Concerning the Stability of Complex Time Steppers

Cian J. Duggan

August 19, 2024

Contents

1 Introduction

A paper by Lloyd N. Trefethen [1] offers two definitions of numerical analysis:

> *The study of rounding errors.*
>
> *The study of algorithms for the problems of continuous mathematics.*

Trefethen argues that the first definition, though an accurate summation of the field's history, does not serve to entice the curious to explore the field. He proposes the second as a more compelling definition, one that emphasises the field's role in solving real-world problems, and inspires curiosity.
He offers an optimistic view of the field, one that is not bogged down by the minutiae of rounding errors, but rather one that is focused on the algorithms that make numerical analysis possible. More to ground, the majoirity of numerical analysis is concerned with the speed of convergence and minimization of error.
Through my work on this paper, I have honed my own definition:

> *Numerical analysis is the field which attempts to straddle the gaps between the discrete and the continuous.*

In this paper we will take a look at the stability of various numerical methods when applied to problems of differential calculus.
In particular, we wish to:

- Introduce key terminology and concepts in the field of numerical analysis.
- Establish the Exponential Decay Problem as the toy problem for analysis.
- Define stability, and its associated concepts, in the context of numerical analysis.
- Establish the concept of Complex time steps for numerical methods.
- Develop a framework for the analysis of the stability of numerical methods.
- Apply this framework to particular numerical methods.
- Expand upon the results of this to contrast the stability regions for Real and Complex time steps.
- MORE?

2 Definitions and Concepts

2.1 Numerical Methods

Numerical methods are techniques used to approximate solutions to problems that cannot be solved exactly.
Numerical methods are prevalent when working with problems of differential calculus in a computational context, where many problems do not have closed-form solutions.
A numerical method can be interpreted as an alogorithm; a series of steps that can be followed to approximate a solution.
Many numerical methods exist, each with their own properties and tradeoffs.
In this respect, different methods may be more or less suitable for different problems.
Core to understanding which to use for a particular problem are the concepts of *error* and *stability*.
Before discussing these properties, however, we must understand how a numerical method works.
The basis of many numerical methods is the concept of a *time step*.

2.2 Time Steps

Let's take a step back to when we first studied derivatives.
Newton's Difference Quotient should be familiar: $f'(x) = \lim_{h \to 0} \frac{f(x+h)-f(x)}{h}$
Viewing Newton's Difference Quotient in the context of numerical methods, we call h a *time step*.
Due to computational constraints, we cannot take h to be infinitesimal. (A computer's memory is as small as it is big)
Instead, we take h to be a small, finite number.
This defines the concept of a *step size*.

For numerical methods, we use $y(t)$ to denote the solution to a differential equation at time t.
We approximate $y'(t)$ by $\frac{y(t+h)-y(t)}{h}$ using Newton's Difference Quotient.
Note this is an approximation because we have dropped the limit; h is not infinitesimal.
This gives $y(t+h) \approx y(t) + hy'(t)$; a first-order approximation of y a small time step h in the future.
For a numerical method we say $y(t+h) = y(t) + hy'(t)$

2.3 Error

In numerical analysis, *error* is the difference between the numerical solution and the exact solution to a problem.
Error can be caused by many factors, such as the choice of numerical method, the choice of time step, or the precision of the computer.
Error can be classified into two categories: truncation error and rounding error.
Truncation Error is the error introduced by approximating a problem, such as using a finite time step.
Rounding Error is the error introduced by the finite precision of a computer.
Error is a crucial concept in numerical analysis, as it determines the accuracy of a numerical method.
Any error mentioned in this document refers to the truncation error; we won't be looking at how comupters run these calculations and the rounding errors that come with it.

2.4 Stability

In numerical analysis, a method is said to be *stable* if small deviations in the input do not lead to large perturbtions in the output. In the context of differential equations, a method is said to be stable if the solution does not grow to be unbounded as the number of time steps increases. (This, of course, only applies if the exact solution is bounded. We will restrict ourselves to this with our analysis.)
When solving differential equations numerically, the choice of time step is crucial. If the step size is too large, the solution may become unstable; the numerical solution will diverge from the analytic solution in proportion with the number of steps. If the step size is too small, the solution may be accurate and stable, but the computation may be too slow.
The tradeoff between error, stability and compute is a common theme in numerical analysis.
The sweet spot for maximal efficiency depends entirely on the choice of numerical method.
This marks the core motivation for this report; solidifying an understanding to better inform the choice of method and step size for a given problem.

3 The Exponential Decay Problem

We would like a simple toy problem with which we can build a framework for the analysis of the stability of a numerical method.
We turn to a classic problem in numerical analysis: the Exponential Decay Problem.
Consider a quatity y that decays relative to its current value.
The quantity at time t is $y(t)$ and the rate of decay is λ.
For the solution to actually decay, we require $Re(\lambda) < 0$.

The Exponential Decay Problem is given by the ODE

$y'(t) = \lambda y(t) \quad \text{with} \quad y(0) = y_0$

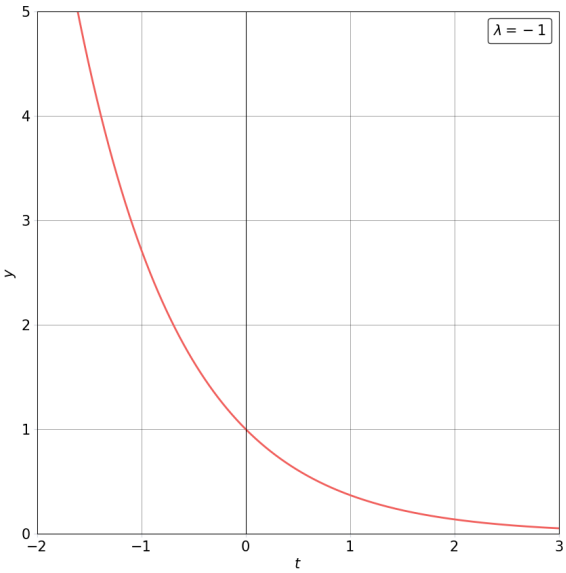
This has the exact solution $y(t) = y_0\, e^{\lambda t}$, hence the name 'Exponential Decay'.

For the sake of keeping things neat, we will take $y_0 = 1$ for the rest of this paper.
This poses no lack of generality, as we can always scale the solution by any initial value, both exact and numerical.

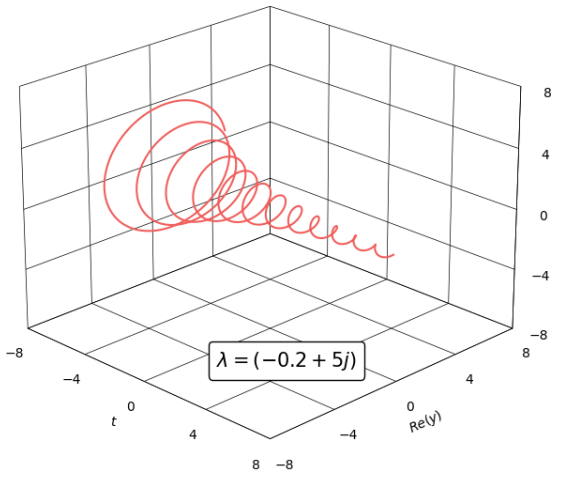
The exact solution we will consider is

$y(t) = e^{\lambda t}$

3.1 A Graphical Representation



If we restrict λ to $\mathbb{R}^-$, we have a simple expoential curve that decays to zero as $t \to \infty$.



If we allow $\lambda \in \mathbb{C}$ and $Re(\lambda) < 0$, we get a spiral in the complex plane, the radius of which decays to zero as $t \to \infty$.

3.2 The Macaulin Series

The Taylor series is given by:

$$y(t) = \sum_{n=0}^{\infty} \frac{y^{(n)}(a)}{n!}(t-a)^n$$

Setting $a = 0$, we get the Macaulin series for the exact solution:

$$y(t) = \sum_{n=0}^{\infty} \frac{(\lambda)^n e^0}{n!} t^n = \sum_{n=0}^{\infty} \frac{(\lambda t)^n}{n!} \quad \text{which we will denote } M(y)$$

This is a useful form for the exact solution – we will see it pop up again later.

3.3 Proponents of the Exponential Decay Problem

There are a few pros to this choice of problem:

- The exact solution is known and easily computed; it can be used to evaluate a numerical solution.
- The solution is a straigtforward and well understood function.
  This simplicity allows us to focus on the numerical method.
- The problem is linear in $y(t)$, making the problem simple to work with.
  Again, our focus can stay on the numerical method.
- The graph of the exact solution is easy to visualise; the more chaotic outputs of the numerical methods can be compared to the smooth curve of the exact solution easily.
- Exponential decay is a model for many physical processes, including radioactive decay, population decline, and capacitor discharge.
  Quantities that decay relative to their current value appear frequntly in nature.
  This means readers from many different disciplines may already posses an intuition for the problem.
- The problem is simple to generalise to complex numbers, allowing us to explore the stability of numerical methods in the complex plane.
- The Exponential Decay Problem is *stiff*; for certain method-step pairs, the numerical solution may oscillate wildly.
  This is exactly the kind of behaviour we want to avoid in a numerical method, when we are looking for stability.
- Generalisation. WRITE MORE

Normally, when using the Exponential Decay Problem in a numerical analysis context, we would allow $\lambda \in \mathbb{C}$ with $Re(\lambda) < 0$.
However, for the purposes of this paper, we will restrict λ to $\mathbb{R}$ as we will be looking at $h \in \mathbb{C}$ and we don't want to overcomplicate things.
Unless otherwise mentioned, we will assume that $\lambda \in \mathbb{R}^-$ for the rest of this paper.

