# Numerical Integration Quadrature Method

## Gauss Legendre and Gauss Laguerre

# Contents

# Motivation behind the designing of the method

**Integration is one of the most important mathematical operation in scientific phenomenons and in practical uses. Sometimes, we can not find integrals because of not having the exact function to be integrated, instead the value of function at certain points are known. Hence, we use use Gauss Quadrature Method for integration like Gauss Legendre and Gauss laguerre.**

$$\int_a^b w(x)f(x)dx \approx \sum_{k=1}^{n} w_k f(x_k)$$

**Gauss Quadrature methods are more accurate than Newton Cotes because in gauss quadrature nodes x(i) and weights w(i) are unknown and we find them by making function exact. But in newton cotes, x(i) are known which make integral inaccurate.**

[Reference :- integration - Comparison of Newton-Cotes Quadrature and Gaussian Quadrature formulas - Mathematics Stack Exchange]

# Theoretical Explanation

**<u>Gauss Legendre</u>** **For this method, $w(x)$ is equal to 1. Then, we change domain from [a,b] to [-1,1] with formula : $x = ((b-a)/2)*t+((b+a)/2)$. Here, e is the exact solution.**

1. **<u>One Point Method</u> : The method is exact for $f(x) = 1$ and $f(x) = x^2$. Then, find $w(0)$ and $x(0)$ from these two equations. i.e. $w(0) = 2$ and $x(0) = 0$.**
   **Error = (1/3)*diff(f,2)(e).**

1. **<u>Two Point Method</u> : The method is exact for $f(x) = 1$, $f(x) = x^2$, $f(x) = x^3$ and $f(x) = x^4$. Then, find $w(0)$, $w(1)$ and $x(0)$, $x(1)$ from these four equations. I.e. $w(0) = w(1) = 1$, $x(0) = (1/3)^{(1/2)} = - x(1)$.**
   **Error = (1/135)*diff(f,4)(e).**

1. **<u>Three Point Method</u> : The method is exact for $f(x) = 1$, $f(x) = x^2$, $f(x) = x^3$, $f(x) = x^4$, $f(x) = x^5$ and $f(x) =x^6$ . Then, find $w(0)$, $w(1)$,$w(2)$ and $x(0)$, $x(1)$, $x(2)$ from these six equations. I.e. $w(0) = w(2) = 5/9$, $w(1) = 8/9$, $x(0) = (3/5)^{(1/2)} = - x(2)$, $x(1) = 0$.**

   **For every point method, integral is equal to weighted sum i.e. $\sum w(i)*f(x(i))$ from i=1 to n**

Reference :- MA204 class/Google Classroom

# Theoretical Explanation

**<u>Gauss Laguerre</u>** **For this method, w(x) is equal to exp(-x). It's domain is from [0,inf). Here, e is the exact solution.**

1. **<u>One Point Method</u> : The method is exact for f(x) = 1 and f(x) = x^2. Then, find w(0) and x(0) from these two equations. i.e. w(0) = 1 and x(0) = 1.**
   **Error = (1/2)*diff(f,2)(e).**

1. **<u>Two Point Method</u> : The method is exact for f(x) = 1, f(x) = x^2, f(x) = x^3 and f(x) = x^4. Then, find w(0), w(1) and x(0), x(1) from these four equations. I.e. w(0) = (1/4)*(2+(2)^(1/2)), w(1) = (1/4)*(2-(2)^(1/2)), x(0) = 2-(2)^(1/2), x(1) = 2+(2)^(1/2).**
   **Error = (1/6)*diff(f,4)(e).**

1. **<u>Three Point Method</u> : The method is exact for f(x) = 1, f(x) = x^2, f(x) = x^3, f(x) = x^4, f(x) = x^5 and f(x) =x^6 . Then, find w(0), w(1),w(2) and x(0), x(1), x(2) from these six equations. I.e. w(0) = 0.71109, w(1) = 0.27852, w(2) = 0.00139 and x(0) = 0.41577, x(1) = 2.29428, x(2) = 6.28995**

