الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

وزارة التعليم العالي والبحث العلمي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

N° Réf : ……

Centre Universitaire

Abd Elhafid Boussouf Mila

Institut des Sciences et Technologie **Département de Mathématiques et Informatique**

Mémoire préparé en vue de l'obtention du diplôme de Master

En : ***Mathématiques***

Spécialité : ***Mathématiques Fondamentales***

Language latex en mathématiques

Préparé par : Damak Feriel

Bezzag Amira

Soutenue devant le jury :

Rouibah Khaoula	***MCB C.U.Abd Elhafid Boussouf***	***Présidente***
Mehazzem Allal	***MCB C.U.Abd Elhafid Boussouf***	***Rapporteur***
Challouf Yassamine	***MCB C.U.Abd Elhafid Boussouf***	***Examinatrice***

Année Universitaire : 2022/2023

❖ Remerciement

Nos premiers remerciements vont à notre professeur de mémoire monsieur **Mehazzem Allal**, qui a encadré ce mémoire avec beaucoup de patience et de gentillesse. Il a motivé chaque étape de notre travail par des remarques pertinentes et il a pu nous faire progresser dans nos recherches.

Nous tenons à exprimer notre profonde gratitude à madame la présidente **Rouibah Khaoula**, qui nous a fait l'honneur de présider le jury.

Merci à madame l'examinatrice **Challouf Yassamine** pour avoir examiné nos travaux au titre des rapporteurs et pour avoir suivi l'évolution de notre mémoire et pour leur conseils essentiels.

Merci à nos parents qui ont veillé depuis l'école primaire pour qu'on puisse arriver à ce niveau on peux que leur exprimer toute notre gratitude et nos sincère reconnaissance.

On pense également à nos chers frères et nos chères sœurs pour leurs présences.

Enfin, merci à tous ceux qui ont partagé nos vie ces dernières années et ont rendu possible ce travail.

❖ Dédicace

Je dédie ce travail avec mes vœux de réussite, de prospérité et de bonheur.

A mes chers parent, pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de mes études, spécialement a mon père que J'ai le droit d'être

fier qu'il soit mon père, et que j'aie été élevé par lui, et que j'ai été meilleur que je ne pourrais jamais l'espérer. Je me sens

er chaque fois que quelqu'un mentionne mon nom, puis mentionne son nom avec lui, et j'aimerais que tout le monde sache qu'il est mon père que j'ai toujours aimé, aimé et dont je suis

fier. Que Dieu ait pitié de toi, mon bien le plus cher et mon espoir.

A mes chéres sœurs et mon chére frère, chacun en son nom pour leurs encouragements permanents, et leur soutien moral,

A ma binôme Feryal merci de votre patience et d'avoir pris la peine de compléter ce mémoire.

Je dédie ce travail à tous ceux qui passent leur temps à m'aider avec volonté et avec un haut niveau de passion et de grandeur.

A tous ceux et celles que j'aime de prés comme de loin, et à tous ceux qui m'ont soutenu.

Amira

❖ Dédicace

J'ai le grand plaisir de dédier ce modeste travail :

A ma très chère mère, qui me donne toujours l'espoir de vivre et qui n'a jamais cessé de prier pour moi

A mon très cher père, pour ses encouragements, son soutien, surtout pour son amour et son sacrifice afin que rien n'entrave le déroulement de mes études,

A mon mari pour son encouragement son soutien et sa motivation ,

A mon frère et ma sœur, A mon fils mes amis, et mes chers collèges , et tous ceux qui m'aident et compulsent ce modeste travail.

Enfin, je remercie mon binôme Amira, qui a contribué à la réalisation de ce travail.

Feryal

Table des matières

Introduction

LaTeX est un langage de composition de documents qui a été spécialement conçu pour répondre aux besoins des professionnels de la mathématique. Il est largement utilisé dans les milieux académiques et scientifiques en raison de sa capacité à produire des documents mathématiques de haute qualité et esthétiquement plaisants.

L'un des avantages majeurs de LaTeX réside dans sa capacité à générer des formules mathématiques complexes. Au lieu de devoir utiliser des éditeurs de texte classiques qui peuvent présenter des limitations en termes de mise en forme des équations, LaTeX permet aux auteurs de mathématiques de saisir des expressions mathématiques directement dans leur code source. Cela permet de produire des formules mathématiques claires, précises et bien formatées.

LaTeX propose une vaste gamme de commandes spéciales pour représenter différents types d'objets mathématiques. Par exemple, il est possible de créer des équations numérotées et non numérotées, des fractions, des racines carrées, des exposants et des indices, des intégrales, des matrices, des opérateurs mathématiques, des ensembles, des théorèmes, des preuves, et bien plus encore. La syntaxe de ces commandes est intuitive et permet aux auteurs de se concentrer sur la logique mathématique plutôt que sur la mise en forme.

En plus de la saisie de formules mathématiques, LaTeX facilite également l'organisation structure ée des documents mathématiques. Il offre une hiérarchie de commandes pour structurer les sections, sous-sections, théorèmes, définitions, preuves, bibliographies, etc. Cette organisation facilite la lecture et la compréhension des documents mathématiques, en fournissant une structure claire et cohérente.

LaTeX offre également une grande flexibilité en termes de personnalisation et de mise en page. Les utilisateurs peuvent choisir parmi une variété de modèles prédéfinis adaptés aux documents mathématiques, ou ils peuvent personnaliser complètement l'apparence de leurs documents en modifiant les polices, les couleurs, les marges, les en-têtes, les pieds de page et d'autres paramètres. Cela permet aux auteurs de créer des documents mathématiques qui répondent à leurs besoins spécifiques et qui correspondent à leur style personnel. De plus, les documents LaTeX sont indépendants du système d'exploitation et peuvent être compilés de manière fiable sur différentes plates-formes. Les fichiers sources LaTeX sont de simples fichiers texte qui peuvent être édités avec n'importe quel édi-

teur de texte et partagés facilement entre les collaborateurs. La compilation des documents LaTeX produit des fichiers PDF, qui peuvent être visualisés sur la plupart des ordinateurs sans avoir besoin de logiciels spécifiques.

En conclusion, LaTeX est un langage de composition de documents largement utilisé en mathématiques pour sa capacité à générer des formules mathématiques précises, à organiser de manière structurée les documents mathématiques et à offrir une flexibilité de personnalisation. Que ce soit pour rédiger des articles de recherche, des thèses, des livres ou des présentations,LaTeX est un outil puissant et indispensable pour les professionnels de la mathématique qui souhaitent produire des documents de haute qualité.

Chapitre 1

Préliminaires

Introduction

Le tex est un language de balisage utilisé pour créer des documents de haute qualité, tels que des articles scientifiques, des rapports techniques, des livres et des thèses. Il a été créé par Leslie Lamport en 1985 et est basé sur le language de programmation TeX créé par Donald Knuth.

Le tex est souvent utilisé dans les domaines de la science, de la technologie, de l'ingénirie et des mathématiques en raison de sa capacité à produire des formules mathématiques complexes avec une grande précision et un formatage cohérent. Il est également populaire dans la communauté universitaire en raison de sa capacité à gérer facilement les citations et les références bibliographiques.

Le language le tex utilise des commandes pour formater le texte, les tableaux, les images et les formules mathématiques. Il est hautement personnalisable et permet aux utilisateurs de créer des modèles réutilisables pour des types de documents spécifiques.

Au fil des ans. Le tex est devenu une norme pour la rédaction de documents scientifiques. Il est largement utilisé dans les universitéset les centres de recherche du monde entier. Aujourd'hui, il existe de nombreuses distributions le tex disponibles gratuitement, telles que TeXLive et MiKTeX, qui permettent aux utilisateurs de travailler avec le tex sur différents systèmes d'exploitation, tels que Windows, macOS et linux.

1.1 Définitions

Le pixel art : Le pixel art est **une forme d'art numérique qui utilise des pixels pour créer des images**. Les pixels sont de petits éléments carrés qui peuvent être agrandis ou réduits pour créer des images de différentes tailles.

MetaPost : est un système qui implémente un langage de construction de figures et est interpréteur de ce `langage`. Il dérive du système `Metafont` du professeur `DonaldKnuth`, et est

spécialisé dans la production de diagrammes en langage PostScript à partir d'une description géométrique et algébrique. Le langage permet, en utilisant la syntaxe du langage Metafont, de combiner des lignes, des courbes, des points et d'effectuer des transformations géométriques. Toutefois, MetaPost se différencie de Metafont sur plusieurs points :

- Metafont est conçu pour produire des polices de caractère, sous forme de fichiers d'image (d'extension .gf) et des fichiers associés (d'extension .tfm) contenant des informations métriques de police, tandis que MetaPost produit des fichiers au format PostScript ;
- Metafont produit des polices avec une résolution fixe dans un format matriciel, tandis que MetaPost produit des graphiques dans un format vectoriel, le PostScript ;
- Metafont travaille en monochrome, tandis que MetaPost gère les couleurs en employant un format RVB
- Le langage MetaPost permet d'inclure des boîtes contenant du texte dans les diagrammes, des chaînes de caractère écrites dans une police donnée, ou beaucoup de choses qui peuvent être composées avec TeX ;
- L'interpréteur de Metafont a été écrit par DonaldKnuth sous une licence de sourcelibre, permettant à John D. Hobby et plus tard à Ulrik Vieth d'adapter l'interpréteur à leurs propres exigences, pour donner MetaPost.

Asymptote : (prononciation :/a.s□p.t□t/) Le terme d'asymptote est utilisé en mathématiques pour préciser des propriétés éventuelles d'une branche infinie de courbe à accroissement tendant vers l'infinitésimal. C'est d'abord un adjectif d'étymologie grecque qui peut qualifier une droite, un cercle, un point… dont une courbe plus complexe peut se rapprocher. C'est aussi devenu un nom féminin synonyme de *droite asymptote*.
Une **droite asymptote** à une courbe est une droite telle que, lorsque l'abscisse ou l'ordonnée tend vers l'infini, la distance de la courbe à la droite tend vers 0.
L'étude du **comportement asymptotique** est particulièrement développée dans lesétudes de fonctions et présente des commodités reconnues par de nombreux mathématiciens. Dans le domaine scientifique, il arrive fréquemment d'étudier des fonctions dépendant du temps (évolution de populations, réaction chimique ou nucléaire, graphique de température, oscillation d'un amortisseur). Un des objectifs du chercheur est alors de connaître l'état *à la fin de l'expérience*, c'est-à-dire lorsqu'un grand intervalle de temps s'est écoulé. L'objectif n'est alors pas de connaître les variations intermédiaires mais de déterminer le comportement stable, à *l'infini* du phénomène mesuré.

XY-pic : XY-pic est un paquetage pour la composition de graphiques et de diagrammes en utilisant le système de composition TEX de Knuth. XY-pic fonctionne avec la plupart des formats

disponibles, par exemple, TEX simple, LATEX et AMS-TEX, LATEX et AMS-TEX. Plusieurs styles d'entrée pour différents types de diagrammes sont pris en charge.
Ils partagent tous une technique basée sur la composition visuels. Ce guide se concentre sur la manière de composer les diagrammes de type "matrice", tels que les diagrammes commutatifs.

Inkscape : Inkscape est un logiciel de dessin vectoriel de qualité professionnelle qui fonctionne sur Windows, macOS et GNU/Linux. Inkscape est utilisé par des designers professionnels et des amateurs dans le monde entier pour créer une grande variété de graphismes tels que des illustrations, des icônes, des logos, des diagrammes, des cartes et des rendus pour le web.

PGFPlots : PGFPlots dessine des tracés de fonctions de haute qualité en échelle normale ou logarithmique avec une interface facile à utiliser directement dans TeX.

CTAN : CTAN est un acronyme pour « *Comprehensive TeX Archive Network* » (en français, « réseau complet d'archives TeX »). On peut y trouver et télécharger du matériel concernant TeX comme des logiciels ou des polices de caractères. Des dépôts pour d'autres projets, tels que MiKTeX, une distribution de TeX, sont des miroirs régulièrement mis à jour de CTAN.

1.2 C'est quoi LaTeX ?

Le LaTeX (prononcé "LAY-tek" ou "LAH-tek") est un outil de composition de documents de qualité professionnelle. Cependant, le mode de fonctionnement du LaTeX est très différent de celui de nombreuses autres applications de production de documents que vous avez peut-être utilisées, telles que Microsoft Word ou Libre Office Writer , ces outils "WYSIWYG" fournissent aux utilisateurs une page interactive dans laquelle ils tapent et modifient leurs texte et appliquent diverses sortes de mise en forme.
Le LaTeX fonctionne très différemment : votre document est un fichier de texte brut entrecoupé de commandes le LaTeX utilisées pour exprimer les résultats (typographiques) souhaités.
Pour produire un document visible et le composer, votre fichier LaTeX est traité par un logiciel appelé moteur TeX, qui utilise les commandes intégrées dans votre fichier texte pour guider et contrôler le processus de composition, convertissant les commandes LaTeX et le texte du document en un fichier PDF composé de manière professionnelle.
Cela signifie que vous n'avez qu'à vous concentrer sur le contenu de votre document et que l'ordinateur, via les commandes LaTeX et le moteur TeX, se chargera de l'aspect visuel (formatage).

1.3 Pourquoi apprendre LaTeX ?

Plusieurs arguments peuvent être présentés pour ou contre l'apprentissage du LaTeX plutôt que d'autres applications de création de documents , mais en fin de compte, il s'agit d'un choix personnel basé sur les préférences, les affinités et les exigences en matière de documentation.
Les arguments en faveur du LaTeX sont les suivants :

- la possibilité de composer des tableaux et des contenus techniques complexes pour les sciences physiques
- des facilités pour les notes de bas de page, les références croisées et la gestion de la bibliographie
- la production facile d'éléments de documents complexes ou longs comme les index, les dictionnaires, les tables des matières et les listes de figures
- un haut degré de personnalisation pour la production de documents sur mesure, grâce à sa programmabilité interne et à son extensibilité par le choix de milliers de modules complémentaires gratuits.

Un avantage important du LaTeX est la séparation entre le contenu et le style du document :
une fois que vous avez rédigé le contenu de votre document, son apparence peut être facilement modifiée. De même, vous pouvez créer un fichier LaTeX qui définit la mise en page/le style d'un type de document particulier et ce fichier peut être utilisé comme modèle pour normaliser la rédaction/-production d'autres documents de ce type ; par exemple, cela permet aux éditeurs scientifiques de créer des modèles d'articles, en LaTeX, que les auteurs utilisent pour rédiger des articles à soumettre à des revues.
Dans l'ensemble, Le LaTeX offre aux utilisateurs un grand contrôle sur la production des documents dont la composition répond à des normes extrêmement élevées. Bien entendu, il existe des types de documents ou de publications pour lesquels le LaTeX n'est pas très performant, notamment de nombreux modèles de pages "libres" que l'on trouve généralement dans les publications de type magazine.

1.4 Installation du LaTeX

Pour utiliser LATEX, il est nécessaire d'installer une distribution appropriée pour votre système d'exploitation. Les distributions fournissent des programmes qui automatisent la configuration et

l'installation de LATEX, TEX ainsi que tous les utilitaires associés.

Sous Unix : on trouve encore la distribution TEX bien que son développement ait été stoppé en 2006. Aujourd'hui, sur un système Unix, on installe généralement la TEXLive http://www.tug.org/texlive

Sous MacOS : la distribution de référence est MacTEX http://www.tug.org/mactex

Sous Windows : le plus simple est sans doute de choisir proTEXt http://www.tug.org/protext qui installe la distribution MiKTEX http://www.miktex.org et quelques outils de développement dont un programme de visualisation de fichiers au format PostScript (gsview).

