

1 Les entrées

- «**Alt+n**» ouvre une nouvelle ligne de commandes.
- «**Alt+p**» ouvre l'environnement de programme.
- «**Alt+t**» ouvre l'environnement de tableur.
- «**Alt+g**» ouvre l'environnement de graphique 2d.
- «**Alt+h**» ouvre l'environnement de graphique 3d.
- «**Alt+c**» ouvre une ligne de commentaires.
- «**Alt+d**» ouvre l'environnement dessin tortue.
- «**Alt+e**» ouvre l'éditeur d'expression.
- Pour **supprimer un niveau**, cliquez sur le numéro du niveau, il devient noir
Edit→Supprimer niveaux sélectionnés

2 Les éléments et constantes prédéfinies

- «**L:=[1,2,4,2]**» Pour créer une **liste** pour les statistiques par exemple.
- «**S:=(1,2,4,2)**» Pour créer une **séquence**.
- «**E:= % {1,2,4,2%}**» ou «**E:=set[1,2,4,2]**» Pour créer un **ensemble**
- Les Constantes prédéfinies
 - «**pi**» est le nombre $\pi \approx 3,14$
 - «**e**» est le nombre $e = \exp(1) \approx 2,71$
 - «**infinity**» est ∞ .
 - «**inf**» ou «**+infinity**» est $+\infty$.
 - «**-infinity**» est $-\infty$.

3 Nombres réels et opération sur les nombres

- «**4^2**» retourne le **carré** de 4.
- «**sqrt(2)**» retourne la **racine carrée** de 4.
- «**approx(x,0)**» **arrondi** x à l'entier
- «**approx(x,2)**» **arrondi** x au centième.
- «**approx(x,-1)**» **arrondi** x aux dizaines
- «**ceiling(x)**» ou «**ceiling(x)**» renvoie le plus petit entier supérieur à x .
- «**floor(x)**» renvoie le plus grand entier inférieur à x .
- «**trunc(x)**» renvoie la **troncature** de x .

- «**iquo(7,2)**» renvoie le **quotient** de la division euclidienne de 7 par 2.
- «**irem(7,2)**» renvoie le **reste** de la division euclidienne de 7 par 2.
Si $\text{irem}(n,2)=0$ alors n est pair sinon n est impair.
- «**cos(x)**» renvoie le **cosinus** de x . x étant en radian
- «**sin(x)**» renvoie le **sinus** de x .
- «**tan(x)**» renvoie la **tangente** de x .
- «**acos(x)**» est la fonction réciproque de la fonction cosinus.
 $\text{acos}(\text{sqrt}(3)/2)$ renvoie $\frac{\pi}{6}$
- «**asin(x)**» est la fonction réciproque de la fonction sinus.
- «**atan(x)**» est la fonction réciproque de la fonction tangente.
- «**ln(x)**» ou «**log(x)**» est le **logarithme népérien** de x .
- «**log10(x)**» est le logarithme de base 10 de x .
- «**exp(x)**» ou «**e^x**» est l'**exponentielle** de x .

4 Les listes (ou vecteur) et les séquences

Les listes sont écrites entre crochets et les séquences entre parenthèses.

- «**x:=[1,2,3,4]**» pour définir la liste (1,2,3,4) dans la variable x .
- «**x:=(1,2,3,4)**» ou «**x:=seq[1,2,3,4]**» ou encore «**x:=1,2,3,4**» pour définir la séquence (1,2,3,4) dans la variable x .
- «**x[2]**» pour accéder au 3^eélément de la liste (ou de la séquence) x .
- «**x[0]**» revoie le 1^{er}élément de la liste (ou de la séquence) x .
- «**x[n]**» revoie le $n+1$ ^eélément de la liste (ou de la séquence) x .
- «**x[0..2]**» pour extraire une sous-séquence (sous-liste) des 3 premiers éléments de la liste (ou de la séquence).
- «**y:=seq(j^2,j,1,4)**» pour créer une liste (des carrés ici).
- «**y:=seq(j^2,j,1..4)**» ou «**y:=seq(j^2,j=1..4)**» pour créer une séquence (des carrés ici).
- «**x:=append(x,d)**» ajoute l'élément d à la fin de la liste x .
- «**x:=x,d**» ajoute l'élément d à la fin de la séquence x .
- «**x,y**» pour concaténer les séquences x et y .
- «**nop(x)**» ou «**[x]**» Pour transformer la séquence x en liste.
- «**op(x)**» Pour transformer la liste x en séquence.

5 Statistiques

On fait des statistiques avec des listes

- «**u:=[8,9,9,8,10,8]**» crée une liste (ou vecteur) u
- Pour les séries statistiques **avec regroupement par classe** on utilise deux listes de même dimension.

valeurs	8	9	10
effectifs	3	2	1

est représenté par $\text{val} := [8,9,10]$ et $\text{eff} := [3,2,1]$
la moyenne est obtenue avec «**moyenne(val,eff)**» et il en est de même pour les autres paramètres statistiques.

- «**size(u)**» renvoie le nombre d'éléments de la liste u .
- «**count_inf(3,u)**» renvoie le nombre d'éléments de la liste u strictement inférieures à 3.
- «**count_sup(3,u)**» renvoie le nombre d'éléments de la liste u strictement supérieures à 3.
- «**count_eq(3,u)**» renvoie le nombre d'éléments de la liste u égales à 3.
- «**moyenne(u)**» renvoie la **moyenne** de u .
«**moyenne(val,eff)**» renvoie la **moyenne** de la série (val,eff).
- «**variance(u)**» renvoie la **variance** de u .
«**variance(val,eff)**» pour la **variance** de la série (val,eff).
- «**ecart_type(u)**» renvoie l'**écart-type** de u «**ecart_type(val,eff)**» renvoie l'**écart-type** de la série (val,eff).
- «**mediane(u)**» renvoie la **médiane** de u .
- «**quartile1(u)**» ou «**quantile(u,0.25)**» renvoie le **premier quartile** de u .
- «**quartile3(u)**» ou «**quantile(u,0.75)**» renvoie le **troisième quartile** de u .
- «**quantile(u,0.1)**» renvoie le **premier décile** de u .
- «**quantile(u,0.9)**» renvoie le **neuvième décile** de u .
- «**quartiles(u)**» renvoie dans l'ordre : le minimum, le 1^{er} quartile, la médiane, le 3^e quartile et le maximum de u .
- «**moustache(u)**» trace la boîte à moustaches de u .

6 Aléatoire

On pourra utiliser pour générer un nombre aléatoire **rand**, **alea** ou **hasard**.

- «**alea(6)**» renvoie au hasard un nombre entier entre 0 et 5.
- «**alea(6)+1**» renvoie au hasard, un nombre entier entre 1 et 6.
- «**alea(1,6)**» renvoie au hasard un nombre décimal entre 1 et 6.

- «**alea(4,1,6)**» renvoie au hasard 4 nombres entiers entre 1 et 6 **sans répétition** (sans remise).
- «**randvector(100,'rand(6)+1')**» renvoie au hasard 100 nombres entiers entre 1 et 6 donc **avec répétition** (avec remise).
- «**alea(randnorm(0,1))**» renvoie au hasard des nombres répartis selon la **loi normale** de moyenne 0 et d'écart type 1.
- «**binomial(n,k,p)**» renvoie $p(X = k)$ lorsque X suit une **loi binomiale** $\mathcal{B}(n, p)$.
- «**comb(n,k)=**» renvoie nk le nombre de **combinaison** de k éléments parmi n .
- «**factorial(4)**» renvoie la **factorielle** 4
- «**perm(10,2)**» renvoie le nombre d'**arrangements** de 2 objets parmi 10.

