Homework #01

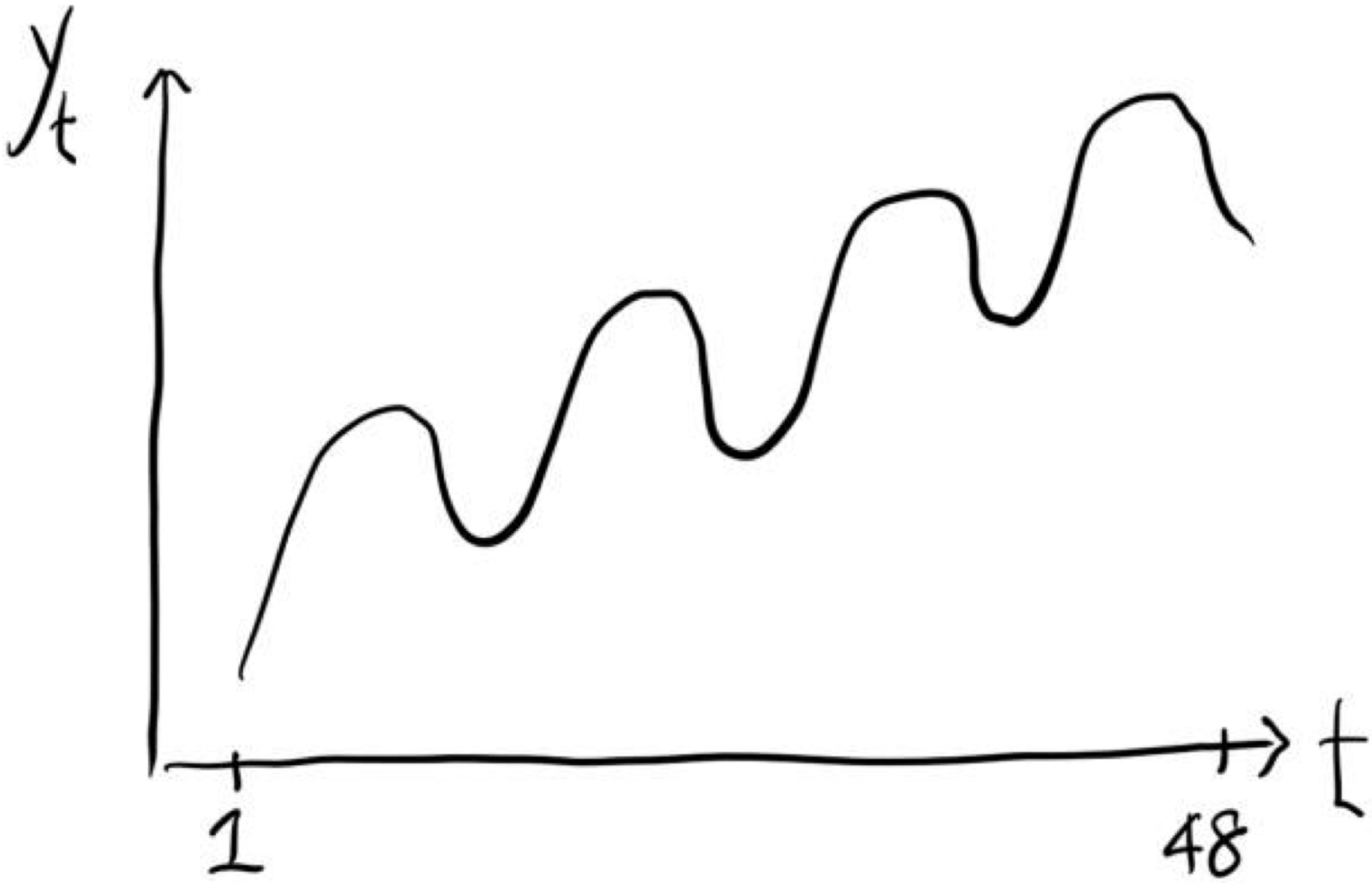
Dhanabordee Mekintharanggur T11902203

Project setup and imports

```
# Imports
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb

# Figure settings
sb.set(rc={'figure.figsize':(14,7)})
sb.set_style({'axes.grid' : True})
```

1. (10%) Write down the scientific procedures to simulate a time series of length 48 similar to the trend below. Use any functions you prefer.