Il faut parfois ajouter à ces distributions (si elles n'en contiennent pas déjà un) un éditeur de texte puisque vous le découvrirez bien assez tôt, utiliser le LATEX c'est taper du texte et des commandes dans des fichiers :

- Emacs et vi, les deux principaux éditeurs Unix, génèrent encore des conflits inutiles entre leurs utilisateurs, souvent teintés de préjugés, malgré la nette supériorité d'Emacs sur vi.
- kile et texmaker sont des environnements de développement intégrés qui offrent aux utilisateurs débutants une expérience facile pour leurs premiers pas. Ces deux outils se distinguent par leur capacité à réunir l'édition, la compilation et la visualisation dans une seule interface. De plus, ils permettent aux utilisateurs de se débrouiller avec les commandes de LaTeX à l'aide de menus, de boîtes de dialogue et d'onglets explicatifs.
- TEXnicCenter est l'équivalent sous la marque « à la fenêtre »
- TEXshop et iTEXmax sont les équivalents sous la marque « à la pomme »

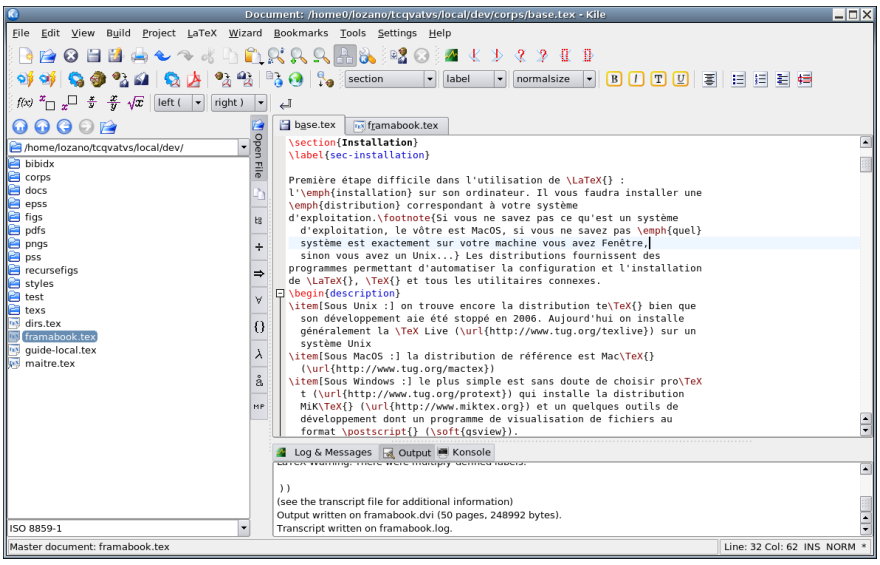


FIG. 1.1 : Kile

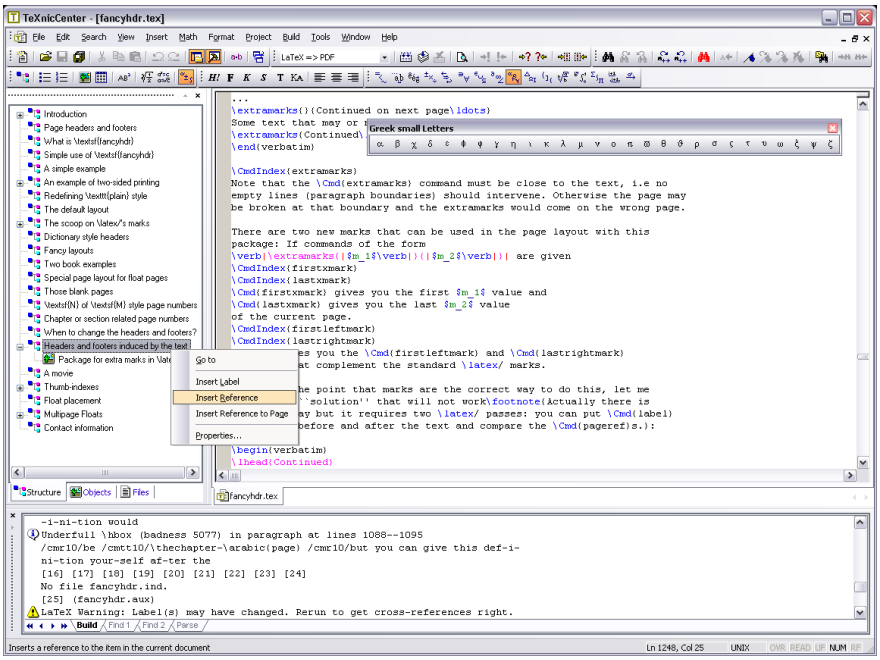


FIG. 1.2 : TEXnicCenter

Deux exemples d'environnement de développement intégré : Kile sous Linux et TEXnicCenter sous Windows.

Ils permettent de centraliser dans une même interface : édition, compilation & visualisation.
Vous apprendrez également bien assez vite que la production d'un document avec LATEX consiste à traduire (on dit aussi compiler) une source

- donc créé par un éditeur de texte
- en un format destiné à l'affichage ou à l'impression

Il existe donc, plus ou moins intégrés aux distributions, des outils célèbres pour la visualisation des différents fichiers résultants de la compilation :

Format PDF : mis à part le célèbre acrobat reader, il existe sous Unix des utilitaires permettant de visualiser le format Pdf : xpdf, evince, ...

Format DVI : xdvi, kdvi sous Unix et yap sous Windows font partie des programmes permettant de visualiser le résultat de la compilation d'un fichier LATEX

Format Postscript : la suite ghostscript disponible sous des noms qui peuvent varier selon la plateforme, permet de visualiser des fichiers au format PostScript

1.5 Utilisation d'Overleaf

Overleaf dispose d'une galerie contenant des milliers de modèles, couvrant un large éventail de types de documents, allant des articles scientifiques, des rapports et des livres aux CV et aux présentations. Comme ces modèles définissent la mise en page et le style du document, les auteurs n'ont qu'à les ouvrir dans Overleaf - en créant un nouveau projet - et à commencer à écrire pour ajouter leur contenu.

1.6 Écrire son premier texte en LaTeX

La première étape consiste à créer un nouveau projet LaTeX. Vous pouvez le faire sur votre propre ordinateur en créant un nouveau fichier .tex ; vous pouvez également lancer un nouveau projet dans Overleaf. Commençons par l'exemple de travail le plus simple, qui peut être ouvert directement dans Overleaf :

```
\documentclass{article}
\begin{document}
Premier document. Il s'agit d'un exemple simple,
sans param\`{e}tres ou paquets suppl\'{e}mentaires.
\end{document}
```

Ouvrez cet exemple dans Overleaf. Cet exemple produit la sortie suivante :
Premier document. Il s'agit d'un exemple simple, sans paramètres ou paquets supplémentaires.

Vous pouvez voir que LaTeX a automatiquement indenté la première ligne du paragraphe, s'occupant de ce formatage pour vous. Regardons de plus près ce que fait chaque partie de notre code.

La première ligne de code, `\documentclass{article}`, déclare le type de document connu sous le nom de classe, qui contrôle l'apparence générale du document. Différents types de documents requièrent différentes classes ; par exemple, un CV nécessitera une classe différente de celle d'un article scientifique qui pourrait utiliser la classe d'article standard de LaTeX. D'autres types de documents sur lesquels vous travaillez peuvent nécessiter des classes différentes, telles que livre ou rapport. Pour avoir une idée des nombreux types de classes LaTeX disponibles, consultez la page correspondante sur CTAN (Comprehensive TeX Archive Network). Après avoir défini la classe de document, notre contenu, appelé corps du document, est écrit entre les balises `\begin{document}` et `\end{document}`. Après avoir ouvert l'exemple ci-dessus, vous pouvez apporter des modifications au texte et, lorsque vous avez terminé, visualiser le PDF dactylographié résultant en recompilant le document.

1.7 Support en arabe

"ArabTEX" a des défauts. Il existe plusieurs packages pour prendre en charge la langue arabe dans LaTeX, dont le plus célèbre et le plus ancien est le package ArabTeX (Arabi). ArabTeX se base sur une seule police qui ne peut pas être modifiée (elle n'est pas esthétiquement plaisante selon les normes d'impression).
Il existe également un package appelé Arabi, plus moderne et très performant, mais il ne fonctionne qu'avec un nombre limité de polices prises en charge préalablement, et chaque nouvelle police doit être ajoutée individuellement (ce qui n'est pas une tâche simple).
Par défaut, le document est aligné de gauche à droite et donne des étiquettes spécifiques à un ensemble de composants du document. XeLaTeX joue un rôle important dans le traitement de la sortie du document, notamment pour des éléments tels que les figures, la bibliographie, les chapitres et la table des matières. De plus, les dates sont généralement formatées en latin. Afin de changer la sortie du document de XeLaTeX en arabe, nous devons lui apprendre à traiter la langue arabe en appliquant certaines commandes spéciales. Nous avons écrit les commandes nécessaires pour faire fonctionner XeLaTeX comme prévu et avons simplifié des centaines de commandes en trois lignes de base, qui doivent être ajoutées à votre document.
La commande principale est `\usepackage{polyglossia}`, qui est essentielle pour que votre texte soit en arabe. Ces lignes sont :

```
\usepackage{polyglossia}
\setdefaultlanguage[calendar=gregorian,numerals=maghrib]{arabic}
\setotherlanguage{english}
\newfontfamily\arabicfont[Script=Arabic,Scale=1.2]{Amiri}
\newfontfamily\arabicfontsf[Script=Arabic]{ae_AlBattar}
\newfontfamily\arabicfonttt[Script=Arabic]{Times New Roman}
```

Nous l'ajoutons directement dans le preamble.
La langue principale, qui est ici l'arabe, est activée par la commande.

```
\setdefaultlanguage[calendar=gregorian,numerals=maghrib]{arabic}
```

Notez que l'option `numerals=maghrib` s'applique aux nombres qui sont arabes dans ce cas (c'est-à-dire utilisés dans divers Partout dans le monde, sauf dans certains pays arabes !

1. La deuxième langue utilisée, qui est l'anglais, est activée avec la commande : `\setotherlanguage{english}`.
2. La quatrième ligne définit le style de calligraphie arabe utilisé pour la sortie du document. Ici, le type Amiri a été choisi. Notez que l'option `Scale=1.2` concerne la multiplication de la ligne 1.2
3. La cinquième ligne et la dernière ligne nous permettent d'utiliser deux types de polices en plus de la police principale
   - La première police secondaire est appelée par la commande `\sffamily{...}`
   - La deuxième police secondaire est appelée par la commande `\ttfamily{...}`

   Cette bibliothèque permet d'ajouter des fonctionnalités à LaTeX pour prendre en charge la langue arabe, ainsi que d'autres possibilités. Par exemple, nous pouvons utiliser la commande "LaTeX is nice" pour écrire une phrase de gauche à droite. Pour ce faire,
4. utilisez `\LR{...}` la commande. Si nous écrivons cette phrase sans utiliser cette commande, elle apparaîtra comme suit "LaTex is nice" : `\LR{LaTeX is nice}` , pour écrire une phrase de droite à gauche, encadrée entre des mots alignés de gauche à droite, utilisez la commande
5. `\RL{...}` . Pour écrire un ensemble de lignes alignées de gauche à droite,
6. utilisez l'environnement LTR .

```
\documentclass[11pt,a4paper,onside]{report} %تقرير
\usepackage{fontspec}
\usepackage{polyglossia}
\setdefaultlanguage[calendar=gregorian,numerals=maghrib]{arabic}
\setotherlanguage{english}
\newfontfamily\arabicfont[Script=Arabic]{Amiri}
\begin{document}
أصبح بإمكاننا إستخدام اللغة العربية للكتابة في لاتخ .\\
\LR{\LaTeX\ is nice}
\end{document}
```

والنتيجة:

أصبح بإمكاننا إستخدام اللغة العربية للكتابة في لاتخ.
LaTeX is nice

1.8 Création de présentations Beamer

Nous allons expliquer cela en détail à travers les points suivants : LATEX apprendra ici comment créer des présentations à l'aide du package Beamer

1. La bibliothèque
2. Les règles de rédaction des textes
3. La mise en forme (arrière-plans, couleurs, animations, ...).

La bibliothèque "Beamer package"

L'une d'entre elles, qui peut être utilisée par le package Beamer pour créer des présentations avec de nombreuses bibliothèques, est LaTeX. Nous pouvons utiliser LaTeX pour définir le contenu de chaque diapositive. Les diapositives sont créées en utilisant l'environnement frame et sont chargées en écrivant l'instruction `\documentclass{beamer}`."

```
\documentclass{beamer}
\begin{document}
محتوى الوثيقة
\end{document}
```

Quelques instructions de base

1. `\frametitle` : Utilisé pour titrer une diapositive avec un titre principal placé entre des accolades {}.

2. \framesubtitle : Utilisé pour titrer une diapositive avec un sous-titre placé entre des accolades {}.

3. \framesubtitle : Utilisé pour afficher le nom de l'auteur et le titre sur la page de titre."

```
\begin{frame}
\frametitle{ title }
\framesubtitle{Subtitle}
\end{frame}
```

```
\begin{frame}[option]
{ title }{subtitle}
[...]
\end{frame}
```

Chapitre 2

Le LaTeX en mathématiques

2.1 Commandes usuels

Un des aspects pratique et rigolo de LATEX est bien sûr la génération de formules mathématiques ; elles seront naturellement belles, sans que vous n'ayez à faire quoique ce soit. De plus, si vous avez un mauvais souvenir d'un certain éditeur d'équations, réjouissez vous vous n'avez pas besoin de souris pour écrire des équations ! La génération d'équations avec LATEX est un domaine particulièrement vaste. Nous présenterons ici les bases requises pour produire les formules « usuelles ». Ce chapitre ne constitue donc qu'une petite introduction à la manipulation des formules avec LATEX.
Les commandes standard de LATEX permettent de produire la plupart des équations mathématiques usuelles. Il est cependant conseillé d'utiliser les extensions de "American Mathematical Society" nommées `amsmath` et `amssymb` simplifiant la mise en forme dans beaucoup de situations.

Les deux façons d'écrire des maths

LATEX distingue deux manières d'écrire des mathématiques. L'une consiste à insérer une formule dans le texte, comme ceci : $ax + b = c$, l'autre à écrire une ou plusieurs formules dans un environnement, par exemple :

$$dU = \delta\mathcal{W} + \delta\mathcal{Q}$$

sachant que chacun de ces deux modes respectent un certain nombre de principes quant à la taille et la position des différents symboles. Voici un exemple avec les deux modes :

```
Déterminer la fonction dérivée
de $f(x)$ :
\begin{displaymath}
f(x)=\sqrt{\frac{x-3}{x+1}}
\end{displaymath}
si elle existe.
```

Déterminer la fonction dérivée de $f(x)$:

$$f(x) = \sqrt{\frac{x-3}{x+1}}$$

si elle existe.

Cet exemple nous montre donc que l'on entre en mode mathématique « interne » grâce au symbole $ et que le même symbole $ permet d'en sortir. D'autre part, on utilise ici l'environnement `displaymath` qui est le plus simple pour produire des équations. Ce dernier peut être saisi grâce aux commandes `\[et\]`

Commandes usuelles

Indice et exposant

_ et ^ sont les commandes permettant de produire respectivement *indice* et *exposant*. Il est nécessaire de « grouper » les arguments entre accolades pour que ces commandes agissent sur plusieurs symboles.

`x_2`	x_2	`x_{2y}`	x_{2y}	`x_{t_0}`	x_{t_0}
`x^2`	x^2	`x^{2y}`	x^{2y}	`x_{t^0}`	x_{t^0}
		`x^{2y}_{t_0}`	$x^{2y}_{t_0}$	`x_{t^1}^{2y}`	$x_{t^1}^{2y}$

Fraction et racine

Voici comment produire *racines* et *fractions* :

- la commande `\frac{num}{denom}` produit une fraction formée par le numérateur `num` et le dénominateur `denom` ;
- la commande `\sqrt[n]{expr}` affiche la racine n^e de son argument `arg`.

Notons que ces deux commandes ne produisent pas le même affichage selon le mode mathématique : interne ou équation. Ainsi voici une fraction : $\frac{1}{\sin x + 1}$ et une racine : $\sqrt{3x^2 - 1}$ et leur équivalent en mode équation :

$$\frac{1}{\sin x + 1} \quad \sqrt{3x^2 - 1}$$

Pour en finir avec ces deux commandes, voyons comment elles peuvent être imbriquées et l'effet que cela produit :

```
\begin{displaymath}
\sqrt{\frac{1+\sqrt[4]{3x^2+1}}
                {3x+\frac{1-x}{1+x}}}
\end{displaymath}
```

$$\sqrt{\frac{1+\sqrt[4]{3x^2+1}}{3x+\frac{1-x}{1+x}}}$$

Symboles

Symboles usuels

Le tableau page suivante donne les macros produisant une partie des symboles dont vous pourriez avoir besoin.
Nous avons recensé près de 450 symboles disponibles avec les `packages` `latexsym` et `amssymb`. Notre but n'est donc pas de les présenter ici ! Le tableau page suivante est une sélection parmi les symboles standard. Nous avons jugé qu'ils faisaient partie des `symboles` les plus utiles — ce qui, malgré la présence tout à fait fortuite dans ce tableau, démontre que le niveau en mathématiques de l'auteur de ce document avoisine le ras des pâquerettes.

Points de suspension

On utilise couramment pour économiser de l'encre des points de suspension dans des formules. Il en existe de trois types. La commande `\dots` produit des points « posés » sur la ligne

```
$C=\{\vec{c}_0,\vec{c}_1,\dots,\vec{c}_N\}$
est 'lensemble des $N$ couleurs.
```

$C = \{\vec{c}_0, \vec{c}_1, \dots, \vec{c}_N\}$ est l'ensemble des N couleurs.

