Modeling and simulation of deformable bodies in low-Reynolds flows

Luca Berti, Francois Der Hovsepian

August 23, 2018

Outline

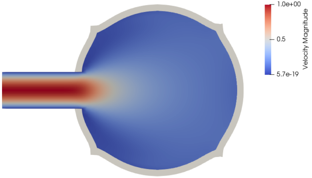
Fluid-Structure Interaction

Fluid inside or surrounding a solid.
Fluid flow $\rightleftarrows$ Solid structure

- Fluid flow $\rightarrow$ deforms the solid structure
- Structure deformation $\rightarrow$ impacts the flow

Examples:

- Airflow around an aircraft wing
- Balloon inflating
- Wind action on a yacht sail



Fluid equations

Incompressible Navier-Stokes equations

$$\rho\Big(\frac{\partial u}{\partial t} + (u \cdot \nabla)u\Big) = -\mu\Delta u + \nabla p$$
$$\nabla \cdot u = 0$$

1. Balance equation for linear momentum and conservation of mass equation
2. Linear dependence of stresses on strain rate
3. Non-linearity arising from the convective term
4. Neglecting inertia leads to Stokes equations

Stokes equations

$$\mu\Delta u - \nabla p = 0$$
$$\nabla \cdot u = 0$$

Solid equations

Elasticity equations

$$\rho \frac{\partial^2 \eta}{\partial t^2} - \nabla \cdot (F\Sigma) = f_s$$

1. Balance equations
2. Formulated in terms of the displacement $\eta(t, x)$ between reference and current configuration
3. Non-linearity depends on the elasticity model, encoded in Σ

Continuous coupling conditions

Coupling conditions on fluid-solid interface

$$\frac{\partial \eta}{\partial t} = u$$

$$\mathbb{T} n_f + F \Sigma n_s = 0$$

1. Velocity continuity
2. Normal stress continuity

Warning

Fluid and elasticity equations are not defined on the same reference frame (Eulerian vs. Lagrangian).

Discrete coupling conditions

There are several techniques to discretize the continuous coupling conditions:

1. Dirichlet-Neumann: velocity continuity (fluid side) - stress continuity (solid side)
2. Robin-Robin: combination of Dirichlet and Neumann conditions (fluid & solid sides)
3. . . .

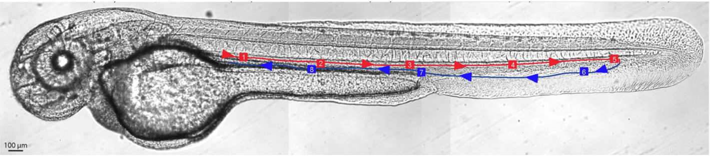
The choice depends on the fluid/solid system and solution technique.

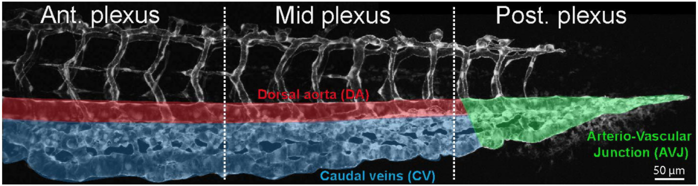
FEEL++

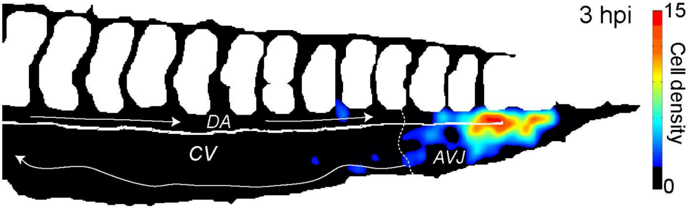
Finite Element Embedded Library in C++

- Open source library → github.com/feelpp
- Galerkin methods
- Domain Specific Embedded Language (DSEL) in C++
- Scaling : from laptops to supercomputers
- Easy deployment → Docker, Singularity
- (Multi)Physics toolboxes: Heat Transfer, Aerothermics, CFD, CSM, FSI, Maxwell
- docs.feelpp.org
- YouTube → Feel++ channel

Project: Cells under flow in the zebrafish



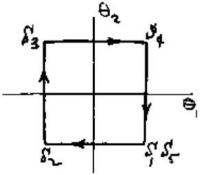




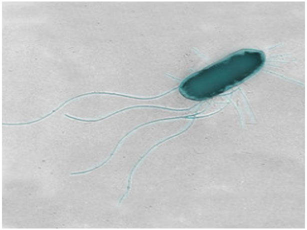
“Life at low Reynolds number” - Purcell (1976)

At the microscale the Reynolds number is small and inertia forces play no role in the motion of microorganisms.

