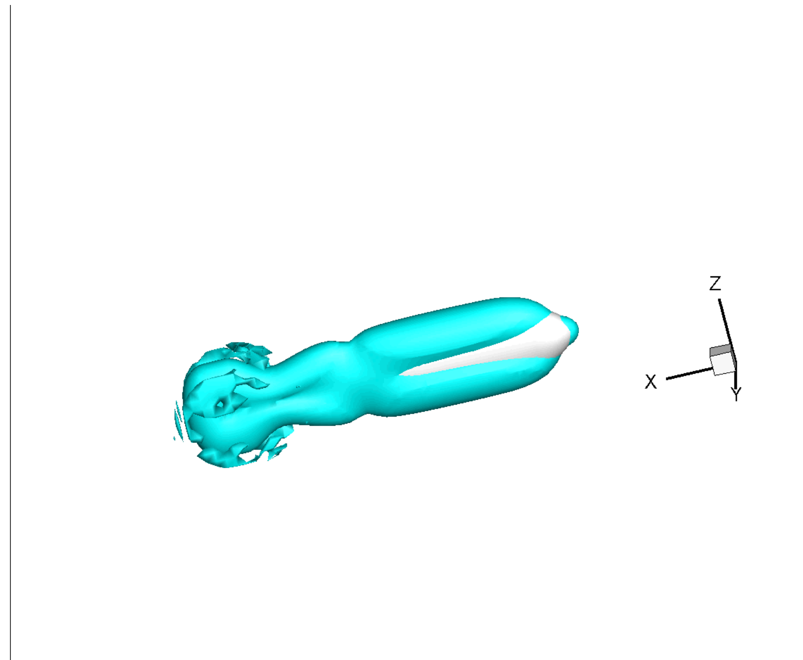
C-Start: $Re_f \approx 2500$

The fish bends its body from rest into a C shape and straightens its body – may be executed as an attack or escape response.

Fish initially aligned along the x-axis.



(Animation)



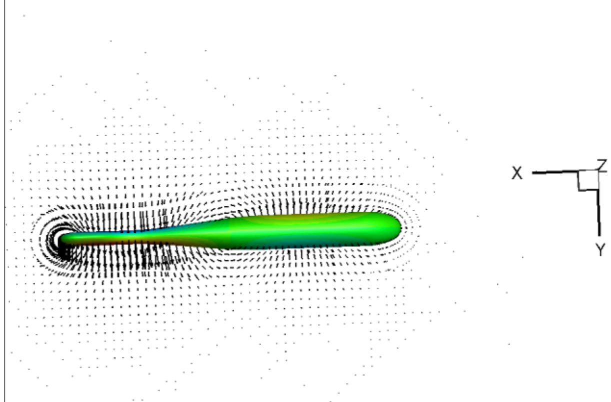
(Animation)

Return

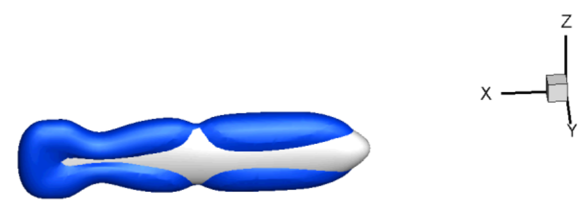
S-Start: $Re_f \approx 2500$

The fish bends its body from rest into an S shape and straighten its body – may be executed as an escape response.

Fish initially aligned along x-axis.



(Animation)

