wxMaxima pour l'analyse en mathématiques
Volume I

Zachary Hannan
Solano Community College

Traduction française : Michel Gosse

Ce document est sous licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International. Pour consulter une copie de cette licence, consulter http://creativecommons.org/licenses/by-nc-sa/4.0/.

La licence CC-BY-NC-SA permet à toute personne de modifier et/ou de redistribuer ce document tant que l'auteur original et tous les auteurs en relation sont cités. Ce document et ses productions dérivées ne doivent pas être utilisés à des fins commerciales sauf avec l'autorisation de l'auteur original (le détenteur des droits d'auteur), et toutes les œuvres dérivées doivent utiliser la même licence. Si vous souhaitez créer un document dérivé, les fichiers .tex sont disponibles ici : https://wxmaximafor.wordpress.com/. Je vous serais reconnaissant de me contacter à zhannan@solano.edu si vous décidez de créer une œuvre dérivée.

Table des matières

Préface **v**

0 Introduction à wxMaxima **1**
0.1 Commandes de base . . . 2
0.1.1 Arithmétique . . . 2
0.1.2 Calcul algébrique . . . 3
0.1.3 Trigonometrie . . . 4
0.2 Expressions et fonctions . . . 6
0.3 Graphiques 2D et 3D . . . 8
0.4 Définir et résoudre des équations . . . 11
0.5 Suites et Sommes . . . 14
0.5.1 Suites . . . 14
0.5.2 Sommes . . . 17
0.6 Application : Droite passant par deux points . . . 18
0.7 Exercices du Chapitre 0 . . . 20

1 Panorama sur les fonctions **21**
1.1 Fonctions polynômes et rationnelles . . . 22
1.1.1 Fonctions polynômes . . . 22
1.1.2 Fonctions rationnelles . . . 23
1.2 Fonctions trigonométriques . . . 26
1.2.1 Les fonctions trigonométriques élémentaires . . . 26
1.2.2 Graphes des fonctions sinusoïdales . . . 28
1.3 Les fonctions exponentielles . . . 33
1.4 Transformation des fonctions . . . 35
1.5 Parité des fonctions . . . 38
1.6 Combinaisons algébriques de fonctions . . . 40
1.7 Composition de fonctions . . . 45
1.8 Fonctions inverses . . . 47
1.9 Exercices du Chapitre 1 . . . 53

2 Limites **55**
2.1 Utiliser les suites pour approcher les limites . . . 56
2.2 Commandes de wxMaxima pour les limites . . . 62
2.3 Explorer la définition formelle de la limite . . . 67
2.4 Exercices du chapitre 2 . . . 72

3 Introduction aux dérivées **73**
3.1 Le problème de la tangente . . . 74
3.1.1 La tangente à partir d'une suite de droites . . . 74
3.1.2 La tangente comme une limite . . . 76
3.2 Les commandes pour la dérivation dans wxMaxima . . . 78
3.3 Produits, Quotients et Combinaisons Linéaires . . . 81
3.4 Dérivation des fonctions composées . . . 83
3.4.1 La règle de la Chaine . . . 83
3.4.2 Différentiation logarithmique . . . 84
3.4.3 Différentiation implicite . . . 85
3.5 Exercices du Chapitre 3 . . . 87

4 Applications de la dérivation **90**
4.1 Croissance, Décroissance et Extremum local . . . 91
4.2 Concavité et Inflexion . . . 96
4.3 Problèmes d'optimisation . . . 98
4.4 La méthode de Newton . . . 101
4.5 Exercices du Chapitre 4 . . . 107

5 Le problème de l'Aire **110**
5.1 Sommes de Riemann : Gauche, Droite, Milieu . . . 111
5.2 L'intégrale définie . . . 115
5.3 Les commandes wxMaxima pour les intégrales définies . . . 119
5.4 Approximer une aire à partir d'une liste de données . . . 124
5.5 Exercices du Chapitre 5 . . . 127

6 Primitives et théorème fondamental **129**
6.1 Primitives . . . 130
6.2 Le théorème fondamental en Analyse . . . 133
6.2.1 Les fonctions d'aire . . . 133
6.2.2 Le théorème fondamental . . . 134
6.3 Calcul des intégrales . . . 136
6.3.1 Primitives de base . . . 136
6.3.2 Changement de variable . . . 137
6.4 Intégrales de surface . . . 139
6.4.1 Intégration de surface et intégration physique . . . 139
6.4.2 Aire comprise entre deux fonctions . . . 141
6.4.3 Valeur moyenne . . . 146
6.5 Exercices du Chapitre 6 . . . 148

Préface

Logiciel de calcul formel :

Un système de calcul formel (CAS) est un ensemble de logiciels conçu principalement pour effectuer des calculs symboliques. Un CAS peut effectuer presque tous les calculs symboliques qu'une personne pourrait faire « à la main », mais il est beaucoup plus rapide, plus précis et capable de gérer une complexité plus grande. Les calculs complexes peuvent être divisés en parties gérables grâce à l'utilisation de fonctions, et les modélisations peuvent être explorées en modifiant rapidement leurs paramètres. En plus des calculs symboliques, un CAS peut produire des graphiques de qualité, faire des approximations numériques de divers types et exécuter des algorithmes pour résoudre des problèmes qui ne peuvent être résolus symboliquement.

wxMaxima :

wxMaxima est une interface utilisateur pour le système de calcul formel Maxima. Cette interface permet à l'utilisateur de créer, éditer et sauvegarder un document (un fichier .wxm ou .wxmx selon les cas) contenant de nombreux calculs et graphiques, et *la plupart* des opérations peuvent être accessibles via l'interface graphique, si souhaité. Maxima et wxMaxima sont des projets open source, ce qui signifie qu'ils seront toujours gratuits et qu'ils s'améliorent constamment grâce au travail bénévole de leurs nombreux passionnés.

La dernière version de wxMaxima pour les ordinateurs Windows et Mac peut être téléchargée ici : http://andrejv.github.io/wxmaxima/.
Lorsque vous cliquez sur le lien de téléchargement pour votre système d'exploitation, vous serez redirigé vers une page de sourceforge.net qui téléchargera automatiquement Maxima, wxMaxima, GNUplot et tout autre programme auxiliaire nécessaire pour que wxMaxima fonctionne sur votre machine. L'installation sur un ordinateur Windows prend généralement environ 5 minutes.

Versions des logiciels

Le texte du document original en anglais a été rédigé avec wxMaxima version 12.04.0 et Maxima version 5.27.0. L'utilisation de versions plus récentes du logiciel ne devrait pas poser de problèmes. Cependant, des bogues peuvent parfois survenir ; c'est pourquoi je recommande que l'enseignant et ses étudiants utilisent tous la même version, au cas où un dépannage serait nécessaire.
Lors de la traduction en français, une actualisation du document a été faite en utilisant la version 20.12 de wxMaxima et Maxima version 5.45.

Autres manuels de la série “wxMaxima pour”

J’ai publié deux livres dans la série “wxMaxima pour” :

“wxMaxima for Calculus I”	June 2015
“wxMaxima for Calculus II”	June 2015

avec le projet de réaliser des documents similaires pour l’algèbre linéaire, les équations différentielles et le calcul à plusieurs variables lors des prochaines années.

Ces livres peuvent être obtenus à l’adresse https://wxmaximafor.wordpress.com/. Les documents sont disponibles sous forme de fichiers pdf gratuits à télécharger ou bien en ”impression à la demande” pour un coût abordable. Les versions françaises de ces livres sont téléchargeables sur le site https://maxima-french-doc.fr/ à la rubrique Documentation.

Ces manuels s’adressent principalement aux étudiants de premier cycle qui suivent des cours de mathématiques destinés aux filières d’ingénieurs, de sciences physiques et de mathématiques appliquées. Les universités attendent de plus en plus que ces étudiants maîtrisent les logiciels mathématiques dès le début de leurs cycles de formation supérieure, et de nombreuses institutions proposent actuellement des laboratoires de mathématiques pour répondre à ce besoin. Chaque texte de la série **≪ wxMaxima pour ≫** peut servir de manuel de référence pour un cours de première année de licence, ou de ressource précieuse ≪ par l’exemple ≫ pour les étudiants apprenant l’informatique de manière autonome.

En se basant seulement sur une expérience minimale avec les ordinateurs (comme être à l’aise avec un système d’exploitation tel que Windows ou Mac OS), chaque texte introduit progressivement l’utilisation de l’informatique scientifique à l’aide d’exemples pertinents pour le cours de mathématiques suivi en parallèle. Les principaux points théoriques de chaque cours sont passés en revue de manière concise, et les commandes sont introduites au fur et à mesure des besoins. Les exemples servent à motiver et renforcer les concepts mathématiques importants tout en illustrant leurs applications dans le contexte de l’informatique. Les commandes textes du logiciel sont utilisées exclusivement pour deux raisons : d’une part, elles sont plus puissantes et flexibles ; d’autre part, se familiariser avec les commandes textes garantit que les compétences acquises ici seront facilement transposables à d’autres logiciels.

Mise en page du document

Chaque texte est divisé en 7 modules, chacun étant composé de plusieurs sections et sous-sections. Chaque sous-section commence généralement par une brève discussion théorique, suivie de plusieurs exemples résolus dans wxMaxima. Chaque module se termine par un court ensemble d’exercices progressant du niveau basique au niveau avancé. Les sections et exercices particulièrement difficiles sont marqués d’un astérisque.

Ce texte n’est pas conçu pour être un manuel de référence encyclopédique, mais chaque module contient une liste de **≪ Commandes clés ≫** sur la page de titre afin de faciliter la recherche d’un exemple utilisant une commande spécifique.

Pour l'étudiant

Pour les étudiants ayant très peu d'expérience avec les ordinateurs, les deux premiers modules progresseront très lentement. Faire des erreurs et déboguer vos commandes fait naturellement partie de l'apprentissage de la syntaxe d'un nouveau programme. Bien que wxMaxima tente de vous aider en cas d'erreur, votre aide la plus précieuse reste la recherche sur Internet. En recherchant un problème particulier sur Google, vous trouverez probablement une variété de forums en ligne où un problème similaire est abordé. Avec le temps, votre connaissance syntaxique de wxMaxima et votre capacité à effectuer des recherches efficaces s'amélioreront. Notez qu'il est judicieux d'inclure **≪ wxMaxima ≫** dans vos requêtes plutôt que **≪ Maxima ≫**, car ce dernier terme a de nombreuses significations autres que le logiciel de calcul formel !

Le manuel officiel de Maxima est disponible à l'adresse suivante :
https://maxima.sourceforge.io/docs/manual/maxima_toc.html, mais il convient de noter qu'il est destiné à un public ayant un haut niveau de connaissances en calcul scientifique informatique. Je consulte occasionnellement le manuel officiel, mais j'ai constaté que rechercher des exemples pertinents est la méthode la plus rapide pour apprendre.

Il est important de travailler les exemples par vous-même, qu'ils soient ou non proposés par votre enseignant. En tapant vous-même les commandes, vous ferez inévitablement des erreurs de syntaxe qu'il faudra corriger. Corriger votre syntaxe sur un exemple résolu est une excellente préparation pour réussir les exercices de manière autonome.

Pour l'enseignant

Ce contenu peut être intégré à votre cours à différents niveaux. Vous pouvez choisir de demander aux étudiants de simplement reproduire et tester tous les exemples du manuel, ou décider de ne proposer que certains exercices et laisser les étudiants utiliser le manuel comme référence en autonomie. Le niveau de difficulté peut être modéré ou très exigeant, en fonction de la quantité d'indications que vous incluez dans votre cours.

Je recommande que les étudiants soumettent leur travail par e-mail sous forme de fichier wxMaxima bien formaté (.wxmx) avec un en-tête clair et les exemples et/ou exercices clairement étiquetés. Les étudiants peuvent insérer des lignes de texte dans leur feuille de travail en sélectionnant **Cell / Insert Text Cell** ou en appuyant sur **Ctrl+1**. Lorsque vous ouvrez la feuille de travail, les commandes devront être ré-exécutées, et cela constitue un moyen rapide de vérifier que tout le code fonctionne et que les solutions attendues sont obtenues. Les étudiants ont la responsabilité de déboguer leur travail jusqu'à ce que leur code fonctionne sans erreur.

Tout commentaire ou suggestion sur ce texte sera grandement apprécié et pris en compte pour les futures éditions.

Merci beaucoup,

Zak Hannan
Enseignant de Mathématiques et de Physique
Solano Community College, Fairfield, CA
zhannan@solano.edu

Traduction du document : Michel Gosse (michel.gosse@free.fr)
En complément du document original, les corrigés des exercices sont proposés dans le fichier correction-volI.pdf, accompagnés des fichiers wxMaxima associés. Le tout est disponible sur https://maxima-french-doc.fr/ à la rubrique Documentation.

Chapitre 0

Introduction à wxMaxima

0.1 Commandes de base **2**
0.1.1 Arithmétique 2
0.1.2 Calcul algébrique 3
0.1.3 Trigonometrie 4
0.2 Expressions et fonctions **6**
0.3 Graphiques 2D et 3D **8**
0.4 Définir et résoudre des équations **11**
0.5 Suites et Sommes **14**
0.5.1 Suites 14
0.5.2 Sommes 17
0.6 Application : Droite passant par deux points **18**
0.7 Exercices du Chapitre 0 . . . . . **20**

Commandes essentielles de ce Chapitre

`float`	`trigsimp`	`kill(all)`	`parametric`	`rhs`
`ratsimp`	`trigreduce`	`wxdraw2d`	`dimensions`	`makelist`
`fullratsimp`	`trigexpand`	`wxdraw3d`	`solve`	`for-do`
`expand`	`subst`	`explicit`	`find_root`	`sum`
`factor`	`sublis`	`implicit`	`lhs`	`simpsum`

0.1 Commandes de base

0.1.1 Arithmétique

wxMaxima utilise `+,-,*,/,^,sqrt,log` pour l'addition, la soustraction, la multiplication, la division, les puissances, la racine carrée et le logarithme népérien. On utilise `%` pour rappeler le résultat précédent et `float` pour obtenir une valeur approchée décimale. Pour obtenir les résultats de chaque commande, nous terminons la ligne (ou cellule) d'entrée avec `;` et appuyons simultanément sur les touches `shift+enter`. Si nous souhaitons masquer le résultat du calcul, nous devons terminer la ligne d'entrée par `$` à la place de `;`.

Exemple 0.1.1. Effectuer les opérations numériques suivantes :

1. Calculer $3 \cdot 2 + 5$.
2. Ajouter $\sqrt{2}$ au résultat précédent.
3. Trouver une valeur approchée décimale du résultat précédent.
4. Elever au carré le résultat précédent.

```
(%i1) 3*2+5;
(%o1) 11
(%i2) %+sqrt(2);
(%o2) sqrt(2)+11
(%i3) float(%);
(%o3) 12.4142135623731
(%i4) %^2;
(%o4) 154.1126983722081
```

Vous verrez que l'expression obtenue est de temps en temps "mieux affichée" que le résultat présenté dans ce document ; par exemple, `sqrt(2)+11` peut s'afficher sous la forme $\sqrt{2} + 11$. (Le menu Maxima ▶ Basculer en affichage 2d de wxMaxima gère ces choix d'affichage des résultats). Nous utiliserons le format enrichi seulement quand cela sera nécessaire pour une meilleure compréhension.

Exemple 0.1.2. Additionner $\frac{5}{6}$ and $\frac{7}{15}$ et trouver la forme simplifiée du résultat, puis donner une valeur approchée décimale de cette réponse.

```
(%i5) (5/6)+(7/15);
(%o5) 13/10
(%i6) float(%);
(%o6) 1.3
```

Remarquer que wxMaxima exprime automatiquement la fraction exacte sous forme réduite.

Exemple 0.1.3. wxMaxima utilise le symbole `%e` pour la constante d'Euler e. Trouver une approximation décimale de e et vérifier que $\ln e = 1$.

```
(%i7) float(%e);
(%o7) 2.718281828459045
(%i8) log(%e);
(%o8) 1
```

0.1.2 Calcul algébrique

wxMaxima est capable de réaliser les opérations algébriques basiques sur des expressions contenant des variables tout comme des nombres : rassembler des termes identiques, développer et factoriser, additionner/soustraire/diviser/réduire des expressions rationnelles, etc. Dans quelques cas, la simplification automatique est complète, et dans d'autres cas nous avons à imposer une simplification avec la commande `ratsimp` ou `fullratsimp` (cette dernière commande applique simplement `ratsimp` de manière récursive). Notons que une entrée comme `5x` provoquera une erreur – nous devons écrire la multiplication de manière explicite en écrivant `5*x`.

Exemple 0.1.4. Réaliser les opérations algébriques suivantes :

1. Simplifier : $(3a + b) + 2(2a - b)$

   On saisit l'expression et on applique `ratsimp` pour rassembler les termes :

```
(%i9) (3*a+b)+2*(2*a-b);
(%o9) b+2*(2*a-b)+3*a
(%i10) ratsimp(%);
(%o10) 7*a-b
```

2. Développer $(a + b)^3$

   `expand` développe l'expression :

```
(%i11) (a+b)^3;
(%o11) (b+a)^3
(%i12) expand(%);
(%o12) b^3+3*a*b^2+3*a^2*b+a^3
```

3. Factoriser : $x^2 - 8x + 12$

   On applique simplement `factor` :

```
(%i13) factor(x^2-8*x+12);
(%o13) (x-6)*(x-2)
```

4. Faire la somme et exprimer sous forme réduite et factorisée : $\frac{5}{x^2-1} + \frac{x-2}{x^2+2x-3}$

   On additionne avec `ratsimp`, puis on exprime la réponse sous forme factorisée et réduite avec `factor` :

```
(%i14) 5/(x^2-1)+(x-2)/(x^2+2*x-3);
(%o14) (x-2)/(x^2+2*x-3)+5/(x^2-1)
(%i15) ratsimp(%);
(%o15) (x^2+4*x+13)/(x^3+3*x^2-x-3)
(%i16) factor(%);
(%o16) (x^2+4*x+13)/((x-1)*(x+1)*(x+3))
```

NDT : on obtient en affichage 2d le résultat suivant :

$$\frac{x^2+4\,x+13}{(x-1)\,(x+1)\,(x+3)}$$

5. Réduire : $\frac{\frac{2x^2+x\,y}{y}}{\frac{x\,y+x}{y^2}}$

On saisit cette fraction de fractions, on la réduit en utilisant `ratsimp` et on la factorise avec `factor` :

```
(%i17) ((2*x^2+x*y)/y)/((x*y+x)/y^2);
(%o17) (y*(x*y+2*x^2))/(x*y+x)
(%i18) ratsimp(%);
(%o18) (y^2+2*x*y)/(y+1)
(%i19) factor(%);
(%o19) (y*(y+2*x))/(y+1)
```

NDT : résultat en affichage 2d : $\frac{y\,(y+2\,x)}{y+1}$

0.1.3 Trigonometrie

wxMaxima dispose de toutes les fonctions trigonométriques. Les angles doivent être exprimés en radians, aussi tout angle mesuré en degrés doit être converti en radians en le multipliant par $\frac{2\pi}{360}$. `%pi` est le symbole spécial pour le nombre π.

Exemple 0.1.5. Calculer $\sin 85°$ et $\tan \frac{11\pi}{12}$.

```
(%i20) 85*2*%pi/360;
(%o20) (17*%pi)/36
(%i21) sin(%);
(%o21) sin((17*%pi)/36)
(%i22) float(%);
(%o22) 0.99619469809175

(%i23) float(tan(11*%pi/12));
(%o23) -0.26794919243112
```

On se sert des commandes `trigsimp` pour simplifier le résultat à l'aide des égalités $\cos^2 x+\sin^2 x=1$ et $\cosh^2 x+\sinh^2 x=1$, `trigreduce` pour linéariser des expressions trigonométriques et `trigexpand` pour développer des fonctions trigonométriques de sommes d'angles.

Exemple 0.1.6. Aperçu des manipulations trigonométriques.

1. Appliquez `trigreduce` à $\sin^2 x+\cos^2 x$. Qu'observe-t-on ? Essayez d'utiliser `trigsimp` à la place.

```
(%i24) trigreduce((sin(x))^2+(cos(x))^2);
(%o24) (cos(2*x)+1)/2+(1-cos(2*x))/2
```

`trigreduce` transforme l'égalité en une expression bien plus complexe – nous appliquons `trigsimp` à la place :

```
(%i25) trigsimp((cos(x))^2+(sin(x))^2);
(%o25) 1
```

2. Exprimer $\sin^2 x$ en fonction du "double de l'angle".
   Cette fois `trigreduce` est la commande adéquate :

```
(%i26) trigreduce((sin(x))^2);
(%o26) (1-cos(2*x))/2
```

3. Trouver une expression pour $\cos(x+y)$ en fonction de $\sin x$ et $\cos x$.
   `trigexpand` va simplifier l'argument de la fonction cosinus :

```
(%i27) trigexpand(cos(x+y));
(%o27) cos(x)*cos(y)-sin(x)*sin(y)
```

0.2 Expressions et fonctions

L'une des fonctionnalités puissantes d'un système de calcul formel est que nous pouvons attribuer un nom à une variété d'objets, puis les « appeler » plus tard en utilisant ce nom. Nous pouvons attribuer un nom à une expression en utilisant « `:` » et attribuer un nom à une fonction en utilisant « `:=` ». Les expressions peuvent être évaluées pour des valeurs spécifiques des variables en utilisant `subst` ou `sublis`, tandis que les fonctions sont évaluées en utilisant la notation fonctionnelle ordinaire. Les expressions et les fonctions présentent divers avantages et inconvénients qui se révéleront au fur et à mesure – souvent, nous pouvons choisir l'un ou l'autre de ces avantages ou inconvénients pour résoudre un problème.

Exemple 0.2.1. Affecter A à l'expression $2x+5$ et B à l'expression $6-x^4$, puis calculer $A+B$, $A-B$ and AB sous forme développée.

On affecte A et B sur deux lignes consécutives avant de les faire interpréter par Maxima avec `shift+enter`.

```
(%i1) A:2*x+5$
      B:6-x^4$

(%i3) A+B;
(%o3) -x^4+2*x+11

(%i4) A-B;
(%o4) x^4+2*x-1

(%i5) A*B;
(%o5) (2*x+5)*(6-x^4)
(%i6) expand(%);
(%o6) -2*x^5-5*x^4+12*x+30
```

Exemple 0.2.2. Utiliser `subst` pour évaluer A quand $x=-3$ puis quand $x=B$.

```
(%i7) subst(-3,x,A);
(%o7) -1

(%i8) subst(B,x,A);
(%o8) 2*(6-x^4)+5
```

Exemple 0.2.3. Définir $f(x)$ comme la fonction racine carrée. Calculer $f(4)$, $f(-4)$ et $f(A)$.

```
(%i9) f(x):=sqrt(x);
(%o9) f(x):=sqrt(x)

(%i10) f(4);
(%o10) 2
```

```
(%i11) f(-4);
(%o11) 2*%i

(%i12) f(A);
(%o12) sqrt(2*x+5)
```

Nous voyons que $\sqrt{-4}$ a pour valeur $2i$. Notons que le symbole spécial désignant l'unité imaginaire i est `%i`. wxMaxima n'est pas restreint uniquement aux nombres réels !

Exemple 0.2.4. Utiliser `sublis` pour évaluer $\frac{-b+\sqrt{b^2-4ac}}{2a}$ quand $a = 1$, $b = -3$ and $c = 5$.

```
(%i9) EXPN:(-b+sqrt(b^2-4*a*c))/(2*a)$

(%i10) sublis([a=1,b=-3,c=5],EXPN);
(%o10) (sqrt(11)*%i+3)/2
```

Les fonctions peuvent avoir un nombre quelconque de variables. Quelquefois, nous utilisons une variable comme un *paramètre* afin de générer une famille de fonctions :

Exemple 0.2.5. Définir $f_n(x) = \cos(n\pi x)$, puis expliciter $f_n(x)$ pour plusieurs valeurs de n.

```
(%i13) f(n,x):=cos(n*%pi*x)$
       f(1,x);
       f(2,x);
       f(3,x);
(%o14) cos(%pi*x)
(%o15) cos(2*%pi*x)
(%o16) cos(3*%pi*x)
```

0.3 Graphiques 2D et 3D

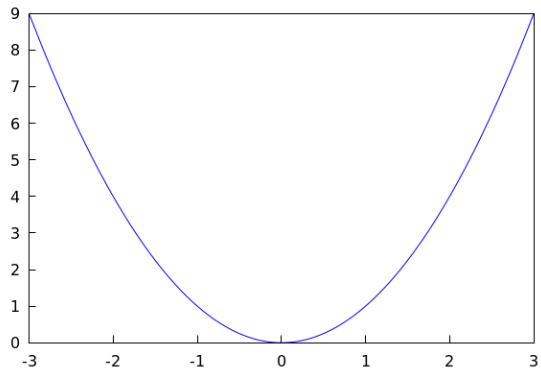
wxMaxima crée des graphiques en appelant un autre programme appelé GNUplot. Dans ce document, nous utilisons exclusivement les commandes `wxdraw2d` et `wxdraw3d` pour créer des graphiques 2D et 3D intégrés. Les commandes associées `draw2d` et `draw3d` créeront le même graphique dans une fenêtre contextuelle de GNUplot. La fenêtre contextuelle est utile pour manipuler des graphiques 3D (nous pouvons faire tourner l'image avec une souris), mais les graphiques intégrés sont plus pratiques pour imprimer notre travail. Dans les anciennes versions de wxMaxima, il peut être nécessaire de saisir `load(draw)$` avant d'utiliser les commandes `wxdraw2d` et `wxdraw3d`.

Les graphiques peuvent être modifiés avec une multitude de paramètres introduits progressivement dans le texte. Nous incluons un petit échantillon d'objets graphiques et de fonctionnalités de tracé ci-dessous.

Exemple 0.3.1. Définir $f(x) = x^2$, puis réaliser un tracé simple de $f(x)$ sur $[-3, 3]$.

Nous commençons cette section en appliquant `kill(all)` pour supprimer toutes les affectations que wxMaxima conserve actuellement en mémoire. En pratique, nous n'avons besoin d'utiliser `kill(all)` que si nos affectations précédentes provoquent une interférence avec un calcul. Nous utilisons `explicit` pour tracer la fonction, car $f(x)$ est donnée *explicitement* comme une fonction de x, puis le domaine est indiqué à côté de la fonction. Nous insérons plusieurs sauts de ligne dans `wxdraw2d` afin d'en faciliter la lecture :

```
(%i1) kill(all)$
(%i1) f(x):=x^2$
(%i2) wxdraw2d(
      explicit(f(x),x,-3,3)
      );
```



NDT : il existe une autre syntaxe pour tracer une courbe cartésienne 2D. Cette syntaxe sépare la ou les fonctions à tracer dans le premier crochet, et l'intervalle où varie x dans le deuxième crochet :

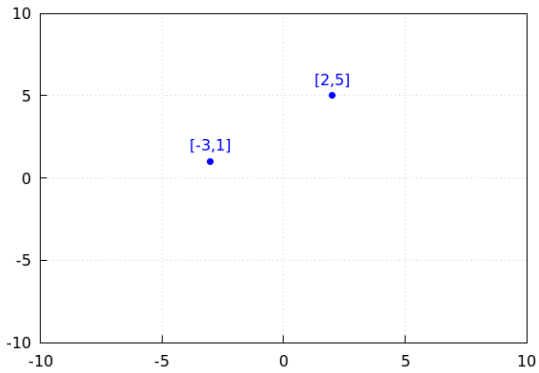
```
wxplot2d([f(x)], [x,-3,3])$
```

Il est aussi possible d'utiliser directement le menu Tracé de Courbe → Courbe 2d de wxMaxima.

Exemple 0.3.2. Dessiner les points $[-3, 1]$ et $[2, 5]$ sur un quadrillage pour x dans $[-10, 10]$ et y dans $[-10, 10]$. Rajouter une étiquette au-dessus de chaque point.

Nous utilison `point_type=7` afin de créer des disques et `points` pour lister les points désirés. `label` est utlisé pour créer les textes de chaque étiquette et les positionner auxs coordonnées voulues (dans ce cas, 1 unité au-dessus des points dessinés) :

```
(%i3) wxdraw2d(
      grid=true,
      xrange=[-10,10],
      yrange=[-10,10],
      point_type=7,
       points([[-3,1],[2,5]]),
      label(["[-3,1]",-3,2],["[2,5]",2,6])
      );
```

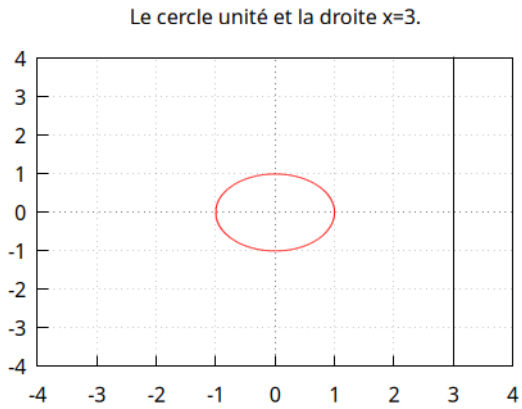


Exemple 0.3.3. Tracer une ligne verticale de couleur noire $x = 3$, et dessiner le cercle unité en rouge avec x variant sur $[-4, 4]$ et y variant sur $[-4, 4]$. Inclure les axes des x et y, un quadrillage et un titre.

Cet exemple illustre bien à quel point un graphique peut rapidement devenir complexe. Une ligne verticale n'est pas une fonction, elle doit donc être définie *paramétriquement*. Nous demandons à wxMaxima de tracer de nombreux points $(3, t)$ lorsque t varie de -4 à 4. De plus, l'équation du cercle unité définit une courbe *implicitement* : il est impossible d'exprimer y en fonction de x. Enfin, le cercle unité sera déformé si nous n'imposons pas un rapport d'aspect carré à l'aide de `dimensions` :

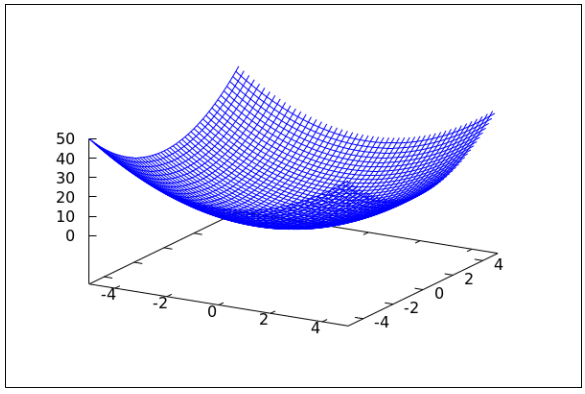
```
(%i4) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      dimensions=[600,600],
      xrange=[-4,4],
      yrange=[-4,4],
```

```
title="Le cercle unité et la droite x=3.",
color=black,
 parametric(3,t,t,-4,4),
color=red,
 implicit(x^2+y^2=1,x,-1,1,y,-1,1)
);
```



Exemple 0.3.4. Effectuer un tracé rapide du paraboloïde $z = x^2 + y^2$ en utilisant `wxdraw3d`. En complément, utiliser `draw3d` pour représenter ce paraboloïde dans une fenêtre GNUplot, puis manipuler cette courbe avec la souris.

```
(%i5) wxdraw3d(
      explicit(x^2+y^2,x,-5,5,y,-5,5)
      );
```



NDT : On peut aussi utiliser les menus de wxMaxima (Tracé de courbes → Courbe 3d pour obtenir ce résultat. wxMaxima génère alors la commande équivalente pour le tracé du paraboloïde :

```
wxplot3d(x^2+y^2, [x,-5,5], [y,-5,5],
 [gnuplot_pm3d,true])$
```

0.4 Définir et résoudre des équations

Dans wxMaxima, le symbole "=" est réservé à la définition des équations. Une fois qu'une équation est définie, nous pouvons utiliser `rhs` et `lhs` pour isoler les membres droit et gauche. De nombreuses équations et systèmes d'équations peuvent être résolus à l'aide de `solve`, mais certaines équations ne peuvent être résolues que numériquement en utilisant `find_root` ou grâce à une autre méthode d'approximation.

Exemple 0.4.1. Attribuer l'étiquette `EQN` à l'équation $3x - 6 = 6x + 5$, puis résoudre l'équation "à la main" en utilisant des opérations algébriques afin d'isoler x. Vérifier votre réponse en remplaçant cette valeur de x dans les membres droit et gauche de l'équation de départ. Enfin, vérifier à nouveau votre réponse en utilisant `solve`.

On effectue les opérations classiques pour résoudre les équations linéaires :

```
(%i1) EQN:3*x-6=6*x+5;
(%o1) 3*x-6=6*x+5
(%i2) %+6;
(%o2) 3*x=6*x+11
(%i3) %-6*x;
(%o3) -3*x=11
(%i4) %/-3;
(%o4) x=-11/3
```

On vérifie notre réponse en utilisant `subst` :

```
(%i5) subst(-11/3,x,rhs(EQN));
(%o5) -17
(%i6) subst(-11/3,x,lhs(EQN));
(%o6) -17
```

Finalement, nous résolvons à nouveau en utilisant `solve` :

```
(%i7) solve(EQN,x);
(%o7) [x=-11/3]
```

Exemple 0.4.2. Essayez de résoudre $\ln x = \sin x$ en utilisant `solve`. Que se passe-t-il ? Reformulez ensuite le problème en termes de recherche d'une *racine* d'une autre fonction. Approximez la solution en utilisant `find_root`.

Commençons par essayer la solution classique, en appelant l'équation `EQN2` :

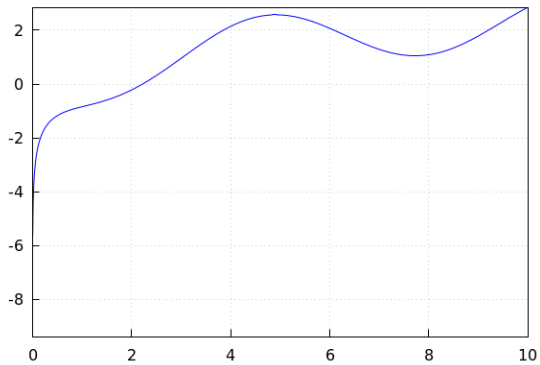
```
(%i8) EQN2:log(x)=sin(x)$
      solve(EQN2,x);
(%o9) [sin(x)=log(x)]
```

wxMaxima répète simplement la question, indiquant ainsi qu'il est en échec pour trouver la solution (en fait, `solve` ne peut résoudre que *certaines* équations polynomiales !). Cependant, nous réalisons que toute solution de $\ln x = \sin x$ est également une solution de

$\ln x - \sin x = 0$. Ainsi, nous pouvons étudier la fonction $f(x) = \ln x - \sin x$ et approcher numériquement ses racines.

Une complication de `find_root` est que nous devons indiquer l'intervalle dans lequel la racine se trouve, et la fonction doit être définie sur l'intervalle que nous avons choisi. Nous pouvons trouver un intervalle en représentant rapidement la fonction :

```
(%i10) wxdraw2d(
       grid=true,
       explicit(log(x)-sin(x),x,0,10)
       );
```



Nous observons l'existence d'une racine quelque part sur $[2, 4]$.
Remarque : si nous choisissons l'intervalle $[0, 4]$, `find_root` échouera car $\ln x$ n'est pas définie en $x = 0$!

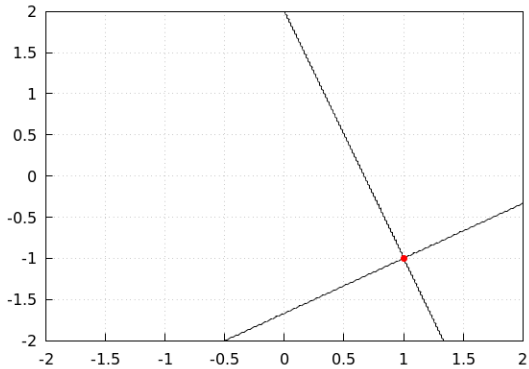
```
(%i11) find_root(log(x)-sin(x),2,4);
(%o11) 2.219107148913746
```

Nous obtenons $x \approx 2.219$ comme valeur approchée de la solution à l'équation.

Exemple 0.4.3. Résoudre le système d'équations $\begin{cases} 2x - 3y &= 5 \\ 3x + y &= 2 \end{cases}$ en utilisant `solve`.
Représentez les deux équations en définissant implicitement les fonctions correspondantes et faire apparaître le point d'intersection sur le graphique.

```
(%i12) L1:2*x-3*y=5$
       L2:3*x+y=2$
       solve([L1,L2],[x,y]);
(%o14) [[x=1,y=-1]]

(%i15) wxdraw2d(
       grid=true,
       color=black,
        implicit(L1,x,-2,2,y,-2,2),
        implicit(L2,x,-2,2,y,-2,2),
       color=red,
```

```
point_type=7,
 points([[1,-1]])
);
```



0.5 Suites et Sommes

0.5.1 Suites

Les suites trouvent une grande variété d'applications, et nous les utilisons fréquemment dans ce texte. wxMaxima génère des suites en utilisant `makelist` ou `for-do`. `makelist` offre l'avantage de permettre l'appel des éléments de la liste plus tard dans le calcul, tandis que `for-do` est beaucoup plus flexible et puissant.

Exemple 0.5.1. Utiliser `makelist` pour générer la suite $L = 1, 3, 5, \ldots 51$. Utiliser wxMaxima pour extraire le dixième élément de la suite.

On se sert de la formule $2n - 1$ avec $n = 1 \ldots 26$ pour générer la suite :

```
(%i1) L:makelist(2*n-1,n,1,26);
(%o1) [1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,35,
        37,39,41,43,45,47,49,51]
```

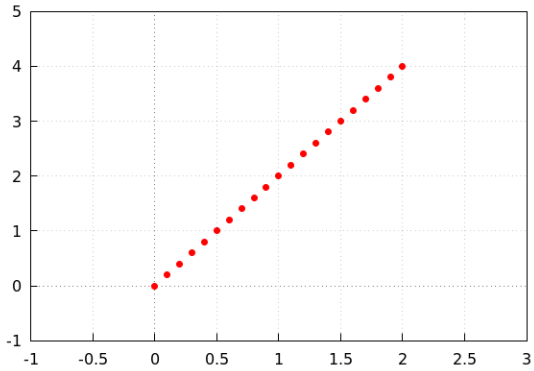
Maintenant, nous obtenons le dixième élément de `L` :

```
(%i2) L[10];
(%o2) 19
```

Exemple 0.5.2. Utiliser `makelist` pour générer une suite de 20 couples ordonnés situés sur la droite $y = 2x$ pour $x = 0.0, 0.1, \ldots, 2$. Transmettre cette liste de couples ordonnés à `wxdraw2d`.

Nous pouvons générer les valeurs de x en utilisant la suite $0.1k$ pour $k = 0, 1, \ldots, 20$. Le résultat de la commande `makelist` est déjà sous la forme d'une "liste", ce qui fait qu'il est déjà formaté pour être un argument de `wxdraw2d` :

```
(%i3) POINTS:makelist([0.1*k,2*(0.1*k)],k,0,20);
(%o3) [[0,0],[0.1,0.2],[0.2,0.4],[0.3,0.6],[0.4,0.8],[0.5,1.0],
      [0.6,1.2],[0.7,1.4],[0.8,1.6],[0.9,1.8],[1.0,2.0],[1.1,2.2],
      [1.2,2.4],[1.3,2.6],[1.4,2.8],[1.5,3.0],[1.6,3.2],[1.7,3.4],
      [1.8,3.6],[1.9,3.8],[2.0,4.0]]

(%i4) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[-1,3],
      yrange=[-1,5],
      point_type=7,
      color=red,
      points(POINTS)
      );
```



Exemple 0.5.3. Utiliser une boucle `for-do` pour générer la même suite que dans l'exemple 0.5.1.

