

Tutoriel xcas

Bernard Parisse
Institut Fourier
CNRS UMR5582
Université Grenoble I

2 mars 2004

Résumé

Ce texte a pour but une prise en main rapide de `xcas`, logiciel libre de calcul formel généraliste et de géométrie dynamique.

Table des matières

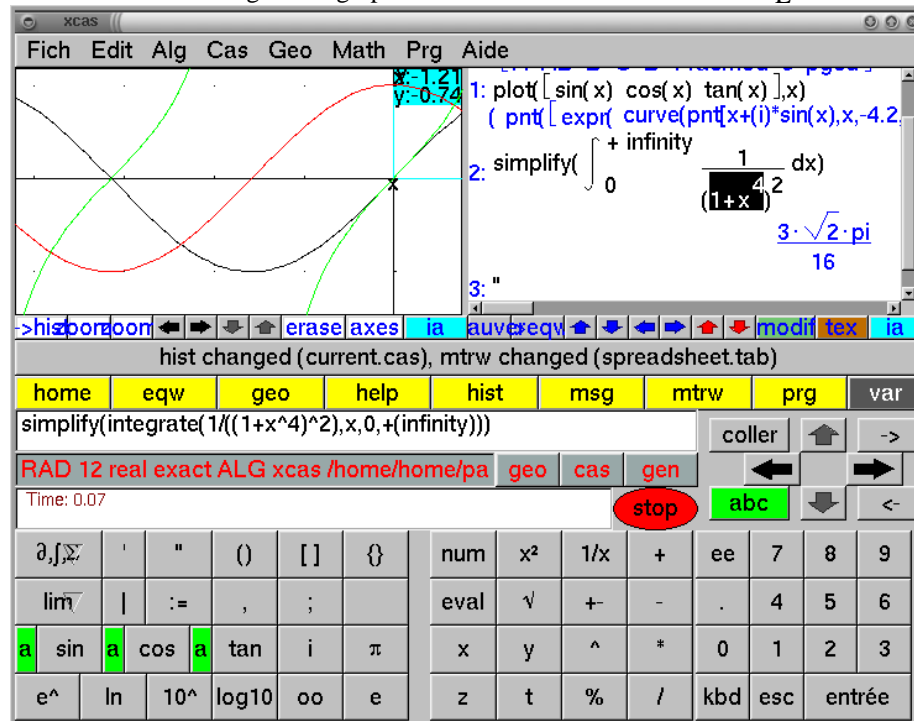
1	Introduction	2
2	Installation.	3
2.1	Installation de xcas avec Windows	3
2.2	Installation de xcas avec Mac OS X	4
2.3	Installation de xcas avec Linux	4
2.3.1	Si vous utilisez Gnu/Debian, Mandrake Linux ou RedHat Linux	4
2.3.2	Si vous avez le mot de passe de <code>root</code>	4
2.3.3	Si vous n'avez pas le mot de passe <code>root</code>	5
2.3.4	Aide en ligne	5
2.3.5	Impression	5
2.3.6	Version en ligne de commande	5
3	Mise à jour de <code>xcas</code> (Windows et Linux)	6
4	Note pour les utilisateurs de Maple.	6
5	Note pour les utilisateurs de TI89/TI92/TI Voyage 200.	7
6	Premiers pas	8
7	La barre de menu	11
8	L'éditeur d'équation	11

9	La géométrie et le graphique	13
9.1	Construction géométrique	14
9.2	Les paramètres.	15
9.3	Réglage de la fenêtre graphique	15
10	Le tableur	17
11	Les scripts	18
12	Le mode programme	18
13	Imprimer, générer des fichiers <code>tex</code>.	19
14	Configuration	19
15	Quelques commandes bien utiles pour <code>xcas</code>, <code>Maple</code> et <code>Mupad</code>.	22
15.1	Commandes scalaires	22
15.2	Algèbre linéaire.	23
16	Conteneurs	24
16.1	Programmation.	24
16.2	TP	25
17	Installation de la librairie <code>giac</code> sous <code>Linux</code>	27
18	Programmation en <code>C++</code> avec <code>giac</code>	27
18.1	Le type <code>gen</code>	28
18.2	Programmes et modules dynamiques.	28
19	Exemples en <code>C++</code>.	28
19.1	Programme (pgcd de 2 entiers).	28
19.2	Module dynamique.	29
20	Compléments.	30

1 Introduction

`xcas` est un logiciel qui rassemble dans une même interface plusieurs aspects des mathématiques : calcul numérique, calcul formel (ou symbolique), représentation graphique de fonctions, géométrie dynamique, tableur. Il possède un langage de programmation de haut niveau qui peut lire et exécuter des programmes simples écrits pour les logiciels `Maple`, `Mupad` ou pour les calculatrices `TI89/TI92/Voyage 200`. On peut partager l'écran pour avoir différentes vues d'un même objet mathématique (par exemple coordonnées d'un point géométrique, vue numérique et graphique pour des données

statistiques, ...). `xcas` génère des fichiers \LaTeX et permet ainsi très facilement la création et l'insertion de figures et graphes de fonctions dans vos fichiers \LaTeX .



2 Installation.

La page home de `xcas` est :

http://www-fourier.ujf-grenoble.fr/~parisse/giac_fr.html
 Vous y trouverez toutes les instructions nécessaires pour installer `xcas` selon votre système d'exploitation.

2.1 Installation de `xcas` avec Windows

Sur un PC avec Windows, récupérez le fichier :

`ftp://ftp-fourier.ujf-grenoble.fr/xcas/xcaszip.exe`
 double-cliquez sur ce fichier, le programme d'extraction de l'archive `xcas` se lance et vous propose de désarchiver `xcas` dans le répertoire de `xcaszip.exe`, vous devriez ici remplacer ce répertoire par `C:\` ce qui désarchivera `xcas` dans le répertoire `C:\xcas` qui sera créé si nécessaire. Si vous désarchiveriez `xcas` dans un autre répertoire vous devrez modifier les fichiers `xcasfr.bat` et `runxcas.fr` pour refléter les chemins d'accès sur votre disque dur (voir ci-dessous). Vous pouvez réaliser une installation globale en ajoutant le répertoire d'installation à la variable `PATH` ou en désarchivant `xcas` dans un répertoire pointé par cette variable. Dans ce cas, vous devrez

modifier `xcasfr.bat` et `runxcas.fr` en indiquant respectivement les variables d'environnement correctes, par exemple si vous avez désarchivé `xcas` sur le disque dur D : dans le répertoire Maths

```
xcasfr.bat: set PATH=%PATH%;d:\Maths\xcas
runxcas.fr: export XCAS_ROOT=/cygdrive/d/Maths/xcas
```

Si vous voulez imprimer, vous devrez installer une distribution \LaTeX fonctionnelle, par exemple <http://www.fptex.org/>.

L'aide en ligne HTML est affichée par défaut par Internet Explorer, vous pouvez modifier le navigateur internet en changeant le fichier `ie.bat`. Attention, sur certaines versions de Windows, l'affichage automatique de l'aide HTML ne fonctionne pas, vous pouvez malgré tout ouvrir Internet Explorer et consulter la documentation qui se trouve par défaut dans le répertoire `c:\xcas\doc`.

2.2 Installation de xcas avec Mac OS X

Récupérez le fichier

```
ftp://ftp-fourier.ujf-grenoble.fr/xcas/xcas\_image.dmg.gz
```

Le fichier devrait se désarchiver automatiquement, puis l'image disque se monter et un disque `xcas` doit apparaître sur le bureau du Finder, si ce n'est pas le cas, double-cliquez sur le fichier `xcas_image.dmg`. À présent, double-cliquez sur l'icone disque `xcas`. Pour lancer `xcas`, double-cliquez sur le programme `xcas` du disque `xcas`. Cette version ne contient pas toutes les fonctionnalités des versions Linux ou Windows (par exemple l'éditeur de programmes n'existe pas).

2.3 Installation de xcas avec Linux

2.3.1 Si vous utilisez Gnu/Debian, Mandrake Linux ou RedHat Linux

Vous pouvez récupérer et installer une archive debian ou RPM (cf. lien sur la page home de `xcas`)

```
http://www-fourier.ujf-grenoble.fr/~parisse/giac_fr.html
```

2.3.2 Si vous avez le mot de passe de root

Récupérez le fichier :

```
ftp://ftp-fourier.ujf-grenoble.fr/xcas/xcas_root.tgz
```

et sauvegardez ce fichier dans le répertoire `/tmp`. Tapez ensuite dans une fenêtre de commande¹ :

```
su
puis votre mot de passe root, puis la touche Entree. Tapez ensuite :
cd /
```

¹Parfois appelé "xterm" ou "kterm" ou "shell", une fenêtre de commandes permet d'exécuter des commandes Unix telles que `ls`, `cd`, Sur les installations Linux récentes, il est souvent nécessaire de lancer une fenêtre de commandes. Par exemple avec l'environnement graphique KDE, sélectionnez dans le menu K exécuter une commande, tapez `xterm` puis la touche Entree.

```
tar xvfz /tmp/xcas_root.tgz
ldconfig
exit
```

Pour lancer le programme, tapez ensuite :

```
xcas &
```

Si le programme n'est pas trouvé, déconnectez-vous et reconnectez-vous et tapez à nouveau : `xcas &`. Si cela ne fonctionne toujours pas, vérifiez que les fichiers suivants sont installés : `xcas` dans le répertoire `/usr/local/bin`, `aide_cas` dans `/usr/local/share/giac`. Si ce n'est pas le cas, recommencez l'installation, sinon ajoutez la ligne :

```
export PATH = "$PATH :/usr/local/bin/"
dans le fichier /etc/profile ou dans votre fichier ~/.bashrc.
```

2.3.3 Si vous n'avez pas le mot de passe root

Récupérez le fichier :

```
ftp://ftp-fourier.ujf-grenoble.fr/xcas/xcas_user.tgz
```

et sauvegardez ce fichier dans votre répertoire courant. Tapez ensuite dans une fenêtre de commande :

```
cd ~
tar xvfz xcas_user.tgz
```

Pour lancer le programme, tapez ensuite :

```
./xcas &
```

2.3.4 Aide en ligne

L'aide en ligne est affichée en utilisant le navigateur Internet lu dans la variable d'environnement `BROWSER` ou par `mozilla` si cette variable n'est pas déclarée. Pour changer de navigateur, il vous suffit d'éditer votre fichier `~/.bashrc` ou le fichier global `/etc/profile` et d'y ajouter la ligne :

```
export BROWSER=konqueror
```

(remplacez `knoqueror` par le nom de votre navigateur habituel).

