Cahiers GUTenberg

LuaLaTeX et MetaPost avec *luamplib*, une introduction

Maxime Chupin

Cahiers GUTenberg, n°58 (2021), pages 55–79.
https://doi.org/10.60028/cahiers.v2021i58.34

LUALATEX ET METAPOST AVEC LUAMPLIB, UNE INTRODUCTION[1]

Maxime CHUPIN

RÉSUMÉ. Cet article est une introduction à l'utilisation de la bibliothèque `mplib` de LuaTEX qui contient le programme METAPOST. Cette utilisation est rendue possible avec le format LATEX grâce à l'extension `luamplib` dont les principales fonctionnalités sont présentées. Cette extension devient mature pour une utilisation intensive, et c'est ce que nous montrons au travers d'un certain nombre d'exemples.
MOTS-CLÉS : LuaLATEX, débutant, METAPOST, `luamplib`

ABSTRACT. *This article is an introduction to the use of the* `mplib` *library of LuaTEX, which is an embedded version of the METAPOST program. This library can be used with the LATEX format thanks to the* `luamplib` *package and we present its main features. This package has become mature and suitable for intensive use, and we illustrate this with some examples.*
KEYWORDS: *LuaLATEX, beginners, METAPOST,* `luamplib`

ZUSAMMENFASSUNG. *Dieser Artikel ist eine Einführung in die Verwendung der LuaTEX-Bibliothek* `mplib`*, die das Programm METAPOST enthält. Das Paket* `mplib`*, dessen Funktionen wir präsentieren, hat die Verwendung von METAPOST ermöglicht. Es wird an Beispielen gezeigt, wie das Paket aktuell weiter entwickelt wird.*
STICHTWÖRTER: *LuaLATEX, Anfänger, METAPOST,* `luamplib`

1. Titre en anglais : *An introduction to LuaLATEX and METAPOST with* `luamplib`; en allemand : *Einführung in LuaLATEX mit* `luamplib`.

1. Introduction

Avec une TEXLive à peu près à jour, le moteur LuaTEX en est à la version 1.13.2. En septembre 2016, LuaTEX a donc la version *beta* en passant à la version 1.0 pour rentrer dans le monde des moteurs utilisables « en production »[2]. Depuis, ce moteur s'installe de plus en plus dans le monde (LA)TEX et celui-ci a apporté beaucoup d'améliorations dans notre vie quotidienne. En effet, avec lui, nous disposons nativement de la gestion d'Unicode, des formats de polices OpenType et TrueType, de l'inclusion du langage Lua[3] comme langage de programmation embarqué, et de bien d'autres merveilles. Ici, nous parlerons d'un autre apport, à savoir la bibliothèque `mplib` de Lua qui est chargée dans LuaTEX et qui est une version embarquée du programme METAPOST[4].

Cette introduction à l'utilisation de METAPOST grâce à LuaTEX va se faire avec le format le plus courant[5] : LATEX. Les développeurs Hans Hagen, Taco Hoekwater, Élie Roux, Philipp Gesang et Kim Dohyun ont créé l'extension, ou *package*, `luamplib` [7], nous permettant de tirer profit de l'inclusion de `mplib` dans LuaTEX avec LATEX. L'utilisation de cette extension est extrêmement simple et c'est ce que nous allons présenter dans cet article. Nous ne parlerons ici que des fonctionnalités qui nous paraissent les plus utiles, et nous invitons à la lecture de la documentation [7] pour approfondir l'utilisation de cette extension.

2. Voir la référence [3] pour une introduction à LuaTEX et la référence [12] pour le manuel de référence.

3. Voir la référence [11] pour la programmation en Lua.

4. Il existe une extension (*package*) gmp.sty [5] qui permet d'utiliser du code METAPOST dans un document LATEX mais en appelant METAPOST de façon externe avec l'option de compilation `-shell-escape`. Cette méthode permet d'utiliser le moteur pdfTEX au lieu de LuaTEX et d'appeler METAPOST à l'intérieur de son document LATEX.

5. Cela fait maintenant de nombreuses années que nous pouvons utiliser METAPOST avec ConTEXt, et qu'ainsi nous pouvons directement coder en METAPOST dans le fichier .tex de ConTEXt.

Nous supposons, pour les divers exemples proposés, que c'est le format LaTeX qui est utilisé et que la compilation se fait avec la commande :

```
user $> lualatex monfichier.tex
```

Nous supposons aussi que les lecteurs et les lectrices sont familiers avec le langage METAPOST [9].

2. L'extension `luamplib`

L'extension `luamplib` nous permet de façon très simple de produire des dessins METAPOST directement dans nos documents LaTeX avec LuaLaTeX. La bibliothèque Lua `mplib` contient le programme METAPOST et est incluse dans LuaTeX. L'extension `luamplib` n'est qu'une simple interface pour que l'utilisateur ou l'utilisatrice ait accès au programme METAPOST depuis LaTeX.