Lorsqu'on programme une boucle `for-do` (également appelée simplement « do-loop »), on demande à wxMaxima de répéter un processus jusqu'à ce que le nombre final soit atteint. Dans ce cas, on demande à wxMaxima d'assigner à x la valeur $2n-1$ et d'afficher x, en répétant le calcul pour $n = 1 \ldots 26$:

```
(%i5) (for n:1 thru 26 do
       (x: 2*n-1,
        print(x))
       );
       1
       3
       5
       7
       9
       11
       13
       15
       17
       19
       21
       23
       25
       27
       29
       31
       33
       35
       37
       39
       41
       43
       45
```

```
        47
        49
        51

(%o5)  done
```

Exemple 0.5.4. La suite de Fibonacci est définie par la formule de récurrence $f_n = f_{n-1} + f_{n-2}$; c'est-à-dire que chaque terme est obtenu en additionnant les deux précédents. Si l'on commence la suite par $0, 1, \ldots$, la suite entière est donnée par $0, 1, 1, 2, 3, 5, 8, \ldots$. Utilisez une boucle do-loop pour générer les vingt premiers termes de la suite de Fibonacci.

Notre utilisation de `for-do` est plus importante dans cet exemple : nous avons à répéter un calcul plusieurs fois et à utiliser le résultat obtenu à chaque étape pour calculer l'étape suivante. Nous définissons les deux termes initiaux f_{n-1} and f_{n-2} en premier, puis la boucle calcule le terme suivant de la suite de Fibonnaci X, remplace par le terme d'ordre $(n-1)$ le terme d'ordre $(n-2)$ et assigne le terme d'ordre $(n-1)$ à X. Enfin, le processus est répété 18 fois pour obtenir la totalité des 20 termes de la suite de Fibonacci.

```
(%i6) N_1:1$
      N_2:0$
(%i8)
      (for i:1 thru 18 do
       (X:N_2+N_1,
        print(X),
        N_2:N_1,
        N_1:X)
       );
       1
       2
       3
       5
       8
       13
       21
       34
       55
       89
       144
       233
       377
       610
       987
       1597
       2584
       4181

(%o8)  done
```

0.5.2 Sommes

wxMaxima calcule les sommes en utilisant `sum`. La syntaxe est très similaire à celle de la notation sigma :

Exemple 0.5.5. Utiliser `sum` pour calculer la somme $2 + 4 + 6 + \cdots + 50$.

Avec la notation sigma, la somme s'écrit $\sum_{n=1}^{25} 2n$, et les paramètres de `sum` se réfèrent simplement aux indices de cette notation :

```
(%i9) sum(2*n,n,1,25);
(%o9) 650
```

Exemple 0.5.6. Trouver une formule algébrique donnant la somme $1 + 2 + \cdots + n$, puis utiliser la substitution pour obtenir la somme des 100 premiers entiers naturels.

En utilisant la notation sigma, nous souhaitons calculer $\sum_{k=1}^{n} k$. La formule classique est obtenue en utilisant `sum` suivi de `simpsum`, puis nous effectuons la substitution $n = 100$:

```
(%i10) sum(k,k,1,n),simpsum;
(%o10) (n^2+n)/2
(%i11) subst(100,n,%);
(%o11) 5050
```

0.6 Application : Droite passant par deux points

Pour démontrer l'utilité du calcul symbolique, nous créons une fonction pour représenter rapidement la droite reliant deux points.

Exemple 0.6.1. Créer une fonction 'LINE(a,b,c,d,x)' qui calcule l'équation d'une droite passant par les points (a, b) et (c, d). Complète ce code de manière à ce que l'affectation simple de a, b, c, d produise directement un tracé de qualité de la droite ainsi que des deux points donnés. Appliquer le code aux points $(-0.51, -3.47)$ et $(7.12, 3.94)$.

En premier, nous définissons chaque point comme une fonction de deux variables. Le résultat de chaque fonction est bienn sous une forme adéquate pour être interprété comme un point par `wxdraw2d`. Puis nous calculons la pente entre les points comme une fonction des quatre variables précédemment définies :

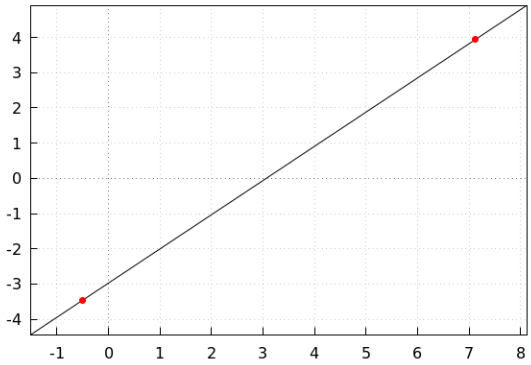
```
(%i1) POINT1(a,b):=[a,b]$
      POINT2(c,d):=[c,d]$
      PENTE(a,b,c,d):=(d-b)/(c-a)$
```

L'étape suivante consiste à remplacer dans la formule de la pente d'une droite $y - y_0 = m(x - x_0)$ et à exprimer y comme fontion de x. Nous utilisons `POINT1` pour (x_0, y_0).

```
(%i4) LINE(a,b,c,d,x):=PENTE(a,b,c,d)*(x-a)+b$
```

Finalement, nous assignons les valeurs pour a, b, c, d et paramétrons `wxdraw2d`. Notez que toutes les commandes restent générales à l'intérieur de `wxdraw2d`, ainsi nous pouvons dessiner la droite entre deux points quelconques de manière immédiate en assignant de nouvelles valeurs à a, b, c, d :

```
(%i5) a:-.51$
      b:-3.47$
      c:7.12$
      d:3.94$

      wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      color=black,
       explicit(LINE(a,b,c,d,x),x,a-1,c+1),
      color=red,
      point_type=7,
      points([[a,b],[c,d]])
      );
```


4
3
2
1
0
-1
-2
-3
-4
-1
0
1
2
3
4
5
6
7
8

0.7 Exercices du Chapitre 0

1. Définir les variables $A = \sqrt{3}$ et $B = 5$, puis trouver une approximation numérique pour $A + B$ et A/B.
2. Définir les expressions $A = x^2$ and $B = e^x$. Dans l'expression de A, remplacer x par B, puis évaluer l'expression obtenue pour $x = 0.1$ et en donner une approximation décimale.
3. Utiliser `trigexpand` pour trouver une formule donnant $\sin(x + y)$ en fonction de $\sin x$, $\sin y$, $\cos x$ et $\cos y$. Utiliser ce résultat pour calculer $\sin \frac{5\pi}{12}$ en se servant du fait que $\frac{5\pi}{12} = \frac{\pi}{6} + \frac{\pi}{4}$. Re-calculer $\sin \frac{5\pi}{12}$ directement et utiliser `float` afin de vérifier votre réponse.
4. Additionner et simplifier : $\frac{2x^2-x-6}{x^2-9} + \frac{x^3-x^2-4x+4}{x^2+5x+6}$. Exprimer la réponse sous forme factorisée.
5. Résoudre l'équation $ax^2 + bx + c = 0$ en x.
6. Représenter graphiquement $f(x) = \sin(\ln x) - 0.2$ sur $[10, 30]$ en incluant l'axe des abscisses. Utiliser `find_root` pour trouver une valeur approchée de la solution de $\sin(\ln x) = 0.2$ sur cet intervalle. Vérifier la réponse en calculant $\sin(\ln x)$ à la valeur de x trouvée.
7. Définir $f(x) = x + 2$ et $g(x) = x^2$. Trouver les intersections de ces deux courbes, puis représenter graphiquement les deux fonctions en dessinant les points d'intersection comme des disques. Nommer chaque point d'intersection en utilisant ses coordonnées décimales arrondies à la deuxième décimale.
8. Utiliser `makelist` et `for-do` pour générer les trente premiers termes de la suite $1, \frac{1}{2}, \frac{1}{4}, \ldots$.
9. La formule de récurrence d'une suite est donnée par $f_n = 2f_{n-1}$ avec comme premier terme $f_0 = 3$. Utilisez une boucle 'do' pour générer récursivement les 10 premiers termes de cette suite (comme dans l'exemple 0.5.4). Une fois la suite écrite, essayez de deviner une formule explicite pour f_n. Une fois cette formule trouvée, utilisez `makelist` pour générer la même suite.
10. Utilisez `makelist` pour tracer 40 cercles centrés à l'origine avec des rayons $0.1, 0.2, 0.3, \ldots$ dans un seul graphique. Ce problème est délicat, car cette liste doit produire des éléments que `wxdraw2d` sait interpréter : `implicit` et ses arguments appropriés doivent être inclus dans chaque élément de la liste !
11. Toute parabole peut s'écrire sous la forme $f(x) = ax^2 + bx + c$. Les paramètres a, b, c définissent complètement la parabole. Si l'on vous donne trois points appartenant à une parabole inconnue, ils génèrent un système de trois équations en a, b, c. La commande `solve` permet de résoudre rapidement ce système. Écrivez une solution similaire à l'exemple 0.6.1 pour définir trois points quelconques et tracer à la fois ces points et la parabole qui passe par ces trois points. La syntaxe correcte pour résoudre un système d'équations est : `solve([eqn1,eqn2,eqn3],[var1,var2,var3])`. Appliquer ce programme aux points $(-6.8, -5.5)$, $(0.1, 6.7)$ et $(3.2, -0.9)$.

Chapitre 1

Panorama sur les fonctions

1.1 Fonctions polynômes et rationnelles **22**
1.1.1 Fonctions polynômes 22
1.1.2 Fonctions rationnelles 23
1.2 Fonctions trigonométriques **26**
1.2.1 Les fonctions trigonométriques élémentaires 26
1.2.2 Graphes des fonctions sinusoïdales 28
1.3 Les fonctions exponentielles **33**
1.4 Transformation des fonctions **35**
1.5 Parité des fonctions **38**
1.6 Combinaisons algébriques de fonctions **40**
1.7 Composition de fonctions **45**
1.8 Fonctions inverses **47**
1.9 Exercices du Chapitre 1 . . . . . **53**

Commandes essentielles de ce Chapitre

`wxdraw2d`	`factor`	`solve`	`dimensions`
`num`	`denom`	`ratsimp`	`float`
`limit`	`find_root`		

1.1 Fonctions polynômes et rationnelles

1.1.1 Fonctions polynômes

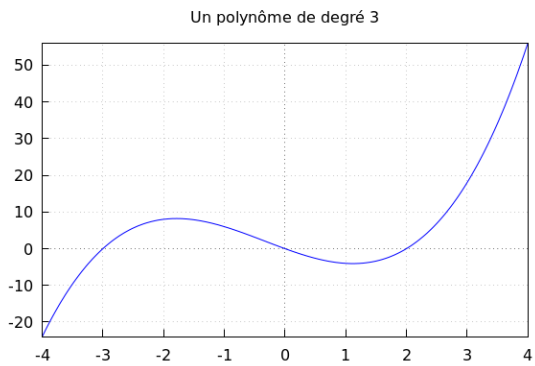
Une **fonction polynôme** est une fonction de la forme :

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

Représenter graphiquement une fonction polynôme avec Maxima est simple, mais il faut noter que la forme factorisée de la fonction peut être très utile pour trouver les racines de cette dernière.

Exemple 1.1.1. Représenter la fonction $P(x) = x^3 + x^2 - 6x$ et déterminer ses racines.

```
(%i1) P:x^3+x^2-6*x$

(%i2) wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      title="Un polynôme de degré 3",
      explicit((P),x,-4,4)
      );
```



Il apparait que la fonction $P(x)$ a pour racines $x = -3$, $x = 0$, et $x = 2$. Nous pouvons vérifier ces valeurs en utilisant wxMaxima pour factoriser $P(x)$:

```
(%i3) factor(P);
(%o3) (x-2)*x*(x+3)
```

Une autre possibilité est aussi de résoudre l'équation $P(x) = 0$:

```
(%i4) solve(P=0,x);
(%o4) [x=-3,x=2,x=0]
```

1.1.2 Fonctions rationnelles

Une **fonction rationnelle** est une fonction de la forme :

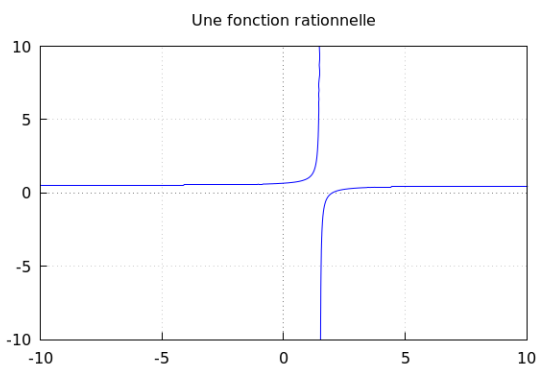
$$f(x) = \frac{P(x)}{Q(x)}$$

où $P(x)$ et $Q(x)$ sont des polynômes, avec un domaine de définition restreint aux valeurs de x pour lesquelles $Q(x) \neq 0$. Nous sommes souvent intéressés à trouver les racines et les asymptotes de ces fonctions rationnelles.

Exemple 1.1.2. Pour la fonction rationnelle $Q(x) = \frac{x^2-x-2}{2x^2-x-3}$, trouver toutes les asymptotes, les racines et les points de discontinuité. Représenter graphiquement $Q(x)$ ainsi que toutes les asymptotes, les racines et les discontinuités.

Si nous n'indiquons pas l'intervalle pour y à wxMaxima, l'échelle verticale de ce graphe est trop étendue pour que nous puissions voir ses propriétés essentielles. Pour remédier à ce problème, nous utilisons `xrange` et `yrange` pour définir de manière explicite la taille de la fenêtre graphique.

```
(%i5) Q:(x^2-x-2)/(2*x^2-x-3);
(%o5) (x^2-x-2)/(2*x^2-x-3)

(%i6) wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      title="Une fonction rationnelle",
      xrange=[-10,10],
      yrange=[-10,10],
      explicit((Q),x,-10,10)
      );
```



La courbe possède un zéro, une asymptote verticale et une asymptote horizontale. Afin de trouver les valeurs exactes de ces éléments nous aurons à factoriser au maximum le numérateur et le dénominateur de $Q(x)$. Les commandes `num` et `denom` permettent d'extraire les numérateur et dénominateur de $Q(x)$ pour les factoriser :

```
(%i7) factor(num(Q));
```

```
(%o7) (x-2)*(x+1)
(%i8) factor(denom(Q));
(%o8) (x+1)*(2*x-3)
```

Nous remarquons qu'il existe un facteur commun qui est $(x+1)$ qui peut se simplifier, mais il ne peut être entièrement ignoré ! Comme le dénominateur de $Q(x)$ s'annule pour $x=-1$, $Q(x)$ n'est pas défini en ce point. Nous représentons ceci comme une discontinuité dans le graphique ; malheureusement wxMaxima ne fait pas apparaître de manière visible cette discontinuité, mais nous pouvons la représenter correctement en insérant un cercle au point $(-1, 0.6)$ sur le graphique.

Lorsque $Q(x)$ est simplifié en $\frac{x-2}{2x-3}$, nous pouvons rapidement déterminer que la racine (valeur qui annule le numérateur) est localisée à $x=2$ et que l'asymptote verticale (où le dénominateur s'annule) a pour équation $x=3/2$. Bien évidemment, nous pouvons laisser wxMaxima réaliser les calculs algébriques correspondants si nous le désirons :

```
(%i9) R:ratsimp(Q);
(%o9) (x-2)/(2*x-3)
(%i10) solve(num(R)=0);
(%o10) [x=2]
(%i11) solve(denom(R)=0);
(%o11) [x=3/2]
```

Nous déterminons l'asymptote horizontale en utiisant la commande `limit` de wxMaxima. Nous étudierons éventuellement les limites plus en détail, mais pour l'instant nous avons juste à savoir que $\lim_{x\to\pm\infty} Q(x) = L$ nous indique qu'il existe une asymptote horizontale d'équation $y=L$. La limite signifie simplement que "$Q(x)$ s'approche de très près de L lorsque x devient très 'grand' ".

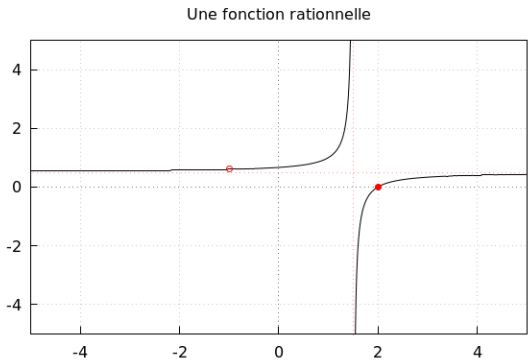
Nous calculons les limites à l'infini avec wxMaxima et trouvons qu'il existe une asymptote horizontale d'équation $y=1/2$.

```
(%i12) limit(Q,x,inf);
(%o12) 1/2
(%i13) limit(Q,x,-inf);
(%o13) 1/2
```

Finalement, nous rassemblons toutes ces informations sur un même graphique. Remarquons que l'asymptote verticale $x=1.5$ doit être définie *paramétriquement* comme nous l'avons vu au Chapitre 0.

```
(%i14) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       title="Une fonction rationnelle",
       xrange=[-5,5],
       yrange=[-5,5],
       color=black,
         explicit((Q),x,-5,5),
       color=red,
```

```
line_type=dots,
  explicit((0.5),x,-5,5),
  parametric(1.5,t,t,-5,5),
point_type=6,
  points([[-1,0.6]]),
point_type=7,
  points([[2,0]])
);
```



1.2 Fonctions trigonométriques

1.2.1 Les fonctions trigonométriques élémentaires

Les fonctions trigonométriques élémentaires sont définies ci-dessous :

$\sin x$ est l'ordonnée du point du cercle unité faisant un angle de x.

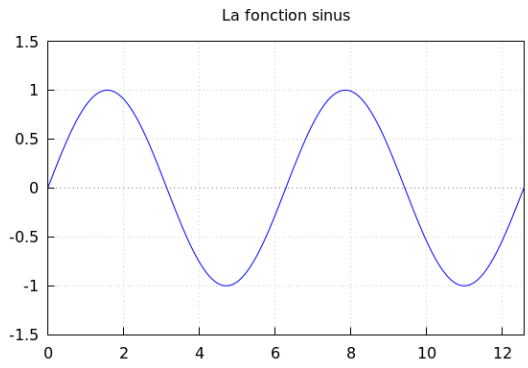
$\cos x$ est l'abscisse du point du cercle unité faisant un angle de x.

$\tan x$ est le rapport $\dfrac{\sin(x)}{\cos(x)}$

L'angle, x, est mesuré en radians dans le sens contraire des aiguilles d'une montre à partir de l'axe des abscisses positives.

Exemple 1.2.1. Dessiner le graphe de la fonction sinus sur l'intervalle $[0, 4\pi]$. Trouver la période et les trois premiers zéros (racines) de la fonction sinus.

```
(%i1) S:sin(x)$
(%i2) wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      title="La fonction sinus",
      xrange=[0,4*%pi],
      yrange=[-1.5,1.5],
      explicit((S),x,0,4*%pi)
      );
```



Nous voyons que la fonction sinus est une onde simple avec une période de $2\pi \approx 6,3$. Les zéros de la fonction peuvent être calculés en utilisant `"find_root"` – malheureusement, la commande ne nous donne que des approximations numériques pour une seule racine à la fois sur un intervalle fermé. Nous devrons estimer graphiquement les valeurs des racines et construire les intervalles fermés en conséquence. Les trois premières racines sont approximées comme suit :

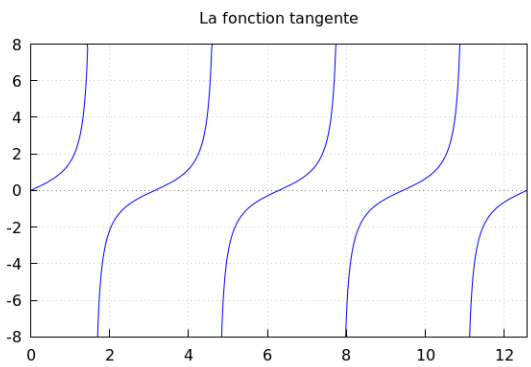
```
(%i3) find_root(S,0,1);
```

```
(%o3) 0.0
(%i4) find_root(S,2,4);
(%o4) 3.141592653589793
(%i5) find_root(S,6,8);
(%o5) 6.283185307179586
```

Nous vérifions que ce sont des valeurs numériques approchées de $0, \pi, 2\pi, \ldots$

Exemple 1.2.2. Faites un tracé de la fonction tangente sur l'intervalle $[0, 4\pi]$. Identifiez la période, les zéros (racines) et les équations des asymptotes verticales.

```
(%i6) T:tan(x)$
(%i7) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       title="La fonction tangente",
       xrange=[0,4*%pi],
       yrange=[-8,8],
       explicit((T),x,0,4*%pi)
       );
```

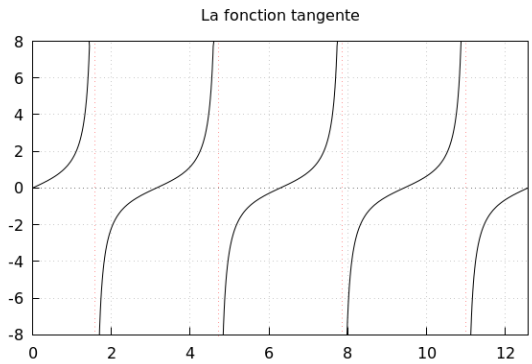


La fonction tangente a une période de seulement $\pi \approx 3,14$. Les racines annulent le numérateur, donc sont solutions de $\sin x = 0$, ce qui correspond à $x = 0, \pi, 2\pi, \ldots$. La fonction tangente possède de nombreuses asymptotes verticales correspondant aux valeurs de x pour lesquelles le dénominateur $\cos x = 0$. Pour cet exemple, nous rappelons simplement que $\cos x = 0$ lorsque $x = \frac{\pi}{2}, x = \frac{3\pi}{2}, \ldots$, ce qui nous donne notre liste des équations pour les asymptotes verticales.

En redessinant la fonction avec les asymptotes verticales, nous obtenons :

```
(%i8) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
```

```
title="La fonction tangente",
xrange=[0,4*%pi],
yrange=[-8,8],
color=black,
explicit((T),x,0,4*%pi),
color=red,
line_type=dots,
  parametric(%pi/2,t,t,-8,8),
  parametric(3*%pi/2,t,t,-8,8),
  parametric(5*%pi/2,t,t,-8,8),
  parametric(7*%pi/2,t,t,-8,8)
);
```



1.2.2 Graphes des fonctions sinusoïdales

Une fonction de la forme $f(x) = A\cos(Bx + \phi) + C$ ou $f(x) = A\sin(Bx + \phi) + C$ est appelée **sinusoïdale**. Le graphe d'une fonction sinusoïdale est une onde simple.

La **ligne médiane** d'une fonction sinusoïdale est la droite $y = C$: une ligne horizontale située à équidistance des valeurs minimale et maximale.

L'amplitude d'une fonction sinusoïdale est donnée par A : la distance entre la ligne médiane et les valeurs maximale et minimale de la fonction.

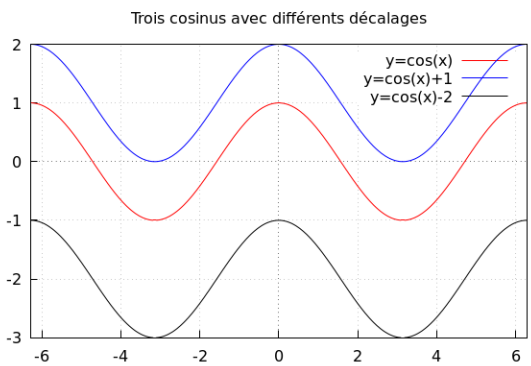
La période d'une fonction sinusoïdale est déterminée par B selon la formule $T = \frac{2\pi}{B}$. La période est la plus petite distance horizontale parcourue avant que la fonction sinusoïdale ne commence à se répéter.

Le déphasage d'une fonction sinusoïdale est donné par ϕ. Le déphasage est la *fraction de période de la translation horizontale* multipliée par 2π, sachant qu'une valeur négative correspond à un décalage vers la droite, et qu'une valeur positive correspond à un décalage vers la gauche.

Exemple 1.2.3. Représenter $f(x) = \cos x$, $g(x) = \cos x + 1$, et $h(x) = \cos x - 2$ sur un

même graphique en les colorant respectivement en rouge, bleu et noir. Inclure une étiquette afin d'identifier chaque courbe sur le graphique.

```
(%i9)  F:cos(x)$
       G:cos(x)+1$
       H:cos(x)-2$
(%i12) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       title="Trois cosinus avec différents décalages",
       key="y=cos(x)",
       color="red",
        explicit((F),x,-2*%pi,2*%pi),
       key="y=cos(x)+1",
       color="blue",
        explicit((G),x,-2*%pi,2*%pi),
       key="y=cos(x)-2",
       color="black",
       explicit((H),x,-2*%pi,2*%pi)
       );
```

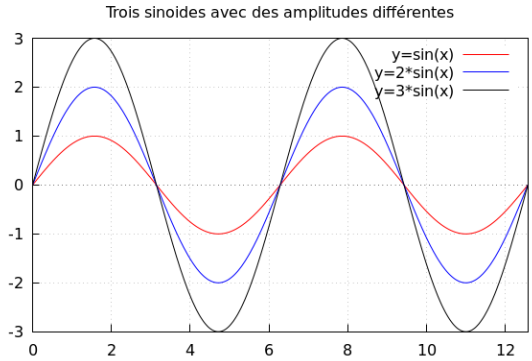


Nous voyons que la constante additive décale simplement la ligne médiane vers le haut ou vers le bas du nombre d'unités correspondant.

Exemple 1.2.4. Représenter les trois fonctions $f_1 = \sin x$, $f_2 = 2\sin x$, et $f_3 = 3\sin x$ respectivement en rouge, bleu et noir. Inclure une étiquette afin d'identifier chaque courbe sur le graphique.

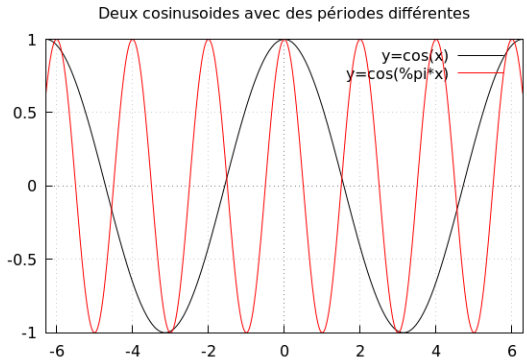
```
(%i13) F1:sin(x)$
       F2:2*sin(x)$
       F3:3*sin(x)$
(%i16) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
```

```
title="Trois sinoides avec des amplitudes différentes",
key="y=sin(x)",
color=red,
  explicit((F1),x,0,4*%pi),
key="y=2*sin(x)",
color=blue,
  explicit((F2),x,0,4*%pi),
key="y=3*sin(x)",
color=black,
  explicit((F3),x,0,4*%pi)
);
```



Exemple 1.2.5. Représenter $f(x) = \cos x$ et $g(x) = \cos \pi x$ sur le même graphique en les colorant respectivement en noir et rouge. Inclure une étiquette pour identifier chaque courbe sur le graphique. Noter les périodes des deux fonctions et les comparer avec la formule $T = \frac{2\pi}{B}$.

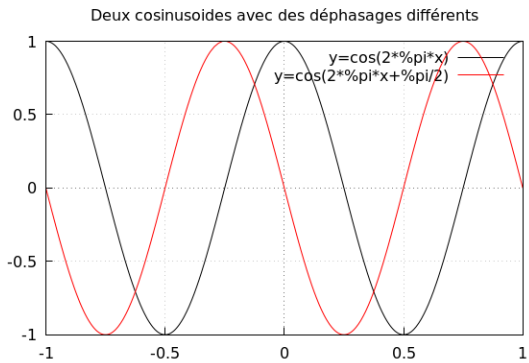
```
(%i17) F:cos(x)$
       G:cos(%pi*x)$
(%i19) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       title="Deux cosinusoides avec des périodes différentes",
       key="y=cos(x)",
       color="black",
        explicit((F),x,-2*%pi,2*%pi),
       key="y=cos(%pi*x)",
       color="red",
       explicit((G),x,-2*%pi,2*%pi)
       );
```



Nous voyons que la fonction cosinus $f(x) = \cos x$ a une période de 2π (un peu plus que 6), tandis que la nouvelle fonction $g(x) = \cos(\pi x)$ a une période exactement de 2. Ceci est en accord avec la formule $T = \frac{2\pi}{B} = \frac{2\pi}{\pi} = 2$.

Exemple 1.2.6. Représenter $f(x) = \cos(2\pi x)$ et $g(x) = \cos(2\pi x + \frac{\pi}{2})$ sur le même graphique en les colorant respectivement en noir et rouge. Inclure une étiquette pour identifier chaque courbe sur le graphique. Noter les périodes des deux fonctions et vérifier la définition du déphasage.

```
(%i20) F:cos(2*%pi*x)$
       G:cos(2*%pi*x+%pi/2)$
(%i22) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       title="Deux cosinusoides avec des déphasages différents",
       key="y=cos(2*%pi*x)",
       color="black",
        explicit((F),x,-1,1),
       key="y=cos(2*%pi*x+%pi/2)",
       color="red",
        explicit((G),x,-1,1)
       );
```



Nous voyons que $f(x) = \cos(2\pi x)$ est une fonction cosinus avec une période de 1, et que $g(x) = \cos(2\pi x + \frac{\pi}{2})$ est une fonction cosinus avec une période de 1 qui a été décalée vers la gauche de $\frac{1}{4}$. Cela correspond à la définition du déphasage , sachant que $\frac{\pi}{2}$ est en fait $\frac{1}{4}$ de 2π : dans ce cas un quart de la période vaut simplement $\frac{1}{4}$.

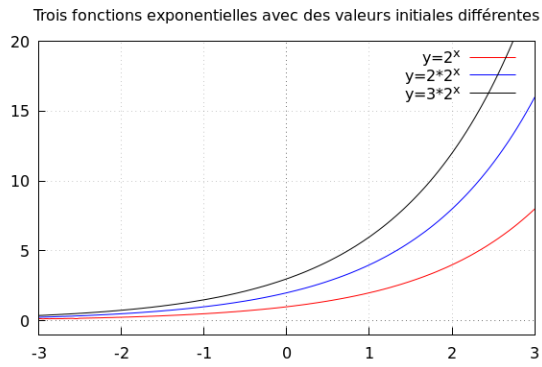
1.3 Les fonctions exponentielles

Une **fonction exponentielle** est une fonction de la forme $f(x) = a \cdot b^x$ ou $f(x) = a \cdot e^{kx}$. Nous passons d'une forme à l'autre en utilisant la relation $k = \ln b$.

Nous sommes souvent intéressés par la *valeur initiale* (intersection avec y) et le *taux de croissance* (qui est déterminé par b ou, de manière équivalente, k). Les fonctions exponentielles simples sont asympotiques à l'axe des x.

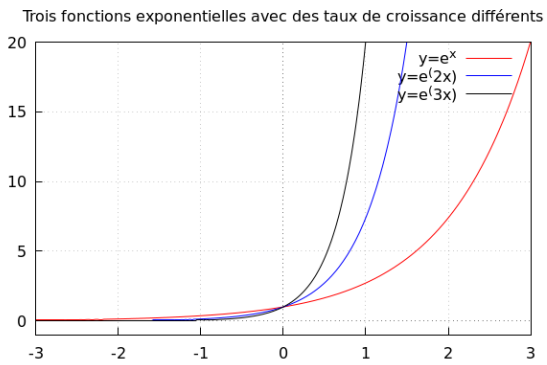
Exemple 1.3.1. Les fonctions $f_1(x) = 2^x$, $f_2(x) = 2 \cdot 2^x$ et $f_3(x) = 3 \cdot 2^x$ ont toutes trois le même taux de croissance mais différentes valeurs initiales. Représenter les fonctions respectivement en rouge, bleu et noir. Inclure une étiquette sur le graphe pour identifier chacune des fonctions.

```
(%i1) F1:2^x$
      F2:2*2^x$
      F3:3*2^x$
(%i4) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       xrange=[-3,3],
       yrange=[-1,20],
       title="Trois fonctions exponentielles avec des valeurs initiales différentes",
       key="y=2^x",
       color=red,
         explicit((F1),x,-3,3),
       key="y=2*2^x",
       color=blue,
         explicit((F2),x,-3,3),
       key="y=3*2^x",
       color=black,
         explicit((F3),x,-3,3)
       );
```



Exemple 1.3.2. Les fonctions $f_1(x) = e^x$, $f_2(x) = e^{2x}$ et $f_3(x) = e^{3x}$ ont toutes la même valeur initiale mais des taux de croissance différents. Représenter les fonctions respectivement en rouge, bleu et noir. Inclure une étiquette dans le graphique pour indiquer chaque fonction.

```
(%i5) F1:exp(x)$
      F2:exp(2*x)$
      F3:exp(3*x)$
(%i6) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       xrange=[-3,3],
       yrange=[-1,20],
       title="Trois fonctions exponentielles avec des taux de croissance différents",
       key="y=e^x",
       color=red,
         explicit((F1),x,-3,3),
       key="y=e^(2x)",
       color=blue,
         explicit((F2),x,-3,3),
       key="y=e^(3x)",
       color=black,
         explicit((F3),x,-3,3)
       );
```



1.4 Transformation des fonctions

Les transformations des fonctions sont de simples transformations de la formule exprimant la fonction qui entrainent des modifications connues sur son graphique.

Par rapport à $f(x)$,

$g(x) = f(-x)$	est symétrique par rapport à l'axe des y.
$g(x) = -f(x)$	est symétrique par rapport à l'axe des x.
$g(x) = cf(x)$	est agrandie verticalement d'un facteur de c.
$g(x) = f(cx)$	est rétrécie horizontalement d'un facteur de c.
$g(x) = f(x + c)$	est translaté vers la gauche de c unités.
$g(x) = f(x) + c$	est translaté vers le haut de c unités.

Exemple 1.4.1. Soit $f(x) = e^{0.5 \cdot x}$. Utilisez wxMaxima pour trouver les formules explicites pour $g(x) = f(-x)$, $h(x) = -f(x)$, $i(x) = f(x + 5)$ et $j(x) = f(x) + 5$. Représentez graphiquement $f(x)$ en noir et les transformées respectivement en bleu, rouge, vert et jaune.

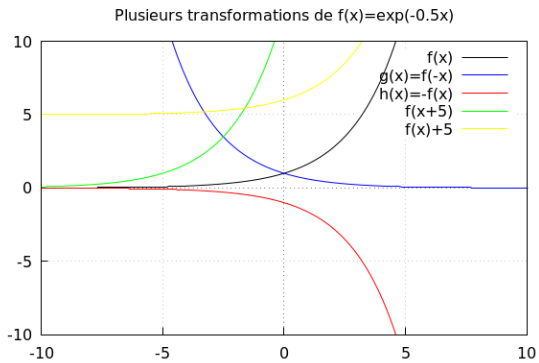
Quand nous voulons utiliser les propriétés des fonctions dans wxMaxima, nous devons définir une fonction (plutôt qu'une expression) en utilisant `':='` à la place de `':'`.

```
(%i1) F(x):=exp(0.5*x)$
(%i2) G(x):=F(-x)$
      H(x):=-F(x)$
      I(x):=F(x+5)$
      J(x):=F(x)+5$
(%i6) G(x);
(%o6) %e^(-0.5*x)
(%i7) H(x);
(%o7) -%e^(0.5*x)
(%i8) I(x);
(%o8) %e^(0.5*(x+5))
(%i9) J(x);
(%o9) %e^(0.5*x)+5
```

Nous nous apercevons que les formules sont données par $g(x) = e^{-.5x}$, $h(x) = -e^{0.5x}$, $i(x) = e^{0.5(x+5)}$, et $j(x) = e^{0.5x} + 5$. Nous représentons les fonctions à l'intérieur d'une fenêtre de dimension raisonnable ci-dessous :

```
(%i10) wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      title="Plusieurs transformations de f(x)=exp(-0.5x)",
      xrange=[-10,10],
      yrange=[-10,10],
      color=black,
      key="f(x)",
       explicit((F(x)),x,-10,10),
```

```
color=blue,
key="g(x)=f(-x)",
 explicit((G(x)),x,-10,10),
color=red,
key="h(x)=-f(x)",
 explicit((H(x)),x,-10,10),
color=green,
key="f(x+5)",
 explicit((I(x)),x,-10,10),
color=yellow,
key="f(x)+5",
 explicit((J(x)),x,-10,10)
);
```

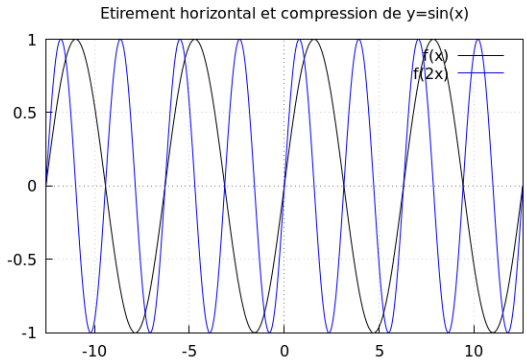


La ligne médiane, l'amplitude, la période et le déphasage d'une fonction sinusoïdale peuvent être considérés comme résultant d'une ou plusieurs transformations de fonctions appliquées aux fonctions sinus ou cosinus. L'exemple suivant illustre comment un étirement ou une compression horizontale peut modifier la *période* d'une sinusoïde.

Exemple 1.4.2. Soit $f(x) = \sin x$. Représenter graphiquement $f(x)$ ainsi que $g(x) = f(2x)$ avec des couleurs différentes et des étiquettes. Puis représenter $f(x)$ et $h(x) = f(0.5x)$ sur un nouveau graphique. L'intervalle pour le tracé est de $[-4\pi, 4\pi]$ pour les deux graphiques. Commenter les périodes de ces trois fonctions.

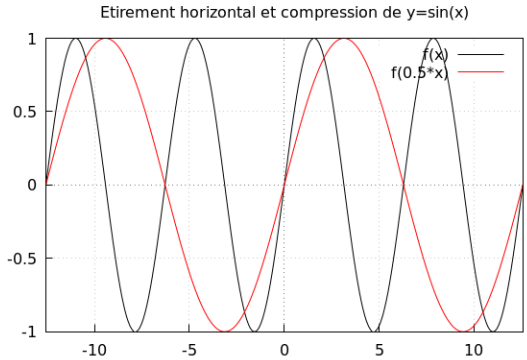
```
(%i11)F(x):=sin(x)$
      G(x):=F(2*x)$
      H(x):=F(0.5*x)$
(%i14) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       title="Etirement horizontal et compression de y=sin(x)",
       key="f(x)",
       color=black,
        explicit((F(x)),x,-4*%pi,4*%pi),
```

```
key="f(2x)",
color=blue,
 explicit((G(x)),x,-4*%pi,4*%pi)
);
```



Nous voyons que $g(x) = f(2x)$ a la moitié de la période de la fonction originale : elle a été compressée horizontalement d'un facteur de 2 en accord avec la formule $T = \frac{2\pi}{B}$.

```
(%i15) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       title="Etirement horizontal et compression de y=sin(x)",
       key="f(x)",
       color=black,
        explicit((F(x)),x,-4*%pi,4*%pi),
       key="f(0.5*x)",
       color=red,
        explicit((H(x)),x,-4*%pi,4*%pi)
       );
```



Nous voyons que $h(x) = f(0.5x)$ a une période qui est double de celle de la fonction originale : elle a été étirée horizontalement d'un facteur de 2 conformément à la formule $T = \frac{2\pi}{B}$.

