Numerical methods
Computer project №2

Gleb Zhdanko

December 2024

General Information

The computations and visualizations were performed on a MacBook Pro (13-inch, 2020, Two Thunderbolt 3 Ports) equipped with a 1.4 GHz Quad-Core Intel Core i5 processor.

I used Python 3.12.4. The following Python libraries were utilized in the implementation:

- `numpy`: for numerical computations,
- `scipy`: for solving linear systems and other numerical methods,
- `sympy`: for symbolic manipulation and validation,
- `matplotlib`: for creating high-quality visualizations.

Problem 1

Problem: If a debt is amortized by regular payments of size R and is subject to interest I, the unpaid balance is P_k, where $P_{k+1} = (1+I)P_k - R$. The initial debt being $P_0 = A$. What is the unpaid balance as a function of A, I, and R. Also, what must be the regular payment R be in order to reduce the balance exactly to zero in N payments?

Solution: Let's prove by induction that: $P_k = (1+I)^k \cdot A - R \cdot \frac{(1+I)^k - 1}{I}$ for $k >= 1$.

Base: k=1: $P_1 = (1+I) \cdot P_0 - R = (1+I)^1 \cdot A - R$

Let's assume that our assumption is correct for $n >= 2$. Let's prove that it's correct for $n+1$:

$$P_{n+1} = (1+I) \cdot P_n - R = (1+I) \cdot \left((1+I)^n \cdot A - R \cdot \frac{(1+I)^n - 1}{I}\right) - R =$$
$$= (1+I)^{n+1} \cdot A - R \cdot (1+I) \cdot \frac{(1+I)^n - 1}{I} - R = (1+I)^{n+1} - R \cdot \frac{(1+I)^{n+1} - 1 - I + I}{I} =$$
$$= (1+I)^{n+1} \cdot A - R \cdot \frac{(1+I)^{n+1} - 1}{I}$$

■

Let's find R, so that $P_N = 0$:

$$P_N = 0 \Rightarrow (1+I)^N \cdot A - R \cdot \frac{(1+I)^N - 1}{I} = 0 \Rightarrow$$
$$\Rightarrow R = \frac{(1+I)^N \cdot A \cdot I}{(1+I)^N - 1}$$

Discussion: - Consider $I \to 0$:

$$\lim_{I \to 0} P_k = \lim_{I \to 0}((1+I)^k \cdot A - R \cdot \frac{(1+I)^k - 1}{I}) =$$
$$= \lim_{I \to 0}(1+I)^k \cdot A - \lim_{I \to 0} R \frac{(1+I)^k - 1}{I} = A - \lim_{I \to 0} R \frac{I \cdot k}{I} =$$
$$= A - R \cdot k$$

- Let's find $\lim_{N \to \infty} \frac{(1+I)^N \cdot A \cdot I}{(1+I)^N - 1}$:

$$\lim_{N \to \infty} \frac{(1+I)^N \cdot A \cdot I}{(1+I)^N - 1} = \lim_{N \to \infty} \frac{A \cdot I}{1 - \frac{1}{(1+I)^N}} = A \cdot I, for\ I > 0$$

Problem 2

Problem: The differential equation describing the ice build-up (on a lake for example) is given by

$$h^2\frac{d^2h}{dt^2} + 2\left(\frac{dh}{dt}\right)^2 + 4L\cdot\left(\frac{1+h}{2+h}\right)\frac{dh}{dt} - \frac{4L}{2+h} = 0,$$

where h is the thermal resistance of the ice slab (thickness divided by thermal conductivity), L is a dimensionless latent heat for which 0.5 is a reasonable value, and t is a non-dimensional time. Appropriate initial conditions are $h(0) = 0.01, h'(0) = 0.5$. Solve the problem for all values of t until $h = 2$.

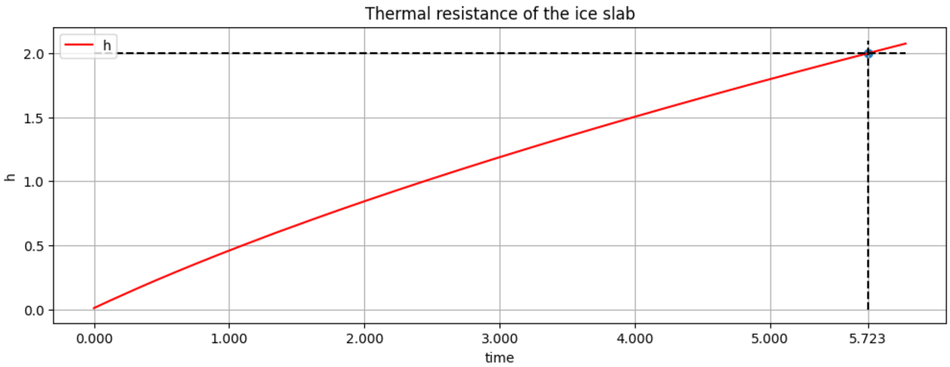
Solution Overview: For this problem I used numerical integration introduced by scipy: solve_ip. To transfer from second order differential equation to the system of first order I did the following:

$$\begin{cases} \frac{dh}{dt} = z \\ h^2\frac{dz}{dt} + 2z^2 + 4L\cdot\left(\frac{1+h}{2+h}\right)z - \frac{4L}{2+h} = 0 \end{cases} \Leftrightarrow \begin{cases} \frac{dh}{dt} = z \\ \frac{dz}{dt} = \left(\frac{4L}{2+h} - 4L\cdot\left(\frac{1+h}{2+h}\right)z - 2z^2\right)\backslash h^2 \end{cases}$$

Then using solve_ip I have solved the system of ODE. By integrating several times, I was assured that $h = 2$ is around $t = 6$, so interval of integration was from 0 to 6. I choose the 1000000 points for integration to get more accurate result. For method of integration I have tested:

- Explicit Runge-Kutta method of order 5 (RK45)
- Explicit Runge-Kutta method of order 3 (RK23)
- Explicit Runge-Kutta method of order 8 (DOP853)

There wasn't much difference in solution, so I have stopped on method of order 8, because I believe it's the most accurate. Then I have truncated the solution where $h \leq 2$. The final time was: $t = 5.723$ I have got the following plot:



Problem 3

Problem: Find the multistep method of the form

$$y_{n+2} + \alpha_1 y_{n+1} + \alpha_0 y_n = h(\beta_1 f_{n+1} + \beta_0 f_n),$$

of the highest possible order. Try this formula on the example $y' = y, y(0) = 1, h = 1/10, 1/20, 1/40$. Explain the results.

Solution Overview: To derive the method with the highest accuracy, I considered the model problem:

$$y' = \lambda y.$$

Substituting into the given formula, we obtain:

$$y_{n+2} + \alpha_1 y_{n+1} + \alpha_0 y_n = h(\beta_1 \lambda y_{n+1} + \beta_0 \lambda y_n).$$

By simplifying and applying the shift operator ($y_{n+k} = E^k y_n$), this can be rewritten as the following quadratic equation:

$$\sigma^2 + \sigma(\alpha_1 - \beta_1 h\lambda) + \alpha_0 - \beta_0 h\lambda = 0,$$

where σ is eigenvalue of E.

After solving the equation we get 2 roots:

$$\sigma_{1,2} = \frac{\beta_1 h\lambda - \alpha_1 \pm \sqrt{(\alpha_1 - \beta_1 h\lambda)^2 - 4(\alpha_0 - \beta_0 h\lambda)}}{2}$$

To get the most accurate method, σ_1 has to have the same coefficients as $\exp(h\lambda)$ - real solution - in first 4 terms (since we have only 4 unknowns we cannot guarantee more coincidence).

We get system of equations from Taylor series expansion of the square root by $h\lambda$ near 0, and Taylor series expansion of $\exp(h\lambda)$ by $h\lambda$ near 0. I have equated coefficients before $(h\lambda)^0, (h\lambda)^1, (h\lambda)^2, (h\lambda)^3$. After solving system of equations for coefficients, we get that:

$$\begin{cases} \alpha_0 = -5 \\ \alpha_1 = 4 \\ \beta_0 = 2 \\ \beta_1 = 4 \end{cases}$$

Then I have checked whether the coefficient before $(h\lambda)^4$ is the same as in exponent series, but it appeared different , so our order of accuracy is 4.

$$\sigma_1 = 2h\lambda - 2 + \sqrt{9 - 6h\lambda + 4(h\lambda)^2}$$

$$\sigma_2 = 2h\lambda - 2 - \sqrt{9 - 6h\lambda + 4(h\lambda)^2}$$

Substituting coefficients into multistep method we get:

$$y_{n+2} + 4y_{n+1} - 5y_n = h(4f_{n+1} + 2f_n)$$

Or in other form:

$$y_n = c_1\sigma_1^n + c_2\sigma_2^n = c_1(2h\lambda - 2 + \sqrt{9 - 6h\lambda + 4(h\lambda)^2})^n + c_2(2h\lambda - 2 - \sqrt{9 - 6h\lambda + 4(h\lambda)^2})^n$$

To start the method we need to calculate y_1. We can use explicit Euler for that:

$$y_1 = y_0 + h \cdot \cdot f_0$$

In our case: $y_0 = y(0) = 1, f(y_n) = y_n$, then

$$y_1 = 1 + h$$

Now for each h we can find c_1, c_2 ($\lambda = 1$) by solving system of two equations:

$$\begin{cases} y_0 = c_1 + c_2 \\ y_1 = c_1\sigma_1 + c_2\sigma_2 \end{cases}$$

Then we substitute c_1, c_2 into the formula for $y_n = c_1\sigma_1^n + c_2\sigma_2^n$ and calculate y_n.

Then we can compare real solution $y_n^{real} = e^{hn}$ with numerical solution y_n.

Since numerical solution is growing up approximately as 5^n, and the smallest c_2 is $5 \cdot 10^{-5}$ after 10 steps solution by modulus is approximately 100, and real solution is no bigger than 3. It's clear that our solution is growing too fast to see some interesting plots and we can only see it on the semilog scale.

