**Logo

Description automatically generated San Francisco Bay University**

**MATH201 - Calculus-I**

**Homework Assignment #5**

**Due day: 11/27/2024**

**Instruction:**

1. **Push the answer sheet to GitHub in word file**
2. **Overdue homework submission could not be accepted.**
3. **Takes academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**
4. **Blood testing** Suppose a blood test for a disease is given to a population of *N* people, where *N* is large. At most, *N* individual blood tests must be done. The following strategy reduces the number of tests. Suppose 100 people are selected from the population and their blood samples are pooled. One test determines whether any of the 100 people test positive. If that test is positive, those 100 people are tested individually, making 101 tests necessary. However, if the pooled sample tests negative, then 100 people have been tested with one test. This procedure is then repeated. Probability theory shows that if the group size is *x* (for example, *x* = 100, as described here), then the average number of blood tests required to test *N* people is *N\**(1-), where *q* is the probability that any one person tests negative. What group size *x* minimizes the average number of tests in the case that *N* = 100 and *q* = 0.95? Assume *x* is a real number between 1 and 150 in Excel or Python program.

The question focuses on **group testing** to minimize the number of tests required for a disease-testing process. In traditional testing:

* If we test each person individually, N tests are required (where N=100).
* To reduce the number of tests, a **group testing strategy** is used:
  1. Combine x samples and test them as a group.
  2. **If the test is negative**, all individuals in the group are disease-free, and only **1 test** is used for the group.
  3. **If the test is positive**, we need to individually test all x people in the group, requiring x+1 tests.
* **Goal**: Find the optimal group size x (a real number) that minimizes the **average number of tests**.

Formula for average number of tests:

T(x)=N⋅(1−)

Where:

* N=100: Total number of people to be tested.
* q=0. 95: Probability that an individual is disease-negative.
* x: Group size (to be optimized).

Understanding the formula T(x) step by step:

1. 1−: Represents the probability that at least **one person** in the group of size x tests positive. This increases with larger x, requiring more individual tests.
2. ​: Represents the efficiency of testing smaller groups. Smaller groups reduce the chance of retesting but increase the number of total groups.
3. Multiply by N: Accounts for the total population size.

The challenge is to balance these components to minimize T(x).

* N=100: Total population size.
* q=0.95: High probability of being disease negative.
* x: A real number (group size) between **1** and **150** (to prevent extremely small or large group sizes).
* Minimize T(x): The average number of tests needed.

For solving the problem:

 Using **Python** to calculate T(x) for different values of x in the range 1 to 150.

 Using **fine-grained steps** (e.g., 10,000 points between 1 and 150) for precise calculations.

 Identifying the value of x that gives the smallest T(x).

 Output:

* The **optimal group size** x.
* The **minimum average number of tests** T(x).

Source Code:

A screenshot of a computer

Description automatically generated

Output:

A black text on a white background

Description automatically generated

When the program runs:

1. **Input**:
   * Constants: N=100, q=0.95.
   * Range of x: From 1 to 150 (generated with np.linspace for precision).
2. **Process**:
   * For each value of xxx, calculate T(x) using the formula.
   * Store the results in a list (T\_values).
   * Find the minimum T(x) and the corresponding x using np.argmin.
3. **Output**:
   * The **optimal x** that minimizes T(x).
   * The **minimum value of T(x)**.

After running the Python code:

1. **Optimal Group Size (x)**:
   * x≈5.02x : A group size of approximately 5 people minimizes the number of tests.
2. **Minimum Average Number of Tests (T(x))**:
   * T(x)≈42. 62: The minimum average number of tests needed for the population.

**Detailed Explanation of the Code**

1. **Define Constants**:
   * N=100: Total population size.
   * q=0. 95: Probability of a person being negative.
2. **Define Formula**:
   * Use a function average\_tests(x) to calculate T(x) for any given x.
3. **Generate x Values**:
   * Create 10,000 equally spaced points between 1.01 and 150 using np.linspace.
   * Avoid x=1 to prevent division by zero in 1/x.
4. **Compute T(x)**:
   * Use a loop to calculate T(x) for each value of x.
   * Store results in a list T\_values.
5. **Find Minimum**:
   * Use np.argmin to find the index of the smallest T(x).
   * Retrieve the corresponding x and T(x) values.
6. **Print Results**:
   * Output the optimal x and the corresponding minimum T(x).

**So , the final answer is:**

**Optimal Group Size(x): 5.02**

**Minimum Average Tests(T(x): 42.62.**

1. **Modified Newton’s method** The function *ƒ* has a root of *multiplicity*2 at *r* if and . In this case, a slight modification of Newton’s method, known as the *modified* (or *accelerated*) Newton’s method, is given by the formula

, for

This modified form generally increases the rate of convergence. Please complete the following questions in Excel or Python program.

