Section: Numerical integration - Lecture 2

Goal: Given a function $f(x)$, $a \leq x \leq b$, we want to evaluate or approximate $\int_a^b f(x)dx$.

As a first note, one can make a change of variable to transform the definite integral $\int_a^b f(x)dx$ to $\int_{-1}^1 \bar{f}(y)dy$. So in the following we only consider the definite integral

$$\int_{-1}^{1} f(x)dx.$$

[2] Gaussian quadrature

The idea of numerical quadrature rules in lecture 1 is to approximate a function by polynomials, then we approximate the definite integral of the function by the definite integral of the polynomial. It leads to a quadrature rule of the form

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

The idea of Gaussian quadrature is the following: We use exactly the same structure as above, i.e., we assume that the definite integral can be approximated as

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

But this time we think of w_i and x_i as unknows and seek for the best choice of them. Ideally, as we have totally $2n$ unknows (w_i and x_i, $i = 1, \cdots, n$), we can have a formula that is exact for polynomial of degree $2n - 1$.

Read Gaussian quadrature in [wiki] for more details.

[2.1] Midpoint rule

Consider $n = 1$, that is, we looks for w_1 and x_1 such that the formula

$$\int_{-1}^{1} f(x)dx = w_1 f(x_1)$$

is true for polynomial of degree less than or equals to 1. It can be shown that the only solution is $w_1 = 2$ and $x_1 = 0$, which is exactly the midpoint rule.

[2.2] Gauss-Legendre quadrature

[2.2.1] $n = 2$

Consider $n = 2$. To solve the unknows we can try $f(x) = 1$,$f(x) = x$,$f(x) = x^2$,$f(x) = x^3$ and solve

$$\begin{cases} w_1 + w_2 = \int_{-1}^{1} 1 dx = 2 \\ w_1 x_1 + w_2 x_2 = \int_{-1}^{1} x dx = 0 \\ w_1 x_1^2 + w_2 x_2^2 = \int_{-1}^{1} x^2 dx = \frac{2}{3} \\ w_1 x_1^3 + w_2 x_2^3 = \int_{-1}^{1} x^3 dx = 0 \end{cases},$$

which gives us $w_1 = w_2 = 1$, $x_1 = \frac{1}{\sqrt{3}}$,$x_2 = -\frac{1}{\sqrt{3}}$, i.e.,

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

In fact, this is called the Gauss-Legendre quadrature. For the formula of general number of points n, see wiki for further details.

[2.2.3] General n

To find x_i and w_i, it is known that x_i are the roots of the Legendre polynomials. To find Legendre polynomials, we can use the recursive formula

$$L_n(x) = \frac{1}{n}((2n-1)xL_{n-1}(x) - (n-1)L_{n-2}(x))$$

where $L_0 = 1$ and $L_1 = x$.

Coefficients of Legendre polynomial

Here we try to find the coeffients of the Legendre polynomials. We define a function that calculate the coefficients of n-th degree legendre polynomial.

```julia
function Legendre_poly(n)
    if n<=0
        c=[1.0];
    elseif n==1
        c = [1.0,0.0]
    else
        a = Legendre_poly(n-1)
        append!(a,0.0)

        c = Legendre_poly(n-2)
        b = [0.0,0.0]
        append!(b,c)

        c = ((2*n-1)*a-(n-1)*b)/n
    end

    return c
end
```

```
Legendre_poly (generic function with 1 method)
```

For example, the Legendre polynomial of degree 2 is $\frac{1}{2}(3x^2 - 1)$ which means $L_2(x) = 1.5x^2 + 0x - 0.5$. Which has roots $\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}$, same as the calculation above.

```julia
Legendre_poly(2)
```

```
3-element Array{Float64,1}:
  1.5
  0.0
 -0.5
```

Next, we need to calculate to roots of $L_n(x)$ and the weights w_i which is not an easy task. We solve it by the help of Polynomials.jl.

Remark:

We'll use packages Polynomials. Run `using Pkg; Pkg.add("Polynomials")` if you don't have Polynomials installed.

```julia
using Polynomials
```

```julia
function Gauss_Legendre(n)
    # returns x: the roots of Lenendre polynomails of order n
    #         w: the desired weights(w_i's)

    p = Poly(reverse(Legendre_poly(n)))
    x = roots(p)

    A = zeros(n,n)
    b = zeros(n,1)
    A[1,:] = ones(n)
    b[1] = 2
    for i=2:n
        for j=1:n
            A[i,j] = x[j]^(i-1)
        end
        if i%2 ==0
            b[i] = 0
        else
            b[i] = (2.0)/i
        end
    end

    w = A\b

    return x,w
end
```

```
Gauss_Legendre (generic function with 1 method)
```

```julia
function Gauss_Legendre_quadrature(f::Function,n::Int)
    x,w = Gauss_Legendre(n)
    y=f.(x)

    return  (y'*w)[1]
end
```

```
Gauss_Legendre_quadrature (generic function with 1 method)
```

Example 1

Evaulate $\int_{-1}^{1} e^x dx$ using Gauss-Legendre quadrature rule. Note that the exact solution for this integral is $e - \frac{1}{e}$.

For this example, we can see that the error is already $O(10^{-16})$ for $n = 8$.

```julia
exact = exp(1) - exp(-1)
```

```
2.3504023872876028
```

```julia
abs(Gauss_Legendre_quadrature(exp,4)-exact)
```

```
2.9513122568047834e-7
```

```julia
abs(Gauss_Legendre_quadrature(exp,8)-exact)
```

```
4.440892098500626e-16
```

[2.3] Chebyshev–Gauss quadrature

The Chebyshev-Gauss quadrature rule is to approximate the definite integral of the form

$$\int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}} dx$$

.

It can be shown that the weights w_i are $\frac{\pi}{n}$ and the points $x_i = \cos\left(\frac{2i-1}{2n}\pi\right)$. Therefore we have

$$\int_{-1}^{1} f(x)w(x)dx \approx \frac{\pi}{n} \sum_{i=1}^{n} f\left(\cos\left(\frac{2i-1}{2n}\pi\right)\right).$$

See wiki Chebyshev–Gauss quadrature for further detail.

Example

Evaulate $\int_{-1}^{1} e^x dx$ using Chebyshev-Gauss quadrature rule for $n = 10, 100, 1000$.

Consider $g(x) = e^x\sqrt{1-x^2}$. Then we can use Chebyshev-Gauss quadrature rule to evaluate the integral.

We define a function to evaluate the integral.

```julia
function ChebyQuadrature(f::Function,n::Int)
    g(x) = f(x)*sqrt(1-x^2)
    sum=0
    for i=1:n
        sum = sum +  g(cos((2i-1)*pi/(2n)))
    end
    return pi*sum/n
end
```

```
ChebyQuadrature (generic function with 1 method)
```

```julia
abs(ChebyQuadrature(exp,10)-exact)
```

```
0.01281402648467811
```

```julia
abs(ChebyQuadrature(exp,100)-exact)
```

```
0.00012692529833824295
```

```julia
abs(ChebyQuadrature(exp,1000)-exact)
```

```
1.269134153325524e-6
```