We will use the Exponential Decay Problem throughout this paper to derive a useful framework for the analysis of the stability of a numerical method.

4 Stability Analysis

4.1 Introduction

In this section we will lay out a framework for analysing the stability of a numerical method.
To begin, we will introduce the concept of the stability function.
We will show how it can be used to define a stability region.
We will explore the stability region as the set of values for which a numerical method is stable.
Finally, we will extend this framework to view the stability of a numerical method in the context of complex time steps.

4.2 Numerical Method: Forward Euler

Let's consider a simple numerical method applied to the Exponential Decay Problem.
The Forward Euler method is a first-order numerical method for solving ODEs with a given initial condition.
Its algorithm can be defined as:

$$y(t_{j+1}) = y(t_j) + hy'(t_j)$$

Where h is the time step and y_j is the numerical solution at time t_j, j steps on from $t_0 = 0$.
This means $t_j = hj$.

For the Exponential Decay Problem, the Forward Euler method can be written as:

$$y_{j+1} = y_j + h\lambda y_j \quad \text{where} \quad y_0 = 1$$

This gives us the following algorithm:

$$y_{j+1} = (1 + h\lambda)y_j$$

This gives an approximation of the exact solution at time t_j as follows:

$$y_j = (1 + h\lambda)^j y_0 = (1 + h\lambda)^j = y(t_j) = y(hj) \approx e^{\lambda h j}$$

We can see that the Forward Euler method is stable if $|1 + h\lambda| < 1$;
both the exact solution and our approximation will decay to zero as $t \to \infty$.
The stability is dependent on the time step h and the value of λ.
We can write this as $s(\lambda, h) = 1 + h\lambda$.
By analysing s for different values of λ and h,
we can infer the stability of the Forward Euler method for the Exponential Decay Problem.
We call s the *stability function* of the Forward Euler method.

4.3 The Stability Function and corresponding Stability Region

By the same methodology, we can define the stability function for any numerical method by writing the algorithm in the form $y_{j+1} = s(\lambda, h)y_j$.
$s(\lambda, h)$ is the *stability function* of the numerical method with a time step h.
Note: This is equivalent to $y_j = s(\lambda, h)^j y_0$
The *stability region* of a numerical method is the set $S = \left\{ (\lambda, h) \;\middle|\; |s(\lambda, h)| < 1 \right\} \subset \mathbb{C}$
This follows from the definition of stability for our Exponential Decay Problem;
a method is stable if the numerical solution decays to zero as $t \to \infty$.
Clearly, $\lim_{j\to\infty} y_j = \lim_{j\to\infty} s(\lambda, h)^j y_0 = 0 \iff |s(\lambda, h)| < 1$. Now, let's explore some examples of stability regions for different numerical methods.

4.3.1 Stability Region for Euler's Forward Method

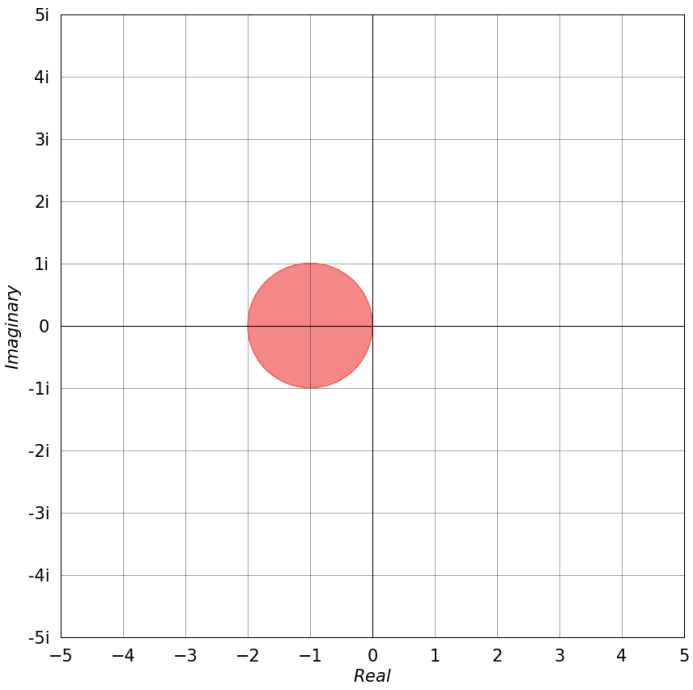
Euler's Forward Method has the stability function

$$s(\lambda, h) = 1 + \lambda h = \sum_{n=0}^{1} \frac{(\lambda h)^n}{n!} = M(y) + \mathcal{O}((\lambda h)^2)$$

The corresponding stability region is

$$S = \left\{ (\lambda, h) \;\middle|\; |1 + \lambda h| < 1 \right\}$$

We can see this region plotted in red on the right; an open unit circle centred at -1.



For a given $\lambda \in \mathbb{C}$, the method is stable for any step-size $h \in \mathbb{R}$ such that $|1 + \lambda h| < 1$.

Expanding $\lambda = a + bi$, we get the restriction

$$|1 + (a + bi)h| < 1 \quad \implies \quad |(1 + ah) + (bh)i| < 1$$

$$\implies \quad a^2h^2 + 2ah + b^2h^2 < 0$$

$$a, b, h \in \mathbb{R} \implies (a^2 + b^2)h^2 > 0$$

$$\implies 2ah < 0 \text{ and } ||2ah|| > (a^2 + b^2)h^2$$

This aligns with our intuition for the problem;

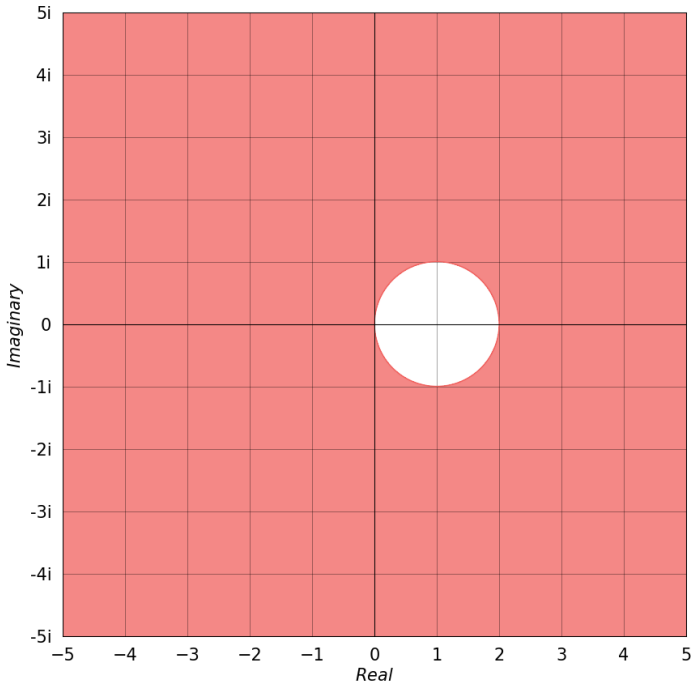
- For the exponential to decay, we must have $Re(\lambda) = a < 0$.
- h must be positive, as it is a time step.

Moreover,

$$0 < (a^2 + b^2)h^2 < ||2ah|| \quad \implies \quad 0 < h < \frac{||2a||}{a^2 + b^2} \quad \implies \quad 0 < h < \frac{2||Re(\lambda)||}{|\lambda|^2}$$

This is also intuitive; if we imagine λ growing linearly, h must shrink quadratically so that their product λh stays inside the stability region.

4.3.2 Stability Region for Euler's Backward Method



Euler's Backward Method can be written as

$$y_{j+1} = y_j + h.y'(t_{j+1})$$

$$\implies y_{j+1} = y_j + h(\lambda y_{j+1}) \implies y_{j+1} = \frac{1}{1 - \lambda h} y_j$$

Thus, the stability function is

$$s(\lambda, h) = \frac{1}{1 - \lambda h}$$

The corresponding stability region is

$$S = \left\{ (\lambda, h) \;\middle|\; \left| \frac{1}{1 - \lambda h} \right| < 1 \right\}$$

This is plotted in red on the left; the region outside a unit circle centred at 1.
The white region of instability is exactly the stability region for Euler's Forward Method, flipped about Imaginary axis.

We could run through the same algebraic proceedure as before, expanding λ and rearranging, and we would find that

$$h > \frac{2Re(\lambda)}{|\lambda|^2}$$

This is a more lax restriction than the $0 < h$ that we have already established.
In fact, this tells us that any positive h will give a stable solution, regardless of the value of λ.
This is a property called *Absolute Stability* or *A-Stability*.
This is often characterised by a stability region that covers the entire left half-plane, as we see here.

