Report on Anguilliform Swimmer Motion Study and Comparison with Existing Code

By Rohit Sunil Kanchi

Under the Guidance of Dr. Ashoke De

Anguilliform swimming is exhibited by lampreys, eels and many other marine organisms for propelling themselves through water. This has been studied over the past many years and many scientists and CFD experts have tried to model this motion and understand the underlying physics behind this motion to help bio-inspired propulsion in fluids, majorly water. In this particular study report, I have chosen the Anguilliform Swimmer C elegans class which exhibits C type curvature state while swimming. Inspired by this, (Battista, 2021) explored a very vast parameter subspace of the C elegans motion based on the variation of three input parameters, the Reynolds input number Rein, the undulation frequency and kinematic parameter, all of which have been explained by me in the forthcoming sections. I studied this to the best of my understanding and observed that some improvements in the model can be done and some additional steps in the study can be taken as well, to better understand the anguilliform motion. This has been explained in succeeding sections of this report. The sections of this report, which I have laid out to present my understanding of the topic, are as follows:

1. Properties of C elegans motion: Parameters Involved.
2. Modelling and Simulation: Approach and Methodology.
3. Modified Code and Comparison of results: How much of a difference does it make? [My contribution]
4. Conclusion and next steps.
5. Properties of C elegans and their motion: Parameters Involved:

As aforementioned, C elegans are class of species which exhibit anguilliform swimming motion. A typical C elegan is shown below in the figure 1.



Figure . A typical C elegan

Modelling its motion and studying the physics behind the swimming requires understanding of some basic **parameters** of the swimming motion, some of which dictate the motion and some of which are a result of the motion and help us to understand the effect of governing parameters which are input. The parameters have been used by Battista N of the college of New Jersey in his numerical studies pertaining to anguilliform swimming. They are mentioned below:

1. Reynold’s Number Re:

Where is the fluid density, is the characteristic length, is the swimming velocity and is the dynamic viscosity of the fluid.

1. Rein or the input Reynold’s number:

Where f is the undulation frequency.

1. Reout or the output Reynold’s number:

Where A is the peak-to-peak undulation amplitude as shown below in the figure 1.

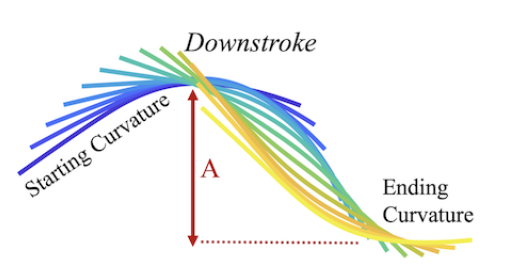


Figure 2. The depiction of stroke undulation amplitude of the Anguilliform Swimmer

Note that Rein and Reout are frequency based Re numbers and have been elected to be used by many researchers. The two Re numbers give a sense of the effect of inertial and viscous forces which act on the body of the swimmer.

1. Non-Dimensional forward swimming speed:

Where is the dimensional forward swimming speed.

Notice that the RHS of equation 4 is essentially the reciprocal of Strouhal Number [1/St] of a body moving with a certain velocity v in a fluid. Here, it is a frequency based Strouhal Number. According to (Battista, 2021) et al, this is in the range of 0.2 to 0.4 for most marine swimmers.

1. Cost of Transport: COT

This has a dimensional form and a non-dimensional form as well.

N is the number of time steps

Fj is the vertical force exerted on the Lagrangian point

Uj is the tangential body velocity of the swimmer and is basically equal to

Where Ux and Uy are the horizontal and vertical velocities of the Lagrangian points. The formula for COT is a power-based cost of transport. One may now ask how this formula was written? What parameters and assumptions went into it? How can the power expended by a swimmer in water be calculated? The answer starts at understanding the forces involved. Broadly speaking, there are viscous and inertial forces. The swimmers, although propagating forward, expend more of their energy in displacing the fluid while undulating side to side. This means that at low Reout, the inertial forces and viscous forces in the axial direction are negligible. However, the inertial forces are large in comparison when looking at the motion in the lateral directions (Bale et al., 2014).