1.5 Parité des fonctions

De manière générale, *la Parité* fait référence à une fonction paire ou impaire. Les définitions algébriques sont les suivantes :

$$f(x) \text{ est une fonction paire si } f(-x) = f(x).$$

$$f(x) \text{ est une fonction impaire si } f(-x) = -f(x).$$

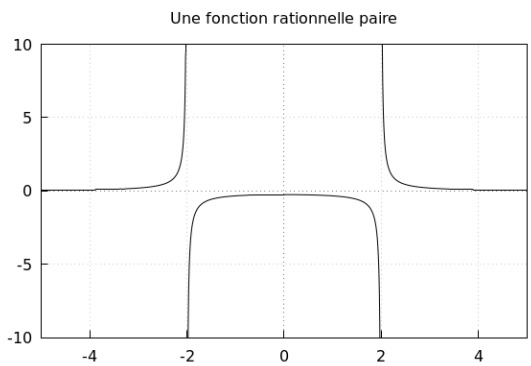
Il est important de se rendre compte que les définitions algébriques correspondent à des *symétries* utiles du graphe d'une fonction. En termes de transformations de fonctions, nous disons que le graphe d'une fonction paire est identique par la réflexion par rapport à l'axe des ordonnées, et que le graphe d'une fonction impaire est identique par des symétries combinées du graphe par rapport aux axes des abscisses et des ordonnées.

Exemple 1.5.1. Utiliser wxMaxima pour vérifier de manière algébrique que $f(x) = \frac{1}{x^2-4}$ est une fonction paire. Représenter graphiquement $f(x)$ et commenter la symétrie de la courbe.

```
(%i1) f(x):=1/(x^2-4)$
(%i2) f(-x);
(%o2) 1/(x^2-4)
```

Nous voyons que la formule pour $f(x)$ est exactement la même que celle de la formule pour $f(-x)$, donc la fonction est paire. Ensuite, nous représentons la fonction $f(x)$ dans une fenêtre adaptée :

```
(%i3) wxdraw2d(
      xaxis=true,
      yaxis=true,
      xrange=[-5,5],
      yrange=[-10,10],
      grid=true,
      title="Une fonction rationnelle paire",
      color=black,
      explicit((f(x)),x,-5,5)
      );
```



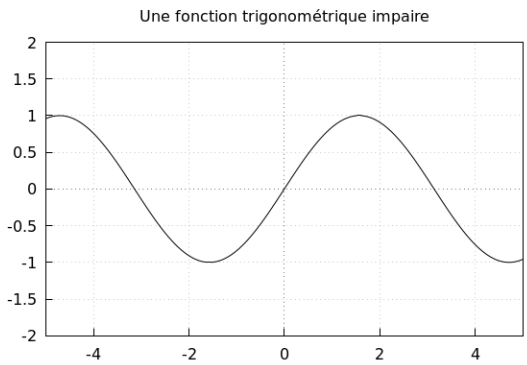
Comme pour toutes les fonctions paires, la courbe de $f(x)$ est symétrique par rapport à l'axe des ordonnées.

Exemple 1.5.2. Utiliser wxMaxima pour vérifier algébriquement que la fonction $f(x) = \sin x$ est une fonction impaire. Représenter graphiquement $f(x)$ et préciser les symétries de la courbe.

```
(%i4) f(x):=sin(x)$
(%i5) f(-x);
(%o5) -sin(x)
```

Nous voyons que la formule pour $f(-x)$ est $-\sin x$. La fonction est impaire car $f(-x) = -f(x)$.

```
(%i6) wxdraw2d(
      xaxis=true,
      yaxis=true,
      xrange=[-5,5],
      yrange=[-2,2],
      grid=true,
      title="Une fonction trigonométrique impaire",
      color=black,
      explicit((f(x)),x,-5,5)
      );
```



Comme toutes les fonctions impaires, $f(x) = \sin x$ est symétrique par rapport à l'origine ; c'est à dire, qu'une composée de réflexions par rapport à l'axe des abscisses et à l'axe des ordonnées redonne le même graphe. La symétrie est équivalente à une rotation de 180° autour de l'origine.

1.6 Combinaisons algébriques de fonctions

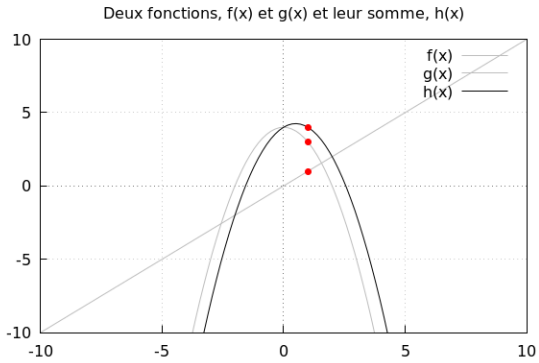
Les fonctions $f(x)$ and $g(x)$ peuvent être combinées algébriquement de manière très variée afin de générer une nouvelle fonction, $h(x)$:

Par somme ou différence :	$h(x) = f(x) \pm g(x)$
Par combinaison linéraire :	$h(x) = af(x) + bg(x)$
Par produit :	$h(x) = f(x)g(x) = (f \cdot g)(x)$
Par quotient :	$h(x) = \dfrac{f(x)}{g(x)}$, sous réserve que $g(x) \neq 0$

Pour appartenir au domaine d'une combinaison algébrique de fonctions, une valeur de x doit appartenir à la fois au domaine de $f(x)$ et de $g(x)$. Dans tous les cas, la combinaison se fait "point par point".

Exemple 1.6.1. Définir $f(x) = 4 - x^2$ et $g(x) = x$, puis $h(x) = f(x) + g(x)$. Effectuer une représentation graphique de ces trois fonctions, avec $f(x)$ et $g(x)$ représentées en gris and $h(x)$ dessinée en noir. En complément, représenter les points $(1, f(1))$, $(1, g(1))$ et $(1, h(1))$ comme des disques et commenter la notion de "point par point" pour une addition de fonctions.

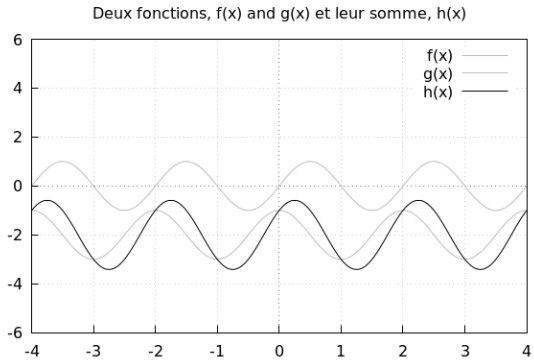
```
(%i1) f(x):=4-x^2$
      g(x):=x$
      h(x):=f(x)+g(x)$
(%i4) wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      title="Deux fonctions, f(x) et g(x) et leur somme, h(x)",
      xrange=[-10,10],
      yrange=[-10,10],
      color=grey,
       key="f(x)",
         explicit((f(x)),x,-10,10),
       key="g(x)",
         explicit((g(x)),x,-10,10),
       key="h(x)",
       color=black,
         explicit((h(x)),x,-10,10),
       key="",
       color=red,
       point_type=7,
         points([[1,f(1)],[1,g(1)],[1,h(1)]])
       );
```



Les points mis en évidence montrent clairement comment fonctionne l'addition de fonctions : $f(1) = 3$, $g(1) = 1$, donc $h(1)$ est simplement $3 + 1 = 4$. Il est toujours possible de tracer rapidement des sommes, des différences, etc., en regardant simplement les graphiques des fonctions initiales point par point.

Exemple 1.6.2. Définir $f(x) = \sin(\pi x)$ et $g(x) = \cos(\pi x) - 2$. Ensuite définir $h(x)$ comme une combinaison linéaire $h(x) = 2f(x) + 3g(x)$. Dessiner le graphe de ces trois fonctions, avec $f(x)$ et $g(x)$ représentées en gris et $h(x)$ dessinée en noir. Commenter la période de chacune de ces fonctions.

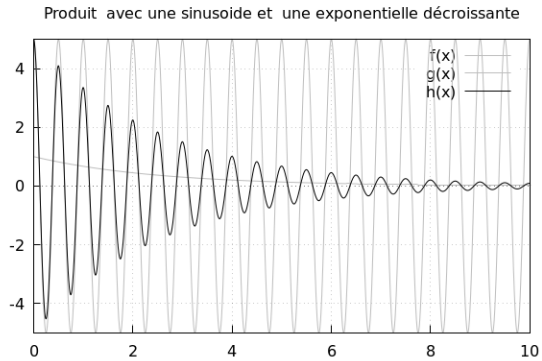
```
(%i5) f(x):=sin(%pi*x)$
      g(x):=cos(%pi*x)-2$
      h(x):=f(x)+g(x)$
(%i8) wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      title="Deux fonctions, f(x) and g(x) et leur somme, h(x)",
      xrange=[-4,4],
      yrange=[-6,6],
       color=grey,
       key="f(x)",
        explicit((f(x)),x,-4,4),
       key="g(x)",
        explicit((g(x)),x,-4,4),
       key="h(x)",
       color=black,
        explicit((h(x)),x,-4,4)
       );
```



De manière remarquable, nous constatons que la combinaison linéaire de deux fonctions sinusoïdales $f(x)$ et $g(x)$ (toutes deux de période 2) donne une autre fonction sinusoïdale $h(x)$ (également de période 2). Comme discuté dans les Exercices, cela ne fonctionne que si les fonctions initiales ont la même période.

Exemple 1.6.3. Définir les fonctions $f(x) = e^{-0.4x}$, $g(x) = 5\cos(4\pi x)$, et $h(x) = f(x) \cdot g(x)$. Représenter graphiquement ces trois fonctions, avec $f(x)$ et $g(x)$ dessinées en gris et $h(x)$ dessinée en noir. Que peut-on dire des racines de $h(x)$ par rapport aux racines de $f(x)$ et $g(x)$?

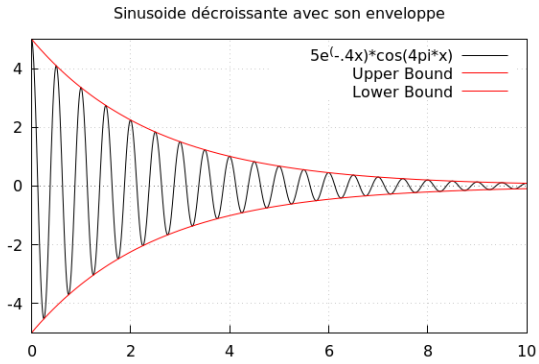
```
(%i9) f(x):=%e^(-0.4*x)$
      g(x):=5*cos(4*%pi*x)$
      h(x):=f(x)*g(x)$
(%i12)wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      title="Produit  avec une sinusoide et  une exponentielle décroissante",
      xrange=[0,10],
      yrange=[-5,5],
      color=grey,
       key="f(x)",
        explicit((f(x)),x,0,10),
       key="g(x)",
        explicit((g(x)),x,0,10),
       key="h(x)",
       color=black,
        explicit((h(x)),x,0,10)
       );
```



$f(x)$ n'est jamais nulle, et $g(x)$ a de nombreux zéros qui apparaissent périodiquement. Puisque $h(x) = f(x) \cdot g(x)$, elle doit s'annuler chaque fois que $f(x) = 0$ ou $g(x) = 0$, donc elle a exactement les mêmes zéros que la fonction cosinus.

Nous pouvons désigner $h(x)$ comme une « fonction sinusoïdale avec une amplitude décroissante ». Ce type de fonction apparaît fréquemment dans les applications physiques. Nous remarquons que $g(x)$ atteint de manière répétée une valeur minimale de -5 et une valeur maximale de +5, et à ces points particuliers, $h(x) = \pm 5e^{-0,4x}$. Si nous traçons ces deux fonctions $i(x) = 5e^{-0,4x}$ et $j(x) = -5e^{-0,4x}$, elles fournissent une borne supérieure et inférieure appelée une *enveloppe* :

```
(%i13) i(x):=5*f(x)$
       j(x):=-5*f(x)$
(%i15) wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      title="Sinusoide décroissante avec son enveloppe",
      xrange=[0,10],
      yrange=[-5,5],
       color=black,
       key="5e^(-.4x)*cos(4pi*x)",
        explicit((h(x)),x,0,10),
       color=red,
       key="Upper Bound",
        explicit((i(x)),x,0,10),
       key="Lower Bound",
        explicit((j(x)),x,0,10)
       );
```


Sinusoide décroissante avec son enveloppe
5e(-.4x)*cos(4pi*x)
Upper Bound
Lower Bound
4
2
0
-2
-4
0
2
4
6
8
10

1.7 Composition de fonctions

La composition de deux fonctions se produit lorsque le résultat d'une fonction est utilisée comme entrée pour une autre fonction. Les compositions de fonctions ont une notation spéciale :

$$(f \circ g)(x) = f(g(x))$$

De manière pratique, cela signifie juste que l'on remplace x par $g(x)$ dans la formule de $f(x)$.

Exemple 1.7.1. Pour les fonctions $f(x) = 2x + 1$ et $g(x) = \sqrt{x}$, utiliser wxMaxima pour calculer $(f \circ g)(x)$ et $(g \circ f)(x)$. Calculer avec le logiciel $(f \circ g)(-0.3)$ et $(g \circ f)(-0.3)$ puis commenter les résultats.

```
(%i1) f(x):=2*x+1$
      g(x):=sqrt(x)$
(%i3) f(g(x));
(%o3) 2*sqrt(x)+1
(%i4) g(f(x));
(%o4) sqrt(2*x+1)
```

Nous voyons que les formules sont différentes : $(f \circ g)(x) = 2\sqrt{x} + 1$ tandis que $(g \circ f)(x) = \sqrt{2x + 1}$. Maintenant, nous calculons les valeurs pour $x = -0.3$ par ces fonctions composées :

```
(%i5) f(g(-0.3));
(%o5) 1.095445115010332*%i+1
(%i6) g(f(-0.3));
(%o6) 0.63245553203368
```

$(f \circ g)(-0,3)$ est *complexe* car nous avons évalué une racine carrée avec un argument négatif, tandis que $(g \circ f)(-0,3)$ ne pose pas le même problème. Cela rappelle l'importance d'évaluer soigneusement le domaine d'une composition de fonctions. Si nécessaire, nous pouvons déterminer le domaine de chaque composition en identifiant les valeurs de x qui correspondent à des nombres non négatifs sous le signe de la racine carrée.

Exemple 1.7.2. Exprimer chacune des fonctions suivantes $i(x)$ comme une composition de deux ou trois fonctions plus simples $f(x)$, $g(x)$ et éventuellement $h(x)$. Dans chaque cas, utiliser wxMaxima pour vérifier que $(f \circ g)(x) = i(x)$ ou $(f \circ g \circ h)(x) = i(x)$

a. $i(x) = 2e^{3x^2+1}$ b. $i(x) = \dfrac{1}{\tan\sqrt{x}}$

Pour le cas a., nous voyons que $i(x)$ est presque comme $2e^x$ à part le fait que x a été remplacé par $3x^2 + 1$. On en déduit que $g(x) = 3x^2 + 1$ et $f(x) = 2e^x$. En vérifiant avec wxMaxima, nous obtenons :

```
(%i7) f(x):=2*%e^x$
      g(x):=3*x^2+1$
      i(x):=f(g(x))$
(%i8) i(x);
(%o8) 2*%e^(3*x^2+1)
```

Pour le cas b., nous voyons que $\tan\sqrt{x}$ est déjà une composée de deux fonctions, donc une troisième fonction $f(x) = \frac{1}{x}$ est nécessaire pour obtenir le résultat :

```
(%i9) f(x):=1/x$
      g(x):=tan(x)$
      h(x):=sqrt(x)$
      i(x):=f(g(h(x)))$
(%i10) i(x);
```

(%o10) $\frac{1}{\tan\sqrt{x}}$

1.8 Fonctions inverses

Nous utilisons souvent x pour indiquer un élément du domaine de définition, y pour indiquer un élément de l'ensemble d'arrivée, et $y = f(x)$ pour indiquer une règle reliant les éléments du domaine de définition aux éléments de l'ensemble d'arrivée. Pour être une *fonction*, cette règle doit associer chaque élément du domaine de définition à un seul élément de l'ensemble d'arrivée (ce qui est parfois appelé le "test de la droite verticale").

Une fonction est dite *injective* (ou *biunivoque*) si chaque élément de l'ensemble d'arrivée est associé à un seul élément du domaine de définition ; c'est-à-dire, chaque y est associé à un seul x. Si une fonction est injective, une *fonction inverse* peut être définie, qui prend les éléments de l'ensemble d'arrivée comme « entrées » et fournit les éléments du domaine de départ comme « sorties ». Si nous résolvons l'équation originale $y = f(x)$ pour x, nous obtenons automatiquement une formule pour la fonction inverse (en fonction de y), et la notation pour cette fonction est f^{-1} :

$$\text{Si } y = f(x), \text{ alors } x = f^{-1}(y).$$

Il est toujours vrai que $f^{-1}(f(x)) = x$ et $f(f^{-1}(y)) = y$.

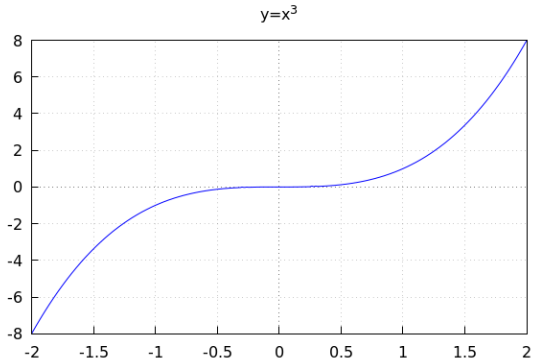
Dans le cas où une fonction n'est *pas* bijective, il peut être utile d'introduire une *restriction du domaine de définition* pour forcer la fonction à devenir bijective. C'est-à-dire, nous retirons des valeurs de x du domaine de définition jusqu'à ce que chaque valeur de y soit associée à une seule valeur de x. Par exemple, les fonctions trigonométriques de base ont des restrictions de domaine « classiques » afin de les rendre inversibles. Cette modification a un coût : parfois, nous devons réfléchir attentivement à la valeur de x que nous essayons réellement d'obtenir lorsque nous appliquons une fonction trigonométrique inverse, et nous devons peut-être « manuellement » convertir en une valeur de x en dehors du domaine restreint.

Parfois, nous souhaitons tracer f^{-1} en fonction de x au lieu de y ; c'est-à-dire, nous traçons $y = f^{-1}(x)$. En effectuant ce changement de notation, nous appelons les éléments de l'ensemble d'arrivée « originaux » de notre fonction « x » au lieu de « y », et nous appelons les éléments du domaine « originaux » de notre fonction « y » au lieu de « x ». Par conséquent, chaque paire ordonnée dans le graphe de $f^{-1}(x)$ a ses coordonnées « échangées » par rapport au graphe de $f(x)$. L'échange des coordonnées équivaut à une symétrie de $f(x)$ par rapport à la droite $y = x$.

Exemple 1.8.1. Définir la fonction $f(x) = x^3$ dans wxMaxima et utiliser le graphe pour établir que la fonction est bijective. Résoudre $y = x^3$ en x afin d'obtenir la formule pour $f^{-1}(y)$, puis définir $f^{-1}(x)$ en tant que nouvelle fonction dans wxMaxima. Vérifier que les composées $f \circ f^{-1}$ et $f^{-1} \circ f$ correspondent aux attentes théoriques. Pour terminer, représenter graphiquement $f(x)$, $f^{-1}(x)$ et $y = x$ pour vérifier que $f^{-1}(x)$ est bien symétrique de $f(x)$ par rapport à $y = x$.

```
(%i1) F(x):=x^3$
(%i2) wxdraw2d(
      xaxis=true,
      yaxis=true,
```

```
grid=true,
title="y=x^3",
explicit((F(x)),x,-2,2)
);
```



Nous pouvons voir que la fonction est bijective car chaque valeur de y est associée à une seule valeur de x. Une autre façon de le dire est qu'une ligne horizontale ne coupe le graphe qu'en un seul point. Maintenant, nous utilisons wxMaxima pour trouver une formule pour l'inverse :

```
(%i3) F:y=x^3$
(%i4) solve(F,x);
(%o4) [x=((sqrt(3)*%i-1)*y^(1/3))/2,x=-((sqrt(3)*%i+1)*y^(1/3))/2,x=y^(1/3)]
```

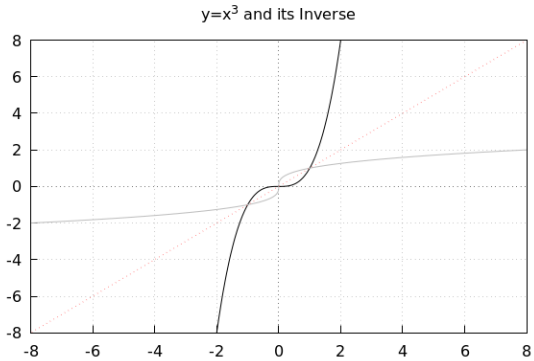
wxMaxima trouve trois solutions pour x, mais nous pouvons ignorer les solutions complexes. Nous voyons que la solution réelle est $x = \sqrt[3]{y}$, donc f^{-1} est la fonction racine cubique. Ensuite, nous pouvons vérifier que les compositions de f et f^{-1} fonctionnent comme prévu :

```
(%i5) F(x):=x^3$
      Finv(y):=y^(1/3)$
(%i6) F(Finv(y));
(%o6) y
(%i7) Finv(F(x));
(%o7) x
```

Il n'était pas strictement nécessaire de définir f^{-1} comme une fonction de y, mais cela rappelle bien que la fonction inverse opère sur les éléments de l'ensemble d'arrivée de f. Enfin, nous produisons le tracé (notez l'utilisation de l'attribut `dimensions` pour rendre le repère orthonomal afin de mieux illustrer la symétrie) :

```
(%i8) wxdraw2d(
       dimensions=[600,600],
       xaxis=true,
       yaxis=true,
       xrange=[-8,8],
       yrange=[-8,8],
       grid=true,
```

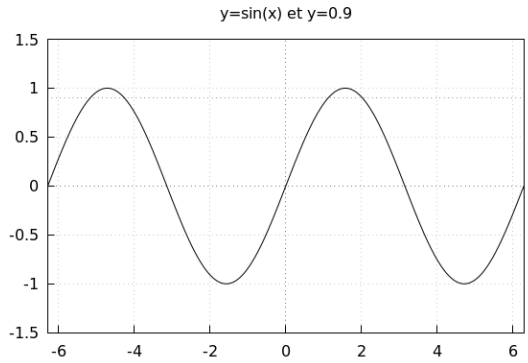
```
       title="y=x^3 and its Inverse",
       color=black,
        explicit((F(x)),x,-8,8),
       color=grey,
        explicit((Finv(x)),x,-8,8),
       color=red,
       line_type=dots,
        explicit((x),x,-8,8)
       );
```



Nous constatons que f et f^{-1} sont bien symétriques par rapport à la droite $y = x$ comme attendu.

Exemple 1.8.2. Définir $f(x) = \sin x$. Représenter graphiquement $f(x)$ et sur le même graphique dessiner $y = 0.9$. Combien de fois ces courbes se coupent-elles ? Qu'implique ce résultat quant à la bijectivité de $f(x)$? Trouver une approximation décimale pour au moins deux de ces points d'intersection. Maintenant, représenter le même graphe avec $f(x) = \sin x$ restreint au domaine utile $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Commenter la bijectivité de cette fonction restreinte au domaine précédent. Utiliser wxMaxima pour trouver une formule pour la fonction inverse, puis représenter cette fonction sur le même graphique que la fonction $f(x)$ restreinte au domaine indiqué.

```
(%i9) f(x):=sin(x)$
(%i10) wxdraw2d(
       xaxis=true,
       yaxis=true,
       xrange=[-2*%pi,2*%pi],
       yrange=[-1.5,1.5],
       grid=true,
       title="y=sin(x) et y=0.9",
       color=black,
        explicit((f(x)),x,-2*%pi,2*%pi),
       line_type=dots,
       explicit(0.9,x,-2*%pi,2*%pi)
       );
```

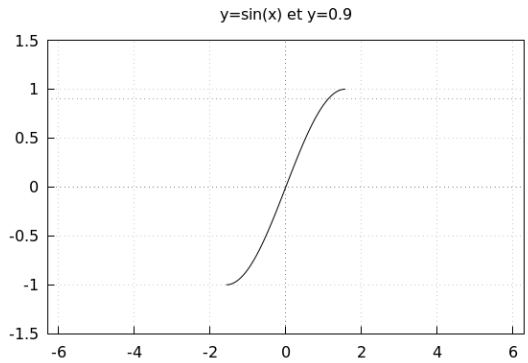


Nous voyons que les graphes se coupent en plusieurs endroits (en fait, il y a une infinité d'intersections). Il y a de nombreuses valeurs de x associées à la seule valeur $0,9$ de y, donc $f(x)$ n'est pas bijective, et par conséquent, n'est pas inversible. Bien sûr, cela est vrai pour toutes les autres valeurs de l'ensemble d'arrivée également.

Nous pouvons trouver des approximations des deux points d'intersection sur $[0,3]$ en utilisant wxMaxima pour résoudre l'équation correspondante sur un intervalle donné. Nous utilisons `find_root` pour trouver les *zéros* de la fonction $\sin(x) - 0.9$. Partout où cette fonction est nulle, nous obtenons une solution de $\sin(x) = 0.9$. Nous devons trouver une estimation de l'intervalle dans lequel se trouve chaque point d'intersection, en s'assurant qu'il n'existe qu'une seule intersection sur chaque intervalle :

```
(%i11) find_root(sin(x)-0.9,0,1.9);
(%o11) 1.119769514998634
(%i12) find_root(sin(x)-0.9,1.9,3);
(%o12) 2.021823138591159
```

Nous reproduisons le même graphique en utilisant le domaine restreint que nous avons défini précédemment pour la fonction sinus :



Avec la restriction du domaine à $[-\frac{\pi}{2}, \frac{\pi}{2}]$, nous voyons que chaque valeur y de $f(x)$ est "intersectée" exactement une fois, donc la fonction restreinte est bijective, et par conséquent inversible. En particulier, il n'y a qu'une seule valeur x pour laquelle $\sin(x) = 0.9$.

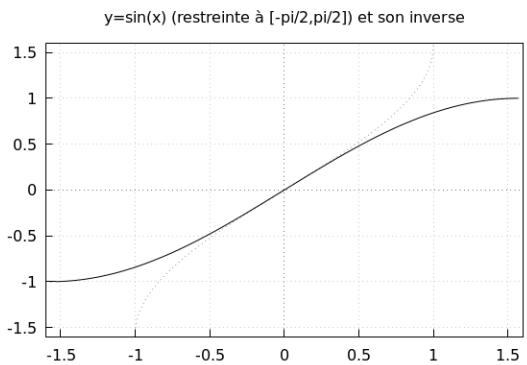
Maintenant, nous utilisons wxMaxima pour trouver $f^{-1}(y)$:

```
(%i13) solve(y=f(x),x);
       solve: using arc-trig functions to get a solution.
       Some solutions will be lost.
(%o14) [x=asin(y)]
```

wxMaxima nous indique que $f^{-1}(y) = \sin^{-1} y$. `asin` est la notation de wxMaxima pour arcsin (un synonyme de $\sin^{-1}$). wxMaxima a également souligné que « certaines solutions seront perdues » en raison de la restriction implicite du domaine de la fonction arc sinus.

Finalement, nous pouvons représenter graphiquement la restriction de la fonction $f(x)$ ainsi que son inverse afin d'observer la symétrie attendue :

```
(%i15) wxdraw2d(
       dimensions=[600,600],
       xaxis=true,
       yaxis=true,
       xrange=[-1.6,1.6],
       yrange=[-1.6,1.6],
       grid=true,
       title="y=sin(x) (restreinte à [-pi/2,pi/2]) et son inverse",
       color=black,
        explicit((f(x)),x,-%pi/2,%pi/2),
       line_type=dots,
       explicit(asin(x),x,-1,1)
       );
```



Une nouvelle fois, nous observons la symétrie par rapport à la droite d'équation $y = x$.

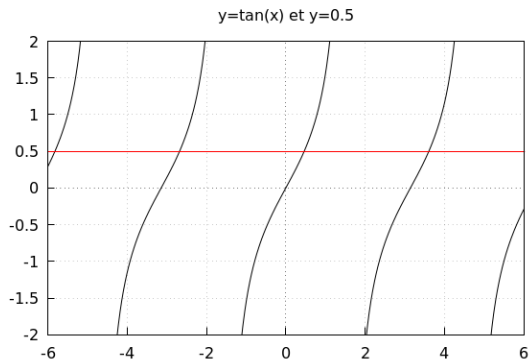
Exemple 1.8.3. Comme pour la fonction sinus, la restriction du domaine de définition classique pour la fonction tangente est $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Résolvez l'équation $\tan x = 0.5$ en appliquant la fonction tangente inverse, sous la condition que x se trouve quelque part dans l'intervalle $[\pi, 2\pi]$. Afin de trouver la valeur correcte de x, vous devrez utiliser la périodicité de la fonction tangente pour convertir « manuellement » la réponse donnée par la fonction tangente inverse. Tracez un graphique de $y = \tan x$ et $y = 0.5$ pour illustrer la relation entre les deux valeurs de x.

Nous commençons simplement en utilisant la commande de wxMaxima `atan` pour résoudre l'équation (une solution alternative consiste à utiliser `find_root` comme nous l'avons fait sur le dernier exemple) :

```
(%i16) atan(0.5);
(%o16) 0.46364760900081
```

Ce n'est certainement *pas* dans l'intervalle $[\pi, 2\pi]$, donc nous devons trouver « à la main » la solution souhaitée. Pour avoir une idée de l'intervalle dans lequel la solution pourrait se trouver, nous traçons un graphique de $y = \tan(x)$ et $y = 0.5$ et cherchons les intersections :

```
(%i17) wxdraw2d(
       xaxis=true,
       yaxis=true,
       yrange=[-2,2],
       xrange=[-6,6],
       grid=true,
       title="y=tan(x) et y=0.5",
       color=black,
        explicit((tan(x)),x,-6,6),
       color=red,
        explicit(0.5,x,-6,6)
       );
```



Nous constatons que les coordonnées de tous les points d'intersection se reproduisent avec la même périodicité que la fonction tangente elle-même. Pour trouver la solution souhaitée, nous ajoutons simplement π à la solution précédente. Pour obtenir une approximation décimale, nous utilisons la commande `float`, et pour vérifier notre réponse, nous évaluons à nouveau la fonction tangente (rappelons que `%` signifie « sortie précédente ») :

```
(%i18) atan(0.5)+%pi;
(%o18) %pi+0.46364760900081
(%i19) float(%);
(%o19) 3.605240262590599
(%i20) tan(%);
(%o20) 0.5
```

1.9 Exercices du Chapitre 1

1. Tracez la fonction rationnelle $f(x) = \frac{x^2-2x+1}{3x^3-2}$ ainsi que toutes les asymptotes horizontales et verticales (utilisez `limit` pour identifier l'asymptote horizontale). N'oubliez pas que l'asymptote verticale doit être définie *paramétriquement* comme indiqué dans l'Exemple 1.1.2.

   Note : Lorsque vous recherchez algébriquement l'asymptote verticale, vous trouverez des solutions superflues (nombres complexes), mais la solution *réelle* sera également incluse dans la liste.
2. Tracez une fonction sinus avec une période de π, une amplitude de 2.5, une ligne médiane $y = 1.5$ et un déphasage positif (vers la gauche) égal à 1/3 de la période.
3. Représente $f(x) = \cos x$ sur $[0, 4\pi]$ et utilise `find_root` pour trouver toutes les approximations décimales de toutes les racines sur cet intervalle. Quelles sont les valeurs *exactes* de ces racines ?
4. Tracer les fonctions $f(x) = e^x$, $g(x) = x^e$ et $h(x) = x^x$ sur l'intervalle $[0, 4]$. Quelle fonction croît le plus rapidement quand x devient très grand ? Laquelle croît le plus lentement ? Sans faire de calculs, notez le point d'intersection des trois graphes.
5. Quel déphasage positif minimal est nécessaire pour transformer une simple fonction sinus en une fonction cosinus ? Une fois que vous avez choisi cette valeur de ϕ, tracez le graphe de $f(x) = \sin(x + \phi)$ pour vérifier que la fonction cosinus est obtenue.
6. Les fonctions trigonométriques $\sec x$, $\csc x$ et $\cot x$ sont définies comme réciproques des fonctions trigonométriques basiques :
   $$\sec x = \frac{1}{\cos x} \qquad \csc x = \frac{1}{\sin x} \qquad \cot x = \frac{1}{\tan x}$$
   Réalisez trois graphiques différents pour explorer la relation entre chaque fonction et sa réciproque. Chaque graphique doit contenir la fonction trigonométrique de base (sin, cos ou tan) en pointillés rouges et la fonction réciproque (csc, sec ou cot) en ligne continue noire. Commentez la relation entre les zéros d'une fonction et les asymptotes verticales de l'autre.
7. Définissez la fonction $f(x) = x^2 - 5x$. Définissez $g_1(x) = f(-x)$ et $h_1(x) = g_1(x + 3)$; c'est-à-dire, effectuez une réflexion horizontale suivie d'un décalage horizontal. Tracez ces trois fonctions sur un même graphique.

   Ensuite, définissez $g_2(x) = f(x + 3)$ et $h_2(x) = g_2(-x)$; c'est-à-dire, effectuez le même décalage horizontal *avant* la même réflexion horizontale. Tracez ces trois fonctions sur le même graphique.

   Commentez l'importance de *l'ordre* pour les deux transformations.
8. En utilisant la même fonction $f(x)$ que dans le dernier exercice, définissez $g_1(x) = 2f(x)$ et $h_1(x) = g_1(-x)$ et tracez les trois fonctions sur le même graphique.

   Ensuite, définissez $g_2(x) = f(-x)$ et $h_2(x) = 2g_2(x)$ et tracez les trois fonctions sur un même graphique.

   L'ordre des transformations a-t-il de l'importance dans ce cas ? Quand pensez-vous que l'ordre a de l'importance, en général ?
9. Vérifier algébriquement que $f(x) = e^{-x^2}$ est une fonction paire. Représenter graphiquement $f(x)$ et vérifier que la courbe est bien symétrique comme attendu.

10. Définissez $f(x) = x^3$ et $g(x) = x^2 + 2x - 3$ et $h(x) = \frac{f(x)}{g(x)}$. Utilisez wxMaxima pour identifier les valeurs de x qui ne sont pas dans le domaine de $h(x)$. Tracez la fonction $h(x)$ ainsi que les asymptotes verticales.
11. Définissez $f(x) = e^{-0,1x^2}$ et $g(x) = 4\cos 3\pi x$. Tracez $h(x) = f(x) \cdot g(x)$ ainsi que les deux fonctions qui forment l'*enveloppe*, comme dans l'Exemple 1.6.3. Affichez $h(x)$ en noir et l'enveloppe en lignes rouges. Choisissez une fenêtre graphique qui illustre clairement pourquoi $h(x)$ est parfois appelé un « paquet d'ondes ».
12. La fonction logarithme naturel, $\ln x$, est la réciproque de la fonction exponentielle e^x. Vérifiez algébriquement que ces deux fonctions sont des fonctions réciproques l'une de l'autre, puis tracez un graphique des deux fonctions ainsi que de la droite $y = x$.
13. Trouvez des approximations décimales pour toutes les solutions de $\cos x = 0,7$ sur $[0, 2\pi]$. Tracez un graphique qui montre toutes les solutions comme les intersections de deux courbes. Enfin, utilisez la fonction arc cosinus pour résoudre l'équation, puis découvrez comment obtenir « manuellement » la deuxième solution à partir de la première (ce n'est *pas* aussi simple que de décaler la solution d'une période !). Remarque : le domaine « classique » pour la fonction cosinus est $[0, \pi]$.
14. Trouvez des approximations décimales pour les intersections de la fonction arc tangente $\tan^{-1}(x)$ et du cercle unité. Indice : le cercle unité devra être divisé en deux demi-cercles (supérieur et inférieur) afin de réussir le test de la "droite verticale". Faites un graphique montrant les intersections.
15. Toute fonction $f(x)$ peut être décomposée en somme de fonctions paire et impaire en utilisant le fait que $f(x) = \frac{1}{2}[f(x) + f(-x)] + \frac{1}{2}[f(x) - f(-x)]$ (le premier terme est pair et le second est impair). Définissez la fonction $f(x) = e^x$ et définissez $E(x) = \frac{1}{2}[f(x) + f(-x)]$ et $O(x) = \frac{1}{2}[f(x) - f(-x)]$. Vérifiez algébriquement que $E(x)$ est pair et que $O(x)$ est impair. Enfin, représentez graphiquement $E(x)$ et $O(x)$ pour vérifier que ces courbes possèdent bien les symétries attendues. $E(x)$ et $O(x)$ ont en fait des noms connus – faites quelques recherches et notez les noms de ces fonctions spéciales.
16. Comme présenté dans l'Exemple 1.6.2, une combinaison linéaire de deux fonctions sinusoïdales de même période donne une autre fonction sinusoïdale de même période. Si nous combinons deux fonctions sinusoïdales de périodes *différentes*, le résultat n'est plus sinusoïdal. Définissez $f(x) = \sin 4x$, $g(x) = \cos x$, et $h(x) = 3f(x) - g(x)$. Tracez les trois fonctions avec $h(x)$ en noir et les autres en gris. Quelles sont les périodes de $f(x)$ et $g(x)$? Quelle est la période de $h(x)$?
17. En fonction du dernier exercice, formulez une règle générale pour prédire la période d'une combinaison linéaire de fonctions sinusoïdales. Votre règle fonctionne-t-elle pour les fonctions $f(x) = \sin 2x$ et $g(x) = \sin 3x$? Réalisez un graphique pour tester votre règle, puis ajustez la règle si nécessaire.
18. En utilisant l'intervalle $[0, 4]$, trouvez une composition de fonctions qui donne une onde qui « ondule » plus rapidement à mesure que x augmente. Réalisez un "joli" graphique de votre fonction.

Chapitre 2

Limites

2.1 Utiliser les suites pour approcher les limites **56**
2.2 Commandes de wxMaxima pour les limites **62**
2.3 Explorer la définition formelle de la limite **67**
2.4 Exercices du chapitre 2 . . . . . **72**

Commandes essentielles de ce Chapitre

`for-do`	`divide`	`plus`
`print`	`find_root`	
`wxdraw2d`	`inf`	
`float`	`minf`	
`limit`	`minus`	

2.1 Utiliser les suites pour approcher les limites

Les définitions intuitives des **limites** sont données comme suit :

$\lim_{x\to a} f(x) = L$ signifie que "$f(x)$ devient aussi proche que l'on veut de L quand x s'approche de plus en plus de a."

$\lim_{x\to a^\pm} f(x) = L$ signifie que "$f(x)$ devient aussi proche que l'on veut de L quand x s'approche de plus en plus de a à partir de la droite/gauche (c'est à dire, pour $x > a$ ou $x < a$)."

$\lim_{x\to a^\pm} f(x) = \pm\infty$ signifie que "$f(x)$ devient aussi grand que l'on veut, soit positivement soit négativement quand x s'approche de plus en plus de a à partir de la droite/gauche."

$\lim_{x\to\pm\infty} f(x) = L$ signifie que "$f(x)$ devient aussi proche que l'on veut de L quand x est de plus en plus grand, soit positivement/négativement"

Nous pouvons explorer ces idées numériquement en utilisant wxMaxima pour évaluer $f(x)$ pour une séquence de valeurs de x qui se rapproche de plus en plus de la valeur étudiée. Dans les exemples suivants, nous utilisons des boucles `for-do` (également appelées simplement « do-loops ») pour générer les suites de valeurs x et $f(x)$ et afficher les résultats sous forme tabulaire. Plusieurs types de séquences sont illustrés, mais l'élément le plus important est que x s'approche de la valeur étudiée.

