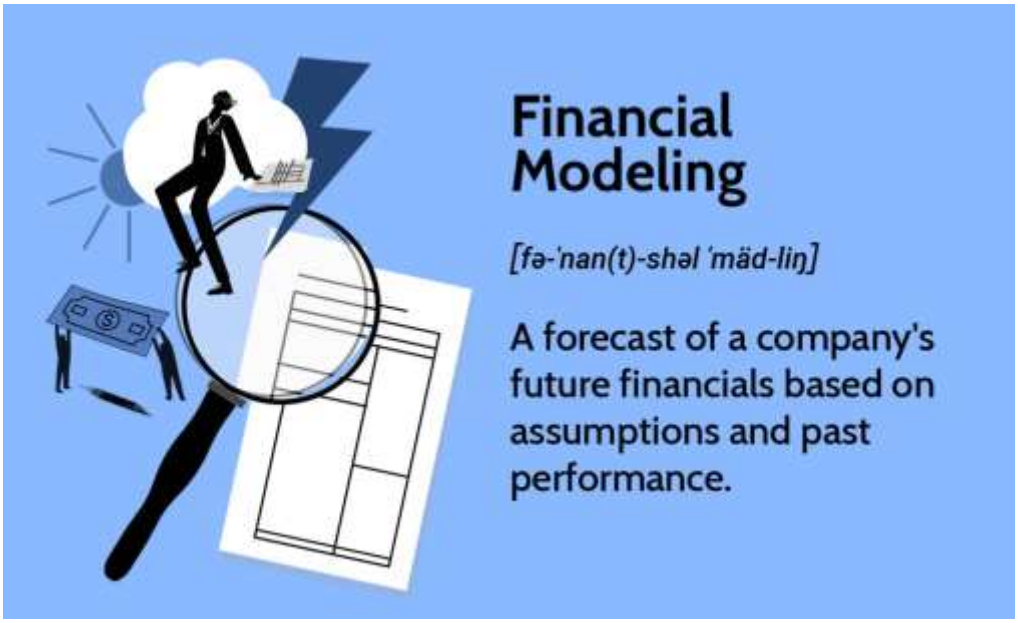
SECANT METHOD



One application of the Secant Method in computer science is in financial modeling for calculating the internal rate of return (IRR) on investments. Investors need to determine the IRR to assess the profitability of projects, but the calculations can be complex. The Secant Method simplifies this by using two previous estimates to quickly find the rate at which the investment breaks even. While it may be a minor part of software development, implementing this algorithm can be very handy for financial analysts, allowing them to make informed decisions and maximize returns efficiently.

The word secant is derived from the Latin word secant, which means to cut. The secant method uses a secant line, a line joining two points that cut the curve, to approximate a root.

MATHEMATICAL REPRESENTATION

Newton's method is very good and fast but its major weakness is the calculation of derivative in

$$p_n = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})}, \quad \text{for } n \geq 1.$$

To avoid $f'(p_{n-1})$ in Newton's method we can replace it with an approximation

$$f'(p_{n-1}) = \lim_{x \to p_{n-1}} \frac{f(x) - f(p_{n-1})}{x - p_{n-1}}$$

If p_{n-2} is close to p_{n-1}, then

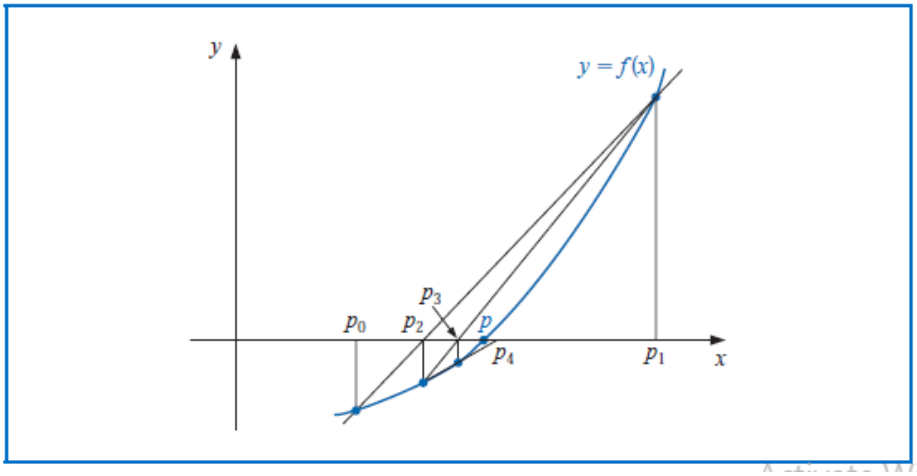
$$f'(p_{n-1}) \approx \frac{f(p_{n-1}) - f(p_{n-2})}{p_{n-1} - p_{n-2}}$$

Putting this approximation in Newton's Formula gives

$$p_n = p_{n-1} - \frac{f(p_{n-1})(p_{n-1} - p_{n-2})}{f(p_{n-1}) - f(p_{n-2})}$$

This technique is called the Secant method.

