ASSIGMENT 1 : CS 215

KOLLAPU KEERTHI ISHWARYA - 24B1038

BHUKYA SUSMITHA - 24B1008

## Question 1 :



Fig 1 : f = 30% corruption



Fig 2 : f= 60% corruption

For f = 30% corruption:

• Relative Mean Square Error (median)=8.7379  
• Relative Mean Square Error (mean)=56.8202  
• Relative Mean Square Error (1st quartile)=0.01505

For f = 60% corruption:

• Relative Mean Square Error (median)=8.4899  
• Relative Mean Square Error (mean)=58.1542  
• Relative Mean Square Error (1st quartile)=0.0176

## Conclusion:

First quartile filter produced the best results in terms of relative mean squared error. The error is close to zero for both 30% and 60% corruption in case of first quartile filter because it ignores the big spikes in the graph unlike other filters.

While the arithmetic mean was heavily influenced by these outliers and median was partially affected, the first quartile remained close to clean sine wave.

Instructions to Run:

Copy the code from file question\_1.m and paste into MATLAB and run the code to get output.

# Question 2

Given:  
An array A of numbers,  
  
- n = total number of elements in the given array  
- NewDataValue = added element to the array  
  
Old Mean, Old Std, Old Median are respectively the mean, standard deviation, and median of the array of numbers before adding 'NewDataValue'.  
  
Formulate derivation for the following:

## 1. New Mean

We know that:

Old Mean = (x₁ + x₂ + ... + xₙ) / n , where xᵢ ∈ A

Old Mean = (Σxᵢ) / n → (1)

Now,

New Mean = (x₁ + x₂ + ... + xₙ + NewDataValue) / (n+1)

New Mean = (Σxᵢ + NewDataValue) / (n+1) → (2)

From equation (1):

Σxᵢ = n × (Old Mean)

Substituting into equation (2):

New Mean = [n × (Old Mean) + NewDataValue] / (n+1)

Hence , the function becomes ,

function newMean = UpdateMean (OldMean, NewDataValue, n)

newMean = ((OldMean\*n ) + NewDataValue)/ (n +1) ;

end

# 2.Derivation of New Median

To derive New Median, there are two cases under this based on the number n:

1. n is odd

If n is odd, we know that for a dataset having an odd number of elements, it has a single median (the middle element).

There are two cases under this:

(i) NewDataValue > OldMedian

As now the number of elements = n + 1 [even number] and the added value is greater than the OldMedian, the number of elements on the right of OldMedian is greater than the number of elements on the left by 1.

So, new Median = (A[(n+1)/2] + A[(n+1)/2 + 1] )/2

If the NewDataValue is less than A[((n+1)/2 + 1 )] th element of the old array, then the NewDataValue itself becomes the( (n+1)/2 + 1 ) th element in the NewArray.

Hence, new Median = ( A[(n+1)/2] + NewDataValue ) / 2

(ii) NewDataValue < OldMedian

As now the number of elements = n + 1 [even number] and the added value is less than the OldMedian, the number of elements on the left of OldMedian is greater than the number of elements on the left by 1.

So, new Median = (A[(n+1)/2] + A[(n+1)/2 - 1] )/2

If the NewDataValue is less than A[((n+1)/2 + 1 )] th element of the old array, then the NewDataValue itself becomes the( (n+1)/2 + 1 ) th element in the NewArray.

Hence, new Median = ( A[(n+1)/2] + NewDataValue ) / 2

1. (ii) n is even

If n is even, then the OldMedian is the average of the two middle elements:

OldMedian = ( A[n/2] + A[(n/2) + 1] ) / 2

So, there are three cases in this:

1. If NewDataValue < A[n/2 + 1] and NewDataValue > A[n/2]

As the New Data Value lies between the A[⌈n/2⌉] and A[⌊n/2⌋] elements, and as n+1 is odd:

NewMedian = NewDataValue

Case (ii): New Data Value > A[n/2+1]

As n+1 is odd, and the middle most becomes the median and a New Data Value would be added to the right of the ⌈n/2⌉+1-th term:

NewMedian = A[n/2+ 1 ]

Case(iii) NewDataValue < A[n/2]

As n+1 is odd, and as the New Data Value is less than A[⌊n/2⌋] , NewDataValue would be added to the left of [n/2]th term , making that the middle term of the array .