Exemple 2.1.1. Définir $f(x) = \frac{1}{x^2}$, puis évaluer chacune des limites suivantes en utilisant une boucle "do-loop" pour générer une suite appropriée de couples ordonnés. Représentez graphiquement $f(x)$ et commentez comment cette courbe est en rapport avec les limites trouvées par l'ordinateur.

a. $\lim_{x\to 0^+} f(x)$ b. $\lim_{x\to 2^-} f(x)$ c. $\lim_{x\to +\infty} f(x)$

a. Pour cette limite, nous utiliserons la séquence de valeurs pour x $.1, .01, .001, \ldots, 10^{-10}$. Notez comment la boucle do-loop est construite pour produire cette séquence : une formule est définie comme $\frac{1}{10^i}$, puis nous faisons varier i de 1 à 10. Il y a quelques guillemets et virgules supplémentaires utilisés afin de créer des espaces supplémentaires dans l'affichage des résultats. Bien que wxMaxima ne crée pas un tableau parfait pour nous, les résultats sont au moins lisibles.

```
(%i1)  F(x):=1/(x^2)$
(%i2)  (print("x "," "," "," f"),
         for i:1 thru 10 do
         (x: 1/(10^i),
          f: F(x),
          print(x,"","",f))
        );
```

```
 x     f
1/10  100
1/100  10000
1/1000  1000000
1/10000  100000000
1/100000  10000000000
1/1000000  1000000000000
1/10000000  100000000000000
1/100000000  10000000000000000
1/1000000000  1000000000000000000
1/10000000000  100000000000000000000
(%o2) done
```

Il semble que $f(x)$ soit croissante sans limite quand x s'approche de 0 par la droite, aussi nous concluons que $\lim_{x\to 0^+} f(x) = \infty$

b. Pour cette limite, nous utilisons la séquence $1.9, 1.99, \ldots, 1.9999999999$, qui est obtenue en utilisant la formule $2 - \frac{1}{10^i}$. Nous utilisons également `float` pour indiquer à wxMaxima de nous donner des approximations décimales (sinon, la sortie en fraction exacte rend difficile la compréhension de ce qui se passe).

```
(%i3) (print("x "," "," "," f"),
        for i:1 thru 10 do
        (x: float(2-1/(10^i)),
         f: float(F(x)),
         print(x,"","",f))
      );

 x    f
1.9  0.27700831024931
1.99  0.25251887578596
1.999  0.25025018762508
1.9999  0.25002500187513
1.99999  0.25000250001875
1.999999  0.25000025000019
1.9999999  0.250000025
1.99999999  0.2500000025
1.999999999  0.25000000025
1.9999999999  0.250000000025
(%o3) done
```

Nous voyons que $f(x)$ est de plus en plus proche de $0.25 = \frac{1}{4}$ quand x tend vers 2 par la gauche. Nou concluons que $\lim_{x\to 2^-} f(x) = \frac{1}{4}$.

c. Pour cette limite, nous utiliserons la séquence $10, 100, \ldots, 10^{10}$, qui est obtenue en se servant de la formule 10^i :

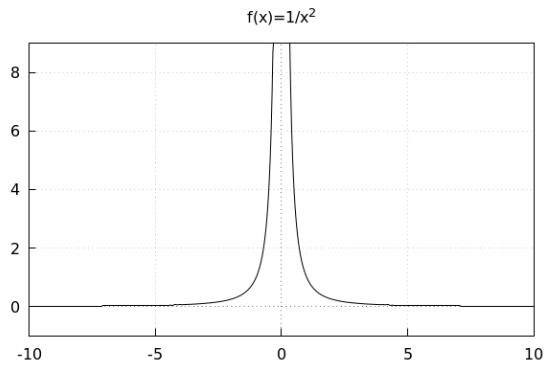
```
(%i4) (print("x "," "," "," f"),
       for i:1 thru 10 do
        (x:(10^i),
```

```
        f: float(F(x)),
        print(x,"","",f))
      );
x  f
10  0.01
100  1.0*10^-4
1000  9.9999999999999995*10^-7
10000  1.0*10^-8
100000  1.0*10^-10
1000000  9.9999999999999998*10^-13
10000000  1.0*10^-14
100000000  9.9999999999999998*10^-17
1000000000  1.0000000000000001*10^-18
10000000000  9.9999999999999995*10^-21
(%o4) done
```

Nous voyons que $f(x)$ devient très petit quand x devient très grand et positif, ainsi nous concluons que $\lim_{x\to\infty} f(x) = 0$.

Finalement, nous représentons la fonction $f(x) = \frac{1}{x^2}$:

```
(%i5) wxdraw2d(
      xaxis=true,
      yaxis=true,
      grid=true,
      xrange=[-10,10],
      yrange=[-1,9],
      title="f(x)=1/x^2",
      color=black,
      explicit((F(x)),x,-10,10)
      );
```



Nous constatons que $f(x)$ croît sans limite lorsque x s'approche de zéro par la droite, $f(x)$ s'approche de $\frac{1}{4}$ lorsque x s'approche de 2 par la gauche et $f(x)$ s'approche de 0 lorsque x devient grand et positif.

Exemple 2.1.2. Définissez $f(x) = \sin \frac{1}{2x}$ puis évaluez chacune des limites $\lim_{x\to 0^+} f(x)$ et $\lim_{x\to\infty} f(x)$ en utilisant une boucle "do-loop" pour générer une suite appropriée de paires ordonnées. Représentez graphiquement $f(x)$ et commentez la manière dont la courbe illustre les limites que vous avez calculées.

Cette fois, nous allons utiliser la séquence $0.20, 0.19, \ldots, 0.01$ en se servant de la formule $.21 - .01i$ et en la générant pour i variant de 1 à 20.

```
(%i6) F(x):=sin(1/(2*x))$
(%i7) (print("x "," "," "," f"),
       for i:1 thru 20 do
       (x: float(0.21-.01*i),
        f: float(F(x)),
        print (x, "","",f))
       );
x   f
0.2   0.59847214410396
0.19  0.48818920886648
0.18  0.35584199140107
0.17  0.19907720062779
0.16  0.016591892229347
0.15 -0.19056796287549
0.14 -0.4167216517535
0.13 -0.64769956634402
0.12 -0.85475260723884
0.11 -0.98609877449093
0.1  -0.95892427466314
0.09 -0.66510151497882
0.08 -0.033179216547556
0.07  0.75762841539272
0.06  0.88729410809469
0.05 -0.54402111088937
0.04 -0.066321897351194
0.03 -0.81844725315795
0.02 -0.13235175009776
0.01 -0.26237485370384
(%o8) done
```

Nous ne voyons aucune tendance se dégager parmi les valeurs de $f(x)$ quand x s'approche de 0, aussi nous concluons que la limite $\lim_{x\to 0^+} f(x)$ n'existe pas.

Pour évaluer la limite à l'infini, nous nous servons de la séquence $20, 40, 80, \ldots, 10 * 2^{10}$ générée par la formule $10 \cdot 2^i$ pour i variant de 1 à 10 :

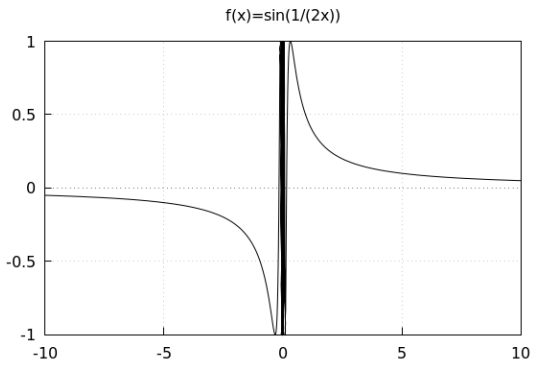
```
(%i8) (print("x "," "," "," f"),
       for i:1 thru 10 do
       (x: float(10*2^i),
        f: float(F(x)),
        print (x, "","",f))
       );
```

```
x        f
20.0     0.024997395914712
40.0     0.01249967448171
80.0     0.0062499593099753
160.0    0.0031249949137395
320.0    0.0015624993642172
640.0    7.8124992052714282*10^-4
1280.0   3.9062499006589261*10^-4
2560.0   1.9531249875823659*10^-4
5120.0   9.7656249844779578*10^-5
10240.0 4.8828124980597446*10^-5
(%o9) done
```

Nous voyons que les valeurs de $f(x)$ s'approchent de plus en plus de la valeur zéro, aussi nous en déduisons que $\lim_{x\to\infty} \sin \frac{1}{2x} = 0$

Finalement, nous représentons graphiquement $f(x)$ pour voir comment nos limites se situent par rapport à la courbe de $\sin \frac{1}{2x}$:

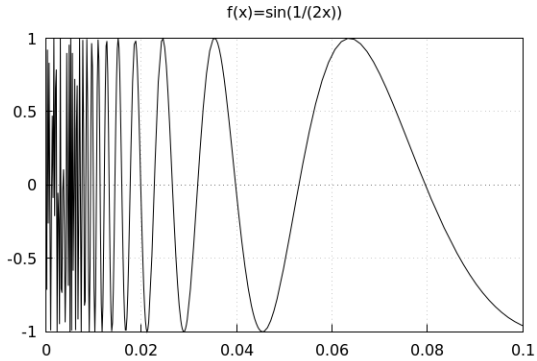
```
(%i9) wxdraw2d(
       xaxis=true,
       yaxis=true,
       grid=true,
       xrange=[-10,10],
       yrange=[-1,1],
       title="f(x)=sin(1/(2x))",
       color=black,
       explicit((F(x)),x,-10,10)
       );
```



Nous observons l'asymptote horizontale sur l'axe des x, indiquant que $\lim_{x\to\infty} \sin \frac{1}{2x} = 0$. Nous constatons également qu'il y a beaucoup de "tracés" concentrés près de l'axe des y, aussi nous réalisons un second graphique pour mettre en évidence ce comportement intéressant.

```
(%i10) wxdraw2d(
       xaxis=true,
```

```
yaxis=true,
grid=true,
xrange=[0,.1],
yrange=[-1,1],
title="f(x)=sin(1/(2x))",
color=black,
explicit((F(x)),x,-10,10)
);
```



Nous voyons que $f(x)$ continue simplement à osciller de plus en plus vite à mesure que x se rapproche de zéro. $\lim_{x\to 0} f(x)$ n'existe pas car $f(x)$ ne se rapproche jamais d'une valeur unique lorsque x se rapproche de 0.

2.2 Commandes de wxMaxima pour les limites

La commande `limit` est utilisée pour calculer les limites à gauche, à droite et "normales" dans wxMaxima (la limite "normale" existe si les limites à droite et à gauche ont toutes les deux la même valeur).

Exemple 2.2.1. Définir la fonction $f(x) = \frac{3x-5}{\sqrt{x^2}}$. Calculer les limites en $\pm\infty$, et calculer ensuite les limites à droite et à gauche de l'asymptote verticale. Utilisez vos résultats pour représenter graphiquement $f(x)$ ainsi que toutes ses asymptotes en rouge.

Nous voyons que le dénominateur s'annule (mais pas le numérateur) en $x = 0$, donc cela devrait correspondre à l'asymptote verticale. Notez que `inf` signifie ∞, `minf` signifie $-\infty$, `minus` indique une limite à gauche, et `plus` indique une limite à droite.

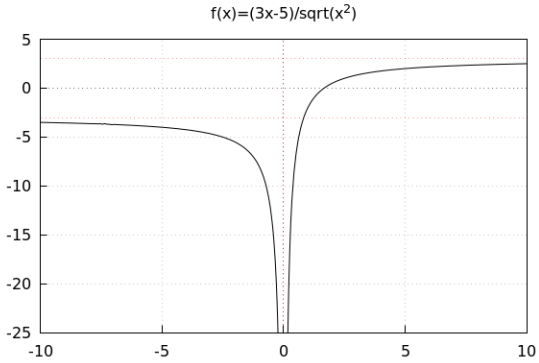
```
(%i1) f(x):=(3*x-5)/sqrt(x^2)$
(%i2) limit(f(x),x,minf);
(%o2) -3
(%i3) limit(f(x),x,inf);
(%o3) 3
(%i4) limit(f(x),x,0,minus);
(%o4) -inf
(%i5) limit(f(x),x,0,plus);
(%o5) -inf
```

Les limites en $\pm\infty$ indiquent que $f(x)$ possède deux asymptotes horizontales, $y = -3$ et $y = +3$. Les limites à gauche et à droite en $x = 0$ sont toutes deux $-\infty$, donc il y a une asymptote verticale à cet endroit. Les deux limites unilatérales coïncident en $x = 0$, donc nous pouvons conclure que la limite "normale" existe et devrait également valoir $-\infty$:

```
(%i6) limit(f(x),x,0);
(%o6) -inf
```

Finalement, nous représentons $f(x)$ avec toutes ses asymptotes :

```
(%i7) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      title="f(x)=(3x-5)/sqrt(x^2)",
      xrange=[-10,10],
      yrange=[-25,5],
      color=black,
       explicit((f(x)),x,-10,10),
      color=red,
      line_type=dots,
       explicit(3,x,-10,10),
       explicit(-3,x,-10,10),
       parametric(0,t,t,-25,5)
      );
```

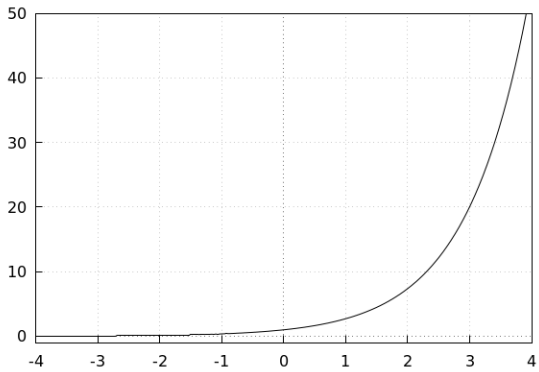


Exemple 2.2.2. Calculez les limites en $\pm\infty$ pour la fonction $f(x) = e^x$, puis tracez $f(x)$ pour vérifier que le tracé de la fonction correspond aux valeurs trouvées pour ces limites.

```
(%i8) f(x):=%e^(x)$
(%i9) limit(f(x),x,minf);
(%o9) 0
(%i10) limit(f(x),x,inf);
(%o10) inf
```

Nous obtenons $\lim_{x\to-\infty} f(x) = 0$ et $\lim_{x\to+\infty} f(x) = +\infty$. Représentons graphiquement $f(x)$

```
(%i11) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[-4,4],
       yrange=[-1,50],
       color=black,
       explicit(f(x),x,-4,4)
       );
```



Nous voyons que l'axe des abscisses est une asymptote pour les grandes valeurs négatives de x, et que la courbe croit vers $+\infty$ en sortant du cadre pour les grandes valeurs de x.

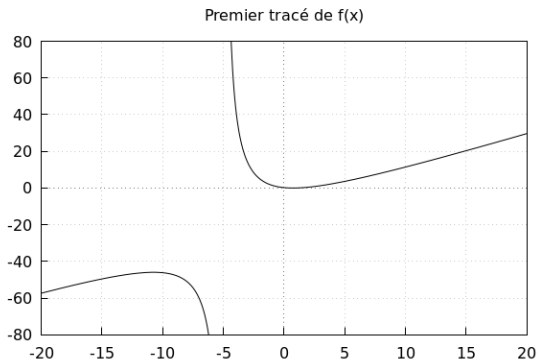
Exemple 2.2.3. Définir $f(x) = \frac{2x^2-3x+1}{x+5}$. Trouver les limites à gauche et à droite pour la valeur de l'asymptote verticale et calculer les limites en $\pm\infty$. Représenter graphiquement $f(x)$ en incluant l'asymptote verticale et l'asymptote *oblique*.

L'asymptote verticale devrait apparaître pour $x = -5$, car le dénominateur s'annule pour cette valeur.

```
(%i12) f(x):=(2*x^2-3*x+1)/(x+5)$
(%i13) limit(f(x),x,minf);
(%o13) -inf
(%i14) limit(f(x),x,inf);
(%o14) inf
(%i15) limit(f(x),x,-5,minus);
(%o15) -inf
(%i16) limit(f(x),x,-5,plus);
(%o16) inf
```

Nous réalisons un graphique préliminaire afin de chercher l'asymptote oblique :

```
(%i17) wxdraw2d(
       grid=true,
       yaxis=true,
       xaxis=true,
       xrange=[-20,20],
       yrange=[-80,80],
       color=black,
       title="Premier tracé de f(x)",
       explicit((f(x)),x,-20,20)
       );
```



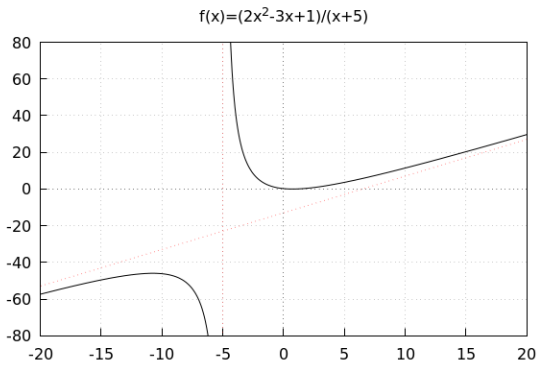
Nous voyons que $f(x)$ semble asymptotique à une droite lorsque x devient très grand ou très petit. Nous pouvons mettre en évidence l'asymptote oblique en effectuant une division polynomiale sur $f(x)$ comme indiqué ci-dessous :

```
(%i18) p:2*x^2-3*x+1$
       q:x+5$
(%i19) divide(p,q);
(%o19) [2*x-13,66]
```

wxMaxima indique que $f(x) = 2x - 13 + \frac{66}{x+5}$. Quand x devient très grand, le terme restant s'approche de 0 et la droite d'équation $y = 2x - 13$ est bien une asymptote oblique correspondant à la courbe de $f(x)$.

En représentant graphiquement $f(x)$ et toutes ses asymptotes, nous obtenons :

```
(%i20) wxdraw2d(
       grid=true,
       yaxis=true,
       xaxis=true,
       xrange=[-20,20],
       yrange=[-80,80],
       title="f(x)=(2x^2-3x+1)/(x+5)",
       color=black,
        explicit((f(x)),x,-20,20),
       color=red,
       line_type=dots,
        explicit((2*x-13),x,-20,20),
        parametric(-5,t,t,-80,80)
       );
```



Exemple 2.2.4. Définir $f(x) = \frac{1-\cos x}{x}$. Calculez $\lim_{x\to 0^+} f(x)$ et réalisez un graphique afin de vérifier votre réponse.

Il s'agit d'une limite intéressante, car le numérateur et le dénominateur s'approchent tous les deux de zéro :

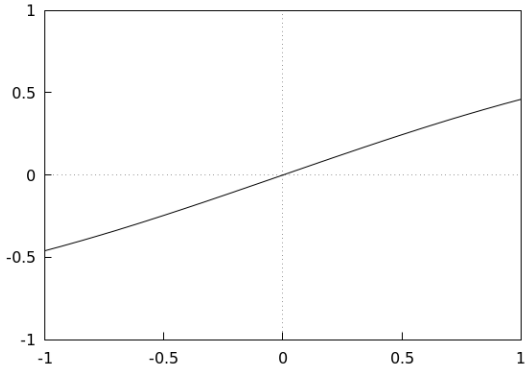
```
(%i21) limit(1-cos(x),x,0,plus);
(%o21) 0
(%i22) limit(x,x,0,plus);
(%o22) 0
```

Comme 0/0 est une forme indéterminée, le calcul "à la main" nécessiterait des transformations algébriques. wxMaxima répond rapidement à la question :

```
(%i23) limit((1-cos(x))/x,x,0,plus);
(%o23) 0
```

Nous vérifions rapidement avec un graphique :

```
(%i24) wxdraw2d(
       xaxis=true,
       yaxis=true,
       color=black,
       xrange=[-1,1],
       yrange=[-1,1],
       explicit((1-cos(x))/x,x,-1,1)
       );
```



La courbe montre clairement que $f(x) \to 0$ quand $x \to 0$ (aussi bien à droite qu'à gauche).

2.3 Explorer la définition formelle de la limite

La **limite ordinaire** d'une fonction $f(x)$ en $x = a$ est définie précisément comme suit :

$$\lim_{x \to a} f(x) = L$$

si et seulement si, pour tout $\epsilon > 0$, il existe un $\delta > 0$ tel que $|f(x) - L| < \epsilon$ lorsque $|x - a| < \delta$.

C'est-à-dire, peu importe la petitesse de notre "tolérance" verticale ϵ autour de L, nous pouvons toujours trouver un δ suffisamment petit pour que $f(x)$ soit à moins de ϵ de L lorsque x est à moins de δ de a. Cela correspond à la définition intuitive selon laquelle « rendre x très proche de a entraîne que $f(x)$ est très proche de L ».

Une **limite à l'infini** a une définition similaire :

$$\lim_{x \to \infty} f(x) = L$$

si et seulement si, pour tout $\epsilon > 0$, il existe un $N > 0$ tel que $|f(x) - L| < \epsilon$ lorsque $x > N$. C'est-à-dire, peu importe la petitesse de notre "tolérance" verticale ϵ autour de L, nous pouvons toujours trouver un N suffisamment grand pour que $f(x)$ soit à moins de ϵ de L lorsque x est supérieur à N. Cela correspond à la définition intuitive selon laquelle « rendre x très grand garantit que $f(x)$ est très proche de L ».

Enfin, une **limite infinie** peut être définie précisément comme suit :

$$\lim_{x \to a} f(x) = \infty$$

si et seulement si, pour tout $M > 0$, il existe un $\delta > 0$ tel que $f(x) > M$ lorsque $|x - a| < \delta$.
C'est-à-dire, peu importe la grande valeur M de y que nous choisissons, nous pouvons toujours trouver un δ suffisamment petit pour que $f(x)$ dépasse M lorsque x est à moins de δ de a. Cela correspond à la définition intuitive selon laquelle « rendre x très proche de a rend $f(x)$ très grand ».

La définition mathématique de la limite est très utile pour prouver des théorèmes en analyse formelle. Ici, nous utiliserons simplement wxMaxima pour améliorer notre *intuition* sur la définition dite « $\epsilon - \delta$ ».

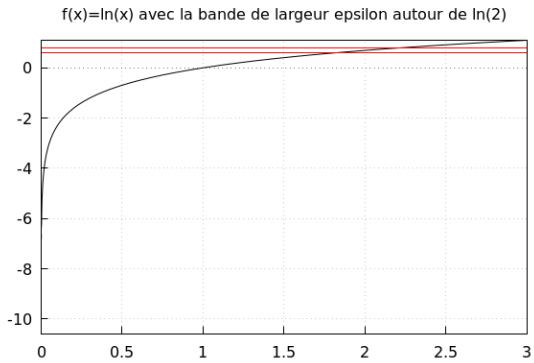
Exemple 2.3.1. Définir $f(x) = \ln(x)$. Utiliser wxMaxima pour vérifier que $\lim_{x \to 2} f(x) = \ln 2$. En appliquant la définition formelle de la limite, choisissez $\epsilon = 0.1$ comme "tolérance" verticale sur $\ln 2$. Pour cette valeur de ϵ, trouvez une valeur approchée pour δ telle que $|f(x) - \ln 2| < \epsilon$ quand $|x - 2| < \delta$.

On rappelle que le symbole utilisé par wxMaxima pour le logarithme népérien est `log` :

```
(%i1) f(x):=log(x);
(%o1) f(x):=log(x)
(%i2) limit(f(x),x,2);
(%o2) log(2)
```

Nous avons vérifié que $\lim_{x\to 2} f(x) = \ln 2$, donc la définition formelle de cette limite doit être satisfaite. Pour $\epsilon = 0,1$, nous pouvons approfondir l'analyse en traçant le graphe de $f(x)$ et en utilisant des lignes horizontales pour délimiter la bande de largeur ϵ autour de $\ln(2)$. La valeur de y supérieure est $\ln(2) + 0,1$ et la valeur inférieure est $\ln(2) - 0,1$.

```
(%i3) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      title="f(x)=ln(x) avec la bande de largeur epsilon autour de ln(2)",
      color=black,
       explicit((f(x)),x,0,3),
      color=red,
       explicit((log(2)+.1),x,0,3),
       explicit((log(2)-.1),x,0,3)
       );
```



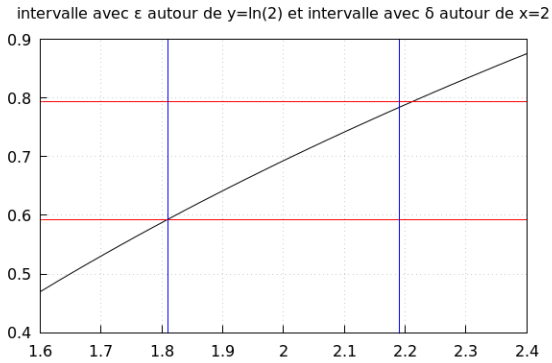
Maintenant, nous devrions être en mesure de trouver un δ suffisamment petit pour imposer que les valeurs de $f(x)$ se situent dans cette bande verticale étroite lorsque x est à moins de δ de 2. Pour obtenir des informations précises, nous trouvons les intersections entre $f(x)$ et les lignes horizontales :

```
(%i4) float(solve(f(x)=log(2)+.1,x));
      rat: replaced -0.1 by -1/10 = -0.1
(%o4) [x=2.210341836151295]
(%i5) float(solve(f(x)=log(2)-.1,x));
      rat: replaced 0.1 by 1/10 = 0.1
(%o5) [x=1.809674836071919]
```

À trois décimales près, nous pouvons dire que lorsque x se situe dans l'intervalle $[1,810, 2,210]$, $f(x)$ appartient à l'intervalle $[\ln(2) - 0,1, \ln(2) + 0,1]$. Nous devons encore trouver le δ qui correspond à la définition formelle, donc nous choisissons la plus petite des deux distances entre $x = 2$ et les extrémités de l'intervalle $[1,810, 2,210]$; c'est-à-dire, $\delta = 0,19$. Maintenant, nous pouvons dire que $f(x)$ est à moins de $0,1$ de $\ln(2)$ lorsque x est à moins de $0,19$ de 2, et la définition mathématique est satisfaite pour $\epsilon = 0,1$.

Finalement, nous devons redessiner le graphique avec les droites $x = 2 - \delta = 1.81$ et $x = 2 + \delta = 2.19$ que nous ajoutons afin d'illustrer la manière dont la définition formelle est satisfaite :

```
(%i6) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[1.6,2.4],
      yrange=[.4,.9],
      title="intervalle avec epsilon autour de y=ln(2) et
      intervalle avec delta autour de x=2",
      color=black,
       explicit((f(x)),x,0,3),
      color=red,
       explicit((log(2)+.1),x,0,3),
       explicit((log(2)-.1),x,0,3),
      color=blue,
       parametric(1.81,t,t,.4,.9),
       parametric(2.19,t,t,.4,.9)
       );
```



Chaque fois que x se situe dans l'intervalle $[2-0,19, 2+0,19]$, nous constatons que $f(x)$ se situe dans l'intervalle $[\ln(2)-0,1, \ln(2)+0,1]$. Bien sûr, si nous choisissions un ϵ encore plus petit, nous pourrions facilement trouver un δ plus petit pour imposer que les valeurs de y se trouvent à moins de ϵ de $\ln(2)$.

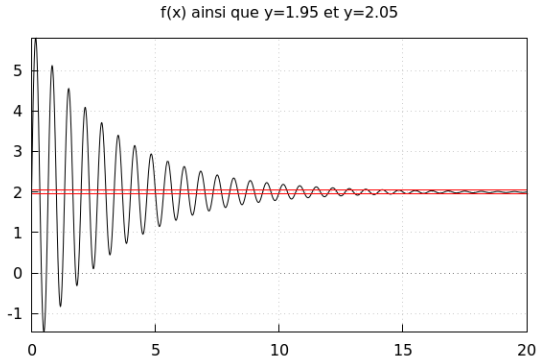
Exemple 2.3.2. Définir $f(x) = 4e^{-0.3\cdot x} \cdot \sin(3\pi x) + 2$. Utiliser wxMaxima pour calculer la limite en plus l'infini. Une fois que cette limite sera trouvée, illustrez l'approche théorique de la limite en fixant $\epsilon = .05$.

En premier, nous calculons la limite :

```
(%i7) f(x):=4*%e^(-.3*x)*sin(3*%pi*x)+2$
(%i8) limit(f(x),x,inf);
       rat: replaced -0.3 by -3/10 = -0.3
(%o8) 2
```

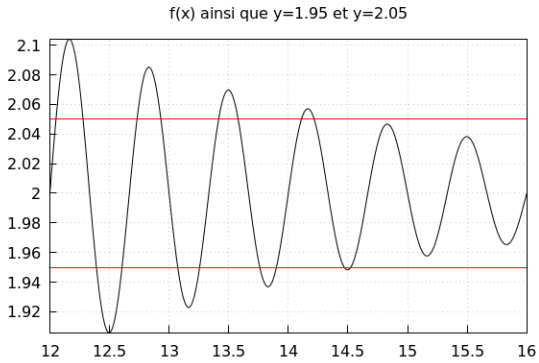
Nous constatons que $\lim_{x\to\infty} f(x) = 2$. Ensuite, nous traçons $f(x)$ avec la bande représentant notre "tolérance" verticale de $\epsilon = 0,05$ autour de $y = 2$:

```
(%i9) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       title="f(x) ainsi que y=1.95 et y=2.05",
       color=black,
        explicit((f(x)),x,0,20),
       color=red,
        explicit(1.95,x,0,20),
        explicit(2.05,x,0,20)
       );
```



En rappelant la définition formelle d'une limite à l'infini, nous cherchons un N suffisamment grand pour garantir que $f(x)$ est à moins de $0,05$ de 2 dès lors que $x > N$. Il est intéressant de trouver le plus petit N possible, et nous pouvons obtenir une bonne approximation en examinant visuellement le tracé de $f(x)$. Nous recherchons l'intersection entre $f(x)$ et les droites horizontales $y = 1,95$ et $y = 2,05$, avant que $f(x)$ ne soit définitivement inclus entre ces deux droites. Nous zoomons pour mieux voir cette intersection :

```
(%i10) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       title="f(x) ainsi que y=1.95 et y=2.05",
       color=black,
        explicit((f(x)),x,12,16),
       color=red,
        explicit(1.95,x,12,16),
        explicit(2.05,x,12,16)
       );
```



La dernière intersection semble se produire juste à droite de $14,5$. Nous pouvons la déterminer à plusieurs décimales près en calculant le point d'intersection avec wxMaxima :

```
(%i11) find_root(f(x)-1.95,x,14.5,15);
(%o11) 14.52360146112245
```

À trois décimales près, nous concluons que lorsque $x > 14,524$, $f(x)$ est à moins de $0,05$ de 2. Encore une fois, nous pouvons voir qu'un ϵ plus petit nécessiterait simplement de trouver un N plus grand.

2.4 Exercices du chapitre 2

1. Calculer $\lim_{x\to\frac{\pi}{3}^+}\cos x$ en utilisant une boucle de 10 étapes avec la formule $(1+\frac{1}{2^i})\frac{\pi}{3}$.
2. Calculer $\lim_{x\to-3^-}\frac{2}{x+3}$ en utilisant une boucle de 20 étapes (choisissez votre propre fonction pour créer la boucle).
3. Utiliser la commande `limit` pour trouver toutes les asymptotes de $f(x)=\frac{3-5x^2}{x^2-8}$. Représenter graphiquement $f(x)$ en y incluant toutes les asymptotes en rouge.
4. Utiliser la commande `limit` pour calculer $\lim_{x\to3}\frac{\sqrt{x}-\sqrt{3}}{x-3}$.
5. Utiliser la commande `limit` pour calculer $\lim_{n\to\infty}(1+\frac{1}{n})^n$ Commenter le résultat obtenu.
6. Pour la fonction de l'Exemple 2.3.2, $f(x)=4e^{-0.3\cdot x}\cdot\sin(3\pi x)+2$, nous pouvons étudier $\lim_{x\to\infty}f(x)$ en utilisant le "théorème des gendarmes". Remplacez les valeurs maximale et minimale de la fonction sinus afin d'obtenir une courbe qui borne $f(x)$ par le haut (appelez-la `UPPER(x)`) et une courbe qui borne $f(x)$ par le bas (appelez-la `LOWER(x)`). Tracez $f(x)$ avec ces deux courbes en rouge. Enfin, utilisez `limit` pour montrer que les bornes supérieure et inférieure tendent toutes deux vers 2 lorsque $x\to\infty$ (le théorème des gendarmes vous indique que $f(x)\to2$ également).
7. La *dérivée* d'une fonction $f(x)$ peut s'écrire $f'(x)=\lim_{h\to0}\frac{f(x+h)-f(x)}{h}$. Utiliser cette définition pour trouver la dérivée de $\cos x$ avec wxMaxima.
8. Une fonction $f(x)$ est *continue* en $x=a$ si $\lim_{x\to a}f(x)=f(a)$. Utiliser wxMaxima pour montrer que $f(x)=\tan x$ est continue en $x=\frac{\pi}{3}$.
9. Le *Théorème des Valeurs Intermédiaires* stipule que, pour toute fonction continue sur $[a,b]$ pour laquelle $f(a)\neq f(b)$, et pour tout nombre u compris entre $f(a)$ et $f(b)$, il existe un c dans (a,b) tel que $f(c)=u$. En d'autres termes, si $f(x)$ est continue, alors toutes les valeurs y entre $f(a)$ et $f(b)$ doivent apparaître sur l'intervalle $[a,b]$. Définissons $f(x)=e^{-x^2}$. Appliquons le Théorème des Valeurs Intermédiaires sur l'intervalle $[0,1]$ en utilisant $u=0,5$; c'est-à-dire, trouvons c.
10. En utilisant l'exemple 2.3.1 comme modèle, appliquez la définition formelle de la limite à $\lim_{x\to\frac{\pi}{2}}(\sin x+\cos x)$. On prendra $\epsilon=.001$.
11. En prenant $M=1000$, appliquez la définition formelle de la limite pour illustrer que $\lim_{x\to0}\frac{1}{x^2}=\infty$.
12. La *fonction échelon* (qui est aussi appelée la fonction Θ) est une fonction par morceaux définie par $\Theta(x)=\begin{cases}0 & x<0\\ 1 & x\geq0\end{cases}$. $\Theta(x)$ est implémentée dans wxMaxima comme `unit_step(x)`. Calculer $\lim_{x\to0^-}\Theta(x)$, $\lim_{x\to0^+}\Theta(x)$ et $\lim_{x\to0}\Theta(x)$. Citer un théorème pour expliquer pourquoi cette dernière limite n'existe pas.

Chapitre 3

Introduction aux dérivées

3.1 Le problème de la tangente **74**
3.1.1 La tangente à partir d'une suite de droites 74
3.1.2 La tangente comme une limite 76
3.2 Les commandes pour la dérivation dans wxMaxima **78**
3.3 Produits, Quotients et Combinaisons Linéaires **81**
3.4 Dérivation des fonctions composées **83**
3.4.1 La règle de la Chaine 83
3.4.2 Différentiation logarithmique 84
3.4.3 Différentiation implicite 85
3.5 Exercices du Chapitre 3 . . . . . **87**

Commandes essentielles de ce Chapitre

`wxdraw2d`	`solve`	`trigexpand`
`makelist`	`float`	`trigsimp`
`dimensions`	`ev`	`exponentialize`
`limit`	`depends`	`expand`
`diff`	`subst`	

3.1 Le problème de la tangente

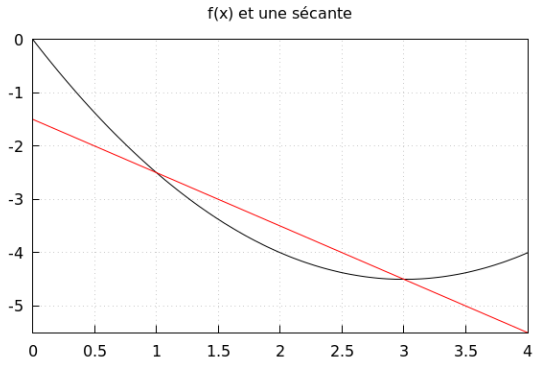
3.1.1 La tangente à partir d'une suite de droites

Une **sécante** est une droite reliant deux points situés sur la courbe d'une fonction, $(a, f(a))$ et $(b, f(b))$. Nous pouvons approximer la pente de la **tangente** au point $(a, f(a))$ en calculant les pentes des droites sécantes sur des intervalles de plus en plus petits, avec $x = a$ fixé à une extrémité.

Exemple 3.1.1. Pour la fonction $f(x) = 0.5x^2 - 3x$, trouvez l'équation de la sécante reliant $(1, f(1))$ to $(3, f(3))$. Réalisez un graphique de $f(x)$ en incluant le graphe de la sécante.

Rappelons que la pente d'une droite reliant deux points $(a, f(a))$ et $(b, f(b))$ est donnée par $m = \frac{f(b)-f(a)}{b-a}$. Nous pouvons utiliser ce résultat pour obtenir une équation de la sécante cherchée : $y - f(a) = \frac{f(b)-f(a)}{b-a} \cdot (x - a)$. Nous résolvons cette équation pour y lorsque nous définissons la droite sécante dans wxMaxima. La droite sécante est définie comme une fonction de trois variables, x, a et b, de sorte que nous pouvons rapidement trouver n'importe quelle droite sécante en changeant simplement a et b à l'intérieur de `wxdraw2d`.

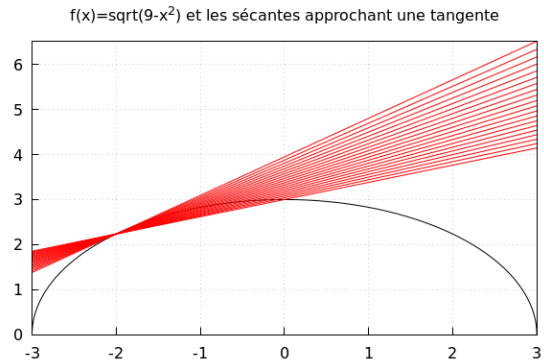
```
(%i1) f(x):=0.5*x^2-3*x$
(%i2) SECANT(x,a,b):=((f(b)-f(a))/(b-a))*(x-a)+f(a)$
(%i3) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      title="f(x) et une sécante",
      color=black,
       explicit((f(x)),x,0,4),
      color=red,
       explicit((SECANT(x,1,3)),x,0,4)
      );
```



Exemple 3.1.2. Pour $f(x) = \sqrt{9 - x^2}$, utiliser une boucle "do-loop" pour calculer les pentes des sécantes sur $[-2, 0], [-2, -.1], [-2, -.2], \ldots, [-2, -1.9]$. Utiliser `makelist` pour dessiner les sécantes en rouge sur le graphique représentant $f(x)$ en noir.