- Swimmers that perform reciprocal strokes are not able to move.

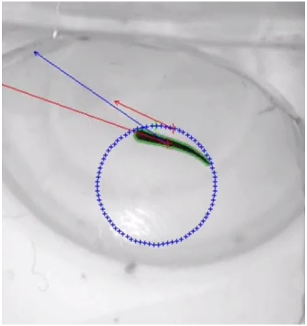


- Microorganisms have adopted non-reciprocal swimming strategies.



Magneto-elastic microswimmer at ISIR - UPMC

Composite microrobot which can perform non-reciprocal strokes.



- Viscous fluid
- Elastic body
- Magnetic head
- Driven by external magnetic field

Micro-robots can be applied for targeted drug delivery and non-invasive surgical operations.

From Navier-Stokes to Stokes equations

Incompressible Navier-Stokes equations

$$\rho\left(\frac{\partial u}{\partial t} + (u \cdot \nabla)u\right) = -\mu\Delta u + \nabla p$$
$$\nabla \cdot u = 0$$

1. Identify typical length (L), velocity (U), timescale (T).
2. Adimensionalize

$$Re\left(\frac{\partial u^*}{\partial t^*} + (u^* \cdot \nabla)u^*\right) = -\Delta u^* + \nabla p^*$$
$$\nabla \cdot u^* = 0$$

where

$$Re = \frac{LU\rho}{\mu} \quad u^* = \frac{u}{U} \quad t^* = \frac{t}{LU} \quad p^* = \frac{pL}{\mu U}$$

From Navier-Stokes to Stokes equations

For microswimmers (spermatozoa, bacteria) in water

- $L \approx 10^{-6}\ m$
- $U \approx 30 \cdot 10^{-6}\ m/s$
- $\mu/\rho \approx 10^{-6}\ m^2/s$

Hence, $Re \approx 10^{-5}$

Stokes equations

$$\mu \Delta u - \nabla p = 0$$
$$\nabla \cdot u = 0$$

Rigid motion with imposed displacement

In the swimmer frame one can decompose the Stokes problem

$$
\begin{aligned}
\mu\Delta u - \nabla p = 0 \quad & \text{in } \Omega \setminus S \\
\nabla \cdot u = 0 \quad & \text{in } \Omega \setminus S \\
u = \omega \times x + v + w_d \quad & \text{on } \partial S \\
u = 0 \quad & \text{on } \partial\Omega \setminus \partial S
\end{aligned}
$$

in 7 (3D) or 4 (2D) independent Stokes problems to extract rigid body motion.
This is possible because Stokes equations are linear in (u, p).

Stokes problem decomposition

Solve the Stokes problem with the following boundary conditions (in the swimmer frame):

1. $e_1 \times x$ produces solution u_1 (3D)
2. $e_2 \times x$ produces solution u_2 (3D)
3. $e_3 \times x$ produces solution u_3 (3D) - u_1 (2D)
4. e_1 produces solution u_4 (3D) - u_2 (2D)
5. e_2 produces solution u_5 (3D) - u_3 (2D)
6. e_3 produces solution u_6 (3D)
7. w_d produces solution u_7 (3D) - u_4 (2D)

$$u = \sum_{i=1}^{3} \omega_i u_i + \sum_{i=4}^{6} v_i u_i + u_7 \qquad p = \sum_{i=1}^{3} \omega_i p_i + \sum_{i=4}^{6} v_i p_i + p_7$$