NewMedian =A[n/2]

Hence, the function is as follows ,

function newMedian = UpdateMedian (OldMedian, NewDataValue, x, n)

if mod(n,2) == 0

if NewDataValue > x(n/2 + 1 )

newMedian = x(n/2+ 1) ;

elseif NewDataValue < x(n/2 )

newMedian = x(n/2 ) ;

elseif NewDataValue > x(n/2 ) && NewDataValue < x((n/2) + 1 )

newMedian = NewDataValue ;

end

end

if mod(n,2) ~= 0

if NewDataValue > OldMedian

if NewDataValue < x((n+1)/2 + 1 )

newMedian = (NewDataValue + x((n+1)/2) )/ 2 ;

else

newMedian = (x ((n+1)/2) + x((n+1)/2 + 1 ) )/ 2 ;

end

end

if NewDataValue < OldMedian

if NewDataValue > x((n+1)/2 - 1 )

newMedian = (NewDataValue + x((n+1)/2) )/ 2 ;

else

newMedian = (x ((n+1)/2) + x((n+1)/2 - 1 ) )/ 2 ;

end

end

end

end

# 3.DERIVATION FOR NEW STANDARD DEVIATION

We know that,

Standard deviation = sqrt( Σ(xi – Mean)^2 / (n – 1) )

Let:

SD0 = Old Standard deviation  
SDN = New Standard deviation  
MO = Old Mean  
MN = New Mean  
n = Number of elements in array  
DV = New Data Value

SD0² = Σ(xi – MO)² / (n – 1)

SD0² = (Σxi² + n(MO)² – 2MO(Σxi)) / (n – 1) … (1)

We know that Σxi = MO × n … (2)

Putting equation (2) in equation (1):

(SD0)² = (Σxi² + n(MO)² – 2MO(MO×n)) / (n – 1)

(SD0)² = (Σxi² – n(MO)²) / (n – 1)

Σxi² = (SD0)² × (n – 1) + n(MO)² … (3)

Now, consider the same equation for New Standard deviation after adding New Data Value (DV).

SDN = sqrt( Σ(xi – MN)² / n )

(SDN)² = (Σxi² + (DV)² + (n+1)(MN)² – 2(MN)(Σxi)) / n

(SDN)² = (Σxi² + (DV)² – (n+1)(MN)²) / n … (4)

Since Σxi² = (SD0)²(n – 1) + n(MO)² (from equation 3),

Putting equation (3) into (4):

(SDN)² = ( (SD0)²(n – 1) + n(MO)² + (DV)² – (n+1)(MN)² ) / n

Therefore,

SDN = sqrt( ( (SD0)²(n – 1) + n(MO)² + (DV)² – (n+1)(MN)² ) / n )

Hence the function is ,

function newStd = UpdateStd (OldMean, OldStd, NewMean, NewDataValue, n)

newStd = sqrt(((OldStd^2 )(n-1) + n(OldMean^2) + NewDataValue^2 - (n+1)\*(NewMean^2))/n ) ;

end

# HISTOGRAM :

When the NewDataValue was still not added in the array(old array), we can use the fuctions min(A) and max(A) to get the minimum and the maximum of the element present in the array A .

Hence , the range would be RANGE= MAX – MIN ;

Fro m struger’s rule ,

Number of bins=⌈log2​(n)+1⌉

Bin width=RANGE/Number of bins

And rounding off the Bin width to the nearest integer , we get a histogram, with the interval

[x1-x2] , [ x2-x3],[x3-x4]……..[xn-1 - xn ]

And lets its frequencies be f1 , f2, f3 , f4 ……fn-1 ;

Where xi+1 – xi = Bin Width ;

Now , when the new element is added , then instead of changing bin width and the intervals , we need to search in loop to find which interval does NewDataValue actually belong to . after finding the interval where actually the NewDataValue belong (consider [xt – xt+1 ] ) and let its frequency be f(t) , then keeping the frequech same for all the other intervals same an increing the value of f(t) by 1 , would be the values for new histogram .