   **For every point method, integral is equal to weighted sum i.e. ∑w(i)*f(x(i)) from i=1 to n**

Refrence :- MA204 class/Google Classroom

# Algorithm : Gauss Legendre

**Step1: Input the function(f), lower limit(a), upper limit(b), gauss point formula(n).**

**Step2 : Change the domain from [a,b] to [-1,1] and update function(f) to that.**

**Step3 : Initialize 3 functions,**

1. **f0(x) = legendre polynomial of n degree - (P(n,x))**
2. **f2(x) = legendre polynomial of n-1 degree - (P(n-1,x))**
3. **f1(x) = differentiation legendre polynomial of degree n - (diff(P(n,x)))**

**Step4 : Calculate roots of f0(x) in x matrix.**

**Step5 : Store the weight in column matrix w by following formula,**

$$w_i = -\frac{2}{(n+1)\,P_{n+1}(x_i)\,P'_n(x_i)}$$
$$= \frac{2}{n\,P_{n-1}(x_i)\,P'_n(x_i)}.$$

Reference :- Legendre-Gauss Quadrature -- from Wolfram MathWorld

# Algorithm : Gauss Legendre

**Step 6 : Calculate weighted sum(WS),**

**WS = ∑w(j)*F(x(j)) from j=1:n**

**i.e. WS = w(1)*F(x(1)) + w(2)*F(x(2)) +.............w(n)*F(x(n))**

**Step 7 : As we know,**

$$\int_{-1}^{1} f(x)\,dx \approx \sum_{i=1}^{n} w_i f(x_i)$$

where

- $n$ is the number of sample points used,
- $w_i$ are quadrature weights, and
- $x_i$ are the roots of the $n$th Legendre polynomial.

**Integration = ∑w(i)*f(x(i)) from i=1:n = ((b-a)/2)*(WS) .**

Reference :- Legendre-Gauss Quadrature -- from Wolfram MathWorld

# Algorithm : Gauss laguerre

**Step1: Input the function(f), gauss point formula(n).**

**Step2 : Initialize 3 functions,**

1. **f0(x) = laguerre polynomial of n degree - (L(n,x))**
2. **f2(x) = laguerre polynomial of n-1 degree - (L(n-1,x))**
3. **f1(x) = differentiation legendre polynomial of degree n - (diff(L(n,x)))**

**Step3 : Calculate roots of f0(x) in x matrix.**

**Step4 : Store the weight in column matrix w by following formula,**

$$w_i = \frac{1}{(n+1)\, L_n'(x_i)\, L_{n+1}(x_i)}$$
$$= -\frac{1}{n\, L_{n-1}(x_i)\, L_n'(x_i)}.$$

Reference :- Legendre-Gauss Quadrature -- from Wolfram MathWorld

# Algorithm : Gauss laguerre

**Step 5 : Add the weight exp(x) in function f(x),**

**F(x) = @(x) exp(x)*f(x).**

**Step 6 : Calculate weighted sum(WS),**

**WS = ∑w(j)*F(x(j)) from j=1:n**

**i.e. WS = w(1)*F(x(1)) + w(2)*F(x(2)) +.............w(n)*F(x(n))**

**Step 7: Print Integration = ∑w(i)*f(x(i)) from i=1:n = WS .**

In this case

$$\int_0^{+\infty} e^{-x} f(x)\, dx \approx \sum_{i=1}^{n} w_i f(x_i)$$

where $x_i$ is the $i$-th root of Laguerre polynomial $L_n(x)$ and the weight $w_i$

Reference :- Legendre-Gauss Quadrature -- from Wolfram MathWorld

# Weights and Nodes

Gauss legendre

| $n$ | $x_i$ | $w_i$ |
|---|---|---|
| 2 | $\pm 0.57735$ | 1.000000 |
| 3 | 0 | 0.888889 |
| | $\pm 0.774597$ | 0.555556 |
| 4 | $\pm 0.339981$ | 0.652145 |
| | $\pm 0.861136$ | 0.347855 |
| 5 | 0 | 0.568889 |
| | $\pm 0.538469$ | 0.478629 |
| | $\pm 0.90618$ | 0.236927 |