7 Les programmes

- «**saisir("X=",x)**» Pour demander une valeur qui sera stockée dans la variable x .
- «**input("X=",x)**» idem mais en anglais.
- «**saisir("age=",age,"taille=",taille)**» pour demander deux valeurs.
- «**print("X="+x)**» pour afficher dans un programme $X=$ suivi de la valeur de x .
- «**print("X="+x+" et Y="+y)**» idem en plus complet.
- «**output("X="+x)**» Pour afficher dans une fenêtre la valeur de la variable x .
- «**F9**» pour compiler le programme.

8 Les tests et opérateurs

- «**x==20**» x est il égal à 20 ?
- «**est_element(20,x)**» 20 appartient il à x ?
- «**x<20 or x>40**» x est il inférieur à 20 **ou** supérieur à 40 ?
- «**x<20 ou x>40**» x est il inférieur à 20 **ou** supérieur à 40 ?
- «**x<40 & x>30**» x est il inférieur à 20 **et** supérieur à 40 ?
- «**!**» signifie le contraire.
- «**x!=20**» x est il différent de 20 ?
- «**&&**» ou «**et**» ou «**and**» signifie **et**. Par exemple $x \geq 1$ et $x \leq 5$
- «**||**» ou «**ou**» ou «**or**» signifie **ou**. Par exemple $x \leq 1$ ou $x \leq 5$

9 Les boucles

- Le test si.
→ En français :
si condition **alors** <instruction1> **sinon** <instruction2> **fsi** ;

→ Avec la même syntaxe que R-cran

```
si condition { <instruction> } sinon { <instruction2> }
```

→ En Anglais :

```
if condition then <instruction> else <instruction2> end_if ;
```

→ Exemple :

Pour plus de clarté il est préférable de rédiger en verticale

```
si condition
alors
<instruction1>;
<instruction2>;
sinon
<instruction3>;
<instruction4>;
fsi
```

```
saisir("entrer un nombre",n) ;
si irem(n,2)=0
alors print("ce nombre est pair");
sinon print("ce nombre est impair");
fsi;
```

• La boucle **pour**.

→ **pour** (k de 1 jusque 100 pas 3) **faire** <instructions> **fpour**
(On peut omettre pas 3, le pas sera alors de 1.

→ En anglais

```
for (k from 1 to 100 by 2) do <instructions> end_for
(on peut utiliser step à la place de by).
```

→ Encore une autre façon de rédiger.

```
for (k:=1;k<=100;k:=k+2) { <instructions> }
```

Exemple :

```
for (k in 1:10) {print(k^2)}
```

• La boucle **tantque**.

→ **tantque** condition **faire** <instructions> **ftantque**

→ **tantque** condition { <instructions>; ... }

→ **while** condition **do** <instructions>; **end_while**

→ Exemple :

```
i:=0;
tantque i<=10 faire
print(i); i=i+1;
ftantque;
```

• La boucle **repetet**.

→ **repetet** <instruction>; **jusqu_a** condition

→ **repeat** <instruction>; **until** condition

→ Exemple :

```
repetet x:=x-5; jusqu_a x<5;
```

10 Les fonctions


- «**f(x):=x²+2x+1**» pour définir la fonction $f(x) = x^2 + 2x + 1$.
- «**graphe(f(x),x=-3..2)**» pour **tracer** la courbe représentative de f sur $[-3; 2]$.
- «**graphe([f(x),g(x)],x=-3..2)**» pour **tracer** la courbe représentative de f et de g .
- «**g:=deriver(f(x))**» donne la **dérivée** de f . g est alors une **expression** pas une fonction.
- «**f'(x)**» donne la dérivée de la fonction f .
- «**f''(x)**» donne la dérivée seconde de la fonction f .
- «**g:=unapply(g,x)**» pour transformer l'expression g en fonction g .
- «**g:=fonction_derivee(f)**» pour avoir directement la **fonction dérivée**.
- «**integrer(f,x)**» pour calculer la **primitive** de f .
- «**integrer(f,x,0,1)**» pour calculer l'**intégrale** $\int_0^1 f(x)dx$
- «**limite(f(x),x,a)**» pour calculer la **limite** de f en a .
 a peut être un nombre, $+\infty$ ou $-\infty$ (pour $+\infty$ et $-\infty$).
On peut rajouter un 4^e paramètre, +1 pour une **limite à droite** de a ou -1 ou une **limite à gauche** de a .
Exemple :

→ $\text{limite}(1/(x-1),x,1,+1)$ pour $\lim_{x \rightarrow 1^+} \frac{1}{x-1}$

→ $\text{limite}(1/(x-1),x,1,-1)$ pour $\lim_{x \rightarrow 1^-} \frac{1}{x-1}$

→ $\text{limite}(1/(x-1),x,+\infty)$ pour $\lim_{x \rightarrow +\infty} \frac{1}{x-1}$

11 Calcul algébrique

- «**developper**((x+1)*(x+3))» pour **développer** $(x+1) * (x+3)$. Le signe * est obligatoire.
Pour modifier une expression déjà écrite (développer, factoriser ...) on sélectionne à la souris cette expression puis on clique sur le bouton du menu d'équation  et on sélectionne factor pour factoriser par exemple.
- «**factoriser**(x^2+4*x+3)» pour **factoriser**, on obtient $(x+1) * (x+3)$.
Par défaut, la factorisation est faite sur \mathcal{Q} . Pour factoriser les polynômes de degré 2 même si cela introduit des radicaux, il faut changer la configuration, cliquer sur : Cfg→Configuration du CAS→Cocher la case Sqrt
- «**factoriser_sur_C**(z^2+1)» pour **factoriser** sur \mathcal{C} , On obtient $(i+z) * (-i+z)$.
- «**resoudre**(x^2+4x+3=0)» ou «**solve**(x^2+4x+3=0)» pour résoudre l'équation dans \mathcal{R} .
«[]» signifie que l'équation n'a pas de solution dans \mathcal{R} .
«[x]» signifie que tous les réels sont solutions.
- «**resoudre_dans_C**(x^2+4=0)» ou «**csolve**(x^2+4=0)» pour résoudre l'équation dans \mathcal{C} .
- «**resoudre_systeme_lineaire**([3*x-2*y=3,5*x+y=7],[x,y])» pour résoudre le système
$$\begin{cases} 3x - 2y = 3 \\ 5x + y = 7 \end{cases}$$
- «**forme_canonique**(x^2+5x-6)» retourne la **forme canonique** $(x-1)^2 + 2$.
- «**propFrac**((x+2)(x-1))» pour **décomposer** une fonction rationnelle. $1 + \frac{3}{x-1}$.
- «**divide**(x^2-2x-5, x-4)» renvoie $[x+2, 3]$ qui est le quotient et le reste de la division euclidienne de $x^2 - 2x - 5$ par $x - 4$.
- «**partfrac**((x^2-2*x+3)/(x^2-3*x+2)) » renvoie $1 + \frac{3}{x-2} + \frac{-2}{x-1}$ qui est la **décomposition en éléments simple** de la fonction rationnelle.
- «**substituer**(f(x),x=sqrt(2))» Remplace dans $f(x)$ la variable x par $\sqrt{2}$.