# INSTRUCTIONS TO RUN :

1. File newmean.m has the code for the function NewMean , to be able to run this , you need to export it to matlab or just need to copy paste the code of this file into one of the files in matlab .

And then write these command in any file(name test.m) like shown below (given below is just a example )

OldMean = 5;

NewDataValue = 6;

n = 10;

newMean = UpdateMean (OldMean, NewDataValue, n) ;

disp(newMean);

and then run the script test .

1. File newmeadian.m has the code for the function NewMedian , to be able to run this , you need to export it to matlab or just need to copy paste the code of this file into one of the files in matlab .

And then write these command in any file(name test.m) like shown below (given below is just a example )

OldMedian= 10 ;

n = 5;

x = [1 , 2, 10 , 56, 67 ] ;

NewDataValue = 50 ;

newMedian = UpdateMedian (OldMedian, NewDataValue, x, n) ;

disp(newMedian);

1. File newstd.m has the code for the function Newstd, to be able to run this , you need to export it to matlab or just need to copy paste the code of this file into one of the files in matlab .

And then write these command in any file(name test.m) like shown below (given below is just a example )

OldMean = 5;

OldStd = 2;

NewMean = 5.1;

NewDataValue = 6;

n = 10;

newStd = UpdateStd(OldMean, OldStd, NewMean, NewDataValue, n);

disp(newStd) ;

# QUESTION 3

Given two events A and B with probability:

P(A) ≥ 1 - q1 ,

P(B) ≥ 1 - q2

To show that:

P(A ∩ B) ≥ 1 - (q1 + q2)

We know that, P(A ∪ B) ≤ 1 [since probability is never > 1]

We also know that,

P(A ∪ B) = P(A) + P(B) - P(A ∩ B) ...(1)

Substituting equation (1) in above inequality:

P(A) + P(B) - P(A ∩ B) ≤ 1 ...(2)

From given data,

P(A) + P(B) ≥ (1 - q1) + (1 - q2)

P(A) + P(B) ≥ 2 - (q1 + q2)

Subtracting 1 on both sides:

P(A) + P(B) - 1 ≥ 1 - (q1 + q2) ...(3)

Consider equation (2):

P(A) + P(B) - P(A ∩ B) ≤ 1

⟹ P(A ∩ B) ≥ P(A) + P(B) - 1 ...(4)

Putting inequality (3) in inequality (4):

P(A ∩ B) ≥ 1 - (q1 + q2)

Hence,

P(A ∩ B) ≥ 1 - (q1 + q2)

# QUESTION 4

Given,

In a certain town, there exist 100 buses out of which 1 is red and 99 are blue

The opthalmologist suggests that XYZ sees red objects as red 99% of the time and blue objects as red 2% of the time.

Let SR and SB be the event of seeing red and blue respectively. Let IR and IB be the events of object being red and blue respectively.

According to the question, we need to find:  
P(IR | SR) → the probability the bus was really a red one, when XYZ observed it to be red

Given:

P(SR | IR) = 99/100 (orthonormal/ophthalmologist accuracy) ...(2)

P(IR) = 1/100 (1 red out of 100 objects) ...(3)

P(SR | IB) = 2/100 ...(4)

P(IB) = 99/100 (99 blue out of 100 objects) ...(5)

We know by Bayes theorem:

P(IR | SR) = [ P(SR | IR) × P(IR) ] / [ P(SR | IR) × P(IR) + P(SR | IB) × P(IB) ] ...(1)

Substituting (2), (3), (4), and (5) into equation (1):  
  
P(IR | SR) = (99/100 × 1/100) / [(99/100 × 1/100) + (2/100 × 99/100)]

P(IR | SR) = (99/10000) / [(99/10000) + (198/10000)]  
P(IR | SR) = 99 / (99 + 198)

**P(IR/SR) = 1/3**

Hence, the required probability is 1/3.