4.3.3 Stability Region for Runge-Kutta 4

Runge-Kutta 4 can be written as

$$y_{j+1} = y_j + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Where $\phi(t, y) = y'(t) = \lambda y$

$$k_1 = \phi(t_j, y_j) \qquad k_2 = \phi(t_j + \frac{h}{2}, y_j + \frac{h}{2}k_1)$$

$$k_3 = \phi(t_j + \frac{h}{2}, y_j + \frac{h}{2}k_2) \qquad k_4 = \phi(t_j + h, y_j + hk_3)$$

For the Exponential Decay Problem, we get

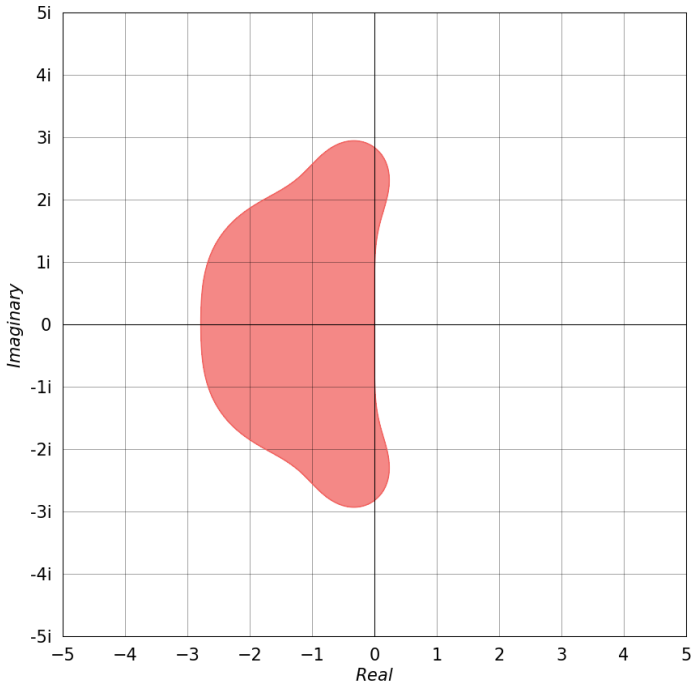
$$y_{j+1} = (1 + \lambda h + \frac{(\lambda h)^2}{2} + \frac{(\lambda h)^3}{6} + \frac{(\lambda h)^4}{24})y_j$$

The stability function is

$$s(\lambda, h) = \sum_{n=0}^{4} \frac{(\lambda h)^n}{n!} = M(y) + \mathcal{O}((\lambda h)^5)$$

The corresponding stability region is

$$S = \left\{(\lambda, h) \;\middle|\; \left|\sum_{n=0}^{4} \frac{(\lambda h)^n}{n!}\right| < 1\right\}$$



4.4 Interpretation of Stability Regions

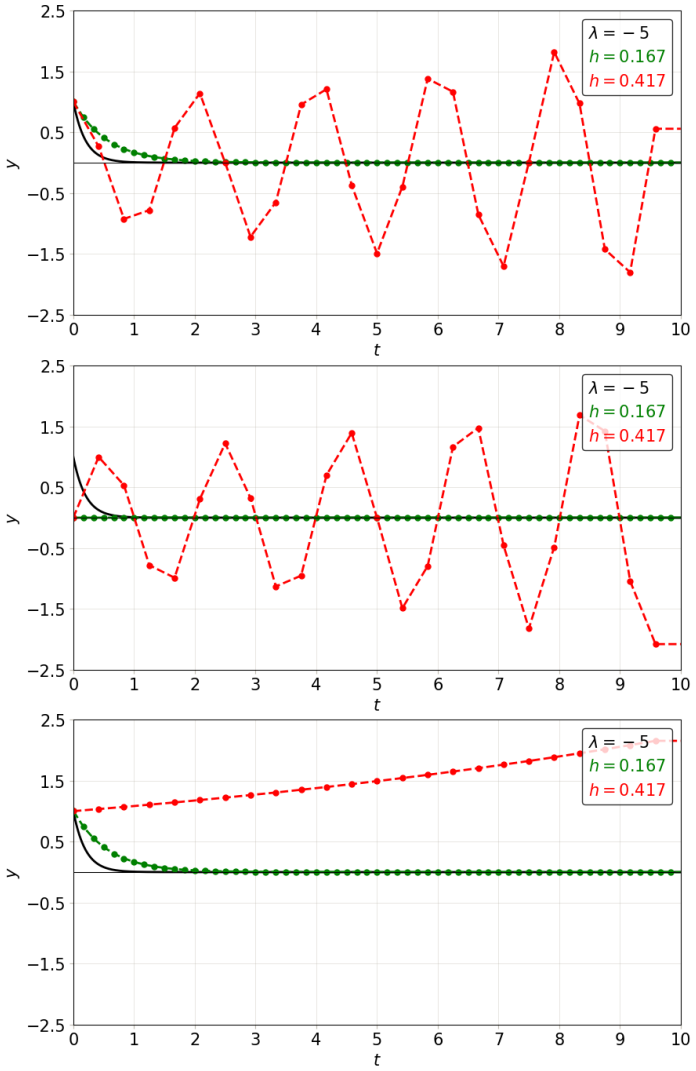
The stability region $S = \left\{(\lambda, h) \;\middle|\; |s(\lambda, h)| < 1\right\} \in \mathbb{C}$ is a set of values for which a numerical method is stable for the Exponential Decay Problem.
We always restrict $h \in \mathbb{R}^+$, as it is a time step.
We have two separate cases for λ:

- $\lambda \in \mathbb{R}^-$: S corresponds to $\left\{s(\lambda, h)^2 < 1\right\}$. Of course, $\lambda, h \in \mathbb{R} \implies S \subset \mathbb{R}$.
- $\lambda \in \mathbb{C} \setminus \mathbb{R}$ with $Re(\lambda) < 1$: S corresponds to $\left\{\left(s(\lambda, h)\right)\left(\overline{s(\lambda, h)}\right) < 1\right\}$.

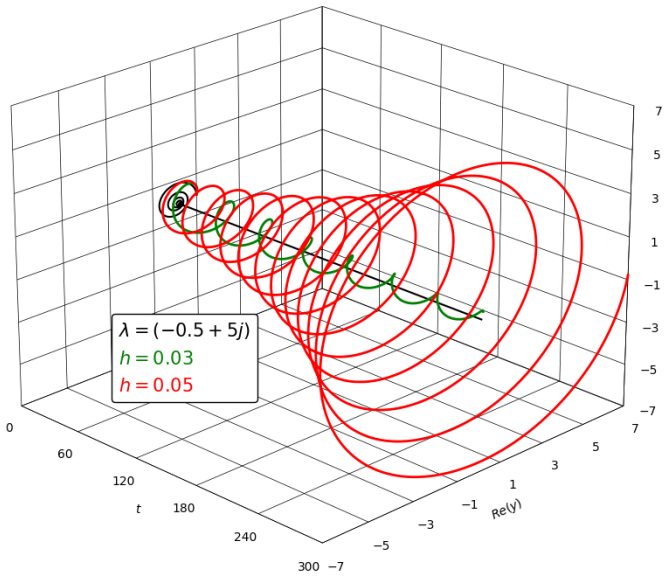
From here, one can derive bounds on h for a given value of λ, or vice versa.
We did this with Euler's Forward and Backward methods above.
The Runge-Kutta 4 method is more complex, as the two cases of λ each give a polynomial of degree 8.

Below we have graphs for the cases where λ is real or complex,
the same as we introduced for the Exponential Decay Problem.
In black is the exact solution. In colour are the numerical solutions due to the method in question.
These (λ, h) pairs were found by first picking a value of λ and then finding h values which met the stability condition.
In green $(\lambda, h) \in S$.
In red $(\lambda, h) \notin S$.

4.4.1 Euler's Forward Method



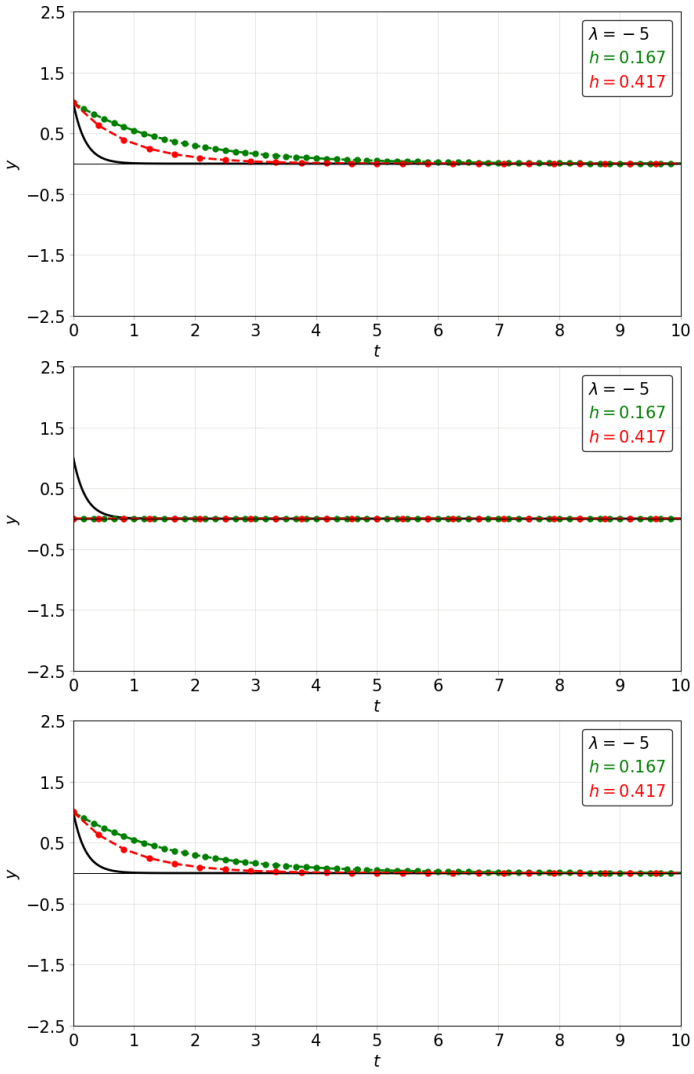
On the left are the aforementioned graphs for
Euler's Forward Method with $\lambda \in \mathbb{R}^-$.
We have 3, as for the unstable $h = \frac{5}{12}$ the method gives complex outputs.
From the top down, the graphs show the real part, the imaginary part and the magnitude of the numerical solution.
In all 3, the black is real, and only, part of the exact solution.
We can notice for the imaginary part that the green, stable $h = \frac{1}{6}$ is 0; this is strictly real.