The Problem

To simulate the plot given in the sketch using any function necessary

The Approach

The Scientific Method and the Generic Scheme of Model Building can be use in tandem with the Generic Scheme of Model Building.

Scientific Mehtod

Given the sketch of the time series plot, the scientific method can be applied to determine the process that must be done to achieve the intended results.

The scientific method can achieved in 6 steps including:

1. Making observations
2. Asking questions
3. Form a hypothesis
4. Make a prediction based on the hypothesis
5. Test the prediction
6. Iterate

The Generic Scheme of Model Building

The Generic Scheme of Model Building is iteratively applied to systematically analyze the models and parameters required to simulate the sketch.

1. Postulate general class of models
2. Identify model to be tentatively entertained
3. Estimate parameters in tentatively entertained model
4. Diagnostic checking (is the model adequate?) If it is, continue to step 5. If not, reiterate from step 2.
5. Use the model for forecasting or control

The Process

1. Making observations + Postulate general class of models

Making Observations

- The plot shows a cycle that increases and decreases periodically
- The period of the cycle is 12 units
- The overall plot trends upward over time
- The domain is from t = 1 and t = 48 units
- Y sub t begins as a non-zero positive value

Postulate general class of models

- Since the X axis is represented by a variable t, the model to be used should be classified in time series model.

2. Asking questions

- How create a plot that increases and decreases periodically over time?
- How to create a plot that starts with a non-zero positive value and starts by trending upwards?
- How to create a plot that repeats itself 4 times between t = 1 and t = 48?
- How to simulate the plot as a time series representation?

3. Form a hypothesis + Identify model to be tentatively entertained

- A combination of sinusoidal function and a linear function is tentatively selected to represent the simulation of the sketched plot.
- Models used:
    - Linear function
    - Sinusoidal function

4. Make a prediction based on the hypothesis + Estimate parameters in tentatively entertained model

- A sinusoidal function added by a linear function with a positive slope to simulate the plot that cyclically increases and decreases and trends in a positive direction.
    - The sinusoidal function provides the cyclical pattern
    - The linear function providers the positive slope that will trend in the positive direction linearly
    - When both are summed, the result should appear as a sinusoidal function that linearly trends upwards.
- The parameter of the sinusoudal function to consider includes:
    - the coefficient of t (angular velocity)
    - the phase shift
    - the amplitude of the sinusoidal function
- Prepare the data for the plot (estimate the parameters)
    - Since there are 4 periods in the sketch, and the whole sketch is confied between 1 and 48, the period is 12 units.

- From the formula T = 2π/w => w = π/6, which is the coefficient of t
- The coefficient of the linear function is selected by trial-and-error until the slope matches the sketch