\pm	$\pm$	\otimes	$\otimes$	\cong	$\cong$	\imath	$\imath$
\mp	$\mp$	\oslash	$\oslash$	\subset	$\subset$	\jmath	$\jmath$
\div	$\div$	\odot	$\odot$	\supset	$\supset$	\ell	ℓ
\ast	$\ast$	\leq	$\leq$	\subseteq	$\subseteq$	\aleph	$\aleph$
\times	$\times$	\geq	$\geq$	\supseteq	$\supseteq$	\nabla	∇
\bullet	$\bullet$	\equiv	$\equiv$	\in	$\in$	\\|	$\|$
\circ	$\circ$	\ll	$\ll$	ni	$\ni$	\partial	∂
\star	$\star$	\gg	$\gg$	\emptyset	$\emptyset$	\wedge	$\wedge$
\setminus	$\setminus$	\sim	$\sim$	\forall	$\forall$	\vee	$\vee$
\oplus	$\oplus$	\simeq	$\simeq$	\infty	∞	\cup	$\cup$
\ominus	$\ominus$	\approx	$\approx$	\exists	$\exists$	\cap	$\cap$

Symboles mathématiques usuels

La commande `\cdots` produit des points centrés verticalement sur le signe égal

```
$\vec{\mu}=\frac{1}{N}
(\vec{c}_0+\vec{c}_1+\cdots+\vec{c}_N)$
est la moyenne des $N$ couleurs.
```

$\vec{\mu} = \frac{1}{N}(\vec{c}_0 + \vec{c}_1 + \cdots + \vec{c}_N)$ est la moyenne des N couleurs.

Enfin les commandes `\vdots` et `\ddots` sont à utiliser essentiellement dans les matrices. Ces deux commandes produisent respectivement : $\vdots$ et $\ddots$.

Flèches

Voici un moyen simple pour mémoriser les commandes permettant de générer des flèches :

- toutes les commandes finissent par `arrow` ;
- le préfixe obligatoire `left` ou `right` indique la direction ;
- le préfixe facultatif `long` donne une version longue ;
- la première lettre de la commande mise en majuscule rend la flèche double ;

- on peut mettre des flèches aux deux extrémités en collant les deux mots `left` et `right`.

ainsi :

`\rightarrow`	donne	$\rightarrow$
`\Longleftarrow`	donne	$\Longleftarrow$
`\Leftarrow`	donne	$\Leftarrow$
`\Longleftrightarrow`	donne	$\Longleftrightarrow$

Lettres grecques

Les lettres grecques s'utilisent de la manière la plus simple qui soit : en les appelant par leur nom. Ainsi : `\alpha` donne « α » et `\pi`, « π ». Mettre une majuscule à la première lettre de la commande, donne la majuscule correspondante : `\Gamma` donne « Γ ». Attention, toutes les majuscules ne sont pas disponibles dans l'alphabet grec, on mettra par exemple α en majuscule, avec la lettre A (la commande `\Alpha` n'existe pas).

L'ensemble des réels

Une question « cruciale » que se posent les rédacteurs potentiels de documents scientifiques est : « Comment peut/doit-on écrire le 'R' de l'ensemble des réels ? ». Les avis sont partagés à ce sujet. Historiquement il semble qu'initialement, dans les ouvrages de mathématiques, le symbole des réels était typographié en gras (« Soit $x \in \mathbf{R}$ ») et que les professeurs pour reprendre ces notations sur un tableau avec une craie avaient recours à l'artifice de repasser plusieurs fois sur la lettre « $\mathbf{R}$ » ; cette pratique pénible aurait évoluée vers l'écriture « bien connue » : « Soit $x \in \mathbb{R}$ ». Il y a donc les adeptes du $\mathbf{R}$, du $\mathbb{R}$, etc. Pour choisir par soi-même, voir les packages :

- `bbm` qui propose la commande `\mathbbm{R}` produisant $\mathbb{R}$, la commande `\mathbbmss{R}` produisant $\mathbb{R}$, etc.
- `bbold` qui propose la commande `\mathbbm{R}` produisant $\mathbb{R}$, etc.
- `amssymb` qui propose les commandes `\mathbb{R}` produisant $\mathbb{R}$ ainsi que `\mathbf{R}` produisant $\mathbf{R}$

Fonctions

Fonctions standards

Lorsqu'on veut produire des fonctions mathématiques classiques (logarithmes, trigonométrie,...), il faut utiliser les fonctions de LATEX prévues à cet effet. Voici un exemple pour vous en convaincre.

```
$\sin^2x + \cos^2 x=1$
```

$\sin^2 x + \cos^2 x = 1$

Et sans les fonctions LATEX :

```
$sin^2x + cos^2x=1$
```

$sin^2x + cos^2x = 1$

La différence réside dans le fait que LATEX traite la chaîne *cos* comme une suite de variable (donc produites en italiques) alors que la fonction `\cos` produit « *cos* » en roman. Une autre différence importante est le placement d'éventuels indices (cf. l'exemple de la fonction `\max` ci-dessous). Parmi les fonctions mathématiques standard de LATEX, on trouvera :

- toutes les fonctions trigonométriques : `\sin`, `\cos` et `\tan`. En rajoutant `arc` devant, vous aurez les réciproques, et `h` derrière vous obtiendrez les versions hyperboliques.
- les logarithmes népérien et décimal définis respectivement par les fonctions `\ln` et `\log`.
- les fonctions `\sup`, `\inf`, `\max`, `\min`, et `\arg` qui vous permettront de générer des formules de ce genre :

```
\begin{displaymath}
        T=\arg \max_{t<0} f(t)
\end{displaymath}
```

$$T = \arg\max_{t<0} f(t)$$

Notez l'utilisation de l'opérateur indice _ et le placement résultant avec la commande `\max`.

Intégrales, sommes et autres limites

LATEX utilise une syntaxe simple pour produire *intégrales*, *sommes*, etc. La syntaxe est la suivante : `\op_{inf}^{sup}`
où op est l'un des opérateurs `sum`, `prod`, `int` ou `lim` et `inf` et `sup` sont les bornes inférieure et supérieure de la somme ou de l'intégrale. Ainsi on peut donc écrire :

```
Somme des termes 'dune suite
géométrique :
\begin{displaymath}
\sum_{i=0}^{n}q^i=\frac{1-q^{n+1}}{1-q}
\end{displaymath}
```

Somme des termes d'une suite géométrique :

$$\sum_{i=0}^{n} q^i = \frac{1-q^{n+1}}{1-q}$$

Le produit $\prod$ s'utilise de manière analogue avec la commande \prod. Un exemple avec une intégrale, en veux-tu en voilà :

```
On définit le logarithme
népérien de $x>0$ comme suit :
\begin{displaymath}
\ln(x)=\int_{1}^{x}\frac{1}{t}\,\mathrm{dt}
\end{displaymath}
```

On définit le logarithme népérien de $x > 0$ comme suit :

$$\ln(x) = \int_1^x \frac{1}{t}\,\mathrm{dt}$$

La commande permet d'insérer un léger blanc avant le « dt ». Si vous êtes plutôt *curviligne*, vous pouvez utiliser \oint qui donne : $\oint$.On donne un exemple avant une limite :

```
$f(x)$ admet une limite $\ell$ en $x_0$:
\begin{displaymath}
\lim_{x\rightarrow x_0}f(x)=\ell
\end{displaymath}
```

$f(x)$ admet une limite ℓ en x_0 :

$$\lim_{x\to x_0} f(x) = \ell$$

J'espère que vous avez apprécié le beau ℓ; pour se fixer les idées sur les deux modes mathématiques, voici les mêmes formules mais incrustées dans le texte. Donc d'abord la sommation : $\sum_{i=0}^{n} q^i = \frac{1-q^{n+1}}{1-q}$, ensuite l'intégrale : $\int_1^x \frac{1}{t} dt = \ln(x)$, et enfin la limite : $\lim_{x\to x_0} f(x) = \ell$.

Des symboles les uns sur les autres

L'opérateur `not`

L'opérateur \not permet de produire la « négation » d'une relation :

`Soit $x \not\in I$ un réel...`	Soit $x \notin I$ un réel...

le résultat est donc un « slash » sur le symbole suivant. **Attention**, cet opérateur n'est pas très performant : `$\not\longrightarrow$` donne : $\not\longrightarrow$,mais est satisfaisant pour les symboles d'une largeur raisonnable.

Accents

Il est souvent utile d'accentuer les symboles en guise de notation particulière. Voici les accents disponibles :

`\hat{x}`	$\hat{x}$	`\check{x}`	$\check{x}$	`\breve{x}`	$\breve{x}$
`\acute{x}`	$\acute{x}$	`\grave{x}`	$\grave{x}$	`\tilde{x}`	$\tilde{x}$
`\bar{x}`	$\bar{x}$	`\dot{x}`	$\dot{x}$	`\ddot{x}`	$\ddot{x}$

Vecteurs

Il existe deux façons d'obtenir un vecteur :

- `\vec` pour les petits symboles car `\vec` est une commande d'accentuation
- `\overrightarrow` dans les autres cas.

`Soit $\overrightarrow{A\!B}$ défini dans la base $(\vec{\imath},\vec{\jmath})$`	Soit $\overrightarrow{AB}$ défini dans la base $(\vec{\imath},\vec{\jmath})$

Notez que `$\vec{A\!B}$` aurait donné : $\vec{AB}$ (voir aussi le paragraphe 1.5 pour la signification de la commande `\!`). Remarquez également les commandes `\imath` et `\jmath` qui fournissent les lettres « i » et « j » sans point : $\imath$ et $\jmath$.

Commande `stackrel`

La commande `\stackrel` permet de poser deux symboles l'un sur l'autre :
`\stackrel{symb 1}{symb 2}`
met le `symb1` sur `symb2` . Par exemple :
`x\stackrel{f}{\longmapsto}y`
donne : $x \stackrel{f}{\longmapsto} y$

Deux principes importants

Pour bien comprendre la manière dont LATEX génère les formules, il faut saisir les deux principes suivants :

Espaces : LATEX ignore les espaces entre les symboles mathématiques ; ainsi : `$x+1$` produira la même formule que `$x + 1$`. C'est LATEX qui insère les espaces à l'endroit qu'il juge le plus judicieux ;

Texte : tout groupe de symboles est considéré comme un groupe de variables ou fonctions ; ainsi `$x=t avec t>0$` produira $x = tavect > 0$ et non ce que vous espériez : $x = t$ avec $t > 0$.

Une fois ces deux principes acquis, voyons comment on peut faire avec.

Espaces en mode mathématique

Tout d'abord, sachez que LATEX fait un choix d'espacement qui est en général correct. Cependant le jour où vous aurez à jouer l'enculeur de mouche, les commandes du tableau 2.1 vous permettront d'insérer un ou des espaces dans des formules. Dans ce tableau, on montre l'effet des commandes d'espacement entre deux symboles □ .
Pour ce qui concerne les mouches, sachez que l'auteur de ce manuel a sournoisement inséré un certain nombre d'espacements au numérateur du calcul de

`\!` □□	$(rien)$ □□	`\,` □ □	`\:` □ □
`\;` □ □	`\␣` □ □	`\quad` □ □	`\qquad` □ □

TAB. 2.1 : Espacement en mode mathématique

la somme des termes de la suite géométrique (1.3 page 20), pour aligner les deux q de la fraction. Voici ce que donnait la formule par défaut :

$$\sum_{i=0}^{n} q^i = \frac{1-q^{n+1}}{1-q}$$

et voyons si les histoires de q vous donnent le sens de l'observation.

Texte en mode mathématique

Le moyen le plus simple d'insérer du texte dans une formule est de de le mettre « en boîte » et d'insérer quelques espaces :

```
Soient les suites $(u_n)$ et $(v_n)$ :
\begin{displaymath}
u_n=\ln n\quad
\mbox{et}\quad v_n=(1+\frac{1}{n})^n
\label{ex-maths-suite}
\end{displaymath}
```

Soient les suites (u_n) et (v_n) :

$$u_n = \ln n \quad \text{et} \quad v_n = (1 + \frac{1}{n})^n$$

L'environnement "Array"

L'environnement `array` est un environnement qui vous permettra de produire la grande majorité de vos formules. Comme son nom l'indique il range des objets en ligne et colonne. En fait c'est le pendant de l'environnement `tabular` du mode texte. Et comme `tabular`, `array` ne passe pas à la ligne.

Comment ça marche

La syntaxe rappelle celle de `tabular` :
`\begin{array}[vpos]{format} ... \end{array}`
où `format` précise pour chaque colonne l'alignement : `c` pour centré, `l` pour aligné à gauche et `r` pour aligné à droite ; l'argument optionnel `vpos` spécifie quant à lui le positionnement vertical du tableau. Comme dans les `tableaux`, on notera l'utilisation des commandes :

- `&` comme séparateur de colonne ;
- `\\` pour passer à la ligne.

```
Soit $A=\begin{array}{rc}
        -3 & 2 \\
        5 & 9
\end{array}$ la matrice ...
```

Soit $A = \begin{array}{rc} -3 & 2 \\ 5 & 9 \end{array}$ la matrice ...

Voici un exemple utilisant les points de suspensions :

```
\begin{displaymath}
A=\left[\begin{array}{ccc}
a_{00} & \dots & a_{0n}\\
\vdots & \ddots & \vdots\\
a_{n0} & \dots & a_{nn}
\end{array}\right]\end{displaymath}
```

$$A = \left[\begin{array}{ccc} a_{00} & \dots & a_{0n} \\ \vdots & \ddots & \vdots \\ a_{n0} & \dots & a_{nn} \end{array}\right]$$

Array et les délimiteurs

On utilise couramment l'environnement `array` pour produire des matrices. Il faut alors avoir recours à des délimiteurs. Ces *délimiteurs* sont de la famille des parenthèses et permettent d'englober un objet mathématique entre crochets, accolades, etc. La syntaxe est la suivante :
`\leftdelim1 mobjet \rightdelim2`
où `delim1` et `delim2` sont deux délimiteurs et `mobjet` un objet mathématique. Parmi les délimiteurs, voici les plus usités :

`(et)`	$(\prod)$	`[et]`	$[\prod]$
`\{et\}`	$\{\prod\}$	`\lfloor et \rfloor`	$\lfloor\prod\rfloor$
`\lceil et \rceil`	$\lceil\prod\rceil$	`\langle et \rangle`	$\langle\prod\rangle$
`|`	$\|\prod\|$	`\\|`	$\|\|\prod\|\|$

L'intérêt des délimiteurs est qu'ils s'adaptent automatiquement à la taille des objets qu'ils entourent :

```
soit $I=
\left[\begin{array}{cc}
        1&0 \\ 0&1
\end{array}\right]$
la matrice identité
```

soit $I = \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right]$ la matrice identité

On peut également reprendre l'exemple 1.5 page 23 avec des délimiteurs pour ajuster la taille des parenthèses :

```
Soient les suites $(u_n)$ et $(v_n)$ :
\begin{displaymath}
u_n=\ln n\quad\mbox{et}
\quad v_n=\left(1+\frac{1}{n}\right)^n
\end{displaymath}
```

Soient les suites (u_n) et (v_n) :

$$u_n = \ln n \quad \text{et} \quad v_n = \left(1 + \frac{1}{n}\right)^n$$

Il doit toujours y avoir une commande `\right` pour une commande `\left`. Cependant, il n'est pas nécessaire d'avoir les mêmes symboles à droite et à gauche. Voici un exemple où on utilise la commande `\right`. pour spécifier que l'on n'utilise pas de symbole à droite :

```
soit $ S_i=\left\{\begin{array}{rl}
-1 & \mbox{si $i$ est pair} \\
1 & \mbox{sinon.}\end{array}\right.$
```

soit $S_i = \begin{cases} -1 & \text{si } i \text{ est pair} \\ 1 & \text{sinon.} \end{cases}$

Pour vous simplifier la vie...

Le package `amsmath` permet de saisir plus simplement les matrices avec notamment deux environnements : `pmatrix` (p pour parenthèse) et `bmatrix` (b pour *bracket*).

```
\begin{displaymath}
\bar{\bar{\sigma}}=\begin{bmatrix}
\sigma_{11} & \sigma_{12} \\
\sigma_{21} & \sigma_{22} \\
\end{bmatrix}
\end{displaymath}
```

$$\bar{\bar{\sigma}} = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

Équations et environnements

Nous présenterons dans ce paragraphe trois environnements standard de LATEX permettant de produire des formules.

L'environnement `displaymath`

Vous l'avez compris, si vous avez lu jusqu'ici, `displaymath` affiche une formule centrée, interrompant le paragraphe. Un raccourci agréable de :

`\begin{displaymath}...\end{displaymath}`
est :`\[...\]`. Ainsi :

```
Distance colorimétrique :\[
\Delta E=\sqrt{
\Delta L^{*2}+ \Delta a^{*2}+
\Delta b^{*2}}
\]
```

Distance colorimétrique :

$$\Delta E = \sqrt{\Delta L^{*2} + \Delta a^{*2} + \Delta b^{*2}}$$

L'environnement `equation`

L'environnement `equation` est l'équivalent du précédent, sauf qu'il numérote la formule.

```
À retenir : si $a>0$ et $b>0$,
\begin{equation}
\ln(ab)=\ln(a)+\ln(b)
\end{equation}
```

À retenir : si $a > 0$ et $b > 0$,

$$\ln(ab) = \ln(a) + \ln(b) \qquad (2.1)$$

L'option de classe de document `leqno` met le numéro des équations à gauche. Et l'option `fleqn` aligne les équations à gauche, au lieu de les centrer.

