wxMaxima pour l'Analyse Volume II

Auteur : Zachary Hannan, Solano Community College
Les corrections sont réalisées par Michel Gosse

Exercices du Chapitre 2 et leur corrigé

Présentation

Le manuel wxMaxima for Calculus Volume II, dont l'auteur est Zachary Hannan, propose l'étude de nombreux domaines de l'analyse à l'aide du logiciel Maxima et de son interface graphique wxMaxima. A l'occasion de la traduction française de ce manuel, la correction des exercices de chaque chapitre est proposée. Ce document reprend les exercices du manuel et leur correction à l'aide de wxMaxima.

La dernière version de ce document, ainsi que les fichiers source pour wxMaxima, sont téléchargeables sur le site https://maxima-french-doc.fr/, à la rubrique Documentation.

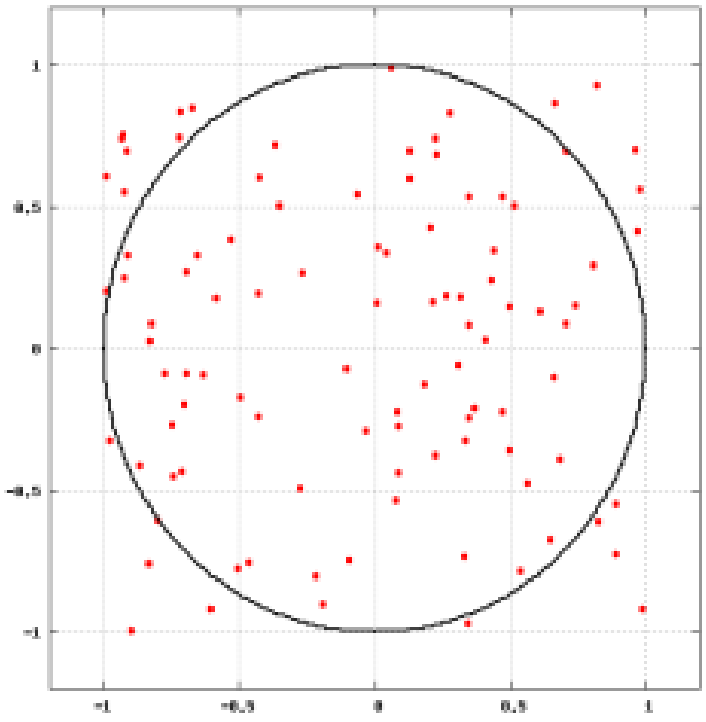
Enoncé des exercices

1. Calculez $\int_1^e \cos(\ln x)\,\mathrm{d}x$ en utilisant une approximation du point médian avec $n = 100$. Construisez un graphique illustrant cette approximation dans le style de l'exemple 2.1.1. Vérifiez votre réponse à l'aide de `integrate`.
2. Recommencer l'exercice précédent en utilisant une approximation par les trapèzes dans le style de l'exemple 2.2.1.
3. Pour $f(x) = \cosh x$ et $g(x) = \cos x$, calculez les approximations par la méthode des trapèzes avec $n = 3$ sur $[0, 1]$, incluant des graphiques illustrant les trapèzes. Quelle approximation est une sous-estimation de la valeur de l'intégrale ? Quelle est une sur-estimation ? Comment les erreurs sont-elles liées à la concavité de f et g ?
4. Des mesures de tension sont effectuées pendant une période dans un circuit alternatif :

   ```
   t(s).....v(t)(V)
   0/1200      4.2
   1/1200     67.5
   2/1200    114.0
   3/1200    112.7
   4/1200     68.1
   5/1200     -0.8
   6/1200    -69.4
   7/1200   -110.7
   8/1200    114.1
   9/1200    -68.7
   10/1200    -4.6
   ```

   Calculez $\int_0^{\frac{10}{1200}} v^2(t)\,\mathrm{d}t$ en utilisant une approximation par trapèzes. Incluez un graphique pour illustrer l'approximation.
5. Répétez l'exercice précédent en utilisant une approximation par la méthode de Simpson. Incluez un graphique montrant les paraboles d'interpolation dans le style de l'exemple 2.3.3.

6. Calculez $\int_0^{\frac{\pi}{2}} \cos(\sin x)\ \mathrm{d}x$ en utilisant `quad_qag`. Produisez un graphique coloré illustrant l'aire représentée par l'intégrale.
7. Recommencez le calcul de l'intégrale de l'exercice précédent en utilisant la méthode de Simpson avec $n = 10$, dans le style de l'exemple 2.3.2. Réalisez un graphique pour illustrer cette approximation.
8. Recalculez l'intégrale de l'exercice précédent en utilisant une méthode de Monte Carlo avec 1000 points aléatoires.
9. * Nous pouvons utiliser une méthode de Monte Carlo pour approximer π. Pour visualiser la méthode, commencez par utiliser `makelist` pour générer 100 points aléatoires dans un carré de côté 2 centré à l'origine. Tracez ces points sur le même graphique que le cercle unité. Votre graphique être similaire à celui-ci :



   L'aire du cercle est donnée par $A_{\text{cercle}} = \pi \cdot r^2 = \pi$ et l'aire du carré est $A_{\text{carré}} = 4$, donc on s'attend à ce que la proportion de points tombant à l'intérieur du cercle soit $\frac{A_{\text{cercle}}}{A_{\text{carré}}} = \frac{\pi}{4}$. Utilisez alors votre graphique pour trouver une approximation de π.
10. ** Pour appliquer la méthode de Monte Carlo à l'exemple précédent, nous voulons automatiser le processus de comptage des points qui se trouvent à l'intérieur du cercle unité. Cela peut être accompli en utilisant une boucle `for-do` pour générer un point à la fois et une instruction `if-then` pour tester si chaque point est à l'intérieur du cercle ou non. Avant l'exécution de la boucle, nous devons définir le point de départ d'un compteur `n:0`. Si un point est à l'intérieur du cercle, on ajoute 1 au compteur en écrivant `n:n+1`. Si nous incluons les points tombant exactement sur le bord du cercle, l'instruction `if-then` appropriée est : `if (x^2+y^2<1 or x^2+y^2=1) then n:n+1`.
    Enfin, nous utilisons une autre instruction `if-then` demandant à wxMaxima d'afficher la valeur de `n` après la dernière itération de la boucle.