Results and Observations:

- σ_2 - is nonphysical solution since it's modulus is greater than 1, and after some steps σ_2 starts to dominate and solution is blowing up.
- Since σ_2 is negative the numerical solution should oscillate.
- I have also tried to start sequence with exact value of $y_1 = e^h$. However, because of the rounding error in python c_2 wasn't zero, and solution was also blowing up.
- Since numerical solution is growing up fast I had to tune time steps and interval for different h.

Short explanation why solution is blowing up. Let's consider σ_1 and find its stable area for real λ.

$$|\sigma_1| \leq 1 \Leftrightarrow -1 \leq 2h\lambda - 2 + \sqrt{9 - 6h\lambda + 4(h\lambda)^2} \leq 1 \Leftrightarrow$$

$$1 - 2h\lambda \leq \sqrt{9 - 6h\lambda + 4(h\lambda)^2} \leq 3 - 2h\lambda \Leftrightarrow$$

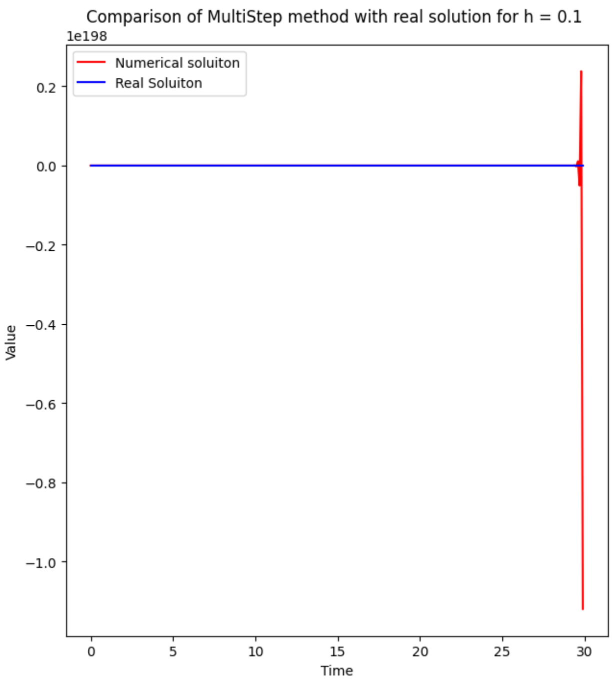
$$\begin{cases} \sqrt{9 - 6h\lambda + 4(h\lambda)^2} \geq 1 - 2h\lambda \\ \sqrt{9 - 6h\lambda + 4(h\lambda)^2} \leq 3 - 2h\lambda \end{cases} \Leftrightarrow$$

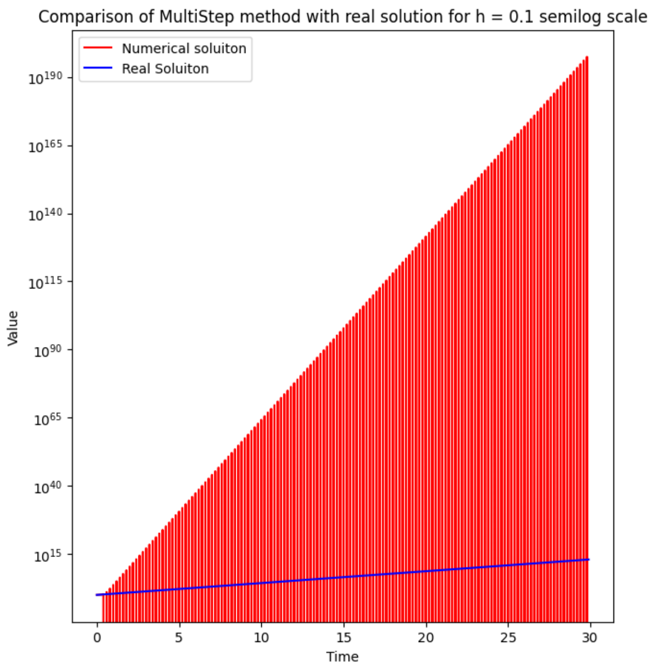
$$\Leftrightarrow \begin{cases} 9 - 6h\lambda + 4(h\lambda)^2 \geq 1 - 4h\lambda + 4(h\lambda)^2 \\ 9 - 6h\lambda + 4(h\lambda)^2 \leq 9 - 12h\lambda + 4(h\lambda)^2 \end{cases} \Leftrightarrow$$

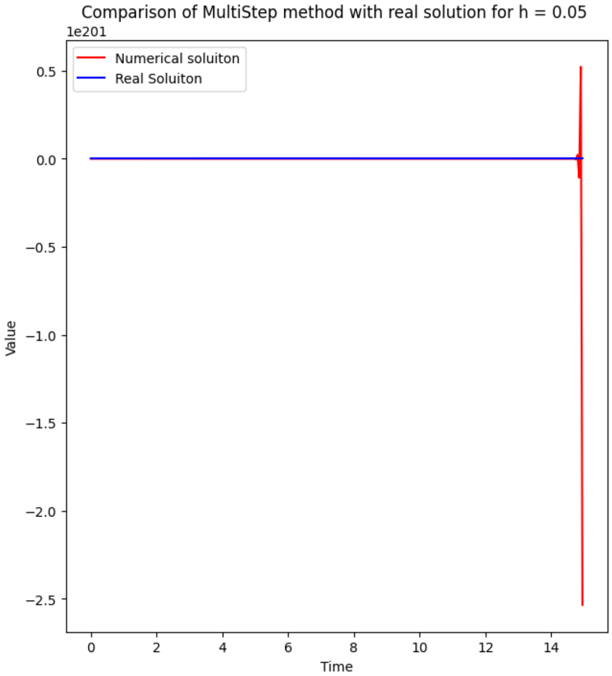
$$\Leftrightarrow \begin{cases} 8 \geq 2h\lambda \\ 6h\lambda \leq 0 \end{cases} \Leftrightarrow$$

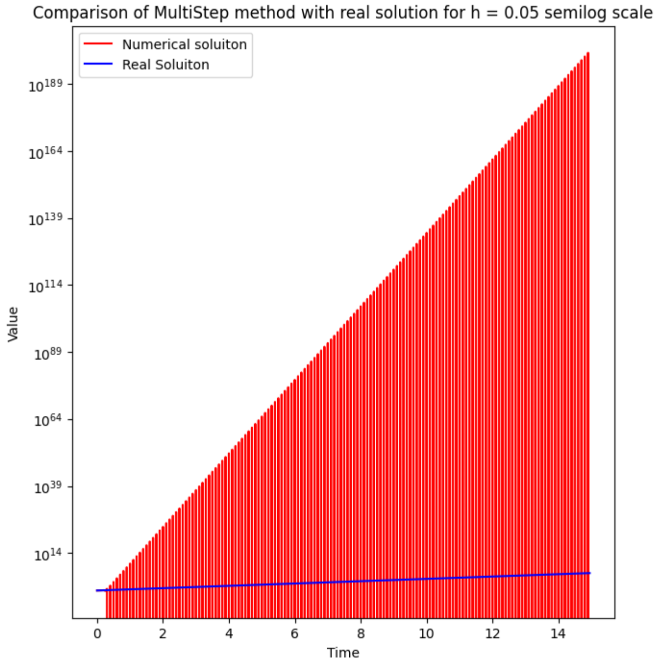
$$\lambda \leq 0$$

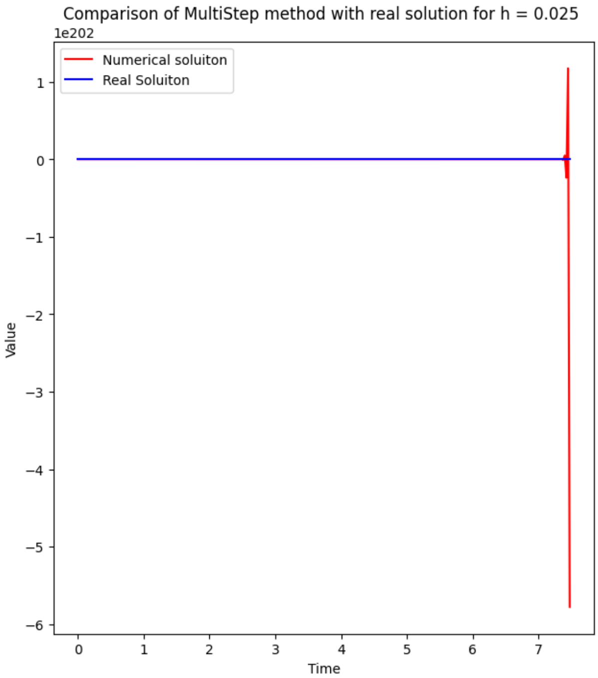
That's why for $\lambda = 1$ our solution is blowing up even if we put $c_2 = 0$.