Newton equations

Newtonian fluid - Stress tensor

$$T(u,p) = -p\mathbb{I} + 2\mu \frac{\nabla u + \nabla u^T}{2}$$

Newton equations

$$\int_{\partial S_t} T(u,p) = 0$$

$$\int_{\partial S_t} x \times T(u,p) = 0$$

Stokes problem decomposition (3D)

$$Mz + N = 0 \text{ where } z = (\omega, v)$$

$$M_{ij} = \begin{cases} \displaystyle\int_{\partial S_t} (x \times e_i) \cdot T(u_j, p_j) n \, \mathrm{d}x & 1 \leq i \leq 3, 1 \leq j \leq 6 \\ \displaystyle\int_{\partial S_t} e_{i-3} \cdot T(u_j, p_j) n \, \mathrm{d}x & 4 \leq i \leq 6, 1 \leq j \leq 6 \end{cases}$$

$$N_j = \begin{cases} \displaystyle\int_{\partial S_t} (x \times e_i) \cdot T(u_7, p_7) n \, \mathrm{d}x & 1 \leq i \leq 3 \\ \displaystyle\int_{\partial S_t} e_{i-3} \cdot T(u_7, p_7) n \, \mathrm{d}x & 4 \leq i \leq 6 \end{cases}$$

Stokes problem decomposition (2D)

$$Mz + N = 0 \text{ where } z = (\omega, v)$$

$$M_{ij} = \begin{cases} \displaystyle\int_{\partial S_t} (x \times e_3) \cdot T(u_j, p_j) n \, \mathrm{d}x & 1 \leq j \leq 3 \\ \displaystyle\int_{\partial S_t} e_i \cdot T(u_j, p_j) n \, \mathrm{d}x & 2 \leq i \leq 3, 1 \leq j \leq 3 \end{cases}$$

$$N_j = \begin{cases} \displaystyle\int_{\partial S_t} (x \times e_3) \cdot T(u_4, p_4) n \, \mathrm{d}x & \\ \displaystyle\int_{\partial S_t} e_j \cdot T(u_4, p_4) n \, \mathrm{d}x & 2 \leq j \leq 3 \end{cases}$$

Self-propulsion constraints

The absence of inertial effects makes the problem independent of mass.

For massless swimmers it's necessary to account for these self-propulsion constraints on the deformation.

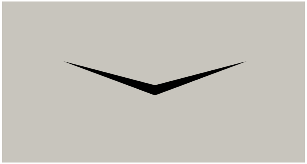
Self-propulsion constraints

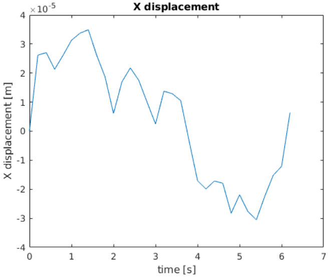
$$\int_{\partial S_t} \partial_t \eta_d(t,x)\,\mathrm{d}x = 0$$

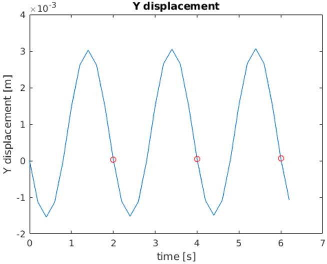
$$\int_{\partial S_t} \partial_t \eta_d(t,x) \times \eta_d(t,x)\,\mathrm{d}x = 0$$

Scallop theorem

A body which deforms and follows the same deformation path back in time will not show a net motion.
An example of such body is the scallop (Coquille de Saint Jacques), which moves by repeatedly opening and closing its valves.

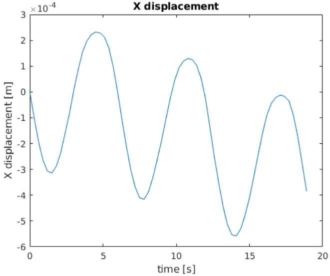




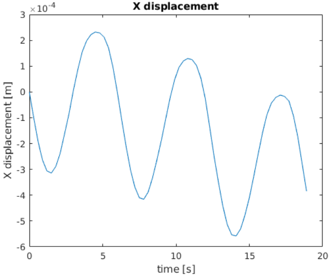


Influence of the viscosity coefficient

In the absence of inertia forces, the rigid body motion computed this way is independent of the viscosity coefficient.