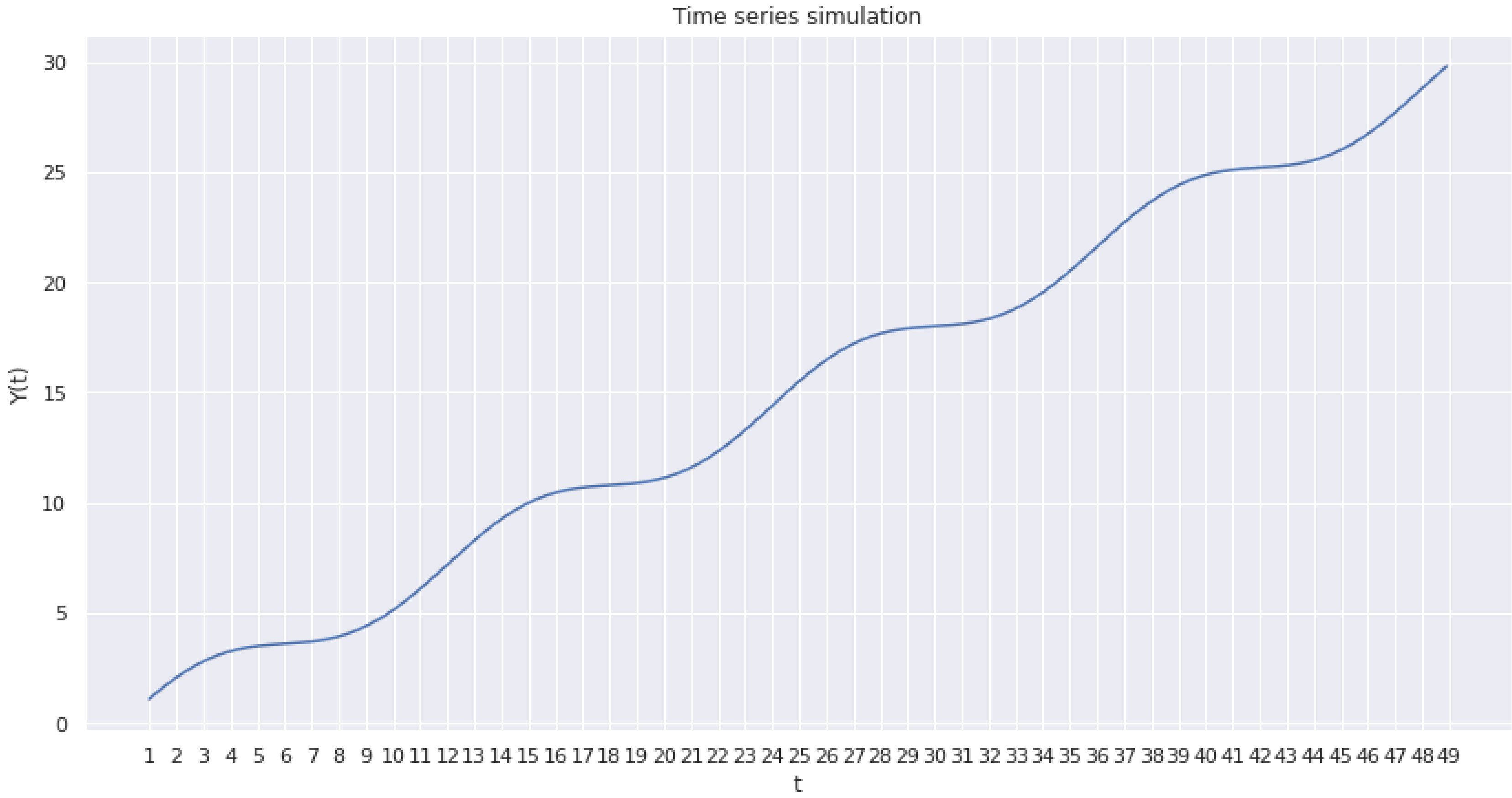
```
t = np.arange(1, 49, 0.1)              # t domain of length 48
y = np.sin((np.pi/6) * t) + 0.6 * t    # function that represents the plot
```

5. Test the prediction + Diagnostic checking (is the model adequate?)

Plot the data

```
plt.xlabel("t")
plt.ylabel("Y(t)")
plt.title('Time series simulation')
plt.xticks(np.arange(min(t), max(t)+1, 1.0))
sb.lineplot(x = t, y = y)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cbdb79a90>
```



Problems identified

- Note that the plot achieved in step 5 appears cyclical, but many features still doesn't match the requirements
- The trough and peaks appear too shallow
- The plot should start around the trough

Diagnostic results

- The model is not adequate enough to represent the sketch

6. Iterate

Iteration 2

1. Adjust the amplitude of the sine function to create a more distinctive separation between the trough and the peak
2. Phase shift the sine function by pi/2 to create a plot that begins from the trough

