

# $\chi$ CAS : Xcas sur calculatrices Casio Graph 90 et 35+ei

Bernard.Parisse@univ-grenoble-alpes.fr

2019, 2022

## Table des matières

<b>1</b>	<b>Introduction et installation</b>	<b>2</b>
1.1	Calculatrice . . . . .	2
1.2	Émulateur . . . . .	2
<b>2</b>	<b>Premiers pas</b>	<b>3</b>
<b>3</b>	<b>Commandes usuelles de calcul formel</b>	<b>4</b>
3.1	Développer et factoriser . . . . .	4
3.2	Analyse . . . . .	5
3.3	Résoudre . . . . .	7
3.4	Arithmétique . . . . .	8
3.4.1	Entiers . . . . .	8
3.4.2	Polynômes . . . . .	10
3.5	Algèbre linéaire, vecteurs, matrices . . . . .	12
<b>4</b>	<b>Probabilités et statistiques</b>	<b>14</b>
4.1	Tirages aléatoires . . . . .	14
4.2	Lois de probabilités . . . . .	15
4.3	Statistiques descriptives 1-d . . . . .	15
4.4	Statistiques descriptives 2-d . . . . .	16
<b>5</b>	<b>Graphes</b>	<b>16</b>
<b>6</b>	<b>L'éditeur d'expressions</b>	<b>20</b>
<b>7</b>	<b>Sessions de calculs</b>	<b>21</b>
7.1	Edition de l'historique. . . . .	21
7.2	Variables . . . . .	22
7.3	Sauvegarde et échange de sessions avec Xcas. . . . .	22

<b>8</b>	<b>Programmation</b>	<b>22</b>
8.1	Utilisation en exemples . . . . .	22
8.2	Quelques exemples . . . . .	25
8.3	Commandes utilisables . . . . .	26
<b>9</b>	<b>Raccourcis claviers.</b>	<b>26</b>
9.1	KhiCAS 90 et 50 (version en 2 fichiers) . . . . .	26
9.2	KhiCAS (version courte en 1 fichier) . . . . .	27
<b>10</b>	<b>Remarques</b>	<b>28</b>
<b>11</b>	<b>Version plus complète (en 2 fichiers)</b>	<b>28</b>
11.1	MicroPython 1.12 . . . . .	29
11.2	Graphes 3d et 4d . . . . .	29
11.3	Géométrie interactive 2d/3d . . . . .	30
11.3.1	Modes, vue graphique et symbolique. . . . .	30
11.3.2	Curseurs . . . . .	32
11.3.3	Mesures et légendes . . . . .	32
11.3.4	Traces . . . . .	33
11.4	Tableur formel . . . . .	33
<b>12</b>	<b>Copyright et Remerciements</b>	<b>34</b>
<b>13</b>	<b>À propos du développement.</b>	<b>34</b>
13.1	Install rapide sous linux . . . . .	34
13.2	Source de giac . . . . .	35
13.3	Mise au point . . . . .	35

## Résumé

Ce document explique comment prendre en main et utiliser efficacement sur calculatrices Casio Graph 90+e et 35+eii le système de calcul formel  $\chi$ CAS, une version adaptée du logiciel Xcas pour ces calculatrices.

Ce document est interactif, vous pouvez modifier les commande et voir le résultat de l'exécution des commandes proposées en exemple en cliquant sur le bouton

(ou en validant avec la touche Entrée).

## 1 Introduction et installation

$\chi$ CAS est une version adaptée du logiciel Xcas pour calculatrices, ici de la marque Casio. Attention,  **$\chi$ CAS n'est pas compatible avec le mode examen sur les calculatrices Casio.** Si vous pensez comme moi qu'il s'agit d'une aberration, faites-le savoir à Casio France, plus il y aura de profs qui le demandent, plus il y aura de chances que Casio mette en place un mécanisme de signature de  $\chi$ CAS pour le rendre compatible avec le mode examen.

Sur les Casio Graph 90 couleur, il existe deux versions, une version light en un seul fichier, et une version plus complète en 2 fichiers. Cette version contient plus

de commandes (par exemple des commandes de géométrie), un moteur d’affichage 3d, un tableur formel et un interpréteur MicroPython un peu plus récent que celui de Casio, ainsi que des modules plus complets et nombreux que ceux du constructeur. Ces commandes supplémentaires sont décrites dans la section 11.

## 1.1 Calculatrice

Pour installer ou mettre à jour  $\chi$ CAS récupérez sur votre ordinateur

- Sur les Graph 90, la version plus complète de  $\chi$ CAS en deux fichiers : `khicas90.ac2` et `khicas90.g3a`  
ou si la version courte (sans géométrie, ...) vous suffit, le fichier `khicas.g3a`
- Sur Graph 35eii, `khicas.g1a`

Branchez le câble USB de la calculatrice, tapez F1 pour que la calculatrice soit considérée comme une clef USB et copiez le ou les fichiers `khicas90.g3a`, `khicas90.ac2` (Graph 90) ou `khicas.g3a` (Graph 90) ou `khicas.g1a` (Graph 35eii) sur la “clef”-calculatrice puis suivre la manipulation de votre ordinateur qui permet de débrancher une clef USB en toute sécurité. Le transfert s’effectue en quelques secondes.

## 1.2 Émulateur

Si vous testez sur l’émulateur (PC, Mac), depuis le menu principal de la calculatrice (MENU), allez à Mémoire, puis F3 (Import/Export), puis F1 (Import files), puis sélectionnez le fichier à transférez depuis le disque dur de votre ordinateur, tapez F1 pour sauvegarder sur la racine de la calculatrice, confirmez par F1 si vous effectuez une mise à jour. Le transfert est relativement long (plusieurs minutes) sur l’émulateur. Une fois le transfert terminé, une nouvelle icone (avec le flocon de Xcas ou le mot Xcas) apparait dans le menu principal de la calculatrice.

Attention, pour installer la version complète en 2 fichiers sur l’émulateur, il faut transférer `khicas90.882` et `emucas90.g3a`.

### Remarques :

- $\chi$ CAS n’est pas compatible avec le simulateur distribué par Casio sur clef USB. Il faut bien installer l’émulateur, avec une licence d’essai de 90 jours, les enseignants peuvent obtenir gratuitement une licence, la demander à Casio Education France.
- la version Windows de l’émulateur Casio tourne sous Linux avec wine, en utilisant la commande suivante après installation : `wine "C:\Program Files (x86)\CASIO\fx-CG Mana`

## 2 Premiers pas

Depuis le menu principal, déplacez le curseur jusqu’à l’icone de Xcas, puis tapez EXE. Lors de la première exécution, on vous demandera de choisir entre syntaxe Xcas et Python. Sélectionnez l’un des deux avec F1 et F6. Pour changer de syntaxe plus tard, utilisez shift-SETUP ou le menu F6.

N.B. : si vous utilisez la version longue de Xcas en 2 fichiers et que le message “Unable to load ram part” apparait, il faut mettre à jour l’OS de votre calculatrice (vers la version 3.30 ou plus).

Vous devez alors voir un historique de calculs (shell) vide dans lequel vous pouvez taper la plupart des commandes de calcul formel de Xcas.

Par exemple, tapez  $1/2+1/6$  puis EXE, vous devriez voir le résultat  $2/3$  s’afficher sur la ligne du dessous.

Astuce : si vous ne voyez pas un caractère sur le clavier de la calculatrice, tapez shift INS.

Vous pouvez recopier dans la ligne de commande une commande de l’historique en utilisant le curseur vers le haut ou vers le bas puis EXE, puis vous pouvez modifier la commande et l’exécuter. Par exemple, taper sur la touche curseur vers le haut, EXE et remplacez  $1/6$  par  $1/3$ .

Lors de la première utilisation, vous pouvez mettre à l’heure l’horloge de la calculatrice en tapant la commande `hh,mm→,`. Par exemple `13,17→,` pour 13h17.

Vous pouvez utiliser le résultat de la dernière commande avec la touche `Ans` de la calculatrice (taper sur shift puis `(-)`). Il vaut en général mieux définir une variable comme résultat d’une commande si on souhaite la réutiliser. Pour cela, on utilise une des deux instructions d’affectation :

- l’affectation vers la droite `=>` s’obtient avec la touche `→` de la calculatrice (au-dessus de `ON`), par exemple `2=>A` met 2 dans la variable A. Vous pouvez ensuite utiliser A dans un calcul, sa valeur sera remplacée par 2.
- l’affectation vers la gauche se fait en tapant sur la touche `shift-→` ou `shift-INS` selon la version de  $\chi$ CAS. Cela correspond à l’affectation en Python (`=`) ou en Xcas (`:` = mais `=` est accepté s’il n’y a pas de risque de confusion). Par exemple `A:=2` fait la même chose que `2=>A`.

**Remarque :** pour saisir un seul caractère alphabétique en majuscule, taper ALPHA en rouge puis la lettre. Pour saisir plusieurs caractères majuscules, taper SHIFT ALPHA. Pour saisir plusieurs caractères en minuscules, taper F5. F5 permet ensuite de basculer entre majuscules et minuscules.

Pour vous aider à saisir les commandes Xcas les plus utiles,  $\chi$ CAS dispose d’un catalogue d’une centaine de commandes, avec une courte description et le plus souvent un exemple d’exécution facile à recopier. Appuyez sur la touche F4 (`cmds`), choisissez une catégorie avec le curseur, par exemple `Algebre`, tapez EXE, puis choisissez une commande avec le curseur, par exemple `factor`. La touche F6 vous affiche une courte description de la commande, en général avec un exemple. En tapant sur F2, vous recopiez l’exemple en ligne de commande. S’il y a un deuxième exemple, on le recopie avec F3. Vous pouvez alors valider (EXE) ou modifier la commande et valider (EXE) pour factoriser un autre polynôme que celui donné en exemple.

Lorsqu’une commande renvoie une expression (i.e. un résultat qui n’est pas entier, ni réel approché, ni une fraction d’entiers), celle-ci est affichée en écriture naturelle (affichage 2-d). Vous pouvez faire défiler l’affichage avec les touches du curseur lorsque l’expression est grande. Tapez sur EXIT pour revenir au shell.

Maintenant essayez de taper la commande `plot(sin(x))`. Indication : taper F4 (CATALOG), puis sélectionner `Courbes`.

Lorsqu'une commande renvoie un graphe, celui-ci est affiché. Vous pouvez modifier la fenêtre graphique d'affichage avec les touches + ou - (zoom in ou out, utiliser la touche (-) pour faire un zoom out partiel selon l'axe  $Oy$ ), les touches du curseur, orthonormaliser le repère (touche /) ou faire une recherche automatique de l'échelle (autoscale touche \*). Pour enlever ou remettre les axes et graduations, tapez sur SIN. Tapez sur EXIT pour revenir au shell.

Vous pouvez effacer l'historique des calculs et les variables pour commencer un nouvel exercice : depuis le menu Fich,Cfg (F6) sélectionnez Effacer l'historique.

Un certain nombre de raccourcis claviers sont prévus pour faciliter les tâches les plus courantes : par exemple F1 (alg) et F2 (calc) ouvrent un menu avec des commandes d'algèbre et d'analyse (calculus en anglais), cf. la section 9 pour une liste complète. L'appui sur ON/AC efface la ligne courante, et si celle-ci est vide propose d'effacer l'historique des calculs.

Pour quitter  $\chi$ CAS, appuyez sur la touche MENU. Lorsque vous lancez une autre application, les variables et l'historique des calculs sont sauvegardés, ils seront restaurés lorsque vous reviendrez dans  $\chi$ CAS. La première sauvegarde prend un peu de temps, en particulier sur l'émulateur, ensuite les sauvegardes sont plus rapides. Vous pouvez sauvegarder plusieurs sessions distinctes (historique et programmes éventuels) depuis le menu F6 de KhiCAS, et les restaurer ultérieurement.

## 3 Commandes usuelles de calcul formel

### 3.1 Développer et factoriser

Depuis le catalogue, sélectionner le sous-menu Algebre ou tapez F1 :

- `factor` : factorisation. Raccourci clavier  $\Rightarrow *$  (touche  $\rightarrow$  puis \*), par exemple  $x^4-1 \Rightarrow *$

$$(x-1)(x+1)(x^2+1)$$

. Utiliser `cfactor` pour factoriser sur  $\mathbb{C}$ .