Alors que PDF est la sortie standard avec LuaTeX, l'extension `luamplib` supporte même le mode DVI depuis sa version 2.7 (au moment de la rédaction de cet article, TeXLive embarque la version 2.20.7).

Lors de la production du document PDF (ou DVI), les figures METAPOST, produites donc grâce à la bibliothèque `mplib` de Lua, sont encapsulées dans des *hbox* avec les dimensions ajustées de la *bounding box* du résultat de la compilation par METAPOST (ou plus exactement `mplib`).

L'utilisation de l'extension est très simple : en *Plain TeX* il suffit de mettre son code METAPOST entre `\mplibcode` et `\endmplibcode`, avec LaTeX il suffit de le mettre dans l'environnement `mplibcode` comme ceci :

```
\begin{mplibcode}
  beginfig(0) ;
    draw (0,0)--(2cm,0) ;
  endfig ;
\end{mplibcode}
```

Ainsi, il devient très simple de faire des graphiques dans son document LaTeX grâce à METAPOST. Voici un exemple plus complexe.

```
\begin{mplibcode}
  u = 2cm ; v = 3 ;
  beginfig(1) ;
    % axes
    drawarrow (-1.5u,0)--(1.5u,0) withpen pencircle
      scaled 0.2 ;
    drawarrow (0,-1.5u)--(0,1.5u) withpen pencircle
      scaled 0.2 ;
    % unités
    draw (-2,u)--(2,u) withpen pencircle scaled 0.2 ;
    draw (u,-2)--(u,2) withpen pencircle scaled 0.2 ;

    % courbe
    draw (-u,-u){1,2.25}..(-0.5u,0)..(0,0.707u)..
      (0.5u,1u){1,0}..(1u,0){0,-1}..(0.5u,-1u){-1,0}..
      (0,-0.707u)..(-0.5u,0)..(-1u,1u){-1,2.25}
    withcolor (0.18,0.55,0.34) ;
    % tangentes
    drawarrow (-u,-u)--(-0.9u,-0.775u) withcolor red ;
    drawarrow (0.5u,1u)--(0.8u,1u) withcolor red ;
    drawarrow (1u,0)--(1u,-0.3u) withcolor red ;
    drawarrow (0.5u,-1u)--(0.2u,-1u) withcolor red ;
  endfig ;
\end{mplibcode}
```

Listing 1. Code d'une figure un peu plus complexe.

dont le résultat est visible en figure 1.

Cette extension nous donne donc accès à toute la puissance de METAPOST ainsi qu'aux nombreux outils développés en METAPOST *via* ses extensions ou *packages* et ceci en s'affranchissant d'un des plus gros défauts, selon moi, de METAPOST lorsqu'on l'utilise pour la production de dessins destinés à être inclus dans un document produit avec LaTeX : devoir compiler ses illustrations à côté en gérant les problèmes de polices, de composition avec TeX, *etc.*

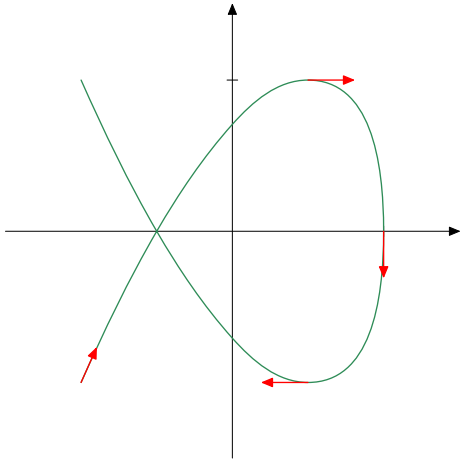


FIGURE 1. Résultat du code donné dans le listing 1.

Lors de la production d'une figure avec `mplibcode`, `luamplib` ne force ni le mode horizontal ni le mode vertical. Les figures ainsi produites ne respectent donc pas, par défaut, les commandes d'alignement comme `\centering` ou `\raggedleft`, à moins d'utiliser explicitement la commande `\leavevmode` grâce au mécanisme `verbatimtex` décrit en section 5.3. Nous verrons aussi, en section 5.4, qu'on peut ajouter cette commande automatiquement à tous les environnements `mplibcode`, avec la commande `\everymplib`.