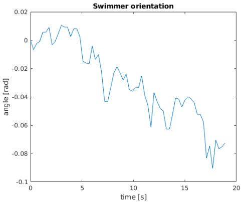


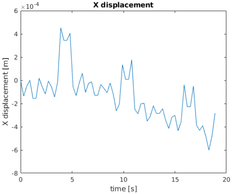
$\mu = 1.0$

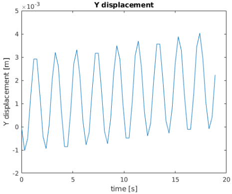


$\mu = 5.0$

It is experimentally observed that micro-swimmers are attired by boundaries. In the case of a reciprocal swimmer, the proximity to the boundary produces a net motion towards it.
In our case the scallop approaches the left corner of the box and rotates.





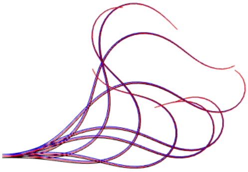


Spermatozoon

The previous method could be applied to spermatozoa by prescribing the velocity w_d coming from deformation. A formula coming from experiments is proposed. The tangent angle is

$$\psi(s,t) = K_0 s + 2A_0 s \cos(\omega t - \frac{2\pi s}{\lambda})$$

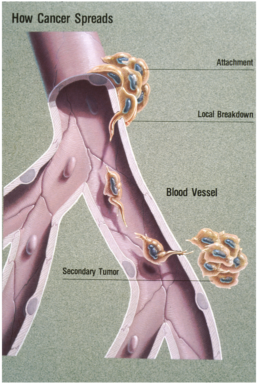
- Mean curvature K_0
- Varying amplitude $A_0 s$
- Travelling wave λ, ω



Perspectives and improvements

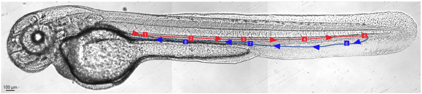
- Parallelize the solution of the Stokes subproblems
- Feed displacement velocity from a FSI code
- Couple the system with equations for magnetic field
- Improve meshing and remeshing strategies

Cancer cell metastasis



Cancer cells metastasis: cells going from primary to secondary site in the host body, using the vascular system.

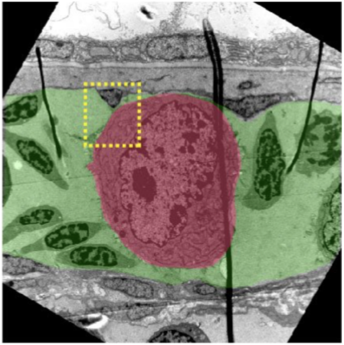
The zebrafish embryo: a model organism



Zebrafish (ZF) features:

- Easy to observe: thin and transparent
- Cheap to maintain
- Endless supply: Hundreds of offspring weekly
- Extremely fast development (1 day ZF $\rightarrow$ 1 month Human)
- Complete genome sequence is known: 70% of genes shared with humans

Two kinds of cells



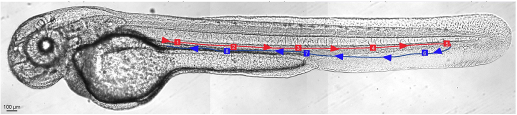
Red Blood Cells (RBC) in... green

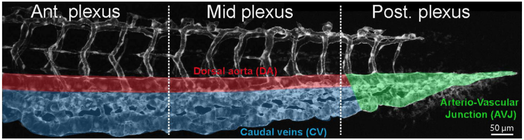
- Elliptical (in embryo: $\approx 5 \times 2.5 \mu m$)

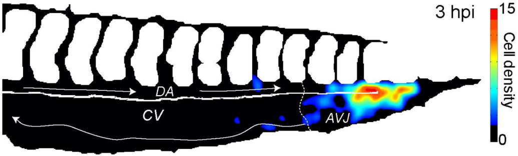
Circulating Tumor Cell (CTC or TC) in red

- Softer
- Bigger (radius $\approx 5 \mu m$)

Region of interest : Caudal plexus







Tumor cells preferentially stop in the caudal plexus.