Gauss laguerre

| $n$ | $x_i$ | $w_i$ |
|---|---|---|
| 2 | 0.585786 | 0.853553 |
| | 3.41421 | 0.146447 |
| 3 | 0.415775 | 0.711093 |
| | 2.29428 | 0.278518 |
| | 6.28995 | 0.0103893 |
| 4 | 0.322548 | 0.603154 |
| | 1.74576 | 0.357419 |
| | 4.53662 | 0.0388879 |
| | 9.39507 | 0.000539295 |
| 5 | 0.26356 | 0.521756 |
| | 1.4134 | 0.398667 |
| | 3.59643 | 0.0759424 |
| | 7.08581 | 0.00361176 |
| | 12.6408 | 0.00002337 |

Reference :- Legendre-Gauss Quadrature -- from Wolfram MathWorld

# Example : Gauss Legendre

Find the value of the integral

$$I = \int_2^3 \frac{\cos 2x}{1+\sin x}\,dx$$

using Gauss-Legendre two and three point integration rules.

**Solution**

Substituting $x = (t + 5) / 2$ in $I$, we get

$$I = \int_2^3 \frac{\cos 2x}{1+\sin x}\,dx = \frac{1}{2}\int_{-1}^{1} \frac{\cos(t+5)}{1+\sin((t+5)/2)}\,dt.$$

Using the Gauss-Legendre two-point formula

$$\int_{-1}^{1} f(x)dx = f\left(\frac{1}{\sqrt{3}}\right) + f\left(-\frac{1}{\sqrt{3}}\right)$$

we obtain $$I = \frac{1}{2}[0.56558356 - 0.15856672] = \boxed{0.20350842}$$

Using the Gauss-Legendre three-point formula

$$\int_{-1}^{1} f(x)dx = \frac{1}{9}\left[5f\left(-\sqrt{\frac{3}{5}}\right) + 8f(0) + 5f\left(\sqrt{\frac{3}{5}}\right)\right]$$

we obtain $$I = \frac{1}{18}[-1.26018516 + 1.41966658 + 3.48936887] = \boxed{0.20271391}$$

Reference :- Numerical Methods By M.K.Jain,S.R.K.Iyengar & R.K.Jain.pdf - Google Drive

# Solution

Command Window

```
>> Guass_legendre
Enter Your funcion as @(x) sin(x) for sin(x): @(x) ((cos(2*x)/(1+sin(x))))
Enter lower limit: 2
Enter upper limit: 3
which guass point formula you want to apply: 2

ans =

0.203508417487091602958244304327

>>
```

Two point

Command Window

```
>> Guass_legendre
Enter Your funcion as @(x) sin(x) for sin(x): @(x) ((cos(2*x)/(1+sin(x))))
Enter lower limit: 2
Enter upper limit: 3
which guass point formula you want to apply: 3

ans =

  0.202713904806269

>>
```

Three point

Reference :- MATLAB

# Example : Gauss Laguerre

$$I = -\int_0^{\infty} e^{-t}\left(\frac{t}{1+e^{-2t}}\right) dt.$$

We can now use the Gauss-Laguerre's integration methods (4.71) for evaluating the integral with $f(t) = t / (1 + e^{-2t})$. We get for

$n = 1:$ $\quad I = -[0.3817 + 0.4995] = -0.8812.$

$n = 2:$ $\quad I = -[0.2060 + 0.6326 + 0.0653] = -0.9039.$

$n = 3:$ $\quad I = -[0.1276 + 0.6055 + 0.1764 + 0.0051] = -0.9146.$

$n = 4:$ $\quad I = -[0.0865 + 0.5320 + 0.2729 + 0.0256 + .0003] = -0.9173.$

$n = 5:$ $\quad I = -[0.0624 + 0.4537 + 0.3384 + 0.0601 + 0.0026 + 0.0000]$

$\quad = -0.9172.$

Reference :- Numerical Methods By M.K.Jain,S.R.K.Iyengar & R.K.Jain.pdf - Google Drive