Hence, only the vertical forces are considered. Note that the power which is a scalar product of velocity and force appears in a vector form in the equation 5. Non-Dimensionalising is the next step which is in equation 6. This is important to understand because the formula seems a little counter intuitive at first glance.

The anguilliform swimmer model data is obtained from (Jung, 2010) and other sources such as (Gutierrez et al., 2014) and (Padmanabhan et al., 2012). The images and motion video graphs were studied by Battista et al but at this point, it is very unclear as to which exact experimental data he did refer to, to obtain the approximate shape of the anguilliform swimmer.

The methods to model and numerical method to solve for the governing fluid equations have been written in the next section as follows.

1. Approach to model and numerical methodology followed:

To solve for the motion and output parameters of the anguilliform swimmer, the method of computational fluid dynamics was used by Battista et al in his research work. The exact numerical method used is a subset of CFD, namely the immersed boundary method [IBM] which is based on Dr. Charles Peskin’s IB method under fluid structure interaction problems (Peskin, 2002). The research started back then on trying to model heart valves and their fluid structure interaction, with the fluid of course being blood (Peskin & McQueen, 1980). Since then, lot of development has happened in the code and many robust solvers have come up in the recent years, one of them being IB2D, an IBM solver in 2D which is a MATLAB/PYTHON based code developed by (Battista et al., 2018). The code has lot of fiber models and can solve a variety of FSIs in 2D.

To begin with, I would like to mention the information which I read and the information presented to me by many others which helped me to understand the physical working and principles behind the immersed boundary method.

Consider a simple wall which is flexible in real life and this walls moves by its own upon stimulations given to it externally or within itself or both. An example of this could be a muscle in the heart moving blood, which is the fluid here, and having periodical motion. One way to model this phenomena would be to take snaps of a video graph at very small time intervals and approximate the motion between those time steps by recording the kinematic parameter points such as [x,y,z] location at time t. This can then be fed into a solver which solves FSI problems and obtains results in an unsteady mode. Note that here, the reaction forces from the fluid or the forces from the fluid as a result of the interaction, post interaction, are being damped and the energy is being dissipated in those regions in different ways. The wall does not move because of the fluid’s motion and forces on it. However, if the walls were to deform under effect of the fluid forces on it, this would be much more accurate.

This was the motivation behind development of the IBM for FSI problems.

Now consider a ball in 2D. If that ball were to be in the same place fixed and a fluid were to flow around it, this is a very simple 2D flow around a cylinder problem most fluid mechanics textbooks talk of. To solve for the flow field around it, we could employ CFD principles and give the circumference a wall boundary condition, make the mesh around the walls and solve the governing equations [NS].

Now what is the meaning of a wall? It means that the velocity at the wall points is zero. Keeping that in mind, now consider the circumference of the circle to be that of a ring which is elastic. This is stretched to an initial position in the fluid and is then released. Note that fluid is within the ring’s periphery and outside as well. We could model this by the same way but govern the motion of the walls. However, this would have to essentially change the mesh around the wall region because the mesh cells are where the governing equations are solved.

Instead, recalling the point previously made, if we were to tell the solver that a certain point represents a wall point and the velocity of the fluid is zero there, then we could give the fluid some velocity by applying a force on it from the boundary when it moves and the fluid, in turn, would move the “pseudo boundary” depending on the velocity of the fluid at that point. The point when moved, will be moved over the mesh and that particular location, the velocity of the fluid is forced to become zero.

Now, the point under consideration need not necessarily be on a mesh node. It can be inside a cell on the boundary of a cell in a mesh as well. Solving the equations and proceeding with the numerical calculations is done by interpolating those values to the nearest node. The mesh is a Eulerian mesh, meaning the Navier Stokes equations are solved in Eulerian form and the points which define the boundary of the body “immersed in fluid” are called Lagrangian points. Names are self-explanatory.