    Construisez la boucle `do` appropriée, en commençant par un petit nombre d'itérations jusqu'à ce que tout fonctionne correctement. Pendant vos expérimentations avec le programme, il est conseillé de demander à wxMaxima d'afficher `n` à chaque itération afin de faciliter la détection de tout problème dans le calcul. Une fois que vous êtes sûr que cela fonctionne, utilisez la boucle pour générer 10 000 points aléatoires. Utilisez la proportion de points à l'intérieur du cercle pour approximer π trois fois séparément afin d'avoir une idée de l'incertitude sur le résultat.

Corrigé des exercices

1 Exercice1

(% i1) f(x) :=cos(log(x));
(%o1) $f(x) := \cos(\log(x))$

(% i2) delx :(%e-1)/100;
(delx) $\frac{\%e-1}{100}$

(% i3) float(delx);
(%o3) 0.01718281828459045

(% i4) define(x(i),delx*i);
(%o4) $x(i) := \frac{(\%e-1)\,i}{100}$

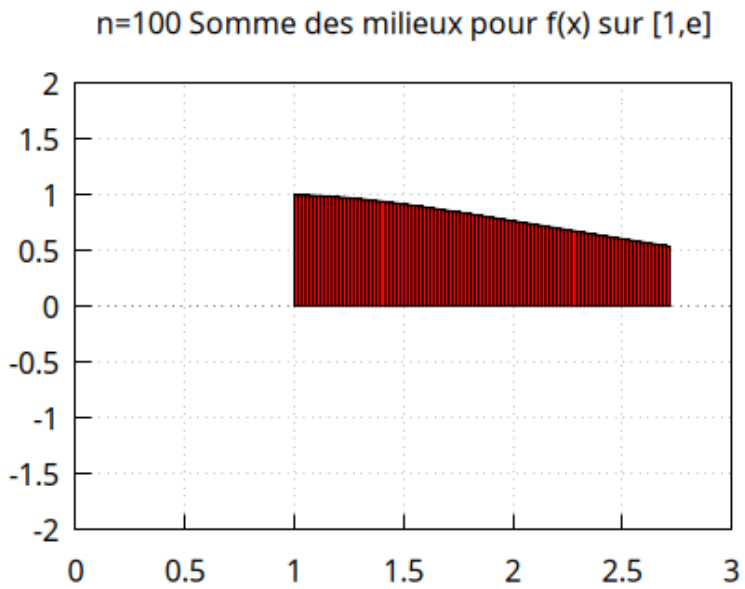
(% i5) 'integrate(f(x),x,1,%e)=float(sum(f((1+x(i)+1+x(i-1))/2)*delx,i,1,100));
(%o5) $\int_1^{\%e} \cos(\log(x))\,dx = 1.378028421455331$ Vérification par calcul direct

(% i6) integrate(f(x),x,1,%e);
(%o6) $\frac{\%e\sin(1)+\%e\cos(1)}{2}-\frac{1}{2}$

(% i7) float(%);
(%o7) 1.3780246135473637

(% i8) RECTANGLES :makelist(rectangle([1+x(i-1),0],[1+x(i),f((2+x(i)+x(i-1))/2)]),i,1,100);
(La configuration indique de supprimer la sortie des cellules longues)

(% i10)
```
load(draw)$
wxdraw2d(
grid=true,
xaxis=true,
yaxis=true,
xrange=[0,3],
yrange=[-2,2],
title= n=100 Somme des milieux pour f(x) sur [1,e] ,
color=black,
fill_color=red,
border=true,
RECTANGLES,
explicit(f(x),x,1,%e)
);
```



(%t10)

(%o10)

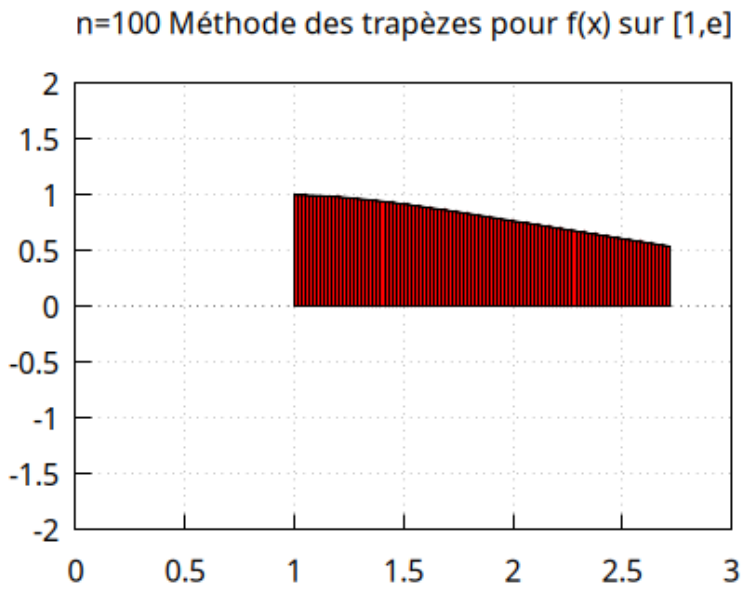
2 Exercice 2

(% i11) 'integrate(f(x),x,1,%e)=float(sum((f(1+x(i-1))+f(1+x(i)))/2*delx,i,1,100));

(%o11) $\int_1^{\%e} \cos\left(\log\left(x\right)\right) dx = 1.378016997462572$ On retrouve bien une approximation identique sur plusieurs décimales par cette méthode

