Installation d'une bibliothèque personnelle

© Pierre Lantagne
Enseignant retraité du Collège de Maisonneuve

Au lancement de Maple, il y a chargement du noyau (*kernel*, en anglais). Ce noyau est la base, le cœur de ce système de calcul symbolique et numérique. Le noyau est hautement optimisé, occupe environ 10 % de la taille totale du système. Le noyau renferme les plus fréquentes routines de calculs sur les entiers et les rationnels et de calculs élémentaires sur les polynômes. On dit de ces macro-commandes qu'elles sont résidantes (*built-in functions*), c'est-à-dire qu'il est impossible d'en faire afficher le code. On a donc pas accès à leur source.

Le reste, soit environ 90 % des macro-commandes réside dans des extensions appelées greffons ou bibliothèques (*library*, en anglais). Il est alors possible d'en faire afficher le source. Cet accès s'avère très intéressant lorsque l'on désire approfondir sa connaissance de Maple et la programmation en Maple.

On distingue deux types de bibliothèques: la bibliothèque principale et les bibliothèques secondaires (*packages*).

La bibliothèque principale de Maple contenant les macro-commandes les plus usuelles sont automatiquement chargées au moment de leur premier appel et ce, de manière imperceptible pour l'usager. La manière dont elles ont été définies en font des macro-commandes auto-chargeables. Ces macro-commandes sont donc immédiatement disponibles aussitôt après le lancement de Maple. C'est le cas par exemple des macro-commandes *plot* et *surd*.

Les bibliothèques secondaires Maple ou greffons (*packages*) contiennent, quant à elles, des macro-commandes dédiées à des domaines plus spécialisés des mathématiques. Certains Français traduisent le terme anglais *package* par *paquets*... mais retenons, ici, l'appellation bibliothèques. Par exemple, la bibliothèque Maple *plots* contient 57 (Maple 2018) macro-commandes utiles afin de manipuler des structures graphiques comme, par exemple, pour faire afficher une superposition de plusieurs tracés dans un même graphique. Un autre exemple de bibliothèque Maple est la bibliothèque *Student* organisée en une collection de sous-bibliothèques dédiées à l'apprentissage des mathématiques.

Dans tous les cas, afin que Maple puisse exécuter une macro-commande saisie dans la zone de requêtes, il faut que Maple sache où retrouver le code de la macro-commande en cause. À l'installation du logiciel Maple (le cœur lui-même et toutes les bibliothèques de Maple), l'installateur de Maple initialise la variable système *libname* en mémorisant le chemin du répertoire contenant toutes les bibliothèques Maple installées lors de l'installation du programme lui-même sur l'ordinateur. Par exemple, voici l'initialisation de *libname* dans les cas de l'nstallation de Maple sur mon ordinateur.

```
> restart;
> libname;
```

"C:\Program Files\Maple 2019\lib" **(1)**

ATTENTION: En tout temps, il ne faut pas rendre libre cette variable ou écraser le contenu de cette variable, car il ne sera plus possible pour Maple d'exécuter toutes macro-commandes durant la session de

travail.

Le dossier *lib* referme toutes les macro-commandes de toutes les bibliothèques Maple. Au cours d'une session Maple, pour accéder aux macro-commandes d'une bibliothèque particulière, nous les rendons disponibles à la session Maple en utilisant la macro-commande *with*. Par exemple,

```
> with(plots);
```

[*animate*, *animate3d*, *animatecurve*, *arrow*, *changecoords*, *complexplot*, *complexplot3d*, *conformal*, *conformal3d*, *contourplot*, *contourplot3d*, *coordplot*, *coordplot3d*, *densityplot*, *display*, *dualaxisplot*, *fieldplot*, *fieldplot3d*, *gradplot*, *gradplot3d*, *implicitplot*, *implicitplot3d*, *inequal*, *interactive*, *interactiveparams*, *intersectplot*, *listcontplot*, *listcontplot3d*, *listdensityplot*, *listplot*, *listplot3d*, *loglogplot*, *logplot*, *matrixplot*, *multiple*, *odeplot*, *pareto*, *plotcompare*, *pointplot*, *pointplot3d*, *polarplot*, *polygonplot*, *polygonplot3d*, *polyhedra_supported*, *polyhedraplot*, *rootlocus*, *semilogplot*, *setcolors*, *setoptions*, *setoptions3d*, *shadebetween*, *spacecurve*, *sparsematrixplot*, *surfdata*, *textplot*, *textplot3d*, *tubeplot*] **(2)**

Ce que l'on vient de faire, c'est d'enrichir Maple, pour la session, d'un vocabulaire de 57 nouveaux noms de macro-commandes. Cela a été possible parce que Maple a été voir dans le dossier *lib* et il y a trouvé la bibliothèque *plots*. Et donc, à partir de cet instant, Maple sait où aller voir pour exécuter les instructions de l'une de ces macro-commandes.

