Bisection

March 14, 2019

In [2]:
```
def bisection(f,a,b,N):
    '''Approximate solution of f(x)=0 on interval [a,b] by the bisection 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
    -------
    x_N : number
        The midpoint of the Nth interval computed by the bisection method. The
        initial interval [a_0,b_0] is given by [a,b]. If f(m_n) == 0 for some
        midpoint m_n = (a_n + b_n)/2, 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
        iteration, the bisection method fails and return None.

    Examples
    --------
    >>> f = lambda x: x**2 - x - 1
    >>> bisection(f,1,2,25)
    1.618033990263939
    >>> f = lambda x: (2*x - 1)*(x - 3)
    >>> bisection(f,0,1,10)
    0.5
    '''
    if f(a)*f(b) >= 0:
        print("Bisection method fails.")
        return None
    a_n = a
    b_n = b
    for n in range(1,N+1):
        m_n = (a_n + b_n)/2
```

```
            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("Bisection method fails.")
                return None
        return (a_n + b_n)/2
```

In [3]:
```
f = lambda x: x**2 - x - 1
approx_phi = bisection(f,1,2,25)
print(approx_phi)
```

```
1.618033990263939
```

In [4]:
```
error_bound = 2**(-26)
print(error_bound)
```

```
1.4901161193847656e-08
```

In [6]:
```
abs( (1 + 5**0.5)/2 - approx_phi) < error_bound
```

Out[6]: True