Section 1

MANE 3351

Subsection 1

Lecture 11

Classroom Management

Agenda

- Test 1 not graded; some people still need to take
- Lab 5 today; very important
- Newton's Method

Subsection 2

Resources

Handouts

- Lecture 11 Slides
- Lecture 11 Marked Slides

Newton's Method

Both the bisection and False Position methods were bracketing approaches to find roots that required two starting points that "bracketed" the root. Today's topic, Newton's method or Newton-Raphson method, require only one starting point. Newton's method also requires the knowledge of the derivative.

Geometric Inspiration

Brin (2020)[a] demonstrate the geometric inspiration for Newton's method

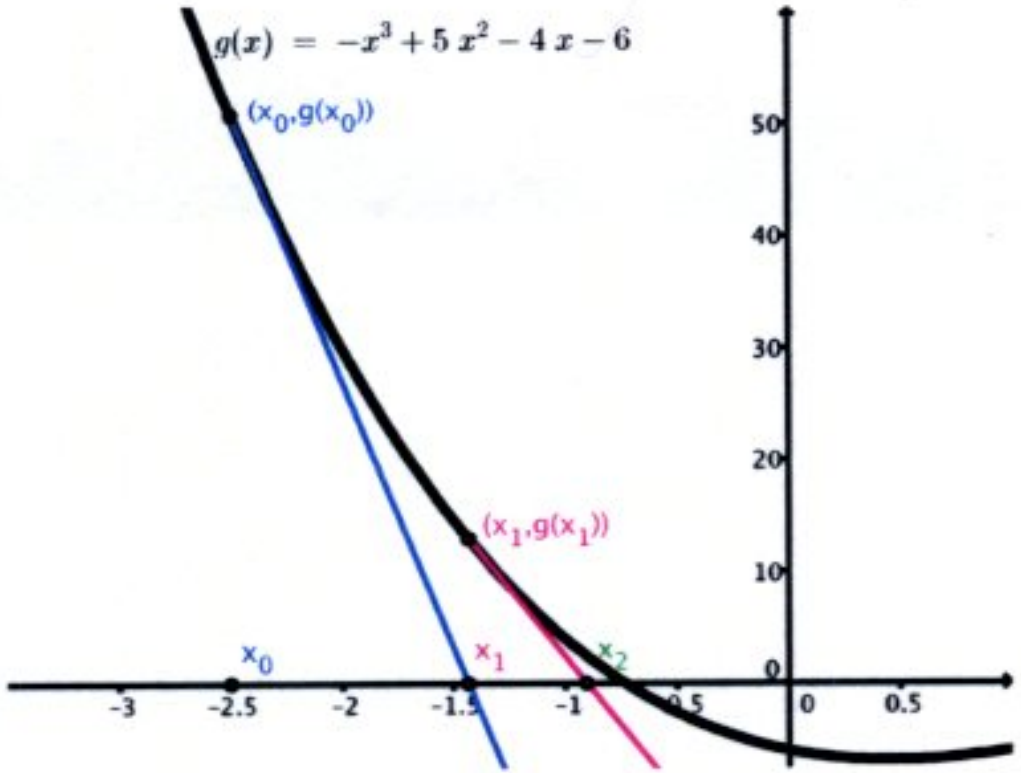

Figure 1: Newton's Method

[a] Brin, L, (2020), *Tea Time Numerical Analysis: Experiences in Mathematics, 3rd edition*