- `partfrac` : développement d'un polynôme ou décomposition en éléments simples pour une fraction. Raccourci clavier  $\Rightarrow +$  (touche  $\rightarrow$  puis +), par exemple  $(x+1)^4 \Rightarrow +$

$$x^4 + 4x^3 + 6x^2 + 4x + 1$$

ou  $1/(x^4-1) \Rightarrow +$

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

- `simplify` : essai de simplifier une expression. Raccourci clavier  $\Rightarrow /$  (touche  $\rightarrow$  puis /), par exemple  $\sin(3x)/\sin(x) \Rightarrow /$

$$2\cos(2x) + 1$$

- `ratnormal` : développer une expression, écrire une fraction sous forme irréductible.

### 3.2 Analyse

Depuis le catalogue, sélectionner le sous-menu *Analyse*, ou tapez F2

- `diff` : dérivation. On peut aussi utiliser la notation ' (raccourci clavier F3) pour dériver par rapport à  $x$ , ainsi  
`diff(sin(x), x)`

$$\cos x$$

et `sin(x)'`

$$\cos x$$

sont équivalents. Pour dériver plusieurs fois, ajouter le nombre de dérivations par exemple `diff(sin(x^2), x, 3)`

$$-8x^3 \cos(x^2) - 12x \sin(x^2)$$

- `integrate` : primitive si 1 ou 2 arguments, par exemple  
`integrate(sin(x))`

$$-\cos x$$

ou `integrate(1/(t^4-1), t)`

$$\frac{\ln|t-1|}{4} - \frac{\ln|t+1|}{4} - \frac{\arctan t}{2}$$

pour  $\int \frac{1}{t^4-1} dt$

Calcul d'intégrale définie si 4 arguments, par exemple `integrate(sin(x)^4, x, 0, pi)`

$$\frac{3}{8}\pi$$

pour  $\int_0^\pi \sin(x)^4 dx$ . Mettre une des bornes d'intégration sous forme approchée si on souhaite un calcul approché d'intégrale définie, par exemple

`integrate(sin(x)^4, x, 0.0, pi)`

$$1.1780972451$$

Raccourci clavier shift F3.

- `limit` : limite d'une expression. Exemple `limit((cos(x)-1)/x^2, x=0)`

$$-\frac{1}{2}$$

Pour une limite à droite [resp. à gauche], ajouter 1 [resp. -1] en dernier argument,  
par exemple

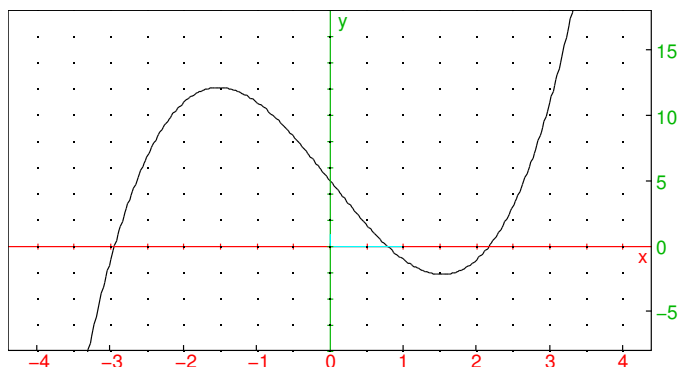
`limit (1/x, x=0, 1)`

$+\infty$

- `tabvar` : tableau de variations d'une expression. Par exemple `tabvar (x^3-7x+5)`

$$\left[ \begin{array}{cccccccccccc} & x & -\infty & "" & -\frac{\sqrt{21}}{3} & "" & 0 & "" & \frac{\sqrt{21}}{3} & "" & +\infty \\ y' = \left(x + \frac{\sqrt{21}}{3}\right) (3x - \sqrt{21}) & +\infty & "+" & 0 & "-" & -7 & "-" & 0 & "+" & +\infty \\ y = x^3 - 7x + 5 & -\infty & "" & \frac{14\sqrt{21}+45}{9} & "" & 5 & "" & \frac{-14\sqrt{21}+45}{9} & "" & +\infty \\ y'' & -\infty & "- ()" & -2\sqrt{21} & "- ()" & 0 & "+ ()" & 2\sqrt{21} & "+ ()" & +\infty \end{array} \right]$$

on peut vérifier avec le graphe `plot (x^3-7x+5, x, -4, 4)`



- `taylor et series` : développement de Taylor (ou développement limité ou asymptotique). Par exemple

`taylor (sin (x) , x=0, 5)`

$$x - \frac{x^3}{6} + \frac{x^5}{120} + x^6 \text{order\_size}(x)$$

- `sum` : somme discrète. Par exemple

`sum (k^2, k, 1, n)`

$$\frac{2(n+1)^3 - 3(n+1)^2 + n + 1}{6}$$

calcule  $\sum_{k=1}^n k^2$ ,

`sum (k^2, k, 1, n) => *`

$$\frac{1}{6}n(n+1)(2n+1)$$

calcule la somme et l'écrit sous forme factorisée.

Raccourci clavier ALPHA F3.

### 3.3 Résoudre

Depuis le catalogue, sélectionner le sous-menu `Resoudre`.

- `solve` permet de résoudre de manière exacte une équation (se ramenant à une équation polynomiale). Il faut préciser la variable si ce n'est pas  $x$  par exemple `solve(t^2-1=0,t)`

$$[-1, 1]$$

. Si la variable est  $x$ , on peut écrire l'expression ou l'équation suivie de 2 appuis sur la touche  $\rightarrow$ , par exemple `x^2-1=>`

Si la recherche exacte échoue, la commande `fsolve` permet de faire une résolution approchée, soit par une méthode itérative en partant d'une valeur initiale `fsolve(cos(x)=x,x=0.0)`

$$0.739085133215$$

, soit par dichotomie `fsolve(cos(x)=x,x=0..1)`

$$[0.739085133215]$$

.  
Pour avoir des solutions complexes, utiliser `csolve`.

On peut faire des hypothèses sur la variable que l'on cherche, par exemple `assume(m>1)`

$$m$$

puis `solve(m^2-4=0,m)`

$$[2]$$

- `solve` permet aussi de résoudre des systèmes polynomiaux simples, on donne en 1er argument la liste des équations, en 2ème argument la liste des variables. Par exemple intersection d'un cercle et d'une droite `solve([x^2+y^2+2y=3,x+y=1],[x,y])`

$$[[0, 1], [2, -1]]$$

- `linsolve` permet de résoudre des systèmes linéaires. On lui passe la liste des équations et la liste des variables (par convention une expression équivaut à l'équation `expression=0`). Par exemple `linsolve([x+2y=3,x-y=7],[x,y])`

$$\left[\frac{17}{3}, -\frac{4}{3}\right]$$

`linsolve` renvoie la solution générale du système (y compris si la solution n'est pas unique).



- `desolve` permet de résoudre de manière exacte certaines équations différentielles, par exemple pour résoudre  $y' = 2y$ , on tape `desolve (y'=2y)`.  
Un exemple où on indique une condition initiale, la variable indépendante et la fonction inconnue :  
`desolve ([y'=2y, y(0)=1], x, y)` Utiliser `odesolve` pour une résolution approchée et `plotode` pour une représentation graphique de solution calculée de manière approchée.
- `rsolve` permet de résoudre de manière exacte certaines relations de récurrences  $u_{n+1} = f(u_n, \dots)$ , par exemple les suites arithmético-géométriques, par exemple  $u_{n+1} = 2u_n + 3, u_0 = 1$   
`rsolve (u(n+1)=2*u(n)+3, u(n), u(0)=1)`

$$[4 \cdot 2^n - 3]$$

### 3.4 Arithmétique

Lorsque cela est nécessaire, on distingue l'arithmétique des entiers de celle des polynômes par l'existence du préfixe `i` (comme `integer`) dans un nom de commande, par exemple `ifactor` factorise un entier (pas trop grand) alors que `factor` factorise un polynôme (et `cfactor` factorise un polynôme sur les complexes). Certaines commandes fonctionnent à la fois pour les entiers et les polynômes, par exemple `gcd` et `lcm`.

#### 3.4.1 Entiers

Depuis le catalogue, sélectionner le sous-menu `Arithmetic, Crypto`

- `iquo(a,b)`, `irem(a,b)` quotient et reste de la division euclidienne de deux entiers.

`iquo(23,13), irem(23,13)`

1, 10

- `isprime(n)` teste si  $n$  est un nombre premier. Le test est probabiliste pour de grandes valeurs de  $n$ .  
`isprime(2^64+1)`

faux

- `ifactor(n)` factorise un entier pas trop grand (les algorithmes utilisés se limitent à la division et Pollard- $\rho$ , il n'y avait pas de place pour le crible quadratique).  
Par exemple  
`ifactor(2^64+1)`

67280421310721 · 274177

Raccourci clavier touches  $\rightarrow$  puis  $*$  ( $\Rightarrow *$ )

– `gcd(a,b), lcm(a,b)` PGCD et PPCM de deux entiers ou de deux polynômes

`gcd(25,15), lcm(25,15)`

5,75

`gcd(x^3-1,x^2-1), lcm(x^3-1,x^2-1)`

$x-1, (x^2+x+1)(x^2-1)$

– `iegcd(a,b)` renvoie 3 entiers  $u, v, d$  tels que  $au + bv = d$  où  $d$  est le PGCD de  $a$  et  $b$ , avec  $|u| < |b|$  et  $|v| < |a|$ .

`u,v,d:=iegcd(23,13); 23u+13v`

$[4, -7, 1], 1$

– `ichinrem([a,m],[b,n])` lorsque cela est possible, renvoie  $c$  tel que  $c = a \pmod{m}$  et  $c = b \pmod{n}$  (si  $m$  et  $n$  sont premiers entre eux,  $c$  existe).

`c,n:=ichinrem([1,23],[2,13]); irem(c,23);  
irem(c,13)`

$[93, 299], 1, 2$

– `powmod(a,n,m)` calcule  $a^n \pmod{m}$  par l’algorithme de la puissance rapide modulaire.

`powmod(7,22,23)`

1

Les commandes `asc` et `char` permettent de convertir une chaîne de caractères en liste d’entiers (entre 0 et 255) et réciproquement, ce qui permet de faire facilement de la cryptographie avec des messages sous forme de chaînes de caractères.

### 3.4.2 Polynômes

Depuis le catalogue, sélectionner le sous-menu Polynômes. La variable est par défaut  $x$ , sinon il faut la spécifier en général en dernier argument, par exemple `degree(x^2*y)` ou `degree(x^2*y,x)` renvoient 2, alors que `degree(x^2*y,y)` renvoie 1

– `coeff(P,n)` coefficient de  $x^n$  dans  $P$ , `lcoeff(P)` coefficient dominant de  $P$ , par exemple

`P:=x^3+3x; coeff(P,1); lcoeff(P)`

$x^3 + 3x, 3, 1$

– `degree(P)` degré du polynôme  $P$ .  
`degree(x^3)`

3

- `quo(P,Q), rem(P,Q)` quotient et reste de la division euclidienne de  $P$  par  $Q$   
 $P:=x^3+7x-5; Q:=x^2+x; quo(P,Q); rem(P,Q)$

$$x^3 + 7x - 5, x^2 + x, x - 1, 8x - 5$$

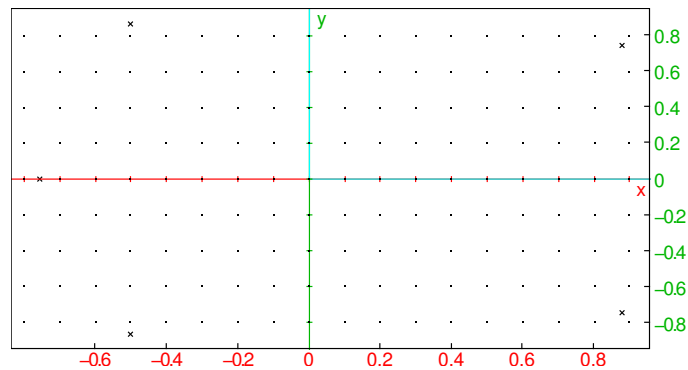
- `proot(P)` : racines approchées de  $P$  (réelles et complexes)

`proot(x^5+x+1)`

$[-0.754877666247, -0.5 - 0.866025403784i, -0.5 + 0.866025403784i, 0.877438833123 - 0.74486176662i, 0.877438833123 + 0.74486176662i]$

Représentation graphique :

`point(proot(x^5+x+1))`



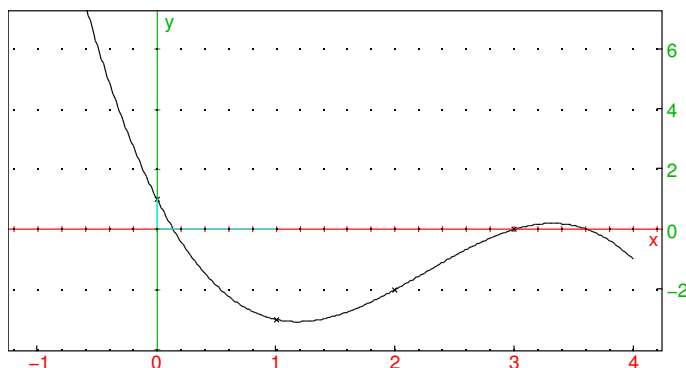
- `interp(X,Y)` : pour deux listes de même taille, polynôme d'interpolation passant par les points  $(X_i, Y_i)$ .