Si cela ne fonctionne pas à la prochaine connexion, cela signifie probablement que vous utilisez `tcsh`, changez alors votre fichier `~/.tcshrc` et ajoutez la ligne :

```
setenv BROWSER konqueror
```

2.3.5 Impression

Vous devez avoir une installation \LaTeX fonctionnelle. Si ce n'est pas le cas, le plus simple est d'installer les paquetages `tetex*` de votre distribution : `tetex`, `tetex-latex`, `tetex-xdvi`, `tetex-dvips`.

2.3.6 Version en ligne de commande

Elle se lance par la commande `giac` ou `./giac`. Elle peut s'utiliser de manière autonome, dans une session du logiciel TeXmacs (<http://www.texmacs.org>) ou dans `mupacs`.

3 Mise à jour de `xcas` (Windows et Linux)

Pour mettre à jour `xcas` :

- Si vous utilisez Linux, Windows NT, windows 2000, windows XP vérifiez que vous avez les droits d'écriture dans le répertoire d'installation de `xcas`
- Si nécessaire, activez votre connexion Internet
- Lancez `xcas` puis choisissez dans le menu `Fich->Mettre a jour xcas`

4 Note pour les utilisateurs de Maple.

`giac/xcas` propose une compatibilité avec les fonctions Maple utiles jusqu'au niveau licence de mathématiques environ. Pour l'activer, il suffit au premier lancement de `xcas` de cliquer sur Maple lorsqu'on vous demande `Choose initial configuration`. Vous pouvez aussi l'activer au cours d'une session par le menu `Edit->Préférences->Configuration cas`

en cliquant sur `Prog style` et en choisissant Maple puis valider le choix par OK.

Pour charger une session Maple dans `xcas`, allez dans le menu `Fich->Import session->maple worksheet` puis sélectionnez le nom du fichier.

Pour charger un programme écrit en langage Maple, utilisez le menu `Fich->Import session->Maple text` puis sélectionnez le nom du fichier.

Il est fortement conseillé de lire la section 6 (Premiers pas) ci-dessous, l'interface de `xcas` étant un peu différente de celle de maple, en particulier :

- les commandes sont numérotées, la commande est alignée à gauche juste après son numéro, et la réponse alignée à droite par défaut. La commande et la réponse sont écrites en bleu (maple écrit la commande en rouge)
- l'entrée d'une nouvelle commande peut se faire en mode 2-d dans l'historique ou en mode 1-d dans la ligne de commande (cf. la figure section 6). Si vous modifiez l'historique, utilisez la touche flèche vers le haut pour valider le changement en cours et faire un changement ailleurs, lorsque vous avez fait tous les changements souhaités appuyez sur la touche `Entree` ce qui réexécute la feuille de calcul modifiée à partir de la première modification.
- lorsqu'on modifie une commande de l'historique, toutes les commandes ultérieures sont recalculées (contrairement à maple où il faut valider à nouveau chacune des commandes ultérieures). On peut soit modifier directement dans l'historique 2-d (cf. ci-dessus) soit sélectionner le niveau souhaité et cliquer sur le bouton `modif` pour modifier dans la ligne de commande.
- les commandes graphiques n'affichent pas leurs résultats (la fenêtre `geo` pour les graphes 2-d et une), on y accède en cliquant sur le bouton jaune `geo` (cliquer sur le bouton jaune `hist` pour revenir à l'historique). La fenêtre graphique est interactive (voir ci-dessous la géométrie interactive). L'affichage d'un graphe de fonction n'efface pas le précédent, utilisez `erase` pour effacer entre deux graphes. Les graphes sont imprimés à chaque commande dont le nom commence par `plot`, ou par l'appel de la commande `graph2tex()` ou `graph3d2tex()` pour les graphes en 3-d.

- le signe % est reconnu, mais pas %% ou %%%, utiliser `ans (-2)` ou `ans (-3)` à la place.
- l'affichage par `print` n'est pas fait dans l'historique mais dans la fenêtre de messages (bouton jaune `msg`). L'historique n'affiche que la valeur de retour d'un programme (renvoyée par `RETURN`). Ceci permet de bien distinguer valeur de retour et affichage dans un programme.

Si vous ne pensez pas utiliser le tableur ou la géométrie interactive, vous pouvez aussi utiliser `texmacs/giac` qui ressemble plus à l'interface de `maple`.

5 Note pour les utilisateurs de TI89/TI92/TI Voyage 200.

`xcas` propose un mode de compatibilité avec les calculatrices graphiques formelles de Texas Instrument. Pour l'activer, il suffit au premier lancement de `xcas` de cliquer sur `TI89` lorsqu'on vous demande `Choose initial configuration`. Vous pouvez aussi l'activer au cours d'une session par le menu `Edit->Préférences->Configuration cas` en cliquant sur `Prog style` et en choisissant `TI89` puis valider le choix par `OK`.

Vous pouvez charger des programmes pour `TI89/TI92` ou `Voyage 200` au format archive groupe (noms de fichiers terminant par `.89g` ou `.92g`), au format programme ou fonction (`.89f`, `.92f`, `.89p`, `.92p`), ou au format texte. Si vous recopiez un programme dans un livre, utilisez un éditeur de texte (par exemple `Wordpad`, `Blocnote`, `Ultraedit`, `emacs` sous `windows` ou `emacs` ou `vi` sous `linux`) et recopiez le texte tel quel y compris les `:` en début de ligne. Si le programme est dans votre calculatrice, vous pouvez utiliser le logiciel `GraphLink` de Texas Instrument pour le transférer vers votre ordinateur. Allez alors dans le menu `Fich->Executer fichier` et sélectionnez le nom du fichier.

Il est fortement conseillé de lire la section 6 (Premiers pas) ci-dessous, l'interface de `xcas` étant un peu différente de celle des calculatrices `TI`, en particulier pour les commandes graphiques.

6 Premiers pas

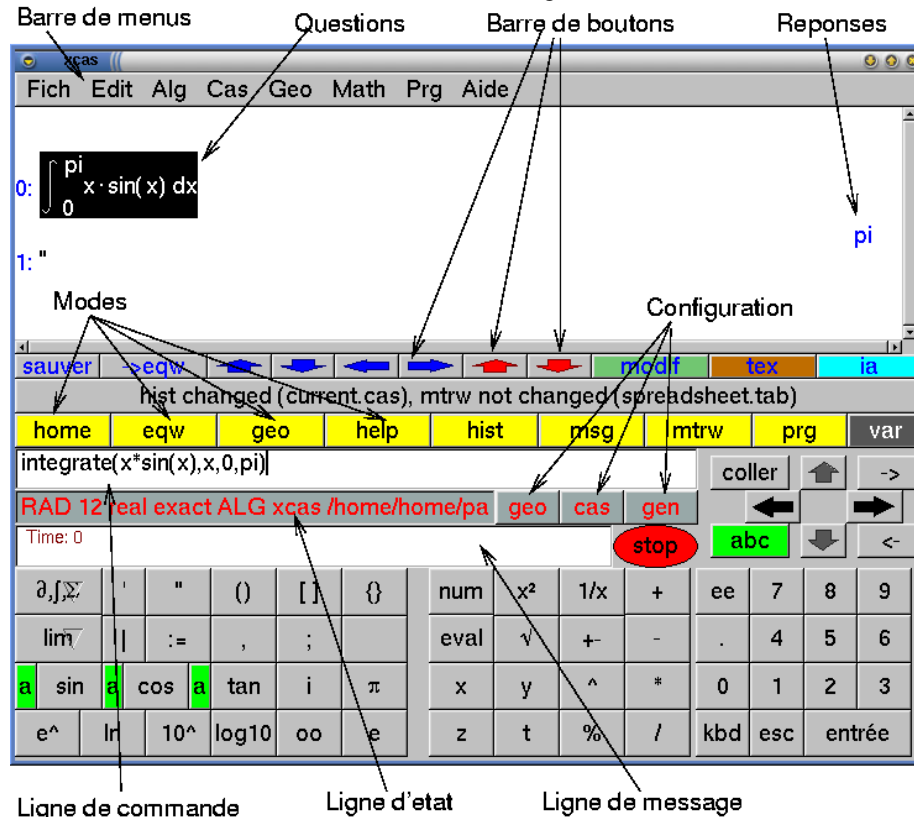
On suppose maintenant que `xcas` est installé sur votre ordinateur. Sous Unix, tapez dans une fenêtre de commandes :

```
xcas &
```

Sous Windows, utiliser l'explorateur pour aller dans le répertoire où est installé `xcas` puis cliquez sur le fichier `xcasfr.bat` (vous pouvez aussi créer un raccourci et le mettre sur le bureau).