Nous utilisons la formule $0.1 - 0.1 \cdot i$ pour générer la suite de points à droite de -2. Nous utilisons `for-do` pour calculer la suite des pentes et `makelist` pour calculer les équations de toutes les droites sécantes. Notez que la liste doit être prête à être interprétée dans `wxdraw2d` – nous incluons `explicit` et l'intervalle pour x dans chaque élément de la liste. Enfin, nous utilisons `dimensions` pour faire apparaître correctement le demi-cercle $f(x)$.

```
(%i4) f(x):=sqrt(9-x^2)$
(%i5) SLOPE(a,b):=(f(b)-f(a))/(b-a)$
(%i6) SECANT(x,a,b):=SLOPE(a,b)*(x-a)+f(a)$
(%i7) (print ("interval",".........","slope"),
       for i:1 thru 20 do
       (S:float(SLOPE(-2,0.1-0.1*i)),
        print("[-2,",0.1-0.1*i,"]",".........",S))
       );

 interval.........slope
[-2, 0.0].........0.38196601125011
[-2,-0.1].........0.40119204874379
[-2,-0.2].........0.42069885106631
[-2,-0.3].........0.44052607871769
[-2,-0.4].........0.46071610747744
[-2,-0.5].........0.48131460936668
[-2,-0.6].........0.50237122417145
[-2,-0.7].........0.52394034743321
[-2,-0.8].........0.54608206788367
[-2,-0.9].........0.56886329704641
[-2,-1.0].........0.5923591472464
[-2,-1.1].........0.61665463321198
[-2,-1.2].........0.64184679934214
[-2,-1.3].........0.66804741345624
[-2,-1.4].........0.69538642464088
[-2,-1.5].........0.72401646770705
[-2,-1.6].........0.75411882647529
[-2,-1.7].........0.78591147120622
[-2,-1.8].........0.81966011250105
[-2,-1.9].........0.8556937574899
(%o7) done

(%i8) L:makelist(explicit(SECANT(x,-2,.1-.1*i),x,-3,3),i,1,20)$
(%i9) wxdraw2d(
      grid=true,
      xaxis=true,
      dimensions=[600,600],
      title="f(x)=sqrt(9-x^2) et les sécantes approchant une tangente",
      color=black,
       explicit(f(x),x,-3,3),
      color=red,
       L
      );
```



À mesure que l'intervalle devient de plus en plus petit (avec -2 toujours du côté gauche de l'intervalle), les droites sécantes tournent dans le sens inverse des aiguilles d'une montre et se rapprochent de plus en plus de la droite tangente en $x = -2$. La pente de la droite tangente semble être légèrement inférieure à 1, ce qui concorde avec le résultat de notre boucle "do-loop".

3.1.2 La tangente comme une limite

Le processus utilisé dans l'exemple précédent peut être généralisé pour trouver la pente exacte de la droite tangente à $f(x)$ en $x = a$. Nous calculons la pente de la droite sécante reliant $(a, f(a))$ et $(a + h, f(a + h))$, puis nous prenons la limite lorsque $h \to 0$. Une approche alternative consiste à prendre la pente de la droite sécante reliant $(a, f(a))$ et $(x, f(x))$, puis à prendre la limite lorsque $x \to a$. La pente de la droite tangente en $x = a$ est appelée la **dérivée** de $f(x)$ en $x = a$, notée $f'(a)$:

$$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h} = \lim_{x \to a} \frac{f(x) - f(a)}{x - a}$$

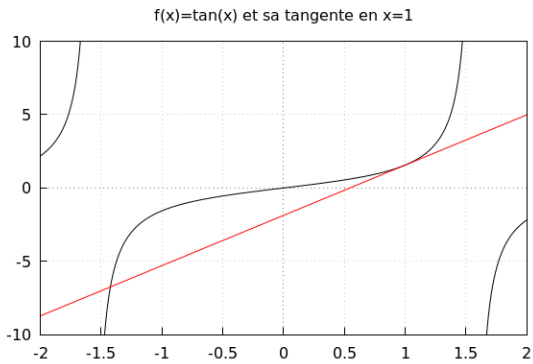
Exemple 3.1.3. Définir $f(x) = \tan x$, puis utiliser la définition de la dérivée comme "limite" pour calculer $f'(1)$. Finalement, représenter graphiquement $f(x)$ en incluant la tangente au point $(1, f(1))$.

Nous pouvons utiliser l'une ou l'autre des "définitions par la limite" pour trouver $f'(1)$:

```
(%i10) f(x):=tan(x)$

(%i11) limit(((f(1+h)-f(1))/(h)),h,0);
(%o11) 1/cos(1)^2

(%i12) limit(((f(x)-f(1))/(x-1)),x,1);
(%o12) 1/cos(1)^2
```

Nous voyons que $f'(1) = \frac{1}{\cos^2 1}$. L'équation de la tangente est ensuite calculée en utilisant le fait que la pente est égale à $f'(1)$ au point $(1, f(1))$ (nous résolvons l'expression de l'équation d'une droite avec sa pente en isolant y).

```
(%i13) SLP:%$
(%i14) TNGT(x):=SLP*(x-1)+f(1)$
(%i15) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[-2,2],
       yrange=[-10,10],
       title="f(x)=tan(x) et sa tangente en x=1",
       color=black,
        explicit((f(x)),x,-2,2),
       color=red,
        explicit((TNGT(x)),x,-2,2)
       );
```



Nous voyons que la tangente se comporte bien comme attendue en $x = 1$.

Exemple 3.1.4. Définir $f(x) = x^n$, puis utiliser la définition de la dérivée par la limite pour trouver une formule générale pour $f'(x)$ (cette formule est quelquefois appelée "la règle des puissances").

Ceci est juste un exemple de la manière dont les formules sont déterminées pour « le formulaire des dérivées » : chaque règle provient du calcul d'une limite. Nous évaluons la dérivée en $x = X$ pour souligner que le résultat est valable pour *n'importe* quel nombre réel.

```
(%i16) f(x):=x^n$
(%i17) limit(((f(x)-f(X))/(x-X)),x,X);
(%o18) n*X^(n-1)
```

Nous concluons que, si $f(x) = x^n$, alors $f'(x) = n \cdot x^{n-1}$ pour toute valeur de x. Nous pouvons considérer $f'(x)$ comme une nouvelle *fonction* de x qui nous indique la pente de $f(x)$ pour toute valeur de x.

3.2 Les commandes pour la dérivation dans wxMaxima

Comme nous pouvons nous y attendre, wxMaxima peut rapidement trouver la dérivée d'une fonction en utilisant la commande `diff` :

Exemple 3.2.1. Pour la fonction $f(x) = \tan^{-1} x$:

a. calculer $f'(1.5)$ b. trouver tous les x tels que $f'(x) = \frac{1}{2}$

a. Noter que nous avons à utiliser `''` quand nous définissons une nouvelle fonction en tant que dérivée d'une fonction précédente. Cela permet à wxMaxima de calculer l'expression générale de $f'(x)$ *avant* que le programme ne tente de substituer des valeurs particulières pour x (substituer $x = 1.5$ dans `diff(f(x),x)` provoque une erreur).

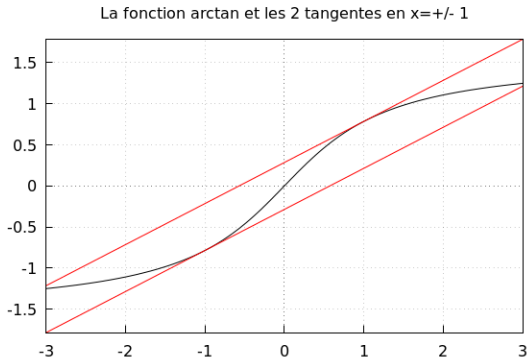
```
(%i1) f(x):=atan(x)$
(%i2) f_prime(x):=''(diff(f(x),x));
(%o2) f_prime(x):=1/(x^2+1)
(%i3) f_prime(1.5);
(%o3) 0.30769230769231
```

b. Ns utilisons `solve` pour trouver les valeurs de x pour lesquelles $f'(x) = 0.5$:

```
(%i4) solve(f_prime(x)=0.5,x);
rat: replaced -0.5 by -1/2 = -0.5
(%o4) [x=-1,x=1]
```

Cela vaut la peine de vérifier graphiquement cette réponse. Nous allons représenter graphiquement $f(x)$ ainsi que les tangentes en $x = \pm 1$:

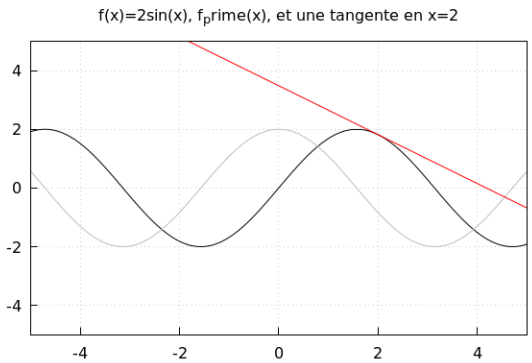
```
(%i5) TANGENT(x,a):=f_prime(a)*(x-a)+f(a)$
(%i6) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      title="La fonction arctan et les 2 tangentes en x=+/- 1",
      color=black,
       explicit((f(x)),x,-3,3),
      color=red,
       explicit((TANGENT(x,-1)),x,-3,3),
       explicit((TANGENT(x,1)),x,-3,3)
      );
```



Nous voyons que ces deux tangentes sont les seules droites tangentes ayant une pente de $\frac{1}{2}$.

Exemple 3.2.2. Définir $f(x) = 2\sin x$. Utiliser wxMaxima pour calculer $f'(x)$. A l'aide du logiciel, donner l'équation de la tangente à $f(x)$ au point d'abscisse $x = 2$. Finalement, dessiner le graphe de $f(x)$, $f'(x)$ et la tangente en $x = 2$. Emettre un commentaire sur la relation liant la pente de la tangente entre $f(x)$ et $f'(x)$ at $x = 2$.

```
(%i7) f(x):=2*sin(x)$
(%i8) f_prime(x):=''(diff(f(x),x))$
(%i9) TANGENT(x,a):=f_prime(a)*(x-a)+f(a)$
(%i10) wxdraw2d(
       grid=true,
       xrange=[-5,5],
       yrange=[-5,5],
       dimensions=[600,600],
       title="f(x)=2sin(x), f_prime(x), et une tangente en x=2",
       color=black,
        explicit((f(x)),x,-5,5),
       color=grey,
        explicit((f_prime(x)),x,-5,5),
       color=red,
        explicit((TANGENT(x,2)),x,-5,5)
       );
```



La pente de la tangente à $f(x)$ est légèrement supérieure à -1 (nous imposons la contrainte `dimensions` pour le graphique afin de faciliter l'estimation). En conséquence, le graphique de $f'(x)$ a une coordonnée y légèrement supérieure à -1 en $x = 2$. Une approximation décimale est rapidement trouvée dans wxMaxima :

```
(%i11) float(f_prime(2));
(%o11) -0.83229367309428
```

Il est important de pouvoir "estimer visuellement" le graphe d'une fonction et d'en esquisser sa dérivée de cette manière – la fonction dérivée a des valeurs y égales à la pente de $f(x)$ à chaque valeur de x.

Exemple 3.2.3. Définir $f(x) = \sin^5(x) \cdot e^x - \cos^2 x$ et trouver ses dérivées d'ordre 1, 2 et 3. Utiliser `trigsimp` pour simplifier les résultats en utilisant les égalités de Pythagore.

L'objectif de cet exemple est simplement d'illustrer comment wxMaxima gère les dérivées d'ordre supérieur, ce qui serait très fastidieux si cela était fait à la main !

```
(%i12) f(x):=(sin(x))^5*%e^x-(cos(x))^2;
```

(%o12) $\mathrm{f}(x) := \sin(x)^5 e^x - \cos(x)^2$

```
(%i13) diff(f(x),x);
```

(%o13) $e^x \sin(x)^5 + 5\, e^x \cos(x)\, \sin(x)^4 + 2\cos(x)\, \sin(x)$

```
(%i14) diff(f(x),x,2);
```

(%o14) $-4\, e^x \sin(x)^5 + 10\, e^x \cos(x)\, \sin(x)^4 + 20\, e^x \cos(x)^2 \sin(x)^3 - 2\sin(x)^2 + 2\cos(x)^2$

```
(%i15) trigsimp(%);
```

(%o15) $-24\, e^x \sin(x)^5 + 10\, e^x \cos(x)\, \sin(x)^4 + 20\, e^x \sin(x)^3 - 4\sin(x)^2 + 2$

```
(%i16) diff(f(x),x,3);
```

(%o16) $-14\, e^x \sin(x)^5 - 50\, e^x \cos(x)\, \sin(x)^4 + 60\, e^x \cos(x)^2 \sin(x)^3 + 60\, e^x \cos(x)^3 \sin(x)^2 - 8\cos(x)\, \sin(x)$

```
(%i17) trigsimp(%);
```

(%o17) $-74\, e^x \sin(x)^5 - 110\, e^x \cos(x)\, \sin(x)^4 + 60\, e^x \sin(x)^3 + 60\, e^x \cos(x)\, \sin(x)^2 - 8\cos(x)\, \sin(x)$

3.3 Produits, Quotients et Combinaisons Linéaires

La dérivée d'un produit, d'un quotient ou d'une combinaison linéaire de fonctions peut être décomposée en utilisant les dérivées des fonctions composantes :

La règle du produit : $$\frac{\mathrm{d}}{\mathrm{d}x}[f(x)\cdot g(x)] = f'(x)\cdot g(x) + f(x)\cdot g'(x)$$

La règle du quotient : $$\frac{\mathrm{d}}{\mathrm{d}x}\frac{f(x)}{g(x)} = \frac{f'(x)\cdot g(x) - f(x)\cdot g'(x)}{[g(x)]^2}$$

Combinaisons linéaires : $$\frac{\mathrm{d}}{\mathrm{d}x}[a\cdot f(x) + b\cdot g(x)] = a\cdot f'(x) + b\cdot g'(x)$$

Exemple 3.3.1. Vérifier la règle du quotient pour les fonctions $f(x) = \sin x$ et $g(x) = \cos x$.

```
(%i1) f(x):=sin(x)$
      g(x):=cos(x)$
      f_prime(x):=(diff(f(x),x))$
      g_prime(x):=(diff(g(x),x))$
(%i2) (f_prime(x)*g(x)-f(x)*g_prime(x))/(g(x))^2;
(%o2) (sin(x)^2+cos(x)^2)/cos(x)^2
(%i3) trigsimp(%);
(%o3) 1/cos(x)^2
```

Nous reconnaissons que $\frac{\sin x}{\cos x} = \tan x$ et que $\frac{\mathrm{d}}{\mathrm{d}x}\tan x = \sec^2 x$ comme cela est attendu. Remarquer que `trigsimp` était nécessaire afin de forcer wxMaxima à appliquer les identités de Pythagore au résultat. Par une vérification directe, nous obtenons :

```
(%i4) diff((sin(x)/cos(x)),x);
(%o4) sin(x)^2/cos(x)^2+1
(%i5) trigsimp(%);
(%o5) 1/cos(x)^2
```

Exemple 3.3.2. Les fonctions hyperboliques de base sont définies par : $\cosh x = \frac{1}{2}(e^x + e^{-x})$ et $\sinh x = \frac{1}{2}(e^x - e^{-x})$. Utiliser la règle de dérivation des combinaisons linéaires pour trouver les dérivées de ces deux fonctions, puis appliquer la règle du quotient pour déterminer la dérivée de $\tanh x = \frac{\sinh x}{\cosh x}$.

Nous utilisons la règle de combinaison linéaire et la règle du quotient "à la main", mais nous utilisons wxMaxima pour tout le reste :

```
(%i6) POS:%e^x$
      NEG:%e^(-x)$
(%i7) POS_prime:diff(POS(x),x)$
```

```
      NEG_prime:diff(NEG(x),x)$
(%i8) COSH:0.5*POS+0.5*NEG$
      SINH:0.5*POS-0.5*NEG$
(%i9) COSH_prime:0.5*POS_prime+0.5*NEG_prime;
      SINH_prime:0.5*POS_prime-0.5*NEG_prime;
(%o9) 0.5*%e^x-0.5*%e^(-x)
(%o10) 0.5*%e^x+0.5*%e^(-x)
```

Nous voyons que $(\cosh x)' = \sinh x$ et $(\sinh x)' = \cosh x$. Les fonctions hyperboliques sont connues de wxMaxima, aussi nous pouvons vérifier notre travail en différenciant directement :

```
(%i10) diff(cosh(x),x);
(%o10) sinh(x)
(%i11) diff(sinh(x),x);
(%o11) cosh(x)
```

En appliquant la règle du quotient " à la main" pour trouver $(\tanh x)'$, nous obtenons :

```
(%i12) TANH_prime:(SINH_prime*COSH -SINH*COSH_prime)/(COSH^2);
(%o12) ((0.5*%e^x+0.5*%e^(-x))^2-(0.5*%e^x-0.5*%e^(-x))^2)
       /(0.5*%e^x+0.5*%e^(-x))^2
(%i13) expand(%);
(%o13) 1.0/(0.25*%e^(2*x)+0.25*%e^(-2*x)+0.5)
```

Nous vérifions notre travail en différentiant directement la tangente hyperbolique puis en appliquant `exponentialize` pour transformer le résultat sous la forme d'une expression avec les fonctions exponentielles :

```
(%i14) diff(tanh(x),x);
(%o14) sech(x)^2
(%i15) exponentialize(%);
(%o15) 4/(%e^x+%e^(-x))^2
(%i16) expand(%);
(%o16) 4/(%e^(2*x)+%e^(-2*x)+2)
```

Nous pouvons diviser le numérateur et le dénominateur du résultat par 4 afin d'obtenir l'expression connue.

3.4 Dérivation des fonctions composées

3.4.1 La règle de la Chaine

La **règle de la chaine** est une règle pour différentier la composition de fonctions. Par exemple, si $h(x) = g(f(x))$, alors $h'(x) = g'(f(x))f'(x)$; c'est à dire, nous calculons $g'(x)$ mais en appliquant sa valeur en $f(x)$, puis nous multiplions par un facteur de $f'(x)$. En sciences physiques, il est d'usage d'utiliser la notation de Leibniz comme moyen mnémotechnique : $\frac{\mathrm{d}h}{\mathrm{d}x} = \frac{\mathrm{d}h}{\mathrm{d}f} \cdot \frac{\mathrm{d}f}{\mathrm{d}x}$, ce qui signifie que nous différentions la fonction $h(x)$ tout en tenant compte de $f(x)$ (en traitant $f(x)$ comme s'il s'agit d'une variable simple), puis nous multiplions par un facteur égal à $f'(x)$.

Exemple 3.4.1. Deux manières d'illustrer la règle de la Chaine.

a. Pour illustrer la règle de la Chaine dans la notation de Newton, trouver la dérivée de $h(x) = e^{x^2}$ en définissant $f(x)$ et $g(x)$ de manière à ce que $h(x) = g(f(x))$ et calculer $g'(f(x)) \cdot f'(x)$.

```
(%i1) f(x):=x^2$
       g(x):=%e^x$
       h(x):=g(f(x))$
(%i2) ev(h(x));
(%o2) %e^x^2
(%i3) g_prime(x):=''(diff(g(x),x));
(%o3) g_prime(x):=%e^x
(%i4) g_prime(f(x))*diff(f(x),x);
(%o4) 2*x*%e^x^2
```

b. Pour illustrer la règle de la Chaine en utilisant "la règle mnémotechnique de Leibniz", définir en premier la fonction $h(x) = e^{x^2}$ comme e^f, où f est censée valoir x^2. Multiplier $\frac{\mathrm{d}h}{\mathrm{d}f}$ et $\frac{\mathrm{d}f}{\mathrm{d}x}$ pour obtenir la dérivée de h.

```
(%i5) h:%e^f$
      dh_df:diff(h,f);
(%o5) %e^f
(%i6) f:x^2$
      df_dx:diff(f,x);
(%o6) 2*x
(%i7) dh_df*df_dx;
(%o7) 2*%e^f*x
(%i8) ev(%);
(%o8) 2*x*%e^x^2
```

On obtient le même résultat. Notez que `ev%` a dû être utilisé pour que wxMaxima simplifie le résultat final.

La règle de la Chaine peut être évidemment généralisée à la composition de plus de deux fonctions. Si $i(x) = h(g(f(x)))$ alors nous appliquons deux fois la règle de la Chaine : $i'(x) = h'(g(f(x))) \cdot (g(f(x)))' = h'(g(f(x)))g'(f(x))f'(x)$. L'équivalent mnémotechnique

dans la notation de Leibniz est $\frac{\mathrm{d}i}{\mathrm{d}x} = \frac{\mathrm{d}i}{\mathrm{d}g} \cdot \frac{\mathrm{d}g}{\mathrm{d}f} \cdot \frac{\mathrm{d}f}{\mathrm{d}x}$. Cette règle mnémotechnique est bien plus élégante que la notation avec des apostrophes (qui devient illisible) : ceci nous indique de différentier i par rapport à g (en considérant g comme une variable simple), puis en différentiant g par rapport à f (en considérant f comme une variable simple), puis en différentiant f with par rapport à x.

Exemple 3.4.2. Considérons $i(x) = \sqrt{\sin^2 x + 2}$. Trouver $i'(x)$ en calculant les trois dérivées $\frac{\mathrm{d}i}{\mathrm{d}g}$, $\frac{\mathrm{d}g}{\mathrm{d}f}$, and $\frac{\mathrm{d}f}{\mathrm{d}x}$. Vérifier le résultat en calculant directement $i'(x)$.

```
(%i9) i:sqrt(g)$
      di_dg:(diff(i,g));
(%o9) 1/(2*sqrt(g))
(%i10) g:f^2+2$
      dg_df:(diff(g,f));
(%o10) 2*f
(%i11) f:sin(x)$
      df_dx:(diff(f,x));
(%o11) cos(x)
(%i12) di_dg*dg_df*df_dx;
(%o12) (f*cos(x))/sqrt(g)
(%i13) ev(%);
(%o13) (cos(x)*sin(x))/sqrt(f^2+2)
(%i14) ev(%);
(%o14) (cos(x)*sin(x))/sqrt(sin(x)^2+2)
```

Remarquer que la commande `ev%` a du être utilisée deux fois afin de permettre à wxMaxima de réaliser les substitutions nécessaires. Vérifions notre réponse :

```
(%i15) diff(sqrt((sin(x))^2+2),x);
(%o15) (cos(x)*sin(x))/sqrt(sin(x)^2+2)
```

3.4.2 Différentiation logarithmique

Rappelons la formule de différentiation $(\ln x)' = \frac{1}{x}$. Supposons maintenant que y dépende de x, et que nous souhaitons différencier une fonction $g(x) = \ln y$. La règle de la chaîne nous dit que $g'(x) = \frac{1}{y} \cdot y'(x)$. Cela peut fournir une astuce utile pour trouver $y'(x)$ lorsque des "parties gênantes" apparaissent dans l'expression de $y(x)$. La technique est connue sous le nom de **différentiation logarithmique**.

Exemple 3.4.3. Calculer la dérivée première de x^x en utilisant la différentiation logarithmique étape par étape. Vérifier la réponse en calculant directement la dérivée.

```
(%i16) EQN1:y=x^x;
(%o16) y=x^x
(%i17) EQN2:log(EQN1);
(%o17) log(y)=x*log(x)
(%i18) depends(y,x)$
(%i19) EQN3:diff(EQN2,x);
(%o19) (dy/dx)/y=log(x)+1
```

```
(%i20) EQN4:y*EQN3;
(%o20) (dy/dx)=(log(x)+1)*y
(%i21) subst(x^x,y,EQN4);
(%o21) (d/dx)x^x=x^x*(log(x)+1)
```

Nous voyons que $\frac{\mathrm{d}}{\mathrm{d}x}x^x = x^x \cdot (\ln x + 1)$. Vérifions la réponse :

```
(%i22) y:x^x$
       diff(y,x);
(%o22) x^x*(log(x)+1)
```

3.4.3 Différentiation implicite

Lorsqu'une formule ne peut être définie que de manière implicite (c'est-à-dire que nous ne pouvons pas résoudre y en fonction de x), nous pouvons tirer parti de la règle de la chaîne pour calculer $y'(x)$ en fonction de x et y. Pour effectuer une *différentiation implicite*, nous différencions simplement les deux membres de la formule par rapport à x, puis nous résolvons en $y'(x)$. Nous devons simplement garder à l'esprit que $\frac{\mathrm{d}}{\mathrm{d}x}f(y) = \frac{\mathrm{d}f}{\mathrm{d}y} \cdot \frac{\mathrm{d}y}{\mathrm{d}x}$.

Exemple 3.4.4. Utilisez la différentiation implicite pour trouver la pente de la tangente au cercle unitaire en tout point (en fonction de x et y). Utilisez votre réponse pour tracer la tangente au point $(-\frac{1}{2}, -\frac{\sqrt{3}}{2})$.

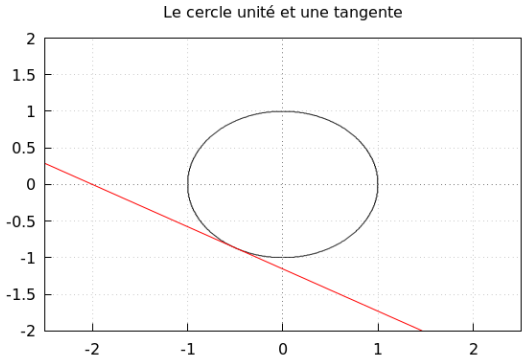
Le cercle unitaire a pour formule $x^2 + y^2 = 1$, qui ne peut pas être résolue explicitement en y sans la diviser en deux fonctions. Nous trouvons $y'(x)$ simplement en différenciant la formule des deux côtés et en isolant algébriquement $\frac{\mathrm{d}y}{\mathrm{d}x}$:

```
(%i23) kill(all)$
(%i1) UNITCIRC:x^2+y^2=1$
(%i2) depends(y,x)$
(%i3) EQN1:diff(UNITCIRC,x);
(%o3) 2*y*(dy/dx)+2*x=0
(%i4) EQN2:EQN1-2*x;
(%o4) 2*y*(dy/dx)=-2*x
(%i5) EQN3:EQN2/(2*y);
(%o5) (dy/dx)=-x/y
```

Nous voyons que $\frac{\mathrm{d}y}{\mathrm{d}x} = -\frac{x}{y}$. Maintenant, nous substituons avec les coordonnées $(-\frac{1}{2}, -\frac{\sqrt{3}}{2})$, calculons la tangente et la représentons sur le même graphique que le cercle unité.

```
(%i6) kill(all)$
(%i1) SLOPE:-(-1/2)/(-sqrt(3)/2)$
(%i2) TANGENT(x):=SLOPE*(x+1/2)-sqrt(3)/2$
(%i3)  wxdraw2d(
       grid=true,
       dimensions=[600,600],
       xaxis=true,
```

```
yaxis=true,
xrange=[-2,2],
yrange=[-2,2],
title="Le cercle unité et une tangente",
color=black,
 implicit((x^2+y^2=1),x,-1,1,y,-1,1),
color=red,
 explicit((TANGENT(x)),x,-2,2)
);
```



3.5 Exercices du Chapitre 3

1. Définissez la fonction $f(x) = (x-2)(x)(x+1)$. Dans le style de l'exemple 3.1.2, trouvez la pente de la tangente en $x = 1$ en utilisant `makelist` pour générer une séquence de dix intervalles $[1, 2], [1, 1.5], [1, 1.25], \ldots$. Faites un tracé de $f(x)$ avec la suite des droites sécantes approchant la tangente.
2. Utilisez la "définition par la limite" de la dérivée pour calculer les formules donnant les dérivées des fonctions suivantes, puis vérifiez les résultats avec la commande `diff`.

   a. $f(x) = \tan x$ b. $f(x) = \sec x$ c. $f(x) = \ln(\cos x)$
3. Définir $f(x) = \sec x$. Utiliser `diff` pour trouver la pente de la tangente en $x = 1.2$. Finalement, représenter graphiquement $f(x)$ ainsi que sa tangente.
4. Utilisez les définitions de $\cosh x$ et $\sinh x$ et la « règle du produit » pour trouver $\frac{\mathrm{d}}{\mathrm{d}x}(\cosh x \cdot \sinh x)$ « manuellement » dans le style de l'exemple 3.3.2. Vérifiez votre réponse en utilisant `diff` et les fonctions intégrées `cosh` et `sinh` de wxMaxima. Une simplification sera nécessaire pour vérifier votre réponse !
5. Lors du calcul de la dérivée d'une fonction trigonométrique compliquée, il peut être utile de simplifier le résultat en utilisant `trigsimp` (simplifie en utilisant les identités Pythagoriciennes), `trigreduce` (simplifie en utilisant les identités de somme d'angles), `trigexpand` (développe en utilisant les identités de somme d'angles) et/ou `trigrat` (simplifie les expressions rationnelles de fonctions trigonométriques). Définissez $f(x) = \sin x \cdot \cos^2(2x)$ et trouvez $f'(x)$ en utilisant `diff`. Utilisez `trigexpand` puis `trigsimp` pour exprimer votre réponse entièrement en termes de puissances de $\cos x$.
6. En suivant l'idée de l'exemple 3.4.1 et de l'exemple 3.4.2, calculez $\frac{\mathrm{d}h}{\mathrm{d}x}$ pour

   a. $h(x) = \cos(3x^3 - 4x + 5)$ b. $h(x) = e^{\sqrt{\sin x}}$

   N'oubliez pas d'utiliser `kill(all)` si les affectations précédentes interfèrent avec vos calculs.
7. Calculer la dérivée cinquième de $f(x) = \sec x$ et simplifier l'expression obtenue autant que possible.
8. Le théorème des accroissements finis stipule que (à condition qu'une fonction soit continue sur $[a, b]$ et différentiable sur (a, b)), nous pouvons toujours trouver $x = c$ sur (a, b) pour lequel la pente de la tangente est égale à la pente de la sécante reliant $(a, f(a))$ et $(b, f(b))$. C'est-à-dire, nous pouvons trouver c tel que $f'(c) = \frac{f(b)-f(a)}{b-a}$. Pour la fonction $f(x) = \tan^{-1} x$, trouvez c pour l'intervalle $[0.5, 2.5]$. Tracez $f(x)$ en noir avec les lignes sécante et tangente appropriées en rouge. Commentez la relation géométrique entre la ligne sécante et la ligne tangente.
9. Soit $y = \cos^{-1} x$. Pour obtenir la formule de $\frac{\mathrm{d}y}{\mathrm{d}x}$, nous devons inverser la formule en $\cos y = x$ et utiliser la différentiation implicite. En suivant le procédé de l'exemple 3.4.4, calculez $\frac{\mathrm{d}y}{\mathrm{d}x}$, en incluant une étape finale de substitution de la définition originale de y et simplifiez le résultat si nécessaire pour obtenir une formule explicite pour $y'(x)$.
10. Pour effectuer une "approximation linéaire", nous utilisons $f(a)$ et $f'(a)$ pour tracer une droite tangente en $x = a$. Ensuite (tant que Δx est relativement petit),

nous pouvons approximer $f(a + \Delta x)$ en utilisant l'équation de la tangente au lieu de $f(x)$ elle-même. Ceci est un cas particulier d'une série de Taylor tronquée – une astuce qui est souvent utilisée en sciences physiques pour simplifier un modèle mathématique.

Définissez $f(x) = \sin x$ et trouvez l'équation de la tangente en $x = 0$. Appelez votre tangente $T(x)$. Utilisez maintenant votre tangente pour faire une approximation linéaire de $f(0.1)$; c'est-à-dire, calculez $T(0.1)$. Comparez avec la valeur réelle de $f(0.1)$. Quelle est l'erreur commise par l'approximation linéaire ? Faites un tracé de $f(x)$ et $T(x)$ sur $[0, 0.2]$ pour mettre en évidence le comportement de l'approximation linéaire près de $x = 0$.

11. L'approximation binomiale est une autre approximation linéaire utile. Supposons que $f(x) = (1 + x)^{50}$. Utilisez une approximation linéaire près de $x = 0$ pour estimer 1.05^{50}. Vérifiez votre réponse en calculant réellement 1.05^{50} et en calculant l'erreur commise par l'approximation. Note : cette approximation est surtout utile pour simplifier une expression variable $(1 + x)^n$ en supposant que x est "petit".

12. Pour l'ellipse $\frac{(x-2)^2}{4} + \frac{(y+3)^2}{25} = 1$, utilisez la différentiation implicite pour trouver une expression pour $\frac{\mathrm{d}y}{\mathrm{d}x}$ en fonction de x et y. Trouvez les équations des deux tangentes qui ont une pente de $\frac{1}{2}$. Afin de trouver les points pour lesquels on a une pente de $\frac{1}{2}$, vous devrez résoudre un système d'équations : une pour la dérivée et une pour l'ellipse d'origine. La syntaxe appropriée pour résoudre un système est `solve([equation1,equation2],[x,y]);`.

    Trouvez les équations des deux droites tangentes perpendiculaires à celles ayant une pente de $\frac{1}{2}$. Enfin, tracez l'ellipse en noir avec les quatre lignes tangentes en rouge. Utilisez `dimensions` pour forcer l'ellipse à apparaître avec le bon rapport visuel.

13. Trouvez la généralisation de la règle du produit pour cinq fonctions. Autrement dit, trouvez une expression générale pour $\frac{\mathrm{d}}{\mathrm{d}x}[f(x) \cdot g(x) \cdot h(x) \cdot i(x) \cdot j(x)]$.

14. En une dimension, la position d'une particule est donnée par sa coordonnée x. Une particule en mouvement aura différentes valeurs de x à différents moments, et nous pouvons tracer son mouvement en utilisant des paires ordonnées (t, x) pour produire un graphique appelé graphique de *position-temps*. La position est une *fonction* du temps, $x(t)$, car à tout moment donné, une particule ne peut être qu'à un seul endroit ! La vitesse moyenne entre deux instants est donnée par le simple rapport $v_{avg} = \frac{x_2 - x_1}{t_2 - t_1}$. Sauf indication contraire, ce manuel utilisera toujours les mètres (m) pour les unités de distance et les secondes (s) pour les unités de temps, donc v_{avg} doit avoir des unités de m/s.
    Supposons qu'une fusée soit observée avec la fonction de position approximative $x(t) = 2.35 \cdot t^3 - 5.1 \cdot t^2 + 4$.

    (a) Trouvez la vitesse moyenne de $t = 2\,\mathrm{s}$ à $t = 3\,\mathrm{s}$; c'est-à-dire, trouvez v_{avg} sur l'intervalle de temps $[2, 3]$.

    (b) Répétez votre calcul pour l'intervalle de temps $[2, 2.5]$, $[2, 2.25]$, et ainsi de suite pour 10 étapes en utilisant `makelist`. La limite semble-t-elle exister ? Quelle est sa valeur ?

    (c) En prenant la vitesse moyenne sur un intervalle très petit, nous trouvons en fait la vitesse *instantanée* (la vitesse n'a pas le temps de changer de manière

significative sur un intervalle de temps très petit). En termes de dérivées, comment la vitesse instantanée doit-elle être définie ?

Chapitre 4

Applications de la dérivation

4.1 Croissance, Décroissance et Extremum local **91**
4.2 Concavité et Inflexion **96**
4.3 Problèmes d'optimisation **98**
4.4 La méthode de Newton **101**
4.5 Exercices du Chapitre 4 . . . . . **107**

Commandes essentielles de ce Chapitre

`diff`	`float`	`declare`
`solve`	`ev`	`rhs`
`wxdraw2d`	`for-do`	`ratsimp`
`subst`	`rectangle`	`makelist`
`find_root`	`polygon`	

4.1 Croissance, Décroissance et Extremum local

En supposant que $f(x)$ est différentiable, $f'(x)$ peut nous en dire beaucoup sur le graphe de $f(x)$:

$f(x)$ est **croissante** sur tout intervalle sur lequel $f'(x) > 0$.

$f(x)$ est **décroissante** sur tout intervalle sur lequel $f'(x) < 0$

Il est possible qu'une fonction continue à être croissante ou décroissante même si $f'(x) = 0$ en un point unique (cela s'appelle un **point selle**).

Un **nombre critique** de $f(x)$ est une valeur de x pour laquelle $f'(x) = 0$ ou $f'(x)$ est indéfinie.

Un **maximum local** de $f(x)$ survient en un nombre critique quand $f'(x)$ passe de positif à négatif et ceci en allant de gauche à droite.

Un **minimum local** de $f(x)$ survient en un nombre critique quand $f'(x)$ passe de négatif à positif et ceci en allant de gauche à droite.

Les minimas et les maximas peuvent être désignés tous deux comme des **extrema**.

Exemple 4.1.1. Trouvez les nombres critiques, les intervalles de croissance ou de décroissance ainsi que les extrema locaux de la fonction $f(x) = -2x^3 + 6x^2 - 5$. Faites un tracé de $f(x)$ avec les intervalles de croissance/décroissance codés en couleur et les extrema locaux clairement étiquetés.

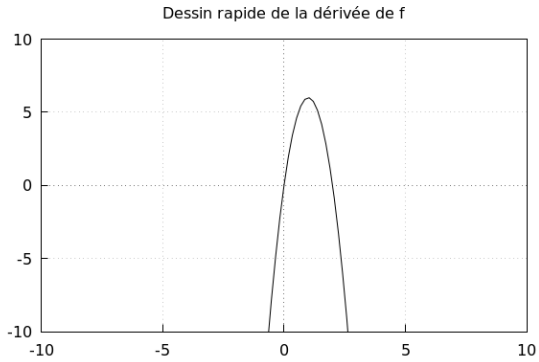
Nous utilisons wxMaxima pour calculer $f'(x)$ et pour trouver toutes les solutions de $f'(x) = 0$ (puisque $f'(x)$ est un polynôme, nous n'avons pas à nous soucier des valeurs de x pour lesquelles $f'(x)$ serait indéfini).

```
(%i1) f(x):=-2*x^3+6*x^2-5$
(%i2) f_prime:diff(f(x),x);
(%o2) 12*x-6*x^2
(%i3) solve(f_prime=0);
(%o3) [x=0,x=2]
```

Les nombres critiques valent 0 et 2. Avec une représentation graphique de $f'(x)$, nous voyons clairement les intervalles où $f'(x) > 0$ et $f'(x) < 0$:

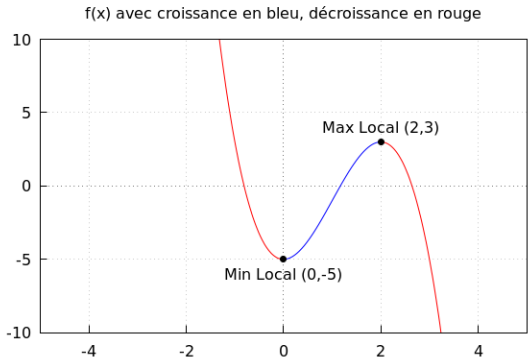
```
(%i4) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[-10,10],
      yrange=[-10,10],
      title="Dessin rapide de la dérivée de f",
      color=black,
```

```
 explicit(f_prime,x,-10,10)
);
```



Nous voyons que $f'(x)$ est négative sur $(-\infty, 0)$, positive sur $(0, 2)$ et négative sur $(2, \infty)$. Nous devrions trouver un minimum local en $(0, f(0))$ parce que $f'(x)$ passe de négatif à positif à cet endroit, et nous devrions trouver un maximum local en $(2, f(2))$ parce que $f'(x)$ passe de positif à négatif à cet endroit. Enfin, nous traçons $f(x)$ avec les intervalles de croissance/décroissance codés en couleur et les extrema locaux marqués par des cercles fermés et étiquetés :

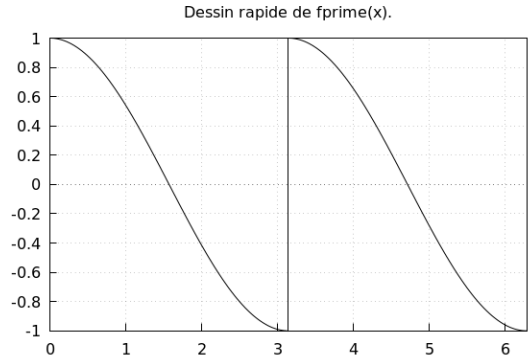
```
(%i5) f(0);
      f(2);
(%o5) -5
(%o6) 3
(%i7) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[-5,5],
      yrange=[-10,10],
      title="f(x) avec croissance en bleu, décroissance en rouge",
      color=blue,
       explicit(f(x),x,0,2),
      color=red,
       explicit(f(x),x,-5,0),
       explicit(f(x),x,2,5),
      color=black,
      point_type=7,
       points([[0,f(0)],[2,f(2)]]),
      label(["Min Local (0,-5)",0,-6],["Max Local (2,3)",2,4])
      );
```



Exemple 4.1.2. Trouver les nombres critiques, les intervalles de croissance/décroissance et les extrema locaux de $f(x) = |\sin x|$ on $[\frac{\pi}{6}, \frac{11\pi}{6}]$.