1. Verify that 0 is a root of multiplicity 2 of the function

**Answer:**

To verify that x=0 is a root of multiplicity 2 for the function:

we need to satisfy the following conditions for multiplicity 2:

1. f(0)= 0
2. f′(0)=0
3. f′′(0)≠0

Substitute x=0 into f(x):

f(0)=

Since sin(0)=0:

f(0)=

**Conclusion**: f(0)=0, so the first condition is satisfied.

**Calculate f′(x) and f′(0)**

The first derivative of f(x) is:

f′(x)=

=

=[x=0]

Since sin(0)=0 and cos(0)=1:

f′(0)=e^0⋅2−2=2−2=0

f′(0)=0, so the second condition is satisfied.

Calculating f′′(x) and f′′(0)

The second derivative of f(x) is:

f′′(x)=

=

**=**

**=**

f′′(0)=

=

**Conclusion**: f′′(0)=4≠0, so the third condition is satisfied.

Source code:

A screenshot of a computer program

Description automatically generated

Output:

A white background with black text

Description automatically generated

 The Python program confirms that f(0)=0, f′(0)=0, and f′′(0)=4≠0.

 Therefore, x=0 is indeed a root of multiplicity 2 for f(x) = .

1. Apply Newton’s method and the modified Newton’s method using to find the value of in each case. Compare the accuracy of these values of .

Answer:

 Apply **Newton's Method** and **Modified Newton's Method** starting with

 Perform **9 iterations** to compute ​ in each case.

 Compare the accuracy of ​ obtained from both methods with respect to the true root r=0.

**Newton's Method and Modified Newton's Method**

1. **Newton's Method**: The formula is:
2. **Modified Newton's Method**: The formula is:

**Function and Derivative**:

The function given is:

f(x)=

Its derivative is:

f′(x)=

We implemented both methods in Python using the following steps:

1. **Define the function f(x) and its derivative f′(x)**:

f(x)=

f′(x)=

**Newton’s Method**:

* Start with .

Use the formula

 **Modified Newton’s Method**:

* Start with .

Use the formula for 9 iterations.

 **Compare Results**:

* Compute for both methods.
* Compare the errors by calculating the absolute difference between and the true root r=0.

Source code:

A screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated

Output:

A number of numbers and a method

Description automatically generated with medium confidence

**Results from Python**

After running the code, the following results are obtained:

1. **Newton's Method**:
   * x9 using Newton's method: 0.00020483725493164077
   * Error = Error in Newton's method: 0.00020483725493164077
2. **Modified Newton's Method**:
   * x9= -1.5258027692339302×[essentially 000, within numerical precision]
   * Error = 1.5258027692339302

**Analysis and Comparison**

1. **Newton’s Method**:
   * Convergence is slower because it is not optimized for roots with multiplicity 2.
   * After 9 iterations, is close to 000, but the error is larger compared to the modified method.
2. **Modified Newton’s Method**:
   * Converges much faster due to the adjustment for root multiplicity.
   * After 9 iterations, is essentially 0 (within numerical precision), with an error that is orders of magnitude smaller than the standard method.

**c.** Consider the function . Use the modified Newton’s method to find the value of using = 0.15. Compare this value to the value of found

in Newton’s method.

Answer: The given function is:

Using the quotient rule:

f′(x) =

=

=

=

**Iterative Methods**

1. **Modified Newton’s Method**: The formula is:
2. **Standard Newton's Method**: The formula is:

Source code:

A screenshot of a computer program

Description automatically generated A screenshot of a computer program

Description automatically generated

Output:

A screenshot of a computer code

Description automatically generated

 **Accuracy**:

* **Newton’s Method** produces an error of 0.0002679099943247544, which is relatively small but much larger than the Modified Newton's Method.
* **Modified Newton’s Method** achieves an error of 4.019730376466843×, effectively reaching the root with very high precision.

 **Convergence Speed**:

* **Newton’s Method** converges more slowly because it is not optimized for roots with multiplicity.
* **Modified Newton’s Method** adjusts for the root’s multiplicity, allowing it to converge faster.

 **Stability**:

* The safeguard against division by zero was triggered during iterations when f′(x) approached zero, ensuring the program did not crash.

So,

 **Modified Newton’s Method** is significantly more accurate and efficient for roots with multiplicity, making it the preferred choice for this problem.

 **Newton’s Method**, while reliable, requires more iterations to achieve the same level of accuracy.