Utilisation de Maxima au lycée

I Feuille de calcul

• « ; » exécute une commande en affichant le résultat. Par exemple `1+2/3;`
• « $ » exécute une commande sans afficher le résultat. Par exemple `a:2 $`
• « % » rappelle le dernier calcul effectué
•`? plot2d` affiche l'aide en ligne sur l'instruction `plot2d`
• `example(expand)` affiche des exemples d'utilisation de l'instruction `expand`
• `kill(all)` réinitialise le système

II Opérateurs

• les quatre opérations usuelles `+ , - , * , /`
• opérateur « ^ » élévation à une puissance. `x^3` est x^3
• opérateur « # » non égal à (ou différent de)
• opérateurs de comparaison `= , < , <= , > , >=`
• opérateur « : » d'affectation.
`a:3` donne la valeur 3 à la variable a.
• opérateur « := » pour définir une fonction.
• opérateur « = » indique une équation dans Maxima.
• opérateur « ! » factoriel d'un entier naturel,
par exemple $5! = 1 \times 2 \times 3 \times 4 \times 5 = 120$.
• opérateur « . » de multiplication de deux matrices.

III Constantes

• `%pi` désigne $\pi \approx 3,14159$
• `%e` désigne $\mathrm{e} = \exp(1) \approx 2,7183$
• `%i` est l'imaginaire pur de module 1, d'argument $\pi/2$
• `true` valeur "vrai"
• `false` valeur "faux"
• `inf` désigne $+\infty$
• `minf` désigne $-\infty$
• `%gamma` constante d'Euler-Mascheroni qui est la limite de la suite de terme général $\left(\sum_{k=1}^{n} \frac{1}{k}\right) - \ln n$

IV Nombres réels

IV.1 fonctions usuelles

• `abs(x)` valeur absolue de x
• `floor(x)` partie entière de x
• `sqrt(x)` racine carrée de x
• `sin(x) , cos(x) , tan(x)`
• `exp(x) , log(x)` *Attention* : log désigne la fonction **logarithme népérien**

IV.2 valeurs approchées

• `float(x)` fournit une valeur décimale approchée de x
• `bfloat(x)` donne une valeur approchée de x en notation scientifique
• `fpprec:20` fixe la précision de la valeur approchée donnée par `bfloat` (20 chiffres affichés au lieu de 16 par défaut)

IV.3 trigonométrie

• `acos(0.2)` donne la mesure en radian de l'angle géométrique ayant pour cosinus 0,2
• `trigexpand(a)` développe l'expression trigonométrique a en utilisant les formules d'addition de cos et sin. Par exemple, `trigexpand(cos(x+y))` renvoie $\cos x \cos y - \sin x \sin y$
• `trigreduce(a)` permet de linéariser un polynôme trigonométrique a. Par exemple, `trigreduce(sin(x)^3)` renvoie $\frac{3\sin x - \sin(3x)}{4}$
• `trigsimp(a)` simplifie l'expression trigonométrique a en utilisant la relation $\cos^2 t + \sin^2 t = 1$ et en remplaçant $\tan t$ par $\frac{\sin t}{\cos t}$

V Arithmétique des entiers

Soit a et b deux entiers. Soit n et p deux entiers naturels.
• `divide(a,b)` division euclidienne de a par b.
Le résultat est une liste dont le premier élément est le quotient et le second élément le reste
• `divisors(a)` ensemble des diviseurs positifs de a
• `divsum(a)` somme des diviseurs positifs de a
• `mod(a,b)` reste de la division de a par b
• `gcd(a,b)` pgcd de a et b
• `load(functs) $ lcm(a,b)` ppcm de a et b
• `primep(p)` teste si p est premier
• `p:prev_prime(n)` donne le nombre premier p qui vient juste avant n, avec $p < n$
• `next_prime(n)` donne le nombre premier qui vient juste après n
• `factor(n)` décompose n en produit de facteurs premiers
• `ifactors(n)` décompose n en produit de facteurs premiers en affichant le résultat sous forme de liste
• `binomial(n,p)` est le cefficient binomial $\binom{n}{p}$
• `random(n)` renvoie un entier naturel, choisi au hasard entre 0 et $n-1$ lorsque $n \in \mathbb{N}^*$