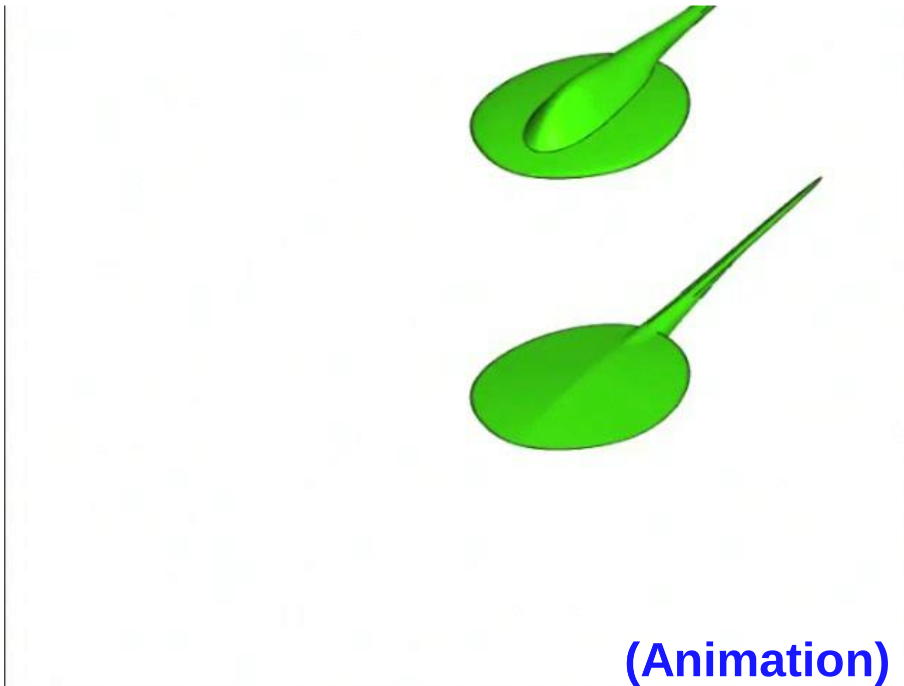


(Animation)