Further elaboration on the method is given in Peskin’s paper as aforementioned.

A general algorithm of the IBM is given as follows:

1. Forces on the boundary [Lagrangian points] are computed from the current updated configuration and current time step.
2. The forces are spread to the fluid lattice nodes in the vicinity of the L-Points.
3. The fluid flow governing equations [NS] arising from application of these forces are solved in the entire Eulerian grid.
4. Update the Lagrangian point positions by moving them at the local fluid velocities at the particular time step.

The steps from a to d are repeated again for the time steps and hence time duration of study specified.

Now, coming to the anguilliform swimmer, the Lagrangian points connected by straight lines form the body of the swimmer and Battista et al gave an approximation to this swimmer by studying its undulation shapes at various time frames of its motion. The body in the initial configuration comprises of a straight line and a cubic line portion as shown in the figure below:



The undulations are time step dependent and are interpolated between two phases as shown below:



1 is the phase 1 on the left side in yellow and the other in red is the phase 2.

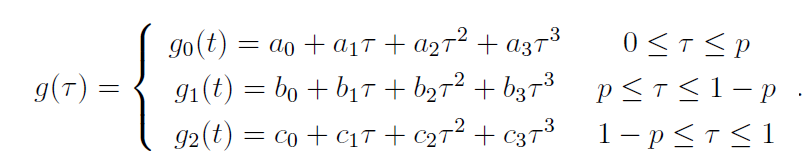
Using higher order polynomial interpolation technique (Battista, 2018), Battista et al constructed the model of the swimmer which can be found in the IB2d first year seminar folder in the git hub page: <https://github.com/nickabattista/IB2d/tree/master/matIB2d/Examples/Examples_First_Year_Seminar/Swimmer>

If the phase 1 is a curvature state denoted by A and 2 is denoted by B, then the polynomial interpolated lagrangian point locations resulting in the time dependent curvature state is given by:



Where h is the time dependent curvature state.

g(t) is defined as:



P is the kinematic parameter.

t is the modular arithmetic value of time in simulation and the period of undulation to get a value between 0 and 1.

is a non-dimensional time given by fraction of upstroke or down stroke and is equal to t / ( 0.5/f ).

Varying p changes the acceleration of the swimmer from the initial phase and the final deceleration of the points before the final phase.

As p increases, the acceleration or deceleration becomes larger in magnitude.

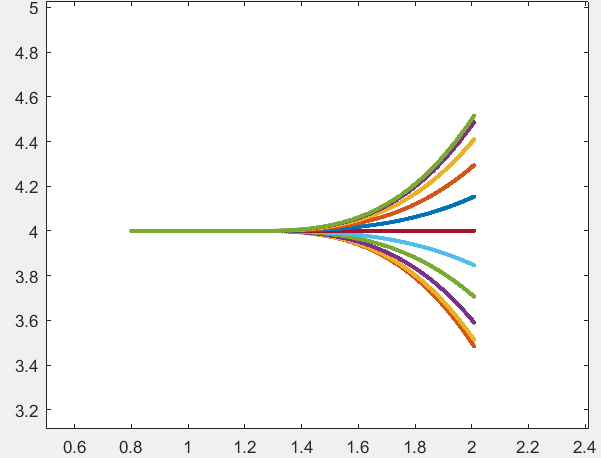
The coefficients c0, c1, etc depend on p and 1-p. They are obtained from the code interp\_Function\_Coeffs.m script found in supplemental materials of (Battista, 2018).

Other parameters such as input Re number, frequency, viscosity, etc can be given to the “input2d” file and “update non invariant beams” file depending on the physics we are looking at to compare or achieve or both.

The swimmer is modelled using Lagrangian points, Lag\_Springs and non-invariant beams (Battista et al., 2018). The beams’ curvatures are updated based on the interpolated curvature state obtained from the higher order polynomial interpolation.