12 Les nombres complexes

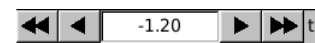
Dans le menu Math (Cmplx) se trouve les fonctions ayant comme paramètre une expression à valeur complexe.

- «**%i**» est le nombre complexe i .
- «**z:=(1+2*i)^2**» On affecte à z le nombre complexe $(1 + 2 * i)^2$.
- «**re(z)**» ou «**real(z)**» renvoie la **partie réelle** de z .

- «**im(z)**» ou «**imag(z)**» renvoie la **partie imaginaire** de z .
- «**evalc(z)**» Écriture de z sous la **forme cartésienne** $re(z) + i * im(z)$.
- «**abs(z)**» renvoie le **module** de z .
- «**arg(z)**» renvoie l'**argument** de z .
- «**conj(z)**» renvoie le **conjugué** de z .

13 Géométrie

- Alt+g (Alt+h) pour obtenir l'environnement de graphique 2d (3d).
- pour choisir des angles en radian (ou en degré) Cfg→Configuration du Cas→cocher (ou décocher) radian
- «**distance(A,B)**» ou textcolorred«longueur(A,B)» pour avoir la **distance** AB .
- «**angle(A,B,C)**» la mesure en radians (ou en degrés) de l'angle $(\overrightarrow{AB}, \overrightarrow{AC})$
- «**droite(3x+3)**» pour tracer la droite $y = 3x + 3$.
- «**graphe(3x+3)**» pour tracer la droite $y = 3x + 3$ et plus généralement la courbe d'équation $f(x)$.
- objets élémentaires
 - «**A:=point(2,3)**» pour placer le **point** A de coordonnée $(2;3)$
 - «**S:=segment**» pour définir et tracer le **segment** $[AB]$.
 - «**d1:=droite(A,B)**» pour définir et tracer la **droite** (AB) .
 - «**dd1:=demi_droite(A,B)**» pour définir et tracer la **demi droite** $[AB)$
 - «**E:=element(d1,1)**» pour créer le point E sur la droite d_1 .
On modifie le nombre 1 pour positionner le point sur la droite.
On peut aussi utiliser un curseur comme ci-dessous.
 - «**t:=element(-2..2)**» pour créer le **curseur** t avec $t \in [-2; 2]$.
 - «**E:=element(d1,t)**» sélectionner alors le curseur et modifier la valeur de t en cliquant.



- «**C1:=cercle(A,2)**» pour définir et tracer le **cercle** de centre A et de rayon 2.
- «**C2:=cercle(A,B)**» pour définir et tracer le **cercle** de diamètre $[AB]$.
- «**C2:=cercle(A,B-A)**» pour définir et tracer le **cercle** de centre A qui passe par B .
- Les vecteurs
 - «**vecteur(A,B)**» pour tracer le **vecteur** \overrightarrow{AB}

- «**vecteur**(B-A)» Pour tracer le **vecteur** \overrightarrow{AB} d'origine O .
- «**vecteur**(C,C+B-A)» Pour tracer le **vecteur** \overrightarrow{AB} d'origine C .
- Constructions élémentaires
 - «**mediatrice**(A,B)» pour tracer la **médiatrice** du segment $[AB]$.
 - «**milieu**(A,B)» pour construire le **milieu** du segment $[AB]$.
 - «**mediane**(A,B,C)» trace la **médiane** du triangle ABC issue de A .
 - «**hauteur**(A,B,C)» trace la **hauteur** du triangle ABC issue de A .
 - «**bissectrice**(A,B,C)» trace la **bissectrice** intérieure de l'angle A du triangle ABC .
 - «**perpendiculaire**(A,d)» pour tracer la **perpendiculaire** à la droite d passant par A .
 - «**parallele**(A,d)» pour tracer la **parallèle** à la droite d passant par A .
 - «**tangent**(C,A)» pour tracer les deux **tangentes** au cercle C passant par A si le point A est extérieur au cercle.
 - «**K:=barycentre**([A,2],[B,1],[C,-2],[D,3],[E,1])» pour créer le **barycentre** du système.
 - «**K:=isobarycentre**(A,B,C,D,E)» pour créer l'**isobarycentre** du système.
- Les transformations
 - «**t:=translation**(C-B)» pour créer la **translation** de vecteur \overrightarrow{BC} .
t(A) pour créer l'image du point A .
 - «**translation**(C-B,A)» pour créer directement l'image de A par la **translation** de vecteur \overrightarrow{BC} .
 - «**symetrie**(d,A)» pour créer l'image de A par la **symétrie** d'axe d .
 - «**symetrie**(B,A)» pour créer l'image de A par la **symétrie** de centre B .
 - «**rotation**(B,u,A)» pour créer l'image de A par la **rotation** de centre B et d'angle u .
 - «**projection**(d,A)» pour créer le **projeté orthogonal** de A sur la droite d .
 - «**h:=homothetie**(A,2)» pour créer l'**homothétie** de centre A et de rapport 2.
 - «**s:=similitude**(B,k,u)» pour créer la **similitude** de centre B , de rapport k et d'angle u .

Pour télécharger le logiciel Geogebra, il faut aller sur le site : <http://www.geogebra.org>

1 Les suites (listes et séquences)

- **L={A,B,C}** : définit une liste contenant trois points A, B, et C créés auparavant.
- **L={(0,0),(1,1),(2,2)}** : définit une liste contenant les points définis, bien qu'ils n'aient pas été nommés.
- **Longueur[liste L]** : Longueur de la liste L (nombre d'éléments de la liste).
- **Elément[L, n]** : n^eélément de la liste L
- **Min[L]** : Plus petit élément de la liste L
- **Max[L]** : Plus grand élément de la liste L

- **Séquence[e, i, a, b]** : Liste des objets créés en utilisant l'expression e et l'indice i variant du nombre a au nombre b . (Se traduit par : de $i = a$ à $i = b$ calculer la valeur de e).

Exemple : $L = \text{Séquence}[(2, i), i, 1, 5]$ crée une liste de 5 points dont l'ordonnée varie de 1 à 5.

- **Séquence[e, i, a, b, s]** : Liste des objets créés en utilisant l'expression e et l'indice i variant du nombre a au nombre b avec un pas de s .

Exemple : $L = \text{Séquence}[(2, i), i, 1, 5, 0.5]$ crée une liste de 9 points dont l'ordonnée varie de 1 à 5 avec un pas de 0.5.

- **ItérationListe[f, x₀, n]** : Liste L de longueur $n+1$ dont les éléments sont les images itératives par la fonction f de la valeur x_0 .

Exemple : la commande $L = \text{ItérationListe}[x^2, 3, 2]$ vous donne la liste

$$L = \{3, 3^2, (3^2)^2\} = \{3, 9, 81\}.$$

- **Définir une suite par sa formule générale $u_n = f(n)$**
Par exemple $u_n = 3n + 1$

→ On utilise **Séquence**[$3n+1, n, 0, 10$] pour obtenir les 10 premiers termes de la suite

→ **Séquence**[($n, 3n+1$), n , 0, 10] pour obtenir sa représentation graphique.

- **Définir une suite par la formule générale $u_{n+1} = f(u_n)$**
Par exemple $u_{n+1} = 2u_n + 1$ avec $u_0 = 1$.

On écrira donc






→ **u0=1** (juste pour avoir plus de clarté dans les formules, ce n'est pas nécessaire)

→ **L=Itération**[$2*x+1, u_0, 10$] pour avoir u_{10} .

