# Numerical Methods Final Project

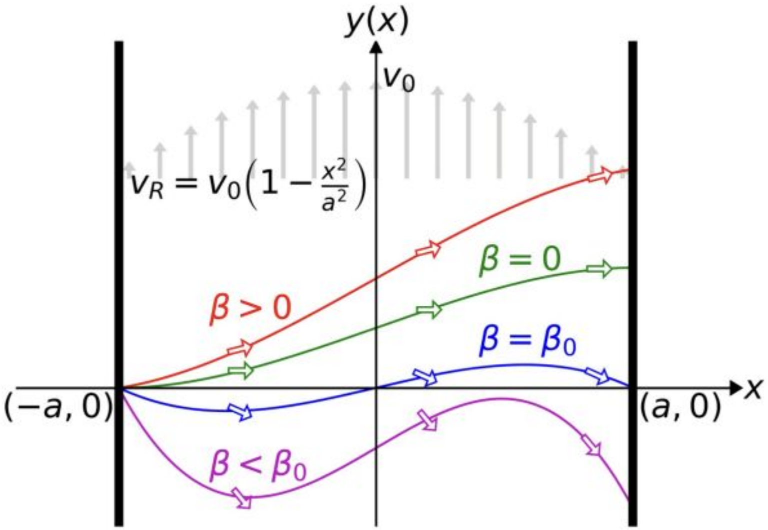
Yulia Grajewska jg6403

# Roadmap

1. Problem Statement
2. Description of the Differential Equation involved
3. Trajectory for $\beta = 0$
4. Length of the path for $\beta = 0$
5. Interpolation of the path for $\beta = 0$
6. Computing drag for different values of $\beta$
7. Interpolation for optimal $\beta$
8. Conclusion

# Problem Statement

**Problem F.2:** We set a 2-dimensional Cartesian coordinate system for a section of a river such that the $y$-axis lies at the center of the river and the $x$-axis runs from west to east. The water is assumed to always flow strictly northward and its speed, denoted by $w(x) = v_0\left(1 - \frac{x^2}{a^2}\right)$, varies depending on the $x$-coordinate of the location. A swimmer crosses the river from a point $(-a, 0)$ to the other bank, moving at a constant speed $v_B$ and at a fixed angle $\beta$ with respect to the $x$-axis. Assume the river flow will drag the swimmer at 100% efficiency.



# Problem Statement

Please do the following:

(1) Write a program to find trajectory of the swimmer for $x \in [-a, +a]$ if the $\beta = 0$.

(2) Compute the length of the trajectory you found in (1).

(3) Interpolate the evenly selected 4 points, on $x \in [-a, +a]$, from the trajectory you found in (1) by the following polynomial

$$y(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

(4) For each of 4 values of $\beta = -\frac{\pi}{8}, -\frac{\pi}{16}, \frac{\pi}{16}, \frac{\pi}{8}$, compute $y_{\text{Drag}} = y(x = +a)$. Note: You need to find the trajectory for each case to find this $y_{\text{Drag}}$.

(5) Use your results in (4) to find a way to "shoot" for an optimal value at which $y_{\text{Drag}} = y(x = +a) \approx 0$.

# Finding the Differential Equation

initial speed:

$$v_{swimmer_x} = v_{swimmer} \times cos(\beta)$$
$$v_{swimmer_y} = v_{swimmer} \times \sin(\beta)$$

the differential equations:

$$\frac{dx}{dt} = v_{swimmer} \times cos(\beta)$$
$$\frac{dy}{dt} = v_{swimmer} \times sin(\beta) + w(x)$$

putting them together:

$$\frac{dy}{dx} = \frac{v_{swimmer} \times sin(\beta) + v_{water}(1 - \frac{x^2}{a^2})}{v_{swimmer} \times cos(\beta)}$$