Goals:

- Model and simulate cells in the zebrafish
- Model cell-cell and cell-wall contacts

The plan:

- Create a 2D mesh of the ZF
- Model and simulate a single cell in the flow using the level set method
- Extend to multiple cells and manage contacts

Blood flow - Stokes model

Blood in a ZF blood vessel :

- incompressible newtonian fluid
- density $\rho = 1000 kg/m^3$, dynamic viscosity $\mu = 2 \times 10^{-3} Pa.s$
- mean velocity $U = 1 \times 10^{-3} m/s$
- characteristic dimension $L = 20 \times 10^{-6} m$ (ZF blood vessel)
- Reynolds number $Re = \frac{\rho U L}{\mu} = 10^{-2}$

$\Rightarrow$ We use the Stokes model for incompressible newtonian fluids

$$\begin{cases} -\mu \Delta u + \nabla p = F \\ \nabla \cdot u = 0 \end{cases}$$

Where u is the velocity, p is the pressure and F are the external forces.

Cell model $\rightarrow$ vesicle: an inextensible membrane filled with fluid.
$\Rightarrow$ A cell in blood flow: an interface Γ separating two fluids.

inextensibility of the interface: $\nabla_s \cdot u = \nabla \cdot u - (\nabla u \cdot n) \cdot n = 0$ on Γ with n the outward normal vector.

The level set method $\rightarrow$ Track the interface implicitly.

The level set method

- define a level set function ϕ on the fluid domain Ω
- ϕ : signed distance to the interface (negative inside the cell)
- advect it using the fluid velocity field

$$\begin{cases} \dfrac{\partial \phi}{\partial t} + u \cdot \nabla \phi = 0 \\ \nabla \cdot u = 0 \end{cases}$$

Smoothed Heaviside function

$$H_\epsilon(\phi) = \begin{cases} 0, & \phi \leq -\epsilon \\ \frac{1}{2}\left(1 + \frac{\phi}{\epsilon} + \frac{sin(\frac{\pi\phi}{\epsilon})}{\pi}\right), & -\epsilon \leq \phi \leq \epsilon \\ 1, & \phi \geq \epsilon \end{cases}$$

→ define quantities in each fluid : $\rho(\phi(x)) = \rho_2 + (\rho_1 - \rho_2)H_\epsilon(\phi)$.

Smoothed delta function

$$\delta_\epsilon(\phi) = \begin{cases} 0, & \phi \leq -\epsilon \\ \frac{1}{2\epsilon}\left(1 + cos(\frac{\pi\phi}{\epsilon})\right), & -\epsilon \leq \phi \leq \epsilon \\ 0, & \phi \geq \epsilon \end{cases}$$

→ define quantities on interface : $\int_\Gamma 1 \approx \int_\Omega \delta_\epsilon(\phi)$

Membrane force

Helfrich model: bending energy proportional to the square of the curvature of the membrane. In 2D :

$$E_b = \int_\Gamma \frac{k_B}{2}\kappa^2$$

with $k_B \approx 10^{-19} J$ for a vesicle.

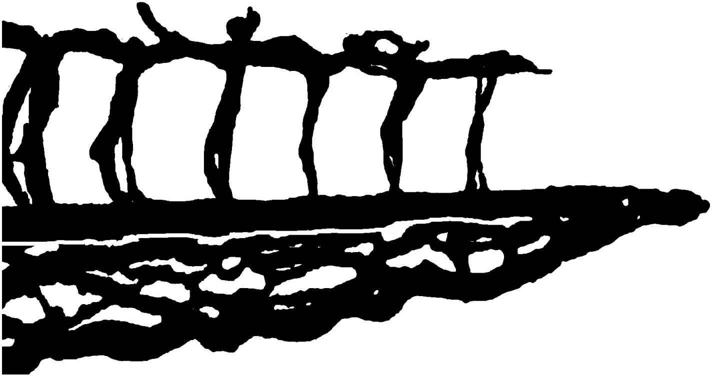
Corresponding force:

$$F_b = \int_\Omega k_B \nabla \cdot \left[\frac{-\kappa^2}{2} \frac{\nabla\phi}{|\nabla\phi|} + \frac{1}{|\nabla\phi|}\left(Id - \frac{\nabla\phi \otimes \nabla\phi}{|\nabla\phi|^2} \right) \nabla\{|\nabla\phi|\}\kappa \right] \delta_\epsilon$$