→ **Séquence**[**Itération**[$2*x+1, u_0, i$], $i, 0, 9$] pour obtenir la liste des valeurs jusqu'à u_9 .

→ **Séquence**[(**i, Itération**[$2*x+1, u_0, i$]), $i, 0, 9$] Pour obtenir la représentation graphique des 10 premiers

2 Quelques icônes importants

-  Permet de définir une variable ou un paramètre qui appartient à un intervalle et que l'on pourra faire varier avec la souris.
-  Permet de tracer le lieu (la trace) d'un point dépendant d'un autre objet que l'on pourra faire varier ou déplacer.
-  Permet de calculer l'aire d'un polygone.
-  Permet de définir et de tracer les points d'intersection entre deux objets que l'on sélectionne avec la souris.
-  Permet de comparer deux objets que l'on sélectionne avec la souris.

3 Quelques fonctions de base

- **abs(x)** : Valeur absolue de x .

- **sgn(x)** : Renvoie $\frac{x}{|x|}$ pour avoir le signe de x .
- **sqrt(x)** : Renvoie la racine carrée de x .
- **exp(x)** : Renvoie l'exponentielle de x .
- **log(x)** : Renvoie le logarithme népérien de x .
- **lg(x)** : Renvoie le logarithme décimal de x .
- **ld(x)** : Renvoie le logarithme en base 2 de x .
- **cos(x)** : Renvoie le cosinus de x .
- **sin(x)** : Renvoie le sinus de x .
- **tan(x)** : Renvoie la tangente de x .
- **acos(x)** : Renvoie arc cosinus de x .
- **asin(x)** : Renvoie arc sinus de x .
- **atan(x)** : Renvoie arc tangente de x .
- **cosh(x)** : Renvoie le cosinus hyperbolique de x .
- **sinh(x)** : Renvoie le sinus hyperbolique de x .
- **tanh(x)** : Renvoie la tangente hyperbolique de x .
- **acosh(x)** : Renvoie arc cosinus hyperbolique de x .
- **asinh(x)** : Renvoie arc sinus hyperbolique de x .
- **atanh(x)** : Renvoie arc tangente hyperbolique de x .
- **floor(x)** : Renvoie le plus grand entier inférieur ou égal à x .
- **ceil(x)** : Renvoie le plus petit entier supérieur ou égal à x .
- **round(x)** : Renvoie l'arrondi à l'unité de x .
- **x(A)** : Renvoie l'abscisse de A .
- **y(A)** : Renvoie l'ordonnée de A .
- **cbrt(x)** : Renvoie la racine cubique de x .
- **random()** : Renvoie un nombre aléatoire entre 0 et 1.
- **gamma(x)** : Renvoie l'image de x par la fonction gamma.
- **x!** : Renvoie factorielle de x .

4 Les Fonctions

- **$f(x) = 3 * x^2 + 5$** : Définit la fonction f qui à x associe $3x^2 + 5$ et trace sa représentation graphique.

- **Fonction[f,a,b]** : Trace C_f entre a et b .
- **Si[C1,f,g]** : Renvoie f si condition C_1 sinon renvoie g . Permet de définir des fonctions par morceaux.
- **PointInflexion [f]** : Tous les points d'inflexion de la fonction f .
- **Extremum[f]** : Tous les extremums locaux de la fonction f .
- **g(x)=f(x+a)** : Définit g comme la fonction qui à x associe $f(x+a)$ et trace C_g .
- **g(x)=f(x) +a** : Définit g comme la fonction $f+a$ et trace C_g .
- **g(x)=af(x) +b** : Définit g comme la fonction $af+b$ et trace C_g .
- **g(x)=f(x)+h(x)** : Définit g comme la fonction $f+h$ et trace C_g .
- **g(x)=f(x)-h(x)** : Définit g comme la fonction $f-h$ et trace C_g .
- **g(x)=f(x)*h(x)** : Définit g comme la fonction $f \times h$ et trace C_g .
- **g(x)=f(x)/h(x)** : Définit g comme la fonction $\frac{f}{h}$ et trace C_g .
- **g(x)=f(x)^n** : Définit g comme la fonction f^n et trace C_g .
- **Translation[f, v]** : Translate C_f par la translation de vecteur \vec{v} .
- **Itération[f,x0,n]** : compose n fois l'image du nombre de départ x_0 par la fonction f .

5 Les équations

- **Racine [f,a]** : Une racine de f à partir de a (par la méthode de Newton).
- **Racine [f, a, b]** : Une racine de f sur $[a;b]$ (par la méthode de fausse position).
- **Racine[f]** : Toutes les racines de la fonction f .

6 Les fonctions dérivées

- **Dérivée[f] ou f'(x)** : Définit et trace la fonction dérivée de f .
- **f'(x)** : Définit et trace la fonction dérivée seconde de f .
- **Dérivée[f,n]** : Définit et trace la fonction dérivée n^{e} de f .

7 Intégrales et primitives

- **Intégrale[f,a,b]** : Renvoie le résultat de l'intégrale de f entre a et b et colorie l'aire entre C_f , l'axe des abscisses et les droites $x=a$ et $x=b$.
- **Intégrale[f, g,a,b]** : Renvoie le résultat de l'intégrale de $f-g$ entre a et b et colorie l'aire entre C_f , C_g , l'axe des abscisses et les droites $x=a$ et $x=b$.
- **Intégrale[f]** : Renvoie une primitive de f .
- **SommeInférieure[f,a,b,n]** : Approximation inférieure de l'intégrale de f sur l'intervalle $[a;b]$ par n .
- **rectangles. Note** : Cette commande dessine aussi les rectangles.
- **SommeSupérieure[f,a,b,n]** : Approximation supérieure de l'intégrale de f sur l'intervalle $[a;b]$ par n .
- **rectangles. Note** : Cette commande dessine aussi les rectangles.

8 Quelques longueurs

- **Longueur[f,x1, x2]** : Longueur de la portion de la courbe de la fonction f entre ses points d'abscisses x_1 et x_2 .
- **Longueur[f,A,B]** : Longueur de la portion de la courbe de la fonction f entre deux de ses points A et B .
- **Longueur[c,t1,t2]** : Longueur de la courbe c entre les deux points de paramètres t_1 et t_2 .
- **Longueur[c,A,B]** : Longueur de la courbe c entre deux de ses points A et B .

9 Les Tangentes

- **Tangente[a, f]** : Tangente à C_f en $x=a$.
- **Tangente[A, f]** : Tangente à C_f en $x=x(A)$.
- **Tangente[A, c]** : Tangente à la courbe c au point A .

10 Les Polynômes

- **Polynôme[f]** : Renvoie l'écriture polynomiale développée de la fonction f .
- **PolynômeTaylor[f,a,n]** : Renvoie le développement de Taylor de la fonction f à partir du point $x=a$ d'ordre n .
- **Racine[f]** : Toutes les racines du polynôme f .
- **Extremum[f]** : Tous les extremums locaux du polynôme f .
- **PointInflexion [f]** : Tous les points d'inflexion du polynôme f .

11 Les intersections

- **Intersection[f1, f2]** : Tous les points d'intersection entre les courbes C_{f_1} et C_{f_2} des polynômes f_1 et f_2 .
- **Intersection[f1, f2, n]** : même point d'intersection entre les courbes C_{f_1} et C_{f_2} des polynômes f_1 et f_2 .
- **Intersection[f, g, A]** : Premier point d'intersection entre C_f et C_g à partir de A (par la méthode de Newton).