Prepare the data for the plot

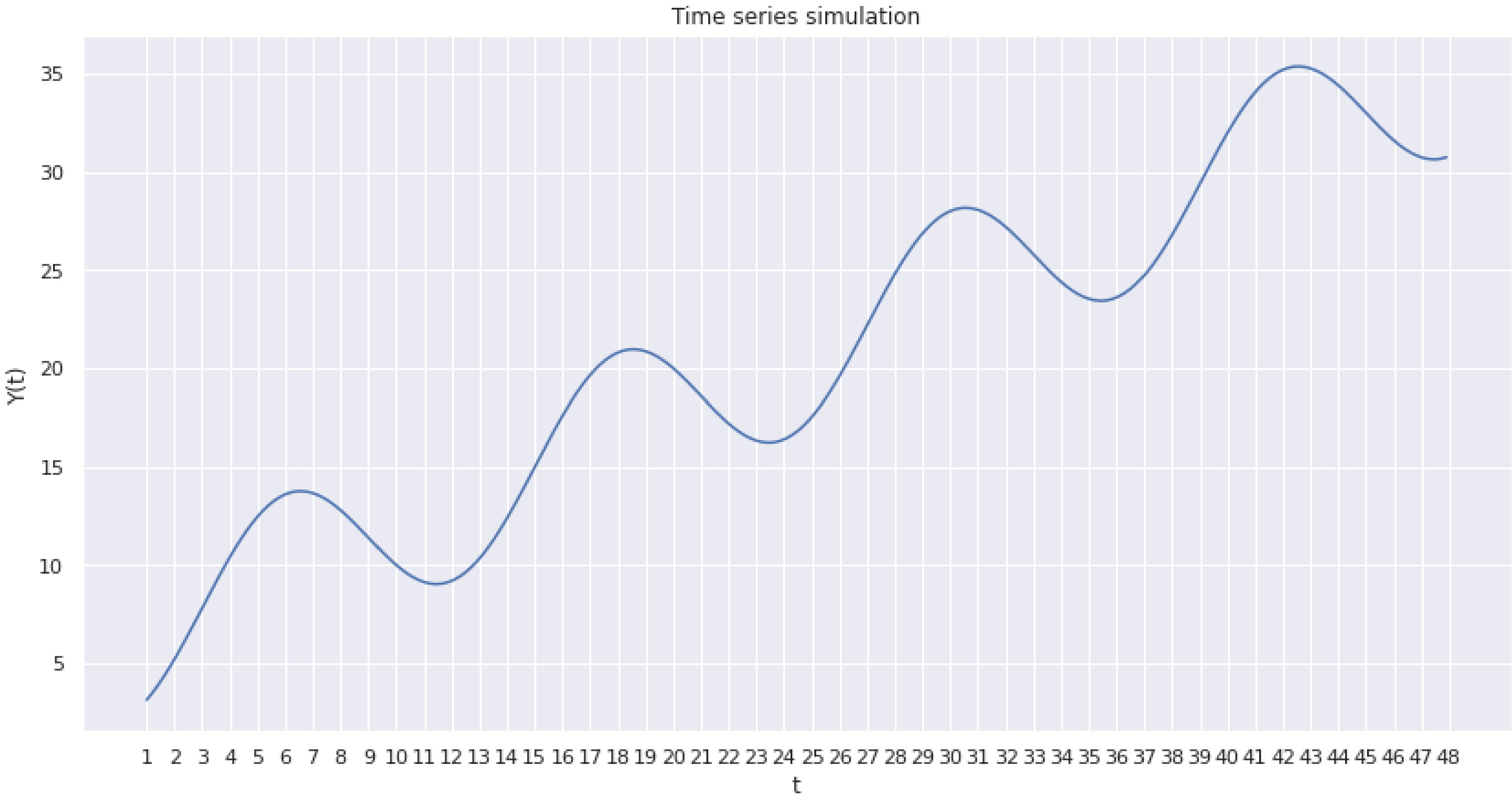
```
t = np.arange(1, 48, 0.1)                                # t domain of length 48
y = 4 * np.sin((np.pi/6) * t - np.pi/2) + 0.6 * t + 6    # function that represents the plot
```

Plot the data (Final simulation)

```
plt.xlabel("t")
plt.ylabel("Y(t)")
```

```
plt.title('Time series simulation')
plt.xticks(np.arange(min(t), max(t)+1, 1.0))
sb.lineplot(x = t, y = y)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cbdb7ec50>
```



Diagnostic results

- The model is now adequate enough to represent the sketch with a high degree of similarity

Conclusion + Use the model for forecasting or control

The function that can be used to simulate given sketch is

```
y = 4 * np.sin((np.pi/6) * t - np.pi/2) + 0.6 * t + 6
```

This function can be used to simulate the sketch provided.

