First Mandatory Assignment

MEK4420

Simon Lederhilger

January 28th 2025

Scope of the assignment

The assignment at hand is to solve the integral equation

$$\pi\phi(\boldsymbol{\varkappa c}) = \int_{\partial\Omega} \left(\phi\partial_{\hat{\boldsymbol{n}}}\,\mathrm{G} - \mathrm{G}\,\partial_{\hat{\boldsymbol{n}}}\phi\right)\mathrm{d}S$$

by numerical implementation, calculate the added mass, and compare the results to theoretical values. The shapes to be considered are a circle, ellipses of aspect ratios $a/b = 1/2$ and $a/b = 1/10$, and a square. As we shall need to consider the theoretical values of these shapes' added masses, these will be derived. Although the lecture notes found in the repository in which this assignment is located already have an explanation of added mass, albeit rather heuristically inclined, a different approach will be taken for this assignment. This assignment will furthermore function as documentation for the `Python`-code developed for the purpose of calculating added mass.

Added mass on bodies

To gain insight into what *added mass* actually represents, the reader is highly encouraged to read the 1953 paper of Charles G. Darwin *Note on Hydrodynamics.*[1] It is therein demonstrated that bodies moving in an ideal fluid cause drift of the fluid, and that the drift volume is equivalent to the added mass. Authors in older works usually provide the kinetic energy of the system instead of the added mass, as the added mass may be inferred. Consider a body in an unbounded fluid being accelerated such that its energy is T_{body}. We know this induces a drift in the fluid, whose energy we shall label T_{fluid}, and that the total energy then is $T = T_{\text{body}} + T_{\text{fluid}}$. Consider now a fluid flow with potential Φ. Letting ϱ denote the fluid density, the kinetic energy of the fluid is then

$$T_{\text{fluid}} = \frac{\varrho}{2}\int_{\Omega} q^2\,\mathrm{d}V, \quad q = |\nabla\Phi| = \frac{\mathrm{d}w}{\mathrm{d}\mathcal{Z}}\frac{\mathrm{d}w^*}{\mathrm{d}\mathcal{Z}^*},$$

integrating in the entire fluid domain Ω, excluding the body. The complex formulation will be necessary later, but we pay no attention to it for the time being. The speed q may be related to the velocity magnitude U by assuming the modal superposition $\Phi(t;\boldsymbol{x}) = \boldsymbol{U}(t)\cdot\boldsymbol{\phi}(\boldsymbol{x})$. It is apparent that in general, the energy may be expressed as

$$T = \frac{(m+m')U^2}{2}, \qquad m' = \varrho k A,$$

where we note the added mass m' depends on the area of the body A, and an inertia coefficient k. This coefficient indicates the orientation of the body in relation to the direction of motion—we can imagine an ellipse moving parallel to its major axis will cause less drift in the fluid than it would moving perpendicular to it. That will at least that will be shown to be the case, mathematically. One may confirm the dependency on orientation readily in one's own kitchen—if imagination does not satisfy—by filling a container with water, and holding a spoon so that the bowl is submerged, moving it every which way. One shall find that moving the spoon with the bowl perpendicular to the direction of motion requires more work than moving it with the bowl parallel to the direction of motion. In fact, maintaining such work with a force F, we have that $FU = \partial_t T_{\text{fluid}}$, and extending Blasius' theorem,[2] we have that

$$\boldsymbol{F} = -\partial_t\boldsymbol{U}\colon \varrho\int_{\partial\Omega}\boldsymbol{\phi}\otimes\hat{\boldsymbol{n}}\,\mathrm{d}S,$$

where S is the contour of the body. $\boldsymbol{\phi}$ is a vector containing the modes of translation,

$$\boldsymbol{\phi}(\boldsymbol{x},t) = \phi_1\hat{\boldsymbol{\imath}} + \phi_2\hat{\boldsymbol{\jmath}} + \phi_6\hat{\boldsymbol{\omega}}_{\hat{\boldsymbol{k}}}.$$

This same formulation is found in the lecture notes, yielding the added mass tensor in terms of the following integral, which will be used for the numerical calculation.

$$\mathfrak{m} = \varrho\int_{\partial\Omega}\boldsymbol{\phi}\otimes\hat{\boldsymbol{n}}\,\mathrm{d}S \tag{1}$$

[1] [2] Darwin (1953)