(% i12)
```
TRAPEZES :makelist(
polygon([
[1+x(i-1),0],
[1+x(i-1),f(1+x(i-1))],
[1+x(i),f(1+x(i))],
[1+x(i),0]
]),
i,1,100
);
```
(La configuration indique de supprimer la sortie des cellules longues)

(% i13)
```
wxdraw2d(
grid=true,
xaxis=true,
yaxis=true,
xrange=[0,3],
yrange=[-2,2],
title= n=100\ Méthode\ des\ trapèzes\ pour\ f(x)\ sur\ [1,e] ,
color=black,
fill_color=red,
border=true,
TRAPEZES,
explicit(f(x),x,1,%e)
);
```



(%t13)

(%o13)

3 Exercice 3

(% i15) f(x) :=cosh(x) ;g(x) :=cos(x) ;
(%o14) $f(x) := \cosh(x)$

(%o15) $g(x) := \cos(x)$

(% i16) delx :(1-0)/3$

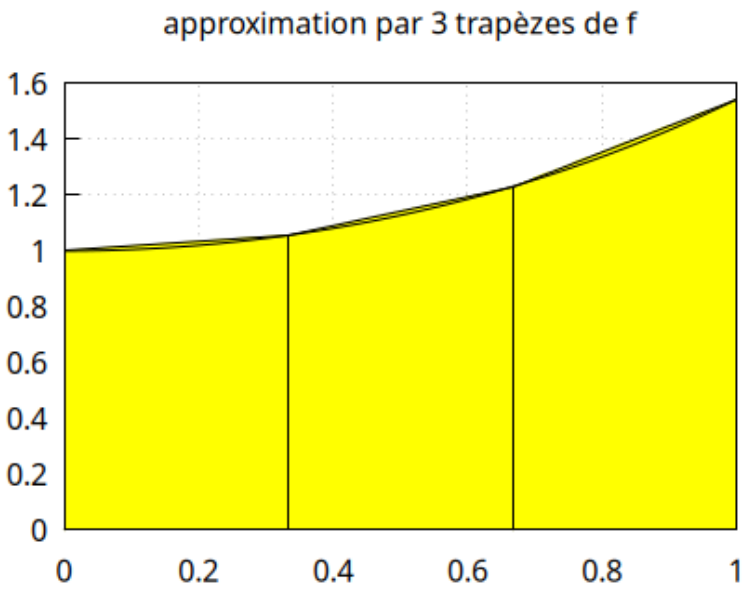
(% i17) x(i) :=delx*i$

(% i18) AREA(i) :=((f(x(i-1))+f(x(i)))/2)*delx ;
(%o18) $AREA(i) := \frac{f(x(i-1)) + f(x(i))}{2}\text{delx}$

(% i19) float(sum(AREA(i),i,1,3)) ;
(%o19) 1.186062588427065

(% i20) TRAPEZOIDS :makelist(polygon([[x(i-1),0], [x(i),0],
[x(i),f(x(i))], [x(i-1),f(x(i-1))]]),i,1,3)$

(% i21)
```
wxdraw2d(
title= approximation par 3 trap\èzes de f ,
grid=true,
xaxis=true,
yaxis=true,
yrange=[0,1.6],
color=black,
border=true,
fill_color=yellow,
TRAPEZOIDS,
line_width=1,
explicit(f(x),x,0,1)
) ;
```



(%t21)

(%o21)

(% i22) float(integrate(f(x),x,0,1));
(%o22) 1.1752011936438014 f est convexe, l'approximation est une sur-estimation de l'aire

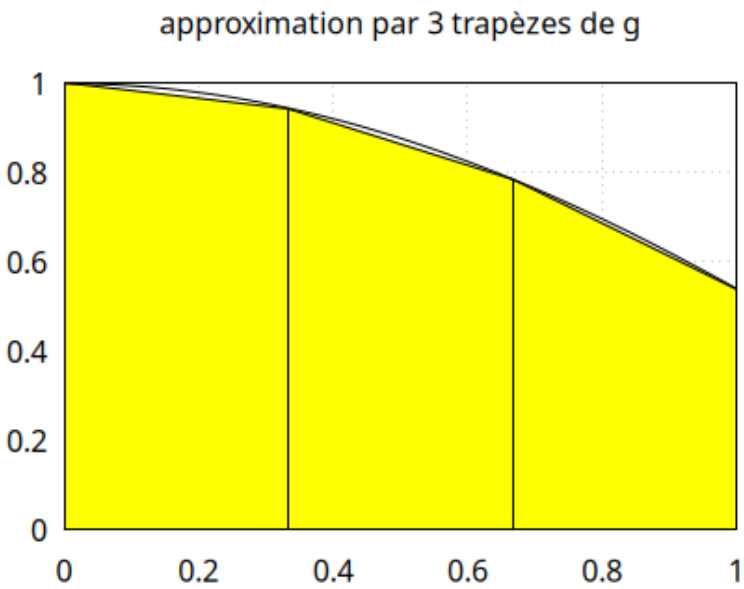
(% i23) AREA2(i) :=((g(x(i-1))+g(x(i)))/2)*delx;
(%o23) $AREA2(i) := \frac{g\left(x\left(i-1\right)\right)+g\left(x(i)\right)}{2}\text{delx}$

(% i24) float(sum(AREA2(i),i,1,3));
(%o24) 0.8336651200085852

(% i25) float(integrate(g(x),x,0,1));
(%o25) 0.8414709848078965

(% i26) TRAPEZOIDS2 :makelist(polygon([[x(i-1),0], [x(i),0],
[x(i),g(x(i))], [x(i-1),g(x(i-1))]]),i,1,3)$

(% i27)
```
wxdraw2d(
title= approximation par 3 trap\èzes de g ,
grid=true,
xaxis=true,
yaxis=true,
yrange=[0,1],
color=black,
border=true,
fill_color=yellow,
TRAPEZOIDS2,
line_width=1,
explicit(g(x),x,0,1)
);
```



(%t27)

(%o27) La fonction g est concave et on obtient une sous-estimation de l'aire

4 Exercice 4

Générons les points à partir des données :