Discussion and Remarks

---

1. Regarding the accuracy of the plot

While the plot matches the structure shown in the sketch, there are some imperfections. For instance, the beginning of the plot at t = 1, the plot didn't start at exact the trough due to the effect of the linear function `0.6 * t + 6` added to the sine function. This can be optimized by changing the coefficient, of the linear function, albeit not to an absolute accuracy. Regardless, I believe this plot still manages to a time series plot that is very similar to the sketch.

2. Regarding the plot

Thanks to the `xticks` function of matplotlib, with `plt.xticks(np.arange(min(t), max(t)+1, 1.0))`, the plot can dislpay a more detailed labels that makes the plot easier to read.

Applying the following seaborn plot settings

```
sb.set(rc={'figure.figsize':(14,7)})
sb.set_style({'axes.grid' : True})
```

allows the plot to display grid lines and make the whole plot larger so it is easier to read.

3. Regarding the scientific method and the generic scheme of model building

By using the scientific method, I am able to systematically approach this problem by first defining the problem, and understanding it through observation. Next, hypothesis is formulated and a prediction is made. Along ths process, the elements of the generic scheme of model building is applied so that I can figure out what mathematical functions to utilize in order to produce a plot that matches the sketch. By testing the hyposis and the selected model through plotting, I am able to identify the shortcomings of my parameters and can adjust them accordingly in the next iteration to better match the provided sketch.

2. (10%) Simulate a time series of length 48 following the settings (below), where Φ is selected from a uniform distribution on the interval [0,1].

$$Y_t = \cos\left[2\pi\left(\frac{t}{12} + \Phi\right)\right] \text{ for } t = 0, 1, 2, \ldots, 47,$$

Begin by prepareing the uniformly distributed value to be used as Φ.

Obtain a number that's selected from the uniform distribution between the closed interval [0,1]

```
uniformDistributionNumber = np.random.uniform(low=0.0, high=1.0)
print("The value of Φ in this case is %s", uniformDistributionNumber)
```

```
The value of Φ in this case is %s 0.353674926985043
```