# QUESTION 5

Given,

In a village , there are 100 people ,

There are two candidates participating in an election , they are A and B .

95 % people favour A over B and 5% people favour B over A .

To find the accuracy of this mini poll:

out of the three selected people for the majority to declare A, then:  
(i) All the three people should favour A.  
(ii) Two out of three people should favour A.

(i) Probability for all of them to favour A:

**P(A, A, A) = (95/100) × (95/100) × (95/100)**

Since replacement is allowed, while choosing the 2nd and 3rd person favouring A, the probability would still be 95/100.

(ii) Probability for exactly two to favour A and one to favour B:

P(A, A, B) = (95/100) × (95/100) × (5/100)

But since there is replacement, then we need to multiply with 3C1 = 3 (because the B can be in 3 different positions).

**P(A, A, B) = 3 × (95/100) × (95/100) × (5/100)**

Therefore, the total probability = (95/100 × 95/100 × 95/100) + 3 × (95/100 × 95/100 × 5/100)

**P(accuracy) = (95/100 × 95/100) × (110/100)**

Since here in the above example the probability seems to be independent of the number of villagers, so even when the number of villagers increases to 10,000, the probability would still remain the same.

So , P(accuracy) = **(95/100 × 95/100) × (110/100)**

QUESTION 6

A )  
Given there are m voters  
  
• Probability that voters prefer A over B = k/m  
  
Which means k out of m people prefer A over B and (m-k) people prefer B over A.  
  
S is a subset containing n truthful voters.  
  
As exit poll asks randomly with replacement, total number of ways of forming such subsets is   
  
  
Why?  
1st person can be chosen from m persons → m ways.  
2nd person can be chosen from m persons → m ways.  
nth person → m ways.  
  
So, total number of ways = m × m × … (n times) =   
  
Consider the subsets of these n members where i persons preferred A over B.  
  
q(S) = i/n

Here, =

means i out of n people, i people prefer A over B.

nCi.

w.k.t,

=

First derivative,

n=  
  
  
  
Thus,

Therefore, = k/m  
  
⇒ P = k/m  
  
Hence proved.

B)= - we have to prove this.

We considered subset s of n people out of which “i” people voted for A.

q(s)=i/n

and number of such subsets =

= ..eq1

w.k.t,

=

First derivative,

n=

Now multiply this equation with x on both sides and then derivate again

We get,= n+ (n-1)nx

Now multiply eq again with x.

= n+ (n-1)n …eq2

Use this formula for eq1 we get,

=(1/).( (n-1)n.)

Divide with on both sides

=(1/).( (n-1)n.)

Substitute k/m as p

=-

Hence proved

c)

To prove

Let us expand the numerator,

=)+ - 2p

Let us substitute the conclusions we got from part a and b of this questions,.

- +

=

Hence proved

# d)

To prove the fraction of n-sized subsets S for which

(|q(S)−p|>δ )<=

We will use **Chebyshev's Inequality**, which states that for any random variable X with mean μ and standard deviation σ, the probability that X deviates from its mean by more than kσ is at most ​

P(|X−μ| ≥ kσ)≤

Apply this to random variable q(s)

w.k.t from part a and c ,

The **mean** of q(S) is p

The **variance** is =p(1−p)/n

Now, suppose we want to bound the probability that ∣q(S)−p∣>δ

δ=k⋅σ ⇒ k= δ / σ

By Chebyshev’s inequality:

P(∣q(S)−p∣>δ) = P(∣q(S)−p∣>kσ)≤ ​

Substituting the value of k gives:

P(∣q(S)−p∣ > δ)≤ / =

Hence, the proportion of such subsets is at most

# Significance of the Result:

This result tells us that the proportion of random subsets whose observed proportion of votes differs significantly from the true population proportion p, decreases as the subset size n increases.we can get an idea about data dispersion around mean without assuming anything about distribution of data by using chebysev’s Inequality.