# Methodes de Newton-Cotes

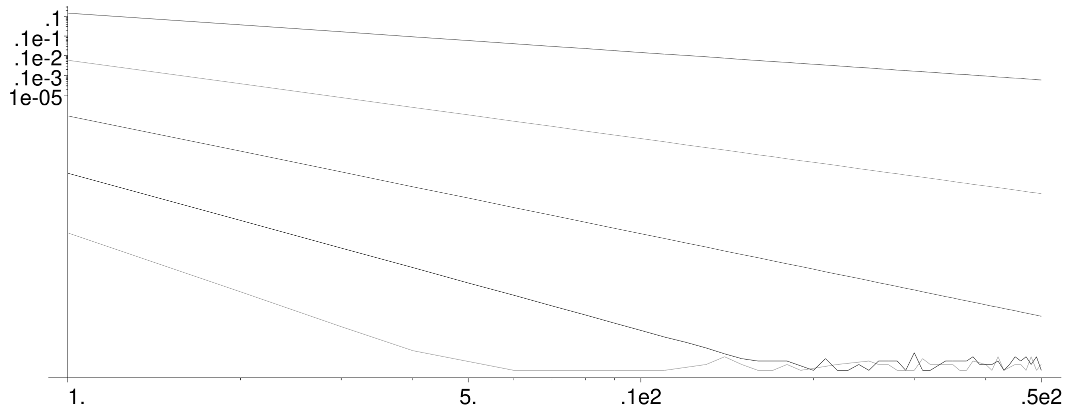
```
> restart;
  ncotes:=proc(l)
    local x, W, eqns;
    x:=array(0..l, [j/l $ j=0..l]):
    W:=array(0..l, ['w[j]' $ j=0..l]):
    eqns:={seq(sum('w'[j]*x[j]^k, j=0..l)=int(x^k, x=0..1),
  k=0..l)}:
    subs(solve(eqns, convert(W, set)), eval(W));
  end:
Warning, `k` in call to `seq` is not local
> ncotes2:=proc(l)
    local x, Q, L;
    x:=array(0..l, [j/l $ j=0..l]):
    Q:=product((z-x[j]), j=0..l):
    L:=map(x->Q/(z-x)/subs(z=x, Q/(z-x)), x):
    map(P->int(P, z=0..1), L);
  end:
> quad:=proc(l)
    option remember;
    local W;
    W:=ncotes(l);
    proc(f, a, b, n)
      local h, ksi;
      h:=(b-a)/n;
      ksi:=h/l;
      evalf((b-a)/n*sum(sum(W[j]*f(a+k*h+j*ksi), j=0..l),
  k=0..n-1));
    end;
  end:

> trapezes:=quad(1):
  simpson:=quad(2):
  boole:=quad(4):
  weedle:=quad(6):
  evalf(weedle(exp, 0, 1, 10));
```

$$1.718281826$$

```
> with(plots):
  Digits:=20;
  erreur:=(l, k)->abs((quad(l))(exp, 0, 1, k)-exp(1.)+1):
  Color:=[blue, red, green]:
  graphes:=[seq(loglogplot([[k, 'erreur'(l, k)] $ k=1..50],
  color=Color[1+(l mod 3)]), l=[1, 2, 4, 6, 8])]:
  display(graphes);
```

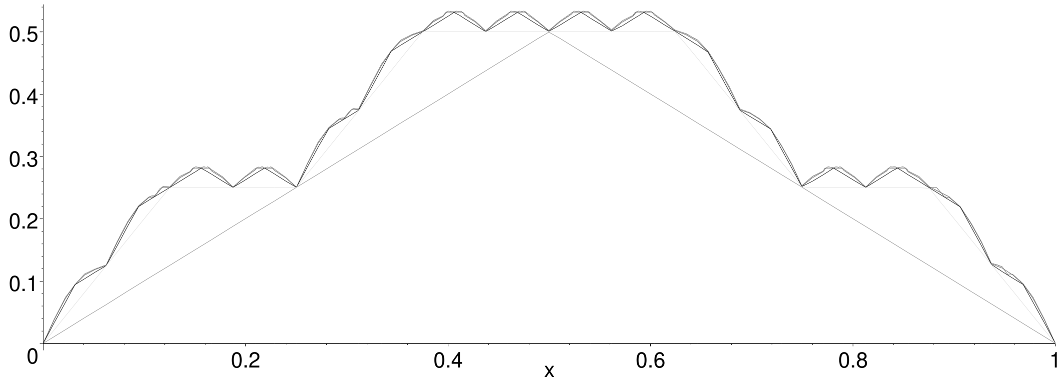
$$Digits := 20$$




```
> h:=x->1/2-abs(x-floor(x)-1/2):
  f:=(n, x)->sum(1/4^k*h(4^k*x), k=0..n-1);
```