3. ET LE TEXTE ?

Avec les premières versions de `luamplib`, il n'était pas possible d'utiliser les commandes classiques de METAPOST `btex ... etex` qui sont souvent essentielles. Celles-ci permettent de produire du texte en utilisant du code TEX à l'intérieur du code METAPOST. C'est désormais possible et c'est tant mieux. En ajoutant les lignes suivantes au code du listing 1 avant « `endfig ;` » :

```
%labels
label.lrt(btex $x$ etex,(1.3u,0)) ;
label.lrt(btex $y$ etex,(0,1.4u)) ;
label.rt(btex $1$ etex,(0,u)) ;
```

```
label.llft(btex $1$ etex,(u,0)) ;
label.lrt(btex $\begin{cases}
                x=\cos 2t \\[2mm] y=\sin 3t
              \end{cases}$ etex,
          (-u,-0.8u)) ;
```

on obtient la figure 2.

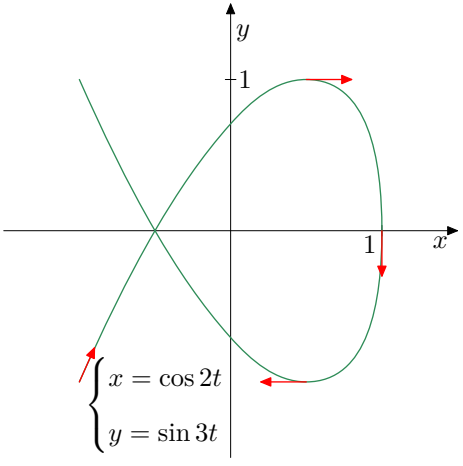


FIGURE 2. Résultat du code 1 avec l'ajout de légende grâce aux commandes `btex ... etex`.

Le texte est alors composé dans la police et dans le corps du document LaTeX dans lequel se trouve l'environnement `mplibcode` et qu'on va appeler *document principal*. De plus, on a accès à tout ce que le préambule du fichier LaTeX *principal* nous permet d'utiliser. Il existe aussi une fonction équivalente à `TEX()` du fichier `TEX.mp` : `textext()` qui s'utilise de la même façon que `TEX()` — en étant plus puissante — et qui permet d'écrire la valeur de la variable en argument. L'utilisation de la fonction `textext()` est illustrée dans le code suivant.

```
\begin{mplibcode}
  beginfig(0) ;
    for i := 0 upto 10 :
```

```
      label(textext("\( n_{" & decimal(i) & "} \)"),
            (5mm*i,0)) ;
    endfor ;
  endfig ;
\end{mplibcode}
```

$n_0\ n_1\ n_2\ n_3\ n_4\ n_5\ n_6\ n_7\ n_8\ n_9\ n_{10}$

La fonction `TEX()` est aussi autorisée comme un synonyme de la fonction `textext()`.

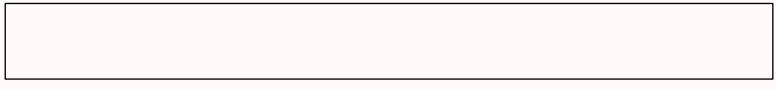
4. Les communications entre LaTeX et METAPOST

Dans cette section, nous allons voir comment l'information peut être transmise (en particulier les dimensions) depuis LaTeX vers METAPOST *via* l'extension `luamplib`.

4.1. Les dimensions de LaTeX

Dans la composition du document LaTeX, il est souvent naturel de vouloir réaliser une figure avec des dimensions qui sont reliées à la taille de la page, de la police, *etc.* Depuis la version 2.3 de `luamplib`, ceci est possible grâce à la commande `\mpdim`. En effet, cette commande LaTeX est autorisée dans le code `mplib`, et permet de transmettre une dimension depuis LaTeX vers METAPOST[6]. Pour avoir un cadre de la largeur de la ligne de composition, il suffit alors d'écrire :

```
\begin{mplibcode}
  beginfig(0) ;
    draw (0,0)--(\mpdim{\linewidth},0)--
      (\mpdim{\linewidth},-1cm)--(0,-1cm)--cycle ;
  endfig ;
\end{mplibcode}
```



6. Ceci reste vrai avec le format *Plain TeX*, comme la plupart des fonctionnalités présentées dans ce document.

4.2. Les couleurs de LaTeX

De la même façon que pour les dimensions, il existe la commande `\mpcolor` qui permet de transmettre à METAPOST les noms de couleurs ou les expressions des extensions `color/xcolor` pour les utiliser à l'intérieur d'un environnement `mplibcode`. Notons tout de même que l'extension `luamplib` ne charge pas automatiquement `color` ou `xcolor`[7]. La commande s'utilise comme `\mpdim` : si la couleur `green` est définie par une des extensions de gestion de couleur, alors on peut l'utiliser comme une couleur METAPOST avec `\mpcolor{green}`. Il est même possible d'utiliser un argument optionnel à `\mpcolor`, comme par exemple `\mpcolor[HTML]{008080}` permettant de fournir à METAPOST une couleur codée en HTML.