$X, Y := [0, 1, 2, 3], [1, -3, -2, 0]; P := \text{interp}(X, Y) \Rightarrow +$

$$[0, 1, 2, 3], [1, -3, -2, 0], \frac{-4x^3 + 27x^2 - 47x + 6}{6}$$

Représentation graphique

`scatterplot(X,Y); plot(P,x,-1,4)`



- resultant (P,Q) : résultant des polynômes P et Q  
 $P:=x^3+7x-5$ ;  $Q:=x^2+x$ ; resultant (P,Q)

$$x^3 + 7x - 5, x^2 + x, 65$$

- hermite (x,n) : n-ième polynôme de Hermite, orthogonal pour la densité  $e^{-x^2}dx$  sur  $\mathbb{R}$
- laguerre (x,n,a) : n-ième polynôme de Laguerre,
- legendre (x,n) : n-ième polynôme de Legendre, orthogonal pour la densité  $dx$  sur  $[-1, 1]$
- tchebyshev1 (n) et tchebyshev2 (n) polynômes de Tchebyshev de 1ère et 2ème espèce définis par :

$$T_n(\cos(x)) = \cos(nx), \quad U_n(\cos(x)) \sin(x) = \sin((n+1)x)$$

### 3.5 Algèbre linéaire, vecteurs, matrices

Xcas ne fait pas de différence entre vecteur et liste. Par exemple pour faire le produit scalaire de deux vecteurs, on peut saisir :

```
v:=[1,2]; w:=[3,4]

dot (v,w)
```

11

Pour saisir une matrice élément par élément, taper sur shift-MATR et sélectionner matrix ou F6, puis 10 (éditer matrice). Vous pouvez ensuite créer une nouvelle matrice ou éditer une matrice existante parmi la liste de variables proposées. Notez que la touche , permet d'insérer une ligne ou une colonne, alors que la touche d'effacement DEL efface la ligne ou la colonne de la sélection (tapez UNDO si vous avez effacé par erreur une ligne ou colonne). Pour de petites matrices, vous pouvez aussi entrer en ligne de commandes une liste de listes de même taille. Par exemple pour définir la matrice

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

12

`A:=[ [1,2], [3,4] ]`

ou `[ [1,2], [3,4] ]=>A`

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Il est fortement conseillé de stocker les matrices dans des variables pour éviter de les saisir plusieurs fois.

Pour entrer une matrice dont les coefficients sont donnés par une formule définissant l'élément  $a_{ij}$  en fonction de la ligne  $i$  et colonne  $j$ , on peut utiliser la commande `matrix(`. Par exemple

`matrix(2,2,(j,k)->1/(j+k+1))`

$$\begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{3} \end{pmatrix}$$

renvoie la matrice dont le coefficient ligne  $j$  et colonne  $k$  vaut  $\frac{1}{j+k+1}$  (attention les indices commencent à 0). Pour obtenir la commande `matrix(`, taper F4 (cmds) puis 17 Matrices ou utiliser le raccourci shift-MATR, EXE, AC/ON.

La matrice identité de taille  $n$  est renvoyée par la commande `idn(n)` (F4 17), alors que `ranm(n,m,loi,[parametres])` (menu shift-MATR ou F4 17) renvoie une matrice à coefficients aléatoires de taille  $n,m$ . Par exemple

`U:=ranm(4,4,uniformd,0,1)`

$$\begin{pmatrix} 0.930321054533 & 0.0115270754322 & 0.0881637986749 & 0.536643749103 \\ 0.360248812009 & 0.630750506185 & 0.0543795586564 & 0.819389336277 \\ 0.251521021593 & 0.0686232172884 & 0.178203014191 & 0.346399624366 \\ 0.852082391735 & 0.614078260958 & 0.714221911505 & 0.784336711746 \end{pmatrix}$$

`N:=ranm(4,4,normald,0,1)`

$$\begin{pmatrix} -1.15067689004 & -0.57241508091 & -0.751538700822 & 0.989770363266 \\ -0.00575141421859 & 0.902209004052 & 0.455987131429 & -1.85967614286 \\ -0.243668831329 & -1.61539092837 & 0.558232005133 & -0.159613435564 \\ 1.35165685556 & -0.16425534864 & 1.27150836747 & 0.245690624229 \end{pmatrix}$$

Pour exécuter une commande sur des matrices, s'il s'agit d'arithmétique de base (+, -, \* inverse), on utilise les opérations au clavier. Pour les autres commandes, depuis le catalogue F4, sélectionner le sous-menu Matrices

- `eigenvals(A)`

$$\frac{\sqrt{33}+5}{2}, \frac{-\sqrt{33}+5}{2}$$

`eigenvects(A)`

$$\begin{bmatrix} \frac{\sqrt{33}-3}{6} & \frac{-\sqrt{33}-3}{6} \end{bmatrix}$$

renvoient les valeurs propres et vecteurs propres d'une matrice carrée  $A$ .

– `P,D:=jordan(A)`

$$\begin{bmatrix} \sqrt{33}-3 & -\sqrt{33}-3 \\ 6 & 6 \end{bmatrix}, \begin{bmatrix} \frac{\sqrt{33}+5}{2} & 0 \\ 0 & \frac{-\sqrt{33}+5}{2} \end{bmatrix}$$

calcule la forme normale de Jordan d’une matrice  $A$  (à coefficients exacts) et renvoie les matrices  $P$  et  $D$  telles que  $P^{-1}AP = D$ , avec  $D$  triangulaire supérieure (diagonale si  $A$  est diagonalisable)

– `Ak:=matpow(A,k)`

$$\begin{bmatrix} \frac{1}{66}(\sqrt{33}-3)\left(\frac{\sqrt{33}+5}{2}\right)^k \sqrt{33} - \frac{1}{66}(-\sqrt{33}-3)\left(\frac{-\sqrt{33}+5}{2}\right)^k \sqrt{33} & \frac{1}{132}(\sqrt{33}-3)\left(\frac{\sqrt{33}+5}{2}\right)^k(\sqrt{33}+11) - \frac{1}{132}(-\sqrt{33}-3)\left(\frac{-\sqrt{33}+5}{2}\right)^k(\sqrt{33}+11) \\ \frac{6}{66}\left(\frac{\sqrt{33}+5}{2}\right)^k \sqrt{33} - \frac{6}{66}\left(\frac{-\sqrt{33}+5}{2}\right)^k \sqrt{33} & \frac{6}{132}\left(\frac{\sqrt{33}+5}{2}\right)^k(\sqrt{33}+11) - \frac{6}{132}\left(\frac{-\sqrt{33}+5}{2}\right)^k(\sqrt{33}+11) \end{bmatrix}$$

calcule la puissance  $k$ -ième d’une matrice  $A$  avec  $k$  une variable formelle.

– `rref` effectue la réduction sous forme échelonnée d’une matrice  $A$  (pivot de Gauss)

– `lu` calcule la décomposition  $LU$  d’une matrice  $A$  et renvoie une permutation de matrice  $P$  et deux matrices  $L$  triangulaire inférieure et  $U$  triangulaire supérieure telles que  $PA = LU$ . Le résultat de la commande

`P,L,U:=lu(A)`

$$[0,1], \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 2 \\ 0 & -2 \end{bmatrix}$$

peut être passé en argument à la commande `linsolve(P,L,U,v)`

$$\left[0, \frac{1}{2}\right]$$

pour résoudre un système  $Ax = b$  de matrice  $A$  en résolvant deux systèmes triangulaires (calcul en  $O(n^2)$  au lieu de  $O(n^3)$ ).

– `qr` calcule la décomposition  $QR$  d’une matrice  $A$  et renvoie deux matrices  $Q$  orthogonale et  $R$  triangulaire supérieure telles que  $A = QR$ .

– `svd(A)` calcule la factorisation en valeurs singulières d’une matrice  $A$ , et renvoie  $U$  orthogonale,  $S$  vecteur des valeurs singulières,  $Q$  orthogonale tels que  $A=U*\text{diag}(S)*\text{tran}(Q)$ . Le rapport de la plus grande valeur singulière de  $S$  sur la plus petite donne le nombre de condition de  $A$  relativement à la norme euclidienne, plus ce nombre est grand, plus on perd en précision en résolvant un système  $Ax = b$  lorsque  $b$  n’est pas connu exactement.

## 4 Probabilités et statistiques

### 4.1 Tirages aléatoires

Depuis le catalogue F4, sélectionner le sous-menu Probabilites puis sélectionnez `rand()`

0.833197717089

(réel selon la loi uniforme dans  $[0, 1]$ ) ou  
`n:=6;; randint(n)`

"Done", 4

(entier entre 1 et  $n$ ). De nombreuses autres fonctions aléatoires existent, avec comme préfixe `rand`, suivi par le nom de la loi, par exemple `randbinomial(n,p)` renvoie un entier aléatoire selon la loi binomiale de paramètres  $n, p$ . Pour créer un vecteur ou une matrice aléatoire, utiliser la commande `ranv` ou `ranm` (menu F4 Matrice ou shift-MATR), par exemple pour un vecteur de 10 composantes selon la loi normale centrée réduite

`ranv(10,normald,0,1)`

`[1.09936454094, -2.28261976775, 0.652261860753, -0.690651824321, -0.323190436625, 0.157460267236, -0.324`

## 4.2 Lois de probabilités

Depuis F4, sélectionner le sous-menu Probabilites. Les lois proposées dans le catalogue sont la loi binomiale, la loi normale, la loi exponentielle et la loi uniforme. D'autres lois sont disponibles mais il n'y a pas assez de place pour les documenter : `chisquared`, `geometric`, `multinomial`, `studentd`, `fisherd`, `poisson`.

Pour obtenir la distribution cumulée d'une loi, on saisit le nom de la loi et le suffixe `_cdf` (sélectionner `cdf` dans le catalogue sous-menu Probabilités et taper F1). Pour obtenir la distribution cumulée inverse, on saisit le nom de la loi et le suffixe `_icdf` (sélectionner `cdf` dans le catalogue sous-menu Probabilités et taper F2).

Exemple : calcul de l'intervalle centré  $I$  pour la loi normale de moyenne 5000 et d'écart-type 200 tel que la probabilité d'être en-dehors de  $I$  soit de 5% :

`M:=5000; S:=200; normald_icdf(M,S,0.025);normald_icdf`  
`(M,S,0.975)`

5000, 200, 4608.00720309, 5391.99279691

## 4.3 Statistiques descriptives 1-d

Ces fonctions agissent sur des listes

`l:=[9,11,6,13,17,10]`

Depuis le catalogue, sélectionner le sous-menu Statistiques.

