AMATH 383 Problem Set 3 Questions 1, 2, 4

Leo Wei

July 2023

1 Tung 7.7.2. Beverton-Holt model.

a. Find the equilibria and determine their stability

To find the equilibria of the difference equation

$$N_{n+1} = \frac{RN_n}{1+\frac{R-1}{K}N_n} \; R > 1$$

We set $N^* = f(N^*)$ and solve for N^*

$$N^* = \frac{RN^*}{1+\frac{R-1}{K}N^*}$$

$$N^* + \frac{R-1}{K}N^{*2} = RN^* \rightarrow N^*(\frac{R-1}{K}N^* + (1-R)) = 0$$

We find that the equilibria are $N_1^* = 0$ and $N_2^* = K$ To find the stability of each equilibria, we will analyze the system near these points. First we find the derivative of this equation

$$\frac{d}{dN_n}(\frac{RN_n}{1+\frac{R-1}{K}N_n})$$

We take the R out of the numerator as it is a constant, set $\frac{R-1}{K} = a$ and apply the quotient rule

$$\frac{\frac{dN_n}{dN_n}(1+aN_n) - \frac{d(1+aN_n)}{dN_n}N_n}{(1+aN_n)^2}$$

Which will simplify to

$$\frac{R}{(1+aN_n)^2} = \frac{R}{(1+\frac{R-1}{K}N_n)^2}$$

Now we plug in both equilibria to determine the stability. At $N_1^* = 0$,

$$\frac{df}{dN_n} = R$$

At $N_2^* = K$,

$$\frac{df}{dN_n} = \frac{1}{R}$$

The equilibria is considered stable if $|f'(N^*)| < 1$. Thus, since the given condition is $R > 1$, N_1^* is unstable since $|f'(N_1^*)|$ is not less than 1 while N_2^* is stable since $|f'(N_2^*)|$ is less than 1.

b. Find an exact, closed form solution. (Hint: Use the substitution $X_n = \frac{1}{N_n}$. The equation for X_n turns out to be linear.) This is one of the rare cases where a nonlinear difference equation can be solved exactly.

We substitute in $X_n = \frac{1}{N_n}$ and by inferring $X_{n+1} = \frac{1}{N_{n+1}}$ into the equation, taking $\frac{R-1}{K} = a$. We will also rewrite the original equation into a simplified version

$$N_{n+1} = \frac{RN_n}{1 + \frac{R-1}{K}N_n} \rightarrow N_{n+1} = \frac{R}{\frac{1}{N_n} + a}$$

From there, we substitute X_n and X_{n+1}

$$\frac{1}{X_{n+1}} = \frac{R}{X_n + a}$$

We "flip" both sides of the equation by applying $\frac{1}{equation}$

$$X_{n+1} = \frac{X_n + a}{R}$$

We then solve for $n = 0, 1, 2...$

$$X_1 = \frac{1}{R}(X_0 + a)$$

$$X_2 = \frac{a}{R}(1 + \frac{1}{R}) + \frac{1}{R_2}X_0$$

$$X_3 = \frac{a}{R}(1 + \frac{1}{R} + \frac{1}{R^2}) + \frac{1}{R^3}X_0$$

Thus we can infer the relation

$$X_n = \frac{a}{R}(1 + \frac{1}{R} + ... + \frac{1}{R^{n-1}}) + \frac{X_0}{R^n}$$

Simplifying the equation

$$X_n = \frac{a}{R}\frac{1 - (\frac{1}{R})^n}{1 - \frac{1}{R}} + \frac{X_0}{R^n}$$

$$X_n = \frac{a}{R^n}\frac{R^n - 1}{R - 1} + \frac{X_0}{R^n}$$

Plugging back in $\frac{R-1}{K} = a$ and $N_n = \frac{1}{X_n}$

$$X_n = \frac{R^n - 1 + X_0 K}{R^n K} \to N_n = \frac{R^n K}{R^n - 1 + X_0 K}$$

Where X_0 is corresponding to the initial population N_0 through the relation $X_n = \frac{1}{N_0}$