Newton's Method

- The formula is simply

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

New Example Problem

- Consider a new function, $f(x) = e^x + 2^{-x} + 2\cos(x) - 6 = 0$

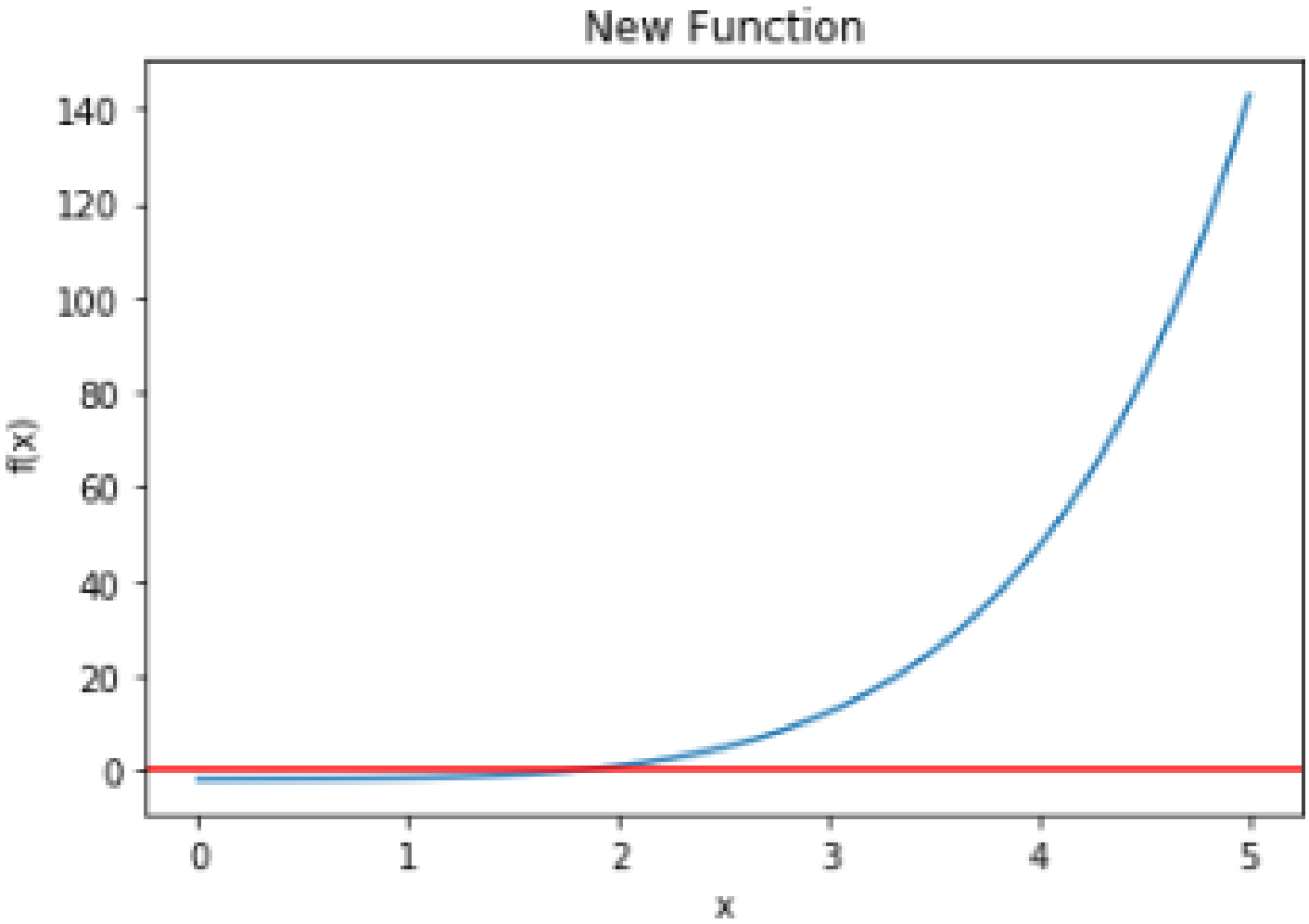

Figure 2: Example Function for Newton's Method

First Derivative

- Newton's Method requires the first derivative:

$$\frac{d}{dx}\left[e^x + 2^{-x} + 2\cos(x)\right] = e^x - 2^{-x}\ln(x) - 2\sin(x)$$

Review of Derivatives

- An excellent table derivatives is found at Table of Derivatives
- A helpful site is Derivative Calculator

Pseudo-code

Brin (2020)[a] provides the following pseudo-code

Newton's Method (pseudo-code)

Unlike Steffensen's method, the denominator appearing in Newton's method is not expected to approach zero as the iterates converge, so generally there is much less trouble with stability of the calculation and no intermediate checks are done before computing one iteration from the previous.

Assumptions: g is twice differentiable. g has a root at $\hat{x}$. x_0 is in a neighborhood $(\hat{x}-\delta, \hat{x}+\delta)$ where the magnitude of $f'(x) = 1 - \frac{g'(x)\cdot g'(x) - g(x)g''(x)}{g'(x)\cdot g'(x)}$ is less than one.

Input: Initial value x_0; function g and its derivative g'; desired accuracy tol; maximum number of iterations N.

Step 1: For $j = 1 \ldots N$ do Steps 2-4:

Step 2: Set $x = x_0 - \frac{g(x_0)}{g'(x_0)}$;

Step 3: If $|x - x_0| \leq tol$ then return x;

Step 4: Set $x_0 = x$;

Step 5: Print "Method failed. Maximum iterations exceeded."

Output: Approximation x near exact fixed point, or message of failure.