[2] [5] Milne–Thomson, pp.255–256

Discrete integral equation

The integral equation

Since the fluid is ideal, we may reduce the incompressibility condition to the Laplace equation,

$$\nabla^2\phi = 0, \qquad \text{in } \Omega.$$

It may be shown with the product rule that for any harmonic functions $\phi, \psi \in \Omega$,

$$\nabla \cdot \big(\phi\nabla\psi - \psi\nabla\phi\big) \equiv 0,$$

which upon being integrated over Ω and applying Gauss' divergence theorem, yields

$$\int_{\partial\Omega} \big(\phi\nabla\psi - \psi\nabla\phi\big)\,\mathrm{d}S = 0.$$

Since we only need the values of the potential on the contour $\partial\Omega$ to calculate the added mass, we look to Green functions, which are defined through the property that they satisfy some differential operator except at some point $\boldsymbol{эю} \in \mathrm{Q}$, where $\mathrm{Q} = \Omega \cup \partial\Omega$, such that

$$\nabla^2\,\mathrm{G} = \delta(\boldsymbol{x} - \boldsymbol{эю}), \qquad \boldsymbol{x} \in \mathrm{Q}.$$

In other words, the Green function is almost harmonic, and we expect the above integral should hold except at the pole. The Green function for the Laplace operator is the natural logarithm,

$$\mathrm{G}(\boldsymbol{x}) = \ln r, \qquad r = |\boldsymbol{x} - \boldsymbol{эю}|,$$

which has a pole at $\boldsymbol{эю}$. Now, by placing $\boldsymbol{эю} \in \partial\Omega$, we find by the Cauchy principal value,[1] that

$$-\pi\phi(\boldsymbol{эю}) + \mathrm{pv}\int_{\partial\Omega} \phi\partial_{\hat{\boldsymbol{n}}} \ln r\,\mathrm{d}S = \mathrm{pv}\int_{\partial\Omega} \partial_{\hat{\boldsymbol{n}}}\phi \ln r\,\mathrm{d}S.$$

An outline of calculating the principal value is found in the lecture notes from January 21st. We drop the principal value notation for brevity.

The boundary element method

To implement the integral equation numerically, we utilize the boundary element method. In essence, we distribute a number of nodes along the boundary on which we would like to solve the integral equation, and linearily interpolate the points to approximate boundary. If the number of nodes is N, then we have $\partial\Omega \sim S = \{S_n \colon n \leq N, n \in \mathbb{Z}^+\}$. We then assume the potential is constant on each of S_n, equal to the potential evaluated at the midpoint, labelling this ${\phi_j}^n \equiv \phi_j(\boldsymbol{ж}_n)$. Now, since the normal derivative of the potential also must be zero on the line segments, the integrals in the integral equation may be approximated by the sum of the integrals evaluated over wach of the line segments as follows.

$$\int_{\partial\Omega} \phi\partial_{\hat{\boldsymbol{n}}} \ln r\,\mathrm{d}S \approx \sum_{n=1}^{N} \phi^n \int_{S_n} \partial_{\hat{\boldsymbol{n}}} \ln r\,\mathrm{d}S \tag{2}$$

$$\int_{\partial\Omega} \ln r\partial_{\hat{\boldsymbol{n}}}\phi\,\mathrm{d}S \approx \sum_{n=1}^{N} \hat{\boldsymbol{n}}^n \int_{S_n} \ln r\,\mathrm{d}S \tag{3}$$

It should be quite clear that the integral equation may then be written as the matrix equation

$$-\pi\phi^n + \sum_{n=1}^{N} \phi^n \mathtt{\Theta}_{m,n} = \sum_{n=1}^{N} \hat{\boldsymbol{n}}^n \mathtt{h}_{m,n},$$

where we have set $\mathtt{\Theta}_{m,n}$ and $\mathtt{h}_{m,n}$ to be approximations of equations (2) and (3), respectively.

The logarithmic gradient

It turns out the integral of the gradient of the logarithm can be determined using complex analysis. The gradient is an operator from the real numbers into the vector space of the complex numbers, in the sense that

$$\nabla u(\mathfrak{z}) = \partial_x u(\mathfrak{z}) + i\partial_z u(\mathfrak{z}), \qquad \mathfrak{z} = x + iz.$$

Considering now an analytic function $\boldsymbol{u} = u + iv$, its complex derivative is given by $\partial_{\mathfrak{z}}\boldsymbol{u} = \partial_x u + i\partial_x v$, which by the Cauchy–Riemann equations yields that $\partial_{\mathfrak{z}}\boldsymbol{u} = \nabla u^*$. We recall that the principal value of the complex logarithm is given by $\log(\mathfrak{z}) = \ln|\mathfrak{z}| + i\,\mathrm{Arg}(\mathfrak{z})$,[2] so that $\partial_{\mathfrak{z}} \log(\mathfrak{z}) = \nabla \ln|\mathfrak{z}|^*$.