Lors de votre première exécution de `xcas`, on vous propose de choisir une configuration initiale parmi Xcas, Maple ou TI89 (il s'agit de la syntaxe pour le langage de programmation). Si vous avez déjà utilisé maple ou la TI89 choisissez le mode correspondant.

Vous devez voir une fenêtre ressemblant à la figure ci-dessous :



La fenêtre est divisée de haut en bas en :

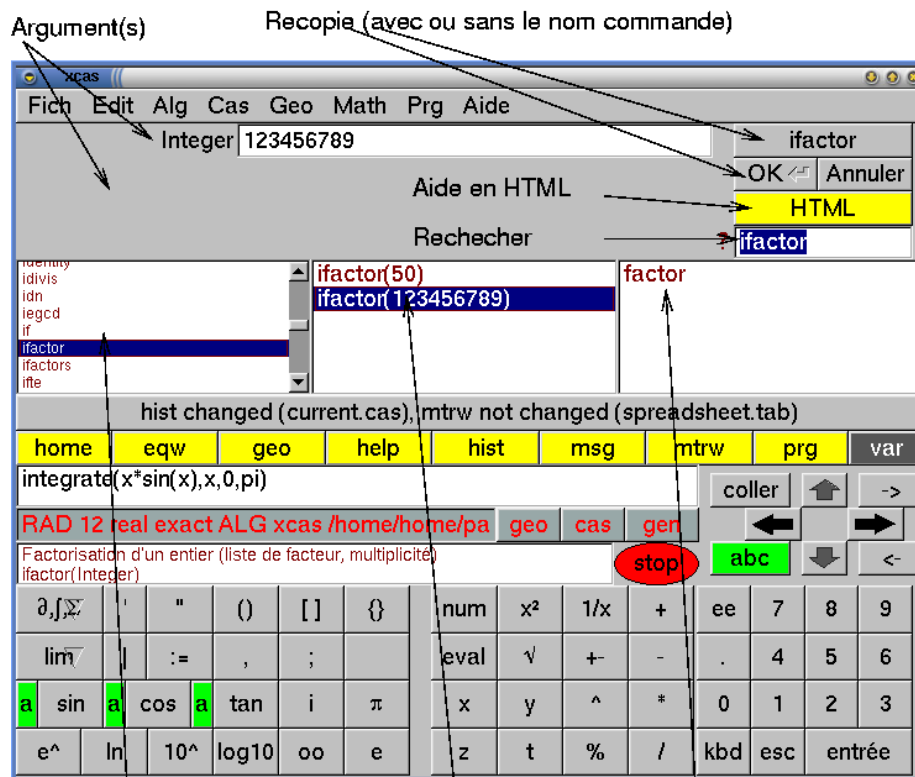
- la barre de menu
- selon le mode, l'éditeur d'équation (eqw), le graphique (geo), l'aide en ligne (help), l'historique (hist), les messages (msg), l'éditeur de programmes (prg), le tableur (mtrw), et le débogueur (Linux et Mac OS X). Plusieurs vues peuvent être affichées simultanément (menu Edit->Windows->1/2/3 Windows).
- la barre de boutons du mode

- l'état de sauvegarde de la session et du tableur (qui peut être remplacé par un bandeau dont la fonction est analogue au bandeau des calculatrices HP et Casio)
- une ligne de boutons jaunes qui permet de changer le mode (hist, eqw, mtrw, etc.)
- la ligne de commande
- la ligne d'état et les boutons pour changer l'état du logiciel
- la ligne de message (le bouton msg permet de voir ces messages en entier)
- le clavier de calculatrice scientifique, les touches de direction et le pavé numérique

Le déroulement normal d'une session consiste à entrer des commandes dans la ligne de commande (ou directement dans l'historique), en s'aidant au besoin des éditeurs (d'équation, de matrice, de programme) et de l'aide en ligne.

Par exemple, pour calculer $1234 * 5678$ on tape dans la ligne de commande $1234 * 5678$ puis la touche `Enter` ou `Entrée`. Ce mode de saisie est semblable à celui des logiciels de calcul formel usuels ou des calculatrices.

Les commandes du logiciel sont regroupées par thème dans la barre de menu. Par exemple pour factoriser un entier, on ouvre le menu `Alg`, sous-menu `Entier`, item `ifactor`. Une courte aide en ligne apparaît alors dans la ligne de messages. Pour avoir des exemples, on peut cliquer sur le bouton jaune `help`, ce qui fait apparaître des champs de saisie pour chaque argument, des exemples (que l'on peut recopier dans les champs de saisie en cliquant dessus, et que l'on peut ensuite modifier), et des commandes sur le même thème ainsi que l'index global de toutes les commandes de `xcas` à gauche.



Liste des commandes

Exemples

Commandes proches

L'éditeur d'équation sert à saisir des expressions mathématiques un peu compliquées en évitant d'avoir à saisir des parenthèses grâce à un affichage 2-d des expressions mathématiques. Ce mode de saisie existe aussi dans l'historique. Il permet également d'exécuter une liste de commandes préparées à l'avance ligne après ligne (menu Fich->Executer script).

Le mode graphique permet d'afficher des courbes dans le plan, et de tracer des figures géométriques.

Le tableur est un tableur minimal comparé aux tableurs des suites bureautiques. Mais c'est un tableur dont les cellules peuvent être des expressions formelles et on peut utiliser le langage de programmation de xcas pour calculer le contenu d'une cellule. On peut s'en servir pour avoir des tables de valeurs de fonctions ou de suites récurrentes (menu Edit->mtrw->tablefunc/tableseq). Il peut aussi faire office d'éditeur de matrice (on peut également utiliser l'éditeur d'équation si une matrice contient des expressions mathématiques complexes).

Le mode programme (Linux et Windows) est destiné à écrire des petits programmes avec le langage de programmation de xcas. Pour des programmes plus conséquents, on préférera utiliser un éditeur comme emacs ou nedit puis on importera les programmes dans une session xcas avec la commande read. Taper votre programme puis cliquez sur le bouton tester pour voir s'il est correct, si c'est le cas, cliquez

sur le bouton `hist` pour le recopier dans l'historique. Sous linux et Mac OS X, on peut exécuter un programme en pas-à-pas avec le débogueur (commande `debug`), et regarder l'évolution des variables, ce qui permet de corriger des erreurs.

7 La barre de menu

Elle se divise en deux parties : `Fich` ou `File` et `Edit` d'une part et les commandes mathématiques d'autre part (`Alg`, `Cas`, `Math`, `Prg`). On présente ici `File` et `Edit`.

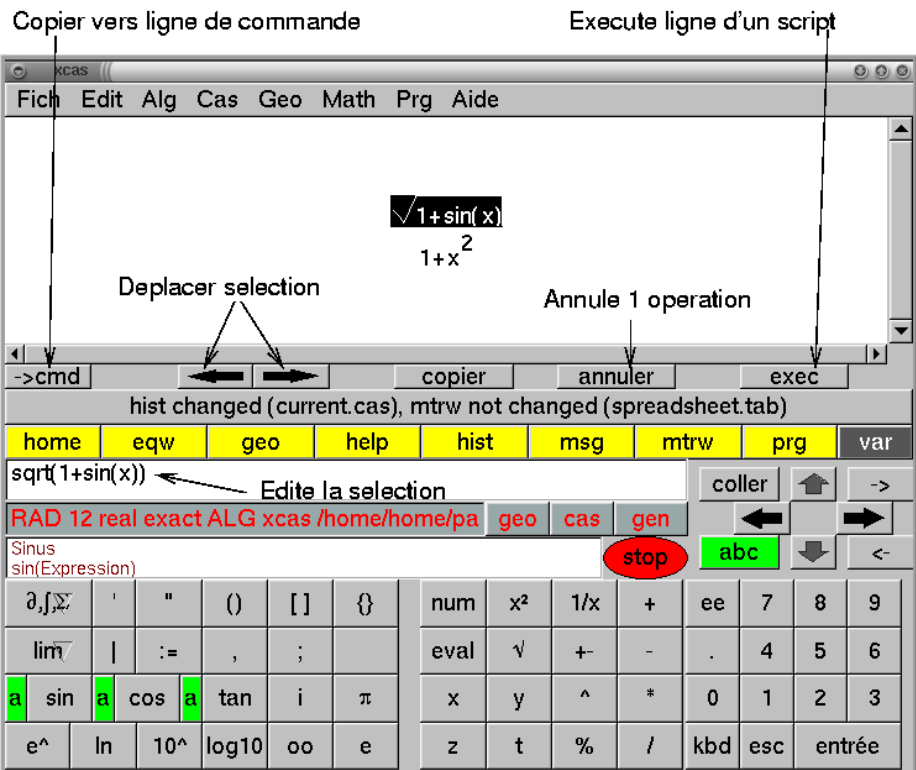
Le menu `File` permet de sauvegarder et charger des données dans les différents modes (historique, tableur, éditeur d'équation, programme). On peut aussi imprimer l'historique, changer la langue de l'aide en ligne, sauvegarder la configuration courante et même mettre à jour `xcas` (à condition d'avoir une connexion Internet).