– `mean(l)`

11

: moyenne arithmétique d'une liste

– `stddev(l)`

$$\frac{\sqrt{105}}{3}$$

: écart-type d’une liste

Utiliser

`stddevp(l)`

$$\sqrt{14}$$

pour avoir un estimateur non biaisé de l’écart-type d’une population dont `l` est un échantillon

– `median(l)`

10.0

, `quartile1(l)`

9.0

,  
`quartile3(l)`

13.0

renvoient respectivement la médiane, le 1er et 3ème quartiles d’une liste

Pour les statistiques 1-d de listes avec effectifs, on remplace `l` par deux listes de même longueur, la 1ère liste est la liste des valeurs de la série statistique, la 2ème liste est la liste des effectifs. Voir aussi les commandes du menu Graphique histogram et `barplot`.

## 4.4 Statistiques descriptives 2-d

Depuis le catalogue, sélectionner le sous-menu Statistiques.

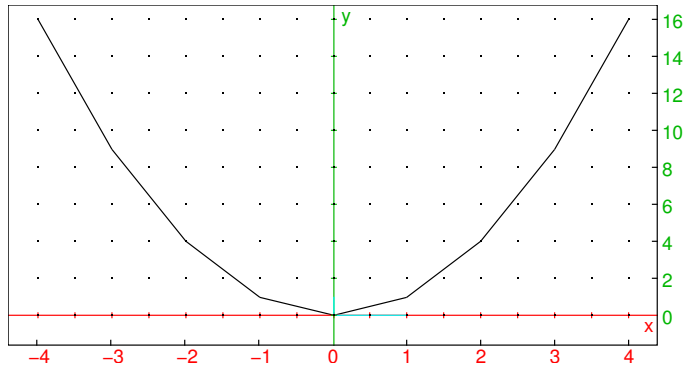
- `correlation(X, Y)` calcule la corrélation entre 2 listes de même taille.
- `covariance(X, Y)` calcule la covariance entre 2 listes de même taille.
- les commandes de suffixe `_regression(X, Y)` calculent des ajustements par régression au sens des moindres carrés, par exemple `linear_regression(X, Y)` renvoie les coefficients  $m, p$  de la droite de régression linéaire  $y = mx + p$ .
- `linear_regression_plot(X, Y)` (respectivement toutes les commandes de suffixe `_regression_plot`) trace une droite (respectivement une courbe) d’ajustement des données contenues dans les listes `X, Y` de même taille. Ces commandes affichent de plus le coefficient  $R^2$  qui permet de quantifier la qualité de l’ajustement (plus  $R^2$  est proche de 1, meilleur est l’ajustement).



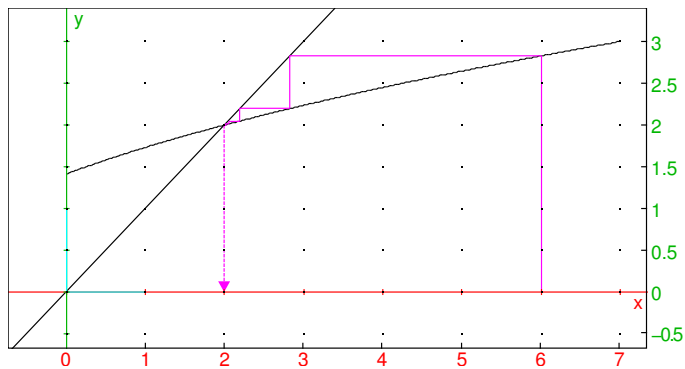
## 5 Graphes

Depuis le catalogue F4, sélectionner le sous-menu Graphiques (7).

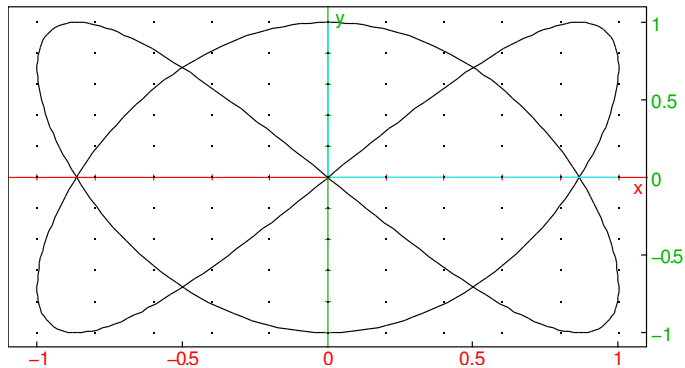
- `plot(f(x), x=a..b)` graphe de  $f(x)$  pour  $x \in [a, b]$ . On peut spécifier un pas de discrétisation avec `xstep=`, par exemple `plot(x^2, x=-4..4, xstep=1)`



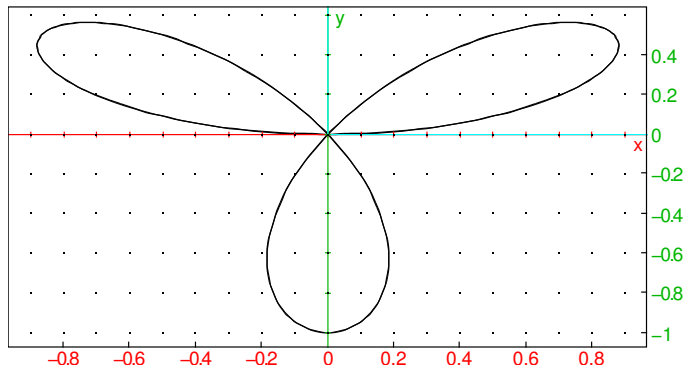
- `plotseq(f(x), x=[u0, a, b])` graphe “en toile d’araignée” de la suite récurrente  $u_{n+1} = f(u_n)$  de premier terme  $u_0$  donné. Par exemple si  $u_{n+1} = \sqrt{2 + u_n}$ ,  $u_0 = 6$  avec une représentation sur  $[0, 7]$   
`plotseq(sqrt(2+x), x=[6, 0, 7])`



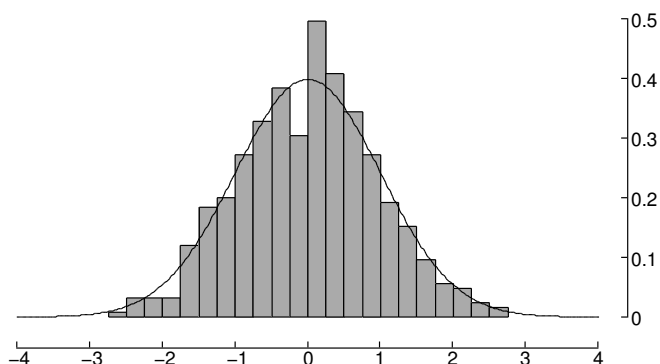
- `plotparam([x(t), y(t)], t=tm..tM)` courbe en paramétriques  $(x(t), y(t))$  pour  $t \in [t_m, t_M]$ . On peut spécifier un pas de discrétisation avec `tstep=` par exemple  
`plotparam([sin(2t), cos(3t)], t, 0, 2*pi)`



- `plotpolar(r(theta), theta=a..b)` courbe en polaires  $r(\theta)$  pour  $\theta \in [a, b]$ , par exemple  
`plotpolar(sin(3*theta), theta, 0, 2*pi)`

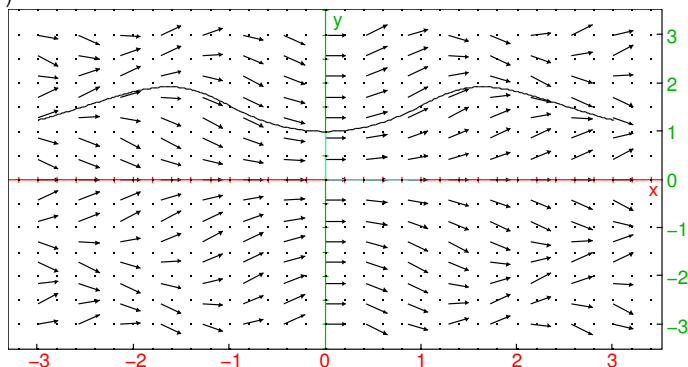


- `plotlist(l)` pour une liste `l`, trace la ligne polygonale reliant les points de coordonnées  $(i, l_i)$  (indice  $i$  commençant à 0).  
`plotlist([X1, Y1], [X2, Y2], ...)` trace la ligne polygonale reliant les points de coordonnées  $(X_i, Y_i)$
- `scatterplot(X, Y)`, `polygonscatterplot(X, Y)` pour 2 listes `X, Y` de même taille, trace un nuage de points ou une ligne polygonale reliant les points de coordonnées  $(X_i, Y_i)$
- `histogram(l, class_min, class_size)` trace l'histogramme des données de la liste `l` avec comme largeur de classe `class_size` en commençant à `class_min`. Par exemple, on peut tester la qualité du générateur aléatoire avec  
`l:=ranv(500, normald, 0, 1); histogram(l, -4, 0.25); plot(normald(x), x, -4, 4)`



- `plotcontour(f(x,y), [x=xmin..xmax,y=ymin..ymax], [l0,l1,...])`  
trace les courbes de niveaux  $f(x,y) = l_0, f(x,y) = l_1, \dots$
- (Graph 90) `plotfield(f(t,y), [t=tmin..tmax,y=ymin..ymax])`  
trace le champ des tangentes à l'équation différentielle  $y' = f(t,y)$ . On peut ajouter en dernier paramètre optionnel `,plotode=[t0,y0]` pour tracer simultanément la solution passant par la condition initiale  $y(t_0) = y_0$ . Exemple  $y' = \sin(ty)$  sur l'intervalle  $[-3,3]$  en temps et  $[-2,2]$  en  $y$

```
plotfield(sin(t*y), [t=-3..3,y=-3..3],plotode=[0,1])
```

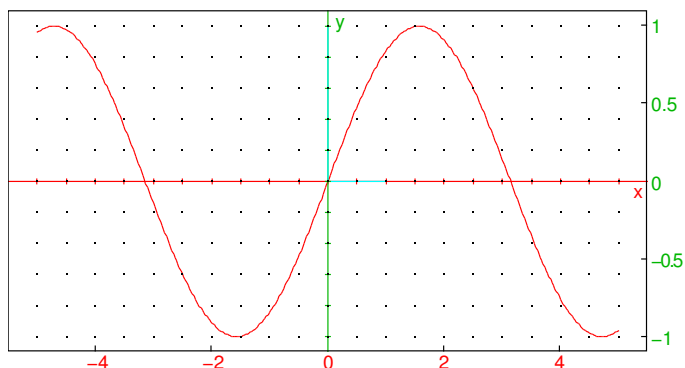


On peut aussi utiliser la commande `plotode` en-dehors d'une commande `plotfield`.  
Sur la graph 35eii, la mémoire est insuffisante pour utiliser `plotfield` avec l'option `plotode`.

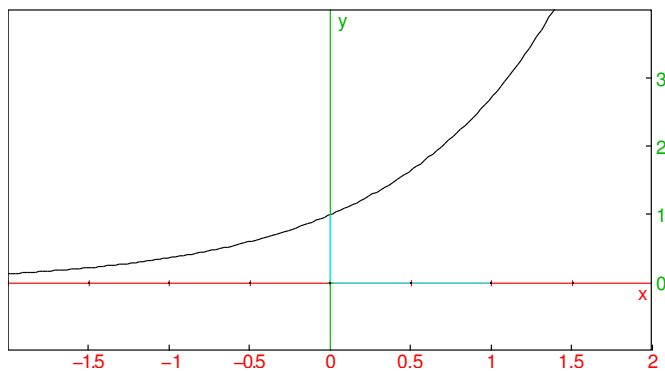
On peut tracer simultanément plusieurs graphiques, il suffit de séparer les commandes de tracé par ;

La touche OPTN vous permet de spécifier certaines options graphiques :

- Pour modifier les couleurs des graphes, utiliser la touche OPTN et sélectionnez une couleur puis tapez F2 (Exemple 1) pour insérer `display=couleur`, par exemple OPTN r e d F2 pour `plot(sin(x), display=red)`



- Pour changer l'épaisseur des segments (y compris les lignes polygonales utilisées pour tracer une courbe), utiliser `display=line_width_2` à `display=line_width_8`. OPTN 1 i puis curseur vers le bas si nécessaire puis F2 Pour changer à la fois la couleur et l'épaisseur, additionnez les attributs, par exemple : `display=red+line_width_2`
- Les cercles ainsi que les polygones peuvent être remplis avec l'attribut `display=filled` (qui peut s'additionner à d'autres attributs).
- Pour remplacer la fenêtre graphique calculée automatiquement par des valeurs prédéfinies, utiliser la touche OPTN, sélectionner `gl_x` ou/et `gl_y` et indiquer l'intervalle en  $x$  ou en  $y$  souhaité, la commande doit précéder une commande de tracé. Par exemple  
`gl_x=-2..2;gl_y=-1..4;plot(exp(x))`



- Pour enlever les axes, sélectionner `axes` puis taper F2 (`axes=0`). La commande doit précéder une commande de tracé.