12 Les fonctions d'arithmétique

- **Reste[a,b]** : Reste de la division euclidienne du nombre a par le nombre b .
- **Quotient[a,b]** : Quotient de la division euclidienne du nombre a par le nombre b .

- **Min[a,b]** : Minimum des deux nombres a et b .
- **Max[a,b]** : Maximum des deux nombres a et b .

13 Courbes paramétrées

- **Courbe[e1, e2, t, a, b]** : Courbe paramétrée de paramètre t variant dans l'intervalle $[a; b]$ l'abscisse d'un point étant expression e_1 et son ordonnée expression e_2 .

Exemple : $c = \text{Courbe}[2\cos(t), 2\sin(t), t, 0, 2\pi]$

- **Dérivée[c]** : Dérivée de la courbe c .
- **Valider c(3)** : retourne le point de la courbe c dont la position correspond à la valeur 3 du paramètre.

1 Transformations géométriques

- **Translation**[*A*, *v*] : Translaté du point *A* de vecteur *v*.
- **Translation**[*g*, *v*] : Translaté de la ligne *g* de vecteur *v*.
- **Translation**[*c*, *v*] : Translatée de la conique *c* de vecteur *v*.
- **Translation**[*f*, *v*] : Translatée de la courbe de la fonction *f* de vecteur *v*.
- **Translation**[*poly*, *v*] : Translation du polygone *poly* de vecteur *v*.
- **Translation**[*pic*, *v*] : Translation de l'image *pic* de vecteur *v*.
- **Translation**[*v*, *P*] : Donne au vecteur *v* le point *P* comme origine.
- **Rotation**[*A*, *phi*] : Tourne le point *A* d'un angle ϕ autour de l'origine.
- **Rotation**[*v*, *phi*] : Tourne le vecteur *v* d'un angle ϕ .
- **Rotation**[*g*, *phi*] : Tourne la ligne *g* d'un angle ϕ autour de l'origine.
- **Rotation**[*c*, *phi*] : Tourne la conique *c* d'un angle ϕ autour de l'origine.
- **Rotation**[*poly*, *phi*] : Tourne le polygone *poly* d'un angle ϕ autour de l'origine.
- **Rotation**[*pic*, *phi*] : Tourne l'image *pic* d'un angle ϕ autour de l'origine.
- **Rotation**[*A*, *phi*, *B*] : Tourne le point *A* d'un angle ϕ autour du point *B*.
- **Rotation**[*g*, *phi*, *B*] : Tourne la ligne *g* d'un angle ϕ autour du point *B*.
- **Rotation**[*c*, *phi*, *B*] : Tourne la conique *c* d'un angle ϕ autour du point *B*.
- **Rotation**[*poly*, *phi*, *B*] : Tourne le polygone *poly* d'un angle ϕ autour du point *B*.

- **Rotation**[*pic*, *phi*, *B*] : Tourne l'image *pic* d'un angle ϕ autour du point *B*.
- **Symétrie**[*A*, *B*] : Symétrique du point *A* par rapport au point *B*.
- **Symétrie**[*g*, *B*] : Symétrie de la ligne *g* par rapport au point *B*.
- **Symétrie**[*c*, *B*] : Symétrie de la conique *c* par rapport à *B*.
- **Symétrie**[*poly*, *B*] : Symétrie du polygone *poly* par rapport au point *B*.
- **Symétrie**[*pic*, *B*] : Symétrie de l'image *pic* par rapport à *B*.
- **Symétrie**[*A*, *h*] : Symétrie du point *A* par rapport à la ligne *h*.
- **Symétrie**[*g*, *h*] : Symétrie de la ligne *g* par rapport à la ligne *h*.
- **Symétrie**[*c*, *h*] : Symétrie de la conique *c* par rapport à *h*.
- **Symétrie**[*poly*, *h*] : Symétrie du polygone *poly* par rapport à la ligne *h*.
- **Symétrie**[*pic*, *h*] : Symétrie de l'image *pic* par rapport à *h*.
- **Homothétie**[*A*, *f*, *S*] : Image du point *A* par l'homothétie de centre *S*, de rapport *f*.
- **Homothétie**[*h*, *f*, *S*] : Image de la ligne *h* par l'homothétie de centre *S*, de rapport *f*.
- **Homothétie**[*c*, *f*, *S*] : Image de la conique *c* par l'homothétie de centre *S*, de rapport *f*.
- **Homothétie**[*poly*, *f*, *S*] : Image du polygone *poly* par l'homothétie de centre *S*, de rapport *f*.
- **Homothétie**[*pic*, *f*, *S*] : Transformée de l'image *pic* par l'homothétie de centre *S*, de rapport *f*.

2 Les coniques

- **Ellipse**[*F*, *G*, *a*] : Ellipse de foyers *F* et *G* et dont la longueur de l'axe principal vaut *a*. Note : Condition : $2a > Distance[F, G]$.

- **Ellipse**[*F*, *G*, *s*] : Ellipse de foyers *F* et *G* et dont la longueur de l'axe principal vaut $a = Longueur[s]$.
- **Hyperbole**[*F*, *G*, *a*] : Hyperbole de foyers *F* et *G* dont la longueur de l'axe principal vaut *a*. Note : Condition : $0 < 2a < Distance[F, G]$.
- **Hyperbole**[*F*, *G*, *s*] : Hyperbole avec foyers *F* et *G* dont la longueur de l'axe principal vaut $a = Longueur[s]$.
- **Parabole**[*F*, *g*] : Parabole de foyer *F* et de directrice *g*.
- **Conique**[*A*, *B*, *C*, *D*, *E*] : Conique passant par les cinq points *A*, *B*, *C*, *D*, et *E*. Note : Quatre de ces points ne doivent pas être alignés.

3 Les angles

- **Angle**[*v1*, *v2*] : Angle entre deux vecteurs *v1* et *v2* (entre 0 et 360°).
- **Angle**[*g*, *h*] : Angle entre les vecteurs directeurs de deux lignes *g* et *h* (entre 0 et 360°).
- **Angle**[*A*, *B*, *C*] : Angle \widehat{ABC} , délimité par *[AB]* et *[BC]* (entre 0 et 360°). *B* représente donc le sommet de l'angle.
- **Angle**[*A*, *B*, *alpha*] : Dessine un angle α à partir de *B* avec pour sommet *B*.
- **Angle**[*c*] : Angle de l'axe principal de la conique *c* par rapport à l'horizontale.
- **Angle**[*v*] : Angle entre l'axe (*Ox*) et le vecteur *v*.
- **Angle**[*A*] : Angle entre l'axe (*Ox*) et le vecteur \overrightarrow{OA} .
- **Angle**[*n*] : Convertit un nombre en un angle (le résultat entre 0 et 2π).
- **Angle**[*poly*] : Tous les angles intérieurs du polygone direct *poly*.

4 Quelques points en géométrie

- **A=(a,b)** : Définie et place le pt *A* de coordonnées (*a*; *b*).
- **Point**[*g*] : Point libre sur la ligne *g*.