Nous commençons par représenter $f'(x)$ afin d'avoir une idée de la localisation des nombres critiques :

```
(%i8) f(x):=abs(sin(x));
(%o8) f(x):=abs(sin(x))
(%i9) diff(f(x),x);
(%o9) (cos(x)*sin(x))/abs(sin(x))
(%i10) f_prime:%;
(%o10) (cos(x)*sin(x))/abs(sin(x))
(%i11) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       title="Dessin rapide de fprime(x).",
       color=black,
       explicit(f_prime,x,0,2*%pi)
       );
```



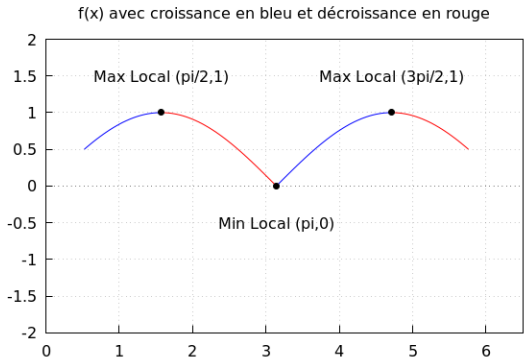
Nous voyons que $f'(x) = 0$ quelque part sur $[1, 2]$ et quelque part sur $[4, 5]$, et il y a un saut soudain sur $[3, 4]$. Nous examinons $f'(x) = \frac{\cos x \sin x}{|\sin x|}$ et décidons que les zéros doivent

être à $\frac{\pi}{2}$ et $\frac{3\pi}{2}$, puisque la fonction cosinus s'annule pour ces valeurs de x. Le dénominateur de $f'(x)$ s'annule lorsque $\sin x = 0$, donc l'emplacement de la discontinuité doit être $x = \pi$. Nous pouvons rapidement vérifier ces valeurs dans wxMaxima :

```
(%i12) subst(%pi/2,x,f_prime);
(%o12) 0
(%i13) subst(3*%pi/2,x,f_prime);
(%o13) 0
(%i14) subst(%pi,x,f_prime);
expt: undefined: 0 to a negative exponent.
 -- an error. To debug this try: debugmode(true);
```

wxMaxima trouve que "quelque chose" provoque une erreur pour $x = \pi$: $f'(x)$ n'est pas défini pour cette valeur.
Nos nombres critiques sont $\frac{\pi}{2}$, π et $\frac{3\pi}{2}$. Nous voyons que $f'(x) > 0$ sur $[\frac{\pi}{6}, \frac{\pi}{2}]$ et $[\pi, \frac{3\pi}{2}]$, tandis que $f'(x) < 0$ sur $[\frac{\pi}{2}, \pi]$ et $[\frac{3\pi}{2}, \frac{11\pi}{6}]$, donc nous pouvons repérer les intervalles de croissance/décroissance en conséquence dans notre tracé de $f(x)$. Nous trouverons des maxima locaux en $(\frac{\pi}{2}, f(\frac{\pi}{2}))$ et $(\frac{3\pi}{2}, f(\frac{3\pi}{2}))$ où $f'(x)$ passe du signe positif au négatif, et nous trouverons un minimum local en $(\pi, f(\pi))$ où $f'(x)$ passe du signe négatif à positif :

```
(%i15) f(%pi/2);
       f(3*%pi/2);
(%o15) 1
(%o16) 1
(%i17) f(%pi);
(%o17) 0
(%i18) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[0,6.5],
       yrange=[-2,2],
       title="f(x) avec croissance en bleu et décroissance en rouge",
       color=blue,
        explicit(f(x),x,%pi/6,%pi/2),
        explicit(f(x),x,%pi,3*%pi/2),
       color=red,
        explicit(f(x),x,%pi/2,%pi),
        explicit(f(x),x,3*%pi/2,11*%pi/6),
       color=black,
        point_type=7,
        points([[%pi/2,f(%pi/2)],[3*%pi/2,f(3*%pi/2)],[%pi,f(%pi)]]),
        label(["Max Local (pi/2,1)",%pi/2,1.5],
        ["Max Local (3pi/2,1)",3*%pi/2,1.5],
        ["Min Local (pi,0)",%pi,-0.5])
        );
```


f(x) avec croissance en bleu et décroissance en rouge
Max Local (pi/2,1)
Max Local (3pi/2,1)
Min Local (pi,0)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
1
2
3
4
5
6

4.2 Concavité et Inflexion

La fonction $f(x)$ est dite **convexe** sur tout intervalle pour lequel $f'(x)$ est *croissante*. En d'autres termes, $f(x)$ est convexe lorsque $f''(x) > 0$.

La fonction $f(x)$ est dite **concave** sur tout intervalle pour lequel $f'(x)$ est *décroissante*. En d'autres termes, $f(x)$ est concave lorsque $f''(x) < 0$.

Un **point d'inflexion** de $f(x)$ est le point frontière entre les intervalles de convexité et de concavité.

On cherche les points d'inflexion aux valeurs de x pour lesquelles $f''(x)$ est nulle ou non définie.

Exemple 4.2.1. Trouvez le point d'inflexion de la fonction $f(x) = e^{-x} \cdot (3x + 2)$ sur l'intervalle $[0, +\infty[$, puis tracez un graphique en couleur de $f(x)$ illustrant les domaines de "convexité" et de "concavité".

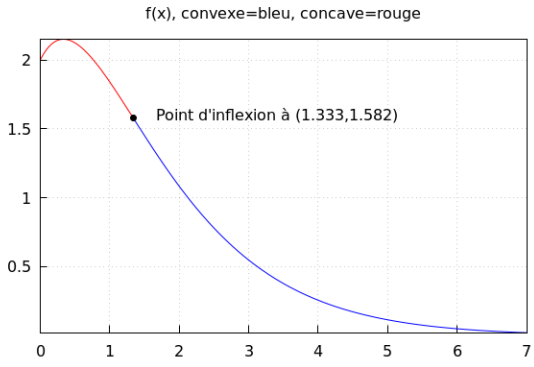
Cette fois, nous adoptons une approche moins graphique pour l'étude préliminaire — en utilisant wxMaxima pour déterminer rapidement l'emplacement où $f''(x) = 0$ et le signe de $f''(x)$ juste à gauche et à droite de ce point. Notez que la dérivée seconde est obtenue en utilisant `diff(f,x,2)`.

```
(%i1) f:%e^(-x)*(3*x+2)$
(%i2) f_prime:diff(f,x);
      f_prime_prime:diff(f,x,2);
(%o2) 3*%e^(-x)-(3*x+2)*%e^(-x)
(%o3) (3*x+2)*%e^(-x)-6*%e^(-x)
(%i4) find_root(f_prime_prime,0,10);
(%o4) 1.333333333333333
(%i5) subst(1.2,x,f_prime_prime);
      subst(1.4,x,f_prime_prime);
(%o5) -0.12047768476488
(%o6) 0.049319392788321
```

Nous voyons que le point d'inflexion marque un changement de la "concavité" vers la "convexité". Maintenant, nous calculons les coordonnées du point d'inflexion et produisons un graphique bien légendé de $f(x)$:

```
(%i7) subst(1.3333333333333,x,f);
(%o7) 1.581582828694387
(%i8) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      title="f(x), convexe=bleu, concave=rouge",
      color=red,
       explicit(f,x,0,1.3333333333333),
      color=blue,
```

```
 explicit(f,x,1.33333333333,7),
color=black,
point_type=7,
 points([[1.3333333333333,1.58158]]),
 label(["Point d'inflexion à (1.333,1.582)",3.4,1.6])
);
```



4.3 Problèmes d'optimisation

L'une des applications scientifiques les plus courantes du calcul différentiel est *l'optimisation* ; c'est-à-dire, trouver la valeur maximale ou minimale d'une fonction soumise aux contraintes du problème. Les définitions suivantes sont utiles :

Pour une fonction $f(x)$ définie sur un intervalle $[a, b]$,

Le **maximum absolu** est une valeur de $f(x)$ supérieure ou égale à n'importe quelle valeur de $f(x)$ dans l'intervalle $[a, b]$.

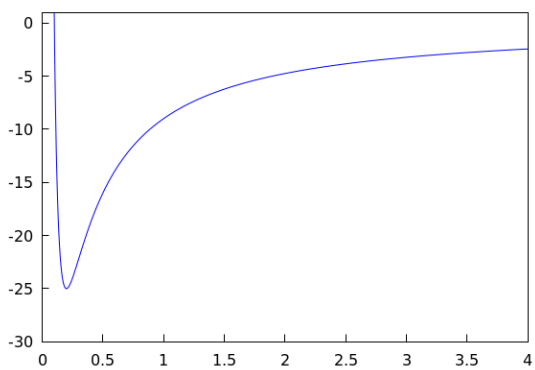
Le **minimum absolu** est une valeur de $f(x)$ inférieure ou égale à n'importe quelle valeur de $f(x)$ sur l'intervalle $[a, b]$.

Nous recherchons des « extrema absolus » aux extrema locaux et éventuellement aux extrémités de l'intervalle. Dans les applications pratiques, un extremum absolu se produit presque toujours à un extremum local.

Exemple 4.3.1. Trouver le minimum absolu de $V(r) = \frac{1}{r^2} - \frac{10}{r}$ on $[0, +\infty[$.

Tout d'abord, nous représentons rapidement $V(r)$ pour avoir une idée de ce qui se passe :

```
(%i1) V(r):=1/r^2-10/r;
(%o1) V(r):=1/r^2-10/r
(%i2) wxdraw2d(
      xrange=[0,4],
      yrange=[-30,1],
      explicit(V(r),r,0,10)
      );
```



Il est clair que le seul minimum local est également le minimum absolu. Nous localisons le minimum en utilisant la dérivée première, puis nous calculons la valeur du minimum absolu de V :

```
(%i3) diff(V(r),r);
(%o3) 10/r^2-2/r^3
(%i4) solve(%=0);
(%o4) [r=1/5]
```

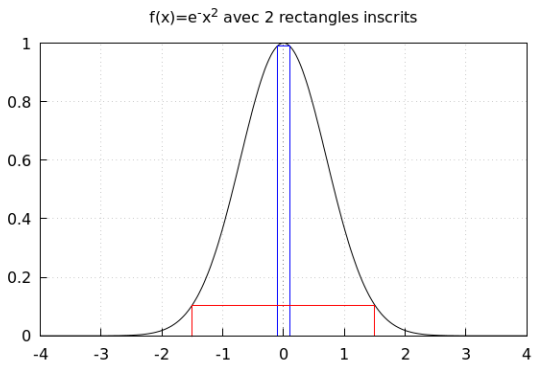
```
(%i5) V(1/5);
(%o5) -25
```

Nous constatons que $V(r)$ atteint son minimum absolu valant -25 en $r = \frac{1}{5}$.

Exemple 4.3.2. Trouvez l'aire maximale d'un rectangle inscrit entre $f(x) = e^{-x^2}$ et l'axe des x, avec deux sommets situés sur $f(x)$ et le côté inférieur situé sur l'axe des x.

Un tel rectangle peut être défini par la coordonnée x de son bord droit. La largeur est $2x$ et la hauteur est $f(x)$ pour tout x que nous choisissons. Pour illustrer, nous traçons deux de ces rectangles pour $x = 0.1$ et $x = 1.5$. Notez que la commande `rectangle` dessine un rectangle basé sur les sommets inférieur gauche, puis supérieur droit. `transparent=true` empêche les rectangles de se remplir automatiquement avec une couleur.

```
(%i6) f(x):=%e^(-x^2)$
(%i7) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      title="f(x)=e^-x^2 avec 2 rectangles inscrits",
      color=black,
       explicit(f(x),x,-4,4),
      transparent=true,
      color=blue,
       rectangle([-.1,0],[.1,f(.1)]),
      color=red,
       rectangle([-1.5,0],[1.5,f(1.5)])
      );
```



Il est clair que les grandes valeurs de x créent des rectangles longs et plats avec une très petite surface. De même, les petites valeurs de x créent des rectangles hauts et étroits avec une très petite surface. Quelque part entre les deux, il y a une surface maximale à trouver ! Nous trouvons une formule pour la surface en fonction de x, puis nous dérivons pour localiser le maximum :

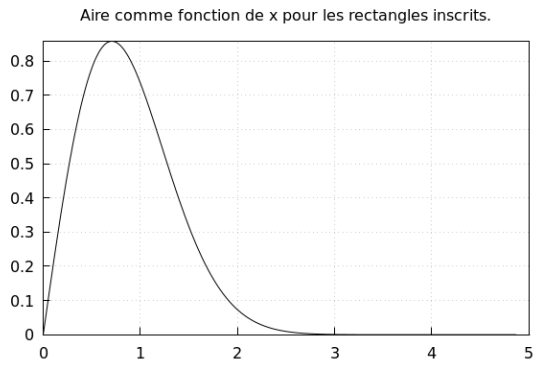
```
(%i8) AREA(x):=2*x*f(x)$
```

```
      diff(AREA(x),x);
(%o9) 2*%e^(-x^2)-4*x^2*%e^(-x^2)
(%i10) solve(%=0);
(%o10) [x=-1/sqrt(2),x=1/sqrt(2)]
(%i11) AREA(1/sqrt(2));
(%o11) sqrt(2)/sqrt(%e)
(%i12) float(%);
(%o12) 0.85776388496071
```

Nous avons implicitement défini des rectangles en utilisant uniquement des valeurs positives de x, nous écartons donc la solution négative. $x = \frac{1}{\sqrt{2}}$ nous donne une aire maximale de $\frac{\sqrt{2}}{\sqrt{e}}$ soit environ 0,86.

Nous représentons graphiquement l'aire comme une fonction de x pour observer les modifications en fonction des choix de x :

```
(%i13) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       title="Aire comme fonction de x pour les rectangles inscrits.",
       color=black,
       explicit(AREA(x),x,0,5)
       );
```



Comme prévu, l'aire présente un maximum pour la valeur ($x = \frac{1}{\sqrt{2}} \approx 0.71$).

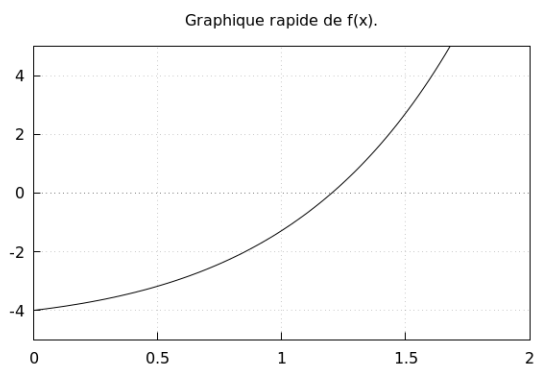
4.4 La méthode de Newton

La méthode de Newton est une technique itérative utilisée pour approximer les racines d'une fonction. Si nous traçons une fonction $f(x)$, nous pouvons choisir une valeur de x relativement proche de la racine qui nous intéresse. Nous traçons ensuite une droite tangente en ce point et trouvons l'abscisse x du point d'intersection entre la tangente et l'axe des abscisses. Si nous avons choisi un point de départ « suffisamment près », l'abscisse de l'intersection entre la tangente et l'axe des abscisses sera plus proche de la racine. Le processus est ensuite répété jusqu'à ce que la précision souhaitée soit obtenue.

Exemple 4.4.1. Mettre en oeuvre la méthode de Newton pas à pas afin de trouver une approximation de la racine de $f(x) = xe^x - 4$ à cinq décimales près.

En premier, nous réalisons un graphique rapide de $f(x)$, de manière à avoir une idée de la localisation de la racine :

```
(%i1) f(x):=x*(%e^x)-4$
(%i2) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[0,2],
      yrange=[-5,5],
      title="Graphique rapide de f(x).",
      color=black,
       explicit(f(x),x,0,5)
      );
```



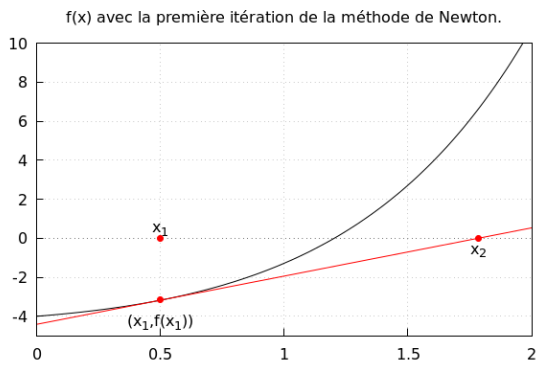
Afin d'illustrer la robustesse de la méthode de Newton, nous faisons une première estimation relativement mauvaise de la racine : $x_1 = 0,5$. Nous calculons $f'(x_1)$ et trouvons l'équation de la tangente en utilisant l'expression "point-pente" (le « point connu » est $(x_1, f(x_1))$). Remarquez que `float` a été utilisé pour empêcher wxMaxima d'exprimer la réponse sous forme de fraction hideuse !

```
(%i3) f_prime:diff(f(x),x);
(%o3) x*%e^x+%e^x
(%i4) x_1:0.5$
(%i5) f_prime_1:subst(x_1,x,f_prime);
```

```
(%o5) 2.473081906050192
(%i6) LINE_1:%*(x-x_1)+f(x_1);
(%o6) 2.473081906050192*(x-0.5)-3.175639364649936
(%i7) float(solve(LINE_1=0,x));
rat: replaced -3.17563936464994 by -51309850/16157329 = -3.17563936464994
rat: replaced 2.473081906050192 by 27877153/11272232 = 2.473081906050195
rat: replaced -0.5 by -1/2 = -0.5
(%o7) [x=1.784081759233688]
(%i8) x_2:ev(x,%);
(%o8) 1.784081759233688
```

Nous avons assigné x_2 à l'abscisse du point d'intersection de `LINE_1` avec l'axe des abscisses (notez que `ev` doit être utilisé pour extraire d'abord la valeur numérique de x_2). x_2 est notre deuxième approximation de la racine de $f(x)$. Nous réalisons un graphique bien légendé pour illustrer :

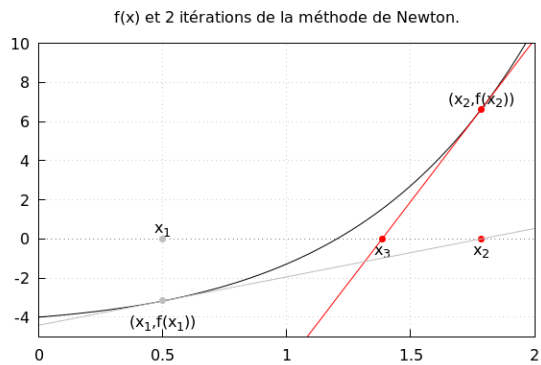
```
(%i9) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[0,2],
       yrange=[-5,10],
       title="f(x) avec la première itération de la méthode de Newton.",
       color=black,
        explicit(f(x),x,0,5),
       label(["x_1",x_1,0.6],["x_2",x_2,-0.5],["(x_1,f(x_1))",x_1,-4.2]),
       color=red,
        explicit(LINE_1,x,0,5),
       point_type=7,
       points([[x_1,0],[x_1,f(x_1)],[x_2,0]])
       );
```



Nous calculons et traçons une deuxième itération pour visualiser comment le processus converge vers la racine. Cette fois, nous supprimons certaines des sorties intermédiaires et combinons `ev` dans la dernière ligne pour obtenir rapidement x_3. Nous utilisons également `ratprint:false$` pour supprimer les notifications ennuyeuses de `rat` :

```
(%i10) f_prime_2:subst(x_2,x,f_prime)$
```

```
(%i11) LINE_2:f_prime_2*(x-x_2)+f(x_2)$
(%i12) ratprint:false$
(%i13) x_3:ev(x,float(solve(LINE2=0,x)));
(%o13) 1.384568736281598

(%i14) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[0,2],
       yrange=[-5,10],
       title="f(x) et 2 itérations de la méthode de Newton.",
       color=black,
        explicit(f(x),x,0,5),
        label(["x_1",x_1,0.6],["x_2",x_2,-0.5],["(x_1,f(x_1))",x_1,-4.2]),
        label(["(x_2,f(x_2))",x_2,7.2],["x_3",x_3,-0.5]),
       color=red,
       point_type=7,
        points([[x_2,0],[x_2,f(x_2)],[x_3,0]]),
        explicit(LINE_2,x,0,5),
       color=grey,
        explicit(LINE_1,x,0,5),
        points([[x_1,0],[x_1,f(x_1)]])
       );
```



Nous voyons que les intersections avec l'axe des abscisses s'approchent de plus en plus de la racine.

Nous voulons créer un processus optimisé qui puisse être facilement répété, et nous l'avons donc réduit à trois lignes par étape. Maintenant, nous devons simplement répéter le processus jusqu'à ce que le résultat se stabilise pour les cinq premières décimales :

```
(%i15) f_prime_3:subst(x_3,x,f_prime)$
       LINE_3:f_prime_3*(x-x_3)+f(x_3)$
       x_4:ev(x,float(solve(LINE_3=0,x)));
(%o17) 1.224019109059294
(%i18) f_prime_4:subst(x_4,x,f_prime)$
       LINE_4:f_prime_4*(x-x_4)+f(x_4)$
```

```
       x_5:ev(x,float(solve(LINE_4=0,x)));
(%o20) 1.202510676401859
(%i21) f_prime_5:subst(x_5,x,f_prime)$
       LINE_5:f_prime_5*(x-x_5)+f(x_5)$
       x_6:ev(x,float(solve(LINE_5=0,x)));
(%o23) 1.202167958618544
(%i24) f_prime_6:subst(x_6,x,f_prime)$
       LINE_6:f_prime_6*(x-x_6)+f(x_6)$
       x_7:ev(x,float(solve(LINE_6=0,x)));
(%o26) 1.202167873197046
```

Nous voyons que la réponse a cessé de se modifier jusqu'à la cinquième décimale, donc nous pouvons écrire la valeur approchée de notre racine comme 1.20217. En vérifiant avec `find_root` de wxMaxima, nous obtenons :

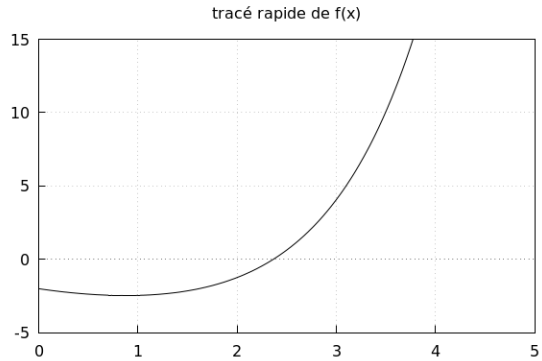
```
(%i27) find_root(f(x),0,2);
(%o27) 1.202167873197043
```

Notre réponse obtenue par la méthode de Newton concorde avec celle de wxMaxima jusqu'à *quatorze* décimales après seulement six itérations.

Exemple 4.4.2. Utilisez la méthode de Newton pour trouver la solution positive de $\cosh x = x + 3$ à cinq décimales près. Construire une boucle `for-do` pour automatiser le processus, incluant une impression des résultats de chaque itération. Enfin, utilisez la commande `find_root` de wxMaxima pour vérifier votre réponse.

Nous commençons par reformuler le problème en termes de racines : une solution de $\cosh x = x + 3$ équivaut à chercher une solution de $\cosh x - x - 3 = 0$. Nous définissons donc la fonction $f(x) = \cosh x - x - 3$ et cherchons une racine positive. Un tracé rapide est utilisé pour trouver une première estimation :

```
(%i28) kill(all)$
       f(x):=cosh(x)-x-3$
(%i2) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[0,5],
      yrange=[-5,15],
      title="tracé rapide de f(x)",
      color=black,
       explicit(f(x),x,0,10)
      );
```



Nous utiliserons une première estimation de $x_1 = 2$. Nous nous référons au dernier exemple pour voir comment les nouveaux x_n ont été générés à partir des anciens, par exemple :

```
f_prime_4:subst(x_4,x,f_prime)$
LINE_4:f_prime_4*(x-x_4)+f(x_4)$
x_5:ev(x,float(solve(LINE_4=0,x)));
```

Afin de généraliser ce processus, nous devons formuler un process dans une boucle `for-do`. Nous utilisons `Y` comme estimation initiale. Nous attribuons à `X` la valeur initiale de `Y`, puis nous calculons la nouvelle estimation `X`, et attribuons à `Y` la nouvelle valeur de `X` (puis la boucle se répète). Nous pouvons ajuster le nombre d'itérations jusqu'à obtenir la précision souhaitée :

```
(%i3) ratprint:false$
(%i4) f(x):=cosh(x)-x-3$
(%i5) f_prime:diff(f(x),x)$
(%i6) Y:2$
(%i7)   for n:1 thru 30
        do(
           X:Y,
           f_prime:subst(X,x,f_prime),
           TANLINE:f_prime*(x-X)+f(X),
           X:ev(x,float(solve(TANLINE=0,x))),
           print(X),
           Y:X
          );

2.471210539097841
2.284899059017247
2.407349289083669
2.335049592696741
2.381363682183222
2.352937265285999
2.37089999759066
2.359742023330758
2.366750365980099
2.362378157528604
```

```
2.365117548400225
2.363405763223947
2.36447721540062
2.363807265474839
2.364226441545812
2.363964277671913
2.364128283962616
2.364025700209412
2.36408987141559
2.364049731672287
2.364074840475577
2.364059134432552
2.364068959017595
2.364062813513952
2.364066657690751
2.364064253064437
2.364065757220357
2.364064816333332
2.364065404882162
2.364065036730068
(%o7) done
```

Nous voyons que l'approximation s'est stabilisée à 2.36406 après 30 itérations. Nous sommes toujours incertains quant à l'arrondi à effectuer, puisque la décimale suivante fluctue encore (nous pourrions effectuer encore plus d'itérations pour trancher). En comparant avec la commande `find_root` de wxMaxima, nous obtenons :

```
(%i8) find_root(f(x),2,3);
(%o8) 2.364065178400271
```

4.5 Exercices du Chapitre 4

1. Définissez $f(x) = \sin x - x^2$. À la manière de l'exemple 4.1.1, tracez $f'(x)$ et trouvez les domaines de croissance/décroissance et les extrema locaux de $f(x)$. Produisez un graphique en couleur de $f(x)$ montrant les domaines de croissance, de décroissance et les extrema locaux.
2. Définissez $f(x) = \ln x + e^x$ sur $]0, +\infty[$. Trouvez les domaines où la fonction est concave ou convexe et réalisez un tracé coloré de $f(x)$ en marquant clairement le point d'inflexion.
3. Définissez $f(x) = x^3$. Trouvez la seule valeur critique pour $f(x)$. Décrivez ce qui se passe au point critique — quel est le nom de cette particularité ?
4. Considérons le "dôme central" de $f(x) = \cos x$ sur l'intervalle $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Nous pouvons définir un triangle isocèle sur cet intervalle avec un sommet à l'origine et deux sommets sur le graphe de $f(x)$. Réaliser un tracé de deux tels triangles avec $f(x)$: l'un très court et large, et l'autre très haut et étroit. Pour tracer un triangle, on peut utiliser `polygon([[sommet1], [sommet2], [sommet3]])`. Enfin, trouver l'aire maximale pour un tel triangle "inscrit".
5. Utilisez la méthode de Newton pour trouver la racine de $f(x) = \sin x$ sur $[\frac{\pi}{2}, \frac{3\pi}{2}]$. Configurez une boucle do comme dans l'exemple 4.4.2. Essayez $x = \frac{\pi}{2}$ comme première estimation. Que se passe-t-il ? Pourquoi ? Maintenant, essayez une meilleure estimation (quelque chose de plus proche de la racine réelle que vous cherchez). Effectuez suffisamment d'itérations pour trouver la racine à 5 décimales près.
6. La règle de l'Hôpital est une règle de limite qui peut être appliquée dans le cas où $\lim \frac{f(x)}{g(x)}$ donne une forme indéterminée $\frac{0}{0}$ ou $\pm\frac{\infty}{\infty}$. À condition que les dérivées existent, $\lim \frac{f(x)}{g(x)} = \lim \frac{f'(x)}{g'(x)}$. Parfois, des applications répétées de la règle de l'Hôpital sont nécessaires, et d'autres fois la règle ne donne rien d'utile.

   Pour les limites suivantes :

   i. $\lim_{x\to\infty} \frac{x^3 \cdot e^x}{x^2 \cdot \ln x}$ ii. $\lim_{x\to 0} \frac{\sin^2 x}{x^2}$

   (a) Montrer que ces limites sont des formes indéterminées.
   (b) Appliquer la règle de l'Hôpital (éventuellement plus d'une fois) en calculant les dérivées nécessaires, jusqu'à obtenir une limite « évidente ».
   (c) Vérifier la réponse directement en utilisant `limit` sur l'expression originale.
7. En dimension 1, la *position* est donnée par la coordonnée x d'une particule. Comme explicité dans les exercices du Module 3, nous pouvons représenter la position sur l'axe vertical et le temps sur l'axe horizontal pour montrer la position en fonction du temps, $x(t)$. La dérivée première de la fonction de position nous donne la vitesse instantanée : $v(t) = x'(t)$ (le taux de changement de la position), et la dérivée seconde nous donne l'accélération instantanée $a(t) = v'(t) = x''(t)$ (le taux de changement de la vitesse). Nos unités par défaut pour la position sont les mètres (m), et nos unités par défaut pour le temps sont les secondes (s). Les unités de la vitesse sont alors $\frac{\text{m}}{\text{s}}$ et les unités de l'accélération sont m/s/s ou $\frac{\text{m}}{\text{s}^2}$.

   Définissez la fonction position $x(t) = -2t^2 + 12t + 12$ sur $[0, 5]$, et répondez aux questions suivantes :

(a) Représenter graphiquement en couleur avec des étiquettes les fonctions $x(t)$, $v(t)$ et $a(t)$ sur $[0, 5]$.

(b) Identifier les intervalles sur lesquels $x(t)$ est croissant/décroissant. Que signifient ces intervalles pour $v(t)$?

(c) Identifier les intervalles sur lesquels $v(t)$ est croissant/décroissant. Que signifient ces intervalles pour $a(t)$?

8. Trouvez et tracez une fonction de position quadratique pour une particule qui se déplace dans la direction négative et accélère. Tracez cette fonction avec $v(t)$ et $a(t)$.

9. Lorsqu'un projectile atterrit à la même hauteur d'où il a été lancé, la portée (distance horizontale du vol) est donnée par la formule $R = \frac{{v_0}^2 \sin 2\theta}{g}$, où v_0 est la vitesse de lancement, θ est l'angle de lancement et g est l'accélération de la gravité. Trouvez l'angle de lancement qui maximise la portée (en considérant v_0 et g comme des constantes). Indice : vous devez déclarer les constantes à wxMaxima avec `declare(v_0,constant)` et `declare(g,constant)` avant de différencier.

10. Le potentiel de Lennard-Jones décrit l'énergie potentielle électrique entre deux atomes sous la forme $V(r) = \frac{A}{r^{12}} - \frac{B}{r^6}$, où A et B dépendent des atomes particuliers, et r est la distance de séparation. Lorsque deux atomes sont combinés dans une molécule diatomique, ils restent très proches de l'énergie potentielle minimale, donc on peut dire que la "longueur" de la molécule se trouve au minimum de $V(r)$. De plus, la *force* électrique entre les atomes est donnée par $F(r) = -\frac{\mathrm{d}V}{\mathrm{d}r}$.

    (a) Utilisez wxMaxima pour trouver la longueur d'une molécule diatomique en fonction de A et B. Indices : avant d'utiliser A et B comme constantes, vous devez les déclarer à wxMaxima en utilisant `declare(A,constant)` et de même pour B. De plus, lorsque vous résolvez votre équation, vous devrez sélectionner uniquement la solution réelle positive de votre liste, puisque r représente une distance de séparation. Vous pouvez extraire cette solution pour une utilisation ultérieure en utilisant `R:rhs(%[N])` où N est le numéro de la solution dans la liste (`rhs` sélectionne uniquement le côté droit d'une équation).

    (b) Trouvez la force entre les atomes à la distance de séparation trouvée dans la partie (a). Continuez à simplifier jusqu'à obtenir la réponse attendue. La fonction `ratsimp` sera utile.

11. Les données sont recueillies par un capteur de position toutes les 0,05 secondes, ce qui donne les paires ordonnées suivantes :

```
t(s)......x(m)
.00        .12
.05        .17
.10        .21
.15        .24
.20        .22
.25        .23
.30        .23
.35        .21
.40        .18
.45        .14
```

```
.50       .09
.55       .02
.60      -.05
.65      -.14
.70      -.25
.75      -.38
.80      -.55
```

Créez une liste (`T`) de toutes les coordonnées t et une liste (`X`) de toutes les coordonnées x. Ensuite, utilisez `makelist` pour créer une liste de paires ordonnées (t, x). Utilisez `makelist` pour calculer la vitesse moyenne en $t = 0.025, 0.075, \ldots$ en utilisant la pente de chaque segment de droite reliant les points consécutifs de position-temps, puis créez une liste de paires ordonnées (t, v). Enfin, utilisez `makelist` pour calculer l'accélération moyenne en $t = 0.05, 0.10, \ldots$, et créez une liste de paires ordonnées (t, a). Faites un graphique codé en couleur de $x(t)$, $v(t)$ et $a(t)$.

Note : les laboratoires de physique utilisent fréquemment des capteurs de position qui sont utilisés de cette manière pour créer des graphiques de vitesse et d'accélération !

Chapitre 5

Le problème de l'Aire

5.1 Sommes de Riemann : Gauche, Droite, Milieu 111
5.2 L'intégrale définie 115
5.3 Les commandes wxMaxima pour les intégrales définies 119
5.4 Approximer une aire à partir d'une liste de données 124
5.5 Exercices du Chapitre 5 127

Commandes essentielles de ce Chapitre

`sum`	`simpsum`	`wxdraw2d`
`float`	`fullratsimp`	`find_root`
`makelist`	`for-do`	`filled_func`
`rectangle`	`print`	`quad_qag`
`ratprint`	`integrate`	`polygon`

5.1 Sommes de Riemann : Gauche, Droite, Milieu

Supposons que nous souhaitons déterminer l'aire entre une fonction $f(x)$ et l'axe des x sur l'intervalle $[a, b]$ (ceci est parfois appelé *le Problème de l'Aire*). Une façon d'approximer une telle surface est de la diviser en rectangles étroits, avec $f(x)$ déterminant la hauteur de chaque rectangle. La somme des aires des rectangles est appelée une **somme de Riemann**.

Pour calculer une somme de Riemann, nous divisons l'intervalle $[a, b]$ en n sous-intervalles, définissant une séquence de valeurs x pour marquer les points de division : $a = x_0, x_1, x_2, \ldots, x_{n-1}, x_n = b$. Il n'est pas nécessaire que les sous-intervalles aient une largeur égale, mais nous choisirons des sous-intervalles de largeur égale lorsque cela sera possible (parfois les données du « monde réel » rendent cela impossible).

Pour l'instant, nous nous limitons au cas où les x_i sont régulièrement espacés, de sorte que nous pouvons dire que

$$x_i - x_{i-1} = \frac{b-a}{n} = \Delta x.$$

Nous définissons les sommes de Riemann à gauche, à droite et au point milieu comme suit :

Une **somme à gauche** est calculée en utilisant le côté gauche de chaque « sous-intervalle » pour trouver la hauteur de chaque rectangle dans la somme. Nous additionnons les aires des rectangles pour obtenir :

$$f(x_0) \cdot \Delta x + f(x_1) \cdot \Delta x + \cdots + f(x_{n-1}) \cdot \Delta x = \sum_{i=1}^{n} f(x_{i-1}) \cdot \Delta x$$

Une **somme à droite** est calculée en utilisant le côté droit de chaque « sous-intervalle » pour trouver la hauteur de chaque rectangle dans la somme. Nous additionnons les aires des rectangles pour obtenir :

$$f(x_1) \cdot \Delta x + f(x_1) \cdot \Delta x + \cdots + f(x_n) \cdot \Delta x = \sum_{i=1}^{n} f(x_i) \cdot \Delta x$$

Une **somme au point milieu** est calculée en utilisant le point milieu de chaque « sous-intervalle » pour trouver la hauteur de chaque rectangle dans la somme. Nous additionnons les aires des rectangles pour obtenir :

$$f\left(\frac{x_0 + x_1}{2}\right) \Delta x + f\left(\frac{x_1 + x_2}{2}\right) \Delta x + \cdots + f\left(\frac{x_{n-1} + x_n}{2}\right) \Delta x = \sum_{i=1}^{n} f\left(\frac{x_{i-1} + x_i}{2}\right) \Delta x$$

Note : dans tous les cas, nous définissons la « hauteur » d'un rectangle par une valeur de $f(x)$. Lorsque $f(x)$ est négative, la contribution de l'aire sera négative. Parfois, nous utilisons l'expression *aire signée* pour clarifier ce que nous calculons avec une somme de Riemann.

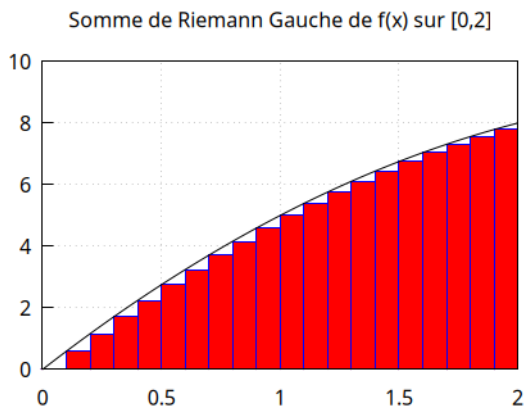
Exemple 5.1.1. Approchez l'aire délimitée par $f(x) = 6x - x^2$ sur $[0, 2]$ en divisant l'intervalle en vingt parties égales et en utilisant la notation de sommation de wxMaxima pour calculer la somme à gauche des aires des rectangles. Produisez un tracé de $f(x)$ avec les rectangles utilisés dans la somme de Riemann. Répétez l'approximation de l'aire avec une somme à droite, y compris le tracé approprié. Laquelle de vos approximations est une sous-estimation ? Laquelle est une surestimation ? Expliquez.