1. Contributions

In the present work by Battista et al, the anguilliform swimmer has been modeled to interpolate in a time dependent fashion and update its curvature state between two phases at a particular time step. A small observation made by me was that C elegans have a body structure which does not elongate significantly when swimming or even crawling. However, in the interpolation functions used by Battista et al in IB2d, the interpolation has only been applied to the ordinates of the Lagrangian point locations, not the abscissae [x co-ordinate]. The same is shown in figure below:



Some curvature states are shown in between the two phases in the figure above on the extreme lateral ends where the curved portion is given by y=x3

As one can see, the end points are in the same straight line and this is causing elongation. Further, the output Re number depends on the length of the swimmer. Although variations seem to be small, the effect of conserving the length could be investigated.

During the simulation, a simple code can be written in the script to display the total length of the swimmer at every time step. The logic is as follows:

Where x and y are the abscissa and ordinate respectively of the Lagrangian point and N is the total number of Lagrangian points. This gives the net length of the swimmer’s body in the simulation. Alternately one can use a simple thread and measure on the screen during the post processing stage, an old traditional and alternate way. This revealed that the length changed by 0.2 units. This could cause difference in the physics of the flow we are looking at, such as length travelled, power consumed, etc.

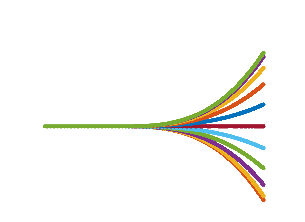
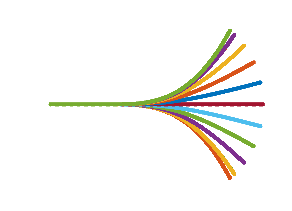
To investigate if this could be an issue, I elected to update the x ordinates as well, on a length conservation basis.

Post simulation, I also used the IB2d post processor script and wrote a code to obtain the forward velocity, power expensed and the non-dimensional COT. (These terms have been explained in the preceding sections).

The code I modified and the IB2d Blackbox can be found at: <https://github.com/rohitroxkp7/Swimmer-Modified>

I have forked the code from Nick Battista’s repository in a different repository where both codes can be compared at: <https://github.com/rohitroxkp7/IB2d>

Shown below is a comparison of Nick Battista’s model (left) and my model (right):

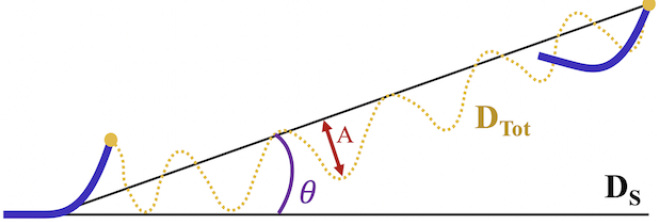
The input parameters for the two simulations were kept exactly same and are listed in the table below:

|  |  |
| --- | --- |
| Input Parameter | Value |
| L | 1.4297 m |
|  | 1000 kg/m3 |
|  | 10 Pas |
| f | 2 Hz |
| Rein | 408.8 |
| tfinal | 3 s |
| Dt (time step duration) | 2.5e-05 |
| p | 0.1 |

The Simulation results are listed in the table below:

|  |  |  |
| --- | --- | --- |
| Output Parameter | My Model | Battista et al |
| Vforward | 0.9058 m/s | 1.44 m/s |
| Length travelled | 2.8 m | 4.4 m |
| Undulation Amplitude | 0.25 m | 0.3 m |
|  | 0 deg | 8 deg |
| Reout | 71.485 | 85.782 |
| COTnon-dim | 1106.685 | 869.09 |
| Vnon-dim-forward [1/St] | 1.812 | 2.4 |

Shown in the figure below is an illustration of what means.