- **Point[c]** : Point libre sur la conique c (par ex. cercle, ellipse, hyperbole).
- **Point[f]** : Point libre sur la courbe représentative de la fonction f .
- **Point[poly]** : Point libre sur la ligne polygonale frontière de poly.
- **Point[P, v]** : Image du point P dans la translation de vecteur v .
- **MilieuCentre[A,B]** : Milieu des points A et B .
- **MilieuCentre [s]** : Milieu du segment s .
- **CentreGravité[poly]** : Centre de gravité du polygone poly.
- **Intersection[g,h]** : Point d'intersection entre les lignes g et h .
- **Intersection[g,c]** : Tous les points d'intersection de la ligne g avec la conique c (max. 2).
- **Intersection[g, c, n]** : n ème point d'intersection de la ligne g avec la conique c .
- **Intersection[c1, c2]** : Tous les points d'intersection entre les coniques c_1 et c_2 (max. 4).
- **Intersection[c1, c2, n]** : n ème point d'intersection entre les coniques c_1 et c_2 .
- **Intersection[f1, f2]** : Tous les points d'intersection entre les courbes \mathcal{C}_{f_1} et \mathcal{C}_{f_2} des polynômes f_1 et f_2 .
- **Intersection[f1, f2, n]** : n ème point d'intersection entre les courbes \mathcal{C}_{f_1} et \mathcal{C}_{f_2} des polynômes f_1 et f_2 .
- **Intersection[f, g]** : Tous les points d'intersection entre la courbe \mathcal{C}_f du polynôme f et la ligne g .
- **Intersection[f, g, n]** : n ème point d'intersection entre la courbe \mathcal{C}_f du polynôme f et la ligne g .
- **Intersection[f, g, A]** : Premier point d'intersection entre \mathcal{C}_f et \mathcal{C}_g à partir de A (par la méthode de Newton).

- **Intersection[f, g, A]** : Premier point d'intersection entre \mathcal{C}_f et la ligne g à partir de A (par la méthode de Newton).

5 Les vecteurs

- **Vecteur[A,B]** : Vecteur AB .
- **Vecteur[A]** : Vecteur OA .
- **Direction[g]** : Vecteur directeur de la ligne g .
- **VecteurUnitaire[g]** : Vecteur directeur unitaire de la ligne g .
- **VecteurUnitaire [v]** : Vecteur unitaire de même direction et même sens que le vecteur donné v .
- **VecteurOrthogonal[g]** : Vecteur orthogonal à la ligne g .
- **VecteurOrthogonal[v]** : Vecteur orthogonal au vecteur v .
- **VecteurUnitaireOrthogonal[g]** : Vecteur orthogonal unitaire à la ligne g .
- **VecteurUnitaireOrthogonal[v]** : Vecteur orthogonal unitaire au vecteur v .
- **VecteurCourbure[A, f]** : Vecteur de courbure de la courbe représentative de la fonction f au point A .
- **VecteurCourbure[A, c]** : Vecteur de courbure de la courbe c au point A .
- **$u \cdot v$** : Produit scalaire $\vec{u} \cdot \vec{v}$.

6 Objets courants de géométrie

- **Segment[A, B]** : Segment $[AB]$.
- **Segment[A, a]** : Segment d'origine le point A et de longueur a .
- **DemiDroite[A, B]** : Demi-droite $[AB)$.
- **DemiDroite[A, v]** : Demi-droite d'origine A et de vecteur directeur v .
- **Polygone[A, B, C, ...]** : Polygone défini par les points donnés A, B, C, \dots

- **Polygone[A, B, n]** : Polygone régulier à n sommets (points A et B inclus)
- **Droite[A, B]** : Droite (AB) .
- **Droite[A, g]** : Droite passant par A et parallèle à la ligne g .
- **Droite[A, v]** : Droite passant par A et de vecteur directeur v .
- **Cercle[M, r]** : Cercle de centre M et de rayon r .
- **Cercle[M, s]** : Cercle de centre M et de rayon = $Longueur[s]$.
- **Cercle[M, A]** : Cercle de centre M passant par A .
- **Cercle[A, B, C]** : Cercle circonscrit à ABC (i.e. cercle passant par A, B et C).
- **DemiCercle[A, B]** : Demi-cercle de diamètre le segment $[AB]$.
- **ArcCercle[M, A, B]** : Arc de cercle de centre M entre les deux points A et B .
- **ArcCercleCirconscrit[A, B, C]** : Arc de cercle passant par les trois points A, B , et C .
- **Arc[c, A, B]** : Arc entre les deux points A et B de la conique c (Cercle ou Ellipse).
- **SecteurCirculaire[M, A, B]** : Secteur circulaire de centre M entre les deux points A et B .
- **SecteurCirculaireCirconscrit[A, B, C]** : Secteur circulaire passant par les trois points A, B , et C .

7 Droites particulières

- **axeX** : Axe des abscisses.
- **axeY** : Axe des ordonnées.
- **Perpendiculaire[point A,ligne g]** : Droite passant par A et perpendiculaire à la ligne g .
- **Perpendiculaire[point A,vecteur v]** : Droite passant par A et orthogonale au vecteur v .
- **Médiatrice[point A,point B]** : Médiatrice du segment $[AB]$.
- **Médiatrice[segment s]** : Médiatrice du segment s .
- **Bissectrice[A, B, C]** : Bissectrice de l'angle .

- **Bissectrice[g, h]** : Les deux bissectrices des lignes g et h .
- **Tangente[A, c]** : (Toutes les) tangentes à c passant par A .
- **Tangente[g, c]** : Toutes les tangentes à c parallèles à g .
- **Tangente[a, f]** : Tangente à \mathcal{C}_f en $x = a$.
- **Tangente[A, f]** : Tangente à \mathcal{C}_f en $x = x(A)$.
- **Tangente[A, c]** : Tangente à la courbe c au point A .
- **Asymptote[h]** : Les deux asymptotes à l'hyperbole h .
- **Directrice[p]** : Directrice de la parabole p .
- **Axes[c]** : Les deux axes de la conique c .
- **PremierAxe[c]** : Axe principal de la conique c .
- **SecondAxe[c]** : Axe secondaire de la conique c .
- **Polaire[A, c]** : Droite polaire de A par rapport à la conique c .
- **Diamètre[g, c]** : Diamètre de la conique c parallèle à g .
- **Diamètre[v, c]** : Diamètre de la conique c ayant pour vecteur directeur v .
- **Pente[g]** : Pente d'une ligne g . Note : Cette commande trace aussi le triangle permettant de visualiser la pente (quand j'avance de 1, je monte de « pente »).
- **Courbure[A,f]** : Courbure de la courbe représentative de f au point A .
- **Courbure[A, c]** : Courbure de la courbe c au point A .
- **Rayon[c]** : Rayon du cercle c .
- **Circonférence[c]** : Retourne la circonférence de la conique c (cercle ou ellipse).
- **Périmètre[poly]** : Périmètre du polygone $poly$.
- **Paramètre [p]** : Paramètre de la parabole p (distance entre la directrice et le foyer).
- **LongueurPremierAxe[c]** : Longueur du premier axe (axe principal) de la conique c .
- **LongueurSecondAxe[c]** : Longueur du second axe de la conique c .
- **ExcentricitéLinéaire[c]** : Excentricité linéaire de la conique (ellipse ou hyperbole) c (à savoir : la demi distance focale).
- **RapportColinéarité[A,B,C]** : Retourne le rapport de colinéarité de 3 points A, B , et C alignés, tel que $AC = \lambda \times AB$ ou $C = A + \lambda \times AB$
- **Birapport [A,B,C,D]** : Birapport de 4 points A, B, C , et D alignés, tel que :

$$\lambda = \frac{\text{RapportColinéarité}[C, B, A]}{\text{RapportColinéarité}[D, B, A]}$$

8 Les nombres de géométrie

- **Longueur[v]** : Norme du vecteur v .
- **Longueur[A]** : Distance OA .
- **Aire[A,B,C, ...]** : Aire du polygone défini par les points A, B , et $C \dots$
- **Aire[c]** : Aire délimitée par la conique c (cercle ou ellipse).
- **Distance[A,B]** : Distance AB .
- **Distance[A,g]** : Distance d'un point A à une ligne g .
- **Distance[g,h]** : Distance des lignes g et h .