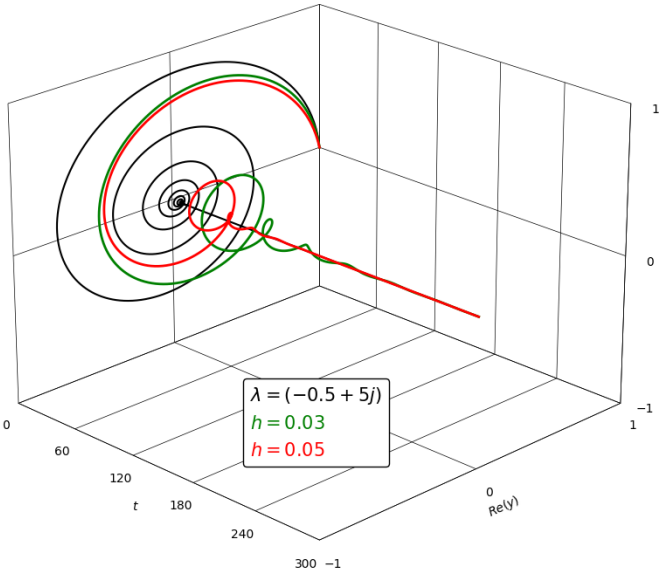


On the right is the case where $\lambda \in \mathbb{C}$. The exact solution decays quickly in black, very close to the $t = 0$ plane.
The green, valid h value converges as t increases, but the red, invalid h value diverges.
While this 3D graph might not be the clearest, it took quite a lot of messing with values of h and λ to find one so demonstrative.
The reader is encouraged to play with '/Python/Exponential Decay/Exact vs Method.py' in the GitHub repository [3] to see for themselves.
This has the added benefit of allowing you to rotate the graph to see the behaviour from different angles.

4.4.2 Euler's Backward Method

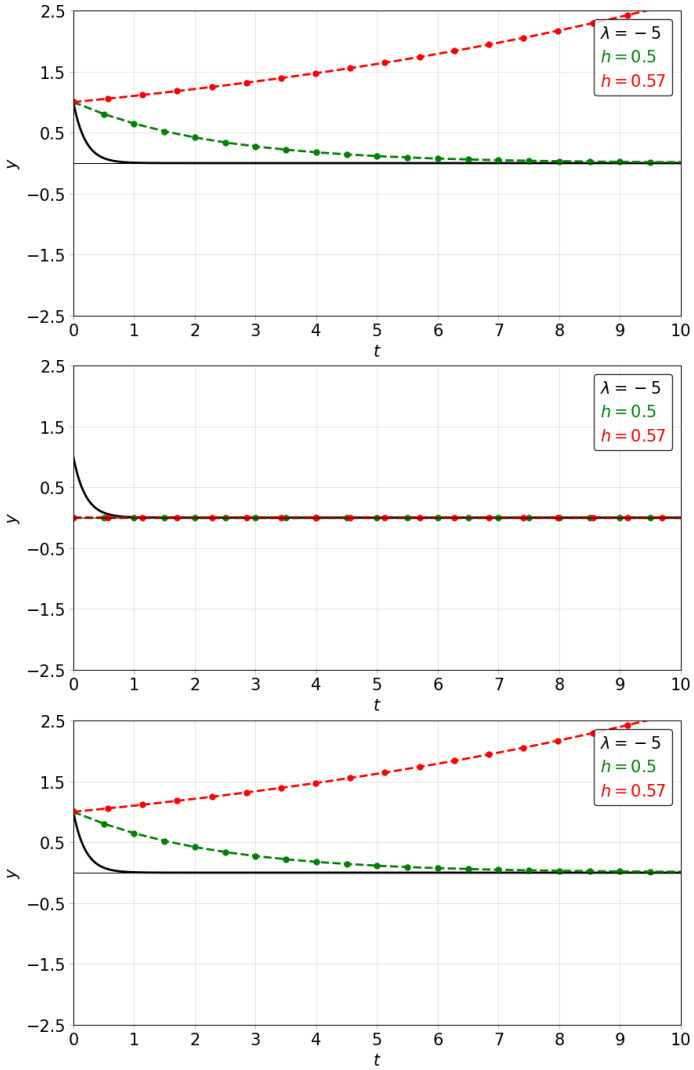
As mentioned before, Euler's Backward Method is A-Stable.
This means that for any $\lambda \in \mathbb{C}$, the method is stable for any $h \in \mathbb{R}^+$.
As a result, there is no choice of (λ, h) that will give us a divergent graph.
What happens if we relax our restrictions on h and λ to attain a value in our white circle of instability, you ask?
If the λh value is in the white circle, we are looking at a growing exponential. Is cuma liom. Instead, these graphs show two different valid h values for the same λ value.
These use the same values as we had for Euler's forward Method.

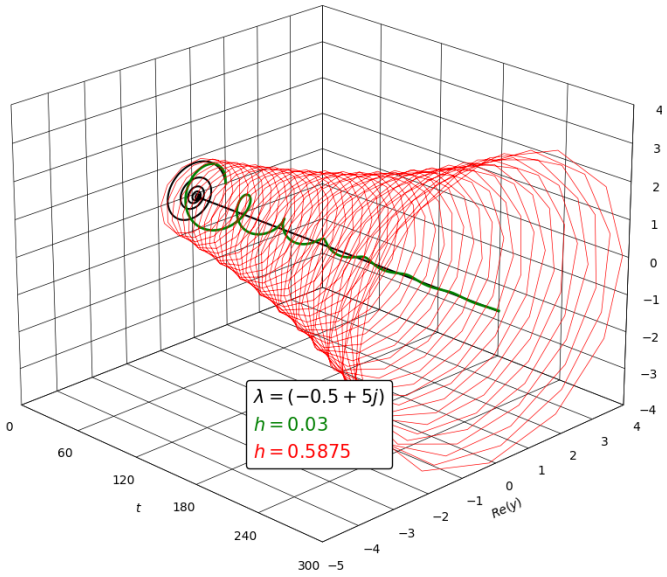




In the $\lambda \in \mathbb{R}$ case, the numerical solution converges to the exact solution for both step sizes.
On the second, imaginary, graph, we can see that both numerical methods are constantly 0 as they are strictly real. The magnitude graph is identical to the real graph, indicating the same behaviour.
In the case where $\lambda \in \mathbb{C}$, the method converges to the exact solution for both step sizes for both real and complex λ.
There is no divergence, as we have already established.
Interestingly, the method converges faster for the larger step size.
Should we explore this further? Like, this makes sense if you think about the stability eq, but it is also counter intuitive. I have graphs showing not only the stability regions, but the 'density' for certain step sizes

4.4.3 Runge-Kutta 4





4.5 Numerical Method: 2-step Abysmal Kramer-Butler Method

This method was designed to be unstable for the sake of demonstration.
The algorithm is as follows:

$$y_{j+1} = y_{j-1} + h(4\phi(t_j, y_j) - 2\phi(t_{j-1}, y_{j-1}))$$

As we know the exact solution to our Exponential Decay problem is $y(t) = e^{\lambda t}$, we know $\phi(t, y) = \lambda y$.
This gives:

$$y_{j+1} = y_{j-1} + h(4\lambda y_j - 2\lambda y_{j-1})$$

Simplified:

$$y_{j+1} = 4\lambda h y_j + (1 - 2\lambda h) y_{j-1}$$

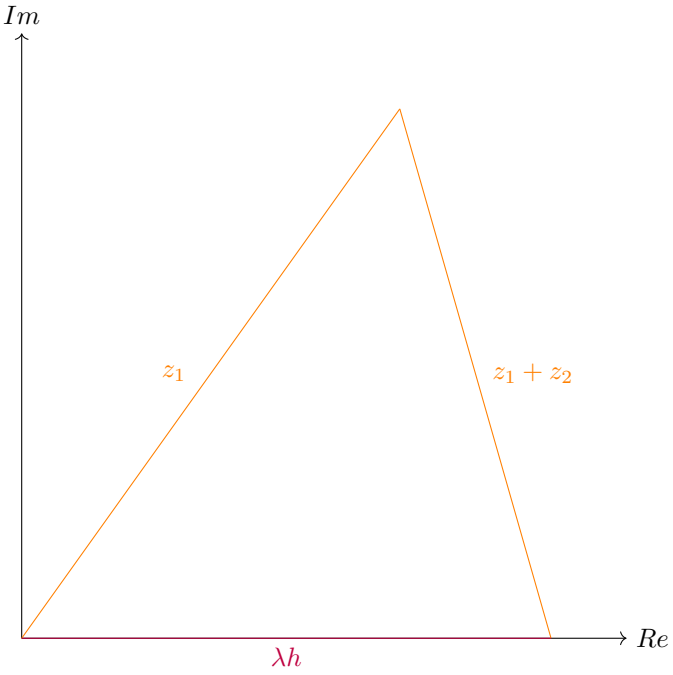
5 Complex Time-Steps

So far, we have restricted ourselves to time-steps in the real domain.
It has often proven useful to extend the analysis of a problem to the complex domain.
Motivated by a paper by *George, Yung and Mangan*[2], we will have a look at the stability of numerical methods when the chosen time-step is complex.
They state:

> *Most numerical methods for time integration use real time steps. Complex time steps provide an additional degree of freedom, as we can select the magnitude of the step in both the real and imaginary directions. By time stepping along specific paths in the complex plane, integrators can gain higher orders of accuracy or achieve expanded stability regions.*

5.1 Complex 2-Step

We want to take an overall step of size $\lambda h \in \mathbb{R}$ comprised of two steps $z_1, z_2 \in \mathbb{C}$ such that $z_1 + z_2 = \lambda h$.
We can set $z_1 = a + bi$ and $z_2 = \lambda h - a - bi$.
For analysing our numerical methods, we require a comparable implementation of the real step setup.