Nous calculons Δx, définissons les x_i pour les points de division le long de $[0, 2]$ (en fonction de i), définissons $f(x)$ et calculons la somme à gauche :

```
(%i1) delx:(2-0)/20$
(%i2) x(i):=''delx*i;
(%o2) x(i):=1/10*i
(%i3) f(x):=6*x-x^2$
(%i4) float(sum(f(x(i-1))*delx,i,1,20));
(%o4) 8.93
```

Nous générons un graphique de la somme de Riemann en utilisant dans un premier temps `makelist` pour générer les rectangles :

```
(%i5) RECTANGLES:makelist(rectangle([x(i-1),0],[x(i),f(x(i-1))]),i,1,20)$
(%i6) load(draw)$
(%i7) wxdraw2d(
      grid=true,
      xrange=[0,2],
      yrange=[0,10],
      title="Somme de Riemann Gauche de f(x) sur [0,2]",
      fill_color=red,
      border=true,
       RECTANGLES,
      color=black,
      explicit(f(x),x,0,2)
      );
```

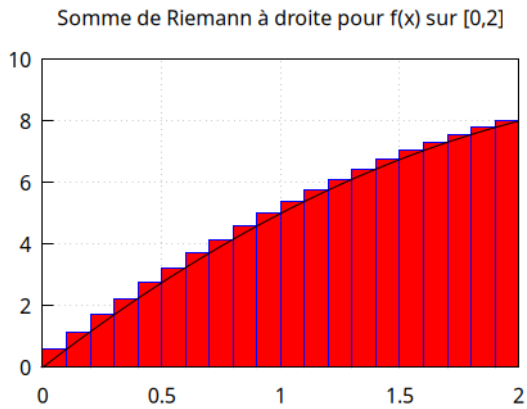


Nous calculons maintenant la somme à droite :

```
(%i8) float(sum(f(x(i))*delx,i,1,20));
(%o8) 9.73
```

Nous terminons en représentant graphiquement la somme à droite :

```
(%i9) RECTANGLES:makelist(rectangle([x(i-1),0],[x(i),f(x(i))]),i,1,20)$
(%i10) wxdraw2d(
       grid=true,
       xrange=[0,2],
       yrange=[0,10],
       title="Somme de Riemann à droite pour f(x) sur [0,2]",
       fill_color=red,
       border=true,
        RECTANGLES,
       color=black,
       explicit(f(x),x,0,2)
       );
```



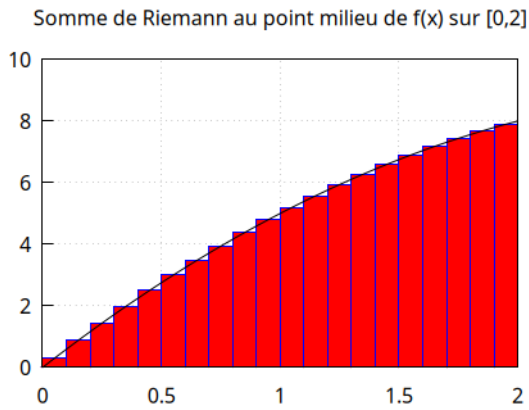
Pour résumer, l'approximation par la somme à gauche est de 8,93, ce qui est une sous-estimation de la vraie aire (nous voyons que chaque rectangle laisse un espace en dessous de $f(x)$). L'approximation par la somme à droite est de 9,73, ce qui est une surestimation de la vraie aire (chaque rectangle a une petite partie située au-dessus du graphique de $f(x)$). La vraie réponse doit se situer quelque part entre ces deux estimations.

Exemple 5.1.2. Pour la même fonction $f(x) = 6x - x^2$, calculez la somme des aires des points milieux pour $n = 20$ sur $[0, 2]$, et tracez le graphique de la somme de Riemann. Comment la réponse numérique se compare-t-elle aux sommes de gauche et de droite des exemples précédents ? Expliquez pourquoi les sommes des points milieux sont généralement plus précises que les sommes de gauche ou de droite.

D'abord, nous définissons un point milieu du $i^{\text{ème}}$ sous-intervalle, puis nous ajustons notre calcul numérique et la définition des rectangles :

```
(%i11) xmid(i):=(x(i-1)+x(i))/2$
```

```
(%i12) float(sum(f(xmid(i))*delx,i,1,20));
(%o31) 9.335000000000001

(%i13) RECTANGLES:makelist(rectangle([x(i-1),0],[x(i),f(xmid(i))]),i,1,20)$
(%i14) wxdraw2d(
       grid=true,
       xrange=[0,2],
       yrange=[0,10],
       title="Somme de Riemann au point milieu de f(x) sur [0,2]",
       fill_color=red,
       border=true,
        RECTANGLES,
       color=black,
       explicit(f(x),x,0,2)
       );
```



Nous voyons que les rectangles des points milieux offrent une approximation de l'aire plus précise car chaque rectangle a à la fois des sous-estimations et des surestimations qui se compensent mutuellement (au moins partiellement). Comme prévu, l'approximation numérique 9.335 se situe entre les estimations données par les sommes de gauche et de droite.

5.2 L'intégrale définie

Les sommes de Riemann deviennent plus précises à mesure que les rectangles deviennent plus petits. Peu importe le choix d'approximation que nous choisissons (gauche, droite ou milieu), la somme de Riemann approche la valeur correcte de l'aire avec la limite lorsque $n \to \infty$. En fait, le choix de la valeur x dans le $i^{\text{ème}}$ sous-intervalle est sans importance – nous pouvons calculer la hauteur de chaque rectangle à n'importe quel x_i^* dans l'intervalle $[x_{i-1}, x_i]$ et l'aire converge toujours correctement vers la limite. Conceptuellement, nous rendons simplement les sous-intervalles si petits que l'erreur commise par l'approximation de l'aire de chaque rectangle approche zéro.

L'aire délimitée par $f(x)$ sur $[a, b]$ s'appelle l'*intégrale définie* :

$$A = \int_a^b f(x)\ \mathrm{d}x = \lim_{n\to\infty} \sum_{i=1}^{n} f(x_i^*) \cdot \Delta x$$

Il n'est généralement pas possible de trouver symboliquement cette limite, car calculer une valeur pour la somme de n termes n'est possible que dans des cas particuliers. Nous illustrons cela avec un exemple symbolique et un exemple numérique.

Exemple 5.2.1. Calculer $\displaystyle\int_{1.5}^{5.2} 0.12x^3 - 0.05x^2 + 3.1x + 0.9\ \mathrm{d}x$ en utilisant une limite d'une somme à droite.

Nous commençons par définir $f(x)$ et calculer Δx (en fonction de n) et x_i (en fonction de n et i) afin de simplifier le processus. Noter que nous devons utiliser `simpsum` pour simplifier la somme avant de trouver la limite lorsque n devient très grand.

```
(%i1) f(x):=0.12*x^3-0.05*x^2+3.1*x+0.9$
(%i2) delx(n):=(5.2-1.5)/n$
(%i3) x(n,i):=1.5+i*delx(n)$
(%i4) ratprint:false$
(%i5) sum(f(x(n,i))*delx(n),i,1,n), simpsum;
(%o5) (3.7*((151959*n^4+303918*n^3+151959*n^2)/(100000*n^3)
      +(134162*n^3+201243*n^2+67081*n)/(60000*n^2)+(1739*n^2+1739*n)
      /(250*n)+(2337*n)/400))/n
(%i6) fullratsimp(%);
(%o6) (183750769*n^2+148176453*n+29277434)/(3000000*n^2)
(%i7) float(limit(%,n,inf));
(%o7) 61.25025633333333
```

Exemple 5.2.2. Utilisez une boucle `for-do` pour approximer $\displaystyle\int_0^{\frac{\pi}{2}} \sin x\ \mathrm{d}x$ en utilisant les sommes à droite avec $n = 10$, $n = 20$, ..., $n = 300$. Selon vous, quelle est la valeur exacte de l'aire ? Répétez votre calcul en utilisant les sommes à gauche et les sommes du milieu au lieu des sommes à droite. Faites un graphique codé en couleur des trois approximations d'aire en fonction de n pour illustrer la manière dont elles convergent.

```
(%i8) kill(all)$
```

```
(%i9) f(x):=sin(x)$
      delx(n):=(%pi/2-0)/n$
      x(n,i):=0+i*delx(n)$
      RSUM(n):=sum(f(x(n,i))*delx(n),i,1,n)$
(%i10) (print("n...RIGHT SUM"),
       for k:1 thru 30 do
        (n:(10*k),
         S:float(RSUM(n)),
         print(n,"","","",S))
       );
```

Pour calculer les sommes de gauche et les sommes du milieu, nous devons simplement apporter de petites modifications au code. Les trois sorties sont affichées côte à côte ci-dessous :

```
(%i11) LSUM(n):=sum(f(x(n,(i-1)))*delx(n),i,1,n)$
(%i12) MSUM(n):=sum(f((x(n,i-1)+x(n,i))/2)*delx(n),i,1,n)$
(%i13) (print("n...LEFT SUM"),
        for k:1 thru 30 do
        (n:(10*k),
         S:float(LSUM(n)),
         print(n,"","","",S))
        );

(%i14) (print("n...MIDPOINT SUM"),
        for k:1 thru 30 do
        (n:(10*k),
         S:float(MSUM(n)),
         print(n,"","","",S))
        );
```

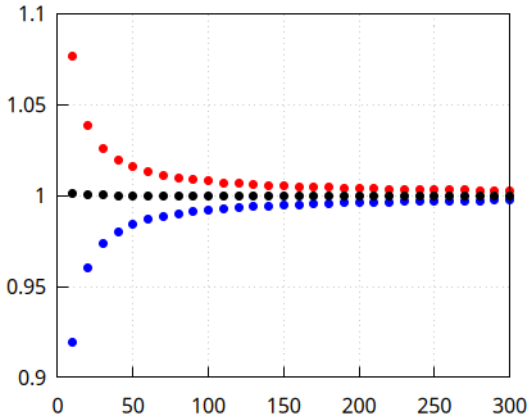
n...	RIGHT SUM	n...	LEFT SUM	n...	MIDPOINT SUM
10	1.076482802694102	10	0.91940317001461	10	1.001028824142708
20	1.038755813418405	20	0.96021599707866	20	1.000257067197303
30	1.025951465275319	30	0.97359158771549	30	1.00011424066727
40	1.019506440307854	40	0.98023653213798	40	1.000064258127218
50	1.015625715211671	50	0.98420978867577	50	1.000041124535493
60	1.013032852971294	60	0.98685291419138	60	1.000028558454002
70	1.011178010806898	70	0.98873806328126	70	1.000020981610027
80	1.009785349217536	80	0.9901503951326	80	1.000016063989881
90	1.008701261346111	90	0.99124796882617	90	1.00001269250526
100	1.007833419873582	100	0.99212545660563	100	1.000010280911905
110	1.007122990125334	110	0.99284302351811	110	1.000008496610797
120	1.006530705712648	120	0.99344073632269	120	1.000007139506457
130	1.006029357632129	130	0.99394630896448	130	1.000006083361951
140	1.005599496208463	140	0.99437952244564	140	1.000005245344728
150	1.005226849216687	150	0.99475487370472	150	1.000004569275912
160	1.004900706603708	160	0.99508322956124	160	1.000004015963601
170	1.004612874419607	170	0.9953728960267	170	1.000003557392221
180	1.004356976925686	180	0.99563033066571	180	1.00000317310517
190	1.004127978782629	190	0.99586062969423	190	1.000002847883216
200	1.003921850392743	200	0.99606786875877	200	1.000002570214103
210	1.003735328738398	210	0.99625534622985	210	1.000002331259568
220	1.003565743368066	220	0.99642576006445	220	1.000002124143224
230	1.003410887725203	230	0.99658133847827	230	1.000001943450265
240	1.003268922609553	240	0.99672393791457	240	1.000001784869924
250	1.003138302783291	250	0.99685511747611	250	1.000001644935961
260	1.00301772049704	260	0.99697619616321	260	1.000001520835631
270	1.002906061553401	270	0.99708829738009	270	1.000001410267224
280	1.002802370776596	280	0.99719238389519	280	1.000001311332571
290	1.002705824619814	290	0.9972892855619	290	1.00000122245502
300	1.002615709246299	300	0.99737972149032	300	1.000001142316237

Les trois sommes approchent de 1 à mesure que n devient de plus en plus grand. Notez que la somme du milieu approche de 1 beaucoup plus rapidement que les sommes de gauche ou de droite – c'est tout simplement un schéma d'approximation plus puissant !

Pour illustrer le comportement des trois limites, nous traçons toutes les aires obtenues par les différentes approximations dans un seul graphique codé par couleur :

```
(%i15) R:makelist([10*k,float(RSUM(10*k))],k,1,30)$
       L:makelist([10*k,float(LSUM(10*k))],k,1,30)$
       M:makelist([10*k,float(MSUM(10*k))],k,1,30)$
(%i16) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[0,300],
       yrange=[.9,1.1],
       point_type=7,
       color=red,
```

```
 points(R),
color=blue,
 points(L),
color=black,
 points(M)
);
```



Nous voyons que les trois limites approchent de 1, mais la somme de milieu se stabilise beaucoup plus rapidement que les sommes de gauche et de droite.

5.3 Les commandes wxMaxima pour les intégrales définies

Comme vous pouvez le supposer, wxMaxima peut calculer directement une intégrale définie grâce à la commande `integrate`.

Exemple 5.3.1. Calculer $\int_{1.5}^{5.2} 0.12x^3 - 0.05x^2 + 3.1x + 0.9 \, \mathrm{d}x$ et $\int_0^{\frac{\pi}{2}} \sin x \, \mathrm{d}x$ en utilisant directement `integrate`.

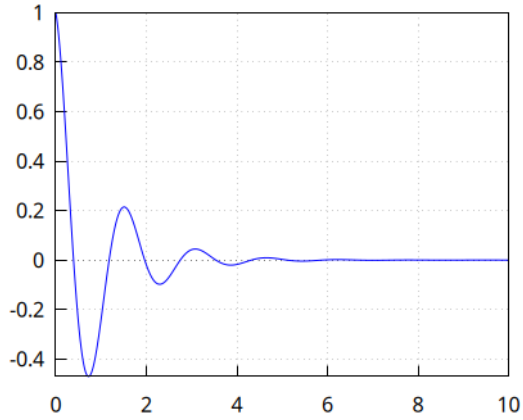
```
(%i1) float(integrate(0.12*x^3-0.05*x^2+3.1*x+0.9,x,1.5,5.2));
(%o1) 61.25025633333333

(%i2) float(integrate(sin(x),x,0,%pi/2));
(%o2) 1.0
```

Nous obtenons les mêmes réponses que celles obtenues par le calcul des limites des sommes de Riemann.

Exemple 5.3.2. Il est parfois utile de visualiser une intégrale comme une surface colorée. Définissez $f(x) = e^{-x} \cos 4x$ sur $[0, +\infty]$ et trouvez les valeurs des deux premières racines, x_1 et x_2. Utilisez `integrate` pour calculer les surfaces bornées sur $[0, x_1]$ et $[x_1, x_2]$, puis colorez la zone positive en rouge et la zone négative en bleu.

Nous commençons par faire un graphique rapide de $f(x)$ pour trouver la valeur approximative des racines :

```
(%i1) f(x):=%e^(-x)*cos(4*x)$
      wxdraw2d(
      grid=true,
      xaxis=true,
      explicit(f(x),x,0,10)
      );
```



Nous voyons des racines dans les intervalles $[0, 1]$ et $[1, 1.5]$.

Maintenant, nous appliquons `find_root` pour déterminer x_1 et x_2 :

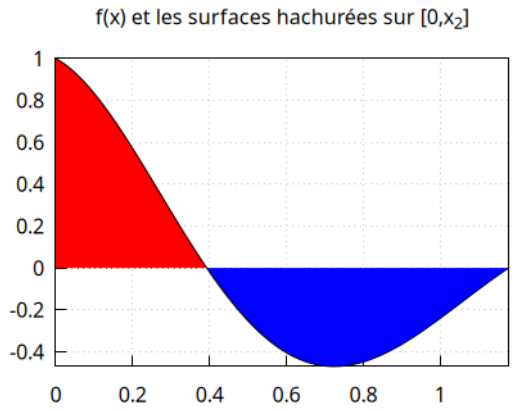
```
(%i3) x1:find_root(f(x),x,0,1);
(%o3) 0.39269908169872
(%i4) x2:find_root(f(x),x,1,1.5);
(%o4) 1.178097245096172
```

Et finalement nous pouvons calculer les aires (algébriques) demandées :

```
(%i5) ratprint:false$
(%i6) float(integrate(f(x),x,0,x1));
      float(integrate(f(x),x,x1,x2));
(%o6) 0.21770162509548
(%o7) -0.23131667717277
```

Nous voyons que la surface au-dessus de l'axe des x est comptée comme positive, et la surface en dessous de l'axe des x est comptée comme négative, comme prévu. Enfin, nous produisons un graphique soigné avec les mises en couleur demandées. Notez que `filled_func` ombre la zone entre deux courbes, donc dans chaque cas, nous devons spécifier la deuxième courbe, $y = 0$ (l'axe des x).

```
(%i8) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      title="f(x) et les surfaces hachurées sur [0,x_2]",
      color=black,
       fill_color=red,
       filled_func=f(x),
       explicit(0,x,0,x1),
       filled_func=false,
      explicit(f(x),x,0,x1),
       fill_color=blue,
       filled_func=f(x),
       explicit(0,x,x1,x2),
       filled_func=false,
      explicit(f(x),x,x1,x2)
      );
```



Exemple 5.3.3. Une *fonction aire*, notée $A(x)$, est un exemple d'intégrale avec une borne d'intégration variable. Par exemple, la fonction

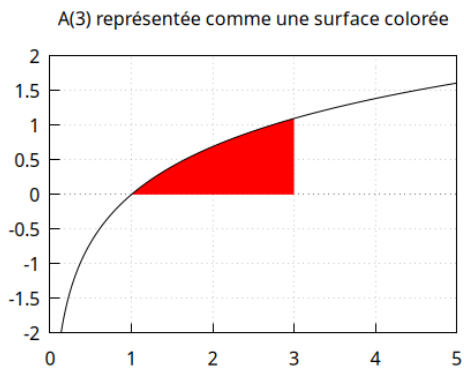
$$A(x) = \int_1^x \ln t \, \mathrm{d}t$$

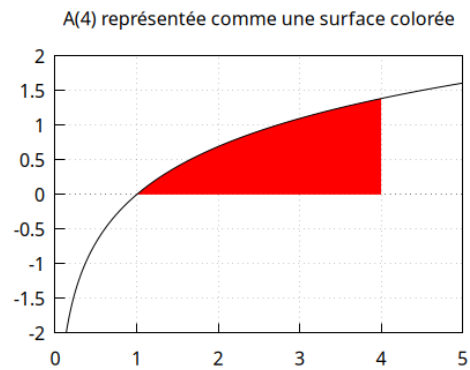
calcule l'aire délimitée par la fonction logarithme naturel sur l'intervalle $[1, x]$.
Remarque : on utilise t comme variable d'intégration pour éviter toute confusion avec x, qui désigne ici la borne supérieure (t est appelée "variable muette" car son nom n'a pas d'importance - une fois l'intégrale calculée, il ne reste qu'une fonction de x).
Définissez $A(x)$ correctement dans wxMaxima, puis utilisez-la pour trouver l'aire délimitée par $y = \ln x$ sur $[3, 4]$. Vérifiez en utilisant directement la commande `integrate` sur l'intervalle $[3, 4]$.

Pour illustrer le comportement de la fonction aire, nous représentons graphiquement $A(3) = \int_1^3 \ln t \, \mathrm{d}t$ et $A(4) = \int_1^4 \ln t \, \mathrm{d}t$:

```
(%i9) f(t):=log(t)$
      A(x):=integrate(f(t),t,1,x)$
(%i11) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[0,5],
       yrange=[-2,2],
       title="A(3) représentée comme une surface colorée",
       color=black,
       fill_color=red,
        filled_func=f(x),
        explicit(0,x,1,3),
        filled_func=false,
       explicit(f(x),x,0,5)
       );
```

```
(%i12) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[0,5],
       yrange=[-2,2],
       title="A(4) représentée comme une surface colorée",
       color=black,
       fill_color=red,
        filled_func=f(x),
        explicit(0,x,1,4),
        filled_func=false,
       explicit(f(x),x,0,5)
       );
```





L'aire délimitée sur $[3,4]$ est simplement la différence entre les deux aires : $A(4) - A(3)$:

```
(%i13) A(4)-A(3);
(%o13) 4*log(4)-3*log(3)-1
```

Nous vérifions maintenant en effectuant la commande `integrate` directement :

```
(%i14) integrate(f(x),x,3,4);
(%o14) 4*log(4)-3*log(3)-1
```

Remarquez que le point de départ (borne inférieure) de $A(x)$ n'a aucune incidence sur le résultat de ce problème. Nous en concluons que les intégrales définies peuvent s'écrire comme des différences de fonctions d'aire — cette idée est essentielle pour introduire le *Théorème fondamental de l'analyse* (que nous étudierons plus en détail dans le Module 6).

Exemple 5.3.4. Essayer de calculer $\displaystyle\int_1^2 \frac{\cos x}{x}\,\mathrm{d}x$ en utilisant `integrate`. Quel est la réponse ? Essayer maintenant de calculer cette intégrale en utilisant `quad_qag` (une des commandes intégrée de Maxima pour les calculs numériques d'intégrales).

```
(%i15) f(x):=cos(x)/x$
       integrate(f(x),x,1,2);
(%o15) -gamma_incomplete(0,2*%i)/2+gamma_incomplete(0,%i)/2
       +gamma_incomplete(0,-%i)/2-gamma_incomplete(0,-2*%i)/2
```

Il semble que wxMaxima ne puisse pas calculer cette intégrale ! Nous allons maintenant utiliser `quad_qag`. La syntaxe de `quad_qag` est identique à celle de `integrate`, à ceci près qu'elle nécessite un entier supplémentaire compris entre 1 et 6, indiquant une méthode d'approximation spécifique (la méthode n'a pas d'importance pour nos besoins). Le résultat de `quad_qag` affiche quatre nombres : le premier est une approximation de l'intégrale, et le second une estimation de l'erreur.

```
(%i16) quad_qag(f(x),x,1,2,1);
(%o16) [0.085576905873897,5.8823808359011771*10^-13,15,0]
```

Nous en concluons que $\displaystyle\int_1^2 \frac{\cos x}{x}\,\mathrm{d}x \approx 0.086$

5.4 Approximer une aire à partir d'une liste de données

Les données réalistes sont généralement présentées sous la forme d'une liste de valeurs discrètes. Si les données nous sont fournies sous forme de paires ordonnées $(x, f(x))$ sur l'intervalle $[a, b]$, nous pouvons approximer $\int_a^b f(x)\,\mathrm{d}x$ en utilisant des rectangles ou des trapèzes. Les méthodes plus sophistiquées sont généralement abordées au second semestre des cours d'analyse.

Exemple 5.4.1. Supposons que la liste suivante de paires ordonnées $(x, f(x))$ ait été obtenue lors d'une expérience simple de physique :

```
(0.34,5.11)
(0.36,4.56)
(0.39,4.40)
(0.44,3.99)
(0.50,3.41)
(0.55,3.37)
(0.65,2.75)
(0.72,2.12)
(0.81,1.78)
(0.90,1.10)
```

Utilisez *une approximation par les rectangles à gauche* pour estimer $\int_{0,34}^{0,90} f(x)\,\mathrm{d}x$. Réalisez un graphique pour visualiser cette approximation.

Nous commençons par créer des listes pour les coordonnées x et y, puis nous utilisons `makelist` pour générer un ensemble de paires ordonnées (dans ce format, toutes les coordonnées et paires ordonnées seront faciles à « appeler » plus tard dans le calcul).

```
(%i1) X:[.34,.36,.39,.44,.50,.55,.65,.72,.81,.9]$
       Y:[5.11,4.56,4.40,3.99,3.41,3.37,2.75,2.12,1.78,1.1]$
(%i2) L:makelist([X[i],Y[i]],i,1,10)$
```

Nous devons établir une expression générale pour l'aire d'un rectangle (dont la hauteur est définie par son côté gauche), en gardant à l'esprit que la notion d'un Δx constant ne s'applique plus ici. Enfin, nous additionnons les aires de tous les rectangles :

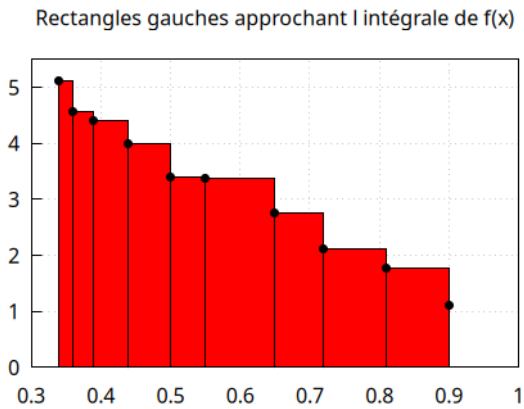
```
(%i3) RECTANGLE(i):=(Y[i])*(X[i+1]-X[i])$
(%i4) sum(RECTANGLE(i),i,1,9);
(%o4) 1.7494
```

Ainsi $\int_{.34}^{.90} f(x)\ \mathrm{d}x \approx 1.7494$.

Un graphique aide à illustrer ce que nous avons réalisé :

```
(%i5) RECTANGLES:makelist(rectangle([X[i],0],[X[i+1],Y[i]]),i,1,9)$
(%i6) wxdraw2d(
```

```
grid=true,
xaxis=true,
xrange=[.3,1],
yrange=[0,5.5],
title="Rectangles gauches approchant l intégrale de f(x)",
fill_color=red,
color=black,
border=true,
 RECTANGLES,
point_type=7,
 points(L)
);
```



Remarquez que l'approximation avec des rectangles à gauche semble largement surestimer l'aire réelle, puisque la fonction $f(x)$ est décroissante.

Exemple 5.4.2. Améliorez l'approximation faite par les aires des rectangles pour $\displaystyle\int_{.34}^{.90} f(x)\ \mathrm{d}x$ en utilisant des trapèzes à la place des rectangles. Réalisez un graphique pour illustrer cette approximation.

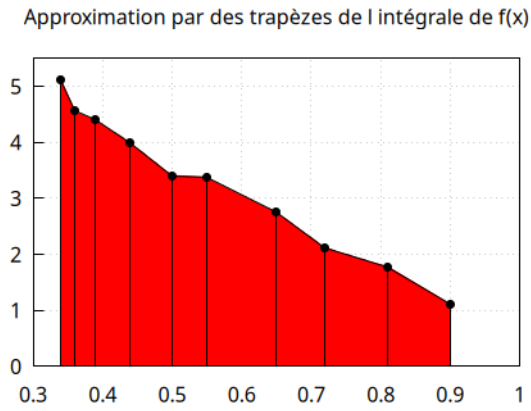
Nous pouvons affirmer que l'aire de chaque trapèze est donnée par $A = \frac{1}{2}(h_1 + h_2) \cdot b$:

```
(%i7) TRAPEZOID(i):=0.5*(Y[i]+Y[i+1])*(X[i+1]-X[i])$
(%i8) sum(TRAPEZOID(i),i,1,9);
(%o8) 1.6139
```

Nous constatons que $\int_{.34}^{.90} f(x)\ \mathrm{d}x \approx 1.614$. Comme attendu, ce résultat est inférieur à celui obtenu en utilisant les rectangles à gauche.
Pour représenter graphiquement cette approximation par des trapèzes, nous aurons à utiliser la commande `polygon`, qui prend en argument les sommets ordonnés d'un polygône qui correspond à la bordure de chaque trapèze.

```
(%i9) TRAPEZOIDS:makelist(polygon([[X[i],0],[X[i+1],0],[X[i+1],Y[i+1]],
      [X[i],Y[i]]]),i,1,9)$
```

```
(%i10) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[.3,1],
       yrange=[0,5.5],
       title="Approximation par des trapèzes de l intégrale de f(x)",
       fill_color=red,
       color=black,
       border=true,
        TRAPEZOIDS,
       point_type=7,
       points(L)
       );
```



5.5 Exercices du Chapitre 5

1. À la manière de l'exemple 5.1.1, trouver et tracer les sommes de Riemann à gauche et à droite pour $n = 30$ de la fonction $f(x) = \frac{1}{x}$ sur l'intervalle $[1, 3]$. Laquelle est une surestimation ? Laquelle est une sous-estimation ?
2. Utiliser une boucle « do » pour produire une suite de sommes des points milieux avec $n = 10$, $n = 20$, ..., $n = 100$ afin d'approcher l'aire délimitée par e^{-x^2} et l'axe des x sur l'intervalle $[-10, 10]$. Comparer le résultat avec celui obtenu en utilisant la fonction `integrate` (il faudra forcer une réponse numérique avec `float`).
3. A la manière de l'exemple 5.2.1, calculer $\displaystyle\int_0^1 x^2 \, \mathrm{d}x$ en appliquant `limit` à une somme de Riemann.
4. Définissez $f(x) = \sin x$ sur l'intervalle $[0, 2\pi]$. Tracez le graphe de $f(x)$ en coloriant les aires positives en rouge et les aires négatives en bleu. D'après le graphique, quelle valeur pensez-vous obtenir pour $\displaystyle\int_0^{2\pi} f(x) \, \mathrm{d}x$? Vérifiez à l'aide de la commande `integrate`.
5. Définissez une fonction d'aire avec un « point de départ » à $x = 0$ pour $\cos x$, puis utilisez cette fonction d'aire pour calculer l'aire délimitée sur l'intervalle $[1, 2]$. Vérifiez en calculant cette aire à l'aide d'une seule intégrale.
6. Définissez les fonctions d'aire $F(x)$ et $G(x)$ pour les deux fonctions $f(x) = \frac{1}{x}$ et $g(x) = \frac{1}{x^2}$, chacune avec un point de départ à $x = 1$. Utilisez une boucle "do" pour produire une liste des aires sur les intervalles $x = [1, 10], [1, 100], \ldots, [1, 10^{10}]$ pour chaque fonction. D'après vos résultats, quelles sont les valeurs de $\displaystyle\int_1^{\infty} \frac{1}{x} \, \mathrm{d}x$ et $\displaystyle\int_1^{\infty} \frac{1}{x^2} \, \mathrm{d}x$?
7. Une symétrie peut souvent être utilisée pour simplifier le calcul d'une intégrale définie sur un intervalle symétrique par rapport à l'origine. Considérez les fonctions $f(x) = \cosh x$ et $g(x) = \sinh x$ sur l'intervalle $[-1, 1]$. Testez la parité de chaque fonction dans wxMaxima. Laquelle est paire ? Laquelle est impaire ? Représentez graphiquement $f(x)$ et $g(x)$ en coloriant les aires selon les signes. Voyez-vous comment simplifier $\displaystyle\int_{-1}^{1} f(x) \, \mathrm{d}x$ et $\displaystyle\int_{-1}^{1} g(x) \, \mathrm{d}x$? Calculez les intégrales en utilisant vos simplifications, puis vérifiez vos réponses en calculant directement les intégrales.
8. Pour calculer l'énergie potentielle totale stockée dans un ressort (pour un allongement de longueur L), nous devons calculer $W = \displaystyle\int_0^L F(x) \cdot \mathrm{d}x$, où $F(x)$ est la force appliquée au ressort pour l'allonger jusqu'à une longueur x (par rapport à sa longueur « naturelle »). Les données suivantes ont été obtenues en étirant un ressort depuis sa longueur naturelle $x = 0$ jusqu'à une longueur finale de $L = 0{,}51$ m. Utilisez une approximation par la méthode des trapèzes pour calculer l'énergie totale stockée dans le ressort (les unités seront dans ce cas exprimées en joules). Réalisez un tracé des points des données expérimentales ainsi que des trapèzes correspondants.

   `x(m).....F(x)(N)`

```
.05          10
.09          20
.14          30
.19          40
.27          50
.38          60
.44          70
.48          80
.51          90
```

Quelle est l'interprétation physique de la fonction d'aire $A(x) = \int_0^x F(t) \cdot \mathrm{d}t$?

9. Un graphique *vitesse-temps* montre la vitesse d'un objet (en m/s) en fonction du temps (en s). Supposons qu'une fonction vitesse soit donnée par $v(t) = 0,2 \cdot t^2$. Quelle est la vitesse de l'objet à $t = 1{,}5\,\mathrm{s}$? En supposant que l'objet ne puisse pas changer trop rapidement de vitesse, quelle distance parcourra-t-il environ dans les 0,05 s suivantes ? À quoi correspond géométriquement ce calcul sur le graphe de $v(t)$? Faites un graphique pour illustrer. Répétez ensuite cette procédure toutes les 0,05 s de $t = 1{,}5\,\mathrm{s}$ à $t = 3\,\mathrm{s}$. Trouvez l'aire totale de tous les rectangles et interprétez votre résultat. Enfin, vérifiez la réponse directement en utilisant `integrate`.

10. Une voiture est flashée par un radar toutes les 0,2 secondes, et les données de vitesse suivantes sont obtenues :

```
t(s)........v(m/s)
0.2          +35.0
0.4          +35.2
0.6          +35.0
0.8          +35.1
1.0          +34.9
1.2          +35.1
1.4          +34.5
1.6          +33.0
1.8          +31.2
2.0          +26.8
2.2          +22.5
2.4          +18.7
2.6          +15.0
2.8          +11.6
3.0          +7.5
3.2          +4.1
3.4          +1.3
3.6          0
3.8          0
4.0          0
```

Utilisez une approximation par des trapèzes pour calculer la distance totale parcourue par la voiture pendant ce processus. Elaborez un graphique avec tous les points des données du radar ainsi qu'avec les trapèzes ombrés. Décrivez ce qui s'est passé (physiquement) aux alentours de 1,2 seconde.

Chapitre 6

Primitives et théorème fondamental

6.1 Primitives **130**
6.2 Le théorème fondamental en Analyse **133**
6.2.1 Les fonctions d'aire 133
6.2.2 Le théorème fondamental 134
6.3 Calcul des intégrales **136**
6.3.1 Primitives de base 136
6.3.2 Changement de variable 137
6.4 Intégrales de surface **139**
6.4.1 Intégration de surface et intégration physique 139
6.4.2 Aire comprise entre deux fonctions 141
6.4.3 Valeur moyenne 146
6.5 Exercices du Chapitre 6 . . . . . **148**

Commandes essentielles de ce Chapitre

`diff`	`ratprint`	`find_root`
`declare`	`filled_func`	
`makelist`	`integrate`	
`solve`	`trigreduce`	
`float`	`subst`	

6.1 Primitives

$F(x)$ est une **primitive** de $f(x)$ si $F'(x) = f(x)$. Il existe une infinité de primitives de $f(x)$, correspondant à différentes constantes additives.

Exemple 6.1.1. Montrer que $F(x) = \frac{1}{3} \cdot x^3$ est une primitive de $f(x) = x^2$, puis établir que $F(x) + C$ est aussi une primitive de $f(x)$, où C est une constante. Représenter graphiquement $f(x)$ ainsi que plusieurs de ses primitives.

Nous vérifions rapidement que $F'(x) = f(x)$:

```
(%i1) F(x):=(1/3)*x^3$
      diff(F(x),x);
(%o1) x^2
```

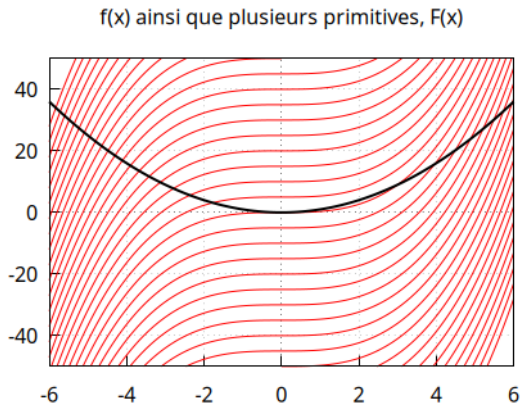
Nous généralisons maintenant à $F(x) + C$, en indiquant à wxMaxima que C est une constante grâce à la commande `declare` :

```
(%i2) declare(C,constant)$
      A(x):=F(x)+C$
(%i4) diff(A(x),x);
(%o4) x^2
```

La dérivée donne bien encore $f(x)$.

Finalement, nous réalisons un graphique de $f(x)$ accompagné de plusieurs primitives correspondant à différentes valeurs de C. Nous utilisons `makelist` afin de générer rapidement la "famille" de courbes correspondant aux primitives :

```
(%i5) f(x):=x^2$
(%i6) FAMILY: makelist(explicit(F(x)+5*i,x,-6,6),i,-20,20)$
(%i7) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[-6,6],
      yrange=[-50,50],
      title="f(x) ainsi que plusieurs primitives, F(x)",
      color=red,
       FAMILY,
      color=black,
      line_width=2,
       explicit(f(x),x,-6,6)
      );
```



Exemple 6.1.2. Pour $f(x) = \cos(5x)$, proposez comme première supposition pour une primitive : $F(x) = \sin(5x)$. Quel est le problème avec cette proposition ? Corrigez-la ensuite de manière appropriée et vérifiez qu'elle fonctionne.

```
(%i8) F(x):=sin(5*x)$
       diff(F(x),x);
(%o8) 5*cos(5*x)
```

On constate que $F'(x)$ « diffère » d'un facteur 5 en raison de la règle de la chaîne, mais il est simple de corriger cela en multipliant par un facteur $\frac{1}{5}$:

```
(%i9) F(x):=(1/5)*sin(5*x)$
      diff(F(x),x);
(%o9) cos(5*x)
```

La forme générale des primitives de $f(x) = \cos(5x)$ est $F(x) = \frac{1}{5}\sin(5x) + C$, où C est une constante arbitraire.

Exemple 6.1.3. Vérifier que $F(x) = \frac{1}{n+1}x^{n+1}$ est une primitive de $f(x) = x^n$ (à condition que $n \neq -1$).

```
(%i10) F(x):=(1/(n+1))*x^(n+1)$
(%i11) diff(F(x),x);
(%o11) x^n
```

Exemple 6.1.4. Supposons que $f'(x) = x^2 - 3x + 5$. Trouver et représenter graphiquement la fonction $f(x)$ dont la courbe passe par le point $(1, 1)$.

Remarquons que nous avons adapté notre notation pour cet exemple, mais nous cherchons bien toujours une primitive : $f(x)$ est une primitive de $f'(x)$. Nous devinons (et vérifions ensuite) la forme d'une primitive en appliquant la formule $\frac{1}{n+1}x^{n+1}$ à chaque puissance de x. Puisque la dérivation respecte les combinaisons linéaires, nous pouvons trouver une primitive terme par terme et considérer les coefficients constants comme des éléments invariants :

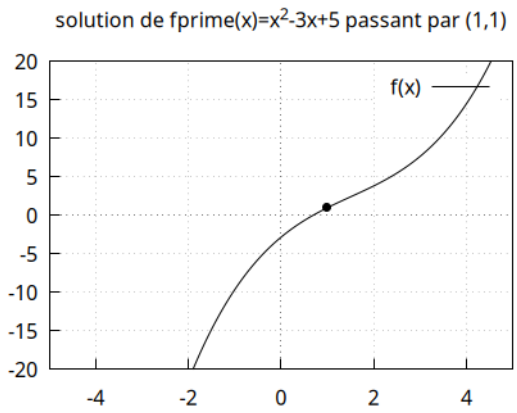
```
(%i12) f(x):=(1/3)*x^3-3*(1/2)*x^2+5*x$
(%i13) diff(f(x),x);
(%o13) x^2-3*x+5
```

La forme générale des primitives est donc $f(x) = \frac{1}{3}x^3 - \frac{3}{2}x^2 + 5x + C$. Il s'agit d'une *famille* de courbes, parmi laquelle nous souhaitons trouver l'unique courbe passant par $(1, 1)$. En d'autres termes, nous avons à appliquer la condition $f(1) = 1$ et à résoudre pour trouver C :

```
(%i14) fgen(x):=f(x)+C$
(%i15) solve(fgen(1)=1,C);
(%o15) [C=-17/6]
```

Enfin, nous représentons graphiquement $f(x)$:

```
(%i16) particular(x):=f(x)-17/6$
(%i17) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[-5,5],
       yrange=[-20,20],
       title="solution de f_prime(x)=x^2-3x+5 passant par (1,1)",
       color=black,
        explicit(particular(x),x,-5,5),
       point_type=7,
        points([[1,1]])
       );
```



6.2 Le théorème fondamental en Analyse

6.2.1 Les fonctions d'aire

Pour une fonction $f(x)$, nous définissons une **fonction d'aire** qui intègre depuis un point de départ c jusqu'à un point final variable x : $A(x) = \int_c^x f(t)\,\mathrm{d}t$. Rappelons que t est définie ici comme une « variable muette » : nous parlons toujours de l'aire sous $f(x)$, mais nous utilisons une lettre différente pour la variable d'intégration, car x est déjà utilisé pour désigner l'extrémité droite variable de l'intervalle d'intégration.