1 Créer des listes (ou vecteurs).

- «**c(8,7,8,9)**» pour générer la liste des nombres 8, 7, 8 et 9.
- «**x=c(8,7,8,9)**» pour générer la liste des nombres 8, 7, 8 et 9 dans la variable x .
- «**x=scan()**» pour écrire dans la variable x , les valeurs à la suite en appuyant sur entrée entre chaque valeur. (2 fois entrée pour finir)
- «**1:10**» pour générer les nombres entiers de 1 à 10
- «**seq(1,10)**» pour générer une **séquence** des nombres entiers de 1 à 10, le pas est 1 par défaut.
- «**seq(1,10,2)**» pour générer une **séquence** des nombres de 1 à 10 avec un pas de 2.
- «**seq(1,10,length=20)**» pour générer une **séquence** de 8 nombres de 1 à 10.
- «**rep(2,6)**» pour **répéter** 6 fois le nombre 2. C'est identique à $c(2,2,2,2,2,2)$.
- «**1,rep(2,6),rep(3,4)**» est identique à $c(1,2,2,2,2,2,2,3,3,3,3)$.
- «**x=c(1,2,3); n=c(1,6,4); rep(x,n)**» On obtient ainsi un 1, six 2 et quatre 3. Très pratique les regroupements par classes. x et n doivent avoir le même nombre d'éléments.
- «**table(x)**» pour visualiser sous forme de tableau les regroupements par classe.

2 Fonctions élémentaires sur les listes.

Par exemple pour $x = c(8,7,8,9)$

- «**min(x)**» est la valeur **minimum** de x .
- «**max(x)**» est la valeur **maximum** de x .
- «**range(x)**» est la liste contenant le minimum et le maximum de x .
- «**sum(x)**» est la **somme** des valeurs de x .
- «**length(x)**» est le nombre de valeurs que contient x .
- «**mean(x)**» retourne la **moyenne** de x .
- «**var**» retourne la **variance** variance (non-biaisée) de x . Ce n'est pas la variance (de la population entière non estimée) que nous utilisons au lycée.

Sur vos calculatrice c'est la variance qui correspond à la touche S. Pour obtenir la variance utilisé au lycée il y a plusieurs méthodes

- $\text{mean}((a-\text{mean}(a))^2)$ (la définition de la variance)
- $\text{length}(x)*\text{sum}((x - \text{mean}(x))^2)$ (la propriété de la variance)

→ $\text{var}(x)*(\text{length}(x)-1)/\text{length}(x)$ lien entre les deux formules.

- «**sd**» retourne l'**écart type** de x (sd pour standard deviation).
- «**summary**» retourne un résumé des valeurs de x : le Minimum, 1^{er}Quartile, la médiane, le 3^equartile et la maximum.
- «**IQR**» retourne l'écart interquartile de x . (Interquartile Range)
- «**sort(x)**» trie x par ordre croissant.
- «**sort(x,decreasing=True)**» ou «**sort(x,de=T)**» trie x par ordre décroissant.
- «**cumsum(x)**» retourne les valeurs **cumulées** et **croissantes** de x
- «**median(x)**» retourne la valeur **médiane** de x
- «**quantile(x,0.25)**» retourne le 1^o quartile Q_1 de x .
- «**quantile(x,0.75)**» retourne le 3^o quartile Q_3 de x .
- «**quantile(x,0.1)**» retourne le 1^o décile D_1 de x .
- «**quantile(x,0.9)**» retourne le 9^o décile D_9 de x .

3 Opération sur les nombres

- «**round(x)**» **arrondi** x à l'entier
- «**round(x, 2)**» **arrondi** x au centième.
- «**round(x, -1)**» **arrondi** x aux dizaines
- «**ceiling(x)**» renvoie le plus petit entier supérieur à x .
- «**floor(x)**» renvoie le plus grand entier inférieur à x .
- «**trunc(x)**» renvoie la **troncature** de x .
- «**7%/2**» renvoie le **quotient** de la division euclidienne de 7 par 2.
- «**7%2**» renvoie le **reste** de la division euclidienne de 7 par 2.
Si $n\%2=0$ alors n est pair sinon n est impair.
- «**x<=30**» x est il plus petit ou égale à 30?
- «**sum(x<=30)**» Le **nombre d'élément** de x inférieur ou égale à 30.
- «**which(x<=30)**» renvoie la **positions** des valeurs de x inférieur ou égale à 30.
- «**factorial (7)**» **Factorielle** de 7, $7!=5040$
- «**choose (10,2)**» Nombre de **combinaisons** de p parmi n
for (n in 0:10) print(choose(n, k = 0:n)) pour avoir le triangle de pascal
- **combn(5,2)** génère toutes les combinaisons de 2 éléments parmi 5.

4 Aléatoire

- «**runif(1,2, 10)**» génère un **nombre aléatoire** entre 2 et 10. runif est la loi uniforme.

- «`runif(1000, 2, 10)`» génère 1000 nombres aléatoires entre 2 et 10.
- «`sample(1:10, 1)`» génère 1 nombre entier aléatoirement **sans répétition** entre 1 et 10.
- «`sample(1:10, 5)`» génère 5 nombres entiers aléatoirement **sans répétition** entre 1 et 10.
- «`sample(1:6, 50, replace=TRUE)`» ou «`sample(1:6, 50, re=T)`» génère 50 nombres entiers aléatoirement **avec répétition** entre 1 et 6.
Par exemple pour générer 50 lancers d'un dé équilibré à 6 faces.
- «`sample(1:6, 50, replace=T, p=c(1,1,1,1,4,4)/12)`» génère 50 lancers d'un dé à 6 faces pipé de probabilité $p_1 = p_2 = p_3 = p_4 = \frac{1}{12}$ et $p_5 = p_6 = \frac{4}{12} = \frac{1}{3}$.
On aurait pu utiliser $p = c(1/6, 1/6, 1/6, 1/6, 1/3, 1/3)$

5 Les lois de probabilité.

Le nom de chaque loi est précédé de r comme randomn, d pour la densité et p pour la fonction de répartition.

• La loi uniforme

- «`runif(x,min,max)`» donne x nombres qui suivent une la **loi uniforme** (voir le paragraphe précédent).
- «`dunif(x, min, max)`» est la densité de la loi uniforme.
- «`punif(x, min, max)`» est la fonction de répartition de la loi uniforme.

• La loi binomiale

- «`rbinom(x,n,p)`» : donne x nombres qui suivent La **loi Binomiale** de paramètre n et p .
- «`dbinom(x, min, max)`» est la densité de la loi binomiale.
- «`pbinom(x, min, max)`» est la fonction de répartition de la loi binomiale.
- Par exemple supposons que X suit la loi binomiale $B(10, 0.6)$.

```
dbinom(4,10,0.6) # donne P(X = 4)
sum(dbinom(4:8,10,0.6) # donne P(4<X<8)
```

• La loi exponentielle

- «`rexp(x,p)`» : donne x nombres qui suivent La **loi exponentielle** de paramètre p .
- «`dexp(x,p)`» est la densité de la loi exponentielle.
- «`pexp(x,p)`» est la fonction de répartition de la loi exponentielle.