Formules multi-lignes

Dans une précédente édition, nous finissions la présentation des environnements standard par l'environnement `eqnarray` qui permet de produire des formules de plusieurs lignes. Sachez que c'est mal. Il existe d'ailleurs des écrits à ce sujet, vous expliquant comment produire des documents « propres ». Prenez bien conscience qu'utiliser `eqnarray` (et bien d'autres choses) est un péché, et que si vous cédez malgré tout à la tentation, l'inquisition vous retrouvera un jour ou l'autre par l'intermédiaire d'un moteur de recherche. Aucune confession ou indulgence ne pourra vous sortir de ce mauvais pas, vous êtes prévenus. Nous vous présentons donc ici l'environnement `align` du `package amsmath` :

- `\\` passe à la ligne ;
- Chaque ligne est numérotée sauf si la commande `\nonumber` est présente dans la ligne ;
- On procède à l'alignement avec deux &.

```
\begin{align}
(a+b)^2 & = (a+b)(a+b)\nonumber\\
& = a^2+b^2+2ab
\end{align}
```

$$(a+b)^2 = (a+b)(a+b)$$
$$= a^2+b^2+2ab \quad (2.2)$$

Il existe une forme « étoilée » de l'environnement : `align*` où aucune des lignes n'est numérotée. Si vous voulez faire référence à certaines lignes d'un `align`, il vous faudra poser autant de `\label` nécessaires sur chaque ligne correspondante.
Pour faire numéroter une équation s'étalant sur plusieurs lignes on peut utiliser l'environnement `split` (lui aussi fourni avec `amsmath`) :

```
\begin{equation}
\begin{split}
(a+b)^2 & = (a+b)(a+b)\\
            & = a^2+b^2+2ab
\end{split}
\end{equation}
```

$$(a+b)^2 = (a+b)(a+b)$$
$$= a^2+b^2+2ab \quad (2.3)$$

Changer le style en mode mathématique

Fontes

LATEX fournit plusieurs commandes permettant de changer de fontes dans les modes mathématiques. Par défaut tout symbole ou suite de caractères (autre qu'une fonction) est produit en italique dans le document final. Or dans certains cas, il est utile de pouvoir forcer le style de fonte. Voici comment réaliser un tel exploit :

Soit	`$\mathit{A\in\prod}$`	soit	$\mathit{A\in\prod}$
Soit	`$\mathrm{A\in\prod}$`	soit	$\mathrm{A\in\prod}$
Soit	`$\mathbf{A\in\prod}$`	soit	$\mathbf{A\in\prod}$
Soit	`$\mathsf{A\in\prod}$`	soit	$\mathsf{A\in\prod}$
Soit	`$\mathtt{A\in\prod}$`	soit	$\mathtt{A\in\prod}$
Soit	`$\mathcal{A\in\prod}$`	soit	$\mathcal{A\in\prod}$

La commande `\mathcal` doit prendre exclusivement des lettres majuscules latines comme argument. Dans le cas contraire, les résultats seront farfelus

Par exemple, la séquence :
`\mathcal{abcd\Gamma}`
donne : ⊣⌊⌋⌈−

Taille des symboles

LATEX distingue quatre *styles* d'écriture des formules. Ces modes sont utilisés suivant la « situation » dans laquelle se trouve LATEX lorsqu'il produit une partie d'une formule :

texte pour une formule insérée dans le texte courant ;

equation pour une formule sous forme d'*équation* ;

indice pour l'écriture des indices ;

sous-indice pour les indices d'indices

chacun de ces modes peut être enclenché explicitement par l'utilisateur grâce aux déclarations suivantes :

- `\textstyle` pour le mode texte ;
- `\displaystyle` pour le mode équation ;
- `\scriptstyle` pour le mode indice ;
- `\scriptscriptstyle` pour le mode indice d'indice

Voici deux exemples illustrant comment forcer le mode *texte* en mode *équation* et inversement :

```
deux produits : $\prod_{1}^{n}f_i$
et $\displaystyle\prod_{1}^{n}f_i$
et inversement :
\[ \prod_{1}^{n}f_i
\mbox{et}\textstyle\prod_{1}^{n}f_i\]
```

deux produits : $\prod_1^n f_i$ et $\displaystyle\prod_1^n f_i$ et inversement :

$$\prod_1^n f_i \text{et} \prod_1^n f_i$$

Créer de nouveaux opérateurs

Imaginez que vous ayez besoin de créer un opérateur spécial nommé « burps ». Il suffira de procédé comme suit :

```
\newcommand{\burps}{%
\mathop{\textrm{burps}}}
$x=\burps_i f(i)$
```

$$x = \operatorname{burps}_i f(i)$$

Un autre exemple, pour franciser la fonction «*arcsinus*» (produisant par défaut $\arcsin$), on pourra écrire :

```
$\theta = \arcsin x$
\renewcommand{\arcsin}{%
\mathop{\textrm{Arcsin}}\nolimits}
$\theta = \arcsin x$
```

$\theta = \arcsin x$ $\theta = \operatorname{Arcsin} x$

La commande `\nolimits` indique que l'opérateur concerné ne fera pas usage d'arguments en indice ou exposant comme le font les opérateurs `\lim`, `\int`, etc. En outre les deux exemples précédents utilisent les commandes `\newcommand` et `\renewcommand` dont il est question au paragraphe 4.5 page 82. Enfin une autre voie possible si vous avez pris soin de charger le package `amsmath` est de déclarer dans le préambule :

```
\DeclareMathOperator*{\vlunch}{vlunch}
\DeclareMathOperator{\zirgl}{Zirgl}
```

```
\[x=\vlunch_i f(\theta)\]
où $\theta = \zirgl y$
```

$$x = \operatorname*{vlunch}_i f(\theta)$$

où $\theta = \operatorname{Zirgl} y$

Conclusion

Ce chapitre présente les fonctions de base pour produire des formules. Ces commandes suffisent pour la plupart des documents scientifiques. Si vous êtes amenés à rédiger des documents truffés de formules complexes, il est possible que les seules macros de LATEX ne suffisent plus. C'est pourquoi la célèbre *American Mathematical Society* a conçu pour vous un package nommé $\mathcal{AMSTEX}$ (mise en route : `\usepackage{amsmath}`) capable de générer des formules particulièrement « tordues. »

2.2 Graphiques dans votre document

Introduction

Un package graphique sur LaTeX est un ensemble d'outils et de fonctionnalités qui permettent de créer des graphiques, des diagrammes, des schémas et des illustrations dans un format professionnel. Ces packages offrent une grande flexibilité pour personnaliser les graphiques en termes de couleurs, de styles et de formes, ce qui permet une présentation claire et concise des données et des informations dans les documents scientifiques et techniques. Les packages graphiques les plus populaires sur LaTeX sont PGF/TikZ et PStricks.

L'utilisation de packages graphiques sur LaTeX est devenue de plus en plus populaire pour la création de documents scientifiques et techniques. Les packages graphiques offrent une variété d'outils pour créer des graphiques, des diagrammes, des schémas et des illustrations dans un format professionnel. Ils permettent également une personnalisation complète des graphiques en termes de couleurs, de styles et de formes.Pour apercevoir le package graphique, il est généralement nécessaire d'importer le package dans le document LaTeX et de créer un environnement spécifique pour le graphique. Ensuite, les utilisateurs peuvent utiliser des commandes spécifiques pour dessiner des formes, des lignes, des courbes et des textes, et pour définir les propriétés graphiques telles que la couleur, l'épaisseur de ligne et le style de remplissage. Une fois le graphique créé, il peut être compilé avec le document LaTeX pour produire une sortie PDF ou autre format. Les utilisateurs peuvent également ajuster les paramètres du graphique et expérimenter avec différentes options pour obtenir le résultat souhaité.

Parmi les packages graphiques les plus populaires sur LaTeX, on trouve PGF/TikZ, qui offre une

grande flexibilité pour la création de graphiques vectoriels en 2D et 3D, ainsi que PStricks, qui est un package graphique basé sur PostScript pour la création de graphiques en 2D.un package qui permet aux utilisateurs de dessiner des graphiques de haute qualité avec une grande flexibilité et un contrôle précis. Il offre une syntaxe intuitive et facile à apprendre pour dessiner des formes, des courbes, des lignes, des flèches, des textes et des images. Les utilisateurs peuvent également définir des styles pour les éléments graphiques, utiliser des transformations pour modifier l'apparence des graphiques, et ajouter des annotations pour améliorer la compréhension. TikZ est largement utilisé dans les domaines de la science, de l'ingénierie et des mathématiques pour créer des diagrammes, des schémas, des graphiques de fonctions et bien plus encore.

De nos jours, la plupart des documents contiennent des graphiques en plus du texte. S'il est facile d'ajouter des photos et des dessins, l'intégration de diagrammes et de schémas dans votre document peut s'avérer difficile.
Les polices, les couleurs et les lignes doivent être ajustées pour ne pas paraître déplacées.
De plus, des modifications ultérieures de la mise en page de votre document peuvent vous obliger à refaire ce travail. Heureusement, il est possible de créer vos graphiques directement dans LATEX. l'ajustement des paramètres mentionnés ci-dessus et les maintenir en harmonie avec le document lui-même.

Aperçu

Tout d'abord, un peu d'histoire sur la façon de réaliser des graphiques. En gros, il existe trois types d'images que vous pouvez utiliser pour traiter :

Les photos sont des images qui contiennent des ombres réalistes et de nombreux détails.
Il s'agit notamment de photos réelles et de captures d'écran de jeux vidéo.
Dans les photos, la couleur exacte des pixels n'est pas vraiment importante, et la compression avec perte peut être utilisée sans dégradation visuelle. Il est préférable de stocker les photos au format JPEG .

Les dessins sont des images avec des couleurs neutres et peu de détails. Cette catégorie comprend le pixel art et les captures d'écran d'interfaces de programmes.
Dans ce cas, la compression avec perte entraînerait une dégradation visible et ne serait pas très efficace.
Il est préférable d'utiliser un format de compression tel que PNG. Celui-ci garantit que chaque image est reproduite avec une précision de pixel, toute en maintenant la taille des fichiers à un niveau réduit.

Les diagrammes ou les graphiques sont des éléments graphiques simples contenant des textes,

des lignes et d'autres objets géométriques. Les logos et les diagrammes en sont de parfaits exemples. Idéalement, ils devraient être stockés sous forme d'instructions de dessin dans un format graphique à vecteurs tel que EPS, SVG ou PDF. Les graphiques vecteurs peuvent être mis à l'échelle dans n'importe quelle taille sans perte de qualité.
on a déjà appris comment inclure des photos et des dessins, dans la section ? on peux utiliser les mêmes techniques pour inclure des diagrammes, et c'est tout à fait possible. Certains programmes, comme "Inkscape" , permettent même d'inclure facilement des graphiques produits dans des documents LATEX. Cependant, ce chapitre se concentrera sur le dessin de diagrammes directement dans LATEX.

On a déjà appris comment inclure des photos et des dessins, dans la section, on peux utiliser les mêmes techniques pour inclure des diagrammes, et c'est tout à fait possible. Certains programmes, comme "Inkscape" , permettent même d'inclure facilement des graphiques produits dans des documents LATEX. Cependant, ce chapitre se concentrera sur le dessin de diagrammes directement dans LATEX.

Cela présente de nombreux avantages, tels que la structuration logique, formatage en texte brut et intégration transparente dans la mise en page du document.

La création de documents graphiques avec LATEX a une longue tradition. Elle a commencé avec l'environnement " `picture`", qui vous permet de créer des graphiques en plaçant de manière intelligente des éléments de texte dans le document.
Malheureusement, cet environnement n'est pas très performant. de nombreuses fonctions ont été développées au cours de ces dernières années, telque `metapost`, `asymptote` ou `xypic`..
Aujourd'hui, le package le plus utilisé est `pgf` (Portable Graphics Format) et son interface utilisateur `TikZ` .
Ce chapitre présente les concepts de base de l'écriture de graphiques dans `TikZ`. Des informations plus détaillées peuvent être trouvés dans la documentation "`pgf`".
`TikZ` est un langage puissant et polyvalent, mais il n'est pas toujours facile à utiliser pour des graphiques complexes.
De nombreux logiciels et bibliothèques sont basés sur `pgf` pour simplifier la création de diagrammes spécialisés. Il s'agit notamment de "`pgfplots`", qui permet de tracer des fonctions et des diagrammes commutatifs.
Généralement c'est une bonne idée de chercher dans le "`CTAN`" Avant d'écrire un nouveau code `TikZ`, recherchez un code qui résoudrait déjà votre problème.

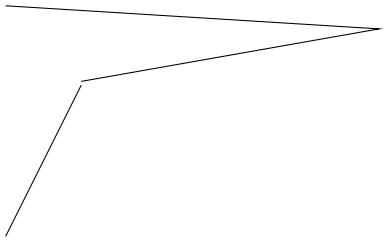
Usage basique

Pour utiliser `pgf` et `TikZ`, il suffit de placer la ligne suivante dans le préambule :

```
\usepackage{tikz}
```

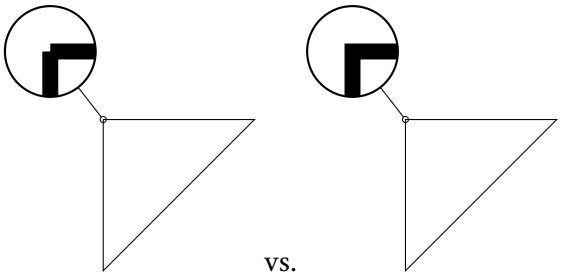
Cela permet d'obtenir l'environnement `tikzpicture` et la commande `\tikz` à l'intérieur desquels vous pouvez exécuter des commandes TikZ. Chaque commande TikZ se termine par un point-virgule (;). La commande la plus simple est la commande `\draw`, qui dessine des points reliés par un chemin. Les points peuvent être donnés en coordonnées cartésiennes (x, y) ou en coordonnées polaires $(\theta : r)$, où θ est spécifié en degrés, tandis que les distances sont spécifiées en centimètres par défaut. Le chemin entre les points peut être spécifié de plusieurs façons, la plus simple étant $--$, qui trace une ligne droite.

```
\begin{tikzpicture}
  \draw (0, 0)
    -- (10:4)
    -- (-1, 1);
\end{tikzpicture}
\tikz{\draw (0, 0) -- (1,2);}
```



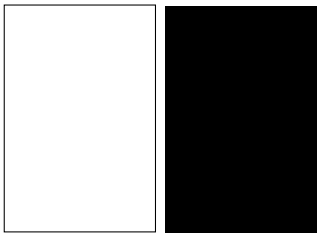
Si vous voulez fermer la forme que vous dessinez, utilisez `cycle` au lieu de répéter le premier point. Outre le fait que l'intention est plus claire, elle permet également d'obtenir une connexion de ligne plus esthétique (une jonction à onglet, comme on dit).

```
\begin{tikzpicture}
  \draw (0, 0) -- (2, 0) --
 (90: -2) -- (0, 0);
\end{tikzpicture}
vs.\
\begin{tikzpicture}
  \draw (0, 0) -- (2, 0) --
(90: -2) -- cycle;
\end{tikzpicture}
```



La commande `\draw` n'est qu'un raccourci de la commande plus générale `\path`. Cette dernière accepte un chemin de la même manière, mais une action doit être spécifiée entre crochets. La commande `\draw` fournit simplement l'action `draw`. L'action `fill` remplit la zone sous le chemin spécifié. Plusieurs actions peuvent être fournies.

```
\begin{tikzpicture}
\path[draw]
(0, 0)rectangle(2,3);
\end{tikzpicture}
\begin{tikzpicture}
\path[fill]
(0, 0)rectangle(2,3);
\end{tikzpicture}
```



Par défaut, les images de TikZ adaptent leur taille au rectangle minimal qui les entoure. Si vous souhaitez spécifier vous-même la zone de délimitation, utilisez l'action use as bounding box, ou la commande équivalente \useasboundingbox.

```
\tikz{\draw (0, 0) -- (1, 0);}
vs.\
\tikz{\draw (0, 1) -- (1, 1);}
vs.\
\tikz{
\useasboundingbox (0, 0)--
(0, 1)--(1,1)--(1,0)--cycle;
  \draw (0, 1) -- (1, 1);
}
```

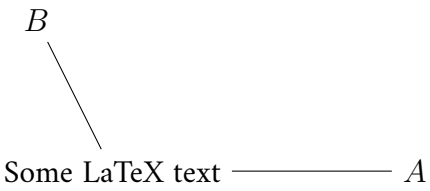
_____ vs. _____ vs.