2 Construct the cobweb diagram for the following and determine the stability of fixed points in each case for any real-valued r. For part (b), find all of the equilibria and only determine the stability of equilibrium at the origin.

a. $N_{t+1} = \frac{(1+r)N_t}{1+rN_t}$ $N_t \geq 0$

We first solve for the equilibria N^*

$$N^* = \frac{(1+r)N^*}{1+rN^*} \to N^* + rN^{*2} = N^* + rN^*$$

$$rN^{*2} - rN^* = 0 \to rN^*(N^* - 1) = 0$$

$$N_1^* = 0, N_2^* = 1$$

To determine the stability of each equilibria, we take the derivative at each point.

$$\frac{d}{dN_t}(N_{t+1}) = \frac{d}{dN_t}(\frac{(1+r)N_t}{1+rN_t})$$

Using the quotient rule, we find that

$$\frac{d}{dN_t}(N_{t+1}) = \frac{1+r}{(1+rN_t)^2}$$

Substitute in N_1^* to determine the stability at $N = 0$

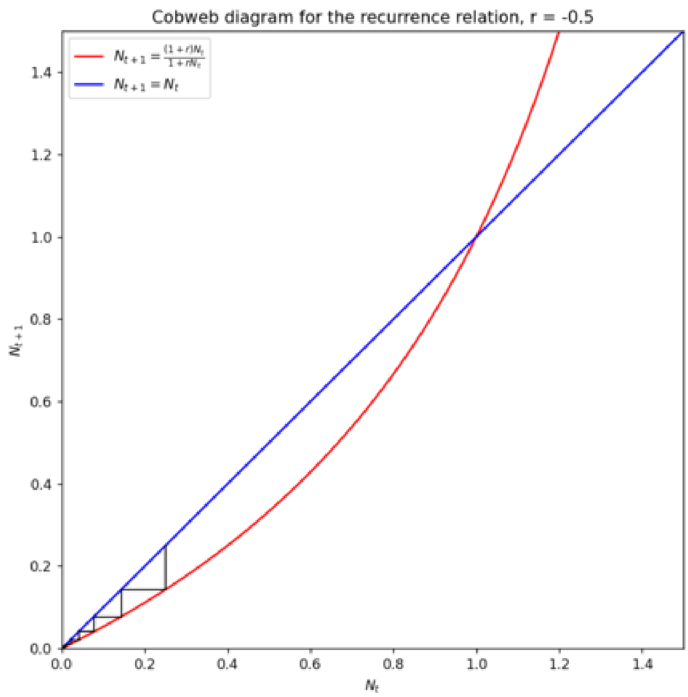
$$N'_{t+1} = 1 + r$$

The stability condition is if $|1 + r| < 1$, so $-2 < r < 0$ Substitute in N_2^* to determine the stability at $N = 1$

$$N'_{t+1} = -\frac{r}{r+1} + 1$$

The stability condition is if $\frac{1}{|1+r|} < 1$, so $r > 0$ We can come to the conclusion that there are no cases where both equilibria are stable. I will plot 2 cobweb diagrams for 2 different r, for each stable equilibria For N_1^* to be stable, we pick