VI Nombres complexes

Soit z un nombre complexe.
• `%i` désigne le complexe i
• `realpart(z)` partie réelle de z
• `imagpart(z)` partie imaginaire de z
• `conjugate(z)` conjugué de z
• `abs(z)` module de z
• `carg(z)` argument de z (dans $]-\pi, \pi]$)
• `rectform(z)` écrit z sous forme algébrique
• `polarform(z)` écrit z sous forme exponentielle

VII Calcul algébrique

Soit P et Q deux polynômes.
- `expand(P)` développe P
- `factor(P)` factorise P
- `gfactor(P)` factorise P dans l'ensemble $\mathbb{C}$
- `divide(P,Q,x)` calcule le quotient et le reste de la divison de P par Q. Le résultat est une liste dont le premier élément est le quotient et le second élément le reste
- `partfrac(P/Q,x)` décompose la fonction rationnelle P/Q (de la variable x) en éléments simples
- `ratsimp(expr)` simplifie l'expression `expr` (en écrivant tout sur le même dénominateur)
- `subst(1/z,x,expr)` remplace x par $1/z$ dans l'expression `expr`

VIII Fonctions numériques

VIII.1 définir une fonction

- `f(x):=x^2+2*x-3`
- `define(f(x),x^2+2*x-3)`
- `f:lambda([x],x^2+2*x-3)`

VIII.2 limites, tangentes et asymptotes

- `limit(sin(x)/x,x,0)` limite en 0
- `limit(1/x,x,0,plus)` limite à droite en 0
- `limit(1/x,x,0,minus)` limite à gauche en 0
- `limit(x*exp(x),x,minf)` limite en $-\infty$
- `taylor(f(x),x,a,1)` permet d'obtenir l'équation réduite de la tangente à $\mathcal{C}_f$ au point $A(a, f(a))$
- `taylor(sqrt(1+x^2),x,inf,2)` permet d'obtenir le développement asymptotique à 2 termes de $x \mapsto \sqrt{1+x^2}$ en $+\infty$

VIII.3 dérivation

- `diff(f(x),x)` calcule la dérivée $f'(x)$
- `diff(f(x),x,2)` calcule $f''(x)$, dérivée seconde

VIII.4 courbes représentatives

Pour afficher les courbes $\mathcal{C}_f$ et $\mathcal{C}_g$ sur le même graphique, dans la fenêtre $[x_1, x_2] \times [y_1, y_2]$, on entre :
- `plot2d([f(x),g(x)],[x,x1,x2],[y,y1,y2])`

VIII.5 intégrales

- `integrate(f(x),x)` calcule une primitive de la fonction f
- `integrate(f(x),x,a,b)` calcule l'intégrale $\displaystyle\int_a^b f(x)\,\mathrm{d}x$
- `romberg(1/log(x),x,2,3)` fournit une approximation de l'intégrale $\displaystyle\int_2^3 \frac{1}{\ln x}\,\mathrm{d}x$

IX Equations

IX.1 résolution d'équations

Résoltuion exacte dans l'ensemble $\mathbb{C}$ des complexes :
- `solve(x^2+x=1,x)`

Résolution approchée dans $\mathbb{R}$:
- `find_root(x^5=1+x,x,1,2)` solution dans $[1, 2]$

IX.2 systèmes linéaires

Pour résoudre le système $\begin{cases} 3x+2y=1 \\ x-y=2 \end{cases}$
- `S1:[3*x+2*y=1,x-y=2]`
- `solve(S1,[x,y])`

IX.3 équations différentielles

Pour résoudre l'équation différentielle $y'' + w^2\, y = \sin x$, on définit d'abord l'équation :
- `eqn:'diff(y,x,2)+w^2*y=sin(x)`