Le menu `Edit` :

- Copier : permet de copier de ou vers la ligne de commande de l'historique
- Préférences : permet le réglage de la configuration par thème (`cas`, géométrie/graphique, général). Analogue aux boutons de la ligne d'état.
- Effacer : au choix l'historique, toutes les variables à 1 lettre, la ligne de commande (raccourci clavier `Escape`)
- Windows : commande de changement de vue analogue aux boutons jaunes du bandeau, ainsi que commandes de partage d'écran
- `geo` : permet de zoomer
- `hist` : permet d'insérer dans l'historique le graphe 2-d ou 3-d actuel
- `msg` (Voir historique comme) : écrit dans la fenêtre de message le niveau actuel de l'historique en utilisant une syntaxe proche de Maple, MuPAD TI89 ou `xcas`
- `mtrw` (Tableur) : commandes d'édition spécifiques au tableur. A noter les commandes `table` et `tableseq` qui permettent de créer facilement le tableau de valeurs d'une fonction ou d'une suite récurrente.
- `prg` : commandes de recherche/remplacement de l'éditeur de programmes.
- Préférences : permet de changer la configuration

8 L'éditeur d'équation

Pour lancer l'éditeur d'équation, cliquez sur le bouton jaune `eqw`. Vous pouvez aussi sélectionner une expression dans l'historique et la copier avec le bouton bleu `>eqw`.



L'éditeur d'équation stocke l'objet à éditer sous forme d'un graphe dont les noeuds sont les opérateurs et les feuilles les arguments des opérateurs. On peut éditer directement les arguments en cliquant à l'endroit désiré, on effectue la modification et on valide en tapant sur la touche flèche vers le haut ou en cliquant ailleurs avec la souris. On peut aussi éditer une sous-expression en la sélectionnant, ce qui la recopie en ligne de commande, on modifie alors la ligne de commande et lorsqu'on tape la touche Entrée, la sélection est remplacée par la ligne de commande. Notez que les flèches de direction permettent de modifier la sélection en se déplaçant dans le graphe représentant l'équation. On peut aussi :

- déplacer la sélection vers la droite ou la gauche (boutons de la barre de boutons) si elle est l'argument d'une somme ou d'un produit,
- appliquer un opérateur à la sélection (par exemple le bouton `eval` permet d'évaluer une sous-expression en place, `simplify` pour simplifier la sélection, `factor` pour factoriser la sélection ou sur le même principe, toutes les commandes de réécriture).
- supprimer un opérateur unaire en tapant `shift` puis `backspace`
- lorsque le curseur est actif, l'appui sur la touche `"` permet de basculer d'une chaîne de caractères à l'expression qu'elle représente.

Pour copier une sous-expression ailleurs, la sélectionner, cliquer sur `copier`, sélectionner l'endroit où la recopier et cliquer sur `coller/paste`.

Pour quitter l'éditeur d'équation, taper la touche `->cmd` qui recopie toute l'expres-

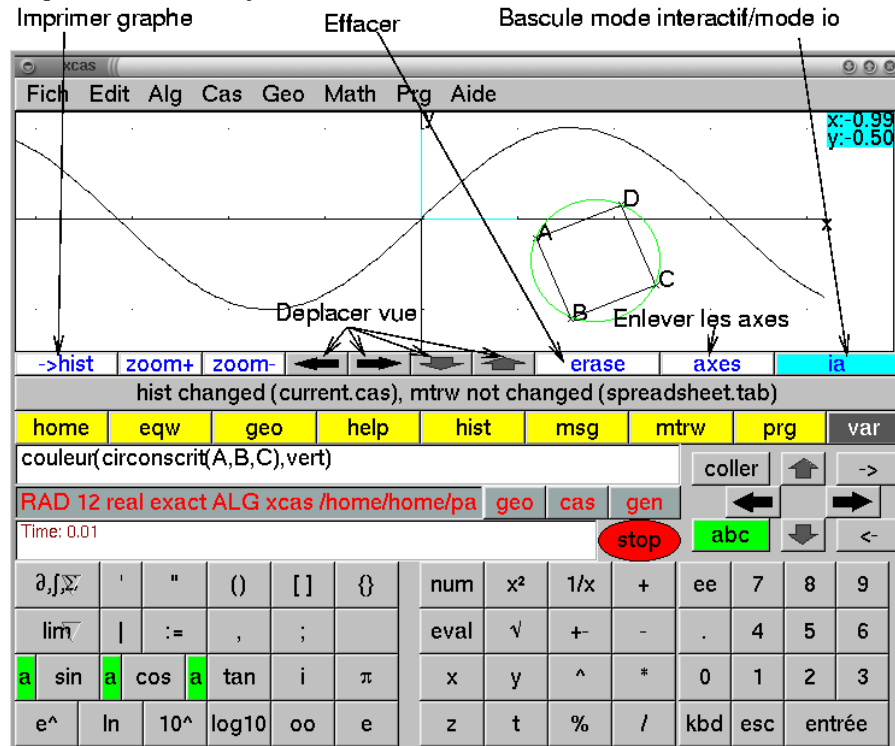
sion en ligne de commande ou `shift -> cmd` qui ne recopie que la sélection.

Exemple : entrer dans l'éditeur d'équation, tapez `x` puis flèche vers le haut, puis la touche `^`, puis `100`, puis flèche vers le haut, puis à nouveau flèche vers le haut pour sélectionner toute l'expression x^{100} , puis la touche `-` puis `1`, puis flèche vers le haut deux fois pour sélectionner $x^{100} - 1$ (remarque : il est aussi possible d'entrer directement $x^{100} - 1$ puis flèche vers le haut, mais le mode décrit précédemment permet de saisir des expressions plus complexes sans jamais devoir utiliser des parenthèses). Sélectionnez `factor` dans le menu `Cas`, puis flèche vers le bas, flèche à droite, etc. pour déplacer la sélection. Sélectionnez $x^2 + 1$ au curseur ou à la souris, appuyez sur la touche `^` puis `2` puis la touche flèche vers le haut deux fois.

Remarque : on peut éditer l'historique de la même façon que dans l'éditeur d'équation. L'appui sur la touche `Entree` (ou `Enter`) valide alors tous les changements effectués et recalcule l'historique à partir de la première modification effectuée. L'appui sur la touche `Echap` (ou `Esc`) annule tous les changements effectués.

9 La géométrie et le graphique

Ce mode est activé automatiquement lorsqu'on tape une commande à sortie graphique (cf. menu `Geo` ou menu `Maths`, sous-menu `Graphes`). On peut aussi l'activer en cliquant sur le bouton jaune `geo`.



Dans ce mode, on peut interagir avec l'historique de manière graphique. On peut aussi

faire varier des paramètres scalaire définis par la commande `element` (ici `a := element(1..2)`) et observer de manière interactive comment varient les objets graphiques qui dépendent de ce paramètre. On peut facilement zoomer sur une partie de l'écran graphique.

9.1 Construction géométrique

Lorsqu'on déplace la souris, la position du curseur dans le système de coordonnées est mise à jour dans le coin supérieur droit. On peut redéfinir le système de coordonnées en cliquant sur le bouton `geo` de la ligne d'état.

Un click de souris permet de tracer un point. On peut aussi tracer un segment en enfonçant un bouton de la souris à une extrémité et en le relâchant à l'autre extrémité. On peut déplacer un point existant (et toutes les constructions géométriques qui en dépendent) en cliquant sur un objet géométrique pour le sélectionner puis en enfonçant un bouton de la souris sur cet objet et en déplaçant la souris bouton enfoncé. Relâcher la souris lorsque la nouvelle position de l'objet est atteinte. Pour annuler un mouvement relâcher le bouton de la souris au-dehors de la zone d'affichage géométrique.

Exemple 1 :

Enfonchez un bouton de la souris, déplacez la souris et relâchez la souris, vous avez tracé un segment : le système nomme automatiquement les deux extrémités et le segment et affiche le segment ainsi que les noms de points (en principe A , B). Recommencez à partir d'une des extrémités A ou B , le système reconnaît que l'origine est A ou B . Il est donc assez simple de tracer un triangle à la souris (en principe votre triangle à pour sommets A , B et C et pour côtés AB , BC et CA). Cliquez sur A puis déplacez A avec la souris (enfonchez le bouton de la souris en A et déplacez la souris), vous verrez l'ensemble de la figure bouger.

Les autres constructions se font en tapant des commandes, à l'aide du menu ou du bandeau de géométrie (`Geo`).

Exemple 2 :

Traçons le triangle ABC avec la souris puis le cercle circonscrit à ABC sans utiliser la commande `circonscrit` :

- tapez `D0:=`, puis dans le menu `Geo`, sous-menu `Lignes` sélectionnez alors `mediatrice` puis tapez `AB` (ou A , B) :
`D0:=mediatrice(AB)`
tapez `Enter` ce qui trace la médiatrice de $[AB]$ et la nomme $D0$ (on ne peut pas utiliser la lettre D seule, car D est un mot clef réservé pour la dérivée en tant qu'opérateur)
- de même pour E médiatrice de $[BC]$:
`E:=mediatrice(BC)`
- cliquez à l'intersection des deux médiatrices : en principe le système détecte votre intention, en cliquant sur la touche jaune `Geo` vous devez voir dans l'historique :
`F:=head(D0 intersect E)`
revenez alors dans l'écran graphique en cliquant à nouveau sur la touche jaune `Geo`. Notez que l'intersection de 2 objets géométriques est un ensemble ici

constitué d'un seul point, la commande `head` renvoie le premier (ici unique) point de l'intersection et stocke le résultat dans `F`

– tapez alors :

```
G:=cercle(F,A-F)
```

la commande `cercle` trace un cercle étant donné par deux points (son diamètre) ou donné par son centre (`F`) et son rayon (`A-F`).