# Solution

```
Command Window
>> guass_Laguerre
Enter Your funcion as @(x) sin(x) for sin(x): @(x) -((exp(-x)*(x))/(1+exp(-2*x)))
which guass point formula you want to apply: 1
integration of given function using 1 point formula is -0.880797
```

One Point

```
guass_Laguerre
Enter Your funcion as @(x) sin(x) for sin(x): @(x) -((exp(-x)*(x))/(1+exp(-2*x)))
which guass point formula you want to apply: 2
integration of given function using 2 point formula is -0.881174
>> guass_Laguerre
```

Two Point

Reference :- MATLAB

## Solution

Three Point

```
>> guass_Laguerre
Enter Your funcion as @(x) sin(x) for sin(x): @(x) -((exp(-x)*(x))/(1+exp(-2*x)))
which guass point formula you want to apply: 3
integration of given function using 3 point formula is -0.903891
>> guass_Laguerre
```

Four Point

```
>> guass_Laguerre
Enter Your funcion as @(x) sin(x) for sin(x): @(x) -((exp(-x)*(x))/(1+exp(-2*x)))
which guass point formula you want to apply: 4
integration of given function using 4 point formula is -0.914596
>> guass_Laguerre
```

Five point

```
>> guass_Laguerre
Enter Your funcion as @(x) sin(x) for sin(x): @(x) -((exp(-x)*(x))/(1+exp(-2*x)))
which guass point formula you want to apply: 5
integration of given function using 5 point formula is -0.917257
>>
```

Reference :- MATLAB

# Code Snippets

## Gauss legendre

guass_Laguerre.m | Guass_legendre.m

```
f = input('Enter Your funcion as @(x) sin(x) for sin(x): ');
a = input('Enter lower limit: ');
b = input('Enter upper limit: ');
n = input('which guass point formula you want to apply: ');
F = @(t) f(((b-a)*t +(b+a))/2);          %% Changing domain fo the function from [a,b] to [-1,1]
syms x                                    %% Constructing symbolic variable x using syms
f0(x) = legendreP(n,x);                   %% initializing legendre polynomial of n degree
f2(x) = legendreP(n-1,x);                 %% initializing legendre polynomial of n-1 degree
f1(x) = diff(f0);                         %% differentiating legendre polynomial of degree n
x = vpasolve(f0 == 0);                    %% finding numerical roots of legendre polynomial of n degree using vpasolve
w = zeros(n);                             %% initializing column zero matrix w for storing n weights
for i=1:n                                 %% finding n weights and storing in column matrix w using following formula
    w(i) = 2/(n*f2(x(i))*f1(x(i)));
end
WS = 0;                                   %% initializing weighted sum to 0
for j=1:n
    WS = WS + w(j)*F(x(j));               %% finding weighted sum i.e. WS=[w(1)*F(x(1))+w(2)*F(x(2))+.....+w(n)*F(x(n))]
end
ans = ((b-a)/2)*WS                        %% printing answer as ((b-a)/2)*(weighted sum)
```

Reference :- MATLAB

# Code Snippets

## Gauss laguerre

guass_Laguerre.m | Guass_legendre.m

```matlab
f = input('Enter Your funcion as @(x) sin(x) for sin(x): ');
n = input('which guass point formula you want to apply: ');
syms x                                   %% Constructing symbolic variable x using syms
f0(x) = laguerreL(n,x);                  %% initializing laguerre polynomial of n degree
f1(x) = diff(f0);                        %% differentiating laguerre polynomial of degree n
f2(x) = laguerreL(n-1,x);                %% initializing laguerre polynomial of n-1 degree
x = vpasolve(f0 == 0);                   %% finding numerical roots of laguerre polynomial of n degree using vpasolve
w = zeros(n);                            %% initializing column zero matrix w for storing n weights
for i=1:n                                %% finding n weights and storing in column matrix w using following formula
    w(i) = (-1)/(n*f2(x(i))*f1(x(i)));
end
F = @(x) exp(x)*f(x);                    %% muliply weight as exp(x)to given function
WS = 0;                                  %% initializing weighted sum to 0
for j=1:n
    WS = WS + w(j)*F(x(j));              %% finding weighted sum i.e. WS=[w(1)*F(x(1))+w(2)*F(x(2))+.....+w(n)*F(x(n))]
end
WS                                       %% printing answer as weighted sum
```