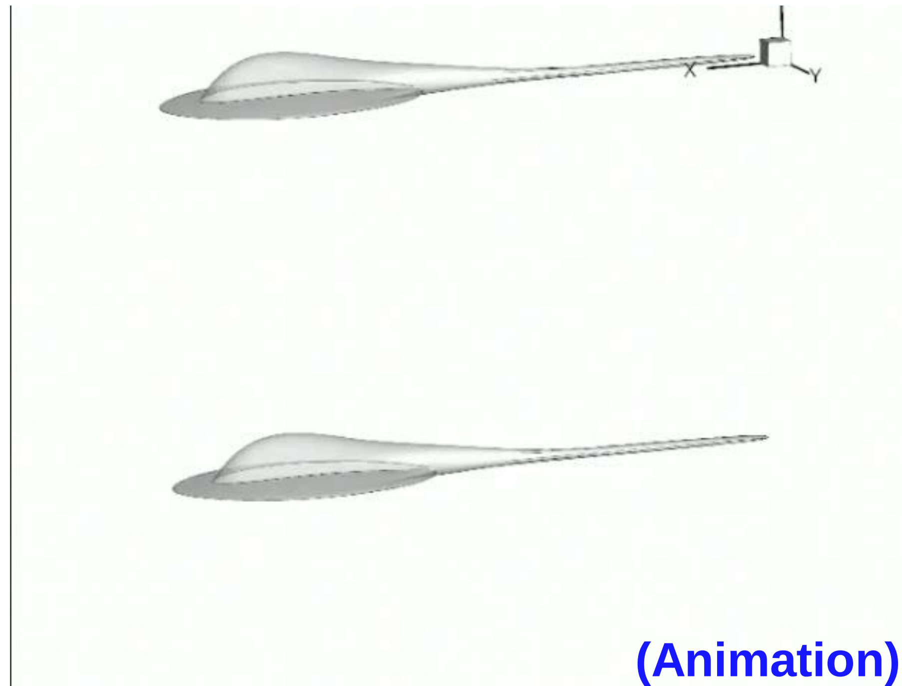
Return

Batoid (Ray-type) Fish

Rajiiform ray swimming at $\mathrm{Re}_f = 5000$



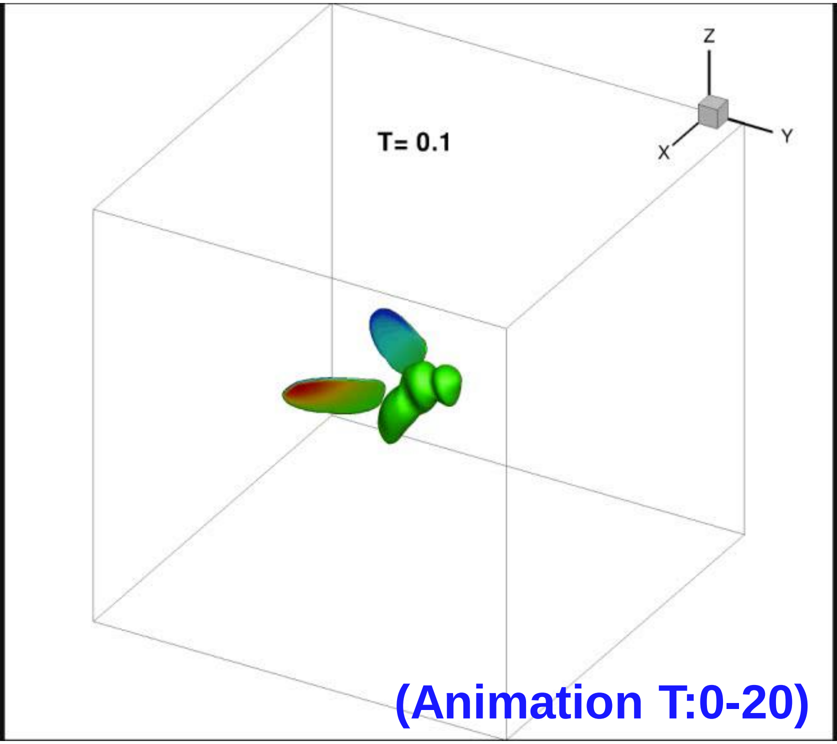
Level swimming showing pressure distribution on top and bottom surfaces

