

Entrées et sorties.

Il est possible de sauver des données dans des fichiers en utilisant Maple, de même que de lire des fichiers textes ou au format Maple. De plus, on peut convertir une grande partie des programmes Maple en d'autres langages (C, Fortran, etc.) ou formats (TEX, L^AT_EX, HTML, etc.)

Nous allons voir ici comment faire pour réaliser de telles opérations.

1 Lire et écrire dans des fichiers.

1.1 Sauver des fichiers avec Maple.

La façon la plus simple de sauver un document Maple est celle que vous utilisez depuis le début de ce cours, en cliquant sur l'icône «save», en haut à gauche de la barre de menu. Ainsi, vous obtenez un fichier avec l'extension¹ «.mws» pour «Maple Work Sheet» (soit «Feuille de travail Maple»).

De tels fichiers sont exportables d'un ordinateur à l'autre (et ce, quelque soit le système d'exploitation), pourvu que les ordinateurs concernés aient tous les deux Maple installé.²

Lorsqu'on utilise cette méthode pour sauver des données, on peut également choisir de sauver au format «Maple Text», ce qui a pour effet de sauver le document au format texte brut, avec des signes mathématiques en ASCII-art. Notez que dans ce cas, si l'on édite le fichier ainsi sauvé, avec un éditeur de texte standard, le texte est précédé d'une dièse et d'une espace (#), tandis que les entrées Maple sont précédés d'un signe supérieur et d'une espace (>).

¹Du moins, si votre système d'exploitation affiche les extensions des fichiers...

²Notons cependant des problèmes au niveau de l'encodage des accents entre un fichier sauvé sur un Macintosh et un PC. Mais ce problème est générique et en rien lié à Maple.

1.2 Sauver, écrire et lire des données dans un fichier

La commande 'save(...)' crée un fichier texte écrit en langage Maple. Il est préférable d'entourer le nom du fichier par des guillemets anglais ("), de manière à ce que Maple interprète le nom du fichier comme une chaîne (et non comme une variable, ou autre).

Les fichiers en langage Maple sont des fichiers codés spécialement pour retenir l'ensemble des informations dans un format le plus compacte possible. De tels fichiers peuvent être lus (ou créés) avec un simple éditeur de texte.

Faites attention à l'endroit où est sauvegardé votre fichier (il vaut mieux mettre un chemin absolu, plutôt que relatif, pour plus de certitude). Notez cependant que dans les exemples suivants, on utilisera un chemin relatif, la syntaxe des chemins absolus dépendant des systèmes d'exploitation.

Si l'on ajoute l'extension «.m» à la fin du nom du fichier de sauvegarde, le fichier sera enregistré dans un format spécial. À l'aide d'un éditeur de texte, comparez les fichiers «temp» et «temp.m» suivants :

```
> f:=x->x*sin(2*x);      # Une fonction f.
g:=exp;                 # Une fonction g.
h:=g@f;                 # La composée des deux fonctions.

# On crée une procédure...
OD:=proc(m::integer,n::integer)
  local i, istriple;
  istriple:= i ->
  'if'(type((i-1)/2,integer) and type(i/3,integer), i, NULL);
  [seq(istriple(i), i=m..n)]
end;
```

```
# On sauve f, g, h et OD dans le fichier temp
save f,g,h,OD,"temp";
# On sauve f, g, h et OD dans le fichier temp.m
save f,g,h,OD,"temp.m";
```

Pour récupérer des données sauvegardées dans un fichier, on se sert de la commande 'read(...)'. De même, il est préférable d'utiliser un chemin absolu, plutôt que relatif, pour être certain d'ouvrir le bon fichier.

```
> read("temp");        # On lit le fichier temp.
h(theta);              # On peut se servir des fonctions
OD(10,40);             # qui sont dedans...
```