# Solving the DE Numerically

Euler's method:

$$y_{n+1} = y_n + h \times f(x_n, y_n)$$

$$x_{n+1} = x_n + h$$

our f(x) is:

$$\frac{dy}{dx} = \frac{v_{swimmer} \times sin(\beta) + v_{water}(1-\frac{x^2}{a^2})}{v_{swimmer} \times cos(\beta)}$$

the parameters and initial conditions are:

$$x_0 = -a$$

$$y_0 = 0$$

$$v_{swimmer} = 1 m/s$$

$$v_{water} = 0.889\ m/s$$

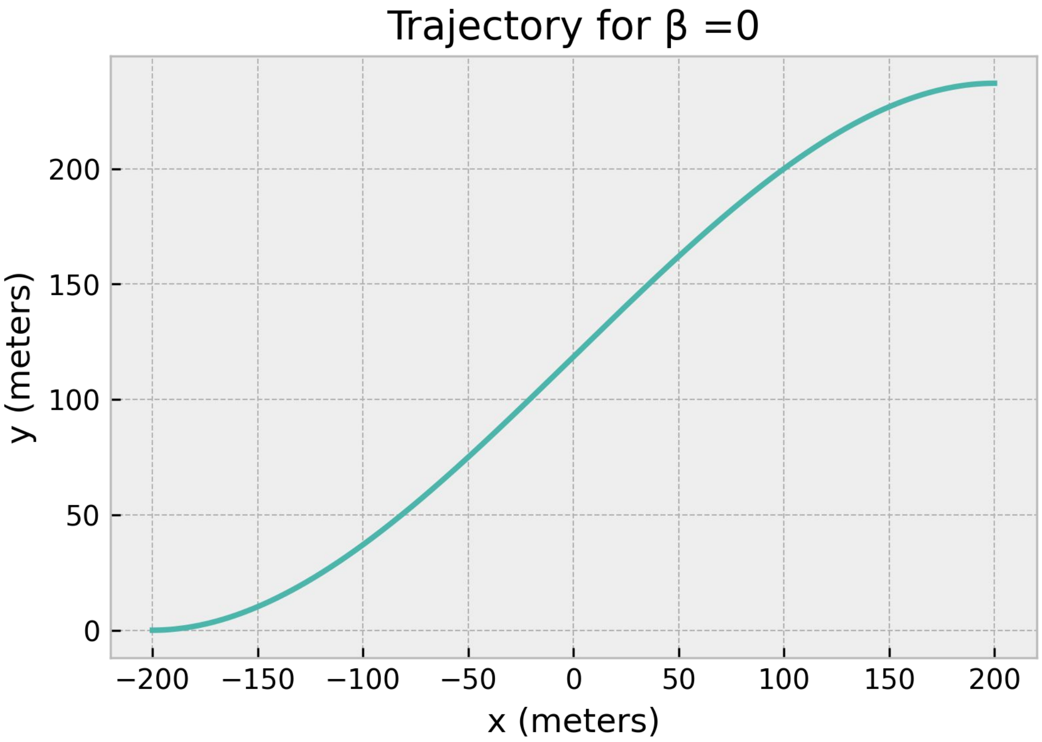
$$a = 200\ m$$

$$\beta = 0 \text{ (for now)}$$

I chose the step size to be h = 0.04.

# (1) Trajectory and drag for β = 0

The resulting drag is:
**237.07 m** (rounded to 2 d.p.)



# (2) Length of the path for β = 0

Method:

- trajectory saved as two arrays of consecutive values of x and y coordinates
- the path between two consecutive points is approximately:

$$d_n = \sqrt{(x_n - x_{n-1})^2 + (y_n - y_{n-1})^2}$$

- total length is the sum of all the partial paths
- **the resulting length for β = 0 is: 474.59 m**