Vous pouvez à nouveau tester l'aspect géométrie dynamique en faisant bouger le point `A` (note : un Pentium 166 est capable de recalculer plus de 20 fois par seconde cette construction, mais on verra moins de 20 images par seconde à cause du temps d'affichage).

9.2 Les paramètres.

En mode géométrie/graphique, la fenêtre est décomposée en deux parties : sur presque toute la fenêtre on affiche des objets géométriques, dans une colonne à droite, on voit en haut `x:` et `y:` suivis des coordonnées courantes de la souris (attention : si rien n'est affiché après `x` et `y` ou si vous changez la fenêtre `xcas` de position ou de taille cliquez en-dessous de `y:` dans cette colonne pour resynchroniser `xcas`).

Le reste de cette colonne à droite de la fenêtre graphique sert à afficher des paramètres et à faire varier leur valeur dans un intervalle. Par exemple, effectuez les commandes suivantes :

```
a:=element(-2..2,1)
```

```
b:=element(-3..3)
```

```
c:=element(-1..1)
```

```
plotfunc(a*x^2+b*x+c)
```

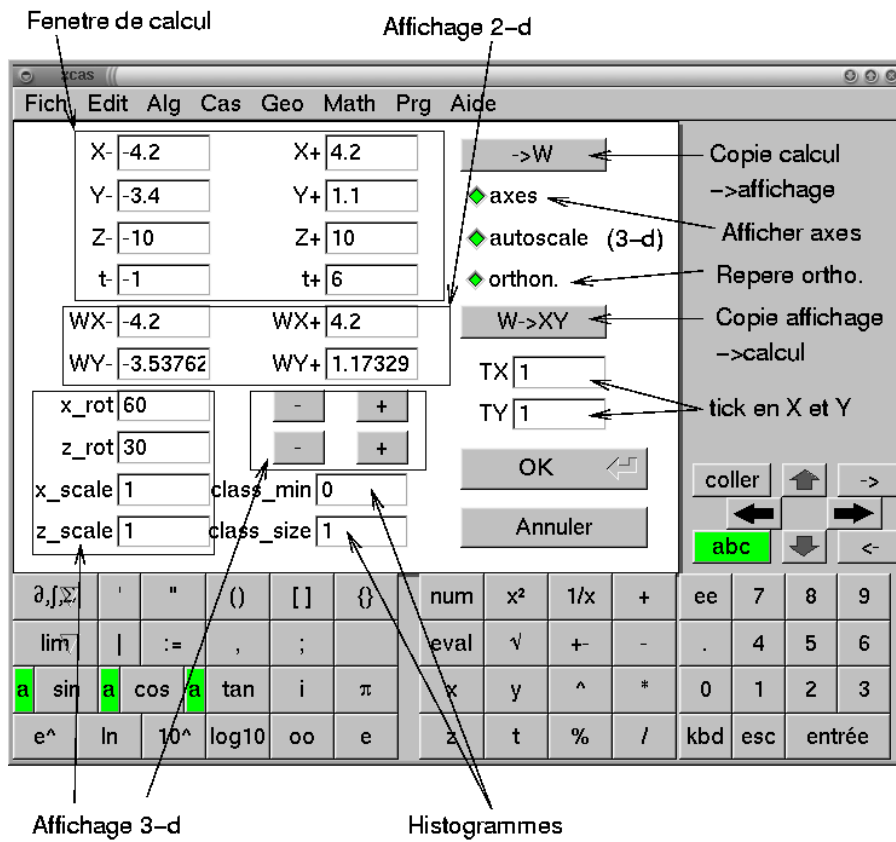
(`element` se trouve dans le menu `Geo`). Vous pouvez maintenant faire varier `a`, `b` et `c` à la souris et observer la modification de la parabole d'équation $y = ax^2 + bx + c$ en temps réel (avec une résolution horizontale de 300 pixels on peut calculer une vingtaine d'images par seconde sur un pentium 166).

9.3 Réglage de la fenêtre graphique

On peut zoomer ou déplacer la fenêtre graphique à l'aide des boutons de la barre de boutons. On peut aussi zoomer de manière interactive : on clique d'abord sur le bouton `io` pour passer en mode `io`. Lorsqu'on clique sur ce bouton, les coordonnées de la souris apparaissent sur fond vert², on peut alors sélectionner un rectangle à la souris pour zoomer sur cette zone.

Enfin, on peut régler la fenêtre graphique en utilisant le bouton rouge `geo` de configuration graphique.

²Ce mode est en réalité un mode de compatibilité TI89/92, il permet de faire des sorties graphiques non interactives dans un programme



10 Le tableur

Cliquer sur le bouton jaune mtrw du bandeau.

Copier cellule Recalculer Ajouter ligne/colonne

Tableur spreadsheet.tab

	A	B	C	D	E	F	G
0	68	-21	56	59	192	''	''
1	82	-60	-32	53	0	''	''
2	-44	10	-4	25	0	''	''
3	-27	2	90	83	0	''	''
4	27	43	-7	21	0	''	''
5	0	0	0	0	0	''	''
6	0	0	0	0	0	''	''

Zone de saisie

Recopie vers le bas ou a droite

Statistiques

remplir 1d 2d inf

hist changed (current.cas), mtrw changed (spreadsheet.tab)

home eqw geo help hist msg mtrw prg var

=sum((A0):(D2))

coller ↑ →

RAD 12 real exact ALG xcas /home/home/pa geo cas gen ← →

Time: 0.01 stop abc ↓ ←

∂, \int, \sum	'	"	()	[]	{ }	num	x ²	1/x	+	ee	7	8	9
lim		:=	,	;		eval	√	+	-	.	4	5	6
a sin	a cos	a tan	i	π		x	y	^	*	0	1	2	3
e [^]	ln	10 [^]	log10	oo	e	z	t	%	/	kbd	esc	entrée	

La syntaxe de remplissage d'une cellule en fonction d'autres cellules est similaire à celle des tableurs usuels. On utilise des lettres pour repérer la colonne, des nombres pour la ligne, le signe \$ devant le nom d'une colonne ou d'une ligne désigne une référence absolue pour les opérations de recopie d'une cellule, le signe = en début d'expression indique une formule de tableur. Pour faire référence à une plage de cellule, on écrit la cellule en haut à gauche, le signe : et la cellule en bas à droite, par exemple =mean(A1 :B4) désigne la moyenne des cellules lignes 1 à 4, colonnes A et B.

Le tableur permet en particulier d'obtenir des tableaux de valeurs de fonction ou de suites récurrentes assez facilement (instructions table et tableseq du menu Edit->Spreadsheet).

On peut aussi sélectionner à la souris une sous-matrice et la coller par exemple dans l'historique. On peut aussi extraire une sous-matrice en indiquant des lignes consécutives et des colonnes dans la zone de saisie de la barre de boutons (par exemple B3 :B7 ,D ,E)

11 Les scripts

Il est souvent intéressant de pouvoir exécuter plusieurs fois une même suite de commandes. Par exemple, pour générer une figure géométrique ou pour affecter certaines variables ou pour expliquer un calcul par étapes. On peut bien sûr exécuter un fichier qui contient ces commandes, toutes les commandes sont alors exécutées en une étape. On peut aussi exécuter ces commandes l'une après l'autre en exécutant un script : menu **File** → **Exécuter script**. Ceci va activer le partage d'écran et charger le script dans l'éditeur d'équations, il suffit alors de cliquer sur le bouton **exec** pour exécuter la ligne sélectionnée et passer à la ligne suivante.

12 Le mode programme

Dans ce mode activé par un click sur le bouton jaune **prg** du bandeau, on accède à un éditeur rudimentaire qui permet d'écrire plusieurs lignes de texte constituant un petit programme. Pour des programmes de taille importante il est conseillé d'utiliser un éditeur de texte.

Copie dans hist Test de syntaxe Aide syntaxe

```

pgcd(x,y):={
  if (y==0)
    return x;
  return pgcd(y,irem(x,y));
}

```

home eqw geo help hist msg mtrw prg var

nodisp(pgcd(x,y):={ if (y==0)

RAD 12 real exact ALG xcas /home/home/pa geo cas gen

Time: 0.02

stop abc

∂, \int, \sum	'	"	()	[]	{ }	num	x ²	1/x	+	ee	7	8	9
lim		:=	,	;		eval	$\sqrt{\quad}$	+	-	.	4	5	6
a sin	a cos	a tan	i	π		x	y	^	*	0	1	2	3
e [^]	ln	10 [^]	log10	oo	e	z	t	%	/	kbd	esc	entrée	

Une ligne de boutons permet de saisir efficacement les formes syntaxiques (groupe d'instruction `{ }`, fin d'instruction `;`, saut de ligne, test `if`, boucle `for`, sortie de fonction `return`). Le bouton `->cmd` quitte le mode programme, recopie le programme

dans la ligne de commande, il ne reste qu'à valider la ligne. Si votre programme commence par une instruction de stockage (par exemple `monprog(a,b) := a+b;`), `xcas` recherche si l'historique contient déjà une validation d'une version précédente de ce programme et effectue alors un remplacement dans l'historique (les niveaux suivants de l'historique sont éventuellement modifiés). Le bouton `exec` sert à exécuter dans l'historique la ligne du curseur. L'appui plusieurs fois sur cette touche permet d'exécuter "en pas-à-pas" tout un fichier de commandes. Enfin, le bouton `parse` permet de vérifier que le programme actuel est syntaxiquement correct.

