Sarvadnya Deshpande PES1UG22AM902

Assignment for Research and Development / AI

In [1]:
```
import pandas as pd
import numpy as np
from math import atan, degrees, radians
```

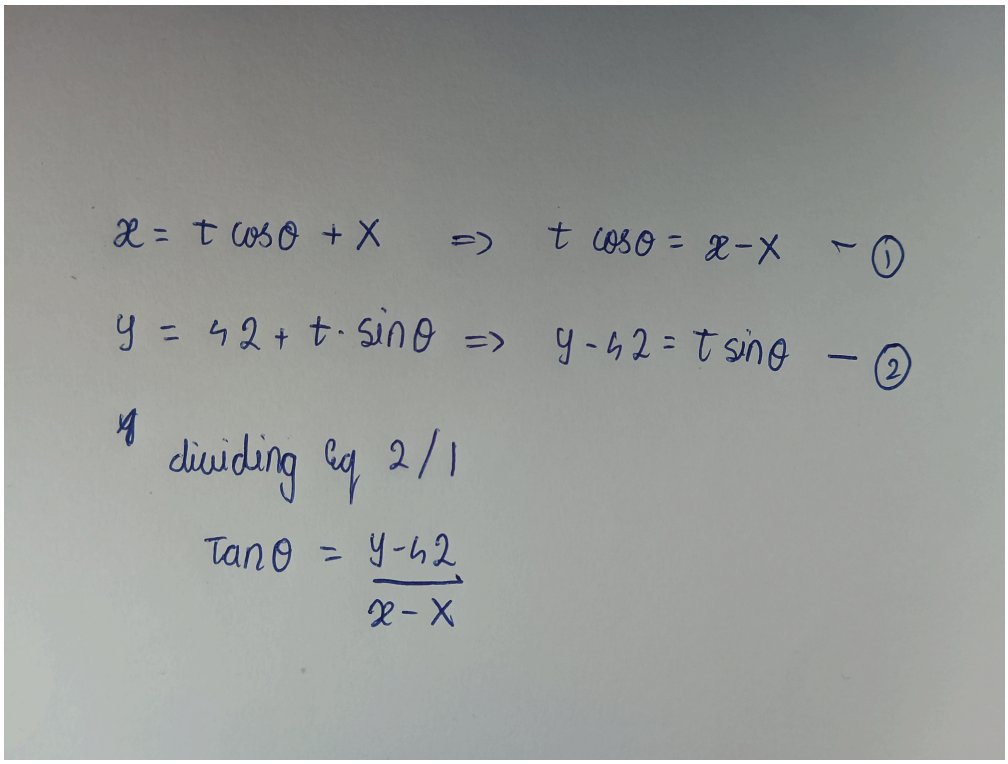
In [2]:
```
data = pd.read_csv("xy_data.csv")
x = data['x'].values
y = data['y'].values
```

The given equations have a linear term and there is also a non linear term to the equation.

Ignoring the non linear term will help us to determine the general direction of our curve which is given by the slope of the lines as below,

the equations become:

- x = t. cos(theta) + X
- y = 42 + t.sin(theta)