# (3) Interpolating the path for β = 0, method

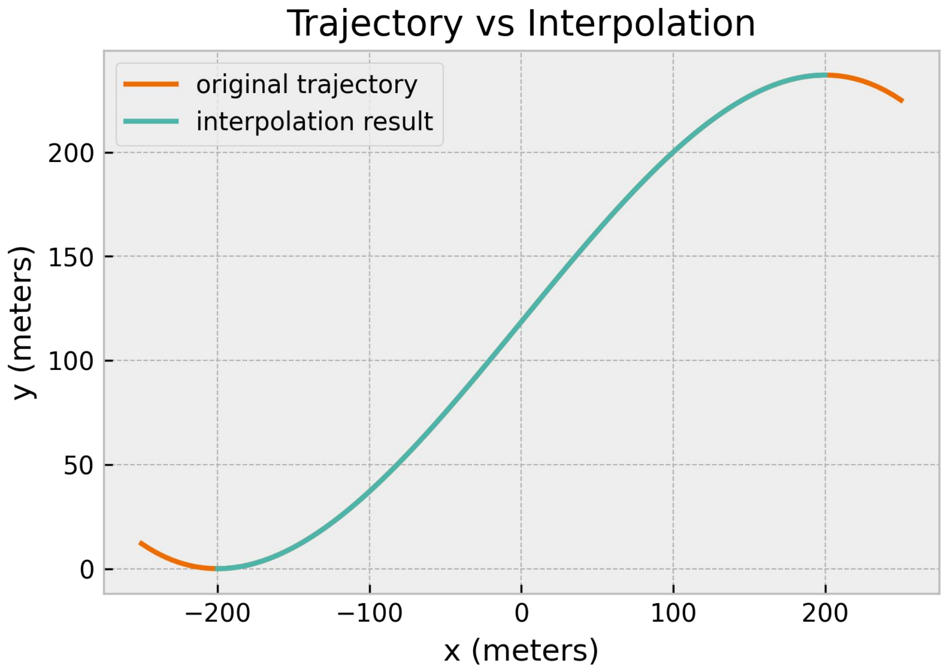
Interpolation using Vandermonde matrix (solving the following system of equations):

$$\begin{bmatrix} x_0^n & x_0^{n-1} & x_0^{n-2} & \dots & x_0 & 1 \\ x_1^n & x_1^{n-1} & x_1^{n-2} & \dots & x_1 & 1 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ x_n^n & x_n^{n-1} & x_n^{n-2} & \dots & x_n & 1 \end{bmatrix} \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_0 \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

# (3) Interpolating the path for ß = 0, result

The resulting polynomial function:

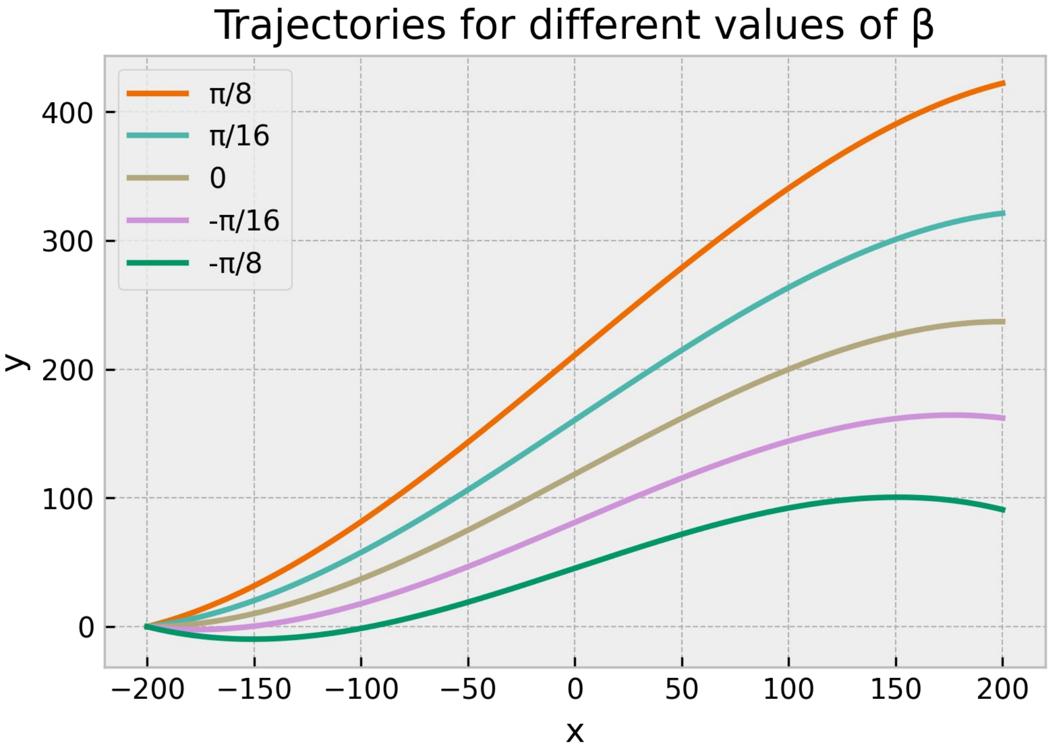
$$f(x) = 1.185 \times 10^{2} + 8.890 \times 10^{-1} x$$
$$+4.445 \times 10^{-7} x^{2} - 7.408 \times 10^{-6} x^{3}$$