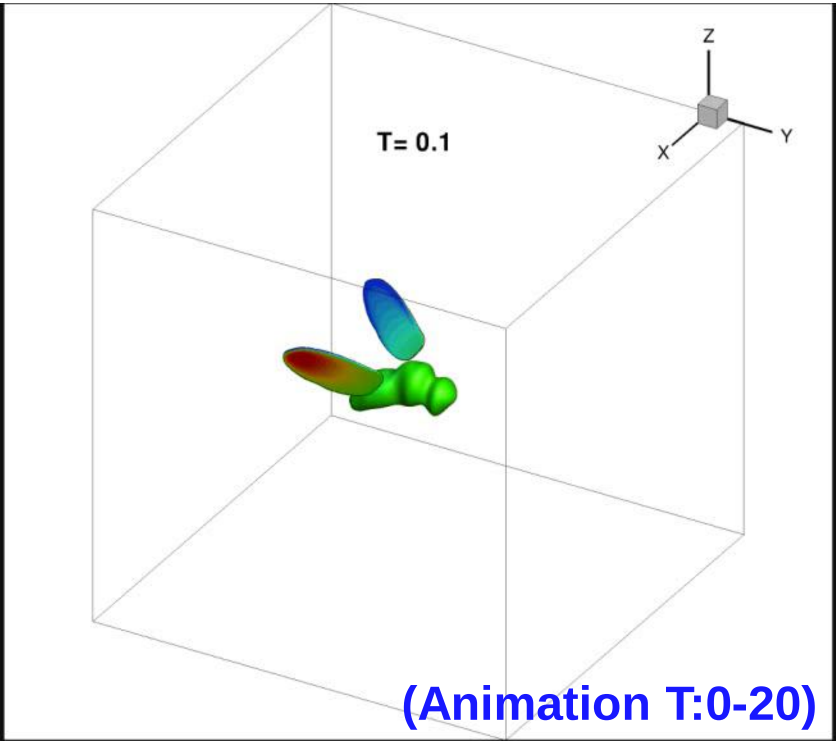


Moving frame figure: y-vorticity

Return

Flapping wings – Active Hovering flight at $Re_f \approx 150$



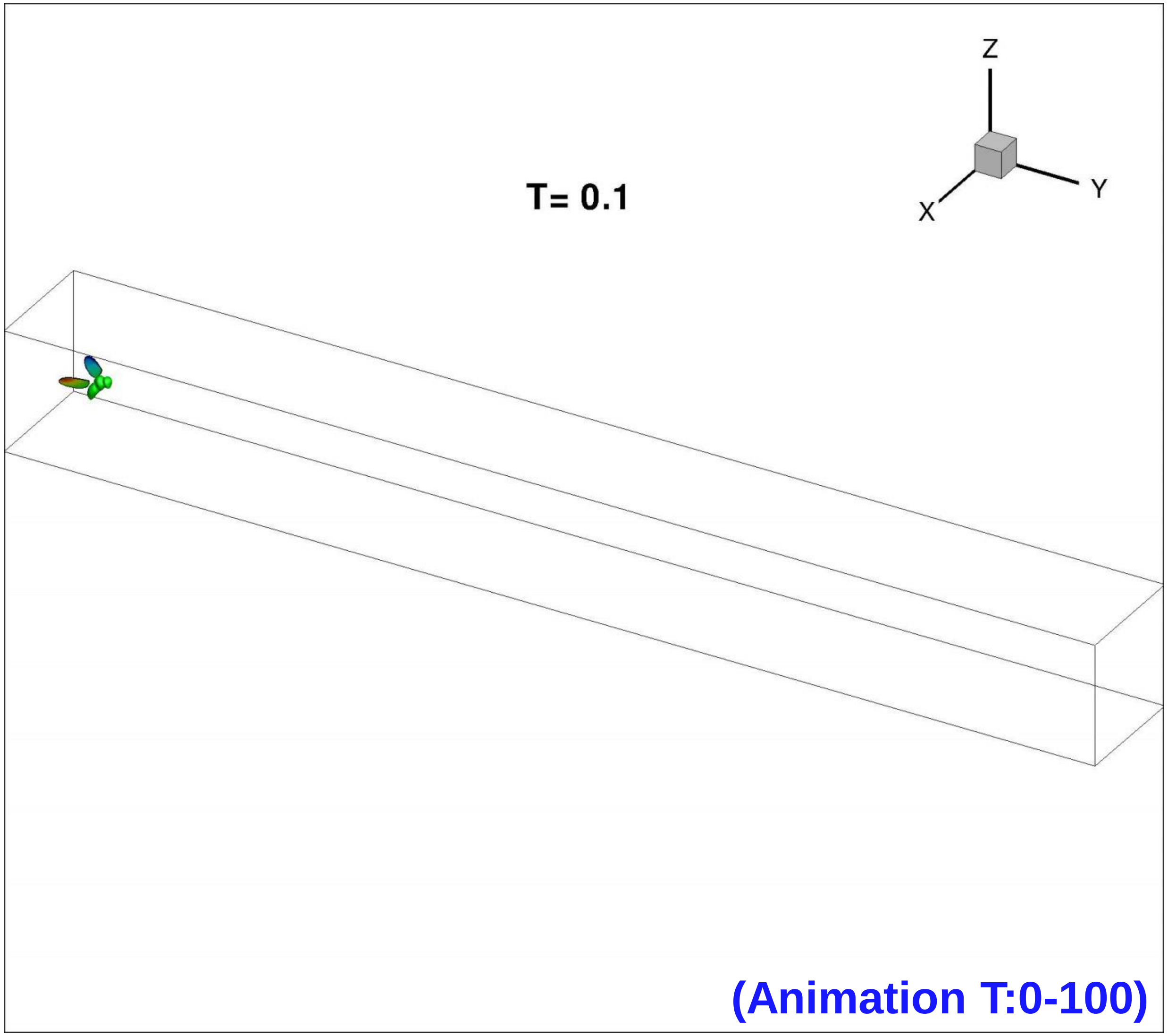


Normal hovering with nearly horizontal stroke plane.

Inclined stroke plane hovering with nearly horizontal body posture.