Par défaut (style de programmation `Xcas`), la syntaxe de programmation est proche de celle du langage C. Les principales différences sont :

- les variables sont non typées, les variables sont par défaut globales sauf si elles sont déclarées locales par `local` en début de définition d'une fonction.
- l'affectation se fait par `:=`
- une fonction se déclare par la syntaxe `f(x,y) := { local z; instructions; }`.

Attention aux priorités respectives de `:=` et `,` qui nécessitent l'usage de parenthèses si vous affectez lors de la déclaration des variables locales :

`f(x,y) := { local (z:=0), v, w; ... }`.

On peut aussi choisir dans l'écran de configuration général d'utiliser une syntaxe similaire à celle de Maple, Mupad ou d'une TI89/92. Ceci affecte alors tout le logiciel, par exemple $\sqrt{-1} = I$ en mode Maple/Mupad et $\sqrt{-1} = i$ en mode `xcas`.

L'instruction `debug` avec comme argument l'appel à une fonction à mettre au point permet d'exécuter un programme en pas-à-pas, de voir et de modifier des variables, de placer des points d'arrêt. Par exemple, chargez le fichier `bezout` du répertoire `exemples/arit` et exécutez `debug(pgcd(15, 25))`. Attention, cette fonctionnalité n'est pas disponible sous Windows.

13 Imprimer, générer des fichiers `tex`.

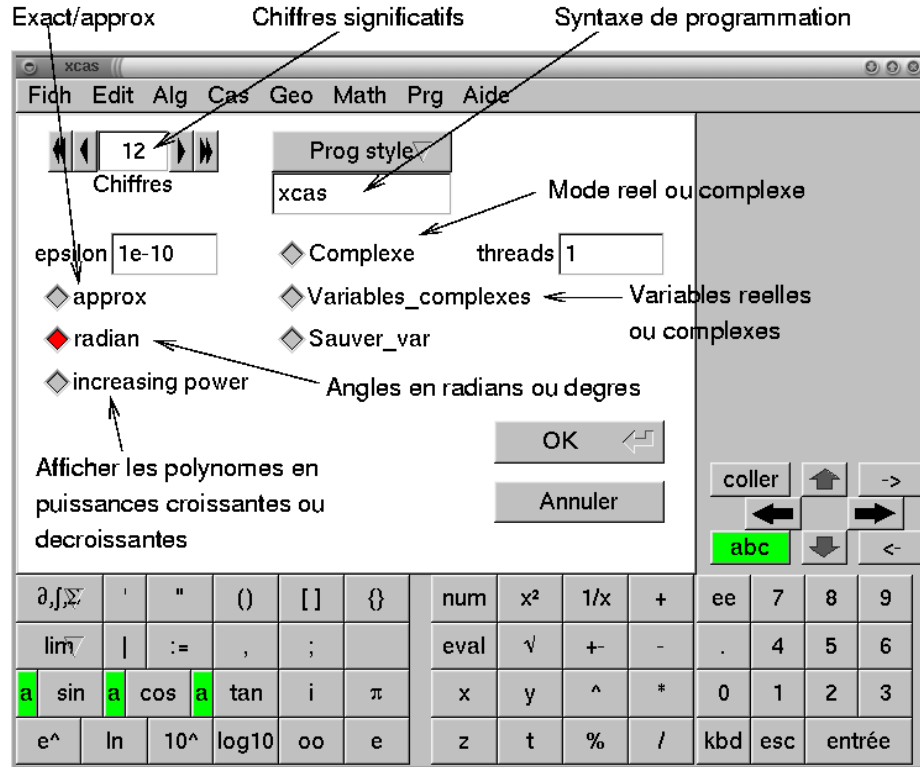
L'historique de `xcas` peut être imprimé si on dispose d'une installation \LaTeX standard. Pour imprimer l'écran graphique³ il faut utiliser la commande `graph2tex()` (ou `graph3d2tex()` pour imprimer une figure 3-d). Pour imprimer le tableur, il suffit de recopier le tableau dans l'historique (bouton `ok` du tableur puis `enter`). L'option de prévisualisation du menu `Fich->Imprimer` génère un fichier d'extension `.tex` du nom de la session choisie, ce fichier est autonome, on peut en insérer des extraits dans un autre fichier `tex`. Lorsqu'on quitte `xcas`, le fichier `session.tex` contient la traduction \LaTeX de l'historique.

14 Configuration

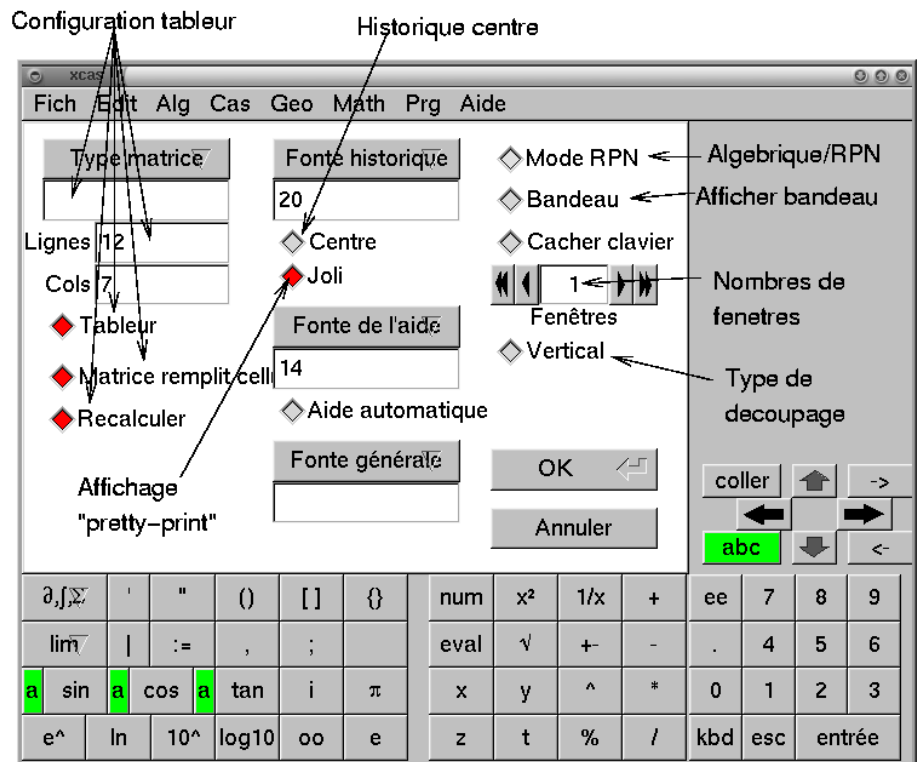
On peut changer la configuration du logiciel en cliquant sur l'un des 3 boutons de la ligne d'état, ou depuis le menu `Edit->Préférences`. L'écran de configuration

³Sauf les commandes de tracé de graphe terminant par `plot` qui insèrent automatiquement une sortie graphique

graphique a été présenté dans la section 9.3, on obtient pour la configuration du calcul formel



et pour la configuration générale



On peut aussi changer la configuration par des commandes comme `epsilon :=1e-10` (cf. le guide du calcul formel). On peut sauvegarder la configuration courante dans le menu File. La configuration se trouve dans le fichier `~/ .xcasrc` (ou `xcas.rc` sous Windows) et est chargée au lancement de `xcas`.

15 Quelques commandes bien utiles pour xcas, Maple et Mupad.

Les commandes de `xcas` sont regroupées dans des menus et sous-menus, elles sont donc faciles à trouver si on ne les connaît pas. La plupart des commandes ont une syntaxe identique pour Maple, Mupad et `xcas`, certaines sont malheureusement différentes. `xcas` est dans une certaine mesure capable de traduire la *grammaire* des 4 systèmes mais pas forcément toutes les instructions. Dans les commandes données ci-dessous, on a favorisé l'utilisation de noms de commandes communs à Maple, Mupad et `xcas` même si sur un système donné il existe parfois une syntaxe plus simple.

Attention, certaines commandes ne sont pas activées au lancement d'une session Maple et Mupad, par exemple les commandes d'algèbre linéaire. Il faut alors les activer (commande `with` en Maple ou `export` en Mupad).

15.1 Commandes scalaires

Lorsqu'il n'existe pas de nom de commande commun aux 3 systèmes, on a indiqué les 3 noms de commande dans l'ordre **Mupad/Maple/xcas**. Dans ces exemples, `f` désigne une expression (par exemple `f := x^2 - 1 ;`).