Une entrée LaTeX normale peut être affichée à l'intérieur de ce que l'on appelle des nœuds. Pour créer un noeud, utilisez la commande

```
\node (<name>) at <coordinate> {<input>};
```

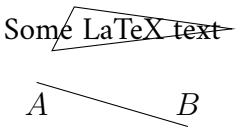
L'argument <name> est facultatif et vous permet d'utiliser la <name> comme raccourci pour les coordonnées d'un nœud.

```
\begin{tikzpicture}
  \node (text) at (0,0) {%
    Some LaTeX text};
  \node (A) at (4, 0) {\(A\)};
  \node (B) at (-1, 2) {\(B\)};
  \draw (A) -- (text) -- (B);
\end{tikzpicture}
```



TikZ essaie de positionner les lignes entre les nœuds de manière intelligente. Si vous souhaitez modifier cette position, vous pouvez spécifier le point de connexion exact sur un nœud après un point. Les points disponibles sont par exemple, `north`, `west`, `south-west`,... etc. Vous pouvez également spécifier un nombre, qui sera interprété comme un angle (en degrés).

```
\begin{tikzpicture}
  \node (T) at (1,1) {%
    Some LaTeX text};
  \node (A) at (0, 0) {\(A\)};
  \node (B) at (2, 0) {\(B\)};
  \draw (A.north) -- (B.south);
  \draw (T.0) -- (T.150)
    -- (T.200) -- cycle;
\end{tikzpicture}
```



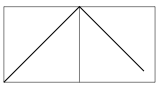
Des nœuds peuvent être créés dans les chemins en tapant `node` après une coordonnée ou une ligne donnée. Un nœud ajouté après une ligne est placé à son point médian.

```
\begin{tikzpicture}
  \draw (0, 0) node {Start}
    -- node {Midpoint}
    (4, 1) node {End};
\end{tikzpicture}
```

Si vous souhaitez créer un nœud vide dans le but de nommer un point spécifique, il est préférable d'utiliser la commande `\coordinate`. Cette commande garantit que le nœud est réellement vide,

alors que les nœuds créés par \node prennent quelques espace par défaut, même s'ils n'ont pas de contenu.

```
\begin{tikzpicture}
\draw[help lines](0,0)grid(2, 1);
\coordinate (A) at (0, 0);
\coordinate (B) at (1, 1);
\node (C) at (2, 0) {};
\draw (A) -- (B) -- (C);
\end{tikzpicture}
```



Courbes et formes

Jusqu'à présent, nous avons toujours utilisé -- pour relier les points. Cependant, ce n'est pas la seule façon de procéder. Par exemple, si vous souhaitez n'utiliser que des lignes horizontales et verticales, vous pourriez spécifier -| ou |- (selon l'ordre préféré) comme connexion entre les points.

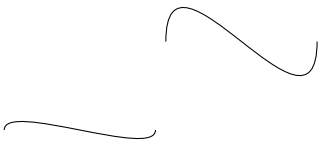
```
\tikz{\draw (0, 0) |- (2, 1);}
\tikz{\draw (0, 0) -| (2, 1);}
```

Si vous souhaitez créer des lignes courbes, la méthode la plus simple consiste à utiliser to entre les points. Il fonctionne de la même manière que --, mais vous pouvez fournir un argument facultatif, avec les clés in et out. facultatif, pour définir les angles terminaux de la connexion.

```
\tikz{\draw(0,0)to[out=90,in=-90](2,0);}

\tikz{\draw(0,0)to[out=45](2,0);}
```

La touche looseness peut être utilisée pour définir le degré d'allongement de la courbe.

```
\tikz{\draw(0,0)
to[out=90,in=-90,looseness=0.5](2,0);}

\tikz{\draw(0,0)
to[out=90,in=-90,looseness=2](2,0);}
```

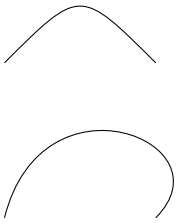


Il est souvent plus facile de spécifier des angles relatifs, en utilisant les touches `bend left` et `bend right`. Si aucun angle n'est fourni, une valeur par défaut est utilisée.

```
\tikz{\draw(0,0)to[bend left](2,0);}

\tikz{\draw(0,0)to[bend right=90](2,0);}
```

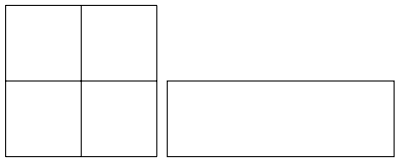
Si vous avez besoin d'un contrôle encore plus fin sur les courbes, vous pouvez utiliser une connexion `.. controls ..` pour spécifier une courbe de Bézier avec un ou deux points de contrôle.

```
\tikz{\draw(0,0)..controls(1,1)..(2,0);}

\tikz{\draw(0,0)..controls(.5,2)and(3,1)..(2,0);}
```



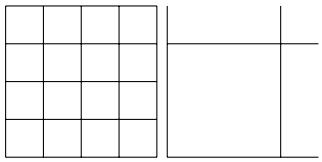
Outre les courbes, les points peuvent également être reliés à l'aide de différentes formes. Par exemple, `grid` dessine une grille entre les points, tandis que `rectangle` dessine un rectangle.

```
\tikz{\draw(0,0)grid(2,2);}

\tikz{\draw(0,0)rectangle(3,1);}
```



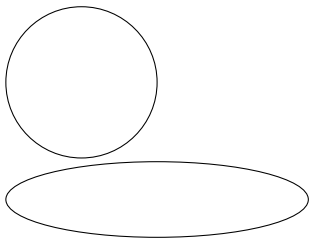
On utilisant le clé `step`,pour spécifier la finesse de la grille.

```
\tikz{\draw(0,0)grid[step=0.5](2,2);}

\tikz{\draw(0,0)grid[step=1.5](2,2);}
```



D'autres formes peuvent également être dessinées de cette manière, nécessitant parfois une syntaxe spécifique. Par exemple, `circle` interprète la coordonnée gauche comme son centre et reçoit son rayon via le clé `radius`. Vous pouvez également spécifier `x radius` et `y radius` séparément, ce qui permet de dessiner une ellipse.

```
\tikz{\draw(0,0)circle[radius=1];}

\tikz{
\draw(0,0)circle[x radius=2,y radius=0.5];}
```



De nombreuses autres formes sont disponibles, telles que `arc`, `parabola` ou `sin`. N'oubliez pas de consulter la documentation pour connaître les spécificités de leur utilisation.

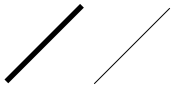
Personnalisation des chemins et des nœuds

Par défaut, tous les chemins sont dessinés avec une ligne noire continue. Vous pouvez les modifier en passant des options à la commande `\draw`. Par exemple, en passant un nom de couleur changera la couleur de la ligne.

```
\tikz{\draw[red](0,0)--(1,1);}

\tikz{\draw[blue](0,0)--(1,1);}
```

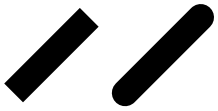
L'épaisseur d'une ligne peut être contrôlée en passant le clé `line width` (spécifiée en points par défaut), ou en utilisant l'une des valeurs prédéfinies, telles que comme `semithick`, `very thin` ou `ultra thick`.

```
\tikz{\draw[line width=2](0,0)--(1,1);}

\tikz{\draw[very thin](0,0)--(1,1);}
```



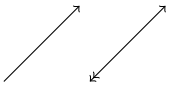
Les fins de lignes peuvent également être personnalisées. Par exemple, `line cap` vous permet de spécifier des arrondis.

```
\tikz{\draw[line width=10](0,0)--(1,1);}

\tikz{\draw[line width=10,line cap=round]
(0,0)--(1,1);}
```



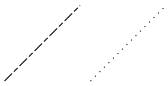
Vous pouvez transformer une ligne en flèche en spécifiant le clé `arrows`.

```
\tikz{\draw[arrows=->](0,0)--(1,1);}

\tikz{\draw[arrows=<<->](0,0)--(1,1);}
```



Les lignes ne doivent pas non plus être continues. Vous pouvez spécifier un motif `dash pattern` ou utiliser l'un des motifs prédéfinis.

```
\tikz{\draw[dash pattern=on 4 off 1 on 2 off 1]
(0,0)--(1,1);}
\tikz{\draw[dotted](0,0)--(1,1);}
```



Ajustez la façon dont les lignes sont connectées en utilisant `line join`. Réglez-le sur `round`, `bevel`, ou `miter`.

```
\tikz{\draw[line join=round]
  (0, 0) -- (0.5, 1) -- (1, 0);}
\tikz{\draw[line join=bevel]
  (0, 0) -- (0.5, 1) -- (1, 0);}
```

Si vous souhaitez que les jonctions de lignes soient arrondies, vous pouvez également passer le clé `rounded corners` avec la valeur fixée au rayon de l'arc.

```
\tikz{\draw[rounded corners]
  (0, 0) -- (0.5, 1) -- (1, 0);}
\tikz{\draw[rounded corners=25]
  (0, 0) -- (0.5, 1) -- (1, 0);}
```

Intéressons-nous maintenant aux nœuds. Par défaut, le contour d'un nœud n'est pas dessiné, mais vous pouvez passer `draw` et `fill`, éventuellement avec une couleur, pour la faire apparaître.

```
\tikz{\node[draw](0,0){Some Text};}

\tikz{\node[fill=red](0,0){Some Text};}
```

Some Text Some Text

Par défaut, tous les nœuds sont des rectangles, mais ils peuvent être changés en cercles en passant le clé cercle à leurs options.

```
\tikz{\node[draw,circle](0,0){Some text};}
```

Some text

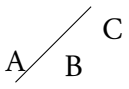
Si vous souhaitez placer du texte sur plusieurs lignes à l'intérieur d'un nœud, vous devez spécifier le clé `align`; sans elle, les nouvelles lignes sont ignorées. Les valeurs possibles sont `left`, `center` et `right`.

```
\tikz{\node[draw,align=left](0,0)
{Some more\\ text};}
\tikz{\node[draw,align=center](0,0)
{Even more \\ text};}
```

Some more text | Even more text

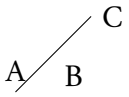
Lorsque les nœuds sont placés le long d'un chemin, leur position peut être ajustée à l'aide de le clé `anchor`. Sa valeur est le point de la frontière du nœud qui doit être ancré sur le point donné du chemin.

```
\tikz{\draw
(0,0)node[anchor=south]{A}
--node[anchor=north west]
{B}(1,1)node[anchor=135]{C};
}
```



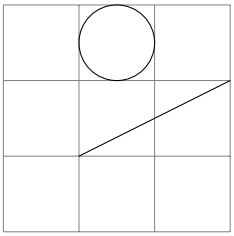
L'utilisation de commandes relatives, telles que `left` ou `above right`, à cette fin, conduit généralement à un code plus lisible, bien qu'elles ne soient pas aussi puissantes.

```
\tikz{\draw
  (0, 0) node[above] {A}
  -- node[below right] {B}
  (1, 1) node[right] {C};
}
```



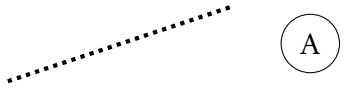
Toutes ces options peuvent être combinées librement, mais la spécification de style qui en résult peut s'avérer longue. Pour éviter de la retaper pour de nombreux nœuds ou chemins, vous pouvez définir de nouveaux styles. Certains styles prédéfinis existent déjà. Par exemple, le style `help lines` définit la couleur des lignes en gris et les rend un peu plus fines, ce qui est utile pour dessiner des grilles d'alignement plus fines, ce qui est utile pour dessiner des grilles d'alignement lors de la construction de vos propres images.

```
\begin{tikzpicture}
  \draw[help lines]
    (0, 0) grid (3,3);
  \draw (1, 1) -- (3,2);
  \draw (1.5, 2.5)
    circle[radius=0.5];
\end{tikzpicture}
```



Pour définir votre propre style, passez une clé de la forme name/.style=options à l'environnement Ti*k*Z ou aux options de la commande

```
\begin{tikzpicture}[
  my line/.style={dotted, ultra thick},
  my node/.style={draw, circle},
]
  \draw[my line] (0, 0) -- (3, 1);
  \node[my node] at (4, 0.5) {A};
\end{tikzpicture}
```

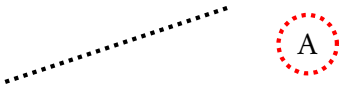


Si vous souhaitez configurer certains styles de manière globale, vous pouvez également utiliser la commande `\tkzset`

```
\tkzset{
  red style/.style={draw=red},
}
\tkz{\draw[red style]
  (0, 0) -- (1, 1) -- (2, 0);}
\tkz{\node[red style]
  at (0, 0) {Red node};}
```

Si vous souhaitez éviter de spécifier le style pour chaque nœud ou chemin dans une image Ti*k*Z, vous pouvez définir les styles spéciaux `every node` et `every path` pour les modifier tous en même temps.

```
\begin{tikzpicture}[
  every path/.style={
    ultra thick, dotted},
  every node/.style={
    circle, draw=red},
]
  \draw (0, 0) -- (3, 1);
  \node at (4, 0.5) {A};
\end{tikzpicture}
```



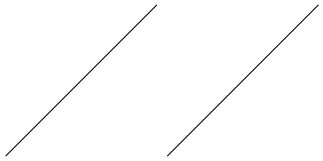
Coordonnées

Jusqu'à présent, nous toujours utilisé l'unité par defaut de centimètres pour spécifier les coordonnées. Cependant, il est possible d'utiliser n'importe quel dimension TeX.

```
\tikz{\draw (0,0)--(1in,1pt);}

\tikz{\draw (0,0)--(1dd,1em);}
```

Il est également possible de redimensionner la façon dont les distances sont mesurées en utilisant le clé `scale`. Ceci est utile si vous trouvez que votre image est trop grande ou trop petite après l'avoir dessinée, ou s'il existe un système de coordonnées intuitif (par exemple, lors du dessin de tracés de fonction).

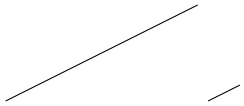
```
\tikz[scale=2]{\draw
(0, 0) -- (1, 1);}
\tikz{\draw (0, 0) -- (2, 2);}
```



Vous pouvez également spécifier `xscale` et `yscale` séparément.

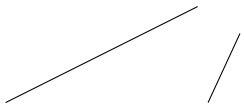
Lors de la spécification de léchelle, vous pouvez utiliser des opérations arithmétiques simples. Par exemple, pour changer les valeurs sans dimension en pouces, vous pouvez passer `scale=1in/1cm`

```
\tikz[scale=1in/1cm]{\draw
(0, 0) -- (1, 0.5);}
\tikz[scale=1em/1cm]{\draw
(0, 0) -- (1, 0.5);}
```



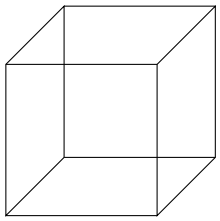
Gardez à l'esprit que les coordonnées avec les cotes seront également mises à l'échelle, ce qui peut avoir des conséquences imprévues. Une façon plus robuste de changer les valeurs sans dimension est les clés `x` et `y`.

```
\tikz[x=1in, y=1in]{\draw
(0, 0) -- (1, 0.5);}
\tikz[x=1em, y=10ex]{\draw
(0, 0) -- (1, 0.5);}
```



Vous pouvez en fait spécifier des coordonnées tridimensionnelles lorsque vous dessinez des images. Par défaut, la troisième coordonnée est interprétée comme un vecteur montrer du doigt 45 degrés vers le bas à gauche et est un peu plus court. Cela donne l'effet d'une projection paralléle (ou axonométrique).

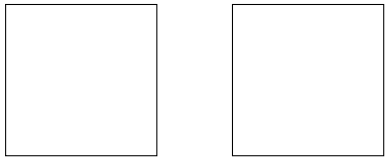
```
\begin{tikzpicture}[scale=2]
\draw (0,0,0)--(0,0,1)--(0,1,1)--
(0, 1, 0) -- cycle;
\draw (1,0,0)--(1,0,1)--(1,1,1)--
(1, 1, 0) -- cycle;
\draw(0,0,0)--(1,0,0)(0,0,1)--(1,0,1)
(0,1,1)--(1,1,1)(1,1,0)--(0,1,0);
\end{tikzpicture}
```



La longueur et la direction peuvent être modifiées à l'aide du clé z.

Parfois, il peut être plus facile de spécifier un chemin en utilisant des coodonnées relatives au point précédent plutôt que des coordonnées absolues. Pour ce faire, faites précéder ++ à la coordonnée, qui sera interprétée comme le point spécifié précédent plus ce vecteur.

