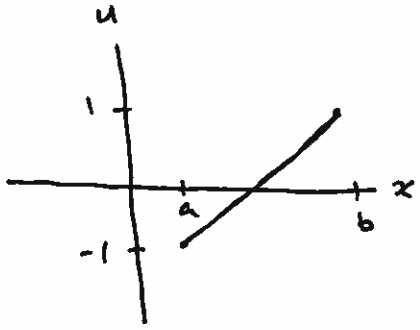
§4.7 Gaussian Quadrature

The idea behind Gaussian Quadrature is to find c_i and x_i so that

$$\int_a^b f(x)dx = \sum_{i=1}^{n} c_i f(x_i)$$

with as high a degree of precission as possible.

First note that



$$u = \frac{2}{b-a}(x-a) - 1$$

$$u = \frac{2x-b-a}{b-a}$$

$$du = \frac{2}{b-a}dx$$

$$\Rightarrow \left(\int_a^b f(x)dx = \frac{b-a}{2} \int_{-1}^{1} f\left(\frac{u(b-a)+a+b}{2}\right)du \right)$$

We will concentrate on

$$\int_{-1}^{1} f(v)dv = \sum_{i=1}^{n} c_i f(x_i)$$

n = 1 case

$f(v)=1$: $\int_{-1}^{1} dv = c_1 \Rightarrow c_1 = 2$

$f(v)=v$: $\int_{-1}^{1} v\,dv = c_1 x_1 \Rightarrow 0 = c_1 x_1 \Rightarrow x_1 = 0$

So $\int_{-1}^{1} f(x)\,dx = 2f(0)$

or $\int_{a}^{b} f(x)\,dx = (b-a)\,f\left(\frac{a+b}{2}\right)$

This is the Midpoint Rule.

n = 2 case

$f(v)=1$: $\int_{-1}^{1} dv = c_1 + c_2 \Rightarrow c_1 + c_2 = 2$

$f(v)=v$: $\int_{-1}^{1} v\,dv = c_1x_1 + c_2x_2 \Rightarrow c_1x_1 + c_2x_2 = 0$

$f(v)=v^2$: $c_1x_1^2 + c_2x_2^2 = \frac{2}{3}$

$f(v)=v^3$: $c_1x_1^3 + c_2x_2^3 = 0$

Solve this 4 by 4 system of equations to get $c_1 = c_2 = 1$ and $x_1 = -\frac{\sqrt{3}}{3}$, $x_2 = \frac{\sqrt{3}}{3}$

So $\int_{-1}^{1} f(x)\,dx \approx f\left(-\frac{\sqrt{3}}{2}\right) + f\left(\frac{\sqrt{3}}{2}\right)$

Remember that

$$\int_a^b f(x)\,dx = \int_{-1}^{1} f\left(\frac{(b-a)u + a+b}{2}\right)\left(\frac{b-a}{2}\right)du$$

We use Legendra Polynomials to get values for c_i, x_i in the case $n > 1$. See pages 230-232 and Table 4.12 on page 232.

Code for Gaussian Quadrature

Inputs:
a, b - limits of integration
n - number of nodes to use
myf - integrand (another routine)
Outputs: approximation to $\int_a^b f(x)dx$

~~Case n:~~
switch n
case 2
r = [0.5773502692 -0.5773502692]';
c = [1.0 1.0];
case 3
r = [0.7745966692 0.0 -0.7745966692]';
c = [0.5555555556 0.8888888889 0.5555555556]';
case 4
⋮

otherwise
msg = 'n must be 2, 3, 4 or 5, Using n = 2'
r = [0.5773502692 -0.5773502692]';
c = [1.0 1.0];
end
c1 = (b-a)/2;
c2 = (b+a)/2;
approximation = c1*(c*myf(c1*r+c2));
% count = count + n;

Note that myf must be able to return a vector of values and we are using Dot Product to calculate $\sum c_i f(x_i)$