A real 2-step method where we take a step of size $\frac{\lambda h}{2}$ allows us to take two steps and go the same distance as the complex 2-step method.
We can write this real 2-step method as $y_{j+1} = s(\lambda, \frac{h}{2})^2 y_j$.
Note: We can write this explicitly as

$$y_{j+1} = s(\lambda, \frac{h}{2}) y_{j+\frac{1}{2}} = s(\lambda, \frac{h}{2}) s(\lambda, \frac{h}{2}) y_j$$

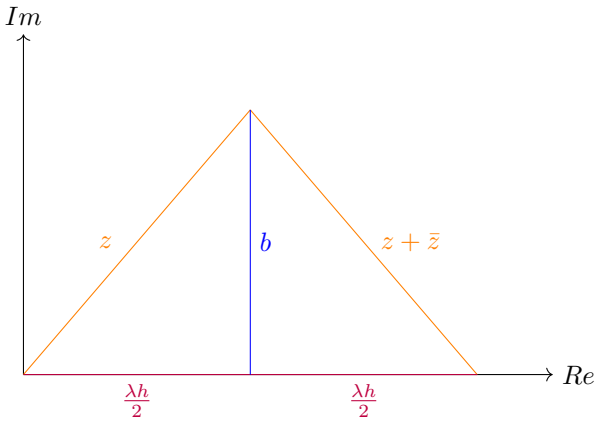
We can write the complex 2-step method as

$$y_{j+1} = s(\lambda, z_1) s(\lambda, z_2) y_j$$

Note: We can write this explicitly as

$$y_{j+1} = s(\lambda, z_1) y_{j+\frac{1}{2}} = s(\lambda, z_1) s(\lambda, z_2) y_j$$

5.2 Complex Conjugate Pairs



When we restrict the exponential decay problem by $\lambda \in \mathbb{R}$, we also find that $s(\lambda, z_1) s(\lambda, z_2) \in \mathbb{R}$
This is because any value attained by $e^{\lambda t}$ is real.
This restricts the values of z_1 and z_2, as any imaginary terms in $s(\lambda, z_1) s(\lambda, z_2)$ must cancel out.

One particularly simple family of such complex pairs is defined by setting $a = \frac{\lambda h}{2}$
This necessitates $z_2 = \bar{z_1}$
We will refer to this family as the *complex conjugate step pairs*.

5.3 Stability of 2-Step Methods for Complex Conjugate Step Pairs

In this section we will compare the stability regions for the real and complex 2-step methods.
The real step size is given by $\frac{\lambda h}{2}$
The complex conjugate step pair is given by $z = \frac{\lambda h}{2} + bi$ and $\bar{z} = \frac{\lambda h}{2} - bi$
The plots in this section show the regions of stability $S = \{h : |s(\lambda, h)| \leq 1\}$ for the case where $b = a = \frac{\lambda h}{2}$
We will have a look at varying b in a subsequent section.

5.3.1 Euler's Forward 2-Step

Theorem 5.1. ***Euler's Forward Complex Step Pairs must be Conjugate for*** $\lambda \in \mathbb{R}$

For the exponential decay problem, $y = e^{\lambda t}$*, let* $\lambda \in \mathbb{R}$
Let $z_1 = a_1 + b_1 i$ *and* $z_2 = a_2 + b_2 i$ *be a complex step pair for Euler's Forward 2-Step Method.*
Then we must have $z_2 = \bar{z_1}$

Proof:

$\lambda \in \mathbb{R} \implies y_j, y_{j+1} \in \mathbb{R}$

Euler's Forward 2-Step Method is given by:

$$\begin{aligned} y_{j+1} &= s(\lambda, z_1)s(\lambda, z_2)\ y_j \\ &= \Big(1 + z_1\Big)\Big(1 + z_2\Big)\ y_j \\ &= \Big(1 + a_1 + b_1 i\Big)\Big(1 + a_2 + b_2 i\Big)\ y_j \\ &= \Bigg(\Big(1 + a_1 + a_2 + a_1 a_2 - b_1 b_2\Big) + \Big(b_1 + b_2 + a_1 b_2 + a_2 b_1\Big) i\Bigg)\ y_j \end{aligned}$$

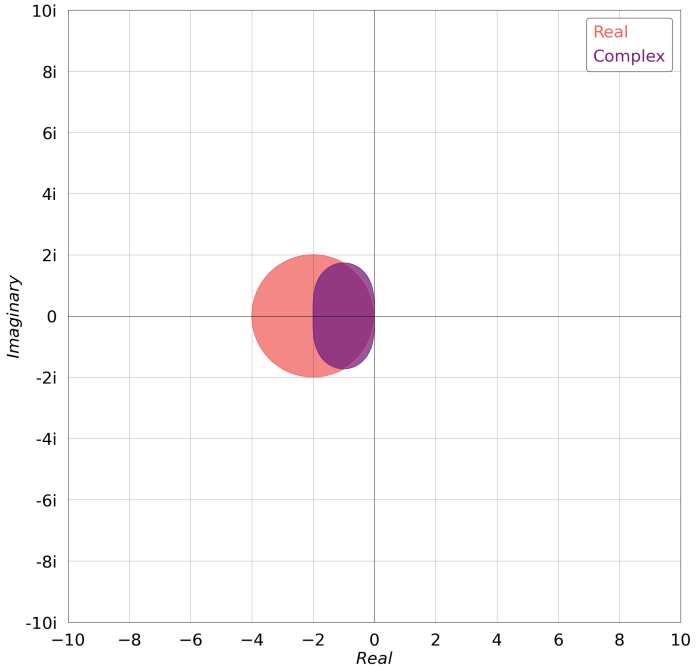
$y_{j+1} \in \mathbb{R} \implies Im(y_{j+1}) = 0 \implies b_1 + b_2 + a_1 b_2 + a_2 b_1 = 0 \implies (1 + a_1)b_2 + (1 + a_2)b_1 = 0$

$z_1 + z_2 = h \in \mathbb{R} \implies a_1 + a_2 + (b_1 + b_2)i = h \implies b_1 + b_2 = 0 \implies b_1 = -b_2$

$(1 + a_1)b_2 + (1 + a_2)b_1 = 0 \implies (1 + a_1)b_2 - (1 + a_2)b_2 = 0 \implies b_2 = 0 = b_1$ *or* $a_1 = a_2$

Thus, $z_1, z_2 \in \mathbb{C} \implies a_1 = a_2$ *and* $b_1 = -b_2$

$\therefore z_1 = a_1 + b_1 i$ *and* $z_2 = a_1 - b_1 i \implies z_2 = \bar{z_1}$ $\square$



Euler's Forward 2-Step $\mathbb{R}$

$$y_{j+1} = \Big(1 + \frac{\lambda h}{2}\Big)^2 y_j$$

$$\implies s_{\mathbb{R}}(\lambda, h) = 1 + \lambda h + \frac{(\lambda h)^2}{4}$$

Euler's Forward 2-Step $\mathbb{C}$

$$\begin{aligned} y_{j+1} &= \Big(1 + z\Big)\Big(1 + \bar{z}\Big) y_j \\ &= \Big(1 + \frac{\lambda h}{2} + bi\Big)\Big(1 + \frac{\lambda h}{2} - bi\Big) y_j \\ &= \Bigg(\Big(1 + \tfrac{\lambda h}{2}\Big)^2 + b^2\Bigg) y_j \end{aligned}$$

$$\implies s_{\mathbb{C}}(\lambda, h) = 1 + \lambda h + \frac{(\lambda h)^2}{4} + b^2$$

$$\implies s_{\mathbb{C}}(\lambda, h) = s_{\mathbb{R}}(\lambda, h) + b^2$$

As $b \in \mathbb{R}$, $s_{\mathbb{C}}(\lambda, h) \geq s_{\mathbb{R}}(\lambda, h)$
This means the stability region for the complex 2-step method is smaller than that of the real 2-step method; there are less values of λh for which the complex 2-step method is stable.
$S_{\mathbb{C}}$ is smaller than $S_{\mathbb{R}}$
The diagram above illustrates the case where $b = a = \frac{\lambda h}{2}$
In this case, we have

$$s_{\mathbb{C}}(\lambda, h) = 1 + \lambda h + \frac{(\lambda h)^2}{2} = M(y) + \mathcal{O}((\lambda h)^3)$$

while

$$s_{\mathbb{R}}(\lambda, h) = 1 + \lambda h + \frac{(\lambda h)^2}{4} = M(y) + \mathcal{O}((\lambda h)^2)$$

The complex 2-step method has a higher order of accuracy than the real 2-step method, with the trade-off of a smaller stability region.