$r = -0.5$ and $N_t = 0.25$ to start at a point near N_1^* We plot it in Python

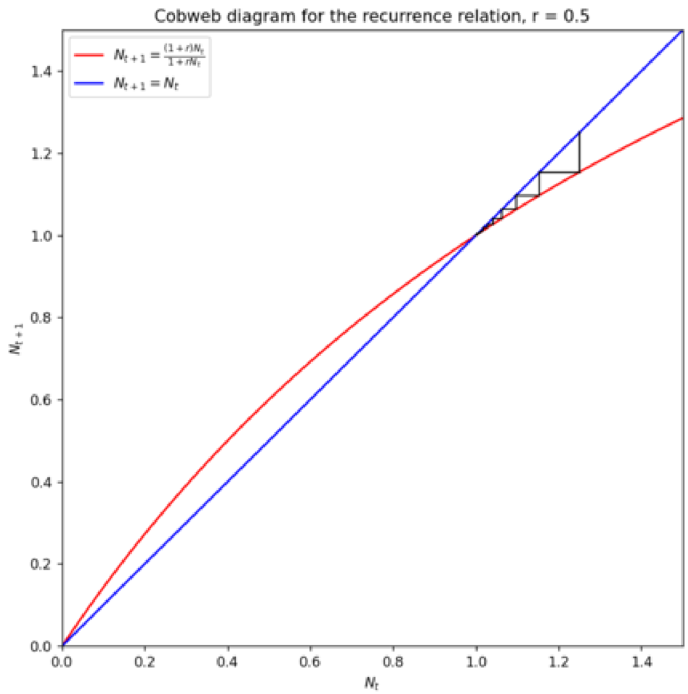


Here is the code

amath.py > plot_cobweb_for_recurrence_N > r

```
import numpy as np
import matplotlib.pyplot as plt

# Define the function for the recurrence relation with a specific r
def recurrence_function_N(N, r = -0.5):
    return ((1+r)*N)/(1+r*N)

def plot_cobweb_for_recurrence_N(ax, initial_value, r = -0.5, iterations=50):
    N_values = np.linspace(0, 1.5, 1000)
    N_t = initial_value

    # Plot N_{t+1} = f(N_t) and N_{t+1} = N_t
    ax.plot(N_values, recurrence_function_N(N_values, r), 'r', label='$N_{t+1} = \\frac{(1+r)N_t}{1+rN_t}$')
    ax.plot(N_values, N_values, 'b', label='$N_{t+1} = N_t$')

    # Generate the cobweb
    for i in range(iterations):
        N_t1 = recurrence_function_N(N_t, r)
        ax.plot([N_t, N_t], [N_t, N_t1], 'k', linewidth=1)  # Vertical line
        ax.plot([N_t, N_t1], [N_t1, N_t1], 'k', linewidth=1)  # Horizontal line
        N_t = N_t1

    ax.set_xlim([0, 1.5])
    ax.set_ylim([0, 1.5])
    ax.set_xlabel('$N_t$')
    ax.set_ylabel('$N_{t+1}$')
    ax.set_title('Cobweb diagram for the recurrence relation, r = {}'.format(r))
    ax.legend()
# Plot cobweb diagram starting at N_t = 0.25 and r = -0.5
fig, ax = plt.subplots(figsize=(8, 8))
plot_cobweb_for_recurrence_N(ax, initial_value = 0.25, r = -0.5)
plt.show()
```

For N_2^* to be stable, we pick $r = 0.5$



Here is the code

```python
amath.py > recurrence_function_N > r
import numpy as np
import matplotlib.pyplot as plt

# Define the function for the recurrence relation with a specific r
def recurrence_function_N(N, r = 0.5):
    return ((1+r)*N)/(1+r*N)

def plot_cobweb_for_recurrence_N(ax, initial_value, r = 0.5, iterations=50):
    N_values = np.linspace(0, 1.5, 1000)
    N_t = initial_value

    # Plot N_{t+1} = f(N_t) and N_{t+1} = N_t
    ax.plot(N_values, recurrence_function_N(N_values, r), 'r', label='$N_{t+1} = \\frac{(1+r)N_t}{1+rN_t}$')
    ax.plot(N_values, N_values, 'b', label='$N_{t+1} = N_t$')

    # Generate the cobweb
    for i in range(iterations):
        N_t1 = recurrence_function_N(N_t, r)
        ax.plot([N_t, N_t], [N_t, N_t1], 'k', linewidth=1)  # Vertical line
        ax.plot([N_t, N_t1], [N_t1, N_t1], 'k', linewidth=1)  # Horizontal line
        N_t = N_t1

    ax.set_xlim([0, 1.5])
    ax.set_ylim([0, 1.5])
    ax.set_xlabel('$N_t$')
    ax.set_ylabel('$N_{t+1}$')
    ax.set_title('Cobweb diagram for the recurrence relation, r = {}'.format(r))
    ax.legend()
# Plot cobweb diagram starting at N_t = 1.25 and r = 0.5
fig, ax = plt.subplots(figsize=(8, 8))
plot_cobweb_for_recurrence_N(ax, initial_value = 1.25, r = 0.5)
plt.show()
```

