Secant Method

The secant method is very similar to the bisection method except instead of dividing each interval by choosing the midpoint the secant method divides each interval by the secant line connecting the endpoints. The secant method always converges to a root of $f(x) = 0$ provided that $f(x)$ is continuous on $[a, b]$ and $f(a)f(b) < 0$.

Secant Line Formula

Let $f(x)$ be a continuous function on a closed interval $[a, b]$ such that $f(a)f(b) < 0$. A solution of the equation $f(x) = 0$ for $x \in [a, b]$ is guaranteed by the Intermediate Value Theorem. Consider the line connecting the endpoint values $(a, f(a))$ and $(b, f(b))$. The line connecting these two points is called the secant line and is given by the formula

$$y = \frac{f(b) - f(a)}{b - a}(x - a) + f(a)$$

The point where the secant line crosses the x-axis is

$$0 = \frac{f(b) - f(a)}{b - a}(x - a) + f(a)$$

which we solve for x

$$x = a - f(a)\frac{b - a}{f(b) - f(a)}$$

Algorithm

The secant method procedure is almost identical to the bisection method. The only difference it how we divide each subinterval.

1. Choose a starting interval $[a_0, b_0]$ such that $f(a_0)f(b_0) < 0$.
2. Compute $f(x_0)$ where x_0 is given by the secant line

$$x_0 = a_0 - f(a_0)\frac{b_0 - a_0}{f(b_0) - f(a_0)}$$

3. Determine the next subinterval $[a_1, b_1]$:
    a. If $f(a_0)f(x_0) < 0$, then let $[a_1, b_1]$ be the next interval with $a_1 = a_0$ and $b_1 = x_0$.
    b. If $f(b_0)f(x_0) < 0$, then let $[a_1, b_1]$ be the next interval with $a_1 = x_0$ and $b_1 = b_0$.
4. Repeat (2) and (3) until the interval $[a_N, b_N]$ reaches some predetermined length.
5. Return the value x_N, the x-intercept of the Nth subinterval.

A solution of the equation $f(x) = 0$ in the interval $[a, b]$ is guaranteed by the Intermediate Value Theorem provided $f(x)$ is continuous on $[a, b]$ and $f(a)f(b) < 0$. In other words, the function changes sign over the interval and therefore must equal 0 at some point in the interval $[a, b]$.

Implementation

Write a function called `secant` which takes 4 input parameters `f`, `a`, `b` and `N` and returns the approximation of a solution of $f(x) = 0$ given by N iterations of the secant method. If $f(a_n)f(b_n) \geq 0$ at any point in the iteration (caused either by a bad initial interval or rounding error in computations), then print `"Secant method fails."` and return `None`.

```
def secant(f,a,b,N):
    '''Approximate solution of f(x)=0 on interval [a,b] by the secant method.

    Parameters
    ----------
    f : function
        The function for which we are trying to approximate a solution f(x)=0.
    a,b : numbers
        The interval in which to search for a solution. The function returns
        None if f(a)*f(b) >= 0 since a solution is not guaranteed.
    N : (positive) integer
        The number of iterations to implement.

    Returns
    -------
    m_N : number
        The x intercept of the secant line on the the Nth interval
            m_n = a_n - f(a_n)*(b_n - a_n)/(f(b_n) - f(a_n))
        The initial interval [a_0,b_0] is given by [a,b]. If f(m_n) == 0
        for some intercept m_n then the function returns this solution.
        If all signs of values f(a_n), f(b_n) and f(m_n) are the same at any
        iterations, the secant method fails and return None.

    Examples
    --------
    >>> f = lambda x: x**2 - x - 1
    >>> secant(f,1,2,5)
    1.6180257510729614
    '''
    if f(a)*f(b) >= 0:
        print("Secant method fails.")
        return None
    a_n = a
    b_n = b
    for n in range(1,N+1):
        m_n = a_n - f(a_n)*(b_n - a_n)/(f(b_n) - f(a_n))
        f_m_n = f(m_n)
        if f(a_n)*f_m_n < 0:
            a_n = a_n
            b_n = m_n
        elif f(b_n)*f_m_n < 0:
            a_n = m_n
            b_n = b_n
        elif f_m_n == 0:
            print("Found exact solution.")
            return m_n
        else:
            print("Secant method fails.")
            return None
    return a_n - f(a_n)*(b_n - a_n)/(f(b_n) - f(a_n))
```

Examples

Supergolden Ratio

Let's test our function with input values for which we know the correct output. Let's find an approximation of the supergolden ratio: the only real root of the polynomial $p(x) = x^3 - x^2 - 1$.

```
p = lambda x: x**3 - x**2 - 1
print(p(1))
print(p(2))
```

```
-1
3
```

Since the polynomial changes sign in the interval $[1, 2]$, we can apply the secant method with this as the starting interval:

```
approx = secant(p,1,2,20)
print(approx)
```

```
1.4655712311394433
```

The exact value of the supergolden ratio is

$$\frac{1 + \sqrt[3]{\frac{29+3\sqrt{93}}{2}} + \sqrt[3]{\frac{29-3\sqrt{93}}{2}}}{3}$$

```
supergolden = (1 + ((29 + 3*93**0.5)/2)**(1/3) + ((29 - 3*93**0.5)/2)**(1/3))/3
print(supergolden)
```

```
1.4655712318767682
```

Let's compare our approximation with the exact solution:

```
error = abs(supergolden - approx)
print(error)
```

```
7.373248678277378e-10
```

Exercises

Under construction