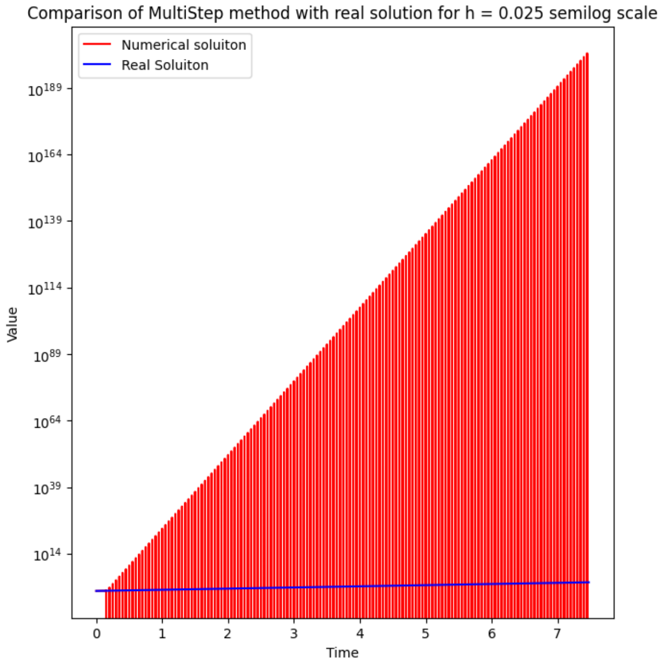





Comparison of MultiStep method with real solution for h = 0.05
1e201
Numerical soluiton
Real Soluiton
Value
Time


Comparison of MultiStep method with real solution for h = 0.05 semilog scale
Numerical soluiton
Real Soluiton
Value
Time


Comparison of MultiStep method with real solution for h = 0.025
1e202
Numerical soluiton
Real Soluiton
Value
Time


Comparison of MultiStep method with real solution for h = 0.025 semilog scale
Numerical soluiton
Real Soluiton
Value
Time

Problem 4

Problem: Consider a simple ecosystem of rabbits that has an infinite food supply and foxes that prey upon rabbits for their food. A mathematical model for this system consists of the following differential equations:

$$\begin{cases} \frac{dr}{dt} = 2r - \alpha r f, r(0) = r_0, \\ \frac{df}{dt} = -f + \alpha r f, f(0) = f_0, \end{cases}$$

where t is time, r is the number of rabbits, f is the number of foxes, and α is a positive constant. When α is zero, the two populations do not interact, and the rabbits do what rabbits do best, and the foxes die of starvation. This system has been studied for various values of α and initial conditions. For the purpose of this problem, we fix the value of α to be 0.01. To answer the questions in parts a) and b) below, use both second order Adams-Bashforth and Leapfrog methods to solve the differential equations. In each case use both the explicit Euler scheme and the second order Runge-Kutta scheme to start the integrations. Compare the results. In each case, what is the maximum time step for reasonable accuracy?

a) Compute the solution with $r_0 = 300$ and $f_0 = 150$. You should observe that the behavior of the system is periodic with period very close to five time units. Plot the solution in the (f, r) plane. If the time step is too large, what happens to the periodicity of the solution?

b) Compute the solution with $r_0 = 15$ and $f_0 = 22$. What happens to the rabbit population? Integrate over several periods.

Solution overview: First, I have implemented 4 algorithms: Adams-Bashforth, Leapfrog, Explicit Euler and Runge-Kutta. For Runge-Kutta I used parameter $\alpha = 1/2$.

Then I have integrated the equation from time 0 to 20 with different steps (h). As it stated in the problem and turned out to be true the period was around 5, so we don't need bigger time integration.

Next, to find maximum time step for reasonable accuracy I have tested integration scheme with different time step (h) and found out that solution starts to show strange behavior for case a:

- Adams-Bashforth + Explicit Euler: $h = 0.29$ is limit time step. For bigger h population of rabbits become negative, and if we increase even further we will see that population of rabbits goes to infinity and population of foxes goes to -infinity.
- Adams-Bashforth + Runge-Kutta: $h = 0.29$ is limit time step. Similiarly, for bigger steps solution is broken.
- Leapfrog + Explicit Euler: $h = 0.0006$ is limit time step.
- Leapfrog + Runge-Kutta: $h = 0.012$ is limit time step.

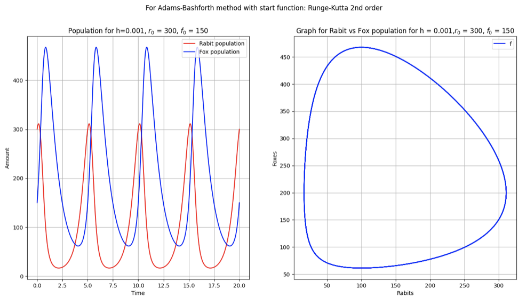
For case b:

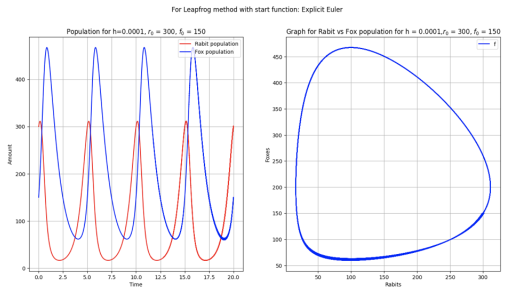
- Adams-Bashforth + Explicit Euler: $h = 0.13$ is limit time step.
- Adams-Bashforth + Runge-Kutta: $h = 0.13$ is limit time step. Similiarly, for bigger steps solution is broken.
- Leapfrog + Explicit Euler: $h = 0.000002$ is limit time step.
- Leapfrog + Runge-Kutta: $h = 0.0003$ is limit time step.

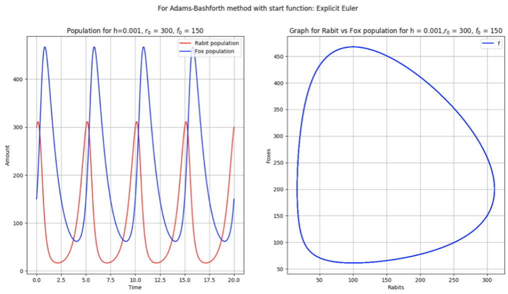
Results and Discusion:

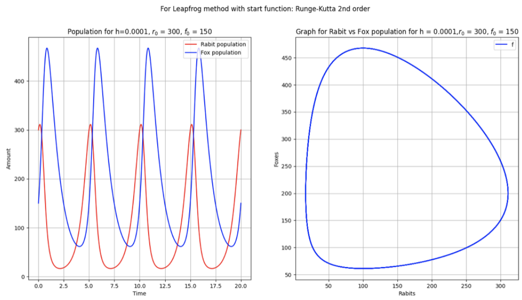
- If the time step is too much then two things happen: first population of foxes or rabbits become negative, second: population of rabbits goes to infinity, population of foxes goes to minus infinity.
- Interesting thing happens with Leapfrog method. If we specify time of integration 100 than even for $h = 10^{-5}$ it's still incorrect. I think this is due to error in numerical method and after sufficient steps error becomes too much and we can see it on the graph.
- Adams-Bashforth, on the other hand, shows great consistency for large value of time integration even with the small time steps as $h = 0.01$ for both Explicit Euler and Runge-Kutta starts.
- Starting with Runge-Kutta gives higher maximum time step. This is due to the higher order of accuracy of Runge-Kutta with comparison to Explicit Euler. (Second versus first).

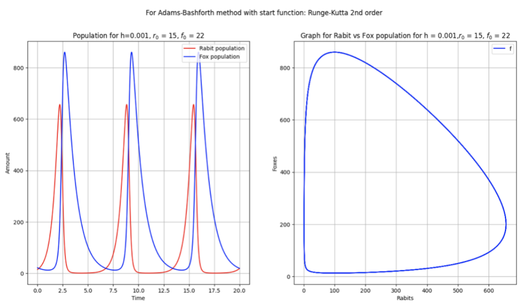
- Adams-Bashforth performs better than leapfrog in this problem.
- In case b, we observe that first the population of rabbits grows, and the population of foxes almost reaches zero, then the population of foxes grows sharply, which brings the number of rabbits to 0 and the pattern repeats.