Maintenant, nous voulons enrichir la session Maple de toutes les macro-commandes de la bibliothèque usager *conique*, celle de l'auteur de ce document.

```
> with(conique);
Error, invalid input: with expects its 1st argument, pname, to
be of type {`module`, package}, but received conique
```

Le message est très clair (et en rose): Maple ne reconnaît pas le nom *conique* comme une bibliothèque.

Avant d'employer la requête *with*, nous devons indiquer à Maple le chemin à suivre pour accéder aux macro-commandes de cette bibliothèque, c'est-à dire accéder au fichier contenant le code de ces macro-commandes. Il faut donc informer Maple du chemin menant au dossier contenant les fichiers dans lequel il trouvera cette bibliothèque. Pour disposer des macro-commandes sur votre ordinateur, il vous faudra procéder à l'installation de deux fichiers « conique.ind » et « conique.lib » sur votre disque rigide: celui de votre ordinateur personnel et non pas celui du poste sur lequel vous travaillez au collège.

À partir de mon site internet, ..., cliquer sur l'onglet EED. Dans la section « EED) » localiser et télécharger les fichiers « conique.ind » et « conique.lib » (Archive rar) et les déposer dans un dossier. Je vous suggère de nommer ce dossier « Bibliothèque conique » et un bon choix d'emplacement pour ce dossier serait de le placer dans le dossier `Users` de votre installation Maple. Tout autre endroit sur votre ordinateur est également possible.

Ensuite, la façon d'informer Maple du chemin dont il est question consiste **à ajouter** à la variable d'environnement *libname* le chemin menant à ces deux fichiers. L'énoncé du chemin doit être une chaîne de caractères de la forme

```
"Nom_du_volume:Disque_lettre:\\Répertoire_1\\Répertoire_2\\.....
```

```
/Répertoire_contenant_la_Bibliothèque_conique"
```

RAPPEL: La variable *libname* possède déjà une assignation qui a été faite au moment de l'installation du logiciel, soit l'assignation du chemin menant aux bibliothèques Maple contenues dans le dossier lib.

```
> libname;
```

"C:\Program Files\Maple 2019\lib" **(3)**

Il ne faut surtout pas écraser le contenu de la variable *libname* par une nouvelle assignation, Maple ne saurait plus comment localiser les macro-commandes de Maple. La façon dont il faut s'y prendre est la suivante:

```
libname := "nouveau_chemin", libname;
```

Par exemple, l'assignation suivante est celle que je dois faire son mon ordinateur personnel.

```
> libname:="C:\\Users\\plant\\Desktop\\Éléments site Maple au
  cégep\\Downloads\\Documents Maple/EED/Bibliothèque conique",libname;
```

libname := **(4)**

"C:\Users\plant\Desktop\Éléments site Maple au cégep\Downloads\Documents Maple/EED/Bibliothèque conique", "C:\Program Files\Maple 2019\lib"

On remarquera le dernier élément dans le résultat de cette assignation: c'est le chemin menant aux bibliothèques Maple, c'est-à-dire que c'est le contenu qu'avait la variable *libname*. Cette façon de faire pour ajouter (concaténer) un nouveau terme à une suite est valable quelque soit la variable en cause. Dans ce cas-ci, la variable en cause est la variable *libname*.

Maintenant, nous pouvons indiquer à Maple la manière d'accéder aux instructions des 7 macro-commandes de la bibliothèque ***conique***.

```
> with(conique);
```

[*cercleplot*, *ellipseplotx*, *ellipseploty*, *hyperboleplotx*, *hyperboleploty*, *paraboleplotx*, *paraboleploty*] **(5)**

Le résultat de cette requête doit la liste suivante de macro-commades:
[*cercleplot*, *ellipseplotx*, *ellipseploty*, *hyperboleplotx*, *hyperboleploty*, *paraboleplotx*, *paraboleploty*]