We can see for both cases they converge at each respective equilibria

b. $x_{n+1} = -x_n - \frac{2}{3}x_n^3 + \frac{4}{3}x_n^4$

We solve for the equilibria x^*

$$x^* = -x^* - \frac{2}{3}x^{*3} + \frac{4}{3}x^{*4}$$

$$2x^* + \frac{2}{3}x^{*3} - \frac{4}{3}x^{*4} = 0$$

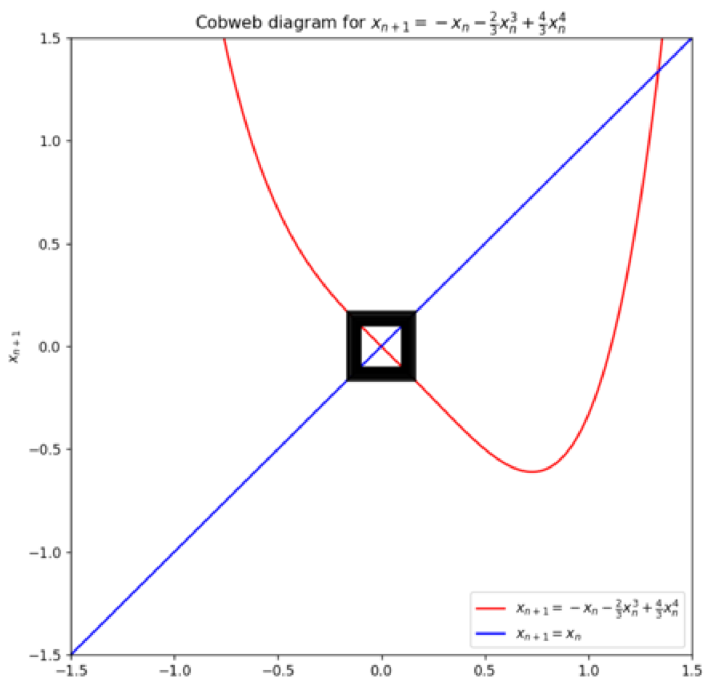
Solving for x^* gives us

$$x_1^* = 0, x_2^* \approx 1.34$$

To determine the stability of the origin $x = 0$, we find the derivative at this point

$$\frac{d}{dx}(x_{n+1}) = \frac{d}{dx}(-x_n - \frac{2}{3}x_n^3 + \frac{4}{3}x_n^4)$$

Since we are determining $f'(0)$, we would realize that only the first term matters, as the rest vanishes due to $x = 0$. This gives us $|f'(0)| = |-1| = 1$. Since the derivative at this point is exactly 1, we cannot determine the stability of this equilibria based on numerical analysis alone. Therefore, we shall construct a cobweb diagram to determine its stability. We will use Python to plot the diagram