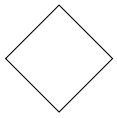
```
\begin{tikzpicture}[scale=2]
\draw (0, 0) -- ++(0, 1) --
++(1, 0) -- ++(0, -1) --
cycle;
\draw (1.5, 0) -- ++(0, 1) --
++(1, 0) -- ++(0, -1) --
cycle;
\end{tikzpicture}
```



La mise à l'échelle n'est pas a seule transformation pouvant être appliquée aux points. Il est

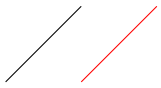
également possible de faire pivoter, décaler ou appliquer une transformation linéaire arbitraire en spécifiant sa matrice. Consulter la documentation pour une description détaillée.

```
\begin{tikzpicture}[rotate=45]
\draw (0, 0) -- (0, 1) --
(1, 1) -- (1, 0) -- cycle;
\end{tikzpicture}
```



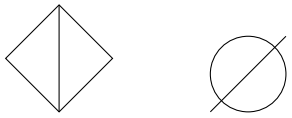
Il est également possible d'appliquer des transformation à une seule commande.

```
\begin{tikzpicture}
\draw (0, 0) -- (1, 1);
\draw[red, xshift=1cm]
(0, 0) -- (1, 1);
\end{tikzpicture}
```



Si vous souhaitez appliquer la même transformation à plusieurs commandes dans la même image, vous pouvez utiliser l'environement `scope`. C'est particulièrement utile lorsque votre image se compose de plusieurs sous-images, chacune étant plus ou moins indépendante.

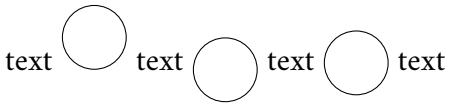
```
\begin{tikzpicture}
\begin{scope}[rotate=45]
\draw (0, 0) rectangle (1, 1);
\draw (0, 0) -- (1, 1);
\end{scope}
\begin{scope}[xshift=2cm]
\draw (0.5, 0.5) circle [radius=0.5];
\draw (0, 0) -- (1, 1);
\end{scope}
\end{tikzpicture}
```



Quand Ti*k*Z image sont positionées dans le texte, leur extrémité inférieure repose sur la ligne de base du texte. Si vous voulez le modifier, vous pouvez utiliser le clé `baseline` pour régler le coordonnée à laquelle la ligne de base doit être. Si aucune position 'est spécifiée, la valeur par défaut

est 0.

```
text \tikz{\draw(0,0)circle
[radius=1em];}
text \tikz[baseline]{\draw
(0, 0) circle [radius=1em];}
text \tikz[baseline=-0.5ex]{\draw
(0, 0) circle [radius=1em];} text
```

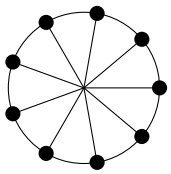


Réutilisation des images

Parfois, vous souhaitez peut-être dessiner la même image à quelques endroits dans un plus grand. Cela serait problématique, car modifier leur emplacement ou leur style n'est pas trivial. Ti*k*Z est livré avec sa propre méthode de définition d'images plus petites, appelées 'pics'. Ils peuvent être créés en passant `<name>/.pic=<commands>` au Ti*k*Zenvironnement ou la commande et sont ensuite utilisés en appelant la commande `\pic`.

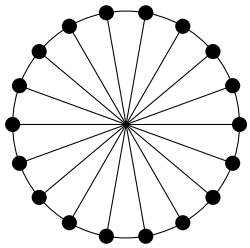
Si vous vous retrouver à répéter de nombreuses commandes simples (par exemple, tracer des graduations sur un axe), vous pouvezsimplifier votre code en utilisant la commnde `\foreach`. Il répète la commande de dessin pour chaque valeur d'une liste, de sorte que vous puissiez écrire du code répétable une fois et n'avoir besoin de modifier que les parties importantes.

```
\begin{tikzpicture}
\draw (0,0) circle[radius=1cm];
\foreach\i in{0, 40, 80, 120, 160, 200,
240,280,320}{
\draw (0, 0) -- (\i: 1);
\fill (\i: 1) circle[radius=0.1cm];
}
\end{tikzpicture}
```

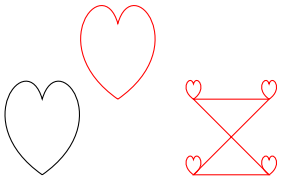


Si vous valeurs sont une séquence arithmétique simple, vous n'avez qu'à fournir les deux première valeurs et la dernière, en remplaçant le reste par des points triples.

```
\begin{tikzpicture}
\draw (0, 0) circle[radius=1.5cm];
\foreach \i in {0, 20, ..., 340} {
\draw (0, 0) -- (\i: 1.5);
\fill (\i: 1.5) circle[radius=0.1cm];
}
\end{tikzpicture}
```

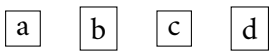


```
\begin{tikzpicture}[
heart/.pic={
\draw (0,0) .. controls (-1, 0.7)
 and (-0.2, 1.7)
.. (0, 1) .. controls
(0.2, 1.7) and (1, 0.7)
.. (0, 0);
},
]
\pic at (0, 0) {heart};
\pic[red] at (1, 1) {heart};
\begin{scope}[xshift=2cm,
every pic/.style={scale=0.2}]
\draw[red] (0, 0) pic {heart} -- (1, 1)
 pic {heart} --
(0, 1) pic {heart} -- (1, 0)
pic {heart} -- cycle;
\end{scope}
\end{tikzpicture}
```



Vous pouvez également itérer sur des paires en séparant les parties respectives avec /.

```
\begin{tikzpicture}
\foreach \i/\j in {0/a,1/b,2/c,3/d}{
\node[draw] at (\i, 0) {\j};
}
\end{tikzpicture}
```



Bibliothèques

pgf et Ti*k*Zne comptez pas sur LaTeX. En fait, ils peuvent être utilisés avec n'importe quel système basé sur TeXou même TeXlui-même. Pour cette raison, Ti*k*Zfournit son propre système d'extensions qui n'utilise pas la commande `\usepackage` de LaTeX. qui reçoit une liste de bibliothèques séparées par des virgules.

Par exemple, si vous avez l'intention de dessiner des problèmes de géométrie, vous vous retrouver souvent à chercher des intesections d'objets. Bien que vouspuissiez calculer leurs coordonnées exactes à la main, le bibliothèque `intersections` le fera pour vous. Un exemple est présenté dans liste 6.2.

D'autres bibliothèqyes étendent le nombre de formes disponibles. Par exemple, le bibliothèque `arrows.meta` définit de nombreuses pointes de flèches supplémentaires, si vous n'aimez pas la classique. Certains sont présentés dans liste 6.3.

Vous pouvez effectuer des calculs supplémentaires sur les coordonnées existantes. le bibliothèqye `calc` vous permet de le faire en les entourant de symboles $. Un exemple est présenté dans liste 6.4.

Si vous avez beaucoup de code de Ti*k*Zdans votre document, vous remarquerez peut-être que sa compilation prend beaucoup plus de temps. C'est parce que Ti*k*Zredessine chaque image avec chaque pass de LaTeX. Si cela devient ennuyeux, vous pouvez mettre vos images en cache dans des fichiers externes et les réutiliser lors d'exécutions ultérieures. Pour cela, mettez simplement le code suivant dans votre préambule.

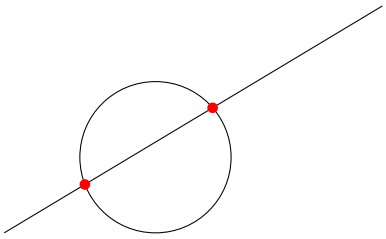
```
\usetikzlibrary{external}
\tikzexternalize
```

Cette méthode a certains limites, mais devrait être suffisante pour la plupart des utilisations. Lisez-le dans la documentation si des problèmes surviennent.

ce ne sont pas les seules bibliothèques--- des formes de nœuds supplémentaires, une perspective 3D réelle, des cartes mentales et bien d'autres sont couverts. La plupart d'entre euxsont décrit dans

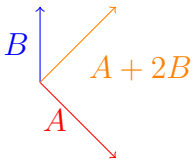
le documentation du paquet pgf. Vérifiez-le si vous n'avez pas trouvé de solution à votre problème ici.

```
% In preamble
\usetikzlibrary{intersections}
% ...
\begin{tikzpicture}
\draw[name path=O](0,0)circle[radius=1];
\draw[name path=L](-2, -1) -- (3, 2);
\fill
[red,name intersections={of=O and L}]
(intersection-1) circle[radius=2pt]
(intersection-2) circle[radius=2pt];
\end{tikzpicture}
```



```
% In preamble
\usetikzlibrary{arrows.meta}
% ...
\tikz{\draw[->](0,0)--(1,1);}
\tikz{\draw[-{Circle}](0,0)--(1,1);}
\tikz{\draw[-{Stealth}] (0, 0)
 -- (1, 1);}
\tikz{\draw[-{Stealth[round]}]
 (0, 0) -- (1, 1);}
\tikz{\draw[-{Diamond[open]}]
 (0, 0) -- (1, 1);}
```

```
% In preamble
\usetikzlibrary{calc}
% ...
\begin{tikzpicture}
\coordinate (A) at (1, -1);
\coordinate (B) at (0, 1);
\draw[red, ->] (0, 0) -- node[left] {\(A\)} (A);
\draw[blue, ->] (0, 0) -- node[left] {\(B\)} (B);
\draw[orange, ->] (0, 0) --
node[below right] {\(A+2B\)} ($(A)+2*(B)$);
\end{tikzpicture}
```



2.3 Représentation de courbe

LaTeX offre une multitude de possibilités pour concevoir des documents. Dans cet article, je vais montrer qu'il est rapidement possible d'effectuer le tracé d'une courbe (et même plus) sans avoir recours à un logiciel externe tel que TeXGraph, ou PstPlus et ceci simplement en ajoutant quelques lignes de code.

Utilisation

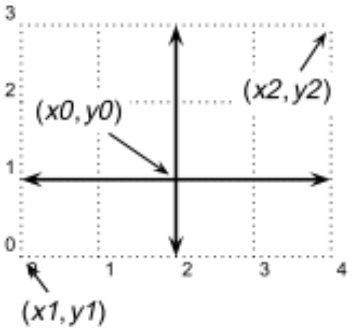
Dans le préambule de votre document LaTeX, il faudra appeler les extensions suivantes :

- pstricks-add, pour le tracé des courbes :
  `\usepackage{pstricks-add};`
- pst-eucl, pour la gestion des intersections des courbes :
  `\usepackage{pst-eucl}`.

Gestion du repère

Tout d'abord, il faudra gérer l'affichage du repère :

- le tracé du repère se fera à l'aide de la commande suivante :
  `\psaxes[par]{arrows}(x0,y0)(x1,y1)(x2,y2) ;`



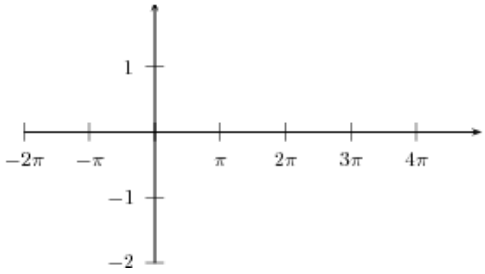
- il est conseillé de mettre les mêmes dimensions pour les axes que les dimensions de l'image.

```
\begin{pspicture}(-2,-2)(5,5)
\psgrid[subgriddiv=0,gridlabels=5pt,
gridwidth=0.5pt,
griddots=5](-2,-2)(5,5)
\psaxes[labels=none]{->}(0,0)(-2,-2)(5,5)
 \end{pspicture}
```

Gestion des labels sur les axes

Il est possible de modifier les unités sur les axes de façon à avoir sur l'axe des abscisses des unités trigonométriques :

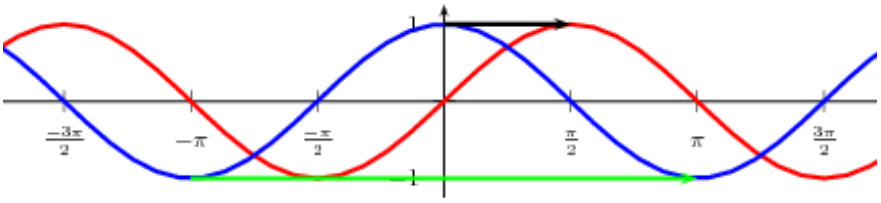
- pour cela, Il faudra ajouter le code suivant : `trigLabels=true` ;




```
\begin{pspicture}(-2,-2)(5,2)
\psset{trigLabels=true,labelFontSize=\small}
\psaxes{->}(0,0)(-2,-2)(5,2)
\end{pspicture}
```

- ainsi, avec un repère de ce type, on pourra tracer des fonctions trigonométriques ;

- il faudra cependant modifier la gestion des unités sur l'axe des abscisses en consultant la documentation de `pstricks-add` ;




```
\begin{pspicture}(-7,-1.25)(7,1.25)
\psset{algebraic=true}
\psaxes[trigLabels=true,trigLabelBase=2,dx=\psPiH,
xunit=\psPi]{->}(0,0)(-2.2,-1.25)(2.2,1.25)
\psplot[linecolor=red,linewidth=1.5pt]{-7}{7}{sin(x)}
\psplot[linecolor=blue,linewidth=1.5pt]{-7}{7}{cos(x)}
\end{pspicture}
```

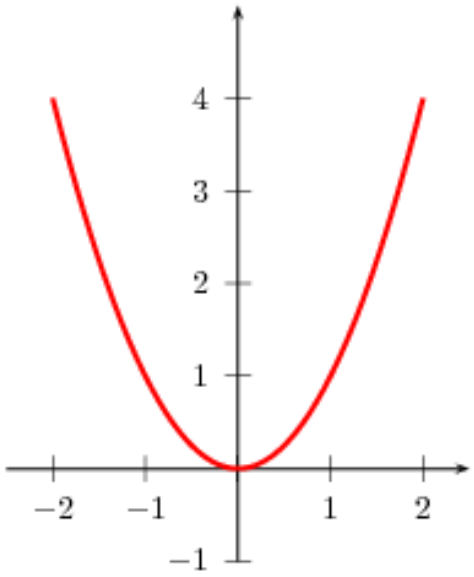
Représentation de fonctions

Courbes classiques

Pour tracer une courbe, la commande de base sera la suivante : `\psplot[par]{xmin}{xmax}{fonction}`.

- Définir une fonction
  - L'instruction `algebraic=true`, permettra de donner la fonction sous sa forme classique, sinon il faudra donner l'expression de la fonction en utilisant la notation polonaise inverse RPN (Reverse Polish Notation), notation souvent utilisée avant les années 90 notamment par les calculatrices HP.
  - Par exemple, la fonction $f : x^2 \mapsto x^2 + 3x - 5$ s'écrit en notation RPN :
    `x 2 exp 3 x mul add 5 sub`
    L'explication vient du fait qu'un opérateur a besoin de deux opérandes.
    Ainsi le code, `x2 exp` se traduit par x^2 le code `3x mul` par $3x$ et le code add qui suit signifie qu'il faut additionner ces deux termes. On obtient ainsi $x^2 + 3x$.Enfin le code qui suit, `5 sub`, signifie qu'il faut enlever 5 à ce que l'on vient d'obtenir. On obtient finalement : $x^2 + 3x - 5$.
- Tracer une courbe

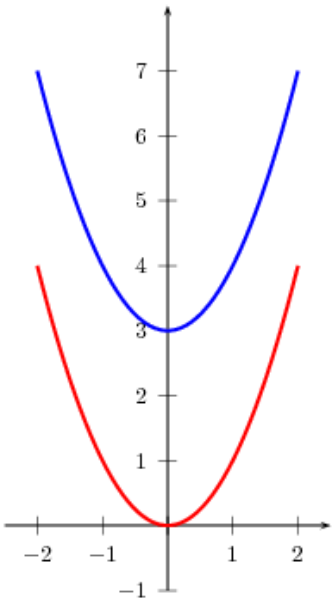
- On pourra d'une part donner directement l'expression de la fonction dans la commande `\psplot`.




```
\begin{pspicture}(-2.5,-1)(2.5,5)
\psset{xunit=1 cm, algebraic=true}
\psaxes{->}(0,0)(-2.5,-1)(2.5,5)
\psplot[linecolor=red,linewidth=1.5pt]{-2}{2}{x^2}
\end{pspicture}
```

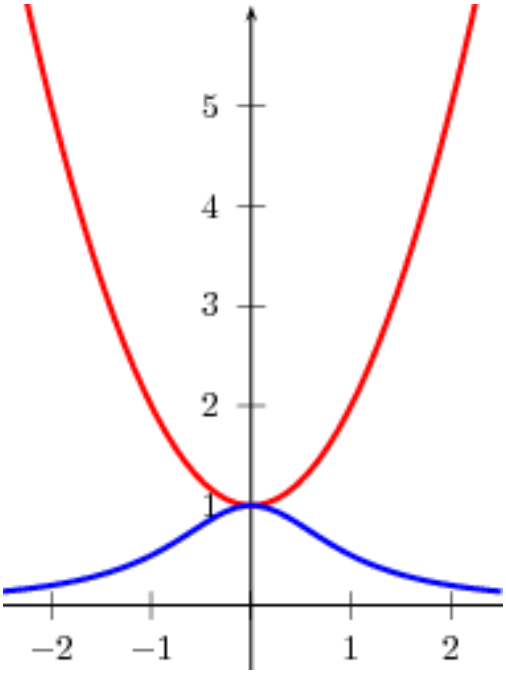
- On pourra aussi définir la fonction par la commande `\def\nom_de_la_fonction{expression}`, puis dans la commande `\psplot`, on appellera la fonction.
  L'avantage de cette méthode est que l'on peut définir une fonction g à partir d'une fonction f.