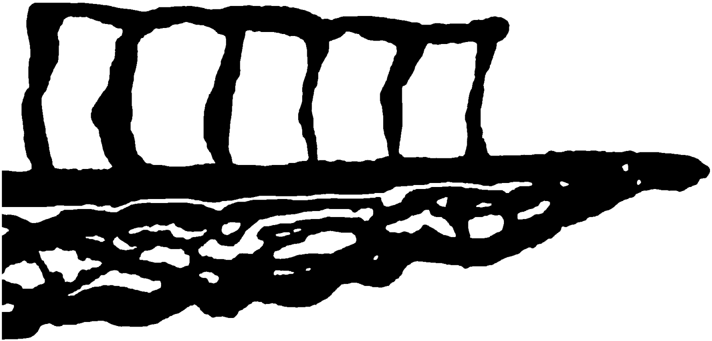
Full system

$$\begin{aligned}
\frac{D(\rho(\phi(x))u)}{Dt} - \nabla \cdot \left(\left[\mu(\phi(x))(\nabla u + (\nabla u)^T) \right] \right) + \nabla p &= F && \text{in } \Omega \\
\nabla \cdot u &= 0 && \text{in } \Omega \\
\nabla_s \cdot u &= 0 && \text{on } \Gamma \\
\frac{\partial \phi}{\partial t} + u \cdot \nabla \phi &= 0 && \text{in } \Omega \\
u &= g && \text{on } \partial\Omega
\end{aligned}$$

Mesh from a 2D image



Mesh from a 2D image



Octave script

Octave script to generate geometry file:

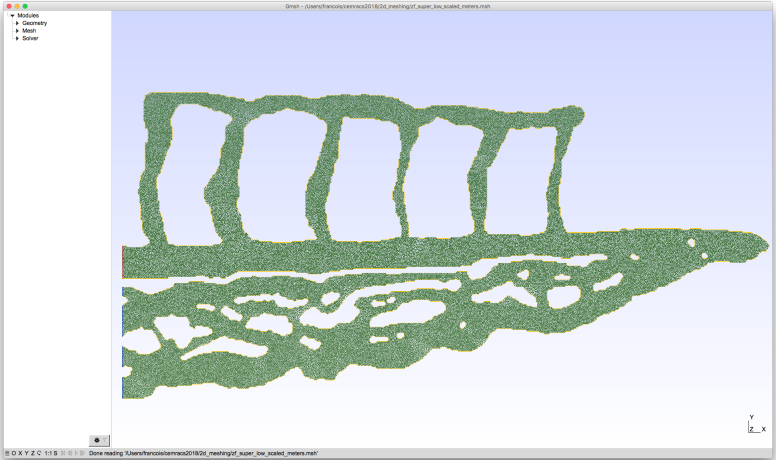
- extract contours
- filter out duplicate points
- write geometry file for *GMSH*

Mesh size

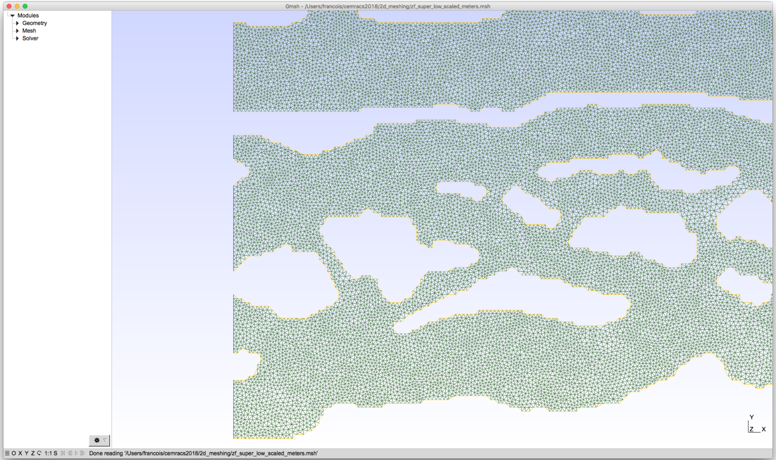
Mesh should be:

- fine enough to preserve geometry
- fine enough to define a cell with level set method
- coarse enough to save computing time

Mesh from a 2D image



Mesh from a 2D image



The point: solve a physics problem without rewriting everything

How it works:

- choose your toolbox : Multifluid with built-in level set method.
- give it a mesh or a geometry
- pick and adjust the physical models
- select and tune the solver
- have fun