## 6 L'éditeur d'expressions

Lorsqu'un calcul renvoie une expression, elle est affichée en plein écran dans l'éditeur d'expression 2d. Depuis l'historique des calculs, si le niveau sélectionné est une expression, l'appui sur F3 (voir) affiche l'expression dans l'éditeur 2d. En ligne de commande, l'appui sur F3 (voir) ouvre aussi l'éditeur 2d, soit avec 0 si la ligne de commande était vide, ou avec le contenu de la ligne de commande si celle-ci est syn-

taxiquement correcte.

Lorsque l'éditeur 2d est ouvert, l'expression est affichée en plein écran et une partie de l'expression est sélectionnée. On peut alors agir sur la sélection en exécutant des commandes saisies via les menus ou le clavier, on peut aussi éditer la sélection (en mode de saisie 1d). Ceci permet de retravailler des sous-expressions ou d'éditer une expression en écriture naturelle.

Exemple 1 : nous allons saisir

$$\lim_{x \rightarrow 0} \frac{\sin(x)}{x}$$

Depuis une ligne de commande vide, taper F3 (voir), vous devez voir 0 sélectionné. Tapez sur la touche x et EXE, maintenant c'est x qui est en surbrillance. Tapez sur la touche SIN, c'est  $\sin(x)$  qui est en surbrillance. Tapez sur la touche de division (au-dessus de -), vous devez voir  $\frac{\sin(x)}{0}$  avec 0 en surbrillance, tapez x puis EXE, vous devez voir  $\frac{\sin(x)}{x}$  avec x au dénominateur en surbrillance. Tapez sur le curseur flèche vers le haut pour mettre  $\frac{\sin(x)}{x}$  en surbrillance, puis F2 4 (pour limit). L'expression est correcte, vous pouvez taper EXE pour la recopier en ligne de commande and EXE again to eval it. Si on avait voulu une limite en  $+\infty$ , il aurait fallu déplacer la sélection avec curseur vers la droite, puis faire F1 8 (oo) EXE.

Exemple 2 : nous allons saisir

$$\int_0^{+\infty} \frac{1}{x^4 + 1} dx$$

Depuis une ligne de commande vide, taper F3 (voir), puis F2 3 (integrate), vous devez voir

$$\int_0^1 0 dx$$

avec  $x$  sélectionné. Il faut donc changer le 1 de la borne supérieure et le 0 à intégrer. Pour modifier le 0, curseur vers la gauche pour le sélectionner puis 1 / ( $x^4 + 1$ ) EXE, puis curseur vers la gauche F1 8 EXE. Taper sur EXE pour recopier vers la ligne de commande puis EXE pour effectuer le calcul, le résultat s'affiche dans l'éditeur 2d, EXE quitte l'éditeur avec dans l'historique l'intégrale et sa valeur (en syntaxe algébrique 1d).

Exemple 3 : nous allons calculer et simplifier

$$\int \frac{1}{x^4 + 1} dx$$

Depuis une ligne de commande vide, taper F3 (voir), puis F2 3 (integrate), vous devez voir

$$\int_0^1 0 dx$$

Déplacez le curseur sur le 0 de la borne inférieure de l'intégrale et tapez sur la touche DEL, vous devez voir

$$\int 0 dx$$

avec le tout sélectionné. Utilisez le curseur vers le bas pour sélectionner 0 et tapez  $1/(x^4+1)$  EXE puis EXE pour recopier en ligne de commande puis EXE pour exécuter le calcul, le résultat s'affiche maintenant dans l'éditeur 2d.

On peut alors sélectionner avec les touches du curseur par exemple l'argument d'un des arctangentes et exécuter F1 EXE (simplify) pour effectuer une simplification partielle du résultat, puis recommencer avec l'autre arctangente.

On peut simplifier encore plus, en rassemblant les logarithmes. Pour cela il faut d'abord échanger deux des arguments de la somme. Sélectionnez un des logarithmes avec des déplacements du curseur, puis tapez

- Graph 90 : shift-curseur droit ou gauche
- Graph 35 : F5 curseur droit ou gauche, ensuite ALPHA

cela échange l'argument sélectionné avec son frère de droite ou de gauche. Tapez ensuite ALPHA curseur vers la droite ou vers la gauche, ceci augmente la sélection en ajoutant le frère de droite ou de gauche. Une fois les deux logarithmes sélectionnés, menu F1 2 EXE (factor), puis descendez la sélection sur la somme ou différence de logarithmes, allez dans le menu F4 (cmds) puis EXE (Tout), tapez les lettres l, n, c ce qui déplace à la première commande commençant par lnc, sélectionnez lncollect, validez et tapez enfin sur F6 (eval).

## 7 Sessions de calculs

### 7.1 Edition de l'historique.

En utilisant la touche curseur vers le haut/bas, on se déplace dans l'historique des calculs, le niveau courant est en surbrillance.

Pour modifier l'ordre des niveaux dans l'historique des calculs, tapez ALPHA-curseur vers le haut ou vers le bas. Pour effacer un niveau, appuyez sur la touche DEL (le niveau est recopié dans le presse-papiers).

Pour modifier un niveau existant, on tape sur F3 ou sur ALPHA-F3. Dans le premier cas, c'est l'éditeur 2d qui est appelé si le niveau est une expression, dans le deuxième cas, c'est l'éditeur texte qui est appelé. Taper EXIT pour annuler les modifications ou EXE pour valider. Si les modifications sont validées, les lignes de commande situées en-dessous de la ligne modifiée seront automatiquement calculées, tenant compte des modifications, par exemple si vous modifiez un niveau comme  $A:=1$ , les lignes situées en-dessous dépendant de A seront actualisées.

Ce processus peut être automatisé en utilisant un curseur, que l'on peut créer avec un assistant, depuis le menu F6, Parameter. Une fois créé, vous pouvez modifier un curseur en tapant sur les touches + ou - lorsque le niveau contenant la commande assume ou parameter est sélectionné (tapez \* ou / pour une modification plus rapide).

### 7.2 Variables

En appuyant sur la touche VARS vous affichez la liste des variables qui ont une valeur, ainsi que des commandes de gestion de variables. Déplacez le curseur vers

une variable puis EXE pour la recopier en ligne de commande, DEL copie en ligne de commande la commande d'effacement de la variable (confirmez ensuite avec EXE). La commande `restart` permet d'effacer toutes les variables. La commande `assume` permet de faire une hypothèse sur une variable, par exemple `assume(x>5)` (> se trouve dans le menu shift-PRGM).

### 7.3 Sauvegarde et échange de sessions avec Xcas.

Depuis l'historique des calculs, avec le menu F6, vous pouvez sauvegarder une session de travail sur la mémoire de stockage de la calculatrice. Les fichiers ont comme extension `xw`. Vous pouvez alors sauvegarder ces fichiers sur votre ordinateur en connectant la calculatrice comme clef USB (choix F1), depuis l'explorateur de fichiers de votre ordinateur.

Ces fichiers sont compatibles avec Xcas et Xcas pour Firefox. Vous pouvez ouvrir une session depuis le menu Fich, Ouvrir de Xcas ou depuis le bouton Charger/Parcourir de la page de Xcas pour Firefox. Vous pouvez sauvegarder des sessions Xcas et Xcas pour Firefox au format `xw` dans le menu Fich, Exporter comme Khicas de Xcas, ou le bouton Export de Xcas pour Firefox.

## 8 Programmation

Vous pouvez programmer en utilisant les structures de commande en français de Xcas ou en utilisant la compatibilité de syntaxe Python. Les programmes très courts (en une ligne) peuvent être saisis directement en ligne de commande. Les programmes plus longs ou que l'on souhaite sauvegarder seront saisis dans l'éditeur de programmes sur la calculatrice, ou bien transférés depuis un PC ou une autre calculatrice.

### 8.1 Utilisation en exemples

#### Un premier exemple en ligne de commande :

une fonction définie par une expression algébrique. On saisit `nom_fonction(parametres):=expression`. Par exemple, pour définir le périmètre d'un cercle de rayon  $r$ , on peut taper

```
peri(r):=2*pi*r
```

$$r \mapsto 2\pi r$$

puis on peut calculer `peri(1)`

$$2\pi$$

. Pour saisir `:=` on peut utiliser le raccourci shift-→ ou shift-INS selon la version de  $\chi$ CAS.

**Autre exemple, pour calculer l'intervalle de confiance de seconde** connaissant une fréquence  $p$  et un effectif  $n$ , on tape

$$F(P, N) := [P - 1/\sqrt{N}, P + 1/\sqrt{N}]$$

$$(P, N) \mapsto \left[ P - \frac{1}{\sqrt{N}} \quad P + \frac{1}{\sqrt{N}} \right]$$

puis on teste  $F(0.4, 30)$

$$[0.217425814165, 0.582574185835]$$

#### Autre exemple : avec la tortue de Xcas

La tortue de Xcas est un petit robot qui se déplace selon des ordres qui lui sont donnés en laissant une trace de son passage. Les commandes de la tortue sont accessibles depuis le dernier item du menu F6, ou par le raccourci shift-QUIT. Saisir la commande `avance` dans ce menu puis valider, vous devez voir la tortue (symbolisée par un triangle) avancer de 10 pixels. Taper EXIT pour revenir en ligne de commande. Saisir la commande `tourne_gauche` et valider, la tortue a tourné de 90 degrés. Répéter 3 fois ces deux commandes pour afficher un carré.

Pour effacer le dessin et ramener la tortue à l'origine, saisir la commande `efface`. Pour faire des dessins tortue, il est conseillé d'utiliser l'éditeur de programmes (cf. ci-dessous).

#### Autre exemple : une boucle “oneliner” en syntaxe Xcas.

Tapez shift-PRGM, puis sélectionnez l'exemple de `pour` (boucle Xcas)

```
pour j de 1 jusque 10 faire print(j, j^2); fpour;
```

0

tapez sur EXE, vous devez voir les carrés des entiers de 1 à 10.

**Exercice :** faire faire un carré à la tortue en utilisant une boucle.

#### Utilisation de l'éditeur

Modifions cet exemple pour faire afficher les carrés de 1 à  $n$  en utilisant la syntaxe compatible Python et l'éditeur de programmes. Vérifiez que la syntaxe Python est cochée (F6 ou shift-SETUP), sinon cochez-la. Tapez F6 (Fich, Cfg) puis sélectionner Editeur script pour ouvrir l'éditeur. S'il n'y a pas encore de script, on vous propose de choisir entre Tortue ou Programme, sélectionnez Programme, vous devriez obtenir `def session(x) : .` Remplacez `session` par `f` et `x` par `n`, puis déplacez le curseur en fin de ligne et passez à la ligne (shift-EXE). Tapez Shift-PRGM puis `1 for`, placez le curseur entre `for` et `in` et tapez F5 `j`, placez le curseur entre les parenthèses de `range` tapez `1, n+1` puis déplacez le curseur à la ligne suivante F4 (cmds), EXE (1 Tout), P, R, déplacez la surbrillance avec le curseur vers le bas jusque `print` et validez (EXE), puis tapez `j, j^2` (F5 j ALPHA , F5 j ALPHA  $x^2$ ). Vous devriez avoir le programme suivant :

```
def f(n):
    for j in range(1, n+1):
        print(j, j**2)
    return
```



N.B. : pour la puissance, on peut utiliser `^` ou `**` dans KhiCAS (sauf si on a exécuté la commande `python_compat(2)`), notez qu'il faut utiliser `**` en Python.