Now, for a curve parametrized by $\lambda(s)$, the normal vector $\hat{\boldsymbol{n}}$ may be represented by[3]

$$\nu(s) = -i\lambda'(s) = \frac{\mathrm{d}z}{\mathrm{d}s} - i\frac{\mathrm{d}x}{\mathrm{d}s} = -i\frac{\mathrm{d}\mathfrak{z}}{\mathrm{d}s}. \tag{4}$$

The inner product may be defined such that $\boldsymbol{u}\cdot\hat{\boldsymbol{n}} = \mathrm{Re}\,(\boldsymbol{u}^*\nu(s))$. We then have that

$$\partial_{\hat{\boldsymbol{n}}} \ln r = \mathrm{Re}\left(-i\frac{\mathrm{d}\mathfrak{z}}{\mathrm{d}s}\partial_{\mathfrak{z}} \log(\mathfrak{z} - \boldsymbol{эю}_m)\right).$$

Since the line segment is parameterized by s, the differential element may then be taken over $\mathfrak{z}$. By the linearity of the $\mathrm{Re}\,(\star)$ operator, we have that

$$\begin{aligned}\int_{S_n} \partial_{\hat{\boldsymbol{n}}} \ln r\,\mathrm{d}S &= -\,\mathrm{Re}\,i\int_{\boldsymbol{x}_{n-1}}^{\boldsymbol{x}_n} \frac{\mathrm{d}}{\mathrm{d}\mathfrak{z}} \log(\mathfrak{z} - \boldsymbol{эю}_m)\,\mathrm{d}\mathfrak{z}. \\ &= \mathrm{Arg}\left(\frac{\boldsymbol{x}_{n-1} - \boldsymbol{эю}_m}{\boldsymbol{x}_n - \boldsymbol{эю}_m}\right).\end{aligned}$$

This is just the angle between $\boldsymbol{x}_n$, $\boldsymbol{эю}_m$, and $\boldsymbol{x}_{n-1}$.

[1] [3] Lavrentev & Šabat, pp.331–332
[2] [3] Lavrentev & Šabat, p.30
[3] [4] Markuševič, p.175

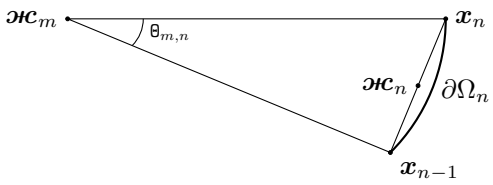

Figure 1: Visualization of $\Theta_{m,n}$. Here $\partial\Omega_n$ denotes the segment along $\partial\Omega$ between $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n-1}$.

If $\boldsymbol{x}$ then is the array of the N nodes x_n, we can fill the matrix Θ with the arguments with `for`-loops, which is implemented in the `assemble` method in the `IntegralEquation` class, found in `integralequation.py`. The class variable $\boldsymbol{\varkappa} = {}^1\!/_2(\boldsymbol{x}_n + \boldsymbol{x}_{n-1})$, containing the aforementioned midpoints is created upon initialization of the class, since multiple methods make use of it.

Algorithm 1 Assemble Θ

```
for m, n ≤ N do
    if m = n then
        Θ_mn ← −π
    else
        Θ_mn ← angle( (x[n−1] − ϰ[m]) / (x[n] − ϰ[m]) )
    end if
end for
```

This method essentially assembles the entire left-hand side of the integral equation, so that we may simply solve $\phi = \Theta^{-1}\mathtt{h}$.

Gauss–Legendre quadrature

To construct $\mathtt{h}$, we evaluate the integral of the logarithm using quadrature. For an integral on the real line, Gaussian quadrature approximates it by transforming it to the unit ball,

$$\int_a^b y(x)\,\mathrm{d}x \mapsto \int_{-1}^{1} \eta(\xi)\,\mathrm{d}\xi,$$

where $\eta(\xi) \equiv y\big(x(\xi)\big)\partial_\xi x$, a change of variables such that $x(-1) = a$ and $x(1) = b$. Gauss–Legendre quadrature of order K has us approximating this integral by

$$\sum_{k=1}^{K} w_k \eta(\xi_k), \quad w_k = \frac{2}{\big(1-{\xi_k}^2\big)\big(\mathrm{P}'_K(\xi_k)\big)^2},$$