(% i28) X : makelist(k/1200, k, 0, 10)$;

(% i29) Y : [4.2, 67.5, 114.0, 112.7, 68.1, -0.8, -69.4, -110.7, 114.1, -68.7, -4.6]$;

(% i30) Y2 : map(lambda([y], y^ 2), Y)$;

(% i31) POINTS :makelist([X[i],Y2[i]],i,1,10)$

(% i32) delt :1/1200$

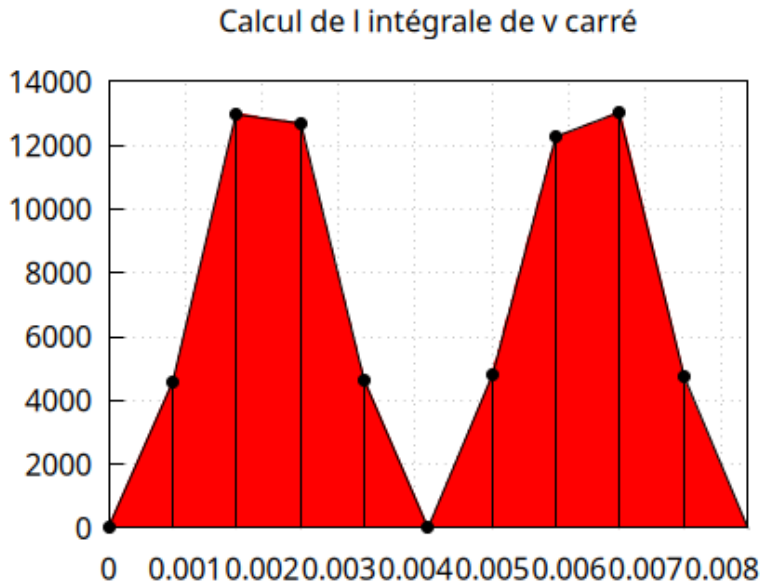
(% i33) TRAPEZOID(i) :=0.5*(Y2[i]+Y2[i+1])*delt$

On additionne l'aire des trapèzes pour obtenir l'approximation cherchée

(% i34) sum(TRAPEZOID(i),i,1,10) ;
(%o34) 58.10045

(% i35) TRAPEZOIDS :makelist(polygon([[X[i],0],
[X[i+1],0],[X[i+1],Y2[i+1]],[X[i],Y2[i]]]),i,1,10)$

```
(% i36) wxdraw2d(
grid=true,
xaxis=true,
yaxis=true,
xrange=[0,10/1200],
yrange=[0,14000],
title= Calcul de l int\égrale de v carr\é ,
color=black,
border=true,
fill_color=red,
TRAPEZOIDS,
point_type=7,
points(POINTS)
);
```



(%t36)

(%o36)

5 Exercice 5

```
(% i37) ratprint : false$
```

Expression générale d'une parabole :

```
(% i38) p(i,x) :=a[i]*x^ 2+b[i]*x+c[i];
```

(%o38) $p\left(i\,,x\right):=a_i x^2+b_i x+c_i$ Parabole passant par (X1,Y1), (X2,Y2), (X3,Y3)

```
(% i39) coeff :solve([p(1,X[1])=Y2[1], p(1,X[2])=Y2[2] ,p(1,X[3])=Y2[3] ],[a[1],b[1],c[1]]);
```

(*coeff*) $\left[\left[a_1=2808820800\,,b_1=3105648\,,c_1=\frac{441}{25}\right]\right]$

```
(% i40) parabole1(x) :=rhs(coeff[1][1])*x^ 2+rhs(coeff[1][2])*x+rhs(coeff[1][3]);
```

(%o40) $parabole1(x):=rhs\left(\left(\text{coeff}_1\right)_1\right)x^2+rhs\left(\left(\text{coeff}_1\right)_2\right)x+rhs\left(\left(\text{coeff}_1\right)_3\right)$

```
(% i41) parabole1(x);
```

(%o41) $2808820800x^2+3105648x+\frac{441}{25}$

(% i42) aire1 :integrate(parabole1(x),x,0,2/1200);

(*aire*1) $\frac{43387}{5000}$ Parabole passant par (X3,Y3), (X4,Y4), (X5,Y5)

(% i43) coeff :solve([p(2,X[3])=Y2[3], p(2,X[4])=Y2[4] ,p(2,X[5])=Y2[5]],[a[2],b[2],c[2]]);

(*coeff*) $\left[\left[a_2 = -5593658400\,,\, b_2 = 22953258\,,\, c_2 = -\left(\frac{972149}{100}\right)\right]\right]$

(% i44) parabole2(x) :=rhs(coeff[1][1])*x^ 2+rhs(coeff[1][2])*x+rhs(coeff[1][3]);

(%*o*44) $parabole2(x) := rhs\left(\left(\text{coeff}_1\right)_1\right) x^2 + rhs\left(\left(\text{coeff}_1\right)_2\right) x + rhs\left(\left(\text{coeff}_1\right)_3\right)$

(% i45) aire2 :integrate(parabole2(x),x,2/1200,4/1200);

(*aire*2) $\frac{6843877}{360000}$ Automatisons ce calcul :

```
(% i46) for i :1 thru 5 do (
        i1 : 2*i - 1,   i2 : 2*i ,    i3 : 2*i + 1,
        coeff :solve([p(i,X[i1])=Y2[i1], p(i,X[i2])=Y2[i2] ,p(i,X[i3])=Y2[i3] ],[a[i],b[i],c[i]]),
        print( coefficients : ,coeff),
        parabole(i,x) :=rhs(coeff[1][1])*x^ 2+rhs(coeff[1][2])*x+rhs(coeff[1][3]),
        p[i] :parabole(i,x),
        print( parabole d'interpolation : p ,i, (x)= ,parabole(i,x)),
        aire[i] :integrate(parabole(i,x),x,2*(i-1)/1200,2*i/1200),
        print( aire sous la parabole : aire ,i, = ,aire[i])
        );
```