We can do the same kind of analysis for other Numerical Methods.
The same derivations and diagrams for the Backward Euler and Runge-Kutta 4 methods follow.

5.3.2 Euler's Backward 2-Step

Theorem 5.2. ***Euler's Backward Complex Step Pairs must be Conjugate for*** $\lambda \in \mathbb{R}$

For the exponential decay problem, $y = e^{\lambda t}$*, let* $\lambda \in \mathbb{R}$
Let $z_1 = a_1 + b_1 i$ *and* $z_2 = a_2 + b_2 i$ *be a complex step pair for Euler's Backward 2-Step Method.*
Then we must have $z_2 = \bar{z_1}$

Proof:

$\lambda \in \mathbb{R} \implies y_j, y_{j+1} \in \mathbb{R}$

Euler's Backward 2-Step Method is given by:

$$\begin{aligned} y_{j+1} &= s(\lambda, z_1)s(\lambda, z_2)\, y_j \\ &= \left(\frac{1}{1-z_1}\right)\left(\frac{1}{1-z_2}\right) y_j \\ &= \left(\frac{1}{(1-z_1)(1-z_2)}\right) y_j \end{aligned}$$

$$\begin{aligned} (1-z_1)(1-z_2) &= (1-a_1-b_1 i)(1-a_2-b_2 i) \\ &= \big[1-a_1-a_2+a_1a_2-b_1b_2\big] + \big[a_1b_2+a_2b_1-b_1-b_2\big]i \\ &= \big[(a_1-1)(a_2-1)-b_1b_2\big] + \big[b_1(a_2-1)+b_2(a_1-1)\big]i \end{aligned}$$

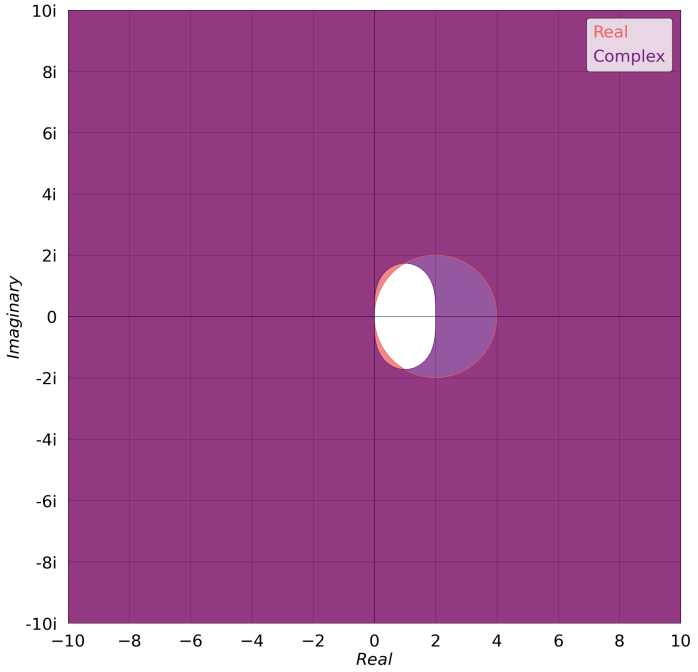
$y_{j+1} \in \mathbb{R} \implies Im(y_{j+1}) = 0 \implies b_1(a_2-1) + b_2(a_1-1) = 0$

$z_1 + z_2 = h \in \mathbb{R} \implies a_1 + a_2 + (b_1+b_2)i = h \implies b_1 + b_2 = 0 \implies b_1 = -b_2$

$b_1(a_2-1) + b_2(a_1-1) = 0 \implies b_1(a_2-1) - b_1(a_1-1) = 0 \implies b_1 = 0 = b_2$ *or* $a_1 = a_2$

Thus, $z_1, z_2 \in \mathbb{C} \implies a_1 = a_2$ *and* $b_1 = -b_2$

$\therefore z_1 = a_1 + b_1 i$ *and* $z_2 = a_1 - b_1 i \implies z_2 = \bar{z_1}$ $\square$



Euler's Backward 2-Step $\mathbb{R}$

$$y_{j+1} = \left(\frac{1}{1-\frac{\lambda h}{2}}\right)^2 y_j$$

$$\implies s_{\mathbb{R}}(\lambda, h) = \frac{1}{1-\lambda h + \frac{(\lambda h)^2}{4}}$$

Euler's Backward 2-Step $\mathbb{C}$

$$y_{j+1} = \left(\frac{1}{1-\frac{\lambda h}{2}+bi}\right)\left(\frac{1}{1-\frac{\lambda h}{2}-bi}\right) y_j$$

$$\implies s_{\mathbb{C}}(\lambda, h) = \frac{1}{1-\lambda h + \frac{(\lambda h)^2}{4} + b^2}$$

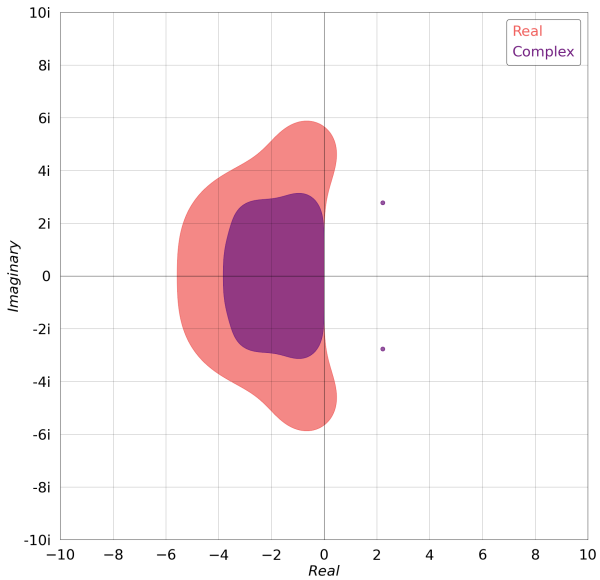
Again, $b \in \mathbb{R}$, so this time, $s_{\mathbb{C}}(\lambda, h) \leq s_{\mathbb{R}}(\lambda, h) \implies S_{\mathbb{C}}$ is larger than $S_{\mathbb{R}}$

5.3.3 Runge-Kutta 4 2-Step

Theorem 5.3. ***Runge-Kutta 4 Complex Step Pairs must be Conjugate***

For the exponential decay problem, $y = e^{\lambda t}$, let $\lambda \in \mathbb{R}$
Let $z_1 = a + bi$ and $z_2 = c + di$ be a complex step pair for Runge-Kutta 4 2-Step Method.
Then we must have $z_2 = \bar{z_1}$