where P_K is the K^{th} Legendre polynomial, ξ_k is its k^{th} zero, and w_k is the corresponding weight. The Legendre polynomials are given by the recursion formula[1]

$$(n+1)\,\mathrm{P}_{n+1}(\xi) = (2n+1)\xi\,\mathrm{P}_n(\xi) - n\,\mathrm{P}_{n-1}(\xi),$$

where we have defined $\mathrm{P}_0(\xi) = 1$, $\mathrm{P}_1(\xi) = \xi$. From the recurrence relation we find the 2nd order Legendre polynomial, its zeros, and its weights:

$$\mathrm{P}_2(\xi) = \frac{3x^2-1}{2}, \quad \xi_k = (-1)^k\frac{\sqrt{3}}{3}, \quad w_k = 1$$

We find below that we have no use for the 3rd polynomial, but we may use the 4th, for which we have

$$\mathrm{P}_4(\xi) = \frac{105\xi^4 - 90\xi^2 + 9}{24}, \quad \xi_k = \pm\sqrt{\frac{3 \pm 2\sqrt{{}^6\!/_5}}{7}}.$$

The zeros ξ_k are readily found by solving the biquadratic equation, the the signs may be chosen independently, yielding four roots. As for the wieghts, they are found after some algebraic manipulation, and turn out to be

$$w_k = \frac{18 \pm \sqrt{30}}{36}.$$

Instead of ginning up some elaborate mathematical expression to relate the roots to their respective weights, an illustration is provided in figure 2 below.

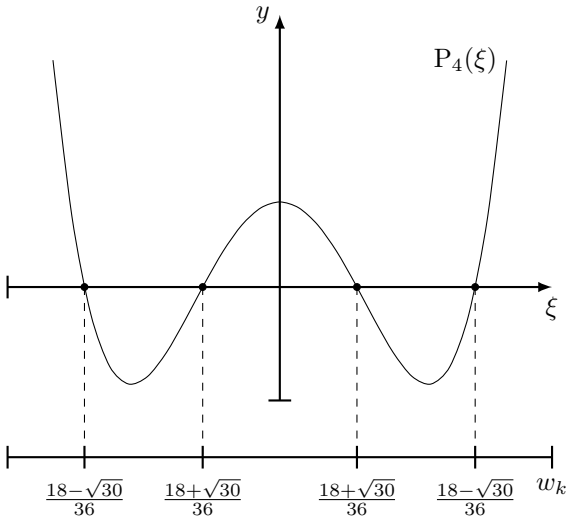

Figure 2: The fourth Legendre polynomial plotted, with the weights associated with its four roots indicated.

To calculate the target integral by quadrature, we may parametrize the variable of integration by writing that the line segment S_n is given by

$$\mathcal{Z}(\xi) = \left(\frac{\boldsymbol{x}_n - \boldsymbol{x}_{n-1}}{2}\right)\xi + \left(\frac{\boldsymbol{x}_n + \boldsymbol{x}_{n-1}}{2}\right),$$

where $\xi \in [-1, 1]$. We recognize the rightmost term above as $\boldsymbol{\varkappa}_n$, and for brevity, we write that $\boldsymbol{x}_n - \boldsymbol{x}_{n-1} = \delta\boldsymbol{x}$, which is implemented as the method `IntegralEquation.Δx`. Now $\ln|\mathcal{Z}(\xi)|$ clearly is a function from the real numbers into the

[1] [1] Abramowitz & Stegun, 22.7.10, p.782

real numbers, so we may use the fact that for any real valued function over the complex plane,

$$\int_C f(\mathfrak{z})\,\mathrm{d}S = \int_a^b f\big(\mathfrak{z}(\xi)\big)|\mathfrak{z}'(\xi)|\,\mathrm{d}\xi.$$

We calculate that $|\mathfrak{z}'(\xi)| = {}^1\!/_2|\delta\boldsymbol{x}|$, which is implemented as the class variable `dS` in the `Quadrature` class of `quadrature.py`. It should now be clear that we may not use an odd order quadrature scheme, as the diagonal of `h` will entail evaluating the logarithm at zero, since $\xi_k = 0$ will always be a root of an odd order LEGENDRE polynomial.

We again utilize the fact that $\operatorname{Re}\big(\log \mathfrak{z}\big) = \ln|\mathfrak{z}|$, so that

$$\int_{S_n} \ln r\,\mathrm{d}S = \frac{\delta\boldsymbol{x}}{2}\operatorname{Re}\int_{-1}^{1} \log\big(\mathfrak{z}(\xi) - \boldsymbol{эc}_m\big)\,\mathrm{d}\xi,$$

the latter of which we may employ quadrature on.