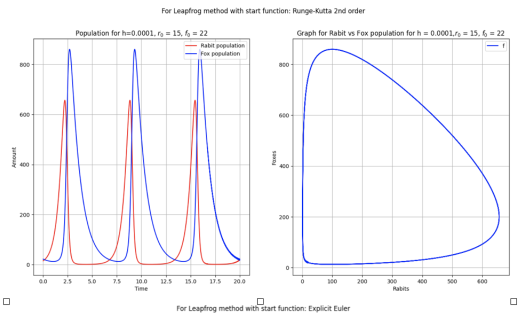


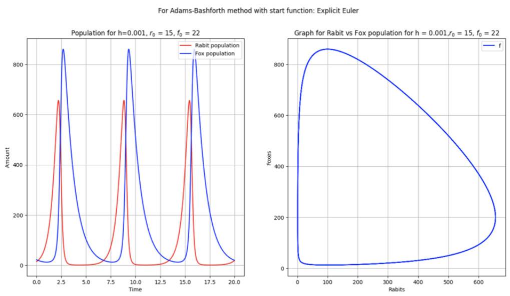


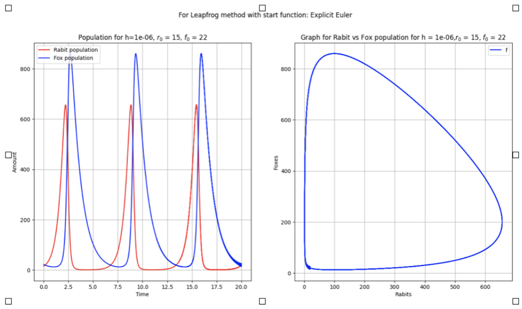


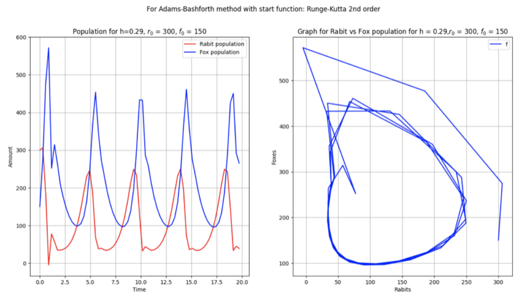


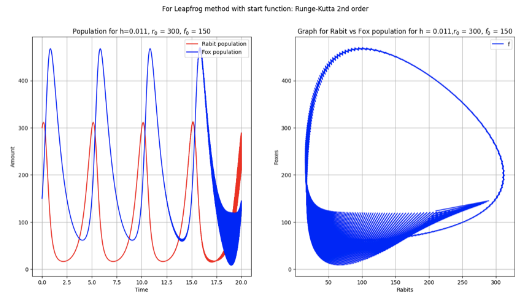


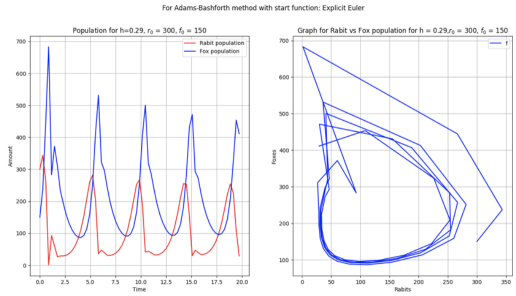


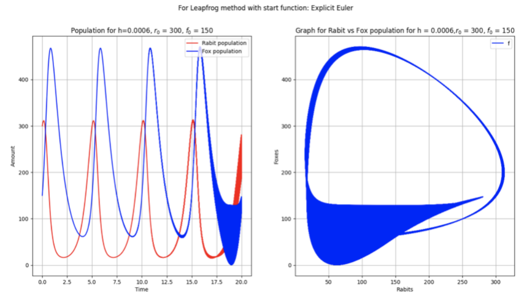


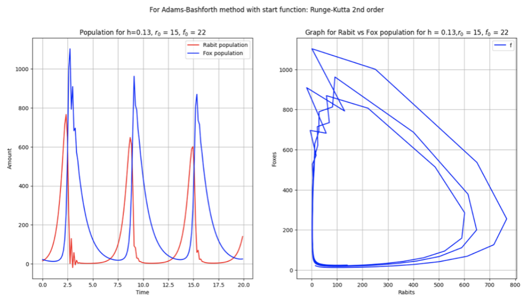


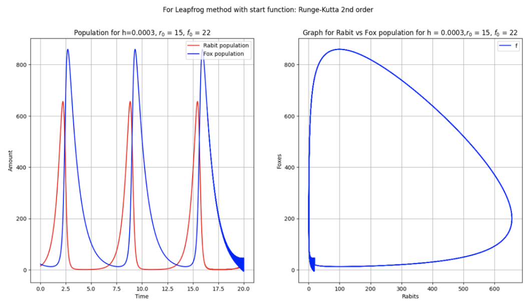

For Adams-Bashforth method with start function: Runge-Kutta 2nd order
Population for h=0.29, r_0 = 300, f_0 = 150
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 0.29,r_0 = 300, f_0 = 150
f
Foxes
Rabits

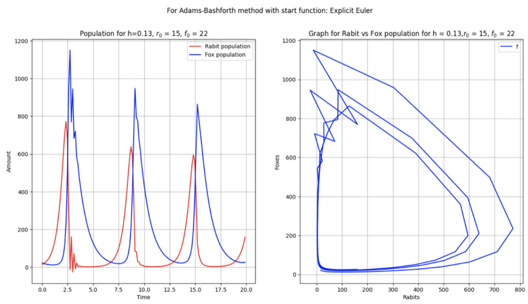

For Leapfrog method with start function: Runge-Kutta 2nd order
Population for h=0.011, r_0 = 300, f_0 = 150
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 0.011,r_0 = 300, f_0 = 150
f
Foxes
Rabits

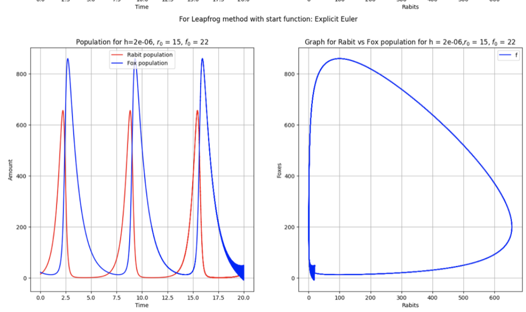

For Adams-Bashforth method with start function: Explicit Euler
Population for h=0.29, r_0 = 300, f_0 = 150
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 0.29,r_0 = 300, f_0 = 150
f
Foxes
Rabits

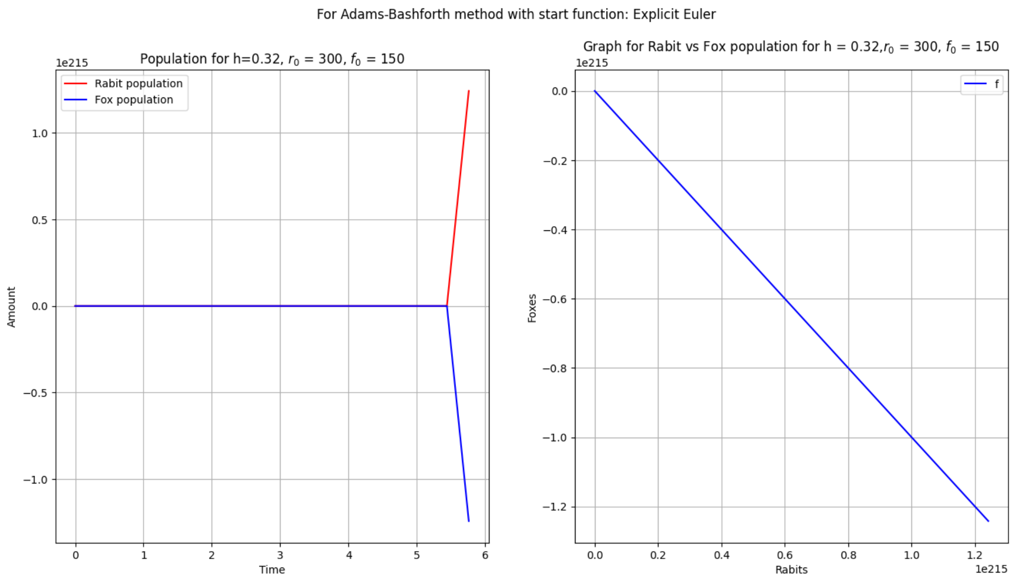

For Leapfrog method with start function: Explicit Euler
Population for h=0.0006, r_0 = 300, f_0 = 150
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 0.0006,r_0 = 300, f_0 = 150
f
Foxes
Rabits


For Adams-Bashforth method with start function: Runge-Kutta 2nd order
Population for h=0.13, r_0 = 15, f_0 = 22
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 0.13,r_0 = 15, f_0 = 22
f
Foxes
Rabits

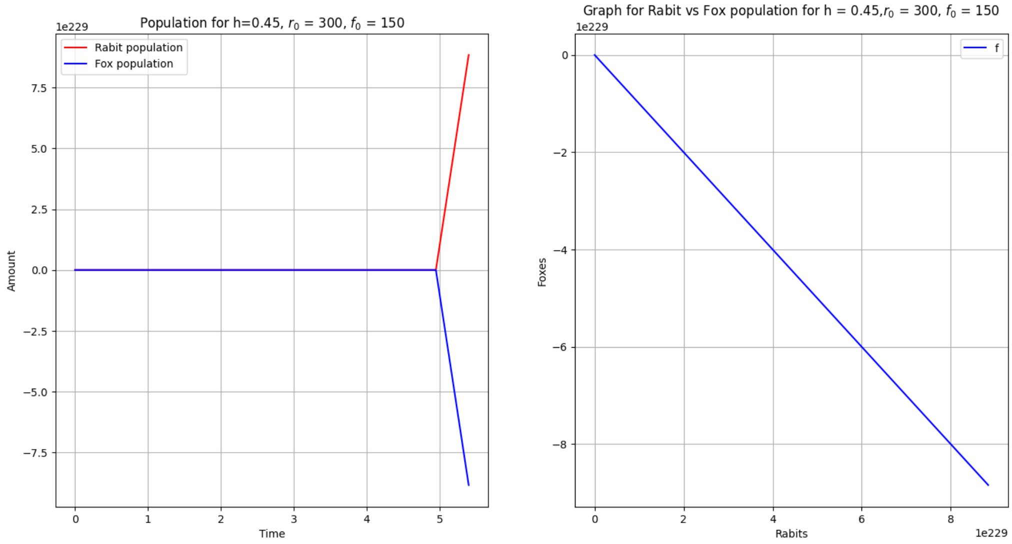

For Leapfrog method with start function: Runge-Kutta 2nd order
Population for h=0.0003, r_0 = 15, f_0 = 22
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 0.0003,r_0 = 15, f_0 = 22
f
Foxes
Rabits


For Adams-Bashforth method with start function: Explicit Euler
Population for h=0.13, r_0 = 15, f_0 = 22
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 0.13,r_0 = 15, f_0 = 22
f
Foxes
Rabits

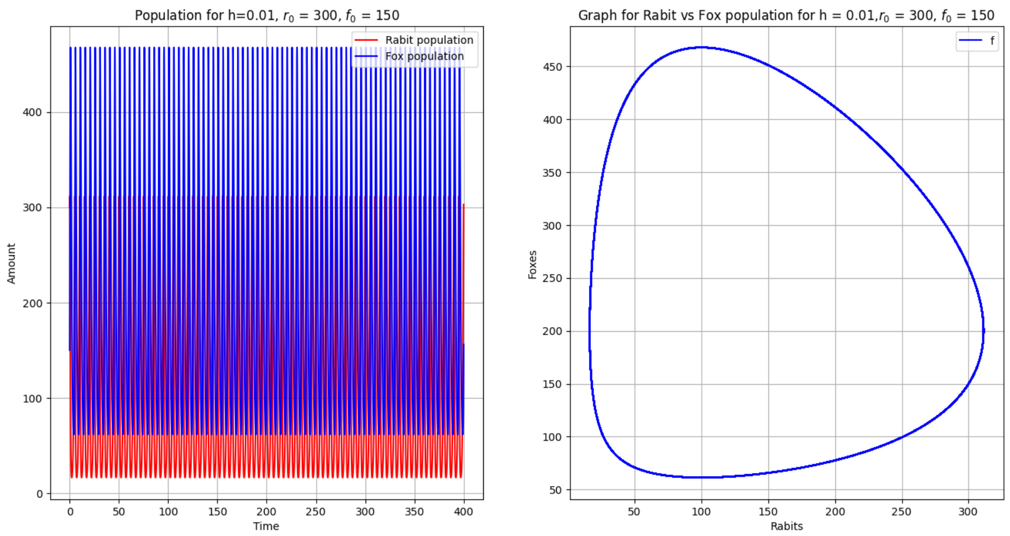

For Leapfrog method with start function: Explicit Euler
Population for h=2e-06, r_0 = 15, f_0 = 22
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 2e-06,r_0 = 15, f_0 = 22
f
Foxes
Rabits


