Lecture 4: Roots of High-Degree Equations

Lecture Notes - AMS 326

Outline

Overview

- High-degree equations are equations containing polynomials or radicals and/or transcendental (trigonometric and logarithmic) functions.
- The simplest example of high-degree equations is the quadratic equation:

$$ax^2 + bx + c = 0$$

- Its solution can be found by

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

- But in case of equations of higher degrees (power) or when terms of transcendental functions exist, numerical methods become the only way to obtain their roots.

Simple Iterations Method

Suppose we want to find the roots of the equation $f(x) = 0$.

1. Re-arrange the equation so that the variable is put on the left side. For example, $x^2 - 2x + 1 = 0$ can be written as $x = 0.5x^2 + 0.5$ or $x = \sqrt{2x - 1}$.
2. Assume an initial value to start the first iteration.
3. Substitute the initial value in the right side of the equation and calculate a new value for the variable

$$x_{new} = 0.5x^2 + 0.5$$

4. If the new value of the variable *is not equal* to the previous value, consider the new one as the value of the variable.
5. Repeat steps 3 and 4 until the new value *is equal* to the old value of the variable.
6. if the new value does not approach the old value (the difference between x and x_{new} increases at each iteration), stop. Try another initial value of another rearrangement of the given equation.

Simple Iterations Method (for Loop)

Example

Find the roots of the equation: $2x^2 - 5x + 3 = 0$.
(Analytical solutions: $x = 1.5$ and $x = 1$)

1. Rearrange the equation in the form $x = \dfrac{2x^2 + 3}{5}$ or $x = \sqrt{\dfrac{5x - 3}{2}}$
2. Let us try to write a code in Python:

```
1 x = 0 # the initial guess
2 for iteration in range(100): # the iteration starts from 0 to 99
3     xnew = (2*x**2 + 3)/5
4     if xnew == x:
5         break
6     x = xnew
7 print(iteration+1, xnew)
```

```
100 0.9999999999515905
```

The Degree of Accuracy

- For example, let's try the following commands:
  ```
  > > > s = 0.1 + 0.2
  > > > print(s)
  ```
  It prints out `0.30000000000000004` which looks weird.
- To satisfy the condition of *exact equality* between numerical values is extremely difficult if not impossible. So, the normal practice in numerical methods is to specify the degree of accuracy required in the solution. It represents the acceptable absolute maximum difference between values of the variable of the equation resulted from the two successive iterations, in other words, the difference between the old and new values.

The Degree of Accuracy

- Let's apply this one to our code.

```
1 x = 0 # the initial guess
2 for iteration in range(1,101): # the iteration starts from 0 to 99
3     xnew = (2*x**2 + 3)/5
4     if abs(xnew - x) < 0.000001:
5         break
6     x = xnew
7 print('The root: %0.5f' % xnew)
8 print('The number of iteration : %d' % iteration)
```

```
The root: 1.00000
The number of iteration : 50
```

- We use `%f` a place holder of the float numbers and use 0.5. This means show me the number to the fifth decimal point.
- The notation `%d` is placeholder for integer iteration.

The Degree of Accuracy

- Let us try another solution by setting a different initial guess $x = 2$.
- This leads to `OverflowError: (34, 'Result too large')`. If we print out x on the loop. we can see that its value at the sixth iteration is too large.
- Let us try couple different initial guesses: $x = 1.3$ and $x = 1.6$.
- Let us try another form:

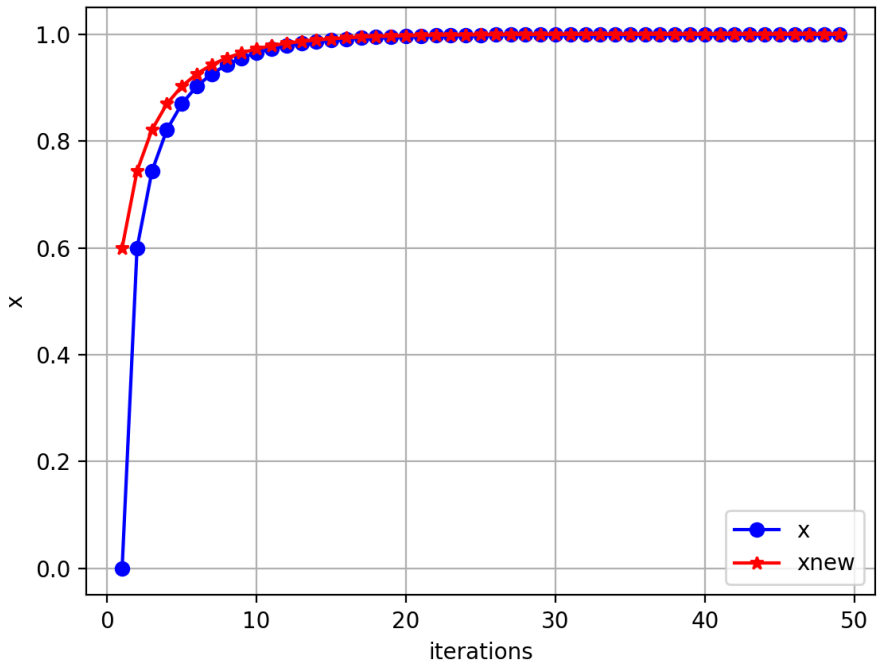
```
1 x = 1.6 # the initial guess
2 for iteration in range(1, 101):  # 100 iterations
3     xnew = sqrt((5*x-3)/2)
4     print(x)
5     if abs(xnew - x) < 0.000001:
6         break
7     x = xnew
8 print('The root: %0.5f' % xnew)
9 print('The number of iterations: %d' % iteration)
```

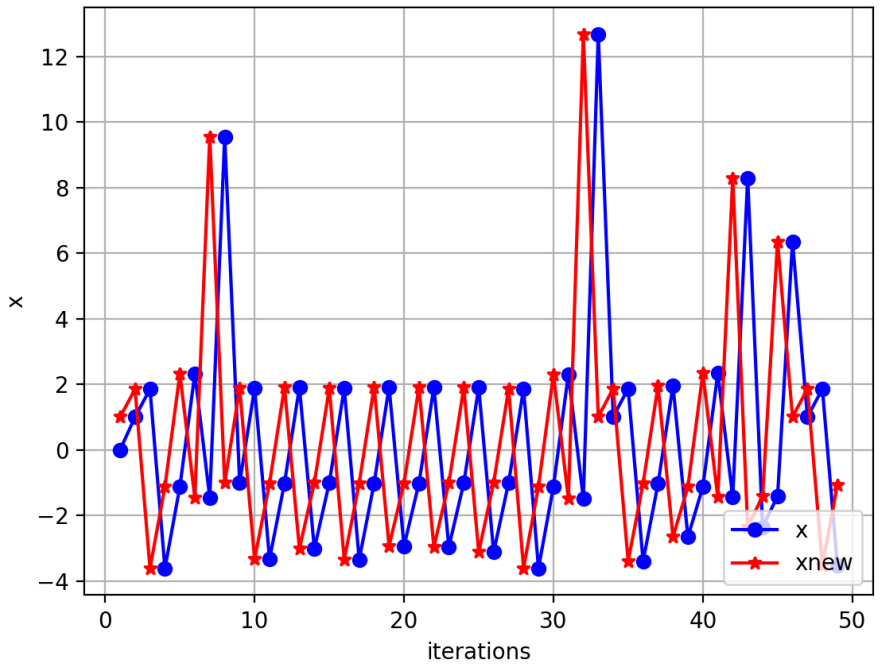
Simple Iterations Method (white Loop)

- The white loop looks simple but it is not as direct as the for loop.
- The code is given here

```
1 x = 5 #arbitrary value
2 xnew = 0
3 iteration = 0
4 while abs(xnew - x) >= 0.000001:
5     iteration +=1
6     x = xnew
7     xnew = (2*x**2 + 3)/5
8 print('The root: %f' % xnew)
9 print('The number of iterations : %d' % iteration)
10
```

Convergence and Divergence





The code can be found here.

Your turn: Solve $x^3 - 5x^2 + 6x = 0$ numerically.