Assembly and solution

Recalling now our assumed property of independence of modes in the potential, that we may write it as the superposition $\Phi(t;\boldsymbol{x}) = \boldsymbol{U}(t)\cdot\boldsymbol{\phi}(\boldsymbol{x})$, we impose the boundary condition of impermeability—no fluid shall cross the boundary $\partial\Omega$. In other words, the fluid displaced by the movement of the body, must at the boundary itself move with that velocity, so that

$$\hat{\boldsymbol{n}}\cdot\nabla\Phi = \boldsymbol{U}\cdot\hat{\boldsymbol{n}}, \qquad \text{on } \partial\Omega.$$

By the product rule, and by virtue of $\boldsymbol{U}$ not being a function of the spatial variable, we have through the commutativity of the inner product that $\boldsymbol{U}\cdot\big(\hat{\boldsymbol{n}}\cdot\nabla\boldsymbol{\phi}(\boldsymbol{x})\big) = \boldsymbol{U}\cdot\hat{\boldsymbol{n}}$. In other words, we have the boundary condition that for each mode j of the potential,

$$\hat{\boldsymbol{n}}\cdot\nabla\phi_j \equiv \partial_{\hat{\boldsymbol{n}}}\phi_j = \hat{n}_j, \qquad \text{on } \partial\Omega,$$

whence the normal vector in (3). As is clear from equation (4), we may approximate get the normal vector on a straight line segment from implementing $\hat{\boldsymbol{n}}^n = (\delta z_n - i\delta x_n)|\delta\boldsymbol{x}|^{-1}$, where $\delta\boldsymbol{x} = \delta x_n + i\delta z_n$, as is done in the `normal_vector` method in the `IntegralEquation` class. The sixth normal vector, coinciding with rotation about the y-axis, is given by $\hat{n}_6{}^n = \boldsymbol{эc}_n \times \hat{\boldsymbol{n}}^n$, where the multiplication sign here indicates the cross product. The method `assemble_h` constructs the matrix representing the integral of the logarithm with a call to the `quad` method in the `Quadrature` class. The actual right-hand side of the integral equation is then calculated with a call to the method `right_hs`, choosing a mode, returning `right_hs` = `assemble_h@n_i`. Importing `linalg` from `numpy`, we may solve $\boldsymbol{\varphi}_j = \mathtt{\Theta}^{-1}\mathtt{h}$ for each of the modes. This is implemented in the `solve` method, constructing `Θ` and `h` only once, then calling `right_hs` three times, to return the tuple $[\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \boldsymbol{\varphi}_3]$.

Approximated added mass

Now having an approximation of $\boldsymbol{\phi}$, we may now approximate the integral in equation (1) as follows.

$$\mathfrak{m} = [m_{ij}] \approx \varrho\sum_{n=1}^{N} \phi_j{}^n\hat{n}_i{}^n\delta S_n$$

We are only interested in the substantive added mass, so we implement unit density. The added mass is then implemented as the `added_mass` method in the `IntegralEquation` class.

Added mass of a circle

Let a be the radius of the circle. We then have that its potential as a result of horizontal translation is given by

$$w = a^2U\mathfrak{z}^{-1} = \Phi + i\Psi, \quad \mathfrak{z} = x + iz.$$

We see that

$$\phi_1 = \operatorname{Re}\big(a^2U\mathfrak{z}^{-1}\big) = \frac{a^2U\cos\theta}{r^2}; \tag{5}$$

$$\phi_2 = \operatorname{Re}\big(a^2Ui\mathfrak{z}^{-1}\big) = \frac{a^2U\sin\theta}{r^2}. \tag{6}$$

By calculating the gradient of ϕ and transforming into polar coordinates, we find that

$$q_r = \frac{a^2U\cos\theta}{r^2}, \quad q_\theta = \frac{a^2U\sin\theta}{r^2}, \quad q^2 = \frac{a^4U^2}{r^4}.$$

We may now calculate the kinetic energy of the fluid by integrating over the entire fluid domain Ω,

$$T_{\text{fluid}} = \frac{\varrho}{2}\int_0^{2\pi}\int_a^\infty q^2 r\,\mathrm{d}r\mathrm{d}\theta = \frac{\pi\varrho a^2U^2}{2}.$$