• La loi normale

- «`rnorm(5000,0,1)`» génère 5000 nombres suivant la loi normale d'espérance 0 et d'écart type 1.
- «`dnorm(5000,0,1)`» est la densité de la loi normale.
- «`pnorm(5000,0,1)`» est la fonction de répartition de la loi normale.
- `x=rnorm(5000,0,1)`
`hist(x, nclass=50, col="blue", main="loi normale")`

6 les conditions

par exemple $x = c(50, 30, 10, 20, 60, 30, 20, 40)$.

- «`20%in%x` » 20 appartient il à x ?
- «`x<20 | x>40` » x est il inférieur à 20 **ou** supérieur à 40.
- «`x<40 & x>30`» x est il inférieur à 20 **et** supérieur à 40.
- «`!`» signifie le contraire
- «`x!=20`» x n'est pas égal à 20.

7 boucles

- `if (condition) { <instruction> } else { <instruction2> }`
`if (condition) {<instruction1>; <instruction2>} else {<instruction3>}`

Pour plus de clarté il est préférable de rédiger en verticale

```
if (condition) {
  <instruction1>
  <instruction2>}
else {
  <instruction3>
  <instruction4>}
```

- `for (i in 1:10) { instruction }` signifie pour i allant de 1 à 10 faire

Exemple :

```
for (i in 1:10) {print(i^2)}
```

- `while (i<=10) { instruction ... }` signifie tant que x est inférieur à 10 faire

Exemple :

```
i=0
while (i<=10) {print(i)}; i=i+1
```

8 Les graphiques

- `«hist(x, breaks»` crée un histogramme de la variable x avec un découpage breaks.
`hist(x, seq(0,2,0.2))` donnera un histogramme de x avec les classes 0,0.2,0.4,0.6,...2
- `«hist(x, nclass=10»` crée un histogramme de la variable x avec 10 classes.
- `«plot(function(x) x^2,col= 'red' , xlim=c(-2,10))»` trace la fonction carrée sur $[0;5]$.
- `«plot(function(x,y,type='l')»` Trace les points de coordonnées x et y relié par une ligne. x et y sont des vecteurs de même dimension
- `«abline(1,0.5)»` trace la droite $y = 0.5x + 1$.
- `«abline(h=0,v=2)»` trace la droite $y = 0$ et $x = 2$.
- `«segments(1,2, 3, 0)»` trace le segment $A(1;2) B(3;0)$
- On trouve les paramètres
 - `«add=TRUE»` ajoute au graphique déjà tracé
 - `«col='cyan'»` La couleur est verte
 - `«xlim=c(2,10)»` trace le graphique avec $x \in [2, 10]$.
 - `«ylim=c(2,10)»` trace le graphique avec $y \in [2, 10]$.
 - `«main='courbe'»` Le titre du graphique est 'titre'
 - `«sub='sous titre'»` sous titre du graphique
 - `«xlab='x'»` Label sur l'axe des abscisses.
 - `«ylab='y'»` Label sur l'axe des ordonnées.
 - `«pch=18»` type de point (de 0 à 25)
 - `«lty=3»` type de ligne.

- Pour sauver un graphique
`postscript('rplot.eps')`
`plot(x,y)`
`dev.off()`

10 Divers

- `options(max.print=10^5)` Pour pouvoir afficher à l'écran 10^5 nombres.
- `function` Pour créer une fonction.
`f=function(x) x^2` pour la fonction carrée.

9 Échange avec les autres logiciels

- `x <- read.table("data.dat")` permet de **lire** le fichier de données data.dat dans la variable x .
`write.table(x, file = "", sep = " ", eol = "\n", header=FALSE)` où file est le nom du fichier, sep est le séparateur de champ, eol le caractère de fin de ligne (\n correspond à un retour chariot) et headers indique si le fichier contient les noms des variables sur la 1^{re} ligne.
- `write.table(x, file = "data.dat")` permet de **sauvegarder** la variable x dans le fichier data.dat.

1 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

2 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.

3 Constantes

- `%pi` désigne $\pi \approx 3,14159$
- `%e` désigne $e = \exp(1) \approx 2,7183$
- `%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 $\left(\sum_{k=1}^n \frac{1}{k}\right) - \ln n$

4 Nombres réels

a 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**

b 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)

c 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 $\frac{\sin t}{\cos t}$
- `load(ntrig)` Permet de charger le package permettant de calculer des lignes pour des valeurs de x non usuelles.

Exemple : Pour calculer $\cos \frac{\pi}{10}$ on fait :

`load(ntrig)`

`cos(%pi/10)` et on obtient $\frac{\sqrt{\sqrt{5}+5}}{2\sqrt{2}}$.

5 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(funcs) $ 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 coefficient binomial $\binom{n}{p}$
- `random(n)` renvoie un entier naturel, choisi au hasard entre 0 et $n-1$ lorsque $n \in \mathbb{N}^*$

6 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

7 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 division 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`

8 Fonctions numériques

a 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)`

b 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,minu)` 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 $+\infty$

c dérivation

- `diff(f(x),x)` calcule la dérivée $f'(x)$
- `diff(f(x),x,2)` calcule $f''(x)$, dérivée seconde
- `define(fp(x),diff(f(x),x))` définit la fonction `fp` comme la fonction dérivée de `f`.

d 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])`

e intégrales

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

9 Suites

a Définition

- `u[n]:=1/n` est la suite définie par son terme général $u_n = \frac{1}{n}$
- `u[0]:1` on définit u_0 .
`u[n]:=1/(1+u[n-1])` est la suite définie par récurrence $u_{n+1} = \frac{1}{1+u_n}$
On ne peut pas écrire $u[n+1]:=1/(1+u[n])$

b Calcul sur les suites

- `makelist(u[k], k, 0, 5)` Affiche les 6 premiers termes de la suite u_n .
- `sum(1/u[n],n,0,20)` pour calculer $S = u_0 + u_1 + \dots + u_{20}$.
- `limit(u[n],n,inf)` pour calculer la convergence de la suite u_n lorsque u_n est définie par sa formule générale.

10 Équations

a résolution d'équations

Résolution 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 $[1, 2]$

b 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])`

c 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.

11 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 `]`. Une liste est numérotée à partir de 0. 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 3^e 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 $[l[1], m[1], l[2], m[2], l[3], m[3], \dots]$.
- `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 .

12 Sommation et produit

a somme finie

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

b produit fini

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

c 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(%)`

13 Vecteurs

- `u:[a,b,c]` Définit les coordonnées du vecteur u .
- `u+v` Renvoie les coordonnées de $u + v$
- `u . v` Renvoie le produit scalaire de u et de v (le point est précédé et suivi d'un espace)
- `load(vect)` Permet de charger le package pour calculer le produit vectoriel de deux vecteurs.
- `express(u ~ v)` : Renvoie le produit vectoriel de u et v .

14 Programmation

a 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))`

b structure conditionnelle

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

c structures itératives

Boucle **For** et affichage d'un tableau de valeurs de la fonction f :

- `f(x):=x^2;
for i from -2 thru 4 step 0.5 do (
print("f(",i,")=",f(i))
);`

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)`

15 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

16 Dénombrement

- `factorial (7)` Factorielle de 7, $7! = 5040$
- `binomial (10,2)` Nombre de combinaisons de p parmi n $\binom{10}{2} = 45$;
- `mod(11,2)` 11 modulo 2.