Maintenant, tapez F6 (Fich,Cfg) et sélectionnez `1. Tester syntaxe` ou tapez EXE. Si tout va bien, vous devez voir `Syntaxe correcte` dans la ligne d'état. Sinon, le numéro de ligne de la première erreur est indiqué ainsi que le mot qui a provoqué l'erreur. Le curseur est positionné sur la ligne où l'erreur a été détectée (il peut arriver que l'erreur soit située avant mais détectée un peu plus loin seulement). Si vous utilisez la syntaxe en Python, notez que les structures de programmation sont traduites en langage Xcas, les erreurs affichées le sont par rapport à cette traduction (donc des mots-clés de fin de structure comme `end` rajoutés par le traducteur peuvent apparaître dans le message d'erreur).

Si le programme est syntaxiquement correct, vous pouvez le sauvegarder depuis le menu F6 (File). Pour l'exécuter, revenez à la ligne de commande en tapant la touche EXIT. En tapant la touche VARS, `f` doit apparaître, sélectionnez `f` puis complétez pour obtenir par exemple la commande `f(10)`, puis EXE, vous devriez voir s'afficher les carrés de 1 à 10.

### 3ième exemple : Calcul de l'intervalle de confiance de terminale S

On peut le saisir en ligne de commande

```
F(P,N):=[P-1.96*sqrt(P*(1-P)/N),P+1.96*sqrt
(P*(1-P)/N)]
```

$$(P,N) \mapsto \left[ P - 1.96\sqrt{P\frac{1-P}{N}} \quad P + 1.96\sqrt{P\frac{1-P}{N}} \right]$$

On peut éviter les calculs redondants en utilisant une variable locale (utiliser shift-PRGM pour saisir `def` et `return`)

```
def F(P,N):
  D=1.96*sqrt(P*(1-P)/N)
  return [P-D,P+D]
```

**Exercice** Créez une nouvelle session contenant un script pour afficher un carré avec la tortue. Créez ensuite une fonction `f` d'argument `n` pour afficher un carré de `n` pixels.

**Solution** Faire F6, Nouvelle session, taper `carre` puis F6, Editeur script, choisir Tortue ce qui crée un script contenant l'instruction `efface`. Ajouter 4 fois `avance`; `tourne_gauche`;. Appuyer sur EXE pour tester. Quitter avec EXIT (et sauvegardez avec F1).

Taper EXIT. Modifiez avant la ligne `efface` en

```
def f(n):
  for j in range(4):
    avance(n)
    tourne_gauche
```

puis après la ligne `efface`; tapez par exemple `f(40)` puis F6 1 Tester syntaxe ou EXE.

### Un exemple de fonction non algébrique : le calcul du PGCD de 2 entiers.

Utiliser shift-EXE pour passer à la ligne. Le caractère ! est accessible depuis le menu Programmation\_cmds (F4 10, raccourci F4 0). En syntaxe Xcas

```
fonction pgcd(a,b)
  tantque b!=0 faire
    a,b:=b,irem(a,b);
  ftantque;
  return a;
ffonction
```

Le même en syntaxe Python

```
def pgcd(a,b):
  while b!=0:
    a,b=b,irem(a,b)
  return a
```

On vérifie pgcd(12345,3425)

5

### Mise au point

Depuis l'historique des calculs, la commande debug (disponible depuis shift-PRGM) permet d'exécuter une fonction en mode pas-à-pas, i.e. visualiser l'évolution des variables instruction par instruction, par exemple

```
debug(pgcd(12345,3425))
```

## 8.2 Quelques exemples

Vous trouverez en ligne des exemples de sessions dont certains sont des TP pour classes de seconde de l'IREM de Grenoble.

## 8.3 Commandes utilisables

Contrairement aux adaptations de MicroPython proposées par les constructeurs (dont celui de la Casio Graph 90+e), la programmation en (simili-)Python dans Khi-CAS n'est pas une application indépendante. Vous pouvez donc utiliser tous les types de Xcas (par exemple les rationnels) et appliquer toutes les commandes de Xcas dans vos programmes. Ceci correspond plus ou moins à un environnement Python avec les modules math, cmath, random (plus complet que le module urandom fourni par les constructeurs), scipy, numpy, un petit module de graphiques pixelisé (set\_pixel(x,y,c), set\_pixel() pour synchroniser l'affichage, clear(), draw\_line(x1,y1,x2,y2,c), draw\_polygon([[x1,y1],[x2,y2],...],c), draw\_rectangle(x,y,w,h,c), draw\_circle(x,y,r,c), la couleur+épaisseur+remplissage c est un paramètre optionnel, draw\_arc(x,y,rx,ry,t1,t2,c) permet de tracer un arc d'ellipse). et pour remplacer matplotlib on peut utiliser les commande graphiques dans un repère de  $\chi$ CAS (point, line, segment, circle, barplot, histogram et les

commandes `plot . . .`). De plus, vous pouvez travailler avec des expressions et faire du calcul formel dessus. La liste complète des commandes portées se trouve en annexe, pour une présentation détaillée, on renvoie à la documentation de Xcas.

## 9 Raccourcis claviers.

### 9.1 KhiCAS 90 et 50 (version en 2 fichiers)

Ces raccourcis sont utilisables dans le shell et l'éditeur texte

- shift INS : table de caractères
- F1 à F6, shift-F1 à shift F6 : selon le bandeau en bas
- OPTN : menu rapide des options de couleur
- shift-OPTN : commandes de programmation
- VARS : liste des variables
- shift-PRGM : menu rapide utile en programmation
- MENU : retour au menu principal Casio
- shift-SETUP : configuration
- EXIT : passage du shell à l'éditeur et réciproquement
- shift-EXIT : affiche l'écran de la tortue logo
- ALPHA-EXIT : si le mode alpha n'est pas verrouillé, affiche le dernier graphe 2d ou 3d
- shift angle : menu rapide géométrie
- fraction : menu rapide polynômes dans le shell, indentation dans l'éditeur, force l'évaluation dans le tableur
- shift-fraction : aide/complétion
- touche  $S \leftrightarrow D$  : applications additionnelles (tableur, finance, ...). Dans le tableur, affiche les graphes du tableur.
- shift , ; :
- shift  $\rightarrow$  : := ou : selon l'interpréteur Xcas ou Python
- AC/ON : annule la sélection ou annule la recherche, sinon efface la ligne courante et la copie dans le presse-papier
- shift CAPTURE : sauve la session ou le fichier texte courant
- shift CLIP : début de sélection de zone ou copie de sélection dans le presse-papier
- shift PASTE : colle le presse-papier
- shift CATALOG : liste toutes les commandes Xcas
- shift FORMAT : commandes de programmation
- shift 6 : menu rapide avec `<>_!` et `comb, rand, binomial, normald!`
- shift List : menu rapide liste
- shift Mat : menu rapide matrice
- shift 3 : menu rapide algèbre (bi-)linéaire
- shift EXE : passe à la ligne dans l'éditeur

Les raccourcis claviers ouvrant un menu rapide sont configurables, en éditant le fichier `FMENU.py`. Pour revenir à la configuration par défaut, effacez ce fichier depuis l'application Memory de la Casio.

## 9.2 KhiCAS (version courte en 1 fichier)

- F1 à F3 : selon les légendes
- F4 : catalogue de commandes.
- F5 : passage majuscules minuscules. Si le mode alphabétique n'est pas actif, bloque le clavier en mode alpha minuscule.
- F6 : menu Fichier et configuration.
- (–) : dans l'éditeur de programmes, renvoie le caractère \_.
- shift PRGM : commandes de programmation
- OPTN : toutes les options
- shift-QUIT : commandes tortues
- shift-List : commandes et édition de listes
- shift-Mat : commandes et édition de matrices
- touche angle : commande nombres complexes
- touche fraction : indentation dans l'éditeur, menu divers dans l'historique
- touche fraction shiftée (jaune) : graphes
- touche r rouge : `abs`
- touche  $\theta$  rouge : `arg`
- touche  $S \leftrightarrow D$  : commandes sur les nombres réels
- touche  $S \leftrightarrow D$  shiftée (jaune) : commandes sur les entiers
- Shift-INS (touche DEL) : `:=`
- Depuis le shell, taper sur la touche flèche vers le bas pour obtenir de l'aide ou/et complétion. Depuis l'éditeur de programmes, taper sur shift touche de fraction (G).

Dans l'éditeur de programmes :

- touches curseur shiftées : déplacement en début/fin de ligne/fichier.
- shift CLIP : sélection. Déplacer le curseur à l'autre extrémité puis taper sur DEL pour effacer la sélection et la copier dans le presse-papier ou à nouveau sur shift CLIP pour copier la sélection sans l'effacer. Taper sur AC/ON pour annuler.
- EXE : si une recherche/remplacement de mot est active (après avoir fait F6 6), recherche l'occurrence suivante d'un mot. Sinon, teste la syntaxe.
- shift EXE : passe à la ligne
- DEL efface le caractère précédent ou la sélection.
- shift PASTE : copie le presse-papier
- AC/ON : annule la sélection ou annule la recherche, sinon efface la ligne courante et la copie dans le presse-papier
- EXIT : quitte l'éditeur et revient en ligne de commandes. On peut revenir ensuite à l'éditeur en tapant à nouveau EXIT.

## 10 Remarques

Pour éteindre la Graph 90 depuis KhiCAS, taper d'abord sur la touche MENU puis shift ON. Lorsque vous rallumez la calculatrice, taper sur MENU.

Pour connecter la Graph 90 au PC comme une clef USB lorsqu'on est dans KhiCAS, il faut taper sur une touche après avoir tapé sur F1, sinon rien ne se passe.

Si  $\chi$ CAS se bloque dans un long calcul, essayez de taper sur la touche AC/ON. Si cela n'a pas d'effet, vous pouvez toujours réinitialiser la calculatrice avec le bouton reset à l'arrière.

Si  $\chi$ CAS plante avec un affichage du type SYSTEM ERROR, etc. essayez de taper sur la touche MENU puis de lancer une autre application, si tout se passe bien, vous pourrez sauvegarder votre session et relancer  $\chi$ CAS sans avoir à effectuer une réinitialisation complète. En cas de problèmes persistants, depuis le MENU principal, ouvrir l'application Mémoire, puis F2 (mémoire de stockage) puis effacez le fichier `session.xw` (déplacez le curseur sur le fichier, puis F1 pour SELECT puis F6 pour DELETE). Si  $\chi$ CAS90 se lance (la version en deux fichiers), vous pouvez aussi depuis le menu F6 sélectionner le dernier item Quit et choisir la réinitialisation de KhiCAS.

Vous disposez dorénavant d'environ 500K pour vos données sur une Graph 90 (ce qui permet de travailler assez confortablement) et de 58K sur une Graph 35eii (ce qui est vite rempli). Sur la Graph 90, il est conseillé d'ouvrir le menu VARS de temps en temps, si vous constatez que la mémoire disponible descend en-dessous des 100K, relancez  $\chi$ CAS (touche MENU, puis ouvrir n'importe quelle autre application puis MENU et réouvrir  $\chi$ CAS).

## 11 Version plus complète (en 2 fichiers)

Cette version de  $\chi$ CAS en deux fichiers contient un portage de MicroPython avec plusieurs modules utiles pour l'enseignement, plus de commandes de Xcas, en particulier des commandes de géométrie, d'algèbre avancée, le support de flottants avec grande précision, un moteur d'affichage 3d. Le menu Fichier permet d'accéder à des applications additionnelles : tableur formel, géométrie, finance, table périodique des éléments ...

Pour installer cette version plus complète, cf. la section 1.

### 11.1 MicroPython 1.12

Pour passer du shell de Xcas à celui de MicroPython, tapez shift-SETUP et sélectionnez MicroPython. Lorsque MicroPython est actif, les menus sont sur fonds jaune-brun, sinon sur fonds rose (Xcas) ou bleu (Xcas compatible Python). Les modules disponibles sont : turtle (version plus complète que celle de Casio, avec en particulier la possibilité de remplir des formes), graphic (version plus complète de casioplot, avec compatibilité Numworks), matplotlib (graphes repérés), arit (arithmétique entière), linalg/numpy (algèbre linéaire), ulab (compatibilité scipy), cas (calcul formel dans Python).