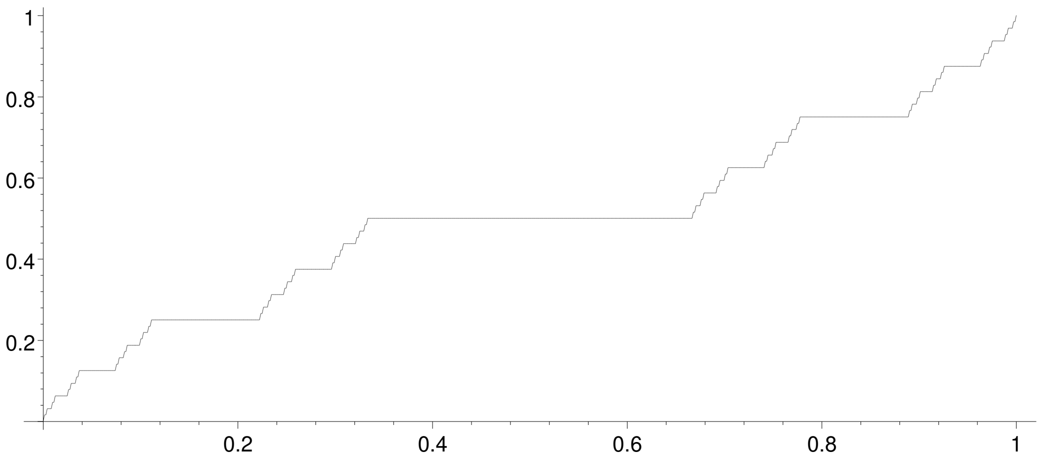
$$f := (n, x) \to \sum_{k=0}^{n-1} \frac{\mathrm{h}(4^k\, x)}{4^k}$$

```
> plot(['f'(k, x) $ k=0..6], x=0..1);
```

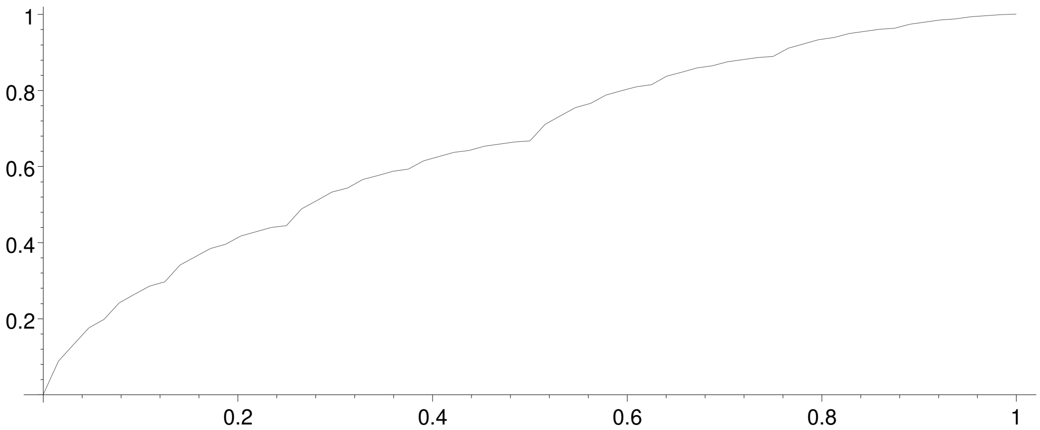



```
> iter:=proc(fonc, x)
    local h, k, y;
    y:=x[1];
    for k to nops(x)-1 do
      h:=x[k+1, 2]-x[k, 2];
      y:=y, fonc(x[k], x[k+1]);
    od;
  [y];
  end:
> diable:=(x, y)->([(2*x[1]+y[1])/3, (x[2]+y[2])/2],
  [(x[1]+2*y[1])/3, (x[2]+y[2])/2], y):
  h:=(x, y)->([(x[1]+y[1])/2, 1/3*x[2]+2/3*y[2]], y):

> id:=[[0, 0], [1, 1]]:
  iterdiable:=x->iter(diable, x):
  iterh:=x->iter(h, x):
  plot((iterdiable@@6)(id));
  plot((iterh@@6)(id));
```



1
0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
1

1
0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
1