À la place de lire le fichier en entier, il est possible de lire un fichier ligne par ligne, avec la commande `'readline(...)`', ou d'écrire une ligne dans un fichier, à l'aide de `'writeline(...)`'. Notez que si un fichier n'a pas encore été ouvert, il est lu depuis sa première ligne. Si un fichier a déjà été ouvert, Maple se souvient de la ligne où il était et lit la ligne suivante. Une fois arrivé à la fin du fichier, Maple reprend à la première ligne.

```
> # Lecture de temp
readline("temp");
readline("temp");
readline("temp");
readline("temp");

# Ecriture dans temp
writeline("emp","hello", "goodbye");
```

Pour écrire l'ensemble des sorties de Maple dans un fichier, plutôt que le faire apparaître sur le terminal, on utilise la commande `'writeto(...)`'. Si l'on veut rediriger la sortie vers le terminal, il suffit de taper la commande `'writeto(terminal)`'.

```
> writeto("temp"); # On redirige la sortie vers temp.
aa:=i->a*r^i;
ss:=n->simplify(sum(aa(i),i=1..n));
ss(n);
writeto(terminal); # Retour vers le terminal.
ss(n);
```

Notez que la commande `'writeto(...)`' crée un nouveau fichier. S'il existe déjà un fichier avec le nom indiqué dans la commande, son contenu sera écrasé au profit du nouveau. Pour ne pas écraser un fichier existant, et rajouter des données à la fin, on se sert de la commande `'appendto(...)`'.

On peut générer des sorties préformatées, soit sur le terminal, soit dans un fichier, à l'aide de la commande `'writedata(...)`', consultez l'aide-en-ligne pour plus de détails à ce sujet.

2 Conversion vers d'autres langages et génération de code.

Maple permet de générer d'autres langages, tels que le Fortran, le C ou d'autres présentations, tels que T_EX, L^AT_EX ou du HTML.

2.1 Conversion des commandes et procédures en Fortran et en C.

Pour convertir des commandes en fortran, on se sert de la commande 'fortran(...)'. Pour plus de détails sur ce langage historique, je vous invite à consulter l'aide en-ligne.

Une commande similaire existe pour les conversions en langage C, mais il faut avant tout utiliser la commande 'readlib(C)'. Encore une fois, pour plus de détails consultez l'aide en ligne. Voici quelques exemples :

```
> S:=proc(x,y)      # Une procedure.
    x*log(y)
end;

fortran(S);        # Conversion en Fortran

readlib(C);        # Lecture de la bibliothèque C.
C(ss,optimized);  # Conversion en C (optimisée)
```

2.2 Conversions de Maple en L^AT_EX

Une simple expression peut être convertie en L^AT_EX en utilisant la commande ...'latex(...)'. Encore une fois, plutôt que d'avoir la sortie sur le terminal, on peut l'envoyer dans un fichier.

```
> # Une integrale compliquée.
EI:=Int(1/sqrt(x^3+1),x=1..X)=int(1/sqrt(x^3+1),x=1..X);
latex(EI);          # On traduit la sortie en LaTeX
# On sauve la sortie dans un fichier nommé temp.
latex(EI, "temp");
```

Notez qu'il est également possible, avec l'interface graphique, de sauver toute une feuille de travail Maple en un fichier L^AT_EX (via le menu «Export As...»).

Via ce menu, on peut également sauver un tel fichier en HTML.

Il faut cependant être réaliste : le code généré par Maple(que ce soit en HTML ou en L^AT_EX) est illisible avec un éditeur de texte, comme avec tous les éditeurs «WYSIWYG»³.

Si vous voulez compiler un document en L^AT_EX, converti à partir d'un feuillet Maple, vous aurez besoin de certaines bibliothèques développées pour (et par) Maple.

Lors de la conversion en HTML, c'est encore pire : toutes les formules mathématiques deviennent des images et ainsi une simple page HTML peut atteindre des centaines de ko. . .

Bref, même si c'est possible, je ne saurais que trop vous déconseiller de faire de telles conversions. Apprenez plutôt à écrire du HTML (et du L^AT_EX) cela vous sera nettement plus utile.

³WYSIWYG est l'acronyme de What You See Is What You Get, soit «Ce que vous voyez est ce que vous obtenez.»