Rotating the cylinder ought not induce drift in the fluid, as there is no mechanisms by which the fluid should be compelled to move from the circle turning. Because of the rotational symmetry of the circle, we must have that

$$\mathfrak{m} = [m_{ij}] = \begin{bmatrix} \pi\varrho a^2 & 0 & 0 \\ 0 & \pi\varrho a^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

We discretize the circle by defining

$$\mathtt{\Theta} = \mathtt{linspace}(0, 2\pi, \mathtt{N}+1), \quad \boldsymbol{x} = a\cos\mathtt{\Theta} + ia\sin\mathtt{\Theta}.$$

Implementing the potentials described in equation (5) and (6) in the `Potentials` class of `potentials.py`, we compare the numerical solution to the analytical potentials. In figures 3 and 4, we have used a fourth order quadrature scheme, and

we see that there is seemingly negligible difference in the numerical solution and the theory.

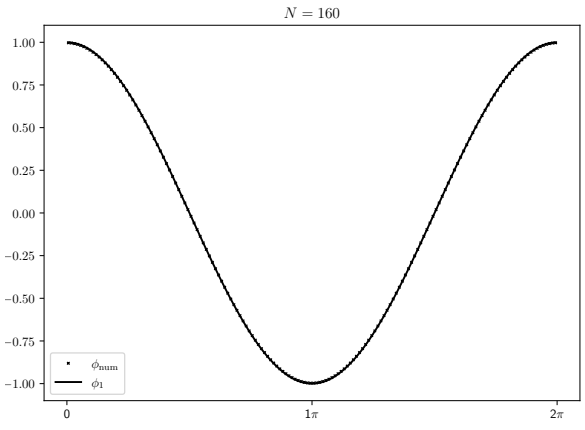

Figure 3: First mode of the potential for a circle using a fourth order quadrature scheme.

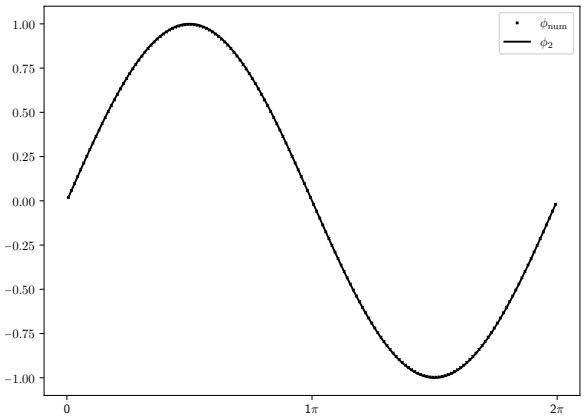

Figure 4: Second mode of the potential for a circle using a fourth order quadrature scheme.

The difference in accuracy for 2^{nd} and 4^{th} order quadrature schemes seems to be negligible. The L^2 error in the potentials is tabulated below in table 1, showing that the difference is at most in the thousandths.

	2^{nd}		4^{th}	
N	ϕ_1	ϕ_2	ϕ_1	ϕ_2
32	0.1777	0.1727	0.1743	0.1690
64	0.0897	0.0884	0.0876	0.0862
96	0.0599	0.0594	0.0584	0.0579
128	0.0450	0.0447	0.0439	0.0435
160	0.0360	0.0358	0.0351	0.0349

Table 1: L^2 error differences in potential functions of a circle between second and fourther order Gauss–Lagrange quadrature.

We measure the error in the added mass absolutely, so that we just measure the absolute difference in the added mass. Since we set the the radius of the circle to be 1, a maximal difference of less than 0.05, as is indicated in figure 5, is fine, considering the added mass would simply be π.

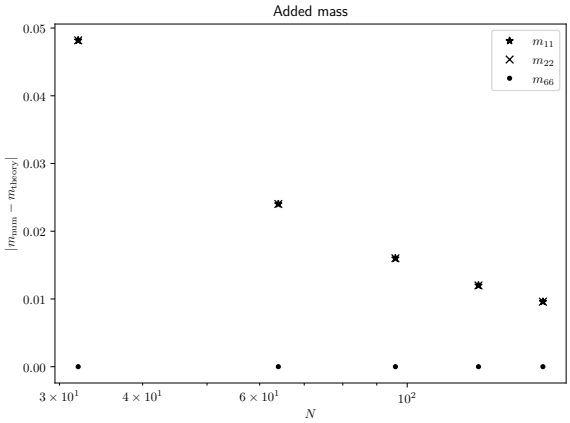

Figure 5: Absolute difference between the theoretical added mass m_{theory} and the calculated added mass m_{num}

We see that the convergence is pretty rapid, and that the code is performant.