Figure 3: Newton's Method pseducode

[a] Brin, L, (2020), *Tea Time Numerical Analysis: Experiences in Mathematics, 3rd edition*

Vectorizing a Function

- All Python functions considered so far have operated on scalars
- Functions can be created to process a vector of values with a single call. This is called a vectorized function
- Consider plotting the new function defined earlier using a vectorized function

Vectorized Function Code

```
import math
import numpy as np
import matplotlib.pyplot as plt
#
def f(x):
    return (math.exp(x)+2**-x+2*math.cos(x)-6)
# convert f(x) in a vectorized function that can be applied to

vec_f=np.vectorize(f)

x=np.linspace(0,5,101)


#print(x)

# y=f(x)
```

Newton Raphson Python Code

```
import math
def f(x):
    return (math.exp(x)+2**(-x)+2*math.cos(x)-6)
def f_prime(x):
    return(math.exp(x)-2.0**(-x)*math.log(2.0)-2*math.sin(x))
N=100
tol=0.0005
x_0=3.5
counter=0
for j in range(N+1):
    counter=counter+1
    x=x_0-f(x_0)/f_prime(x_0)
    print("x={}".format(x))
    if math.fabs(x-x_0)<tol:
        print("the root is {} with value {}, required {} steps
        break
    x_0=x
print("completed")
```

Convergence

- Cheney and Kincaid (2004)[a] study the performance of Newton's methods
- Assumptions
  - f contains two continuous derivatives, f' and f''
  - r is a simple root, $f'(r) \neq 0$
- If r is started sufficiently close to r, **converges quadratically** to r
  - $|r - x_{n+1}| \leq c|r - x_n|^2$
- In other words, x_{n+1} has approximately twice as many correct digits as x_n!

[a] Cheny, W., and Kincaid, D., (2004), *Numerical Mathematics and Computer, 5th edition*

Other Comments

Kiusalaas (2013)[a] provides the following introduction to the Newton-Raphson Method

> *The Newton-Raphson algorithm is the best known method of finding roots for a good reason: It is simple and fast. The only drawback of the methods is that it uses the derivative $f'(x)$ of the function as well as the function $f(x)$ itself. Therefore, the Newton-Raphson method is usable only in problems where $f'(x)$ can be readily computed.*

[a] Kiusalaas, J., (2013), *Numerical Methods in Engineering with Python 3*

Importance of Good Starting Point

Cheney and Kincaid (2004)[a], provide several illustrations of bad starting points and the problems that can occur.

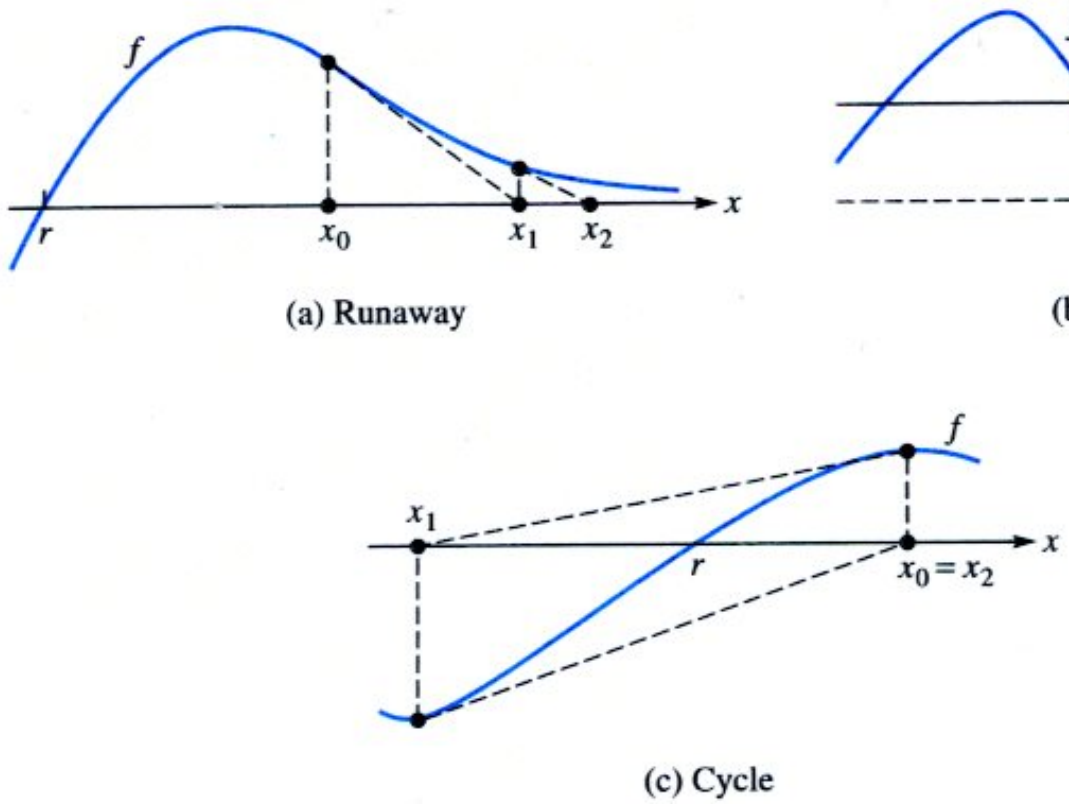

FIGURE 3.4
Failure of Newton's method due to bad starting points

Figure 4: Bad Starting Points