coefficients : $\left[\left[a_1 = 2808820800\,,\, b_1 = 3105648\,,\, c_1 = \frac{441}{25}\right]\right]$

parabole d'interpolation : p 1 (x)= $2808820800x^2 + 3105648x + \frac{441}{25}$

aire sous la parabole : aire 1 = $\frac{43387}{5000}$

coefficients : $\left[\left[a_2 = -5593658400\,,\, b_2 = 22953258\,,\, c_2 = -\left(\frac{972149}{100}\right)\right]\right]$

parabole d'interpolation : p 2 (x)= $-\left(5593658400x^2\right) + 22953258x - \frac{972149}{100}$

aire sous la parabole : aire 2 = $\frac{6843877}{360000}$

coefficients : $\left[\left[a_3 = 6805936800\,,\, b_3 = -56608890\,,\, c_3 = \frac{11771239}{100}\right]\right]$

parabole d'interpolation : p 3 (x)= $6805936800x^2 - 56608890x + \frac{11771239}{100}$

aire sous la parabole : aire 3 = $\frac{945653}{360000}$

coefficients : $\left[\left[a_4 = -4805143200\,,\, b_4 = 60981474\,,\, c_4 = -\left(\frac{17996243}{100}\right)\right]\right]$

parabole d'interpolation : p 4 (x)= $-\left(4805143200x^2\right) + 60981474x - \frac{17996243}{100}$

aire sous la parabole : aire 4 = $\frac{6685313}{360000}$

coefficients : $\left[\left[a_5 = 2592424800\,,\, b_5 = -46684962\,,\, c_5 = \frac{20903301}{100}\right]\right]$

parabole d'interpolation : p 5 (x)= $2592424800x^2 - 46684962x + \frac{20903301}{100}$

aire sous la parabole : aire 5 = $\frac{3191873}{360000}$

(%*o*46) done Aire approxlmée avec la méthode de Simpson et 5 paraboles

(% i47) sum(aire[i],i,1,5);

(%o47) $\frac{1039529}{18000}$

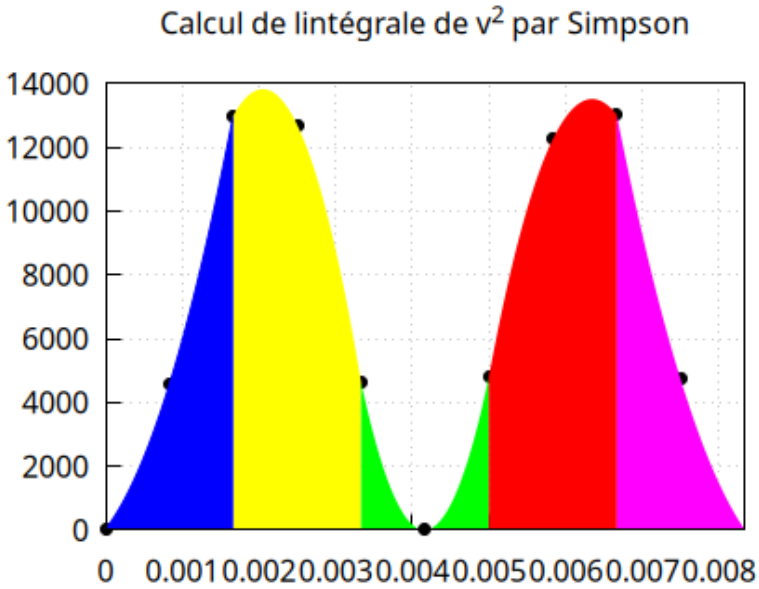
(% i48)
```
float(%);
```
(%o48) 57.75161111111111

(% i49)
```
parabole(4,x);
```
(%o49) $2592424800x^2 - 46684962x + \frac{20903301}{100}$ Autre méthode pour cette automatisation

(% i57)
```
/* données */
X : makelist(k/1200, k, 0, 10)$
Y : [4.2, 67.5, 114.0, 112.7, 68.1, -0.8, -69.4, -110.7, 114.1, -68.7, -4.6]$
Y2 : map(lambda([y], y^ 2), Y)$

/* polynôme général */
p(x) := a*x^ 2 + b*x + c$

/* liste des aires */
aires : []$

for i :1 thru 5 do (
i1 : 2*i - 1,
i2 : 2*i,
i3 : 2*i + 1,

coeff : solve(
[p(X[i1]) = Y2[i1],
p(X[i2]) = Y2[i2],
p(X[i3]) = Y2[i3]],
[a,b,c]
),

pi(x) := ev(subst(first(coeff), p(x))),
ai : integrate(pi(x), x, X[i1], X[i3]),
aires : endcons(ai, aires),

print( Parabole , i,  : , pi(x)),
print( Aire , i,  : , ev(ai, numer))
)$

approx_simpson : apply( + , aires)$
float(approx_simpson);
```
Parabole 1 : $2808820800x^2 + 3105648x + \frac{441}{25}$
Aire 1 : 8.6774
Parabole 2 : $-\left(5593658400x^2\right) + 22953258x - \frac{972149}{100}$
Aire 2 : 19.010769444444445
Parabole 3 : $6805936800x^2 - 56608890x + \frac{11771239}{100}$
Aire 3 : 2.626813888888889
Parabole 4 : $-\left(4805143200x^2\right) + 60981474x - \frac{17996243}{100}$
Aire 4 : 18.57031388888889
Parabole 5 : $2592424800x^2 - 46684962x + \frac{20903301}{100}$
Aire 5 : 8.86631388888889