A link to the videos of the motion with scale on x and y axes is given below, which has the videos of the velocity magnitude field and the Lagrangian Swimmer. There are two videos, one is Nick Battista’s model and one is mine.

Link to Battista et al model: <https://youtu.be/l5gnTvT5rkg>

Link to My model: <https://youtu.be/pQaPte9zTnU>

**Remarks:**

It is evident that Battista’s model, if made in real life could possibly be much more efficient as the COT is lesser and the swimming speed and distance traversed are higher. However, this is for the current set of input parameters. A wider range of input parameters and simulations would be needed to thoroughly conclude which model is more efficient. However, this is not the aim of this investigation study. The aim is to bring out differences, if any and we can see that there are some noticeable differences.

I would like to make mention here that nothing/no one is exactly right or wrong here, but a mere comparison was made. This is because both motions are possible as there are a variety of C elegans and subjected to different environments and different variations in body type, stiffness and length. Further, when the length decreases in between the phases, the Lagrangian springs store elastic energy which when released, give a kind of springing effect, elongating and pushing fluid and giving some more forward velocity. This is why the swimmer in Battista’s case was able to travel farther along in the same time duration of 3s when compared to my case where the length of the swimmer remains same, as can be seen in the video links above. Another thing which can be noticed is that the angular trajectory is not present in my case. It travels more or less in the same straight path. However, this is for one case and one set of input parameters. More investigation in needed in this. The swimming motion remains rather similar and there is no other new flow physics I could see noticeably.

1. Next Steps and References

The swimmer was modelled and compared with Battista et al’s model and there were some noticeable differences.

In the study done by Battista, the simulations performed are done in a rather calm environment, i.e., the fluid is static initially. A simulation such as a swimmer in a channel flow can be done in which there is either an upstream or a downstream flow already present and the swimmer swims through it.

**References:**

Bale, R., Hao, M., Bhalla, A. P. S., & Patankar, N. A. (2014). Energy efficiency and allometry of movement of swimming and flying animals. *Proceedings of the National Academy of Sciences of the United States of America*, *111*(21), 7517–7521. https://doi.org/10.1073/pnas.1310544111

Battista, N. A. (2018). *Fluid-Structure Interaction for the Classroom: Interpolation, Hearts, and Swimming!* *08628*.

Battista, N. A. (2021). Swimming through parameter subspaces of a simple anguilliform swimmer. *Integrative and Comparative Biology*, *60*(5), 1221–1235. https://doi.org/10.1093/ICB/ICAA130

Battista, N. A., Strickland, W. C., Barrett, A., & Miller, L. A. (2018). IB2d Reloaded: A more powerful Python and MATLAB implementation of the immersed boundary method. *Mathematical Methods in the Applied Sciences*, *41*(18), 8455–8480. https://doi.org/10.1002/mma.4708

Gutierrez, J., Sorenson, M., & Strawbridge, E. (2014). Modeling fluid flow induced by C. elegans swimming at low Reynolds number. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8890*, 71–82. https://doi.org/10.1007/978-3-319-13749-0\_7

Jung, S. (2010). Caenorhabditis elegans swimming in a saturated particulate system. *Physics of Fluids*, *22*(3), 3–6. https://doi.org/10.1063/1.3359611

Padmanabhan, V., Khan, Z. S., Solomon, D. E., Armstrong, A., Rumbaugh, K. P., Vanapalli, S. A., & Blawzdziewicz, J. (2012). Locomotion of C. elegans: A piecewise-harmonic curvature representation of nematode behavior. *PLoS ONE*, *7*(7). https://doi.org/10.1371/journal.pone.0040121

Peskin, C. S. (2002). *The immersed boundary method*. 479–517. https://doi.org/10.1017/S0962492902000077

Peskin, C. S., & McQueen, D. M. (1980). Modeling prosthetic heart valves for numerical analysis of blood flow in the heart. *Journal of Computational Physics*, *37*(1), 113–132. https://doi.org/10.1016/0021-9991(80)90007-8