Proof:
This is merely the expansion of the RK4 stability equation . . .
Real: $1 + \frac{a^4c^4}{576} + \frac{a^4c^3}{144} - \frac{a^4c^2d^2}{96} + \frac{a^4c^2}{48} - \frac{a^4cd^2}{48} + \frac{a^4c}{24} + \frac{a^4d^4}{576} - \frac{a^4d^2}{48} + \frac{a^4}{24} - \frac{a^3bc^3d}{36} - \frac{a^3bc^2d}{12} + \frac{a^3bcd^3}{36} - \frac{a^3bcd}{6} + \frac{a^3bd^3}{36} - \frac{a^3bd}{6} + \frac{a^3c^4}{144} + \frac{a^3c^3}{36} - \frac{a^3c^2d^2}{24} + \frac{a^3c^2}{12} - \frac{a^3cd^2}{12} + \frac{a^3c}{6} + \frac{a^3d^4}{144} - \frac{a^3d^2}{12} + \frac{a^3}{6} - \frac{a^2b^2c^4}{96} - \frac{a^2b^2c^3}{24} + \frac{a^2b^2c^2d^2}{16} - \frac{a^2b^2c^2}{8} + \frac{a^2b^2cd^2}{8} - \frac{a^2b^2c}{4} - \frac{a^2b^2d^4}{96} + \frac{a^2b^2d^2}{8} - \frac{a^2b^2}{4} - \frac{a^2bc^3d}{12} - \frac{a^2bc^2d}{4} + \frac{a^2bcd^3}{12} - \frac{a^2bcd}{2} + \frac{a^2bd^3}{12} - \frac{a^2bd}{2} + \frac{a^2c^4}{48} + \frac{a^2c^3}{12} - \frac{a^2c^2d^2}{8} + \frac{a^2c^2}{4} - \frac{a^2cd^2}{4} + \frac{a^2c}{2} + \frac{a^2d^4}{48} - \frac{a^2d^2}{4} + \frac{a^2}{2} + \frac{ab^3c^3d}{36} + \frac{ab^3c^2d}{12} - \frac{ab^3cd^3}{36} + \frac{ab^3cd}{6} - \frac{ab^3d^3}{36} + \frac{ab^3d}{6} - \frac{ab^2c^4}{48} - \frac{ab^2c^3}{12} + \frac{ab^2c^2d^2}{8} - \frac{ab^2c^2}{4} + \frac{ab^2cd^2}{4} - \frac{ab^2c}{2} - \frac{ab^2d^4}{48} + \frac{ab^2d^2}{4} - \frac{ab^2}{2} - \frac{abc^3d}{6} - \frac{abc^2d}{2} + \frac{abcd^3}{6} - abcd + \frac{abd^3}{6} - abd + \frac{ac^4}{24} + \frac{ac^3}{6} - \frac{ac^2d^2}{4} + \frac{ac^2}{2} - \frac{acd^2}{2} + ac + \frac{ad^4}{24} - \frac{ad^2}{2} + a + \frac{b^4c^4}{576} + \frac{b^4c^3}{144} - \frac{b^4c^2d^2}{96} + \frac{b^4c^2}{48} - \frac{b^4cd^2}{48} + \frac{b^4c}{24} + \frac{b^4d^4}{576} - \frac{b^4d^2}{48} + \frac{b^4}{24} + \frac{b^3c^3d}{36} + \frac{b^3c^2d}{12} - \frac{b^3cd^3}{36} + \frac{b^3cd}{6} - \frac{b^3d^3}{36} + \frac{b^3d}{6} - \frac{b^2c^4}{48} - \frac{b^2c^3}{12} + \frac{b^2c^2d^2}{8} - \frac{b^2c^2}{4} + \frac{b^2cd^2}{4} - \frac{b^2c}{2} - \frac{b^2d^4}{48} + \frac{b^2d^2}{4} - \frac{b^2}{2} - \frac{bc^3d}{6} - \frac{bc^2d}{2} + \frac{bcd^3}{6} - bcd + \frac{bd^3}{6} - bd + \frac{c^4}{24} + \frac{c^3}{6} - \frac{c^2d^2}{4} + \frac{c^2}{2} - \frac{cd^2}{2} + c + \frac{d^4}{24} - \frac{d^2}{2}$
Imaginary: $\frac{a^4c^3d}{144} + \frac{a^4c^2d}{48} - \frac{a^4cd^3}{144} + \frac{a^4cd}{24} - \frac{a^4d^3}{144} + \frac{a^4d}{24} + \frac{a^3bc^4}{144} + \frac{a^3bc^3}{36} - \frac{a^3bc^2d^2}{24} + \frac{a^3bc^2}{12} - \frac{a^3bcd^2}{12} + \frac{a^3bc}{6} + \frac{a^3bd^4}{144} - \frac{a^3bd^2}{12} + \frac{a^3b}{6} + \frac{a^3c^3d}{36} + \frac{a^3c^2d}{12} - \frac{a^3cd^3}{36} + \frac{a^3cd}{6} - \frac{a^3d^3}{36} + \frac{a^3d}{6} - \frac{a^2b^2c^3d}{24} - \frac{a^2b^2c^2d}{8} + \frac{a^2b^2cd^3}{24} - \frac{a^2b^2cd}{4} + \frac{a^2b^2d^3}{24} - \frac{a^2b^2d}{4} + \frac{a^2bc^4}{48} + \frac{a^2bc^3}{12} - \frac{a^2bc^2d^2}{8} + \frac{a^2bc^2}{4} - \frac{a^2bcd^2}{4} + \frac{a^2bc}{2} + \frac{a^2bd^4}{48} - \frac{a^2bd^2}{4} + \frac{a^2b}{2} + \frac{a^2c^3d}{12} + \frac{a^2c^2d}{4} - \frac{a^2cd^3}{12} + \frac{a^2cd}{2} - \frac{a^2d^3}{12} + \frac{a^2d}{2} - \frac{ab^3c^4}{144} - \frac{ab^3c^3}{36} + \frac{ab^3c^2d^2}{24} - \frac{ab^3c^2}{12} + \frac{ab^3cd^2}{12} - \frac{ab^3c}{6} - \frac{ab^3d^4}{144} + \frac{ab^3d^2}{12} - \frac{ab^3}{6} - \frac{ab^2c^3d}{12} - \frac{ab^2c^2d}{4} + \frac{ab^2cd^3}{12} - \frac{ab^2cd}{2} + \frac{ab^2d^3}{12} - \frac{ab^2d}{2} + \frac{abc^4}{24} + \frac{abc^3}{6} - \frac{abc^2d^2}{4} + \frac{abc^2}{2} - \frac{abcd^2}{2} + abc + \frac{abd^4}{24} - \frac{abd^2}{2} + ab + \frac{ac^3d}{6} + \frac{ac^2d}{2} - \frac{acd^3}{6} + acd - \frac{ad^3}{6} + ad + \frac{b^4c^3d}{144} + \frac{b^4c^2d}{48} - \frac{b^4cd^3}{144} + \frac{b^4cd}{24} - \frac{b^4d^3}{144} + \frac{b^4d}{24} - \frac{b^3c^4}{144} - \frac{b^3c^3}{36} + \frac{b^3c^2d^2}{24} - \frac{b^3c^2}{12} + \frac{b^3cd^2}{12} - \frac{b^3c}{6} - \frac{b^3d^4}{144} + \frac{b^3d^2}{12} - \frac{b^3}{6} - \frac{b^2c^3d}{12} - \frac{b^2c^2d}{4} + \frac{b^2cd^3}{12} - \frac{b^2cd}{2} + \frac{b^2d^3}{12} - \frac{b^2d}{2} + \frac{bc^4}{24} + \frac{bc^3}{6} - \frac{bc^2d^2}{4} + \frac{bc^2}{2} - \frac{bcd^2}{2} + bc + \frac{bd^4}{24} - \frac{bd^2}{2} + b + \frac{c^3d}{6} + \frac{c^2d}{2} - \frac{cd^3}{6} + cd - \frac{d^3}{6} + d$



Runge-Kutta 4 2-Step $\mathbb{R}$

$$y_{j+1} = \left(1 + \frac{\lambda h}{2} + \frac{\left(\frac{\lambda h}{2}\right)^2}{2} + \frac{\left(\frac{\lambda h}{2}\right)^3}{6} + \frac{\left(\frac{\lambda h}{2}\right)^4}{24}\right)^2 y_j$$

$$y_{j+1} = \left(1 + \lambda h + \frac{(\lambda h)^2}{2} + \frac{(\lambda h)^3}{6} + \frac{(\lambda h)^4}{24} + \frac{(\lambda h)^5}{128} + \frac{5(\lambda h)^6}{4608} + \frac{(\lambda h)^7}{9216} + \frac{(\lambda h)^8}{147456}\right) y_j$$

$$\implies s_{\mathbb{R}}(\lambda, h) = 1 + \lambda h + \frac{(\lambda h)^2}{2} + \frac{(\lambda h)^3}{6} + \frac{(\lambda h)^4}{24} + \frac{(\lambda h)^5}{128} + \frac{5(\lambda h)^6}{4608} + \frac{(\lambda h)^7}{9216} + \frac{(\lambda h)^8}{147456}$$

Runge-Kutta 4 2-Step $\mathbb{C}$

$$y_{j+1} = \left(1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24}\right)\left(1 + \bar{z} + \frac{\bar{z}^2}{2} + \frac{\bar{z}^3}{6} + \frac{\bar{z}^4}{24}\right) y_j$$

$\implies s_{\mathbb{C}}(\lambda, h) = 1 + \lambda h + \frac{\lambda h^2}{2} + \frac{\lambda h^3}{6} + \frac{\lambda h^4}{24} + \frac{\lambda h^5}{128} + \frac{5\lambda h^6}{4608} + \frac{\lambda h^7}{9216} + \frac{\lambda h^8}{147456} + \frac{b^2\lambda h^3}{48} + \frac{b^2\lambda h^4}{128} + \frac{b^2\lambda h^5}{768} + \frac{b^2\lambda h^6}{9216} + \frac{b^4\lambda h}{24} - \frac{b^4\lambda h^2}{96} + \frac{b^4\lambda h^3}{192} + \frac{b^4\lambda h^4}{1536} - \frac{b^6}{72} + \frac{b^6\lambda h}{144} + \frac{b^6\lambda h^2}{576} + \frac{b^8}{576}$

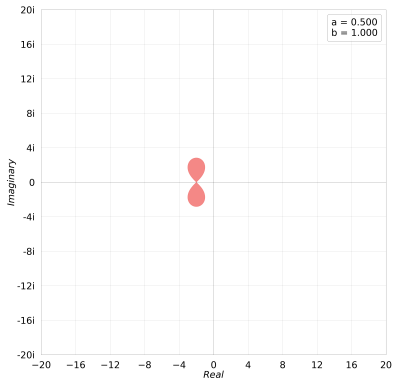
5.4 Varying b for Complex Conjugate Step Pairs

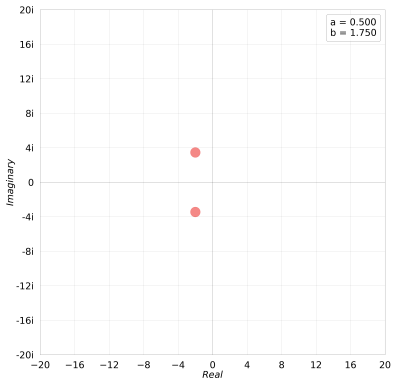
As mentioned earlier, we need not only focus on the case where $b = \frac{\lambda h}{2}$
We can vary b and see how the stability regions change.
In the cases below, we have preserved $a = \frac{\lambda h}{2}$
Consequently, these are still complex conjugate step pairs.
Videos showing the stability regions for varying b values can be found in the *GitHub Repository*[3] for this project.
Below are some of the video frames for each method.
Observations:

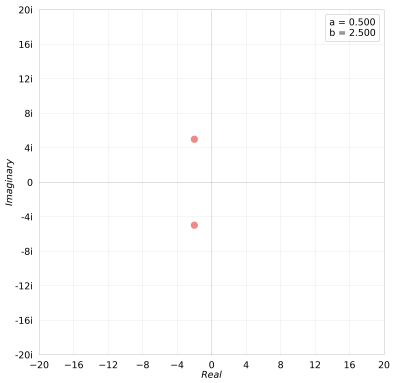
- The stability region for complex conjugate pairs is always symmetric accross the real axis.