### 11.2 Graphes 3d et 4d

Cette version de  $\chi$ CAS en deux fichiers contient toutes les commandes de base de Xcas de géométrie 2d et 3d. On peut en particulier tracer des graphes de fonction de 2 variables, des cones, des plans, etc. Par exemple tapez F4 \* 5 F2 pour sélectionner un exemple de cube ou shift F4 1 pour sélectionner la commande plot, puis tapez  $x^2 - y^2$ .

Pour tracer rapidement le graphe d'une fonction de  $\mathbb{C}$  dans  $\mathbb{C}$ , on peut utiliser la commande `plot` avec en argument une fonction de 2 variables (partie réelle et imaginaire) à valeurs complexes, par exemple `plot((x+i*y)^2-9)` ou pour les fonctions holomorphes/méromorphes ayant une expression en fonction de la variable complexe sans passer par partie réelle et imaginaire, on peut utiliser la commande `plot3d` par exemple `plot3d(x^2-9)` (la commande `plot(x^2-9)` ne convient pas car déjà utilisée pour un graphe de  $\mathbb{R}$  dans  $\mathbb{R}$ ). Le module de la fonction est représenté selon  $z$ , l'argument est représenté par les couleurs de l'arc en ciel, de  $-\pi$  en bleu violet à 0 en vert (en passant par jaune et orange) et de 0 à  $\pi$  en passant par cyan.

Pour préciser des options, il faut utiliser la commande `plotfunc`, par exemple `plotfunc((x+i*y)^3-1, [x=-2..2, y=-2..2], nstep=500)` pour tracer  $z \rightarrow z^3 - 1$  depuis le carré du plan complexe centré en l'origine de coté 4, avec une discrétisation utilisant 500 petits rectangles.

Le moteur de rendu 3d étant relativement lent sur la calculatrice, la précision de tracé par défaut est moyenne ce qui impacte en premier lieu le rendu des objets avec des angles, comme les polyèdres. On peut changer la précision du tracé en tapant F2 (plus rapide, moins précise) ou F3 (moins rapide, plus précis)<sup>1</sup>

Pour obtenir un unique tracé précis sans changer la précision par défaut, tapez sur la touche `^` et soyez patients. On peut interrompre un tracé précis en cours en tapant sur la touche DEL. Les touches de déplacement permettent de changer de point de vue, taper 5 pour revenir au point de vue initial, `-` ou `+` pour faire un zoom in ou out. Lorsqu'on laisse une touche de déplacement appuyée, la précision de tracé est diminuée temporairement pour donner une certaine fluidité, lorsqu'on relâche la touche, la dernière position est retracée avec la précision par défaut. F4 permet de montrer ou cacher une 2ème surface cachée, F5 affiche ou cache des points intermédiaires, F6 montre ou cache les arêtes des polyèdres.

### 11.3 Géométrie interactive 2d/3d

L'application de géométrie permet de construire des figures dans le plan ou dans l'espace, et de faire bouger un point et tout ce qui en dépend pour illustrer certaines propriétés (géométrie dynamique). On peut faire des constructions de géométrie euclidienne pure, mais aussi avec des graphes de fonction, des coniques, etc. L'application possède deux "vues" : la vue graphique et la vue symbolique qui contient les commandes Xcas permettant de créer la figure (la philosophie de cette application est proche de celle du logiciel Geogebra, avec les commandes de Xcas).

Cette section donne un aperçu rapide de l'application, pour une documentation plus complète avec quelques captures d'écran, cf. ici

#### 11.3.1 Modes, vue graphique et symbolique.

Taper F6 1 (ou la touche `S↔D`) pour afficher la liste des applications additionnelles, puis EXE puis sélectionnez soit une nouvelle figure 2d ou 3d soit une figure existante.

1. Pendant le déroulement de  $\chi$ CAS, le processeur peut être mis en mode overclocké, en utilisant du code de Sébastien Michelland, semblable à l'addon Ptune3

Vous pouvez aussi ouvrir l'application de géométrie depuis un graphe (par exemple après avoir tapé `plot(sin(x))`) en tapant F6 puis Sauvegarder figure.

Au lancement on est dans la vue graphique en mode repère, les touches de curseur permettent de changer de point de vue. Pour changer le mode, utiliser la touche F4, pour passer en vue symbolique et vice-versa taper EXE. Par exemple tapez F4 3 pour passer en mode point qui permet de construire des points en déplaçant le pointeur et en tapant EXE ou tapez F4 5 pour passer en mode triangle qui permet de construire un triangle à partir de ses 3 sommets, on déplace le pointeur et on tape EXE trois fois. Pour déplacer le pointeur, utiliser les touches de déplacement, pour se déplacer plus rapidement, faire shift touche de curseur. Si on est proche d'un point existant, son nom apparaît en bas. Pour déplacer le pointeur vers un point existant, vous pouvez aussi taper le nom du point (par exemple touche alpha A). Pour une figure 3d, les objets créés depuis la vue graphique seront situés dans le plan qui apparaît en jaune sur la figure. On peut modifier ce plan en utilisant les touches 4 et 6. Il est fortement conseillé de conserver un point de vue avec l'axe  $Oz$  vertical (donc changer de point de vue uniquement avec les touches flèche droite ou gauche qui effectuent une rotation autour de  $Oz$ ).

Le mode pointeur permet de sélectionner un point et de le déplacer pour observer comment la construction varie, ce qui permet de mettre en évidence des propriétés de la figure, par exemple concurrence de 3 droites.

Si vous tapez sur la touche EXIT depuis la vue graphique de l'application, vous revenez au mode repère ou si vous y étiez vous passez en vue symbolique. Vous pouvez ajouter des objets à la construction depuis cette vue, en mettant une commande par ligne. Tapez shift-EXE pour passer à la ligne. Tapez EXE pour revenir à la vue graphique. Dans la vue symbolique, vous pouvez sauvegarder la construction géométrique au format texte (avec une extension `.py`, même s'il ne s'agit pas d'un script Python). Tapez EXIT pour quitter l'application de géométrie.

Lorsque vous quittez l'application de géométrie, la figure est automatiquement sauvegardée dans une variable Xcas qui a le même nom que celui du nom de fichier affiché dans la vue symbolique. Vous pouvez purger la variable Xcas si vous voulez effacer la figure de la session.

#### **Exemple : cercle circonscrit.**

Depuis le shell, taper F6 1 sélectionner nouvelle figure 2d et valider EXE. Puis F4 5 Triangle, EXE pour créer le premier sommet du triangle puis déplacer le pointeur avec les touches de déplacement, EXE pour créer le 2ème sommet du triangle, déplacer le pointeur à nouveau et EXE pour créer le triangle.

Version longue en construisant le centre : Taper F4 7, sélectionner 8 Médiatrice, déplacer le pointeur de sorte que seul un segment du triangle soit sélectionné (affichage en bas à droite `perpen_bisector D5,D`), taper EXE pour créer la médiatrice du segment, déplacer le pointeur sur une autre arête du triangle et EXE pour créer la 2ème médiatrice, optionnellement sur le 3ème segment pour avoir les 3 médiatrices. Puis F4 6 et 4 Intersection unique. Déplacer le pointeur vers une des médiatrices, taper EXE puis vers une autre médiatrice, taper EXE, ceci crée le centre du cercle circonscrit. Pour tracer le cercle, taper F4 4, déplacer le pointeur au centre du cercle (vous pouvez utiliser les touches de déplacement ou juste taper ALPHA H ou la bonne lettre si le centre du cercle s'appelle autrement), puis EXE puis sur un des sommets et EXE.

Version courte avec la commande `circonscrit` : taper F4 9 puis `circonscrit` puis

sélectionner chaque sommet avec EXE (ALPHA A EXE ALPHA B EXE ALPHA C EXE, remplacez A, B, C par les lettres du sommet du triangle).

Version en vue symbolique : taper EXIT puis en fin de script sur une ligne vide (taper shift EXE s'il faut en créer une), taper

```
c:=circonscriit(A,B,C) EXE
```

### Exemple 3d : bac septembre 2019

Taper F6 1 pour lancer l'application de géométrie puis nouvelle figure 3d. Puis EXIT ou EXE pour passer en vue symbolique. Puis F5 c ALPHA shift = puis F4 flèche haut deux fois pour sélectionner 3D puis EXE puis 5 pour cube puis F6 (aide), qui explique que les 2 premiers arguments de cube sont les sommets d'une arête, le troisième est un point d'un plan d'une face. Le premier exemple nous convient ici exactement, on tape F2 et on obtient `c=cube([0,0,0],[1,0,0],[0,1,0])` On tape EXE pour voir le cube puis + plusieurs fois pour zoomer et EXE pour revenir à la vue symbolique. Vous pouvez sauvegarder à tout moment la construction au format texte depuis le menu F6. On passe à la ligne en tapant shift EXE. Puis on définit les sommets du cube en tapant A,B,C,D,E,F,G,H= (taper ALPHA A , ALPHA B etc.), puis F4 et flèche vers le haut 3 fois pour sélectionner Géométrie puis flèche vers le haut 4 fois pour sélectionner sommets EXE et mettre c en argument sommets(c). Taper EXE pour visualiser puis EXE à nouveau pour revenir en vue symbolique. Passer à la ligne avec shift EXE puis créer le plan ABG en tapant ALPHA P = puis F2 pour ouvrir le menu rapide lines et 8 pour saisir plane. La commande plan prend en arguments 3 points pour définir le plan (on peut aussi donner une équation cartésienne, ici A,B,G, `P=plan(A,B,G,` on va lui ajouter une couleur avec le menu rapide F3 `disp display=filled+green,` vérifier en visualisant avec EXE EXE. On passe à la ligne (shift EXE) et on crée le segment DE ALPHA S = F2 sélectionner la commande segment avec EXE puis D,E et F3 pour lui donner une couleur `S=segment(D,E,color=cyan)` (on pouvait aussi créer le segment depuis la vue graphique en mode Lignes mais sur la Casio c'est un peu lent). La construction est donc la suivante :

```
c=cube([0,0,0],[1,0,0],[0,1,0])
A,B,C,D,E,F,G,H=sommets(c)
P=plan(A,B,G,display=filled+green)
S=segment(D,E,display=cyan)
```

Vous pouvez taper EXE pour la visualiser et utiliser les flèches de déplacement pour changer de point de vue. Taper EXE ou EXIT pour revenir en vue symbolique. Pour quitter l'application taper EXIT. Taper F1 pour sauvegarder la figure si nécessaire. Vous pouvez depuis le shell de KhiCAS accéder à de nombreuses informations de géométrie analytique, par exemple `equation(P)` (menu F4 Géométrie) vous donnera l'équation cartésienne du plan  $P$  ou `is_orthogonal(P,S)` (F4 Géométrie) vous confirmera que le plan  $P$  est orthogonal au segment  $S$ .

### 11.3.2 Curseurs

Vous pouvez créer un paramètre qui se déplace entre 2 valeurs extrêmes par saut de 1% depuis le menu F4 Curseur en vue graphique ou la commande `element` en vue symbolique.



### Exemple : explorateur quadratique avec des curseurs

pour explorer comment une parabole d'équation  $y = ax^2 + bx + c$  dépend de la valeur de  $a, b, c$ , créer 3 curseurs (F4 curseur vers le haut 4 fois EXE EXE). En vue symbolique on doit avoir quelque chose ressemblant à

```
a:=element(-1..1)
b:=element(0..1,0.5)
c:=element(-1..1)
```

puis ajouter un graphe, depuis la vue graphique taper F4 0 (pour 10 Courbe) et sélectionner plot, ou en vue symbolique shift F6 et sélectionner plot, remplir entre les parenthèses par  $a*x^2+b*x+c$  (attention à ne pas oublier les \*), puis valider avec EXE. En vue graphique, vous devez voir 3 curseurs  $a, b$  et  $c$  et le graphe correspondant. Vous pouvez maintenant faire varier les valeurs de  $a, b, c$  depuis la vue graphique en mode pointeur (F4 2) en déplaçant le pointeur vers  $a, b, c$  et en tapant EXE puis les flèches de déplacement gauche ou droit et EXE pour arrêter.