(%o57) 57.75161111111111

(% i58)
```
wxdraw2d(
grid=true, xaxis=true, yaxis=true,
xrange=[0,10/1200], yrange=[0,14000],
title= Calcul de lintégrale de v^2 par Simpson ,

/* Points */
color=black, point_type=7, points(POINTS),

/* Parabole 1 - BLEU */
color=blue, fill_color=blue,
explicit(p[1], x, 0, 2/1200),
filled_func=true,
explicit(0, x, 0, 2/1200),

/* Parabole 2 - JAUNE */
color=yellow, fill_color=yellow,
explicit(p[2], x, 2/1200, 4/1200),
filled_func=true,
explicit(0, x, 2/1200, 4/1200),

color=green, fill_color=green,
explicit(p[3], x, 4/1200, 6/1200),
filled_func=true,
explicit(0, x, 4/1200, 6/1200),

color=red, fill_color=red,
explicit(p[4], x, 6/1200, 8/1200),
filled_func=true,
explicit(0, x, 6/1200, 8/1200),

color=magenta, fill_color=magenta,
explicit(p[5], x, 8/1200, 10/1200),
filled_func=true,
explicit(0, x, 8/1200, 10/1200),

color=blue, fill_color=blue,
explicit(p[1], x, 0/1200, 2/1200),
filled_func=true,
explicit(0, x, 0/1200, 2/1200)
);
```



(%$t58$)

(%$o58$)

6 Exercice 6

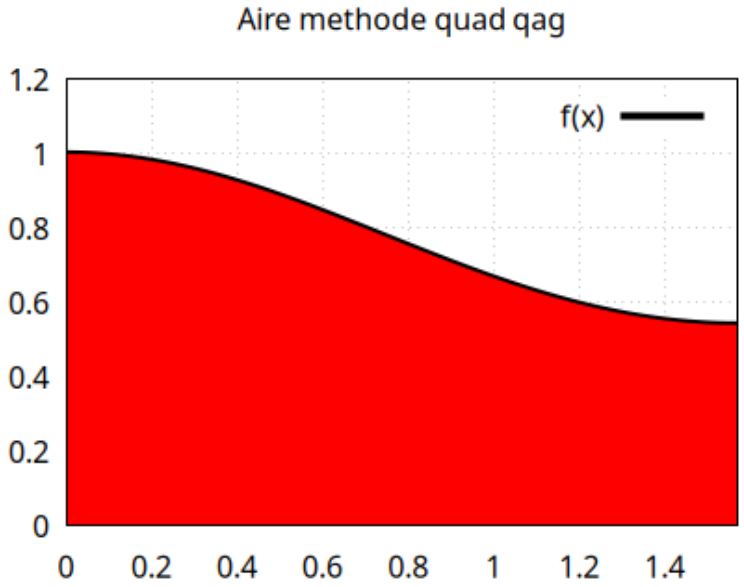
(% i59) kill(all);
(%$o0$) done

(% i1) f(x) :=cos(sin(x));
(%$o1$) $f(x) := \cos(\sin(x))$

(% i2) makelist(quad_qag(f(x),x,0,%pi/2,i),i,1,6);
(%$o2$) $[[1.2019697153172066, 1.471841510586141710^{-12}, 15, 0]$
$, [1.2019697153172066, 1.334454452847307810^{-14}, 21, 0]$
$, [1.2019697153172066, 1.334454452847307810^{-14}, 31, 0]$
$, [1.2019697153172064, 1.334454452847307710^{-14}, 41, 0]$
$, [1.2019697153172064, 1.334454452847307710^{-14}, 51, 0]$
$, [1.2019697153172062, 1.334454452847307310^{-14}, 61, 0]]$

Les 6 méthodes donnent bien une approximation identique

```
(% i3)  wxdraw2d(
        title= Aire methode quad\_ qag ,
        grid=true,
        xaxis=true,
        yaxis=true,
        xrange=[0,%pi/2],
        yrange=[0,1.2],
        color=black,
        line_width=4,
        key= f(x) ,
        explicit(f(x),x,0,%pi/2),
        key= ,
        filled_func=true,
        explicit(f(x),x,0,%pi/2),
        explicit(0, x, 0,%pi/2)
        )
        ;
```



(%t3)

(%o3)

7 Exercice 7

Nous réutilisons les programmes de l'exercice 5. Pour n=10, on a 5 paraboles :

(% i4) X : makelist(k*%pi/20, k, 0, 10) ;

$$(X)\quad \left[0,\frac{\pi}{20},\frac{\pi}{10},\frac{3\pi}{20},\frac{\pi}{5},\frac{\pi}{4},\frac{3\pi}{10},\frac{7\pi}{20},\frac{2\pi}{5},\frac{9\pi}{20},\frac{\pi}{2}\right]$$

(% i5) Y2 :makelist(f(X[k+1]),k,0,10) ;

$$(Y2)\quad [1,\cos\left(\sin\left(\frac{\pi}{20}\right)\right),\cos\left(\sin\left(\frac{\pi}{10}\right)\right),\cos\left(\sin\left(\frac{3\pi}{20}\right)\right)$$

$$,\cos\left(\sin\left(\frac{\pi}{5}\right)\right),\cos\left(\frac{1}{\sqrt{2}}\right),\cos\left(\sin\left(\frac{3\pi}{10}\right)\right),\cos\left(\sin\left(\frac{7\pi}{20}\right)\right),\cos\left(\sin\left(\frac{2\pi}{5}\right)\right),\cos\left(\sin\left(\frac{9\pi}{20}\right)\right),\cos\left(1\right)]$$