- `binomial(n, k)` (coefficient binomial $\binom{n}{k} = C_n^k = \frac{n!}{k!(n-k)!}$),
- `diff(f, x)` (dérive f par rapport à x),
- `diff(f, x$ k)` (dérive f k -fois ($0 < k$ un entier) par rapport à x),
- `DIGITS := k : float(f) / Maple : Digits := k ; evalf(f) ; / les 2` (donne l'évaluation numérique de f avec k chiffres significatives),
- `eval(f)` : évalue l'expression f .
- `expand(f)` (développe f),
- `factor(f)` (factorise f),
- `300 !` (factorielle de n),
- `float(hold(solve)(f=0, x)) / fsolve(f=0, x) / fsolve(f=0, x)` (cherche une solution numérique de l'expression f en la variable x),
- `float(f) / evalf(f) / les 2` (donne l'évaluation numérique avec précision machine de f),
- `ifactor(n)` (factorise un entier n),
- `int(f, x)` (primitive de f par rapport à la variable x),
- `int(f, x=a..b)` (intègre f de a à b),
- `limit(f, x=a)` (limite de f lorsque $x \rightarrow a$),
- `mod(a mod n)` calcule le reste de a modulo n ,
- `normal(f)` (écrit f sous forme d'une fraction rationnelle)
- `plotfunc2d(f, x=a..b) / plot(f, x=a..b) / les 2` (graphe (x, f) pour $x \in [a, b]$, f est vu comme une expression dépendant de x),
- `'f' / hold(f) / les 2` : empêche l'évaluation de f .
- `reset/restart` / non disponible (réinitialise la session),
- `series(f, x=a, k)` (développement à l'ordre k de f autour de a),

- `simplify(f)` (essaie de simplifier f),
- `solve(f=0,x)` (cherche les solutions en x de $f(x) = 0$),
- **Mupad** : `solve(ode(diff(y(x),x$ 2)+y(x)=0,y(x)))` /
Maple : `dsolve(diff(y(x),x$ 2)+y(x)=0,y(x))` /
xcas : `desolve(y''+y=0,y)`
 (cherche la solution générale de l'équation différentielle $y'' + y = 0$),
- `subs(f,x=a)/subs(x=a,f)` / les 2 (selon le mode) ou `subst(f,x=a)` :
 (remplace x par a dans f),
- `sum(f,n=a..b)` : calcule $\sum_{n=a}^b f(n)$ pour a, b entiers ou $\pm\infty$

15.2 Algèbre linéaire.

Attention, pour Mupad/Maple, il faut commencer par activer les commandes du "package" `linalg` (algèbre linéaire) par `export(linalg)` (Mupad) ou `with(linalg)` ; (Maple).

Lorsqu'il n'existe pas de nom de commande commun aux 3 systèmes, on a indiqué les 3 noms de commande dans l'ordre **Mupad/Maple/xcas**

- `A[j,k]` : accède à l'élément ligne j et colonne k de la matrice A . `xcas` accepte aussi la notation `C A[j][k]`
- `A+B/evalm(A&+B)` ou `add(A,B)` / `A+B`
 (addition de matrices ou de vecteurs),
- `1/A/inverse(A)` / les 2 (calcule l'inverse d'une matrice inversible A),
- `A*B/evalm(A&*B)` ou `multiply(A,B)` / `A*B`
 (multiplication de deux matrices),
- `f*A/scalarmul(A,f)` / `f*A`
 (multiplication de la matrice ou du vecteur A par le scalaire (ou la fonction) f),
- `charpoly(A,x)`
 (déterminant de $xId - A$ (polynôme caractéristique))
- `det(A)`
 (déterminant d'une matrice A),
- `eigenvalues(A)/eigenvals(A)` / les 2
 (valeurs propres de la matrice A . Si vous voulez des valeurs numériques (et si A est à valeurs numériques) écrivez au moins un des coefficients de A comme nombre réel ou complexe. Le calcul se fait alors numériquement et c'est beaucoup plus rapide),
- `eigenvectors(A)/eigenvects(A)` / les 2
 (vecteurs propres de A),
- `A=B/equal(A,B)` / les 2 : teste l'égalité de 2 matrices
- `matlinsolve(A,b)/linsolve(A,b)/linsolve(A,b)` (résout l'équation linéaire $Ax = b$, A une matrice, b un vecteur),
- `matrix()` (création d'une matrice) :
`matrix(2,2,[[5,4],[6,3]])`
`f :=(j,k)->(1/(j+k-1));matrix(2,2,f)`
`Dom::Matrix()([[5,4],[6,3]])/matrix([[5,4],[6,3]])//[[5,4],[6,3]],`
- `ncols(A)/coldim(A)` / les 2

- nombre de colonnes
- `nrows(A)/rowdim(A)/les 2`
- nombre de lignes
- `scalarProduct(A,B)/dotprod(A,B)/les 2`
(produit scalaire des deux vecteurs A et B),
- `trace/tr/trace` : trace d’une matrice
- `transpose(A)` (transposée de A)
- Création d’un vecteur : c’est une matrice $n \times 1$ (sauf s’il s’agit d’une matrice ligne $1 \times n$).

Exemples :

```
matrix(3,1,[[1],[2],[3]])
```

```
Dom::Matrix()([1,2,3])/vector([1,2,3])/[1,2,3].
```

Pour exécuter une commande sur toutes les composantes d’un vecteur ou d’une matrice, on utilise la commande `map`, par exemple :

```
map(A,normal)
```

exécute la commande `normal` sur toutes les composantes de A .

16 Conteneurs

C’est un type d’objet qui peut contenir d’autres objets : les principaux conteneurs sont les listes, les séquences, les ensembles, les vecteurs et les matrices (on utilise aussi les tableaux associatifs, c’est-à-dire des tableaux indicés par n’importe quelle valeur et pas forcément par des entiers). Les éléments sont séparés par des virgules, le type du conteneur est indiqué par des délimiteurs : `[]` pour les listes, rien pour les suites et `{ }` pour les ensembles (attention en mode `xcas`, `xcas` utilise `set []` ou `%{ %}` pour les ensembles, car `{` et `}` servent de délimiteurs de bloc comme en `C/C++`). L’élément d’indice k d’un conteneur c est désigné par `c[k]` (attention, `xcas` utilise l’indexage commençant en 0 en mode `xcas`, comme en `C/C++`). Pour un élément de matrice, on écrit `m[j,k]`.

Quelques instructions :

- `nops(m)` (ou `size(m)` avec `xcas`) : la taille
- `op(l)` : transforme la liste en suite
- `[e, op(l)]` : place e en tête de liste
- `[op(l), e]` : place e en queue de liste
- `[op(l1), op(l2)]` : concatène 2 listes
- `convert(maple,xcas)` ou `coerce(Mupad)` permettent de convertir un conteneur d’un type en un conteneur d’un autre type.
- `subsop` : permet de remplacer un indice dans une liste par une valeur, on peut utiliser la valeur `NULL` si on veut supprimer l’élément ayant cet indice.

16.1 Programmation.

Les programmes sont des fichiers texte qui sont lus pendant une session par la commande `read` avec comme argument le nom du fichier texte entre guillemets. Par convention, on utilisera un nom se terminant par `.mu` pour un programme MuPAD,

.map pour un programme Maple⁴, et .cxx pour xcas. Par exemple `essai.mu` pour un programme MuPAD.

Voici sur quelques exemples les principales instructions de contrôle. Notez que xcas accepte aussi la syntaxe Maple ou Mupad ou TI89/92⁵

- en-tête de fonction et création de variables locales :


```
f:= proc(x) local n; begin n:= 2; n*x end_proc (mupad)
f:= proc(x) local n; n:= 2; n*x end; (maple)
f(x):={ local n; n:=2; return(n*x); } (xcas)
:f(x):Func:Local n: 2 => n: Return n*x:EndFunc (TI89)
```
- faire un test :


```
if x2>x1 then m:=x2; else m:=x1; end_if; (mupad)
if x2>x1 then m:=x2; else m:=x1; fi; (maple)
if (x2>x1) { m:=x2; } else { m:=x1; } (xcas)
:If x2>x1 Then: x2 => m :Else : x1 => m :EndIf (TI89)
```
- boucle indéfinie :


```
while (x=0) do x:=x-1; end_while; (mupad)
while (x=0) do x:=x-1; od; (maple)
while (x=0) { x:=x-1; }; (xcas)
:While x=0 :x-1 => x :EndWhile (TI89)
```
- boucle définie :


```
for j from 1 to 15 do x:=j+x; end_for; (mupad)
for j from 1 to 15 do x:=j+x; od; (maple)
for (j:=1;j<=15;j++) { x:=j+x; } (xcas)
:For j,1,15 :j+x => x :EndFor (TI89)
```
- break permet de sortir d'une boucle, continue passe immédiatement à l'itération suivante.

16.2 TP

Utilisation interactive

1. Déterminer la liste des diviseurs de 45768.
Factoriser 100!
2. Développer $(x+3)^7 \times (x-5)^6$.
3. Simplifier les expressions suivantes :

$$\sqrt{3+2\sqrt{2}}, \quad \frac{1+\sqrt{2}}{1+2\sqrt{2}}, \quad e^{i\pi/6}, \quad 4\operatorname{atan}\left(\frac{1}{5}\right) - \operatorname{atan}\left(\frac{1}{239}\right)$$

4. Factoriser sur \mathbb{R} et \mathbb{C} :

$$x^8 - 3x^7 - 25x^6 + 99x^5 + 60x^4 - 756x^3 + 1328x^2 - 960x + 256$$

$$x^6 - 2x^3 + 1, \quad (-y+x)z^2 - xy^2 + x^2y$$

⁴Attention, les fichiers de type session Maple .mws sont différents des fichiers programme Maple, xcas permet d'effectuer la conversion .mws vers .map par `read`

⁵attention toutefois en mode xcas, les tableaux commencent à l'indice 0 et on ne peut pas utiliser $i = \sqrt{-1}$ comme variable.

5. Calculez les intégrales et simplifiez le résultat :

$$\int \frac{1}{e^x - 1} dx, \quad \int e^{x^2} dx, \quad \int x \sin(x) e^x dx$$

Vérifiez en dérivant les expressions obtenues.

6. Déterminer la valeur de :

$$\int_1^2 \frac{1}{(1+x^2)^3} dx, \quad \int_1^2 \frac{1}{x^3+1} dx, \quad \int_0^x \frac{t}{\sqrt{1-t}} dt$$

7. Calculer les sommes suivantes

$$\sum_{k=1}^N k, \quad \sum_{k=1}^N k^2, \quad \sum_{k=1}^{\infty} \frac{1}{k^2}$$

8. Vérifier que

$$\sin(2x) - 2 \sin(x) \cos(x) = 0$$

en passant en exponentielles complexes.