On la résout :
- `sol:ode2(eqn,y,x)`

Pour trouver la solution satisfaisant aux conditions initiales $y(0) = 1$ et $y'(0) = -1$, on entre :
- `ic2(sol,x=0,y=1,diff(y,x)=-1)`

Pour trouver la solution satisfaisant aux conditions $y(0) = 1$ et $y(1) = 0$, on entre :
- `bc2(sol,x=0,y=1,x=1,y=0)`
- `rhs(sol)` saisit le membre de droite de l'égalité `sol` obtenue ci-dessus.

X Listes

Une liste est un type de données, qui tient compte de l'ordre, accepte les répétitions d'éléments et est délimité par les caractères [et]. Voici quelques fonctions importantes concernant les listes :
- `L:makelist(k^2,k,0,9)` permet de créer la liste des carrés des 10 premiers naturels, k prenant toutes les valeurs entières de 0 jusqu'à 9.
- `L[2]:5` remplace le 2ème élément de la liste L par 5.
- `length(L)` donne le nombre d'éléments de la liste L.
- `first(L) ; second(L) ; last(L)` renvoient respectivement le premier, le second, le dernier élément de L.
- `member(x,L)` vaut `true` si x appartient à la liste L (`false` sinon).
- `append([a,1,3],[2,7])` regroupe les deux listes en une seule liste $[a, 1, 3, 2, 7]$.
- `join(l,m)` crée une nouvelle liste constituée des éléments des listes l et m, intercalés. La liste obtenue est $[l[1], m[1], l[2], m[2], l[3], m[3], \ldots]$.
- `sort(L)` permet de ranger les éléments de la liste L par ordre croissant.
- `map(f,L)` permet d'appliquer la fonction f à tous les éléments de la liste L.

XI Sommation et produit

XI.1 somme finie

• `sum(1/k^2,k,1,10)` calcule la somme des inverses des carrés des entiers compris entre 1 et 10.

XI.2 produit fini

• `product(sqrt(k),k,1,10)` calcule le produit des racines carrées des entiers compris entre 1 et 10.

XI.3 somme infinie

On peut montrer que la suite $(u_n)_{n\in\mathbb{N}^*}$ de terme général $u_n = \sum_{k=1}^{n} \frac{1}{k^2}$ est convergente. Sa limite est notée $\sum_{k=0}^{+\infty} \frac{1}{k^2}$.
On peut demander sa valeur exacte comme suit :
• `load(simplify_sum) $ sum(1/k^2,k,1,inf) $ simplify_sum(%)`

XII Programmation

XII.1 syntaxe d'une procédure

nom(paramètres en entrée) := block([variables locales], <instruction 1>, <instruction 2>, ...
/* ——-Commentaire—— */
)$
Voici un exemple simple de procédure qui additionne deux nombres.
• `somme(a,b):=block([c], c:a+b, return(c))`

XII.2 structure conditionnelle

```
• if (condition)
then (<instruction1> , <instruction2>)
else (<instruction3> , <instruction4>)
```

XII.3 structures itératives

Boucle **For** et affichage de la table de 7 :
• `for k from 1 thru 10 do`
`( print("7 fois",k,"égale",7*k) )`

Boucle **While** et affichage de la table de 7 :
• `k:1 $ while k<11 do`
`( print("7 fois",k,"égale",7*k) , k:k+1 )`

XIII Matrices

Soit B une matrice de taille 3×3.
On définit la matrice $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & -9 \end{pmatrix}$ ligne par ligne de la façon suivante :
• `A:matrix([1,2,3],[4,5,6],[7,8,9])`
• `A+B` somme des matrices A et B
• `3*A` produit de la matrice A par le réel 3
• `A.B` produit des matrices A et B
• `A^^3` matrice A élevée à la puissance 3
• `invert(A)` inverse A^{-1} de la matrice A