GEOMETRIC REPRESENTATION



Example

Use Secant method to find solution accurate to within 10^{-5} for $f(x) = x^3 - 2x^2 - 5$ in $[1, 4]$.

Solution: For Secant method, let us take $p_0 = 2$ and $p_1 = 3$, from [1, 4].

Formula for Secant method is

$$p_n = p_{n-1} - \frac{f(p_{n-1})(p_{n-1} - p_{n-2})}{f(p_{n-1}) - f(p_{n-2})}$$

* For calculator take $p_0 = x$ and $p_1 = y$ and write function as

$$y - \frac{(y^3 - 2y^2 - 5)(y - x)}{((y^3 - 2y^2 - 5) - (x^3 - 2x^2 - 5))}$$

and for p_2 $take$ $x = 2$ and $y = 3$.

For p_3 $take$ $x = 3$ and $y = 2.55556$ and so on.

So for n=2, we have

$$p_2 = p_1 - \frac{f(p_1)(p_1 - p_0)}{f(p_1) - f(p_0)} = 2.55556$$

$$p_3 = p_2 - \frac{f(p_2)(p_2 - p_1)}{f(p_2) - f(p_1)} = 2.66905$$

$$p_4 = 2.69237$$

$$p_5 = 2.69063$$

$p_6 = 2.69065 = p_7$ is the solution of *f*(*x*).

PROBLEM STATEMENT

As a software developer working on a financial modeling application, you need to implement a feature that calculates the break-even point for various investment projects. To find the break-even point, you've modeled it using the equation

$$2x\cos(2x) - (x - 2)^2 = 0$$

and need to solve for x within the range of $2 \leq x \leq 3$. To ensure accurate results, you decide to apply the Secant Method to find a solution that is accurate to within 10^{-3}. This implementation will allow users to quickly assess the profitability of their investments, making your software a valuable tool for financial decision-making.

PYTHON CODE, SOLUTION, AND RESULT

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

def f(x):
    return 2 * x * np.cos(2 * x) - (x - 2) ** 2

def secant_method(f, x0, x1, tol=1e-3, max_iter=100):
    for _ in range(max_iter):
        f_x0 = f(x0)
        f_x1 = f(x1)
        if f_x1 - f_x0 == 0:  # Prevent division by zero
            break
        # Secant method formula
        x2 = x1 - f_x1 * (x1 - x0) / (f_x1 - f_x0)