- L'exemple suivant montre la courbe d'une fonction et l'image de cette courbe par une translation.




```
\begin{pspicture}(-2.5,-1)(2.5,8)
\psset{xunit=1 cm,algebraic=true}
\def\f{x*x}
\def\g{\f+3}
\psaxes{->}(0,0)(-2.5,-1)(2.5,8)
\end{pspicture}
```

- Cet autre exemple montre la représentation graphique d'une fonction f et la représentation graphique de l'inverse de la fonction $\frac{1}{f}$.

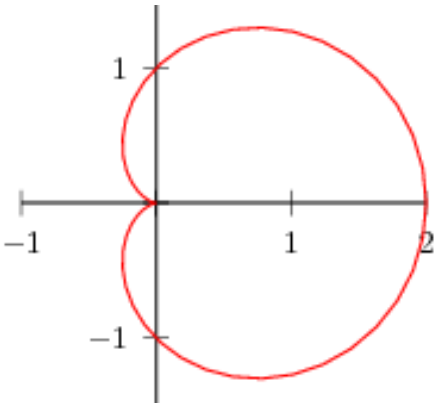


```
\begin{pspicture}(-3,-1)(3,6)
\psset{xunit=1 cm,algebraic=true}
\def\f{x*x+1}
\def\g{1/(\f)}
\psaxes{->}(0,0)(-3,-1)(3,6)
\end{pspicture}
```

Courbes paramétrique

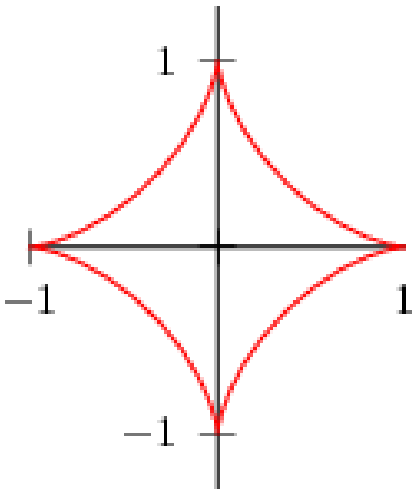
Pour tracer une courbe, la commande de base sera `\parametricplot[par]{tmin}{tmax}{x(t)y(t)}` qui utilise la notation RPN (le retour à la ligne permet de séparer les expressions $x(t)$ et $y(t)$).

- On peut voir ci-dessous un premier exemple.




```
\begin{pspicture}(-1,-2)(2,2)
\psset{xunit=1.5cm,yunit=1.5cm}
\psaxes(0,0)(-1,-2)(2,2)
\parametricplot[linecolor=red,linewidth=0.7pt]{0}{360}{
                                          %x(t)
                                  }
\end{pspicture}
```

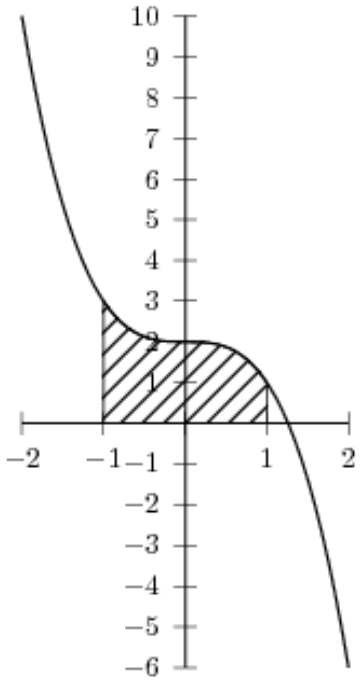
- Si on utilise l'instriction `algebraic=true` et la commande `\parametricplot[par]{tmin}{tmax}{x(t) y(t)}`, la variable t sera alors exprimée en radians.
  L'exemple ci-dessous en est une illustration.



Aires

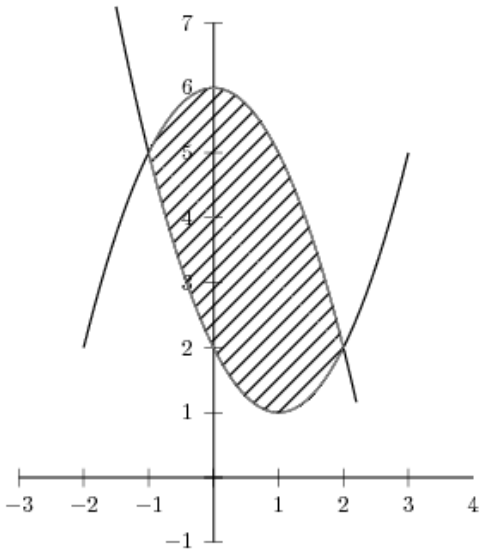
Il est possible de représenter l'aire comprise entre la courbe d'une fonction, l'axe des abscisses et les deux droites d'équations $x = a$ et $x = b$.
Pour cela, il s'agit d'utiliser la commande
`\pscustom[par]{segment(a,0)(a,f(a)) courbe segment(b,f(b))(b,0)}`
On fera attention au dernier segment tracé puisque la première extrémité du segment part de la courbe.

- L'exemple ci-dessous montre l'utilisation de ce code.



```
\psset{xunit=1cm, yunit=0.5cm}
\begin{pspicture}(-2,-6)(2,10)
\psaxes(0,0)(-2,-6)(2,10)
\def\f{2 x 3 exp sub}
\psplot{-2}{2}{\f}
\end{pspicture}
```

- On pourra aussi représenter l'aire comprise entre deux courbes.
  Pour cela, il faudra utiliser la commande`\pscustom[par]{courbe du bas courbe du haut}`




```
\psset{xunit=1cm, yunit=1cm,algebraic=true}
\begin{pspicture}(-3,-1)(4,7)
\psaxes(0,0)(-3,-1)(4,7)
\def\f{x*x-2*x+2}
\def\g{-x*x+6}
\end{pspicture}
```

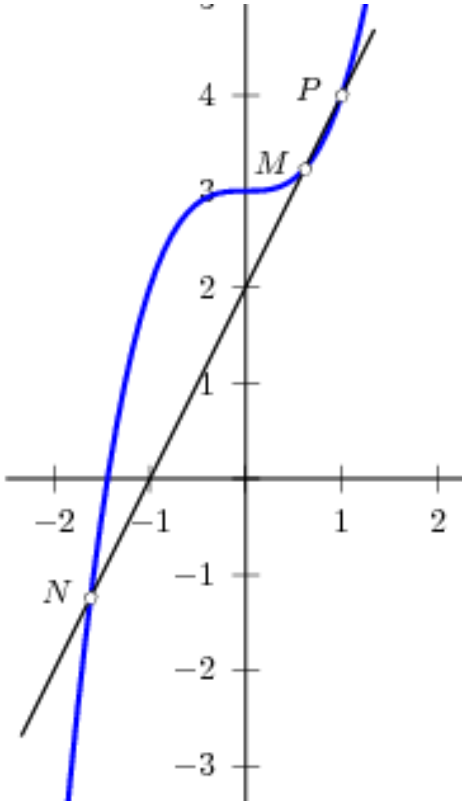
Intersection de deux fonctions

Pour obtenir les points d'intersections de deux courbes, il est nécessaire de déclarer dans le préambule du document, le package `pst-eucl` par le code
`\usepackage{pst-eucl}`.

C'est la seule partie où la notation RPN est nécessaire.

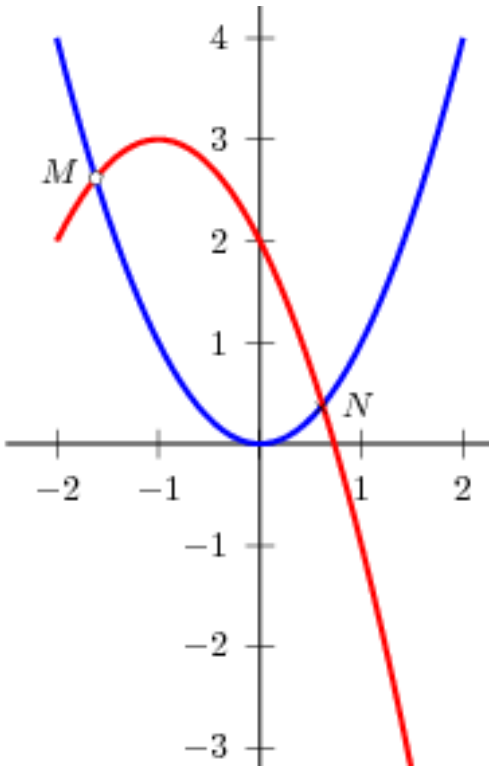
De plus, il est possible que la recherche de l'intersection ne puisse aboutir puisque celle-ci utilise l'algorithme de NEWTON.

- Pour obtenir l'intersection entre une courbe et une droite (AB), on utilisera le code `\pstInterFL[par]{fonction}{A}{B}{abscisse}{nom_du_point}`.
  L'abscisse donnée n'est pas forcément l'abscisse précise du point d'intersection mais juste un "positionnement".




```
\begin{pspicture}(-2.5,-5)(2.5,8)
\psset{xunit=1 cm}
\def\f{x 3 exp 3 add}
\psaxes{->}(0,0)(-2.5,-5)(2.5,8)
\psplot[linecolor=blue,linewidth=1.5pt]{-2}{2}{\f}
\end{pspicture}
```

- Pour obtenir l'intersection entre deux courbes, on utilisera le code `\pstInterFF[par]{fonction_1}{fonction_2}`
  L'abscisse donnée n'est pas forcément l'abscisse précise du point d'intersection mais juste un "positionnement".




```
\begin{pspicture}(-2.5,-5)(2.5,8)
\psset{xunit=1 cm}
\def\f{x 2 exp}
\def\g{3 x 1 add 2 exp sub}
\psaxes{->}(0,0)(-2.5,-5)(2.5,8)
\end{pspicture}
```

Tableau de relation entre la notation RPN et la notation classique.

Nom	Syntaxe	notation classique
add	x y add	$x + y$
sub	x y sub	$x - y$
mul	x y mul	x×y
exp	x y exp	x^y
div	x y div	$\frac{x}{y}$
neg	x y neg	$f(x) = x^2$
sqrt	x sqrt	$\sqrt{x}$
abs	x abs	$\|x\|$
cos	x cos	$\cos(x)$
sin	x sin	$\sin(x)$
tan	x tan	$\tan(x)$
ln	x ln	$\ln(x)$

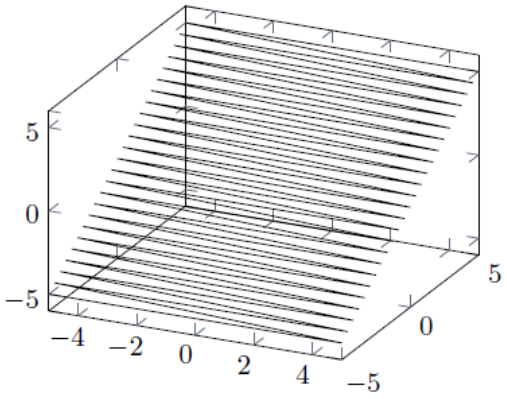
2.4 Graphiques en 3D plots

Pour tracer des courbes dans l'espace en trois dimensions, l'utilisation de cette instruction est similaire à celle de `\addplot` spécifique au tracé de courbes en deux dimensions. L'instruction `\addplot` 3 permet le tracé en trois dimensions.

Il existe de nombreuses propriétés qui permettent de contrôler la couleur des surfaces et la forme des lignes dans ces graphiques en 3D, nous mentionnerons donc quelques-unes d'entre elles. Pour le reste, nous appliquerons les mêmes propriétés (mots-clés) que celles que nous avons abordées précédemment pour les graphiques en 2D.

Axe : référence à l'axe.

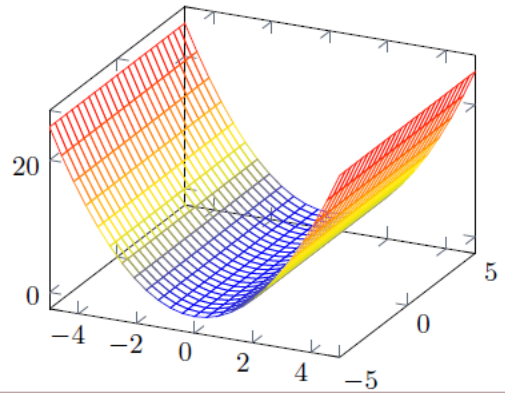
```
\begin{tikzpicture}
\begin{axis}
\addplot3[]{y};
\end{axis}
\end{tikzpicture}
```



2.5 Tracé de grilles de maillage

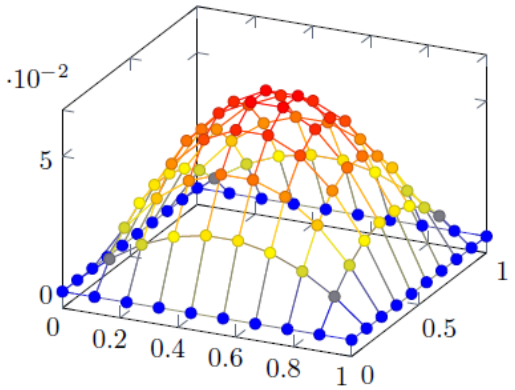
Le résultat ressemble à une grille (x, y, z). Cette grille est créée en reliant les points adjacents par des lignes droites ayant des coordonnées mesh. Les nœuds sont placés aux points de données et cela est effectué à l'aide de la commande "`mesh`".

```
\begin{tikzpicture}
\begin{axis}
\addplot3[mesh]{x^2};
\end{axis}
\end{tikzpicture}
```



Les points interconnectés peuvent être affichés avec plus de précision en ajoutant une grille (mesh) lors du processus de traçage à l'aide de la commande 'scatter'.

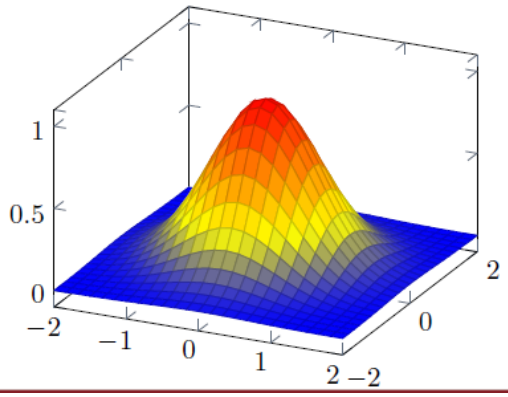
```
% Preamble: \pgfplotsset{width=7cm,compat
    =1.12}
\begin{tikzpicture}
\begin{axis}
\addplot3 [mesh,scatter,samples=10,domain
    =0:1]
{x*(1-x)*y*(1-y)};
\end{axis}
\end{tikzpicture}
```



2.6 Dessiner des surfaces

Dessiner des surfaces remplies et colorées, où les 'patches' dessinent les surfaces de manière similaire au traçage de maillages, à l'exception des espaces entre les lignes, appelés les 'patchs'. Les graphiques de ce type sont générés à l'aide de l'instruction 'surf', où la couleur varie en fonction de l'axe z.