(% i6) POINTS :makelist([X[i+1],Y2[i+1]],i,0,10)$

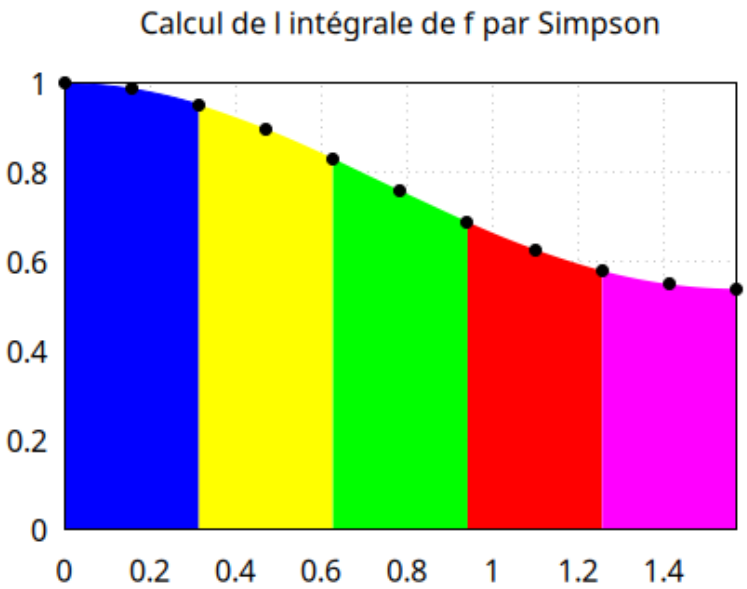
```
(% i7)   p(i,x) :=a[i]*x^ 2+b[i]*x+c[i];
```
$$(\%o7)\quad p\left(i\,,x\right):=a_i x^2+b_i x+c_i$$

```
(% i8)   for i :1 thru 5 do (
         i1 : 2*i - 1,   i2 : 2*i ,    i3 : 2*i + 1,
         coeff :float(solve([p(i,X[i1])=Y2[i1], p(i,X[i2])=Y2[i2] ,p(i,X[i3])=Y2[i3] ],[a[i],b[i],c[i]])),
         print( coefficients : ,coeff),
         parabole(i,x) :=rhs(coeff[1][1])*x^ 2+rhs(coeff[1][2])*x+rhs(coeff[1][3]),
         p[i] :parabole(i,x),
         print( parabole d'interpolation : p ,i, (x)= ,parabole(i,x)),
         aire[i] :float(integrate(parabole(i,x),x,2*(i-1)*%pi/20,2*i*%pi/20)),
         print( aire sous la parabole : aire ,i, = ,aire[i])
         )$
```
coefficients : $[[a_1=-0.46496572634767747\,,b_1=-0.004700603479215709\,,c_1=1.0]]$
parabole d'interpolation : p 1 (x)= $-\left(0.46496572634767747x^2\right)-0.004700603479215709x+1.0$
aire sous la parabole : aire 1 = 0.3091216812223839
coefficients : $[[a_2=-0.2554224340734059\,,b_2=-0.14271294265578047\,,c_2=1.0226767610896945]]$
parabole d'interpolation : p 2 (x)= $-\left(0.2554224340734059x^2\right)-0.14271294265578047x+1.0226767610896945$
aire sous la parabole : aire 2 = 0.28167627878350304
coefficients : $[[a_3=0.038334330564988604\,,b_3=-0.5120905636623693\,,c_3=1.1387930429156548]]$
parabole d'interpolation : p 3 (x)= $0.038334330564988604x^2-0.5120905636623693x+1.1387930429156548$
aire sous la parabole : aire 3 = 0.23893693451540132
coefficients : $[[a_4=0.2791261537115665\,,b_4=-0.9620191514744397\,,c_4=1.348953943376145]]$
parabole d'interpolation : p 4 (x)= $0.2791261537115665x^2-0.9620191514744397x+1.348953943376145$
aire sous la parabole : aire 4 = 0.19821102464962148
coefficients : $[[a_5=0.4029276761445312\,,b_5=-1.2682337439306708\,,c_5=1.5382550209025823]]$
parabole d'interpolation : p 5 (x)= $0.4029276761445312x^2-1.2682337439306708x+1.5382550209025823$
aire sous la parabole : aire 5 = 0.17402379614629204

```
(% i9)   sum(aire[i],i,1,5);
```
$$(\%o9)\quad 1.201969715317202$$

Le résultat obtenu est identique à celui obtenu par la méthode quad_qag de Maxima

```
(% i10)  wxdraw2d(
         grid=true, xaxis=true, yaxis=true,
         xrange=[0,%pi/2], yrange=[0,1],
         title= Calcul de l intégrale de f par Simpson ,

         /* Parabole 1 - BLEU */
         color=blue, fill_color=blue,
         explicit(p[1], x, 0, 2*%pi/20),
         filled_func=true,
         explicit(0, x, 0, 2*%pi/20),

         /* Parabole 2 - JAUNE */
         color=yellow, fill_color=yellow,
         explicit(p[2], x, 2*%pi/20, 4*%pi/20),
         filled_func=true,
         explicit(0, x, 2*%pi/20, 4*%pi/20),

         color=green, fill_color=green,
         explicit(p[3], x, 4*%pi/20, 6*%pi/20),
         filled_func=true,
         explicit(0, x, 4*%pi/20, 6*%pi/20),

         color=red, fill_color=red,
         explicit(p[4], x, 6*%pi/20, 8*%pi/20),
         filled_func=true,
         explicit(0, x, 6*%pi/20, 8*%pi/20),

         color=magenta, fill_color=magenta,
         explicit(p[5], x, 8*%pi/20, 10*%pi/20),
         filled_func=true,
         explicit(0, x, 8*%pi/20, 10*%pi/20),

         color=blue, fill_color=blue,
         explicit(p[1], x, 0, 2*%pi/20),
         filled_func=true,
         explicit(0, x, 0, 2*%pi/20),

         color=black, point_type=7, points(POINTS)
         );
```



(%t10)

(%o10)

8 Exercice 8

(% i11) kill(all)$

(% i1) f(x) :=cos(sin(x));
(%o1) $f(x) := \cos(\sin(x))$

(% i2) float(((%pi/2-0)/1000)*sum(f(random(float(%pi/2))),i,1,1000));
(%o2) 1.2083396193173022

Le résultat est bien cohérent avec les approximations précédentes.