        # Check for convergence
        if abs(x2 - x1) < tol:
            return x2

        # Update points for the next iteration
        x0, x1 = x1, x2

    return None  # If the method fails to converge
```

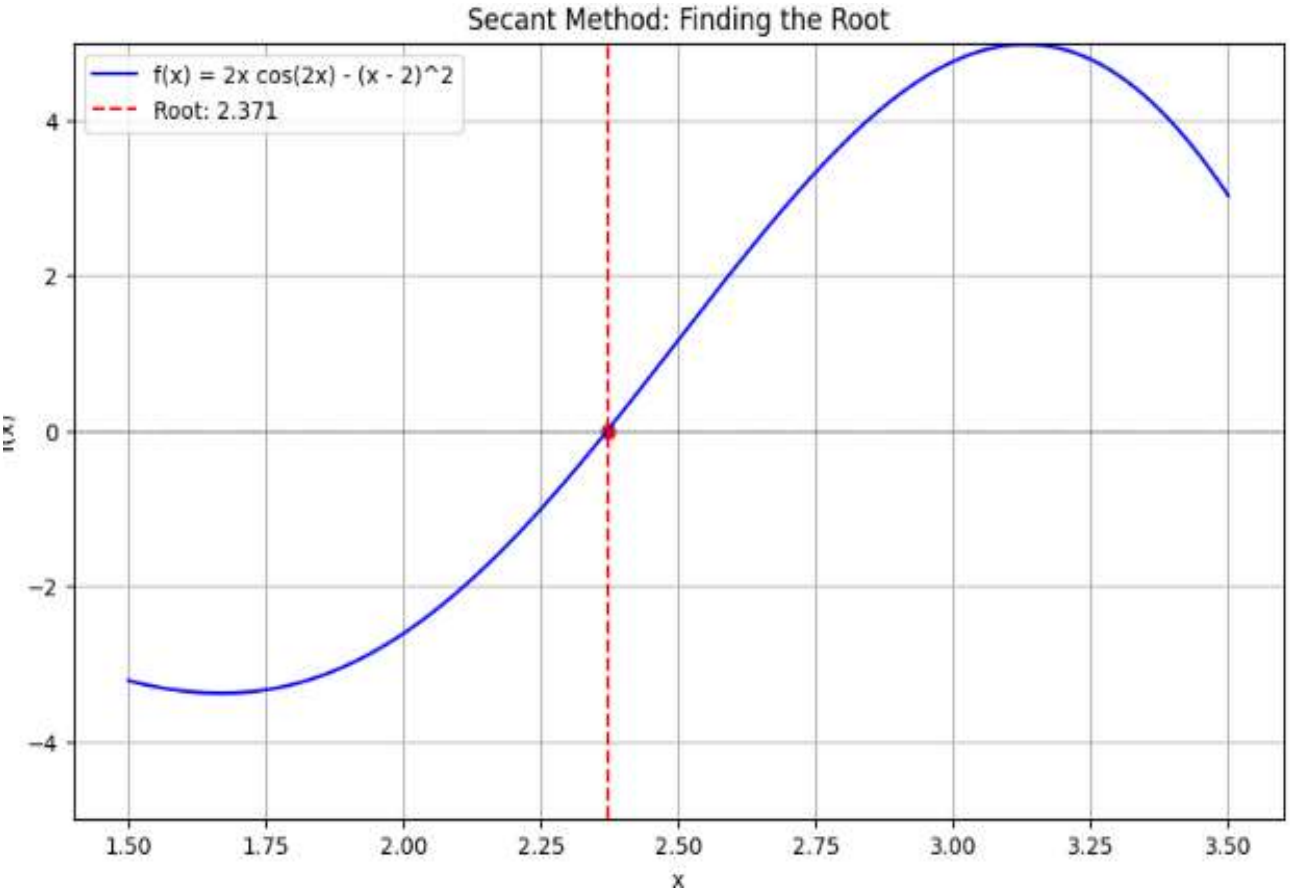
```python
# Initial guesses
x0 = 2.0
x1 = 3.0

# Finding the root using the Secant Method
root = secant_method(f, x0, x1)

# Display the result
print(f"Root found: {root}")

# Plotting the function and the root
x_values = np.linspace(1.5, 3.5, 400)
y_values = f(x_values)

plt.figure(figsize=(10, 6))
plt.plot(x_values, y_values, label='f(x) = 2x cos(2x) - (x - 2)^2', color='blue')
plt.axhline(0, color='black', lw=0.5, ls='--')
plt.axvline(root, color='red', ls='--', label=f'Root: {root:.3f}')
plt.scatter(root, f(root), color='red')  # Mark the root on the plot
plt.title('Secant Method: Finding the Root')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.legend()
plt.grid()
plt.ylim(-5, 5)  # Set limits for better visibility
plt.show()
```



ADVANTAGE/DISADVANTAGE

The convergence of the Secant method is much faster than functional iteration but slightly slower than Newton's method. This is generally the case. Newton's method or the Secant method is often used to refine an answer obtained by another technique, such as the Bisection method, since these methods require good first approximations but generally give rapid convergence.

PRACTICE

Q1. The accumulated value of a savings account based on regular periodic payments can be determined from the *annuity due equation*,

$$A = \frac{P}{i}[(1+i)^n - 1].$$

In this equation, *A* is the amount in the account, *P* is the amount regularly deposited, and *i* is the rate of interest per period for the *n* deposit periods. An engineer would like to have a savings account valued at $750,000 upon retirement in 20 years and can afford to put $1500 per month toward this goal. What is the minimal interest rate at which this amount can be invested, assuming that the interest is compounded monthly?

Q2. Use Secant method to find solution of $x^3 + 3x^2 - 1 = 0$ accurate to within 10^{-5} for in [-3, -2].