For Adams-Bashforth method with start function: Explicit Euler
Population for h=0.32, r_0 = 300, f_0 = 150
1e215
Rabit population
Fox population
Amount
Time
Graph for Rabit vs Fox population for h = 0.32,r_0 = 300, f_0 = 150
1e215
f
Foxes
Rabits
1e215

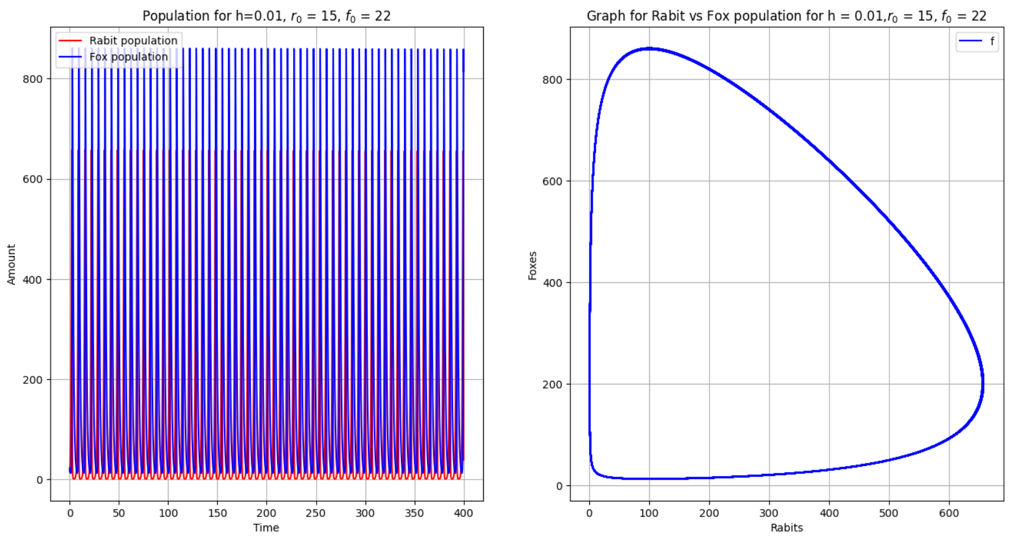
For Adams-Bashforth method with start function: Runge-Kutta 2nd order



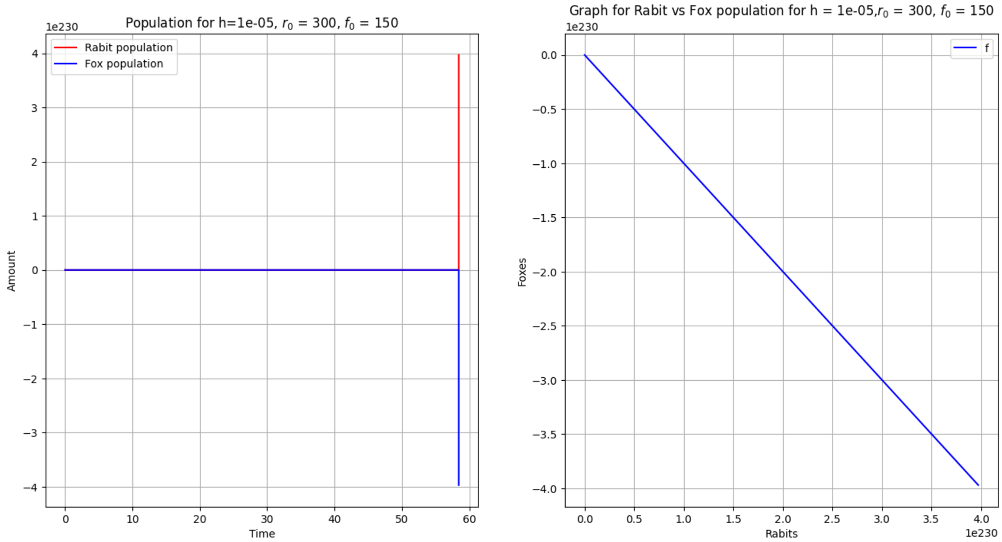
For Adams-Bashforth method with start function: Explicit Euler



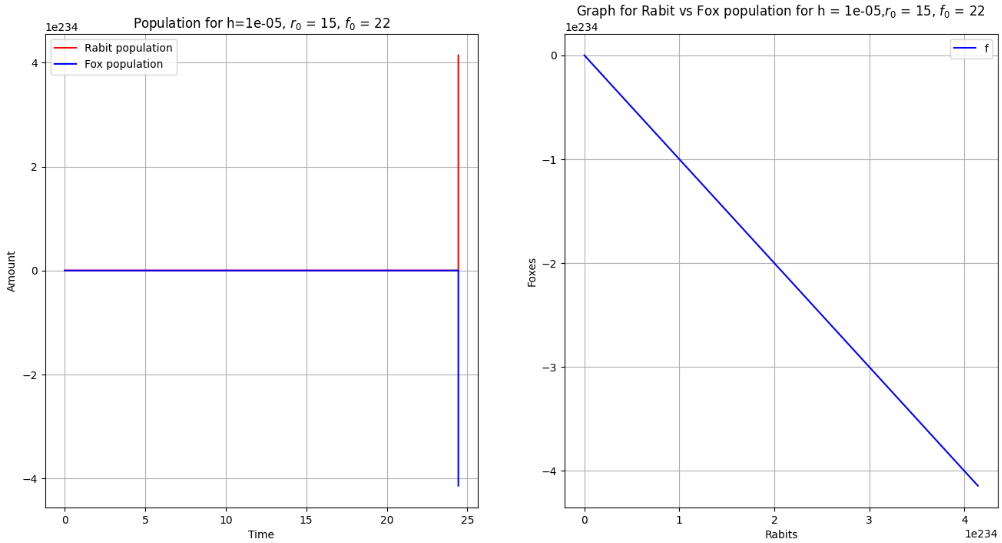
For Adams-Bashforth method with start function: Runge-Kutta 2nd order



For Leapfrog method with start function: Runge-Kutta 2nd order



For Leapfrog method with start function: Runge-Kutta 2nd order



Problem 5

Problem:

$$\frac{d^2x}{dt^2} + \mu\frac{dx}{dt} + bx + cx^3 = k\cos(\omega t), \tag{1}$$

$\mu = 1, b = -10, c = 100, \omega = 3.5$, and $k = 1.55$.

a) Integrate the equation for a sufficiently long time so that all transients due to arbitrary initial conditions have decayed.

b) Plot the solution as a function of time for a sufficiently long duration to confirm chaotic behavior.

c) Since Eq. (1) is invariant under the transformation $x \to x, y \to y$, and $t \to t + 2\pi n/\omega$ for $n = 0, 1, \ldots$, contract the Poincare section by plotting all points in the phase plane with coordinates $[x(\frac{2\pi n}{\omega}), y(\frac{2\pi n}{\omega})]$, where $y = dx/dt$.

Next, consider the coupled anharmonic system:

$$\frac{d^2x}{dt^2} + \mu\frac{dx}{dt} + bx + cx^3 = k\cos(\omega t) + g\frac{du}{dt},$$
$$\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + bu + cu^3 = k\cos(\omega t) + g\frac{dx}{dt},$$

where g is the coupling constant. For $g = 0$, the equations are decoupled, and for the same parameter values chosen earlier, the individual equations will produce chaotic solutions.

Examine the behavior of the coupled system as the coupling parameter g varies. More specifically, plot solutions $x(t), u(t)$ and their respective Poincare sections for values of g between 0 and 0.4 in increments of 0.05.

Solution overview: First let's rewrite equation of second order to system of equations of the first order:

$$\begin{cases} \frac{dx}{dt} = z \\ \frac{dz}{dt} + \mu z + bx + cx^3 = k\cos(\omega t) \end{cases} \Leftrightarrow \begin{cases} \frac{dx}{dt} = z \\ \frac{dz}{dt} = k\cos(\omega t) - \mu z - bx - cx^3 \end{cases}$$

Now we can solve it using scipy method solve_ivp with method RK45 - Runge-Kutta of 5th order. Since our solution is chaotic it doesn't matter what initial conditions are. I chose $x(0) = 0, x'(0) = z(0) = 1$. I have specified time of integration from 0 to 300, with 10000000 points in between.

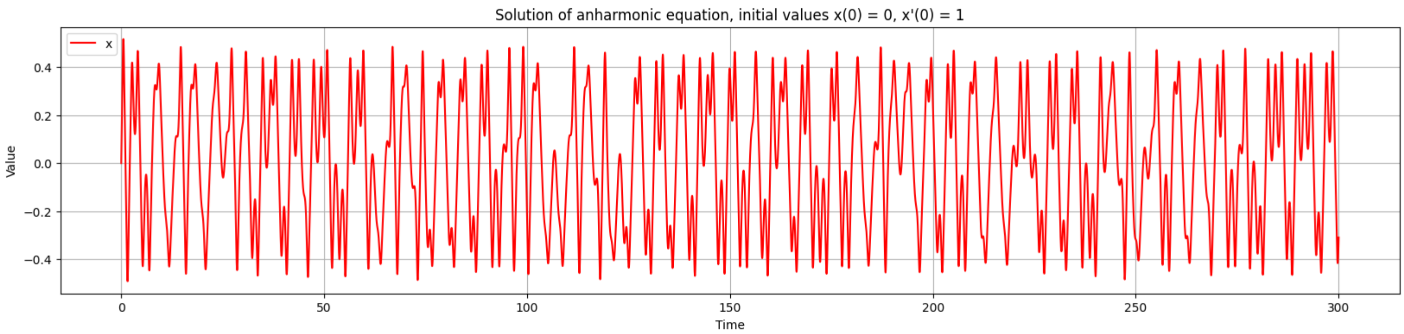
To get contracted Poincare section I have created a mask for right indexes(ind variable in code). I get it by using condition: abs(t[i] - 2*np.pi*k/w)¡1e-4 - since we can not integrate continuously and need to take into account the difference between grid and actual values. On the graph on Poincare section it's clear that we get strange attractor case for one equation.