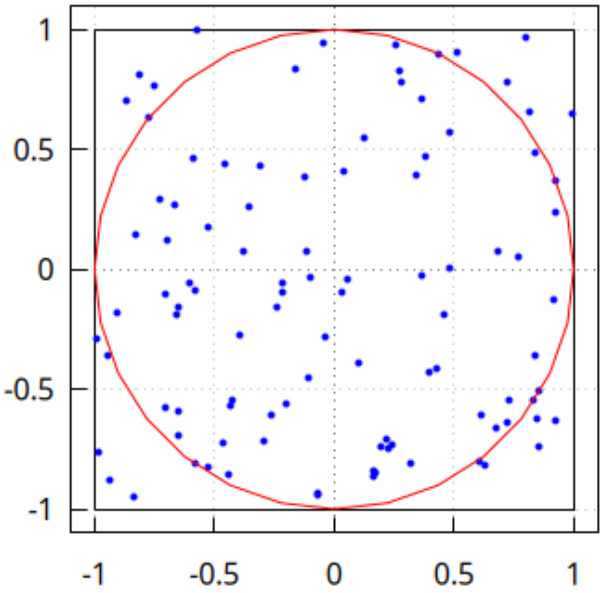
9 Exercice 9

Pour générer les points à l'intérieur du carré, on utilise random pour avoir l'abscisse et l'ordonnée entre -2 et 2

(% i4) X :makelist(2*random(1.0)-1, i, 1, 100)$Y :makelist(2*random(1.0)-1, i, 1, 100)$

(% i5) mcpts :makelist([X[i],Y[i]],i,1,100);
(La configuration indique de supprimer la sortie des cellules longues)

```
(% i6)  wxdraw2d(
        color=black,proportional_axes =xy,
        grid=true, xaxis=true, yaxis=true,
        xrange=[-1.1,1.1], yrange=[-1.1,1.1],
        points_joined = true,
        point_type = none,
        points([[-1,-1], [-1, 1]]),  points([[-1,1], [1,1]]),
        points([[1,1], [1,- 1]]),  points([[1,-1], [-1,- 1]]),
        points_joined = false,
        color=blue,
        point_type=7, point_size=0.5,
        points(mcpts),
        color=red,
        parametric(cos(t), sin(t), t, 0, 2*%pi)
        );
```



(%t6)

(%o6)

On compte le nombre de points à l'extérieur du cercle : 11, soit 89 à l'intérieur

```
(% i7)  π=float(89/100*4);
```
(%o7) $\pi = 3.56$

Lors d'une nouvelle évaluation des cellules, ce résultat se modifie, d'ou l'automatisation suivante :

10 Exercice 10

```
(% i8)  approx_pi_mc(n) :=block(
        [compteur,i ],
        compteur :0 ,
        for i :1 thru n do (
        pt :[2*random(1.0)-1,2*random(1.0)-1],
        print( point = ,pt, somme = ,pt[1]^ 2+pt[2]^ 2),
        if (pt[1]^ 2+pt[2]^ 2<1 or pt[1]^ 2+pt[2]^ 2=1 ) then compteur : compteur +1,
        print( compteur = ,compteur)
        ),
        print( compteur final = ,compteur),
        print( aproximation de pi = ,float(4*compteur/n))
        ) ;
```

(%o8) $approx\_pi\_mc(n) := block\ (\ [\text{compteur}\ ,i]\ ,\text{compteur}:0\ ,for\ i\ thru\ n\ do\ (\ \text{pt}:[2\,random\,(1.0)-1\ ,2\,rando$

```
(% i9)  approx_pi_mc(10)$
```

point = $[0.013665206491056292\ ,0.6296014587294336]$ somme = 0.3965847347026739
compteur = 1
point = $[0.9524934330915755\ ,-0.7212193429149409]$ somme = 1.4274010806772348
compteur = 1
point = $[0.6377381314154857\ ,-0.307407448710852]$ somme = 0.5012092637842305
compteur = 2
point = $[-0.6406764177364432\ ,0.1946162661711428]$ somme = 0.4483417633019986
compteur = 3
point = $[0.8420242103707163\ ,0.47636922287043415]$ somme = 0.9359324073486096
compteur = 4
point = $[0.9472839096824872\ ,0.3770917417763089]$ somme = 1.039544987259229
compteur = 4
point = $[0.04876396374139347\ ,0.2661621480530294]$ somme = 0.07322021321597467
compteur = 5
point = $[-0.44224965829681606\ ,-0.5177936186647836]$ somme = 0.4636949917936219
compteur = 6
point = $[0.4150640180082408\ ,0.21747012100538488]$ somme = 0.21957139257524194
compteur = 7
point = $[0.14162569303423655\ ,-0.7655330062144943]$ somme = 0.6060986205312288
compteur = 8
compteur final = 8
aproximation de pi = 3.2

On supprime les résultats intermédiaires pour n'afficher que la conclusion et passer à 10000

```
(% i10) approx_pi_mc2(n) :=block(
        [compteur,i ],
        compteur :0 ,
        for i :1 thru n do (
        pt :[2*random(1.0)-1,2*random(1.0)-1],
        if (pt[1]^ 2+pt[2]^ 2<1 or pt[1]^ 2+pt[2]^ 2=1 ) then compteur : compteur +1
        ),
        print( compteur final = ,compteur),
        print( aproximation de pi = ,float(4*compteur/n))
        ) ;
```

(%o10) $approx\_pi\_mc2(n) := block([compteur, i], compteur : 0, for\ i\ thru\ n\ do\ (pt : [2\,random(1.0) - 1, 2\,rand$

(% i11) approx_pi_mc2(10000)$

compteur final = 7810
aproximation de pi = 3.124

(% i12) approx_pi_mc2(10000)$

compteur final = 7806
aproximation de pi = 3.1224

(% i13) approx_pi_mc2(10000)$

compteur final = 7918
aproximation de pi = 3.1672

A priori, on constate que la première décimale est exacte, et on a une incertitude de 0,08

(% i14) approx_pi_mc2(100000)$

compteur final = 78402
aproximation de pi = 3.13608

(% i15) approx_pi_mc2(1000000)$

compteur final = 785282
aproximation de pi = 3.141128

(% i16) float(%pi);
(%o16) 3.141592653589793

Il faut donc augmenter fortement le nombre de points pour avoir une bonne approximation.