Here is the code for the plot

amath.py > plot_cobweb_for_recurrence

```python
import numpy as np
import matplotlib.pyplot as plt

# Define the function for the recurrence relation
def recurrence_function(x):
    return -x - (2/3)*x**3 + (4/3)*x**4

def plot_cobweb_for_recurrence(ax, initial_value, iterations=50):
    x_values = np.linspace(-1.5, 1.5, 1000)
    x_n = initial_value

    # Plot x_{n+1} = f(x_n) and x_{n+1} = x_n
    ax.plot(x_values, recurrence_function(x_values), 'r', label='$x_{n+1} = -x_n - \\frac{2}{3}x_n^3 + \\frac{4}{3}x_n^4$')
    ax.plot(x_values, x_values, 'b', label='$x_{n+1} = x_n$')

    # Generate the cobweb
    for i in range(iterations):
        x_n1 = recurrence_function(x_n)
        ax.plot([x_n, x_n], [x_n, x_n1], 'k', linewidth=1)  # Vertical line
        ax.plot([x_n, x_n1], [x_n1, x_n1], 'k', linewidth=1)  # Horizontal line
        x_n = x_n1

    ax.set_xlim([-1.5, 1.5])
    ax.set_ylim([-1.5, 1.5])
    ax.set_xlabel('$x_n$')
    ax.set_ylabel('$x_{n+1}$')
    ax.set_title('Cobweb diagram for $x_{n+1} = -x_n - \\frac{2}{3}x_n^3 + \\frac{4}{3}x_n^4$')
    ax.legend()

# Plot cobweb diagram starting close to the origin
fig, ax = plt.subplots(figsize = (8, 8))
plot_cobweb_for_recurrence(ax, initial_value=0.1)
plt.show()
```

From the cobweb disgram, we can see that the fixed point at the origin is an unstable equilibria

4 Tung 3.7.6. Power of compounding.

a. You borrowed \$1,000 from a loan shark at 5% monthly interest. How much do you owe four years later?

To see how much we owe the loan shark, we first calculate how much time has passed

$$12 * 4 = 48 \; months$$

Since we have a 5% monthly interest, we multiply it by 12 to get the annual rate

$$\frac{5}{100} * 12 = 0.6$$

which is 60% per year We then use the equation for compound interest

$$A = P(1 + \frac{r}{n})^{nt}$$

where r = annual rate, n = number of times that it is compounded, t = time passed, and P is initial amount of money.
Plugging in the numbers

$$A = 1000(1 + \frac{0.6}{12})^{12*4} = 10401.27$$

You would owe the loan shark \$10401.27

b. You are trying to build a nest egg for your retirement. You have estimated that you will need income of $5,000 a month in order to live comfortably after retirement. How large a nest egg (principal P0) must you have at retirement? Assume that at your retirement you put that money, P0, in an annuity with a guaranteed annual return of 8%. And suppose you think you will live forever.

To find the principal P_0, we use the formula

$$P_0 = \frac{C}{r}$$

where C = periodic payment or in this case the amount to withdraw each month and r = periodic interest rate. Since we are looking to withdraw each month, we should divide the annual interest rate by 12 to obtain the monthly interest rate

$$r = \frac{0.08}{12}$$

Plugging in all the numbers

$$P_0 = \frac{5000}{\frac{0.08}{12}} \approx \$750000$$

You would need approximately $750000 at retirement to live comfortably with an APY of 8% (Which frankly speaking is impossible in the current economy)

c. You are now 25 and plan to retire at age 65. You want to start saving so that you can build a nest egg of $1 million. How much should you save each month? Assume that your savings will be earning 10% interest each year, compounded monthly.

We have 40 years or 480 months to save up to 1000000. The annual interest rate is 10% compounded monthly. Therefore, the rate of interest gain would be

$$r = 0.1/12 \approx 0.0083$$

To find the value of an annuity, we use the formula

$$\frac{(1+r)n - 1}{r}$$

where r is rate of interest gain and n is time taken. Plugging in the numbers

$$\frac{(1+0.0083)480 - 1}{0.0083} \approx 6324.08$$

The target future value is given by $Monthly * Annuity$. Therefore to find the amount needed to save monthly, we get $\frac{Future}{Annuity}$

$$Montly = \frac{1000000}{6324.08} \approx \$158.13$$

You would need to save about \$158.13 per month to save \$1000000