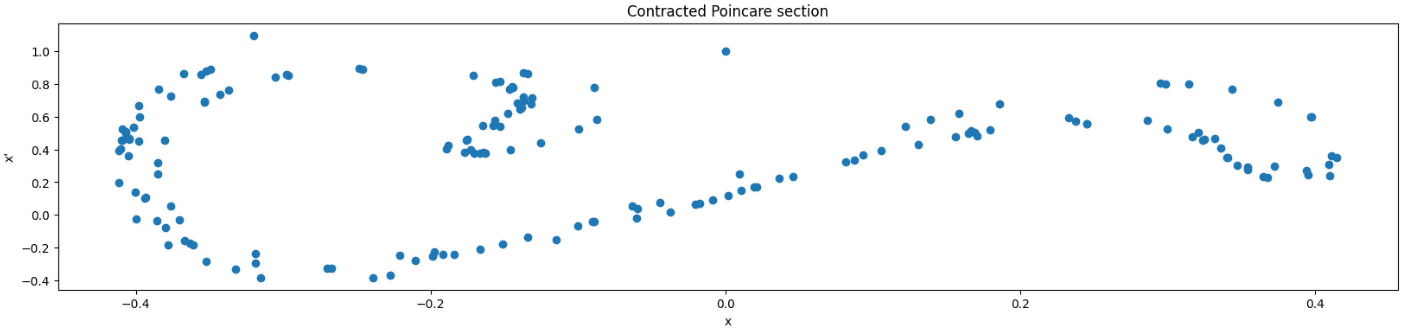
Let's examine system of coupled equation. For that as previously, from system of two equations of second order I got system of equations of the fist order. The procedure is similar. Next I add a small perturbation (0.01) to initial condition (eps variable in code) to get different solutions. Time interval and the grid I chose the same as for one equation. For contracted Poincare section I used previous mask. From the plots we can see that for $g \leq 0.25$ systems behavior is chaotic, but for $g \geq 0.3$ systems becomes periodic and we get cycle diagram for phase space. Additionally, I plot full phase diagram to see the pattern.

Based on these results, the following observation can be made:

- Parameter g influences the chaotic behavior of the coupled system. The bigger the g the more organized the system.

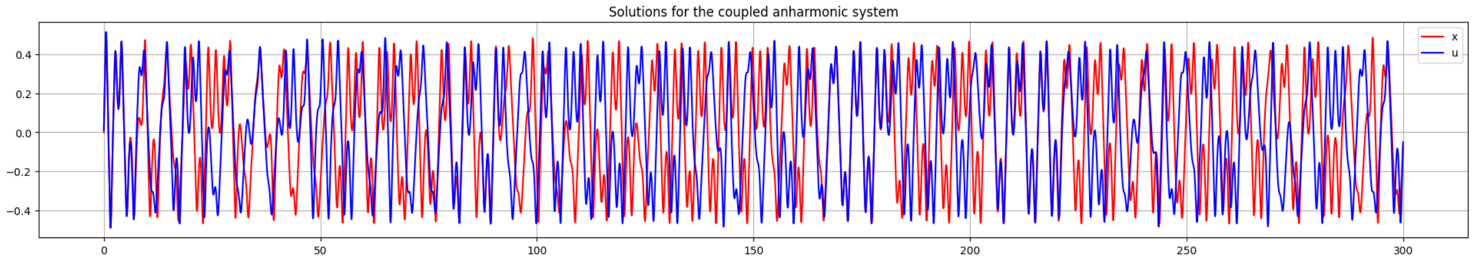
Plots for solution of one equation:

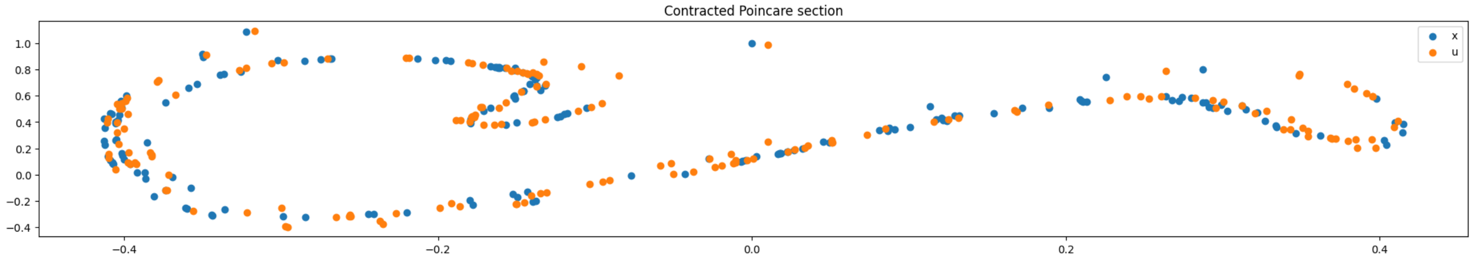




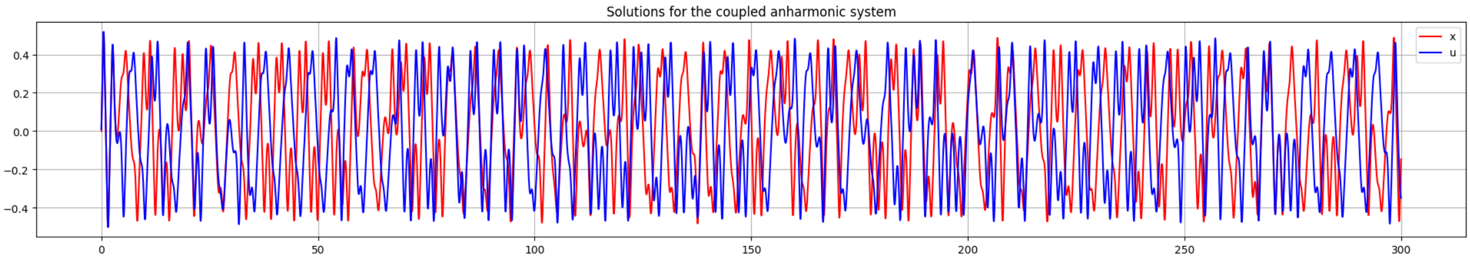
Plots for coupled system:

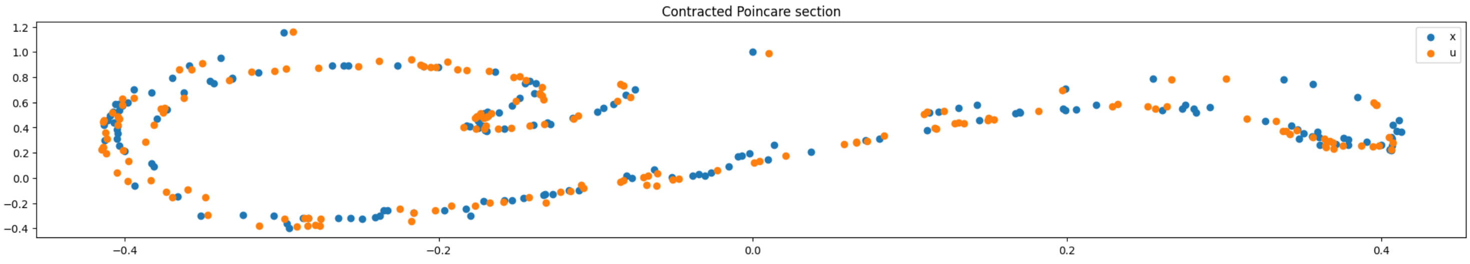
For g = 0.0

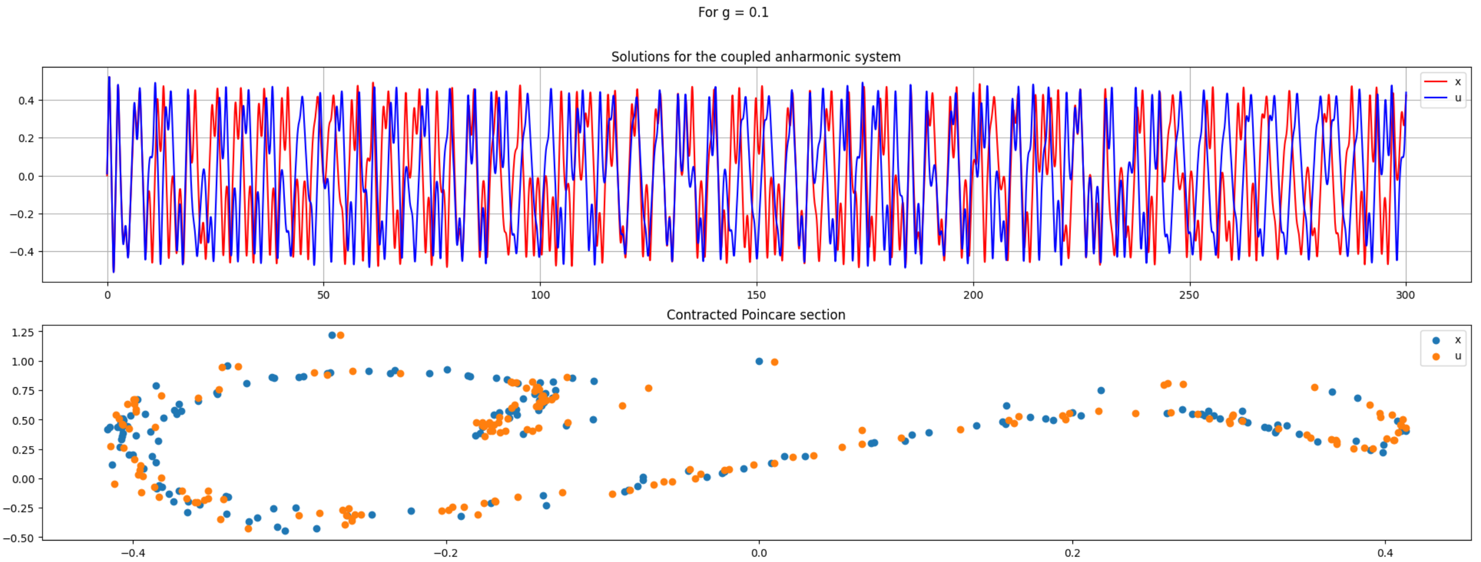


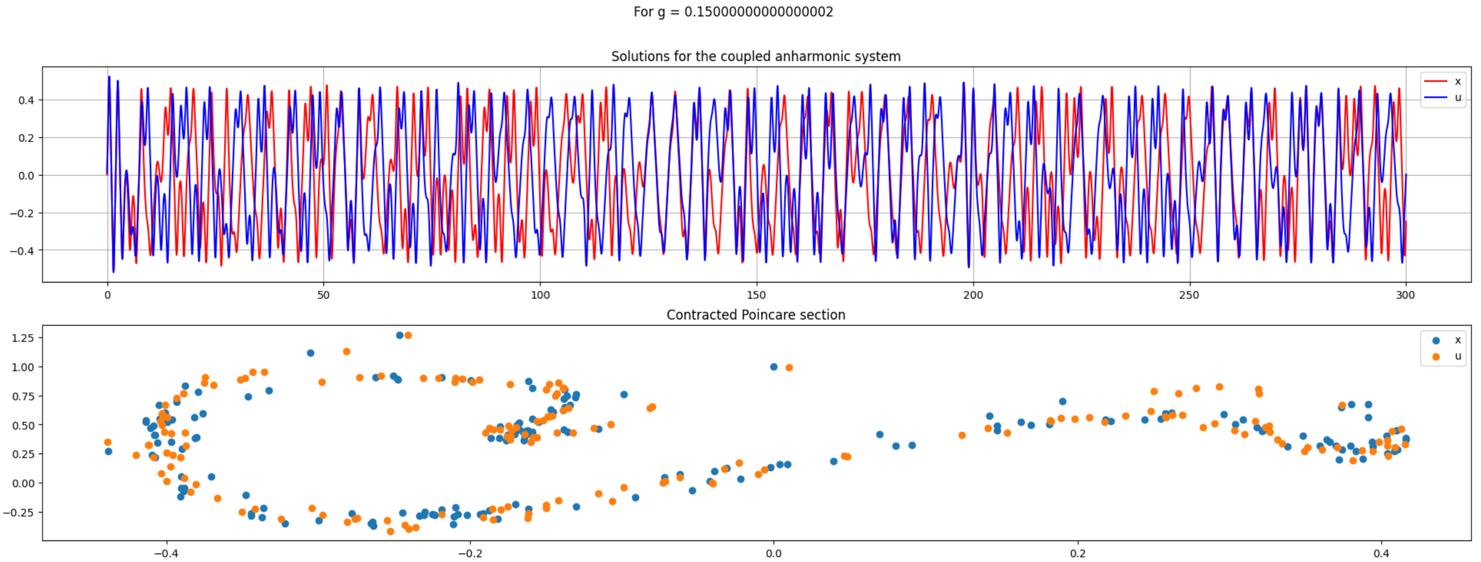


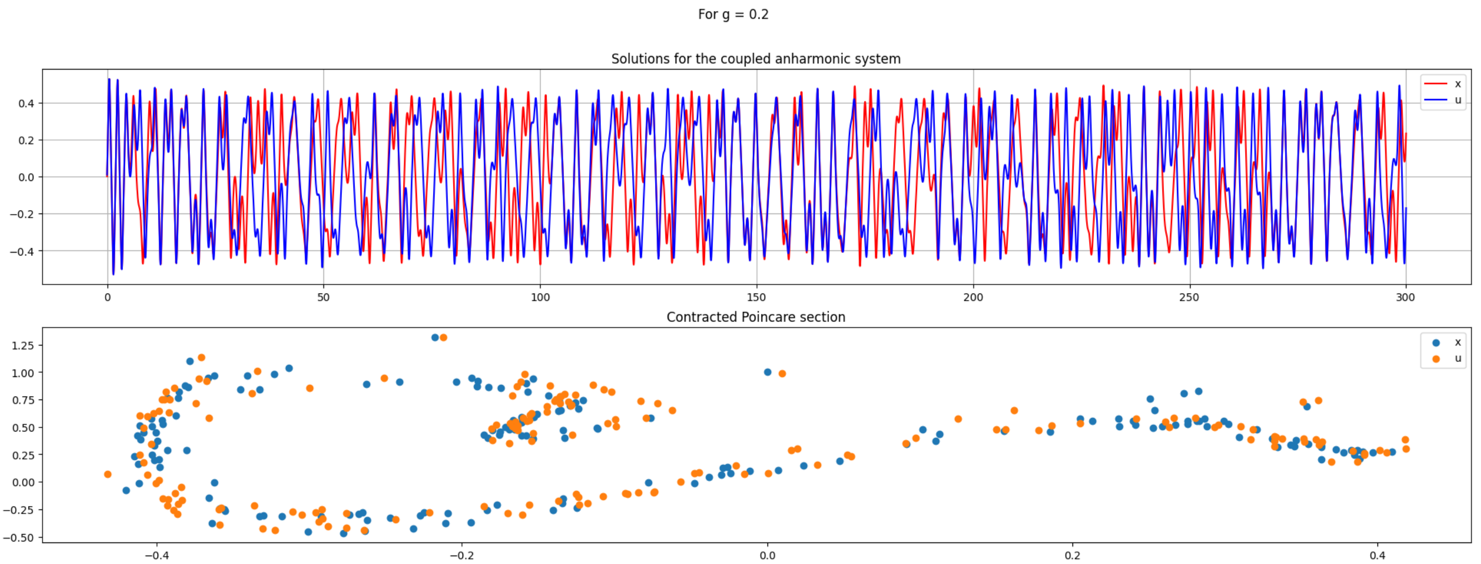
For g = 0.05



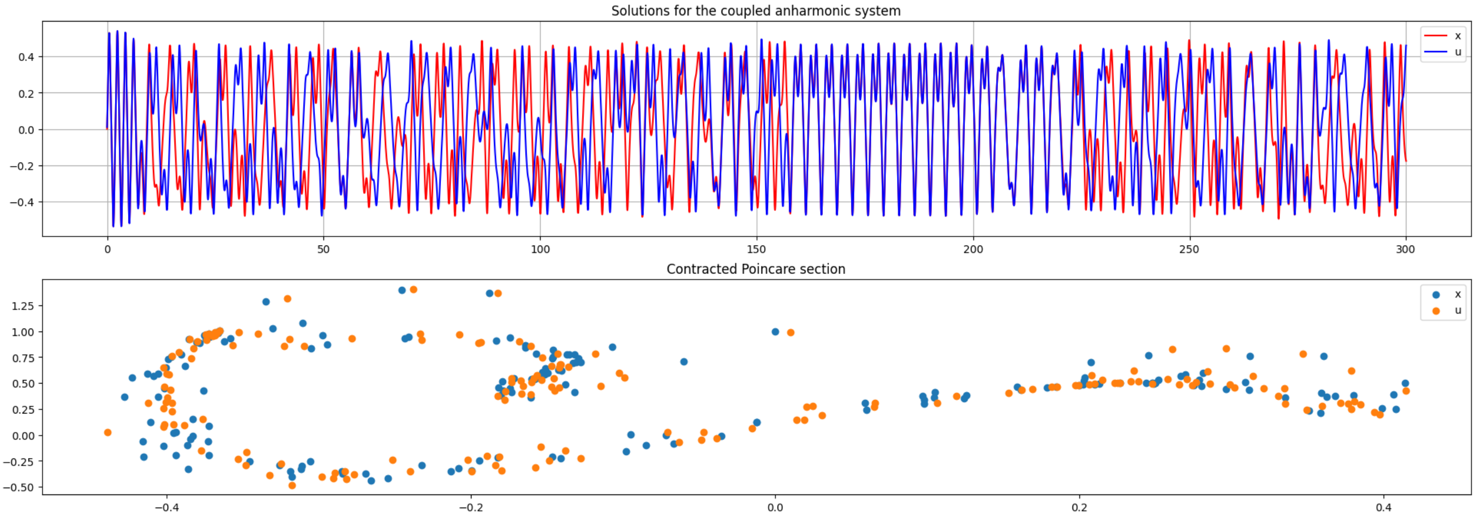



For g = 0.1
Solutions for the coupled anharmonic system
x
u
Contracted Poincare section
x
u