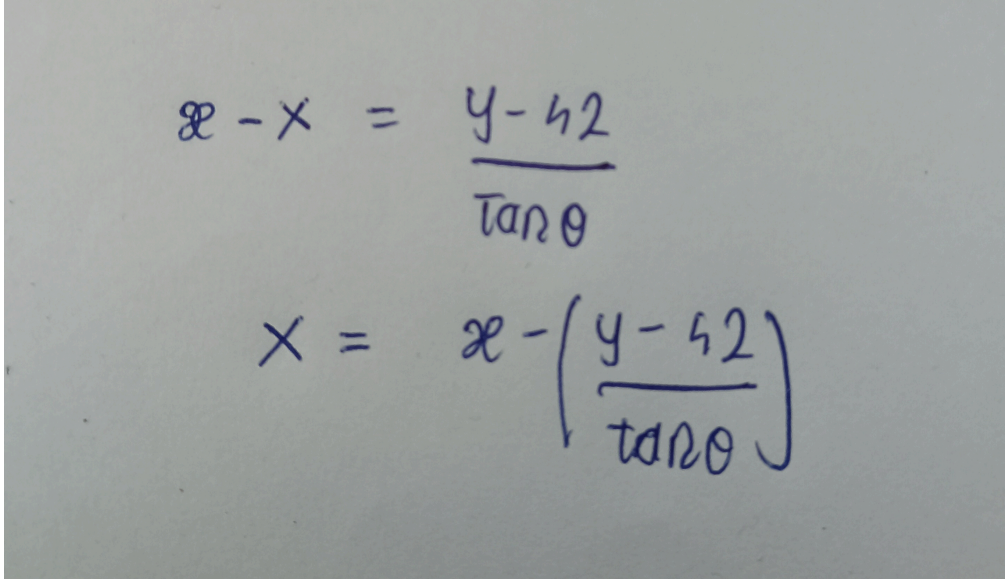


Hence by finding the line that best fits the data points given we can determine the slope of this linear line. using the concept of linear regression

In [3]:
```
m, c = np.polyfit(x, y, 1)
theta = degrees(atan(m))
slope = m
print("tan theta is:",slope)
print("estimated theta: ", theta,"degree")
```

```
tan theta is: 0.5261566552753717
estimated theta:  27.75140022784558 degree
```

Now that we have an estimated value of theta, we can rearrange the above equation to get a relation for X.



since X is a constant, for all the given points of (x,y) in the csv, the value of X should remain nearly same. so we can use the tan theta as estimated above and use the data points given to find the median value of X.

I considered median because with median i can take into account the changes made by the non linear term as well which was ignored initially.

In [4]:
```
eqn = x - ((y-42)/slope)
X = np.median(eqn)
print("estimated value of X:", X)
```

```
estimated value of X: 53.548084439225
```

To Estimate M,

since we have an estimate for theta and X, we can use equation 1 from above to estimate the t value for all x data points.

t = (x-X)/cos(theta)

then we can use equation 2 , y expected = t.sin(theta) - 42 to get what the y value would be for the corresponding x value.

subtracting this y value from the actual y datapoint will give the estimate for the non linear term.

In [5]:
```
rad_angle = radians(theta)
print(rad_angle)
t = (x-X)/np.cos(rad_angle)

y_expected = 42 + (t*np.sin(rad_angle))

non_linear = (y - y_expected)

print("mean of the difference:",non_linear.mean())
print(f"Oscillation range: [{non_linear.min()},{non_linear.max()}]")
```

```
0.4843533060146099
mean of the difference: 0.391558085438451
Oscillation range: [-4.5835037154942455,6.007520341960202]
```

the non linear term is : exp(M * |t|) * sin(.3 t) And its similar to the amplitude modulated wave equation [1]

which has the general form A(t) * sin(ωt) A represents the amplitude and sin is the wave.

we need to find M which controls the amplitude.

RMS amplitude = A/root(2) for a sine wave [2] and the standard deviation in a small region will give us the amplitude of the wave in that region. log(std) = M|t| + c

hence we can use regression to find slope.

To extract the envelope from the amplitude-modulated oscillation term, I employed a sliding window RMS method, a standard technique in digital signal processing for envelope estimation[3]

The data is divided into time windows, and the RMS amplitude is computed in each. These RMS values trace the signal's envelope, since the RMS of a modulated sinusoid reflects its amplitude. Fitting an exponential curve to these points yields the growth rate M. [4]

In [6]:
```
n_bins = 10
sorted = np.argsort(t)
t_sorted = t[sorted]

non_lin_sorted = non_linear[sorted]
bin_size = len(t_sorted) // n_bins
amplitudes = []
t_centers = []

for i in range(n_bins):
    start = i * bin_size
    end = (i + 1) * bin_size if i < n_bins - 1 else len(t_sorted)
    amplitudes.append(np.std(non_lin_sorted[start:end]))
    t_centers.append(np.mean(t_sorted[start:end]))

M, log_A = np.polyfit(t_centers, np.log(amplitudes), 1)
print(f"Estimated M: {M:.6f}")
```

```
Estimated M: 0.022099
```

Sources:

- [1] https://en.wikipedia.org/wiki/Amplitude_modulation
- [2] https://en.wikipedia.org/wiki/Root_mean_square
- [3] https://en.wikipedia.org/wiki/Envelope_(waves)
- [4] https://www.mathworks.com/help/signal/ug/envelope-extraction-using-the-analytic-signal.html

In [ ]:
```
```