9. Calculer le développement de Taylor en $x = 0$ à l'ordre 4 de :

$$\ln(1+x+x^2), \quad \frac{\exp(\sin(x)) - 1}{x+x^2}, \quad \sqrt{1+e^x}, \quad \frac{\ln(1+x)}{\exp(x) - \sin(x)}$$

10. Résoudre le système linéaire :

$$\begin{cases} x + y + az = 1 \\ x + ay + z = 2 \\ ax + y + z = 3 \end{cases}$$

11. Déterminer l'inverse et le déterminant de :

$$A = \begin{pmatrix} 1 & 1 & 1 & a \\ 1 & 1 & a & 1 \\ 1 & a & 1 & 1 \\ a & 1 & 1 & 1 \end{pmatrix}$$

Diagonaliser la matrice A .

Programmation de la méthode de Horner.

1. Écrire une fonction évaluant un polynôme en un point, on représente le polynôme sous forme de la liste de ces coefficients.
2. En utilisant cette fonction, écrire une fonction qui calcule le développement de Taylor complet d'un polynôme en un point.

17 Installation de la librairie giac sous Linux

Pour les programmeurs souhaitant utiliser les fonctionnalités de calcul formel dans du code C++, le code source de la librairie C++ giac sur laquelle l'interface xcas est basée est disponible à l'adresse :

```
ftp://ftp-fourier.ujf-grenoble.fr/xcas/giac.tgz
```

La licence d'utilisation est la GPL 2.0 ou ultérieure (si vous diffusez du code lié à giac, ce code doit donc être diffusé sous licence GPL).

Avant de configurer giac, vous devrez d'abord compiler et installer des bibliothèques telle que FLTK, FLVW, GSL, etc. Vous pouvez récupérer des bibliothèques statiques précompilées et les fichiers en-tête correspondants pour Linux à l'adresse suivante :

```
ftp://ftp-fourier.ujf-grenoble.fr/xcas/statlibs.tgz
```

```
ftp://ftp-fourier.ujf-grenoble.fr/xcas/headers.tgz
```

pour les installer, logguez-vous en utilisateur root désarchivez les deux fichiers depuis le répertoire /usr/local, ajoutez dans votre fichier /etc/ld.so.conf la ligne :

```
/usr/local/lib
```

puis tapez ldconfig.

Décompressez maintenant l'archive contenant giac :

```
tar xvfz giac.tgz
```

```
cd giac-0.3.0
```

compilez-le :

```
./configure
```

```
make
```

et enfin installez-le :

```
su
```

tapez le mot de passe de root, puis la touche Entree puis :

```
make install
```

puis :

```
exit
```

pour revenir à la session normale (vous pouvez lire les fichiers README et INSTALL pour plus de détails et changer la configuration par défaut).

Remarque : vous pouvez installer giac sous Windows de la même manière avec le compilateur gcc sous Windows cygwin. Installez cygwin puis GMP, GSL, FLTK, FLVW puis Giac comme sous Unix (sauf pour FLTK qui a une procédure d'installation spécifique décrite dans son source).

18 Programmation en C++ avec giac

L'ensemble des fonctions et des types de giac sont placés dans l'espace de nom giac. Un source C++ utilisant giac devra donc précéder les identificateurs et fonctions de giac:: ou doit exporter l'espace de noms giac par la ligne de source `using namespace giac;`

18.1 Le type `gen`

La librairie `giac` utilise un type générique `gen` pour représenter les différents types (entiers, réels, vecteurs, matrices, polynômes, etc.). `gen` est un type structuré, composé d'un champ `type`, dont la valeur détermine le type de l'objet, d'un champ `subtype` qui permet de distinguer par exemple un vecteur d'une matrice ou d'une liste lorsqu'ils utilisent la même représentation interne (un vecteur, c'est-à-dire un `std::vector<gen>` ici), et d'un champ union anonyme pour la valeur de l'objet (ou un pointeur). Les `double` et les petits entiers sont représentés par des objets immédiats, les autres types par des pointeurs sur des objets alloués dynamiquement (avec un compteur de référence). Pour accéder à la représentation interne, on commence par tester que le type du `gen` considéré est bien le type prévu, puis on copie l'objet pointé.

Exemple :

```
gen g( "[1,2]" );
déclare une variable g de type gen qui sera compilé comme une ligne de commande
contenant [1, 2], donc g contiendra le vecteur [1, 2]. On a alors g.type==_VECT,
et le vecteur sous-jacent est vecteur v(*g._VECTptr) ; (copie du vecteur) ou
vecteur &v = *g._VECTptr ; (référence sans copie). On a alors v.size()==2
et v[0]==gen(1) et v[1]==gen(2).
```

Les noms de variables sont de type `identificateur *g._IDNTptr` (si `g.type==_IDNT`). Ce type structuré comporte un nom et peut avoir une valeur (ou une hypothèse assume) locale ou globale. Les identificateurs de noms `a` à `z` sont prédéfinis.

Un objet symbolique est de type `symbolic *g._SYMBptr`. Ce type est composé d'un sommet (de type `unary_function_ptr`) qui est la fonction au sommet du graphe représentant l'objet symbolique et d'une feuille qui est l'argument de la fonction, s'il s'agit d'une fonction non unaire, les arguments sont regroupés en un vecteur. Les fonctions usuelles sont prédéfinies dans le système et ont comme valeur `at_` suivi par le nom de la fonction.

Par exemple `sin(x)` est un objet symbolique de sommet `at_sin` et de feuille l'identificateur prédéfini `x(x__IDNT_e)`.

18.2 Programmes et modules dynamiques.

Vous pouvez créer des programmes dont vous définissez l'interface, ou vous pouvez créer des fonctions qui s'ajoutent à la librairie `giac` ou à `xcas` (que l'on charge pendant l'exécution avec l'instruction `insmod`). Dans le deuxième cas, on crée une librairie dynamique.

19 Exemples en C++.

19.1 Programme (pgcd de 2 entiers).

```
#include <giac/giac.h>
using namespace std;
using namespace giac;
```

```

gen pgcd(gen a,gen b){
    gen q,r;
    for (;b!=0;){
        r=irem(a,b,q);
        a=b;
        b=r;
    }
    return a;
}

int main(){
    cout << "Enter 2 integers";
    gen a,b;
    cin >> a >> b;
    cout << pgcd(a,b) << endl;
    return 0;
}

```

Pour compiler ce programme (fichier `pgcd.cc` du répertoire `doc/en`), tapez la commande :

```
g++ -g pgcd.cc -lgiac -lgmp
```

19.2 Module dynamique.

Il faut alors définir une nouvelle fonction (qui sera accessible depuis `xcas`). Ici cette fonction s'appellera `pgcd` dans `xcas` (c'est le rôle de la ligne `const string _pgcd_s("pgcd");`). La fonction de calcul (de nom C++ `pgcd`) est identique à la précédente, la fonction de nom C++ `_pgcd` est la fonction appelée par `xcas` avec comme argument les arguments de la ligne de commande regroupés en un vecteur. Elle doit donc vérifier que l'utilisateur appelle la fonction avec 2 arguments (donc l'argument passé à `_pgcd` doit être un vecteur ayant 2 arguments).

La compilation de ce fichier `pgcd.cpp` (répertoire `doc/en` du source de `giac`) se fait automatiquement en l'ouvrant dans `emacs` ou en recopiant la chaîne qui suit `compile-command`. Pour l'insérer dans une session `xcas`, il faut taper la commande `insmod("/home/user/giac-0.2.2/doc/en/libpgcd.so")` (en supposant que le source de `giac` a été désarchivé dans le répertoire `/home/user`).

```

// -*- mode:C++ ; compile-command: "g++ -I.. -fPIC -DPIC -g -c pgcd.cpp \
-o pgcd.lo && ln -sf pgcd.lo pgcd.o && gcc -shared pgcd.lo -lc \
-Wl,-soname -Wl,libpgcd.so.0 -o libpgcd.so.0.0.0 && \
ln -sf libpgcd.so.0.0.0 libpgcd.so.0 && \
ln -sf libpgcd.so.0.0.0 libpgcd.so" -*-
using namespace std;
#include <stdexcept>
#include <cmath>
#include <cstdlib>
#include <giac/giac.h>

```

```

#include "pgcd.h"

#ifndef NO_NAMESPACE_GIAC
namespace giac {
#endif // ndef NO_NAMESPACE_GIAC

    gen pgcd(gen a,gen b){
        gen q,r;
        for (;b!=0;){
            r=irem(a,b,q);
            a=b;
            b=r;
        }
        return a;
    }
    gen _pgcd(const gen & args){
        if ( (args.type!=_VECT) || (args._VECTptr->size()!=2))
            setsizeerr();
        vecteur &v=*args._VECTptr;
        return pgcd(v[0],v[1]);
    }
    const string _pgcd_s("pgcd");
    unary_function_unary __pgcd(&_pgcd,_pgcd_s);
    unary_function_ptr at_pgcd (&__pgcd,0,true);

#ifndef NO_NAMESPACE_GIAC
} // namespace giac
#endif // ndef NO_NAMESPACE_GIAC

```

20 Compléments.

Pour les programmeurs C++, le code source de `giac` contient un peu de documentation dans le fichier `doc/en/giac_us.texinfo` qui s'installe dans `/usr/local/info/giac_us.info` (à lire avec `emacs` commande `Ctrl-U Ctrl-H I` suivi du chemin d'accès précédent). Pour le reste, il faut consulter directement le code source (les fichiers en-tête et source contiennent quelques commentaires).