On peut bien sûr faire des exemples plus simples avec un ou deux curseurs et une courbe dépendant d'un ou de deux paramètres. Par exemple un explorateur linéaire droite ( $y=a*x+b$ ) ou trigonométrique plot( $\sin(a*x+b)$ ).

### 11.3.3 Mesures et légendes

En tapant F4 puis en sélectionnant 13 Mesures, on peut afficher une mesure en un point de la figure. Par exemple après avoir construit un triangle, on peut afficher son périmètre ou son aire en tapant F4 puis deux fois flèche vers le haut, EXE. Déplacer le pointeur près du triangle, taper EXE, puis déplacer le curseur à l'endroit où on souhaite mettre la mesure et taper EXE.

On peut afficher n'importe quelle légende avec la commande `legende()` depuis la vue symbolique. Le premier argument de légende peut être un point de la figure, ou bien un vecteur de deux entiers donnant la position absolue en pixels mesuré depuis le coin en haut à gauche. Le deuxième argument est la légende, cela peut être une chaîne de caractères ou n'importe quelle expression.

Si la légende est une valeur numérique, elle peut être utilisée pour un paramètre numérique d'une commande, par exemple le rapport d'une homothétie ou l'angle d'une rotation en utilisant la commande `extract_measure`. Par exemple `r:=legend([20,40], "2")`  
`homothetie(A, extract_measure(r), B)`

### 11.3.4 Traces

La commande `trace()` permet de conserver la trace du déplacement d'un objet géométrique.

**Exemple** Enveloppe des normales à une courbe paramétrée (ici une ellipse).

```
E:=plotparam([cos(t),2*sin(t)],t=-pi..pi)
a:=element(-pi..pi)
M:=element(E,a)
```

```

T:=tangent (M)
N:=perpendiculaire (M,T)
trace (N)

```

En faisant varier  $a$ , on observe une courbe séparant la zone des normales à une zone sans point tracé qui est l'enveloppe des normales, c'est la développée de l'ellipse

```
evolute (E,color=red)
```

On peut effacer les traces à tout moment avec la touche F6, puis touche vers le haut pour sélectionner le dernier item du menu de configuration.

## 11.4 Tableau formel

Le tableau de  $\chi$ CAS est un tableau utilisant la syntaxe standard des tableaux pour les références de cellule, mais il est capable de travailler avec des données exactes (par exemple fraction ou racines carrées) et symboliques (par exemple on peut calculer la dérivée d'une cellule du tableau), et on dispose de l'ensemble des commandes de Xcas, ainsi que les fonctions définies par l'utilisateur.

Taper F6 1 (ou la touche  $S \leftrightarrow D$ ) pour afficher la liste des applications additionnelles, sélectionner le tableau et valider avec EXE.

Pour donner une valeur fixe à une cellule, tapez simplement un nombre. Vous pouvez aussi taper une chaîne de caractère ou un autre objet reconnu par Xcas. Si vous saisissez une liste ou une commande renvoyant une liste, celle-ci remplira plusieurs cellules (en allant vers le bas ou vers la droite selon la configuration du tableau). Par exemple `range(10)` remplira 10 cellules consécutives avec les entiers de 0 à 9 (pour saisir `range`, vous pouvez utiliser le menu rapide F1).

Pour définir le contenu d'une cellule en faisant référence à d'autres cellules, taper = puis utilisez la syntaxe habituelle des tableaux, les caractères : et \$ sont accessibles depuis le menu F3 edit, : est aussi accessible par shift  $\rightarrow$ . On peut sélectionner une cellule lorsqu'on édite le contenu d'une autre cellule en déplaçant le curseur vers le haut ou vers le bas, puis vers n'importe quelle direction. Pour sélectionner une plage de cellules, déplacez-vous vers un des coins de la plage, taper shift-CLIP, puis déplacez-vous vers l'autre coin et tapez EXE.

Dans la définition d'une cellule, vous pouvez utiliser toutes les commandes de Xcas via le menu F4 ou les menus rapides (F1 à F6 éventuellement précédés de alpha ou shift) ou shift CATALOG. Vous pouvez aussi utiliser les structures de contrôle Xcas (test/boucle) ainsi que des fonctions Xcas que vous avez définies. Attention, seules les fonctions définies dans l'interpréteur Xcas sont reconnues (quelle que soit la syntaxe utilisée), les programmes écrits en MicroPython ne sont pas reconnus.

Une cellule peut contenir une commande renvoyant un résultat graphique. Pour voir le graphique correspondant à toutes les cellules du tableau ayant un résultat graphique, taper la touche  $S \leftrightarrow D$ .

## 12 Copyright et Remerciements

– Giac et  $\chi$ CAS, noyau de calcul (c) B. Parisse et R. De Graeve, 2019.

- Interface de  $\chi$ CAS adaptée par B. Parisse à partir de l'interface utilisateur du code source d'Eigenmath créée par Gabriel Maia pour la 90 et par Mike Smith et Nemh pour la 35eii, et de l'interface utilisateur de Xcas.
- License d'utilisation de  $\chi$ CAS : GPL2. Voir le détail des conditions dans le fichier `LICENSE.GPL2` de l'archive `khicasio.zip` ou sur la page GPL2 du site de la Free Software Foundation. Le code source de  $\chi$ CAS ainsi que des bibliothèques `libtommath` et `USTL`, se trouvent sur la section Casio de ma page web (cf. la section 13)
- Remerciements à tous les membres actifs de tiplanet et de Planète Casio pour l'aide qu'ils m'ont apportée en répondant à mes questions ou/et en testant  $\chi$ CAS, en particulier à LePhenixNoir pour son expertise et son aide technique pour la 35 et la 90, Redoste pour le debugger, Nemhardy pour la 90, et à critor pour les articles, tests et la diffusion de KhiCAS via tiplanet. Remerciements à tous les contributeurs du Prizm programming portal. Remerciements à Pavel Demin pour les astuces de compilation ayant permis d'économiser 135K environ.
- Remerciements à Camille Margot pour l'intérêt porté au projet et à Casio France pour le soutien par le prêt de calculatrices Graph 90/35 et l'attribution d'une licence d'utilisation de l'émulateur.

## 13 À propos du développement.

### 13.1 Install rapide sous linux

Installer la bibliothèque dynamique `libmpfr.so.4` dans `/usr/local/lib`, vérifiez que `/usr/local/lib` est dans la liste des chemins de `/etc/ld.so.conf` et exécutez `sudo ldconfig`. Puis désarchivez `casiolocal.tgz`, c'est une version du cross-compiler gcc avec certaines bibliothèques (`libc`, `ustl`, `tommath`). Pour créer des addins pour la 90, vous devez aussi installer `mkg3a`.

### 13.2 Source de giac

Le source pour la 90 est `giacbf.tgz` (avec support des flottants multi-précision). Il devrait se compiler avec `make`. La version light en un seul fichier est `giac90.tgz`. Pour la 35gii, c'est `giac35.tar.bz2` et on fait `make` dans le répertoire `giac35/src0`.

Si quelque chose ne fonctionne pas, voici quelques détails. Pour développer l'add-in, j'ai installé le cross-compiler gcc pour processeur `sh3eb`, en m'inspirant de ce tutoriel. Téléchargez gcc puis (remplacer la version de gcc par la version téléchargée)

```
../gcc-5.3.0/configure --target=sh3eb-elf --prefix="$HOME/opt/sh3eb-elf" --disab
```

Malheureusement, il n'y a pas de support pour `sh3eb` dans la `newlib` (bibliothèque C) fournie avec gcc, encore moins pour la `libstdc++`.

J'ai donc installé `libfxcg.tar.gz`, quelque peu modifiée par mes soins (corrections de plusieurs petits bugs dans la bibliothèque C, ajout de fonctions manquantes comme `qsort`, ...), à récupérer dans ce répertoire (désarchiver et compiler avec `make`). Toujours dans ce répertoire, on trouve `tommath.tgz` (gestion des entiers multi-précision) et `ustl.tar.gz` (implémentation de la standard template library) que j'ai

du pas mal modifier pour le faire fonctionner avec sh3eb-elf-g++, avec un résultat partiel qui est suffisant pour porter Giac (support de vector/string/map mais pas des flux E/S fichiers, j'ai créé à part un fichier iostream pour avoir un ersatz de cin/cout, cf. le répertoire ustl). Tout ces fichiers se désarchivent et on compile en principe avec make.

Comme il n'y a pas de support du compilateur pour la newlib et la libstdc++ et pour éviter les problèmes d'ordre d'initialisation des variables statiques, ce portage de Giac se caractérise par l'absence de variables statiques classes, il n'y a que des variables statiques C. La très grande majorité des variables globales (tous les noms de commandes Xcas par exemple) sont déclarées via des alias sur des structures C en constantes pour ne pas occuper de place en mémoire vive. Il a fallu tenir compte de l'endianness du CPU par rapport à d'autres portages de Giac (plus précisément le CPU peut se comporter des deux manières little ou big endian mais on est obligé de choisir le même ordre que celui de l'OS Casio, et c'est l'inverse de celui qu'on rencontre sur les architectures Intel).

### 13.3 Mise au point

Les etapes pour utiliser gdb pour Casio sous linux avec wine sont les suivantes :

1. installer wine si nécessaire

2. installer l'emulateur casio avec wine avec une commande qui ressemble à

```
wine /chemin_vers/fx-CG_Manager_PLUS_Subscription_for_fx-CG50series_Ver.3.40
```

3. installer l'équivalent de gdb-server

```
cd .wine/drive_c/Program\ Files\ \ (x86\)/CASIO/fx-CG\ Manager\ PLUS\ Subscri
mv CPU73050.dll CPU73050.real.dll
wget https://www-fourier.univ-grenoble-alpes.fr/~parisse/casio/CPU73050.dll
cp CPU73050.dll CPU73050.dbg.dll
```

(j'ai mis la dll sur ma page, parce que j'ai eu un peu de mal a le compiler, le source est [https://github.com/redoste/fx-CG50\\_Manager\\_PLUS-gdbserver](https://github.com/redoste/fx-CG50_Manager_PLUS-gdbserver)).

4. compiler gdb pour sh3 :

```
wget https://ftp.gnu.org/gnu/gdb/gdb-11.1.tar.gz
cd casio (ou autre repertoire de build)
tar xvfz ../gdb-11.1.tar.gz
mkdir sh3eb-gdb
cd sh3eb-gdb
../gdb-11.1/configure --srcdir=../gdb-11.1 --target=sh3eb-elf
```

(ajouter --prefix=chemin dans la commande configure si on n'a pas les droits sur /usr/local/bin). Puis

```
make
sudo make install
```

Ensuite on crée deux scripts, casioemu pour lancer en mode normal

```

#! /bin/bash
cd ~/.wine/drive_c/Program\ Files\ \ (x86\)/CASIO/fx-CG\ Manager\ PLUS\ Subsc
/bin/cp CPU73050.real.dll CPU73050.dll
cd
wine "C:\Program Files (x86)\CASIO\fx-CG Manager PLUS Subscription for fx-CG
et casiodbg pour lancer en mode debug

```

```

#! /bin/bash
cd ~/.wine/drive_c/Program\ Files\ \ (x86\)/CASIO/fx-CG\ Manager\ PLUS\ Subsc
/bin/cp CPU73050.dbg.dll CPU73050.dll
cd
wine "C:\Program Files (x86)\CASIO\fx-CG Manager PLUS Subscription for fx-CG

```

5. Pour utiliser l'émulateur en mode normal taper simplement casioemu, pour le mode debug taper

```

casiodbg
sh3eb-elf-gdb
target remote localhost:31188

```

Ou directement pour utiliser avec les infos de debug de emucas.elf

```
sh3eb-elf-gdb -i=mi -ex "target remote localhost:31188" emucas.elf
```

Pour mettre un point d'arrêt utiliser b ou hb (hbreak (hardware break, nécessaire pour le premier point d'arrêt).