Toute aire sous $f(x)$ peut être calculée comme une différence de fonctions d'aire, comme illustré dans le Chapitre 5. L'aire délimitée sur l'intervalle $[a, b]$ s'obtient par $A(b) - A(a)$.

Si nous effectuons un zoom sur une bande d'aire très étroite, par exemple l'aire délimitée sur $[x, x + h]$, nous constatons qu'il existe deux façons de représenter cette aire : soit comme un simple rectangle AIRE $= f(x) \cdot h$, soit comme une différence de fonctions d'aire AIRE $= A(x + h) - A(x)$. Le rectangle est une *approximation*, mais celle-ci devient plus précise au fur et à mesure que h diminue.

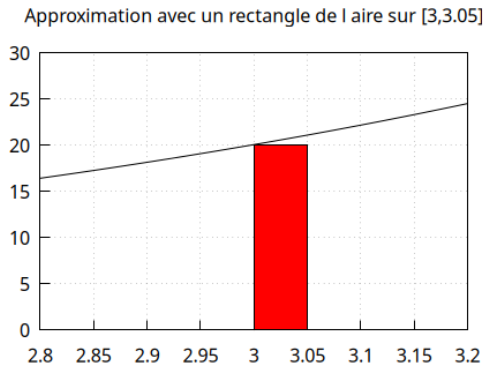
Exemple 6.2.1. Pour la fonction $f(x) = e^x$, calculez l'aire délimitée sur l'intervalle $[3, 3{,}05]$ de deux manières différentes : d'une part, comme un rectangle de hauteur $f(3)$, et d'autre part, comme une différence de fonctions d'aire. Dans chaque cas, illustrez le calcul avec un graphique adapté. Quel est l'erreur commise par l'approximation avec le rectangle ?

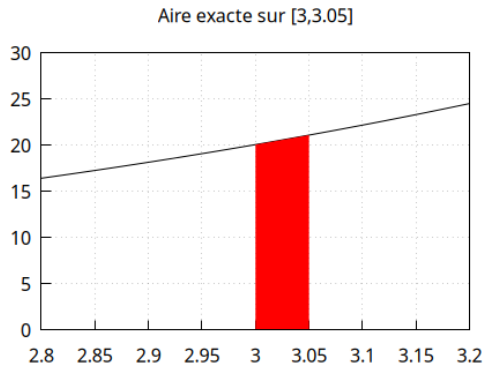
Nous commençons en se servant de l'approximation par un rectangle $AREA = f(3) \cdot (0.05)$, puis nous représentons graphiquement $f(x)$ ainsi que le rectangle très fin correspondant à cette "tranche" :

```
(%i1) f(x):=%e^x$
(%i2) float(f(3)*0.05);
(%o2) 1.004276846159383
(%i3) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[2.8,3.2],
      yrange=[0,30],
      title="Approximation avec un rectangle de l aire sur [3,3.05]",
      color=black,
       explicit(f(x),x,2.8,3.2),
      fill_color=red,
       rectangle([3,0],[3.05,f(3)])
      );
```

Nous définissons maintenant une fonction d'aire $A(x) = \int_0^x e^t\,\mathrm{d}t$ et calculons AIRE $= A(3{,}05) - A(3{,}00)$. Rappelons que le point de départ de la fonction d'aire n'a pas d'importance — nous choisissons $t = 0$ simplement parce que c'est « visuellement agréable ».

```
(%i4) A(x):=integrate(f(t),t,0,x)$
(%i5) ratprint:false$
      float(A(3.05)-A(3));
(%o6) 1.029807499352945
(%i7) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[2.8,3.2],
      yrange=[0,30],
      title="Aire exacte sur [3,3.05]",
      color=black,
       explicit(f(x),x,2.8,3.2),
      filled_func=true,
       filled_func=f(x),
       explicit(0,x,3,3.05),
      filled_func=false
      );
```





Enfin, nous calculons l'erreur relative commise par l'approximation avec un rectangle :

```
(%i8) float(A(3.05)-A(3.00)-f(3)*0.05);
(%o8) 0.025530653193553
(%i9) PERCENT_ERROR:100*%/(A(3.05)-A(3.0));
(%o9) 2.479167534669774
```

6.2.2 Le théorème fondamental

L'exemple précédent nous donne une démonstration géométrique pour le **théorème fondamental de l'analyse** (TFA). Nous pouvons écrire que les deux représentations de l'aire sont égales pour en déduire
$f(x) \cdot h = A(x+h) - A(x)$ à la limite quand $h \to 0$. Si nous divisons par h, nous reconnaissons la *dérivée* de $A(x)$ pour le membre de droite :
$f(x) = \lim\limits_{h \to 0} \dfrac{A(x+h) - A(x)}{h} = A'(x)$. Autrement dit, la fonction d'aire est une primitive de $f(x)$.

Ainsi, pour calculer l'aire délimitée par $f(x)$ sur $[a, b]$, nous pouvons trouver une primitive $A(x)$ et évaluer $A(b) - A(a)$ (une autre notation pour cette différence est $A(x)|_a^b$). Remarquez que la constante arbitraire n'a pas d'importance, car elle s'annule lorsque l'on effectue la différence. Le théorème fondamental de l'analyse fournit un outil symbolique puissant qui nous permet de calculer des intégrales définies sans avoir recours à un processus limite compliqué ou à une approche par approximation.

Exemple 6.2.2. Trouver mentalement une primitive de e^{-2x}, puis calculer $\displaystyle\int_1^3 e^{-2x}\,\mathrm{d}x$ en appliquant le TFA. Verifier la réponse en utilisant `integrate`.

L'exponentielle est égale à sa propre dérivée, donc une première hypothèse raisonnable pour déterminer une primitive est e^{-2x}. Ensuite, nous vérifions la réponse :

```
(%i10) diff(%e^(-2*x),x);
(%o10) -2*%e^(-2*x)
```

Nous remarquons que nous avons un facteur -2 à éliminer, car la règle de la chaîne ajoute un facteur de $\frac{\mathrm{d}}{\mathrm{d}x}(-2x) = -2$. Nous compensons en ajoutant un facteur de $-\frac{1}{2}$ pour obtenir $F(x) = -\frac{1}{2}e^{-2x}$. Nous vérifions rapidement notre réponse, puis appliquons le théorème fondamental de l'analyse en évaluant $F(3) - F(1)$:

```
(%i11) F(x):=-(1/2)*%e^(-2*x)$
      diff(F(x),x);
(%o11) %e^(-2*x)
(%i12) F(3)-F(1);
(%o12) %e^(-2)/2-%e^(-6)/2
(%i13) float(%);
(%o13) 0.066428265529973
```

Vérifions avec la commande `integrate` :

```
(%i14) float(integrate(%e^(-2*x),x,1,3));
(%o14) 0.066428265529973
```

6.3 Calcul des intégrales

6.3.1 Primitives de base

Le TFA nous indique que le calcul des intégrales définies (du moins celles explicitées de manière symbolique) revient à trouver des primitives. Nous utilisons le terme **intégrale indéfinie** comme synonyme de *primitive*, et nous l'écrivons sous la forme d'une intégrale sans bornes : $F(x) = \int f(x)\ \mathrm{d}x$. Si nous souhaitons évaluer une intégrale *définie* sur $[a, b]$, nous calculons simplement la différence $F(b) - F(a)$. wxMaxima peut calculer de nombreuses intégrales indéfinies à l'aide de `integrate`, bien que beaucoup d'intégrales ne puissent pas être exprimées sous forme symbolique.

Exemple 6.3.1. Calculer $\int \frac{1}{1+x^2}\ \mathrm{d}x$, puis vérifier le résultat en utilisant `diff`.

```
(%i1) integrate(1/(1+x^2),x);
(%o1) atan(x)
(%i2) diff(%,x);
(%o2) 1/(x^2+1)
```

Exemple 6.3.2. Trouvez une approximation décimale de $\int_1^2 \sin x \cos x\ \mathrm{d}x$ en calculant d'abord l'intégrale indéfinie, puis en évaluant aux bornes de l'intervalle d'intégration. Vérifiez votre réponse en calculant directement l'intégrale définie.

```
(%i3) f(x):=cos(x)*sin(x)$
       F:integrate(f(x),x);
(%o3) -cos(x)^2/2
(%i4) float(subst(2,x,F)-subst(1,x,F));
(%o4) 0.059374196079117
(%i5) float(integrate(f(x),x,1,2));
(%o5) 0.059374196079117
```

Exemple 6.3.3. Définissez $f(x) = e^{-x^2}$ et calculez $\int f(x)\ \mathrm{d}x$. Commentez la fonction spéciale que wxMaxima utilise pour exprimer le résultat.

```
(%i6) f(x):=%e^(-x^2)$
(%i7) integrate(f(x),x);
(%o7) (sqrt(%pi)*erf(x))/2
```

wxMaxima exprime la solution à l'aide de la fonction spéciale $\mathrm{erf}(x)$ (la "fonction d'erreur"). Une recherche rapide montre que $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-x^2}\ \mathrm{d}x$. En d'autres termes, wxMaxima n'a pas réussi à trouver une expression algébrique pour exprimer cette intégrale : $\mathrm{erf}(x)$ est simplement la fonction d'aire correspondant à e^{-x^2}. $\int e^{-x^2}\ \mathrm{d}x$ est en réalité une intégrale très classique (notamment en calcul des probabilités), mais elle ne peut pas s'exprimer de manière algébrique ! En revanche, il est toujours possible d'obtenir des approximations numériques de l'intégrale *définie* si nécessaire.

Exemple 6.3.4. Calculer $\int \sin^2 x \, \mathrm{d}x$ en devinant une primitive. Astuce : `trigreduce` pourra être d'une bonne aide.

Il n'est pas facile d'imaginer l'expression d'une primitive de $\sin^2 x$. Une première tentative pourrait être $F(x) = \frac{1}{3}\sin^3 x$ ou $G(x) = \frac{1}{3}\cos^3 x$, mais aucune de ces deux fonctions ne répond au problème posé (la règle de la chaîne génère des facteurs supplémentaires) :

```
(%i8) F(x):=(1/3)*(sin(x))^3$
      diff(F(x),x);
(%o9) cos(x)*sin(x)^2
(%i10)G(x):=(1/3)*(cos(x))^3$
      diff(G(x),x);
(%o10) -cos(x)^2*sin(x)
```

Nous appliquons `trigreduce` pour transformer l'intégrande :

```
(%i11) f(x):=(sin(x))^2$
       trigreduce(%);
(%o12) f(x):=(1-cos(2*x))/2
```

En séparant l'intégrande en deux parties, il n'est plus difficile de trouver une primitive pour chaque expression : $f(x) = \frac{1}{2} - \frac{1}{2}\cos(2x) \implies F(x) = \frac{1}{2}x - \frac{1}{4}\sin(2x)$.
On le vérifie rapidement avec la commande `integrate` :

```
(%i13) integrate(f(x),x);
(%o13) (x-sin(2*x)/2)/2
```

6.3.2 Changement de variable

Lors du calcul d'intégrales indéfinies à la main, il est souvent nécessaire de reconnaître les effets de la règle de la chaîne. Ce processus est formalisé en effectuant un changement de variable pour transformer l'intégrale en une forme plus simple.

Pour effectuer un changement de variable u, nous :

1. Choisissons une nouvelle variable et nous utilisons `diff` pour expliciter la différentielle $\mathrm{d}u$ (appelée `del(u)` dans wxMaxima), puis nous exprimons $\mathrm{d}x$ en fonction de $\mathrm{d}u$ à l'aide de `solve`.
2. Extrayons l'équation résultante avec `%[1]` et nous remplaçons `del(x)` par son expression en fonction de `del(u)` dans l'intégrande.
3. Utilisons `subst` pour exprimer l'intégrande entière en fonction de u, puis effectuer l'intégration, en gardant à l'esprit que `integrate` attend uniquement le *coefficient* de `del(u)`.
4. Substituons la définition de u en fonction de x dans la primitive obtenue. Alternativement, pour une intégrale définie, on peut choisir de transformer les bornes d'intégration en fonction de u avant l'évaluation.

Exemple 6.3.5. Calculer $\int e^{5x}\,\mathrm{d}x$ en réalisant de manière explicite la substitution $u = 5x$. Vérifier la réponse en utilisant `diff`.

```
(%i14) INTEGRAND:(%e^(5*x))*diff(x);
(%o14) %e^(5*x)*del(x)
(%i15) solve(diff(u)=diff(5*x),del(x));
(%o15) [del(x)=del(u)/5]
(%i16) %[1];
(%o16) del(x)=del(u)/5
(%i17) subst(rhs(%),del(x),INTEGRAND)$
       subst(u,5*x,%);
(%o18) (%e^u*del(u))/5
(%i19) integrate(coeff(%,del(u)),u);
(%o19) %e^u/5
(%i20) subst(5*x,u,%);
(%o20) %e^(5*x)/5
```

Finalement, nous vérifions la réponse en utilisant `diff`.

```
(%i21) diff(%,x);
(%o21) %e^(5*x)
```

Exemple 6.3.6. Calculer $\int \frac{\sec^2 x}{\tan x}\,\mathrm{d}x$ en utilisant un changement de variable pour transformer l'intégrale. Vérifier la réponse en utilisant `diff`.

Nous remarquons que la dérivée de $\tan x$ apparaît à côté de $\mathrm{d}x$ dans l'intégrande, nous pouvons donc transformer l'intégrale en une fonction de $u = \tan x$ à l'aide d'une substitution.

```
(%i22) INTEGRAND:((sec(x))^2)/tan(x)*diff(x);
(%o22) (sec(x)^2*del(x))/tan(x)
(%i23) solve(diff(u)=diff(tan(x)),del(x));
(%o23) [del(x)=del(u)/sec(x)^2]
(%i24) %[1];
(%o24) del(x)=del(u)/sec(x)^2
(%i25) subst(rhs(%),del(x),INTEGRAND)$
       subst(u,tan(x),%);
(%o26) del(u)/u
(%i27) integrate(coeff(%,del(u)),u);
(%o27) log(u)
(%i28) subst(tan(x),u,%);
(%o28) log(tan(x))
```

Vérifions notre travail :

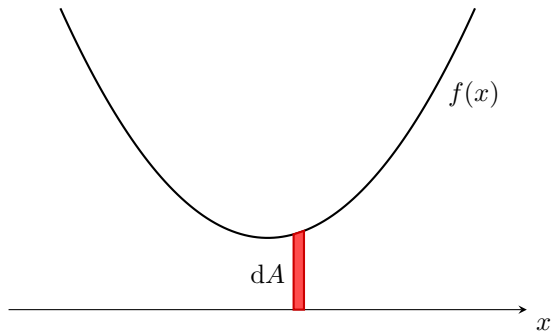
```
(%i29) diff(%,x);
(%o29) sec(x)^2/tan(x)
```

6.4 Intégrales de surface

6.4.1 Intégration de surface et intégration physique

Rappelons que $\int_a^b f(x)\ \mathrm{d}x$ nous donne l'aire délimitée entre $f(x)$ et l'axe des x. Il est utile de visualiser un élément d'aire en x, représenté par un rectangle fin de largeur $\mathrm{d}x$. Nous appelons cette "fine tranche" $\mathrm{d}A$.

Un élément d'aire $\mathrm{d}A$ pour une fonction $f(x)$



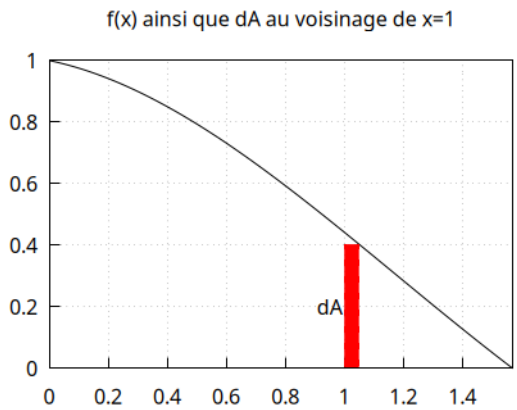
Une fois que $\mathrm{d}A$ est exprimé entièrement en fonction d'une seule variable (ici, $\mathrm{d}A = f(x) \cdot \mathrm{d}x$), nous utilisons l'intégration pour sommer tous les éléments d'aire. Nous pouvons écrire $A = \int \mathrm{d}A = \int_a^b f(x)\ \mathrm{d}x$, et nous considérons l'intégrale comme un *processus de sommation* permettant d'additionner les $\mathrm{d}A$. Cette approche conceptuelle est souvent appelée "intégration physique", et elle s'avère très puissante lorsque nous appliquons l'intégration dans un contexte géométrique.

Exemple 6.4.1. Représentez graphiquement $f(x) = e^{-0.2\cdot x} \cdot \cos x$ sur $[0, \frac{\pi}{2}]$ ainsi qu'un élément d'aire au voisinage de $x = 1$. Définissez l'intégrale $A = \int \mathrm{d}A$ et exprimez $\mathrm{d}A$ en fonction de x afiin d'obtenir une intégrale définie. Pour terminer, utilisez wxMaxima pour calculer l'aire et réalisez un graphique incluant un ombrage de l'aire.

On commence par l'esquisse de $f(x)$ et de dA :

```
(%i1) f(x):=%e^(-0.2*x)*cos(x)$
      wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      title="f(x) ainsi que dA au voisinage de x=1",
      color=black,
       explicit(f(x),x,0,%pi/2),
```

```
      border=false,
      color=red,
       rectangle([1,0],[1.05,f(1.05)]),
      color=black,
       label(["dA",0.95,0.2])
      );
```



dA est simplement le produit de la hauteur par la largeur pour l'élément d'aire : $f(x) \cdot \mathrm{d}x$.
Mainternant, nous définissons l'intégrale : $A = \int \mathrm{d}A = \int f(x)\ \mathrm{d}x = \int_0^{\frac{\pi}{2}} e^{-0.2\cdot x} \cdot \cos x\ \mathrm{d}x$

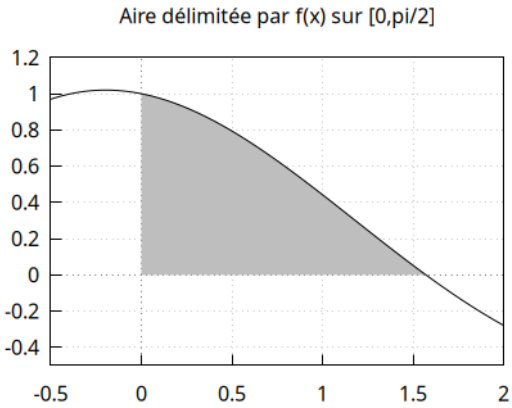
L'intégrale est simple à calculer avec `integrate`. On commence par `ratprint:false` pour éviter une liste d'avertissements émis par la commande `ratprint`, puis on obtient une approximation décimale avec `float` :

```
(%i2) ratprint:false$
(%i3) integrate(f(x),x,0,%pi/2);
(%o3) (25*%e^(-%pi/10))/26+5/26
(%i4) float(%);
(%o4) 0.89461797216216
```

On termine par un tracé avec l'aire ombrée :

```
(%i5) wxdraw2d(
      grid=true,
      xrange=[-0.5,2],
      yrange=[-0.5,1.2],
      xaxis=true,
      yaxis=true,
      title="Aire délimitée par f(x) sur [0,pi/2]",
      fill_color=grey,
      filled_func=true,
       filled_func=f(x),
       explicit(0,x,0,%pi/2),
      filled_func=false,
```

```
color=black,
 explicit(f(x),x,-0.5,2)
);
```



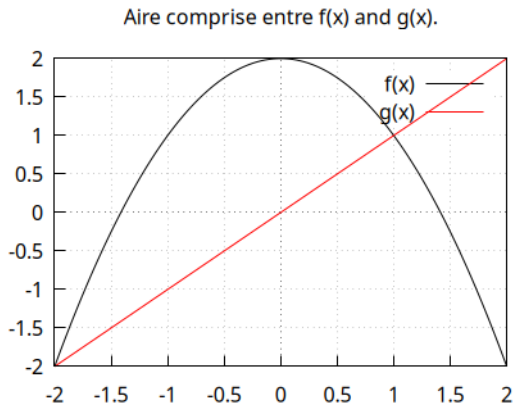
6.4.2 Aire comprise entre deux fonctions

Pour calculer l'aire comprise *entre* deux fonctions, il suffit de visualiser un élément de surface et d'exprimer $\mathrm{d}A$ à l'aide des fonctions données. Ensuite, on utilise l'intégrale $A = \int \mathrm{d}A$ pour sommer tous les éléments de surface sur l'intervalle approprié.

Exemple 6.4.2. Calculez l'aire comprise entre $f(x) = 2 - x^2$ et $g(x) = x$.

Nous commençons par une esquisse rapide :

```
(%i6) f(x):=2-x^2$
      g(x):=x$
(%i8) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[-2,2],
      yrange=[-2,2],
      title="Aire comprise entre f(x) and g(x).",
      color=black,
      key="f(x)",
       explicit(f(x),x,-2,2),
      color=red,
      key="g(x)",
      explicit(g(x),x,-2,2)
      );
```

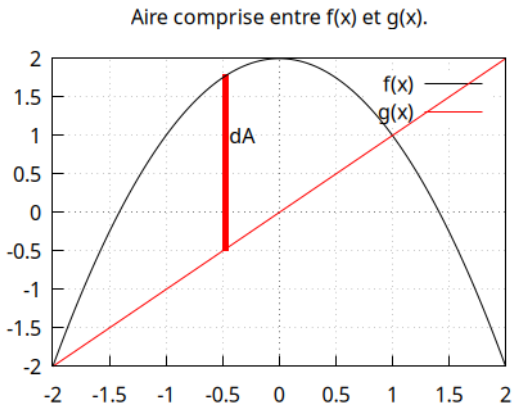


Maintenant, nous déterminons l'intervalle d'intégration approprié : l'aire cherchée se situe entre les deux points d'intersection de nos courbes.

```
(%i9) solve(f(x)=g(x),x);
(%o9) [x=1,x=-2]
```

Enfin, nous pouvons visualiser un élément d'aire sur l'intervalle d'intégration et en écrire l'expression algébrique. Ici, nous utilisons un rectangle fin pour représenter un élément de surface au voisinage de $x = -0{,}5$:

```
(%i10) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[-2,2],
       yrange=[-2,2],
       title="Aire comprise entre f(x) et g(x).",
       color=red,
       border=false,
        rectangle([-0.5,g(-0.5)],[-0.45,f(-0.45)]),
       color=black,
        label(["dA",-0.33,1]),
       key="f(x)",
        explicit(f(x),x,-2,2),
       color=red,
       key="g(x)",
        explicit(g(x),x,-2,2)
       );
```

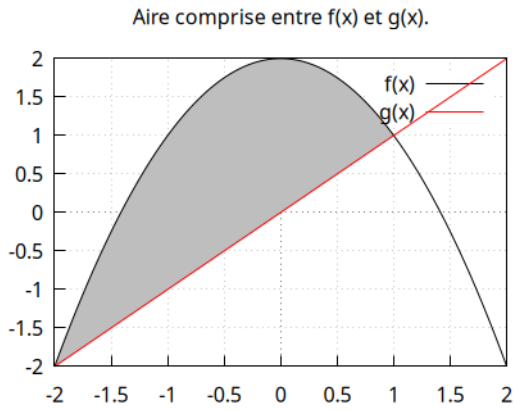


Nous voyons que l'élément d'aire a une hauteur de $f(x) - g(x)$, donc nous en déduisons que $\mathrm{d}A = [f(x) - g(x)] \cdot \mathrm{d}x$. Nous sommons les éléments d'aire en calculant l'intégrale $A = \int \mathrm{d}A = \int_{-2}^{1} [f(x) - g(x)]\ \mathrm{d}x$:

```
(%i11) float(integrate((f(x)-g(x)),x,-2,1));
(%o11) 4.5
```

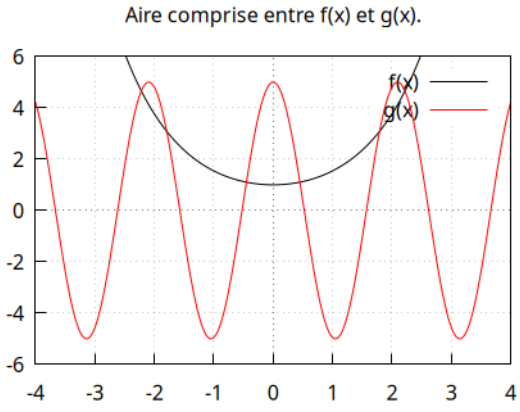
Nous obtenons $A = 4.5$ unités d'aire. Nous pouvons aussi réaliser un graphique mettant en évidence l'aire calculée précédemment :

```
(%i12) wxdraw2d(
      grid=true,
      xaxis=true,
      yaxis=true,
      xrange=[-2,2],
      yrange=[-2,2],
      title="Aire comprise entre f(x) et g(x).",
      fill_color=grey,
       filled_func=true,
        filled_func=f(x),
        explicit(g(x),x,-2,1),
       filled_func=false,
     color=black,
     key="f(x)",
      explicit(f(x),x,-2,2),
     color=red,
     key="g(x)",
      explicit(g(x),x,-2,2)
     );
```



Exemple 6.4.3. Calculer l'aire comprise entre $f(x) = \cosh x$ et $g(x) = 5\cos 3x$.
Nous commençons par l'esquisse des courbes :

```
(%i13) f(x):=cosh(x)$
       g(x):=5*cos(3*x)$
       wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[-4,4],
       yrange=[-6,6],
       title="Aire comprise entre f(x) et g(x).",
       color=black,
       key="f(x)",
        explicit(f(x),x,-4,4),
       color=red,
       key="g(x)",
       explicit(g(x),x,-4,4)
       );
```



Nous constatons que les deux fonctions sont paires, donc nous pouvons simplement calculer l'aire à droite de $x = 0$ et doubler le résultat. Pour vérifier que les fonctions sont paires :

```
(%i14) f(-x);
       g(-x);
(%o14) cosh(x)
(%o15) 5*cos(3*x)
```

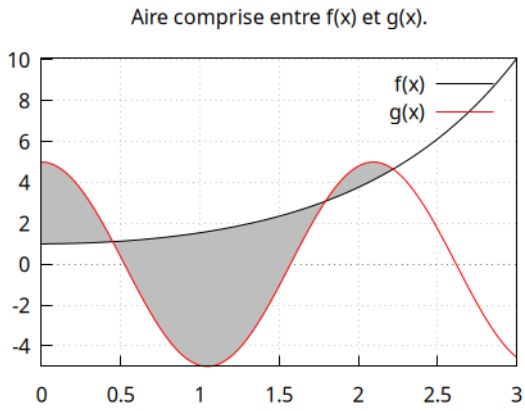
Il faut garder à l'esprit que nous ne calculons plus une aire *algébrique*, mais une aire *géométrique*, qui est toujours positive. Ainsi, lorsque $f(x)$ est au-dessus de $g(x)$, l'élément d'aire est $[f(x) - g(x)] \cdot \mathrm{d}x$, mais lorsque $g(x)$ est au-dessus de $f(x)$, l'élément d'aire devient $[g(x) - f(x)] \cdot \mathrm{d}x$. Nous devons diviser l'intervalle d'intégration en trois parties selon les points d'intersection de $f(x)$ et $g(x)$, puis additionner les aires et multiplier par 2 :

```
(%i16) x1:find_root(f(x)-g(x),0,1);
       x2:find_root(f(x)-g(x),1.5,2);
       x3:find_root(f(x)-g(x),2,2.5);
(%o16) 0.44947432956866
(%o17) 1.792473165779467
(%o18) 2.219127938640189
(%i19) ratprint:false$
       A1:float(integrate(g(x)-f(x),x,0,x1));
       A2:float(integrate(f(x)-g(x),x,x1,x2));
       A3:float(integrate(g(x)-f(x),x,x2,x3));
(%o20) 1.160865665363456
(%o21) 5.39123022939955
(%o22) 0.29427502144913
(%i23) A1+A2+A3;
(%o23) 6.846370916212132

(%i24) %*2;
(%o24) 13.69274183242426
```

Nous obtenons pour l'aire totale l'approximation suivante $A \approx 13.7$. Nous terminons avec un graphique où l'aire est ombrée sur la partie droite :

```
(%i25) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       xrange=[0,3],
       title="Aire comprise entre f(x) et g(x).",
       fill_color=grey,
        filled_func=true,
         filled_func=g(x),
         explicit(f(x),x,0,x1),
        filled_func=false,
        filled_func=true,
```

```
 filled_func=f(x),
 explicit(g(x),x,x1,x2),
filled_func=false,
filled_func=true,
 filled_func=g(x),
 explicit(f(x),x,x2,x3),
 filled_func=false,
color=black,
key="f(x)",
 explicit(f(x),x,0,3),
color=red,
key="g(x)",
 explicit(g(x),x,0,3)
);
```



6.4.3 Valeur moyenne

La **valeur moyenne** d'une fonction $f(x)$ sur $[a, b]$ répond à la question suivante : « quelle fonction *constante* $g(x) = f_{moy}$ délimiterait la même aire que $f(x)$ sur $[a, b]$? ». La valeur moyenne est particulièrement utile lorsque les fonctions représentent des grandeurs physiques dont l'aire bornée a une interprétation concrète (vitesse, puissance, etc.).

Nous trouvons rapidement une formule pour la valeur moyenne :

$$f_{avg} \cdot (b - a) = \int_a^b f(x)\ \mathrm{d}x \implies f_{avg} = \frac{1}{b-a} \int_a^b f(x)\ \mathrm{d}x$$

Exemple 6.4.4. Calculer la valeur moyenne de $f(x) = e^x$ sur $[1, 3]$.

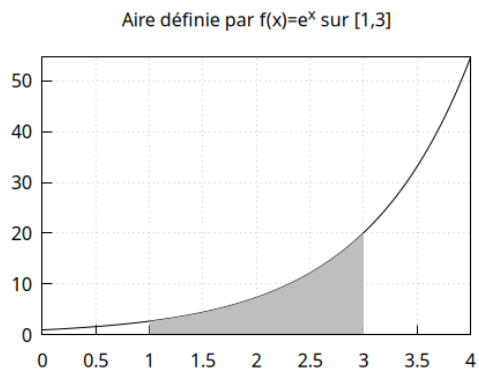
Nous intégrons et divisons simplement par la longueur de l'intervalle :

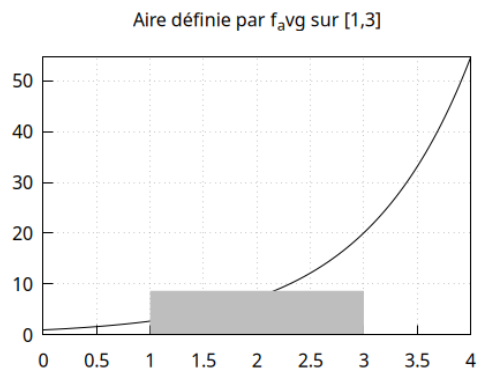
```
(%i26) f(x):=%e^x$
```

```
(%i27) float(f_avg:(1/(3-1))*integrate(f(x),x,1,3));
(%o27) 8.683627547364312
```

Donc $f_{avg} \approx 8.7$. Nous pouvons visualiser cette interprétation de l'aire grâce à des graphiques mettant en évidence les aires définies par les deux courbes :

```
(%i28) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       title="Aire définie par f(x)=e^x sur [1,3]",
       color=black,
        explicit(f(x),x,0,4),
       fill_color=grey,
        filled_func=true,
        filled_func=f(x),
        explicit(0,x,1,3),
        filled_func=false
       );

(%i29) wxdraw2d(
       grid=true,
       xaxis=true,
       yaxis=true,
       title="Aire définie par f_avg sur [1,3]",
       color=black,
        explicit(f(x),x,0,4),
       fill_color=grey,
        filled_func=true,
        filled_func=f_avg,
        explicit(0,x,1,3),
        filled_func=false
       );
```





La valeur moyenne f_{moy} délimite la même aire que $f(x)$ à l'aide d'un simple rectangle.

6.5 Exercices du Chapitre 6

1. Trouvez des primitives des deux fonctions suivantes en utilisant la méthode d'essai-erreur. Vérifiez vos propositions avec `diff` et affinez-les jusqu'à obtenir une primitive correcte.
   a. $f(x) = A\cos\frac{n\pi x}{L}$ (A, n, L constantes) b. $f(x) = xe^{-x^2}$
2. Devinez l'expression d'une primitive de la fonction $f(x) = \sin(2x)$ puis vérifier la réponse. Utilisez-la ensuite pour générer une famille de vingt primitives en faisant varier la constante additive. Tracez ensuite le graphique de $f(x)$ avec les primitives définies précédemment.
3. Pour l'exercice précédent, trouvez la primitive passant par le point $(-2, 3)$. Tracez le graphique de cette primitive particulière en montrant qu'elle passe par le point donné.
4. Utilisez `diff` pour montrer que $A(x) = \displaystyle\int_c^x f(t)\,\mathrm{d}t$ est une primitive de $f(x)$.
5. Utilisez une différence de fonctions d'aire avec comme point de départ $c = -1$ pour calculer l'aire délimitée par $f(x) = \frac{1}{\sqrt{1+x^2}}$ sur l'intervalle $[0, 1]$. Vérifiez votre réponse directement en utilisant la commande `integrate`. Tracez la courbe représentative de $f(x)$ et colorez l'aire que vous avez calculée.
6. Vérifiez la réponse de l'exemple 6.3.4 en utilisant `diff` au lieu de `integrate`. Que se passe-t-il ? Transformez la sortie de wxMaxima pour obtenir $\sin^2 x$ en utilisant `trigexpand` afin d'éliminer l'« angle double », puis utilisez `subst` pour imposer l'identité Pythagoricienne $\cos^2 x = 1 - \sin^2 x$.
7. Avec la pratique, le changement de variable "u" est généralement considéré comme "évident" et peut-être réalisé mentalement – nous regardons simplement l'intégrale comme une fonction de u à côté de la différentielle de u. Cependant, il existe encore des cas où un changement de variable plus formalisé est nécessaire. Utilisez le changement de variable explicite $u = \sqrt{2x-1}$ pour calculer $\displaystyle\int 3x\sqrt{2x-1}\,\mathrm{d}x$ à la manière des exemples 6.3.5 et 6.3.6. Vérifiez votre réponse directement avec `integrate`.
8. Lorsqu'on transforme une intégrale *définie* de x en u, il n'est pas nécessaire de revenir à x – on peut simplement transformer les bornes d'intégration en fonction de u et évaluer une primitive sur $[u_1, u_2]$:
   $\displaystyle\int_{x_1}^{x_2} f(x)\,\mathrm{d}x = \int_{u_1}^{u_2} g(u)\,\mathrm{d}u = G(u_2) - G(u_1)$ (pour trouver u_1, on remplace x_1 dans la définition de u utilisée pour la transformation, et de même pour u_2).
   Utilisez la substitution $u = \cos 2x$ pour calculer $\displaystyle\int_{\frac{\pi}{6}}^{\frac{\pi}{2}} \sin(2x)\cos^5(2x)\,\mathrm{d}x$.
   Transformez les bornes en fonction de u plutôt que d'exprimer la primitive en fonction de x. Vérifiez votre réponse avec `integrate`.
9. Rappelons que la position, la vitesse et l'accélération sont liées par $v(t) = x'(t)$ et $a(t) = v'(t) = x''(t)$. À partir d'une fonction d'accélération, il est possible de déterminer un ensemble d'équations décrivant la position et la vitesse d'un objet à tout instant futur (ces équations sont appelées les *équations du mouvement*). Trouvez les équations du mouvement pour un objet soumis à une accélération constante, a. Commencez par calculer une primitive de a (en incluant une

constante arbitraire) pour obtenir $v(t)$. Ensuite, évaluez $v(t)$ en $t = 0$ afin de déterminer la valeur de la constante arbitraire. Notez que la notation standard pour $v(0)$ est v_0, considérée ici comme une constante donnée. Répétez le processus pour obtenir une équation pour $x(t)$ en fonction de x_0, v_0, a et t.

10. Trouvez les équations du mouvement pour un objet avec $a(t) = x^3 e^{-0.5x}$. Supposons que $v(0) = 0$ et $x(0) = 0$; c'est-à-dire que la position et la vitesse initiales sont toutes deux nulles.

11. La position d'un objet est la primitive de la fonction vitesse, $v(t) = x'(t)$, donc le changement de position d'un objet (le *déplacement*) est calculé par $\int_{t_1}^{t_2} v(t)\ \mathrm{d}t = x(t_2) - x(t_1)$. Les déplacements sont positifs lorsque $v > 0$ et négatifs lorsque $v < 0$. Si l'on souhaite mesurer la *distance totale parcourue*, on peut le faire en comptant tous les déplacements négatifs comme positifs. On prend la valeur absolue de $v(t)$ dans l'intégrale :

$$DISTANCE = \int_{t_1}^{t_2} |v(t)|\ \mathrm{d}t$$

Un oscillateur avec une amplitude décroissante pourrait avoir sa hauteur donnée par l'équation $h(t) = 0{,}15e^{-0{,}2t} \cos(4{,}45 \cdot t)$ où la hauteur est mesurée en mètres (m) et le temps en secondes (s). Tracez la courbe de $h(t)$, puis calculez la distance totale parcourue par l'oscillateur avant qu'il ne se "stabilise" autour de zéro. Notez que wxMaxima ne peut pas gérer les intégrales impliquant des valeurs absolues, mais vous pouvez utiliser une boucle `do` pour arriver à trouver une suite de distances sur des intervalles de temps plus grands jusqu'à ce que la réponse se stabilise à au moins quelques décimales. Vous devrez calculer de nombreuses intégrales en divisant l'intervalle d'intégration en sous intervalles sur lesquels l'intégrande est positive et en sous intervalles sur lesquels l'intégrande est négative.

12. *La puissance* est le taux de variation de l'énergie par rapport au temps. Par exemple, si la puissance délivrée par une ampoule est de 100 Watts, cela signifie que 100 Joules d'énergie sont transférés par seconde. La puissance est relevée chaque heure au tableau électrique pendant 12 heures, et on obtient les données suivantes :

Puissance (W)	Heure (hr)
535	6 : 00 am
940	7 : 00 am
1135	8 : 00 am
1050	9 : 00 am
720	10 : 00 am
400	11 : 00 am
410	12 : 00 pm
390	1 : 00 pm
410	2 : 00 pm
370	3 : 00 pm
405	4 : 00 pm
890	5 : 00 pm
1410	6 : 00 pm

Comme la puissance est la dérivée par rapport au temps de l'énergie, on peut

écrire $P(t) = E'(t)$. L'intégrale

$$\int_{t_1}^{t_2} P(t)\,dt$$

peut être calculée théoriquement en trouvant unh primitive de $P(t)$ et en l'évaluant aux bornes de l'intervalle de temps :

$$\int_{t_1}^{t_2} P(t)\,dt = E(t_2) - E(t_1)$$

Autrement dit, l'intégrale donne la variation totale de l'énergie sur l'intervalle considéré.

Utiliser une approximation trapézoïdale pour trouver la consommation totale d'énergie sur cette période de douze heures, puis calculer la puissance moyenne consommée.

13. Soit $f(x) = \sin x$. Considérons une "fine tranche" rectangulaire près de $x = \frac{\pi}{2}$, c'est-à-dire sur l'intervalle $\left[\frac{\pi}{2}, \frac{\pi}{2} + h\right]$. Utilisez une boucle `for-do` pour calculer la différence en pourcentage entre l'*aire exacte* et l'approximation rectangulaire pour $h = \frac{1}{2^i}$, $i = 1, 2, \ldots$. Continuez jusqu'à ce que l'approximation soit suffisamment précise pour que l'erreur soit inférieure à 0,1%.