# (4) Trajectories for different values of β

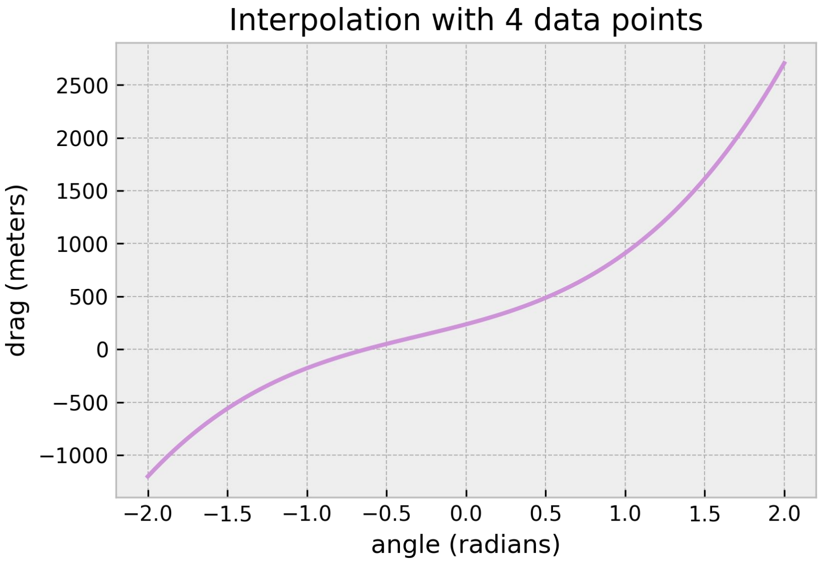
Values of drag:

- $\beta = \pi/8$:  422.28 m
- $\beta = \pi/16$:  321.28 m
- $\beta = 0$: 237.07 m
- $\beta = -\pi/16$: 162.15 m
- $\beta = -\pi/8$: 90.91 m

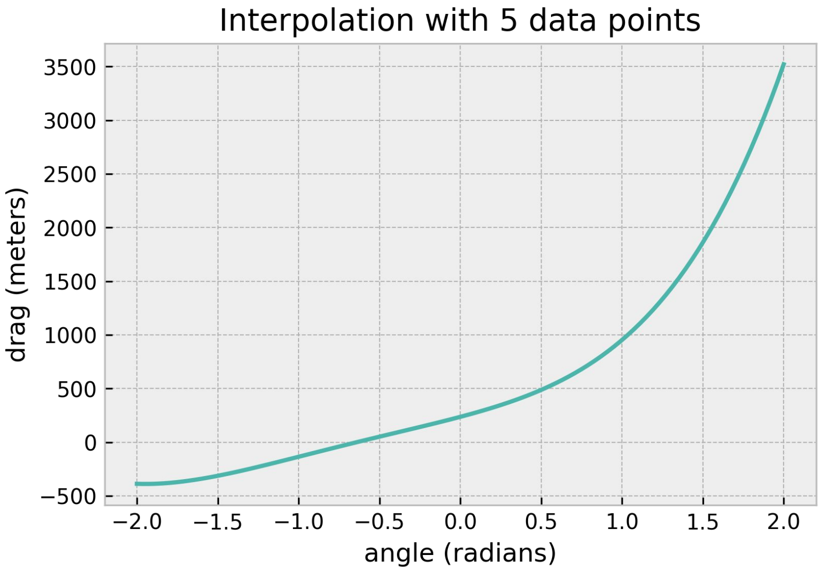


# (5) Interpolation for optimal β

using 4 points (without β=0)