Added mass of an ellipse

Elliptic integrals

The incomplete elliptic integral of the first kind

$$\mathrm{F}(\phi|k) = \int_0^{\phi} \frac{1}{\sqrt{1 - k\sin^2\theta}}\,\mathrm{d}\theta \tag{7}$$

The complete elliptic integral of the first kind

$$\mathrm{K}(k) = \mathrm{F}(\pi/2|k) = \int_0^{\pi/2} \frac{1}{1 - k\sin^2\theta}\,\mathrm{d}\theta \tag{8}$$

The incomplete elliptic integral of the second kind

$$\mathrm{E}(\phi|k) = \int_0^{\phi} \sqrt{1 - k\sin^2\theta}\,\mathrm{d}\theta \tag{9}$$

The arithmetic-geometric mean

Discretizing the ellipse

There are multiple ways to approach discretizing the ellipse, the most apparent of which being

$$\boldsymbol{x} = a\cos t + b\sin t, \qquad t \in [0, 2\pi), \tag{10}$$

where the parameter is *not* a polar variable. It is indeed related, as will be shown.

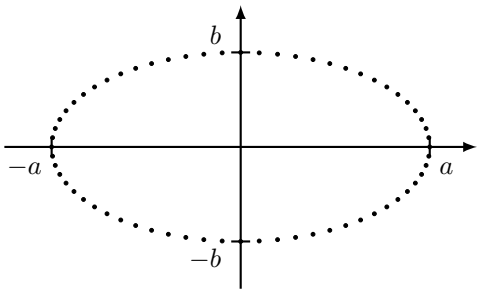

Figure 6: Ellipse parametrized with equidistant spacing in the perimeter.

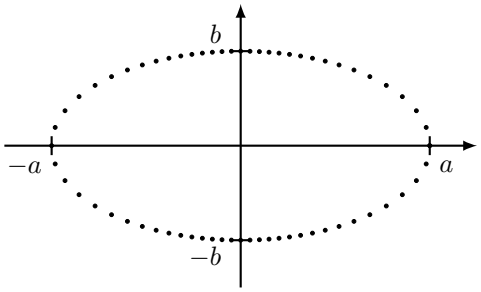

Figure 7: Ellipse parametrized with equiangular spacing.

Added mass of a square

Å skulle diskretisere kvadratet er ikke like trivielt som sirkelen eller ellipsen. Vi ønsker å skulle kunne gjenbruke samme klasse for integrallikningen, så en polar implementering av diskretiseringen er mest opplagt. En kontinuerlig mulighet er en såkalt superellipse, som kan diskretiseres med

$$\boldsymbol{x}(\theta) = \left(|\cos\theta|^{2/N} \operatorname{sgn}(\cos\theta), |\sin\theta|^{2/N} \operatorname{sgn}(\sin\theta)\right),$$

som vil konvergere raskt mot et kvadrat når N blir stor. Problemet her er at θ ikke faktisk er vinkelen i parametriseringen. Parametriseringsvariablen vil bruke lang tid i hjørnene på superellipsen, og en enorm andel av normalvektorene langs randa vil derfor være fullstendig feil.

En enklere løsning vil være å heller kjøre θ gjennom `if`-sjekker, som følger.

Algorithm 2 Konstruer kvadrat

```
for n ≤ N do
    if θ_n ∈ [−π/4, π/4) then
        x_n, y_n ← 2a, 2a tan(θ_n)
    else if θ_n ∈ [π/4, 3π/4) then
        x_n, y_n ← 2a sec(θ), 2a
    else if θ_n ∈ [3π/4, 5π/4) then
        x_n, y_n ← −2a, −2a tan(θ_n)
    else if θ_n ∈ [5π/4, 7π/4) then
        x_n, y_n ← −2a sec(θ), −2a
    end if
end for
```

Denne løsningen er ikke veldig vakker, men den forsikrer at alle nodene faktisk ligger på kvadratet. Dette kan ikke sies for hjørnene i $\boldsymbol{x}$, ettersom hjørnenodene kan havne innenfor randa i dersom ikke enten $\boldsymbol{x}_p$ eller $\boldsymbol{x}_m$ ligger akkurat i hjørnet for denne N-verdien. Vi ser i figur 8 at dette kan føre til merkelige mønstre i konvergensen til den adderte massen, hvor tilnærmingen faktisk kan bli dårligere for en større N-verdi.