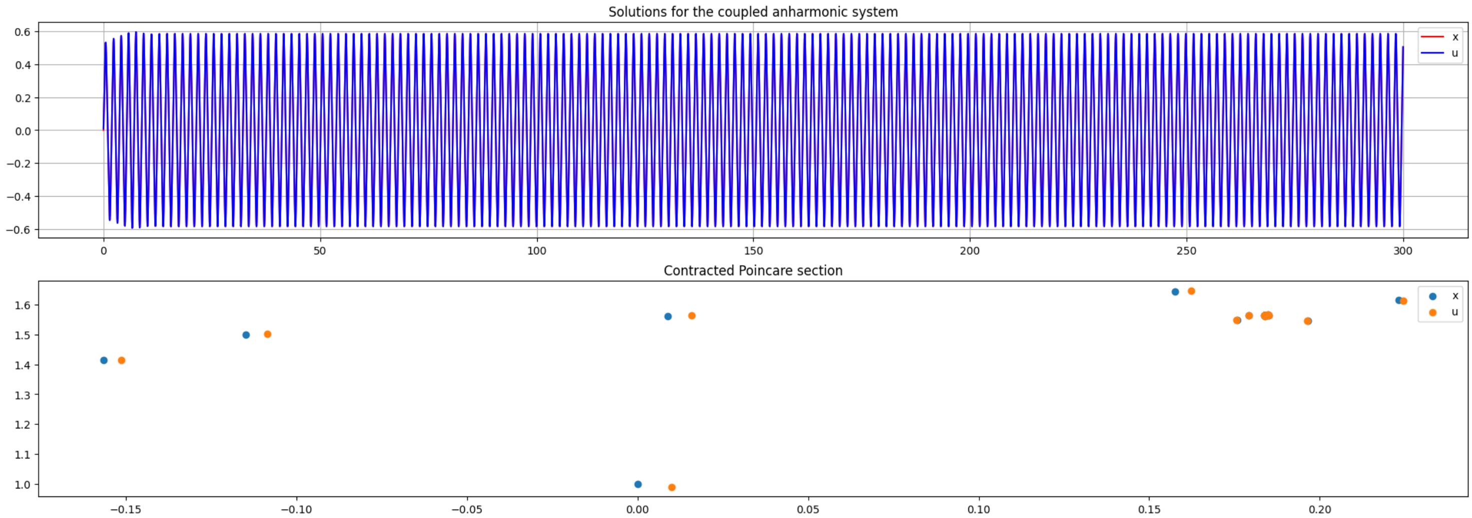

For g = 0.15000000000000002
Solutions for the coupled anharmonic system
x
u
Contracted Poincare section
x
u


For g = 0.2
Solutions for the coupled anharmonic system
x
u
Contracted Poincare section
x
u

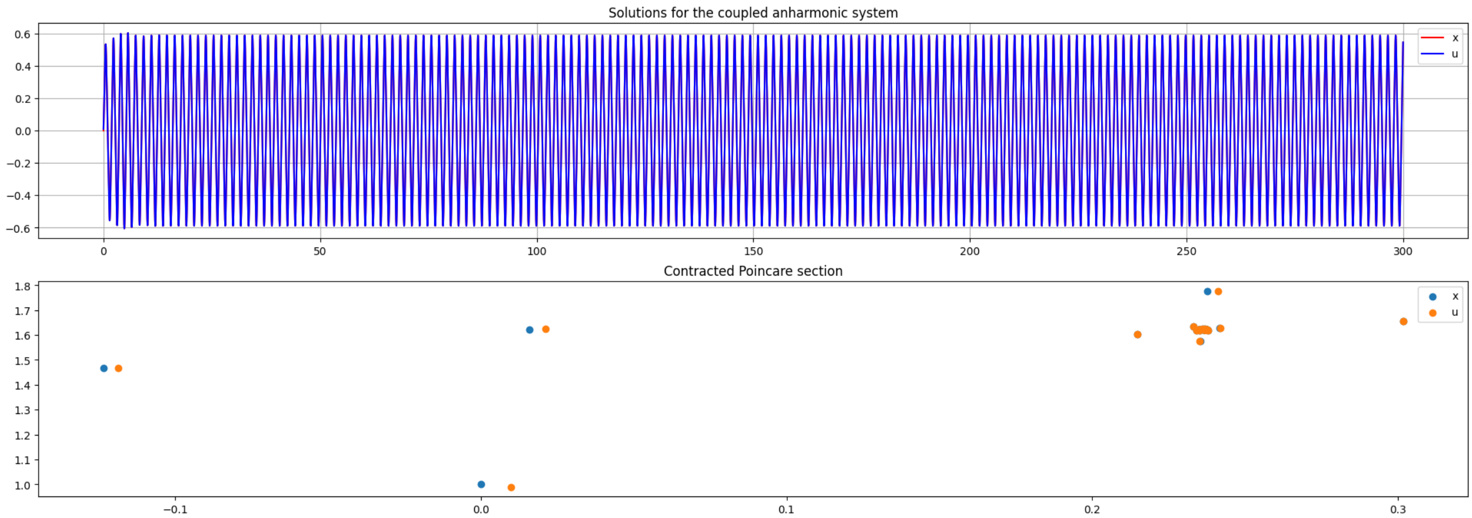
For g = 0.25

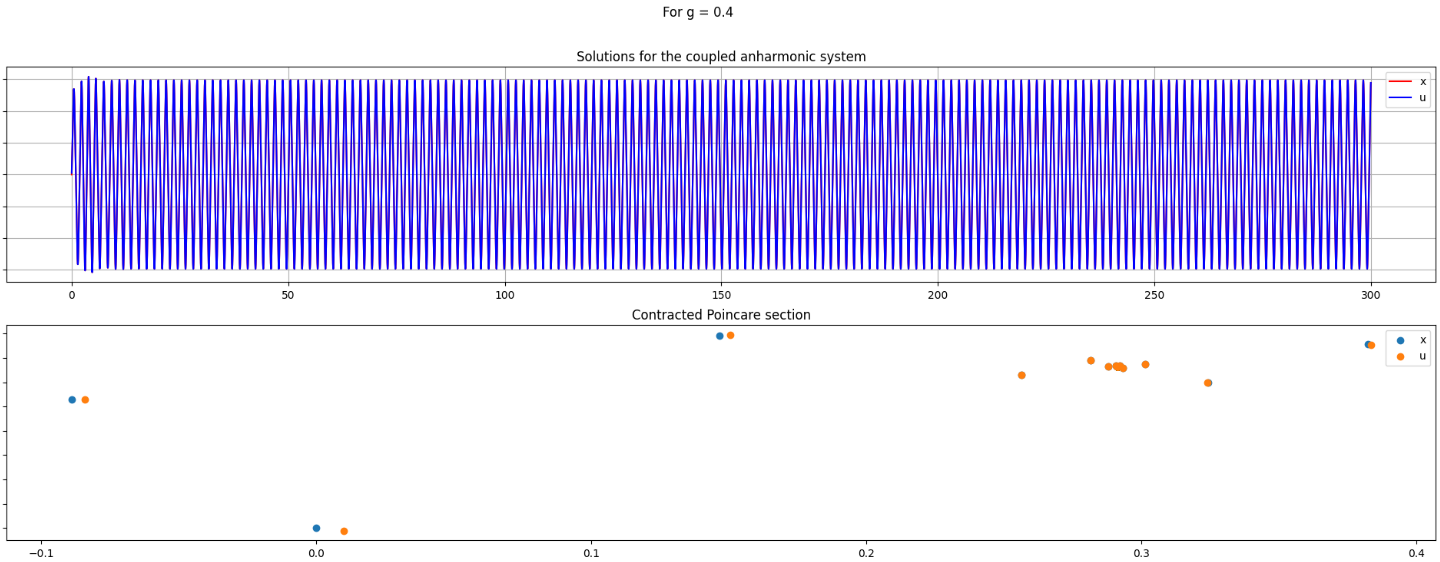


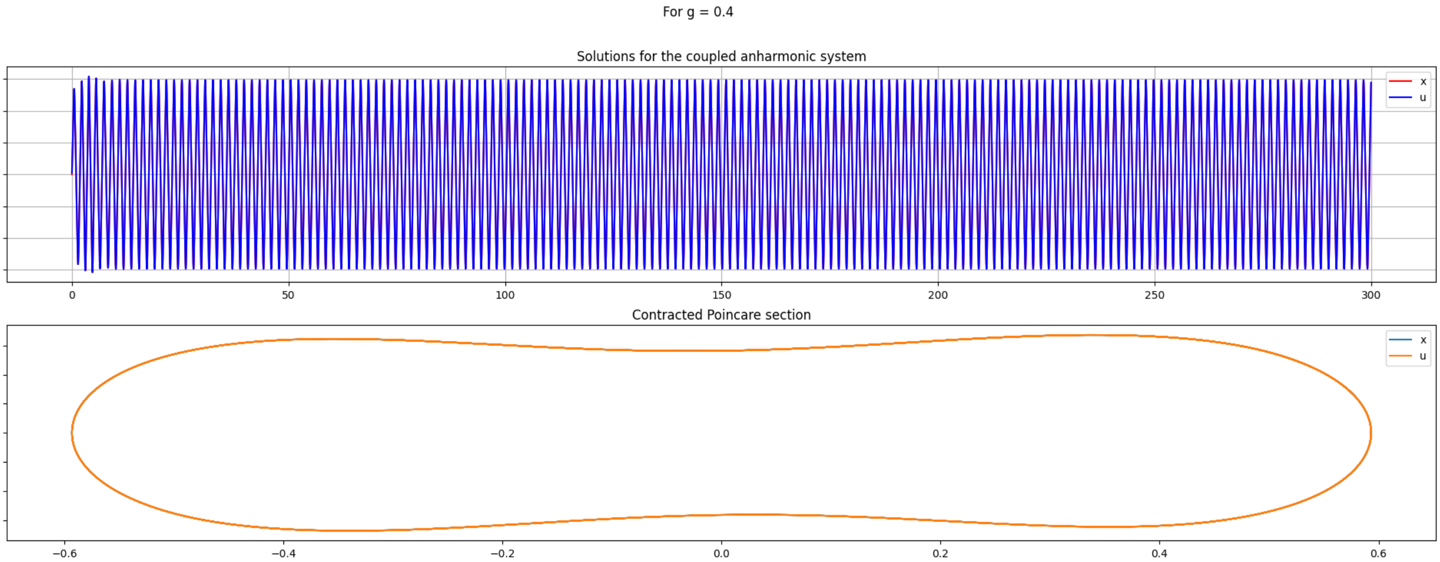
For g = 0.30000000000000004



For g = 0.35000000000000003




For g = 0.4
Solutions for the coupled anharmonic system
Contracted Poincare section
x
u


For g = 0.4
Solutions for the coupled anharmonic system
Contracted Poincare section
x
u