· As b increases, the stability region shrinks.
This can be seen quite simply in the $s_{\mathbb{C}}(\lambda, h)$ equation for Euler's Forward and Backward methods.

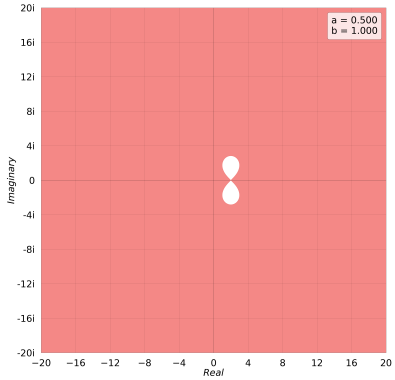
5.4.1 Euler's Forward

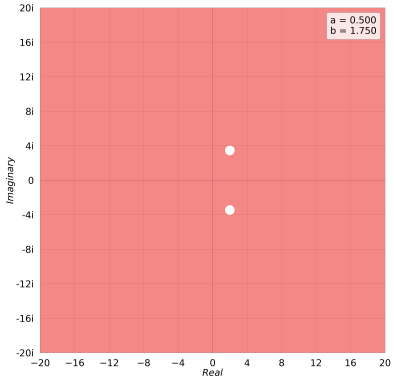


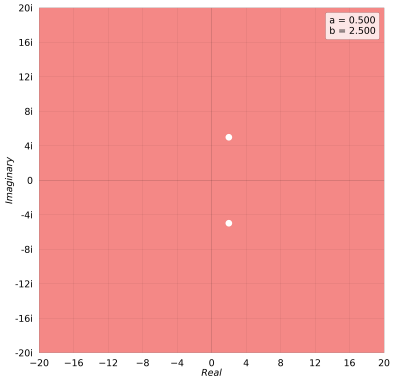




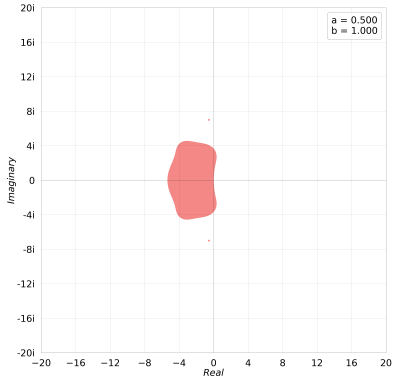
5.4.2 Euler's Backward

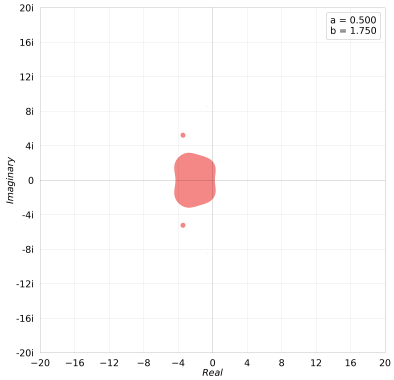


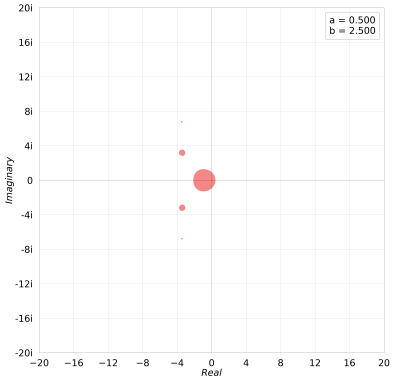




5.4.3 Runga-Kutta 4





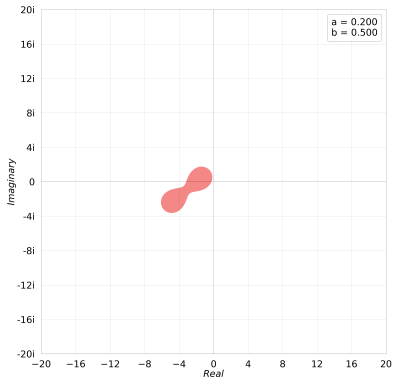


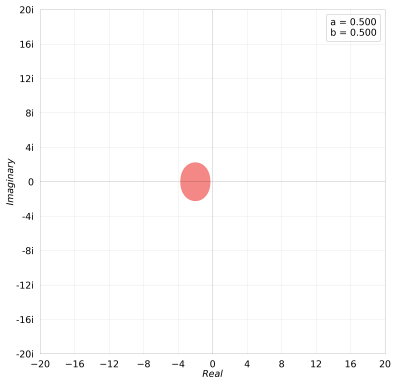
5.5 Varying a

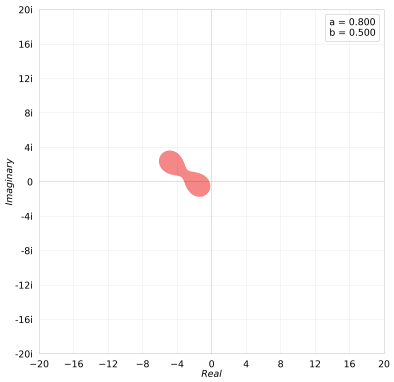
We can also vary a and see how the stability regions change.
This falls outside the scope of complex conjugate step pairs.
In the cases below, we have set $b = \frac{\lambda h}{2}$
Videos showing the stability regions for varying a values can be found in the *GitHub Repository*[3] for this project.
Below are some of the video frames for each method.
Observations:

- The stability region for varying a values is not symmetric accross the real axis, except for the case where $a = 0.5$.
- The stability regions for $a = 0.5 \pm \alpha$ are symmetric accross the real axis for any $|\alpha| < 0.5$.
- The stability region for Euler's Backward Method is the inverse of that for Euler's Forward Method, after it has been reflected across the imaginary axis.

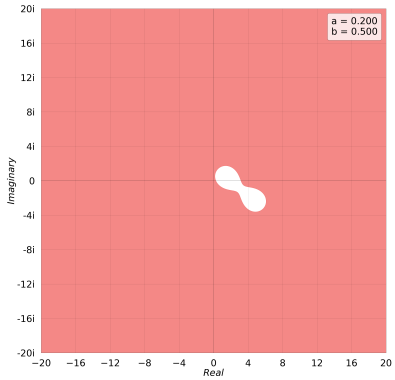
5.5.1 Euler's Forward

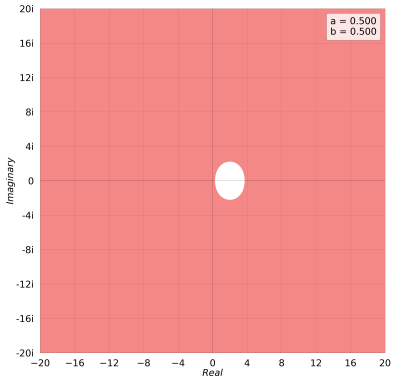


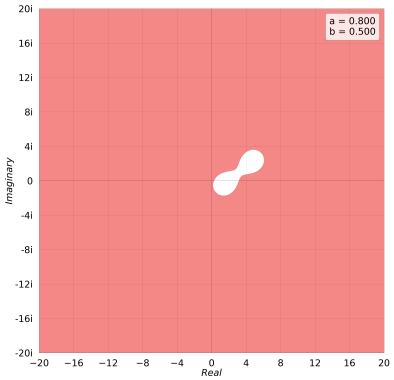




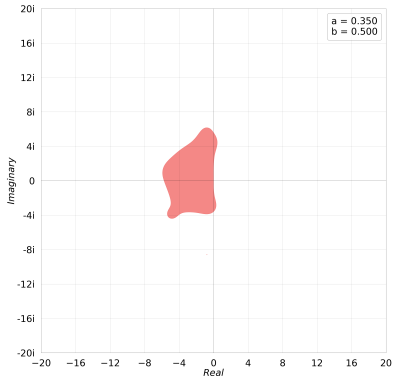
5.5.2 Euler's Backward

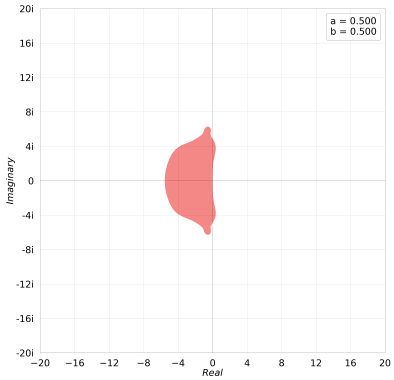


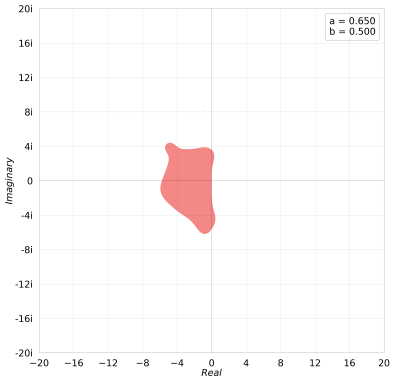




5.5.3 Runga-Kutta 4







References

[1] Llyod N. Trefethen, *The definition of numerical analysis*, Cornell University, 1992.

[2] Jithin D. George, Samuel Y. Jung, and Niall M. Mangan, *Walking into the complex plane to 'order' better time integrators*, `https://arxiv.org/abs/2110.04402`, 2021.

[3] The GitHub repository for this project, `https://github.com/CianJDuggan/Capstone-Project`

[4] Next reference.