Reference :- MATLAB

# Functions Used in Codes

- legendreP(n,x) : generates legendre polynomial in x variable of n degree.
- laguerreL(n,x) : generates laguerre polynomial in x variable of n degree.
- vpasolve(equation = 0) : Find numerical solution of algebraic equations.
- diff(function) : Used to approximate derivatives of function.
- int(function) : Used to approximate integrals of function.
- syms arg1, arg2, …..argn : Shortcut for constructing symbolic variables.
- zeroes(n,m) : Used to generate a matrix with n rows and m column with all entries zero.

Reference :- MATLAB (help function)

# Error Order

**Gauss Legendre**

The error term is

$$E = \frac{2^{2n+1}(n!)^4}{(2n+1)[(2n)!]^3} f^{(2n)}(\xi).$$

**Gauss Laguerre**

The error term is

$$E = \frac{(n!)^2}{(2n)!} f^{(2n)}(\xi)$$

Reference :- Numerical Methods By M.K.Jain,S.R.K.Iyengar & R.K.Jain.pdf - Google Drive

# Convergence Criteria

**Gauss legendre and Gauss laguerre methods convergence for all continuous functions. Because integration which is summation of weighted sum at finite point is always finite if the function is bounded at roots or nodes of legendre polynomial for gauss legendre method and laguerre polynomial for gauss laguerre method. Integration quadrature methods converges to actual value if integration of given function is convergent.**

# Computational Costs

**The classic approach for computing Gauss quadrature nodes and weights is the Golub–Welsch algorithm, which requires O(n^2 ) operations. However, in recent years several fast algorithms have been developed that require only O(n) operations. Currently, for classic weight functions, Bogaert's algorithm for Gauss–Legendre and the Glaser–Lui–Rokhlin algorithm for Gauss–Laguerre.**

**In my code, computational cost is O(n+t) where n is point formula we used and t is time taken by function vpasolve to solve legendre polynomial of degree n.**

# Gauss Quadrature VS Newton Cotes

- **Gaussian quadrature find a rule that lets us exactly integrate polynomials of as large a degree as possible. This winds up being degree 2n−1, since we have control of 2n parameters in specifying a quadrature rule at n nodes, and the parameters are independent.**
- **Order is O(n^2).**
- **Gauss Quadrature is numerically more stable than Newton Cotes.**
- **Methods such as Gaussian quadrature and Clenshaw–Curtis quadrature with unequally spaced points (clustered at the *endpoints* of the integration interval) are stable and much more accurate, and are normally preferred to Newton–Cotes.**

- **Newton-Cotes pick evenly spaced points in the interval, draw the interpolating polynomial of minimal degree through them, and integrate the polynomial. A Newton-Cotes rule on n nodes is exact for polynomials of degree at most n−1.**
- **Order is O(n^2).**
- **Higher forms of Newton's also tend to have large coefficients. These will act as error multipliers and therefore it is less numerically stable than Gaussian.**

# Method Implementation in Real Life

**The phenomenon of gradual movement in landslides creates a special challenge to the geology and engineering community because their geometric character can change over time. Geometrical changes and progressive displacements in earth flows and other slow moving landslides triggered by climatic changes may be addressed by digital modeling. Geometric models showing the progression of the landslides over time can serve as important tools to determine or predict the evolution of a given slope. Gaussian quadrature, a numerical integration technique though fixed points, is employed to compute geometrical areas defined by stratigraphic (soil or rock layering) units, vertical pole projections and a slip surface, based on kinematic admissibility.**

Reference :- Some Applications of Gaussian Quadrature and Neural Network Modeling in Earth Flows and Other Slow-Moving Landslides in Cohesive Slope Materials (wmich.edu)