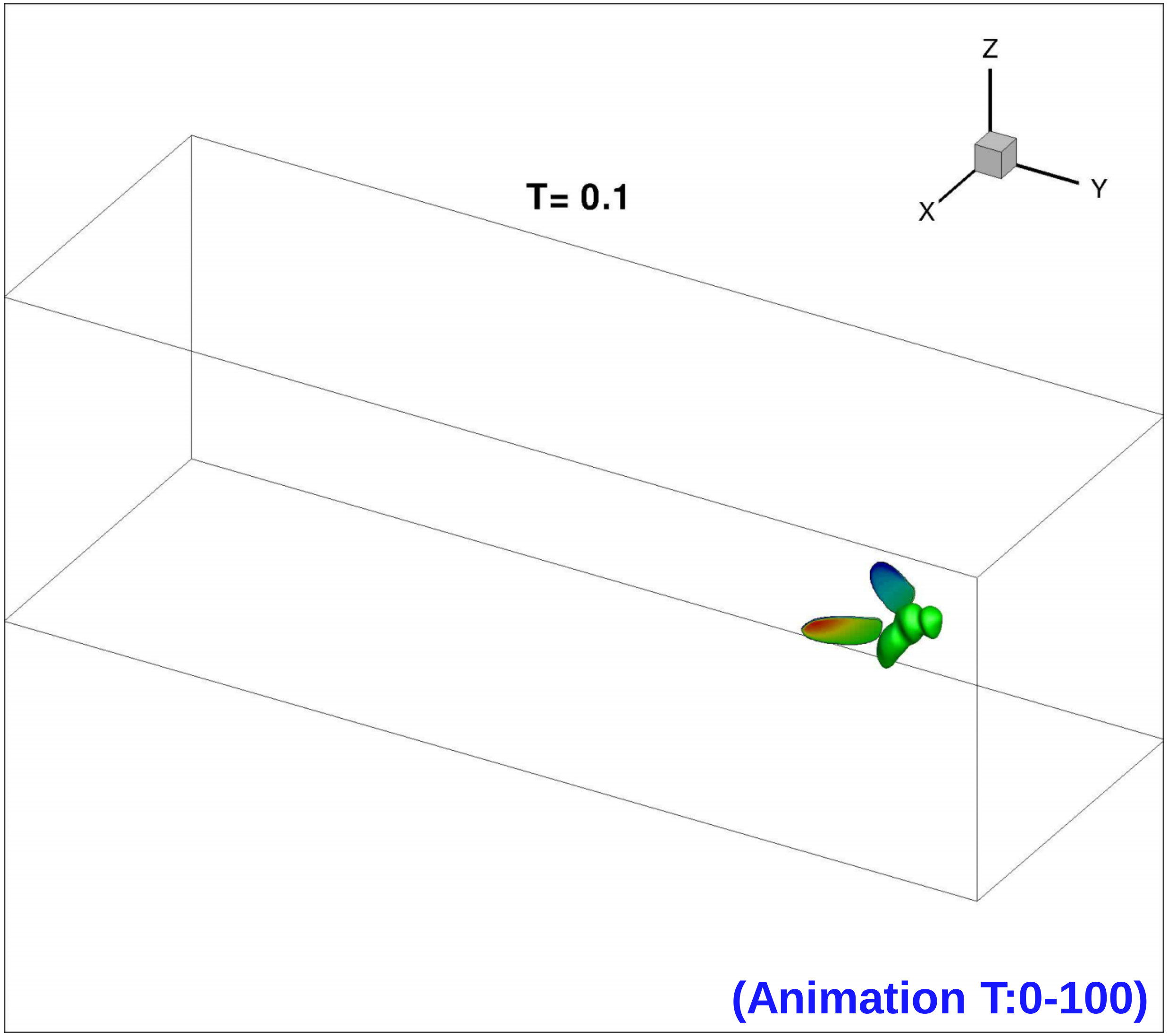
Return

Flapping wings – Accelerating-Decelerating flight ($Re_f \approx 150$)



Return

Flapping wings – Reverse flight at 5 cm/s ($Re_f \approx 150$)



Return