using 5 points



# (5) Interpolation for optimal β, finding the zeros

- using secant method to find the zeros of the resulting polynomial
- **for 4-point interpolation, the result is : β = -0.63464264; resulting drag: -0.11 m**
- for 5-point interpolation, the result is: β = -0.6266455173268011, resulting drag: 3.08m

```python
def secant(f,a,b,arr):
  c = 0
  while np.absolute(f(b,arr)) >= 0.5 * 0.0001:
    c = b
    if b!=a:
      b = b - (b-a)*f(b,arr)/(f(b,arr)-f(a,arr))
      a = c
    return(b)
```

# Conclusions

- drag extremely sensitive to the initial angle:

| value of β (radians) | drag (meters) |
|---|---|
| -0.634 | 0.15 |
| -0.635 | -0.25 |
| -0.636 | -0.65 |
| -0.637 | -1.05 |

| value of β (radians) | drag (meters) |
|---|---|
| 1.000 | 1061.72 |
| 1.001 | 1063.79 |
| 1.002 | 1065.85 |
| 1.003 | 1067.92 |

# Conclusions

- for $\beta=0$, the resulting drag is around 237.07m, and the length of the path: 474.59 m
- the optimal value of $\beta$ which minimizes the drag, is around $\beta$ = -0.63464264, which results in a drag of around 11 cm

# Comparison with the analytical solution

The analytical solution is:

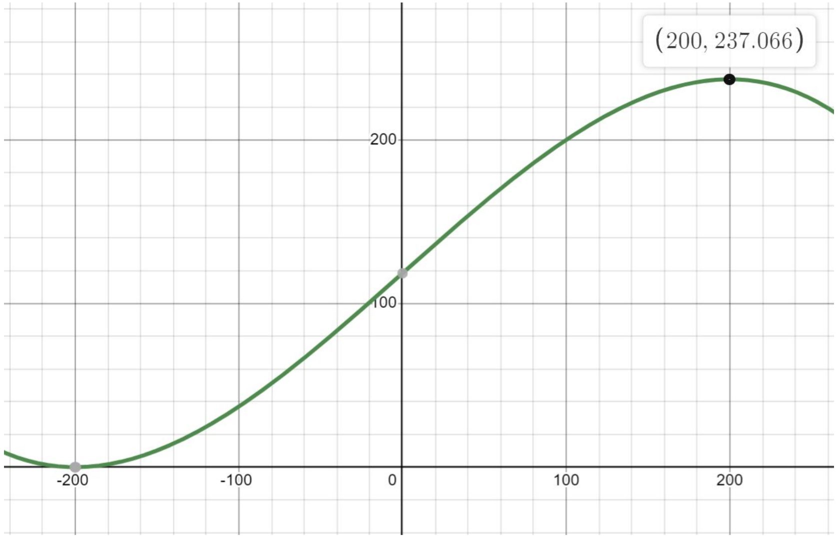
$$\left(-7.40833 \times 10^{-6}\, x^3 + 0.889\, x + 118.533\right) \sec(b) + (x + 200) \tan(b)$$

(courtesy of WolframAlpha)

For β = 0, the drag is approximately **237.07m**, which is the same value that we got numerically

The optimal value of β is around **-0.6344**, which is similar to the numerical result: -0.63464264

Path length found analytically is **474.59 m**, which is the same as the numerical solution



# Possible Further Research

- analysis of the effect of other parameters, like the speed of the swimmer of the speed of the water
- optimization with respect to time