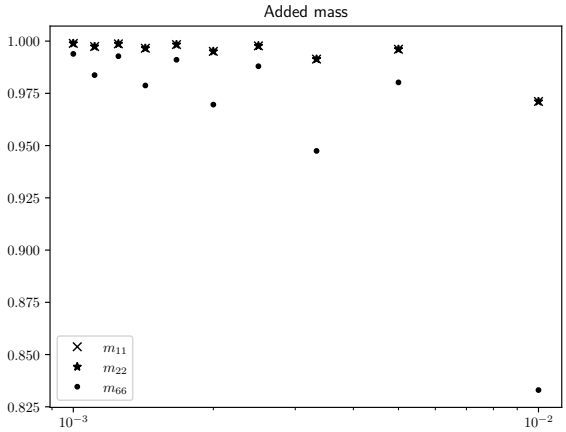

Figure 8: Numerisk utregnet over teoretisk addert masse på et kvadrat.

Samtidig ser vi at den utregnede adderte massen fakitsk konvergerer. Vil man unngå å se denne unøyaktigheten, kan man forsikre seg at $\{\pi/4, 3\pi/4, 5\pi/4, 7\pi/4\} \in \theta$ ved å la N alltid være delelig på 8, slik som i figur 9.

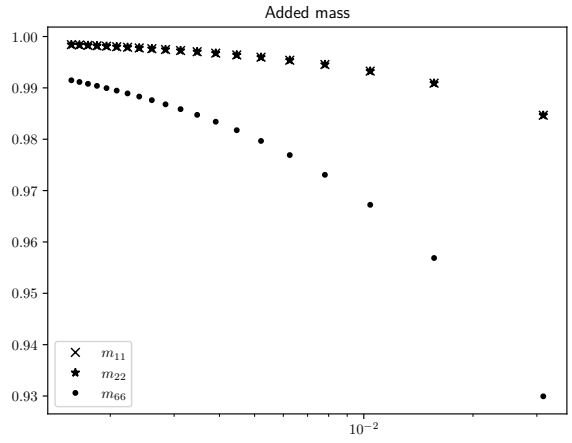

Figure 9: Numerisk utregnet over teoretisk addert masse på et kvadrat med N delelig på 8.

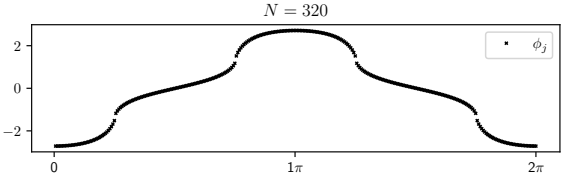

Figure 10: ϕ_1 er gitt numerisk med punkter.

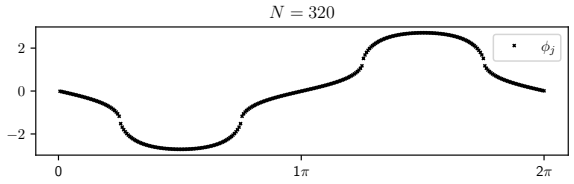

Figure 11: ϕ_2 er gitt numerisk med punkter.

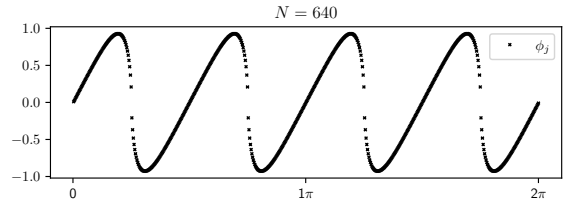

Figure 12: ϕ_6 er gitt numerisk med punkter.

References

[1] Abramowitz, Milton and Stegun, Irene A. *Handbook of Mathematical Functions*. 3rd ed. National Bureau of Standards, 1965.

[2] Darwin, Charles Galton. "Note on Hydrodynamics". In: *Proceedings of the Cambridge Philosophical Society* 49 (1953), pp. 342–354.

[3] Lavrentev, Mihail Alekseyevič (Лаврентьев) and Šabat, Boris Vladimirovič (Шабат). *Methoden der komplexen Funktionentheorie*. VEB Deutscher Verlag der Wissenschaften, 1967.

[4] Markuševič, Aleksey Ivanovič (Маркушевич). *Theory of Functions of a Complex Variable, Volume II*. Ed. by Richard A. Silverman. Prentice-Hall, inc., 1965.

[5] Milne–Thomson, Louis Melville. *Theoretical Hydrodynamics*. Macmillan & co. ltd., 1968.