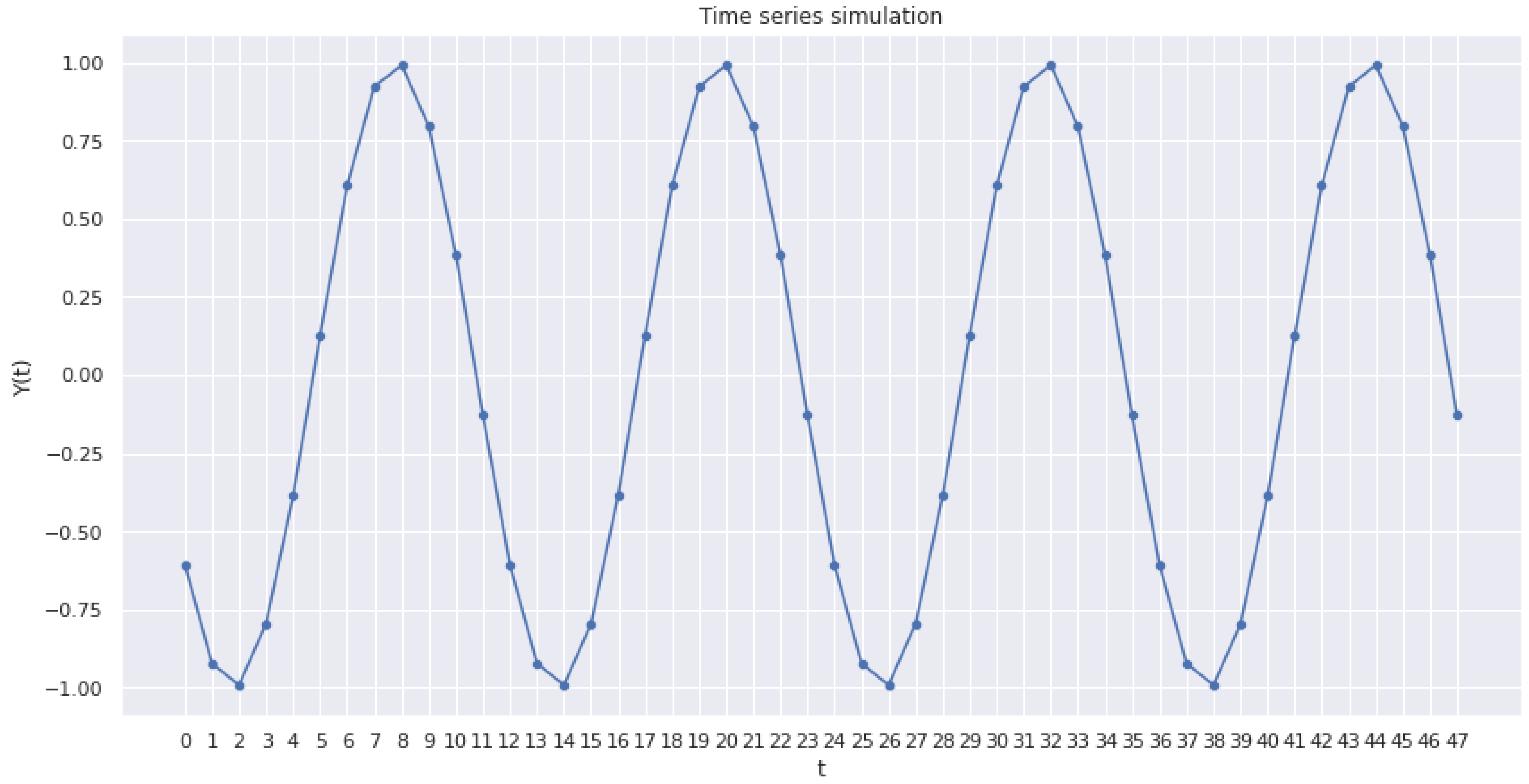
Create a function that simulates the time series data

```
t = np.arange(0, 48, 1)                                                        # t domain of length 48
y = [np.cos(2 * np.pi * (t/12 + uniformDistributionNumber)) for t in range (0,48)]  # function that represents the plot
```

Plot the time series simulation

```
plt.xlabel("t")
plt.ylabel("Y(t)")
plt.title('Time series simulation')
plt.xticks(np.arange(min(t), max(t)+1, 1.0))
plt.plot(t,y)
sb.scatterplot(x = t, y = y)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cbd9c5510>
```



Conclusion

The function `y = np.cos(2 * np.pi * (t/12 + uniformDistributionNumber))` for t=0,1,2,...47 can be used to simulate given sketch as a sinusoidal function with a phase shift of Φ

Discussion and Remarks

1. Regarding the phase

The phase shift Φ selected from the uniform distribution is responsible for slightly moving the time series plot result horizontally. As a result, the first plot did not begin at Y(t) = t as a regular cosine function.

Although, since the uniform distribution is limited within the closed range of [0,1], the change is rather small and the whole time series plot still maintains the features of a cosine function structure.

2. Regarding the phase shift Φ

The current problem determines the value of Φ as a constant value selected from a uniform distribtion. This is the reason why the value of Φ only affect the plot by shifting the entire plot's phase.

Suppose the Φ is instead defined in the function as Φ(t), a new value of Φ would be selected for each t. This will result the plot to be phase shifted at every t, and cause the plot to look "jagged" albeit still retaining the overall structure of a cosine function.

3. Regarding the plot

`plt.plot(t,y)` is used so that the line plot is shown and is easier to see the structure.

`sb.scatterplot(t, y)` is used so that the values can be easily seen in the plot as points.