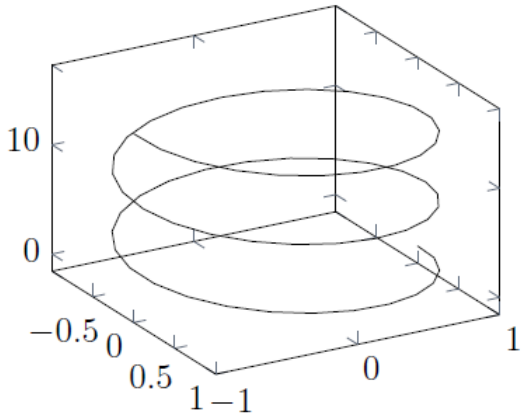
```
% Preamble: \pgfplotsset{width=7cm,compat
    =1.12}
\begin{tikzpicture}
\begin{axis}
\addplot3[surf,domain=-2:2] {exp(-x^2-y
    ^2)};
\end{axis}
\end{tikzpicture}
```



2.7 La représentation paramétrique

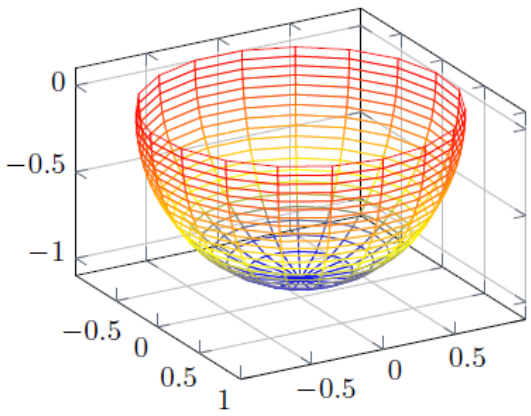
Nous pouvons tracer une fonction paramétrique en trois dimensions selon le style de graphique que nous souhaitons mesh,surf, La méthode de traçage n'est pas différente de celle de mesh et surf. Nous pouvons tracer une fonction paramétrique en trois dimensions selon le style de graphique que nous souhaitons en utilisant la commande `\addplot3[option]`(α, β, γ) avec des options spécifiques. Cependant, cela nécessite une méthode spéciale pour définir les coordonnées.
Quelques exemples illustrant cela :

```
\begin{tikzpicture}
\begin{axis}[view={60}{30},]
\addplot3[domain=0:5*pi,samples = 60,samples y=0,]
({sin(deg(x))},{cos(deg(x))},{x});
\end{axis}
\end{tikzpicture}
```



Par exemple, à l'intérieur de la représentation paramétrique, chaque variable est associée à son domaine de définition, y et x. Nous pouvons utiliser deux variables pour représenter la moitié d'une sphère, dont la représentation paramétrique est donnée par
:$(\sqrt{1-z^2}\cos\theta, \sqrt{1-z^2}\sin\theta, z)$; $z \in [-1;0]; \theta \in [0;2\pi]$
Ici, il n'est pas possible de placer une variable unique comme dans l'exemple précédent, donc nous définissons deux variables x et y

```
\begin{tikzpicture}
\begin{axis}[view={60}{30},grid=major]
\addplot3[mesh,z buffer=sort,
samples=20,domain=−1:0,y domain=0:2*
    pi]
({sqrt(1−x^2) * cos(deg(y))},
{sqrt( 1−x^2 ) * sin(deg(y))},
x);
\end{axis}
\end{tikzpicture}
```



2.8 Utiliser Matlab pour dessiner dans LaTeX

Les dessins précédents peuvent être créés à l'aide du programme Matlab puis convertis en un fichier avec l'extension TEX. Ensuite, nous le rappelons et l'utilisons directement dans le texte. Dans

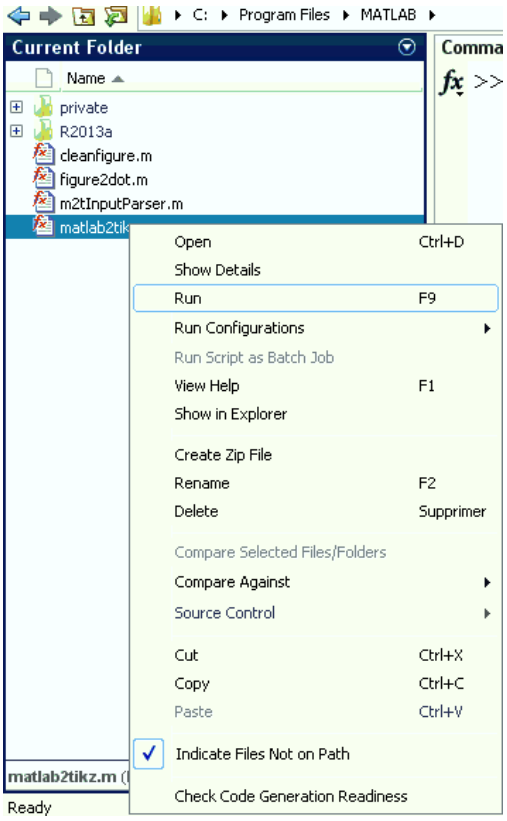
cet exemple simple, nous montrons le processus complet. - Nous ouvrons Matlab et dessinons la sphère :

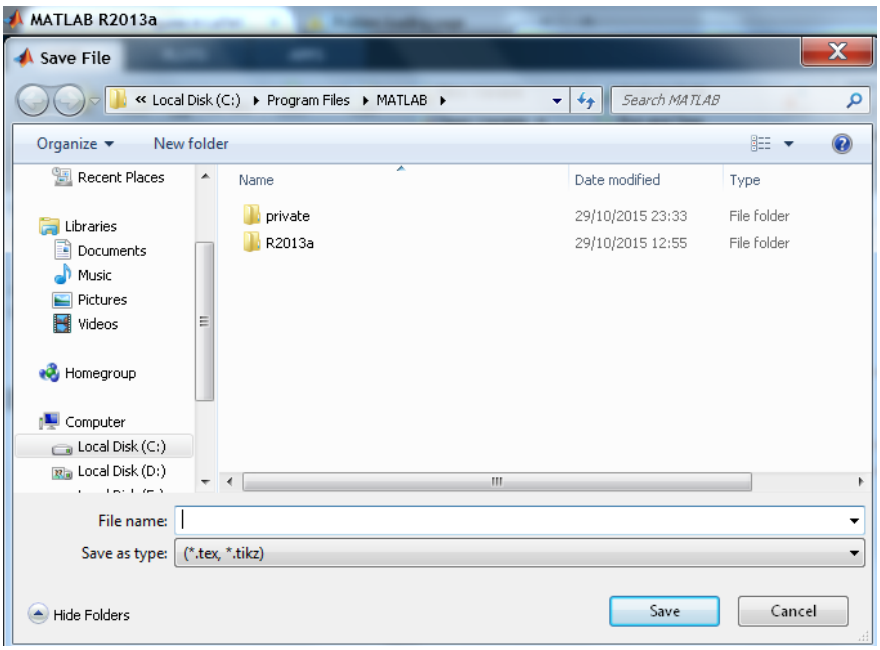
```
sphere(30);
title ('a sphere : x^2+y^2+z^2');
xlabel ('x');
ylabel ('y');
zlabel (' z ');
axis equal
```

1. Nous téléchargeons le dossier "matlab2tikz" depuis `http://www.mathworks.com/matlabcentral/fileexchange/22022?controller=file_infos&download=true`

2. de téléchargement, nous décompressons le dossier, l'ouvrons et entrons dans un autre dossier src, copions les fichiers qu'il contient : `C:\Program Files\MATLAB`

3. après dessiné la boule précédente dans Matlab, nous effectuons les opérations suivantes :
   1- On localise les fichiers que l'on a copié précédemment dans le programme MATLAB, puis on clique sur « matlab2tikz.m » puis on choisit « Exécuter » F9, puis on nomme le fichier, par exemple, matlab1 et on le place avec le fichier Tex

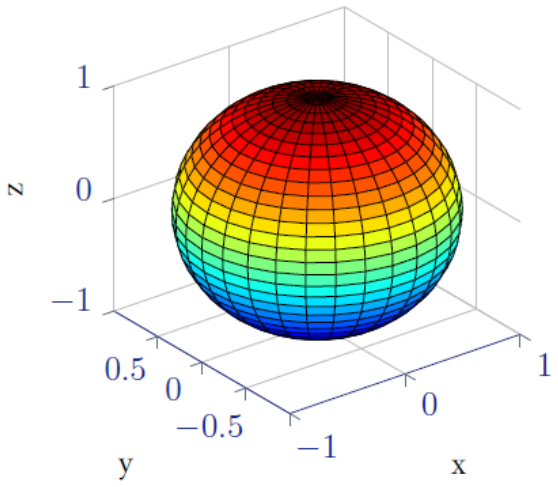




4. Nous appelons le package pgfplots, c'est-à-dire `\usepackage{pgfplots}` dans le préambule

5. Nous appelons le fichier matlab1 via la commande `\input {file}` à l'intérieur de la figure.

```
%\usepackage{pgfplots}
\begin{figure}[H]
\centering
\input{matlab1.tikz}
\caption*{A plot of a sphere}
\label{fig :sphere}
\end{figure}
```



A plot of a sphere

Conclusion

En conclusion, le langage LaTeX est un outil incontournable pour la rédaction de documents mathématiques de qualité professionnelle. Il permet de créer des formules mathématiques complexes avec une grande précision et une grande flexibilité, tout en offrant des options de mise en forme et de présentation avancées. Bien que son apprentissage puisse sembler intimidant au début, il s'avère être un investissement précieux pour les étudiants, les enseignants et les chercheurs en mathématiques. En somme, LaTeX est un langage indispensable pour toute personne qui souhaite communiquer efficacement des idées mathématiques de manière claire, concise et professionnelle.

bibliographie

[1] F. BOERKMANN, *Des courbes en LaTeX* : *Extension pstricks-add*. 2008

[2] F. Mittelbach, M. Goossens,J. Braams , D .Carlisle, Ch. Rowley, *The LaTeX Companion*.

[3] G. Grätzer, *More Math Into LaTeX.* 1996

[4] H. Kopka et Patrick , W. Daly, *A Guide to LaTeX.* 1992

[5] https://www.overleaf.com/learn/latex/Bibliography_management_with_bibtex

[6] https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes

[7] https://fr.wikipedia.org/wiki/MetaPost

[8] L. Lamport, *a Document Preparation System User's Guide and Reference Manual*. 1994

[9] S. Kottwitz, *LaTeX Beginner's Guide.* 2011

[10] V. Lozano. *Tout ce que vous avez toujours voulu savoir sur LATEX sans jamais oser le demander*. 2008

[11] W. Appel, E. Chevalier, E. Cornet, Desreux S., Fleck J.- J., and Pichaureau P - *LATEX pour l'impatient. In Technique et pratique.* H & K, 2007

Résumé

Le langage LaTeX est largement utilisé dans le domaine des mathématiques en raison de ses fonctionnalités avancées pour la rédaction et la mise en forme d'équations mathématiques. Il offre un contrôle précis sur la disposition des symboles, des indices, des exposants et des opérateurs mathématiques. Voici un résumé général du langage LaTeX en mathématiques :

1. Équations mathématiques : LaTeX permet de créer des équations mathématiques en ligne ou centrées. Il propose des environnements spécifiques tels que "`equation`" pour les équations numérotées et "`align`" pour les alignements d'équations.

2. Symboles mathématiques : LaTeX offre une large gamme de symboles mathématiques prédéfinis. On peut les insérer en utilisant des commandes spécifiques, par exemple "`\alpha`" pour α "`\sum`" pour $\sum$, ou "`\int`" pour $\int$.

3. Opérations mathématiques : LaTeX permet de réaliser diverses opérations mathématiques telles que l'addition, la soustraction, la multiplication, la division, les exposants et les indices. Les commandes telles que "`^`" pour l'exposant et "`_`" pour l'indice sont utilisées pour spécifier ces opérations.

4. Matrices et vecteurs : LaTeX propose des environnements spécifiques pour la création de matrices et de vecteurs. Par exemple, l'environnement "`matrix`" permet de créer des matrices et l'environnement "`pmatrix`" permet de créer des matrices entourées de parenthèses.

5. Fonctions mathématiques : LaTeX offre des commandes pour les fonctions mathématiques courantes telles que les logarithmes, les exponentielles, les trigonométriques, les fonctions hyperboliques, etc. Par exemple, "`\sin`" pour $\sin$, "`\log`" pour $\log$. ou "`\sqrt`" pour la racine carrée $\sqrt{}$.

6. Notation mathématique avancée : LaTeX prend en charge des notations mathématiques avancées telles que les intégrales, les dérivées, les limites, les fractions, les ensembles, les probabilités, etc. Il propose des commandes spécifiques pour chacune de ces notations.

7. Mathématiques : LaTeX permet d'ajouter des références et des références et annotations aux équations mathématiques. On peut les numéroter et les référencer dans le texte à l'aide de commandes spécifiques.

8. LaTeX offre une multitude de possibilités pour concevoir des documents. Dans cet article, je vais montrer qu'il est rapidement possible d'effectuer le tracé d'une courbe sans avoir recours à un logiciel externe

En conclusion, LaTeX est un outil puissant pour la rédaction et la mise en forme d'équations mathématiques. Il offre une flexibilité et un contrôle précis sur la présentation des symboles, des opérations et des notations mathématiques. C'est pourquoi il est largement utilisé dans le domaine des mathématiques, de la recherche scientifique et de l'enseignement des mathématiques.

Abstract

The LaTeX language is widely used in the field of mathematics due to its advanced features for writing and formatting mathematical equations. It provides precise control over the layout of symbols, subscripts, superscripts, and mathematical operators. Here is a general summary of the LaTeX language in mathematics :

1. Mathematical Equations : LaTeX allows for the creation of inline or centered mathematical equations. It provides specific environments such as "equation" for numbered equations and "align" for aligning multiple equations.

2. Mathematical Symbols : LaTeX offers a wide range of predefined mathematical symbols. These symbols can be inserted using specific commands, for example, "`\alpha`" for α, "`\sum`" for $\sum$, or "`\int`" for $\int$.

3. Mathematical Operations : LaTeX enables various mathematical operations such as addition, subtraction, multiplication, division, exponents, and subscripts. Commands like "`^`" for exponent and "`_`" for subscript are used to specify these operations.

4. Matrices and Vectors : LaTeX provides specific environments for creating matrices and vectors. For example, the "`matrix`" environment allows for the creation of matrices, while the "`pmatrix`" environment creates matrices surrounded by parentheses.

5. Mathematical Functions : LaTeX offers commands for common mathematical functions such as logarithms, exponential, trigonometric functions, hyperbolic functions, etc. For instance, "`\sin`" for $\sin$, "`\log`" for $\log$, or "`\sqrt`" for square root $\sqrt{}$.

6. Advanced Mathematical Notation : LaTeX supports advanced mathematical notations such as integrals, derivatives, limits, fractions, sets, probabilities, etc. It provides specific commands for each of these notations.

7. Mathematical References and Annotations : LaTeX allows for the addition of references and annotations to mathematical equations. Equations can be numbered and referenced in the text using specific commands.

8. LaTeX offers a multitude of possibilities for designing documents. In this article, we will show that you can quickly draw a curve without having to use external software

In conclusion, LaTeX is a powerful tool for writing and formatting mathematical equations. It offers flexibility and precise control over the presentation of symbols, operations, and mathematical notations. That is why it is widely used in the field of mathematics, scientific research, and mathematics education.

ملخص

لغة LaTeX مستخدمة على نطاق واسع في مجال الرياضيات بسبب إمكانياتها المتقدمة في كتابة وتنسيق المعادلات الرياضية. إنها توفر تحكمًا دقيقًا في تخطيط الرموز والأسهم والأسس والعمليات الرياضية. فيما يلي ملخص عام للغة LaTeX في مجال الرياضيات:

1. المعادلات الرياضية: LaTeX تسمح بإنشاء المعادلات الرياضية على السطر أو في وسط الصفحة. تقدم بيئات خاصة مثل "equation" للمعادلات المرقمة و "align" لتنسيق المعادلات.
2. الرموز الرياضية: LaTeX توفر مجموعة واسعة من الرموز الرياضية المعرفة مسبقًا. يمكن إدراجها باستخدام أوامر محددة مثل "\alpha" لعرض الرمز α، "\sum" لعرض الرمز $\sum$، أو "\int" لعرض الرمز $\int$.
3. العمليات الرياضية: LaTeX تسمح بأداء مختلف العمليات الرياضية مثل الجمع والطرح والضرب والقسمة والأسس والأسهم. تُستخدم الأوامر مثل "^" للأس و "_" للسهم لتحديد هذه العمليات.
4. المصفوفات والمتجهات: LaTeX تقدم بيئات خاصة لإنشاء المصفوفات والمتجهات. على سبيل المثال، بيئة "matrix" تُستخدم لإنشاء المصفوفات و بيئة "pmatrix" تُستخدم لإنشاء المصفوفات محاطة بالأقواس.
5. الوظائف الرياضية: LaTeX توفر أوامر للوظائف الرياضية الشائعة مثل اللوغاريتمات والأسس والدوال المثلثية والدوال الهايبربوليكية، وما إلى ذلك. على سبيل المثال، "\sin" لعرض الجيب الجبري sin، "\log" لعرض اللوغاريتم الطبيعي log، أو "\sqrt" لعرض الجذر التربيعي $\sqrt{}$.
6. العلامات الرياضية المتقدمة: LaTeX تدعم علامات رياضية متقدمة مثل التكاملات والمشتقات والحدود والكسور والمجموعات والاحتمالات وغيرها. تقدم أوامر محددة لكل من هذه العلامات.

7. الإشارات والتعليقات الرياضية: تُمكن LaTeX من إضافة إشارات وتعليقات إلى المعادلات الرياضية. يمكن ترقيمها والرجوع إليها في النص باستخدام أوامر خاصة.
8. LaTeX يقدم العديد من الإمكانيات لتصميم المستندات. في هذا المقال، سأوضح كيف يمكن بسرعة رسم منحنى دون الحاجة إلى استخدام برامج خارجية.

في الختام، LaTeX هو أداة قوية لكتابة وتنسيق المعادلات الرياضية. إنه يقدم مرونة وتحكمًا دقيقًا في تقديم الرموز والعمليات والعلامات الرياضية. ولهذا السبب، يُستخدم على نطاق واسع في مجال الرياضيات والبحث العلمي وتعليم الرياضيات.