4.3. Depuis METAPOST vers LaTeX

Il est aussi possible de faire passer de l'information depuis l'environnement `mplibcode` vers le reste du code LaTeX. Le code LaTeX placé dans la fonction `VerbatimTeX(...)` ou entre « `verbatimtex` » et « `etex` », se trouvant entre « `beginfig(...)` » et « `endfig` », est inséré après que la figure en question a été traitée et évacuée vers la page PDF[8]. Dans le code suivant, la valeur de `D` est récupérée *via* la commande définie à la fin du traitement du code de l'environnement `mplibcode`.

```
\begin{mplibcode}
  D := sqrt(2)**7 ;
  beginfig(0) ;
    draw fullcircle scaled D ;
    VerbatimTeX
      ("Ceci se place après la figure. \gdef\Dia{" &
       decimal D & "}") ;
  endfig ;
\end{mplibcode}
```

7. Les extensions `(x)spotcolor` (pour le mode PDF) et `xespotcolor` (pour le mode DVI) sont aussi supportées.

8. Nous verrons en section 5.3 que lorsque les éléments `verbatimtex ... etex` sont utilisés avant « `beginfig(...)` », alors le code LaTeX produit est placé *avant* la boite hoizontale produite par `mplib`.

```
La valeur de \verb+D+ est \Dia.
```

 Ceci se place après la figure. La valeur de D est 11.3138.

Signalons que si la figure est vide, alors la commande `VerbatimTeX()` est ignorée.

4.4. Les dimensions de la figure produite

Il peut être intéressant de récupérer les dimensions des figures produites. Pour cela, `luamplib` fournit les commandes LaTeX `\MPwidth` qui stocke la largeur de la dernière figure produite et `\MPheight` qui en stocke la hauteur.

```
\begin{mplibcode}
  beginfig(0) ;
   draw fullcircle xscaled 1cm yscaled 0.5cm ;
  endfig ;
\end{mplibcode}
La dimension de la figure produite est de \MPwidth\
      par \MPheight.
```

 La dimension de la figure produite est de 28.95462pt par 14.72827pt.

De plus, les informations de la *bounding box* de la dernière figure produite sont stockées dans les commandes `\MPllx`, `\MPlly`, `\MPurx` et `\MPury`, mais contrairement aux deux commandes précédentes, elles ne contiennent pas l'unité `pt`.

4.5. La mémoire entre environnements `mplibcode`

La commande `\mplibcodeinherit{enable}` permet l'héritage des variables, des constantes, et des macros entre les divers environnements `mplibcode`. En effet, la valeur par défaut :

```
\mplibcodeinherit{disable}
```

fait que chaque contenu d'environnement `mplibcode` est traité indépendamment des autres. Voici un exemple d'utilisation d'une telle fonctionnalité :

```
\mplibcodeinherit{enable}
\begin{mplibcode}
  beginfig(0) ;
    u := 10 ;
    draw fullcircle scaled u ;
  endfig ;
\end{mplibcode}
\begin{mplibcode}
  beginfig(0) ;
    draw fullcircle scaled 2u ;
  endfig ;
\end{mplibcode}
```

5. Quelques fonctions plus sophistiquées

Nous présentons dans cette section quelques fonctionnalités un peu plus *avancées*.

5.1. Fichiers `.mp` externes

Lors de l'inclusion de fichiers externes avec la commande `input` de METAPOST un seul élément peut poser problème : le traitement des éléments `verbatimtex ... etex`, et `btex ... etex`. Pour les premiers, ils sont simplement ignorés[9] par le traitement de `luamplib` avant d'envoyer le code à la bibliothèque Lua `mplib`. Pour les éléments :

```
btex ... etex
```

`luamplib` inspecte tous les fichiers inclus par la commande `input` et traite des parties dans le cache si nécessaire. Lorsque le fichier à inclure ne

9. Avant la version 2.5 de l'extension, même les éléments `btex ... etex` dans les fichiers externes étaient ignorés.

contient pas de tels éléments, cela peut rendre la compilation inutilement plus longue. L'extension `luamplib` fournit des macros qui permettent à l'utilisateur ou utilisatrice de spécifier les fichiers qui ne requièrent pas l'inspection par `luamplib` :

```
\mplibmakenocache{<filename>[,<filename>,\ldots]}
\mplibcancelnocache{<filename>[,<filename>,\ldots]}
```

où les `<filename>` sont les noms de fichiers sans l'extension `.mp`.

5.2. Le format

Il existe principalement deux formats pour METAPOST : *plain* et *metafun*. Par défaut, c'est le format *plain* qui est utilisé, mais on peut choisir le format utilisé pour les figures suivantes à n'importe quel moment grâce à la commande `\mplibsetformat{}`.

Voici un exemple d'utilisation de *metafun* [6] qui permet, entres autres, d'avoir accès à la transparence.

```
\mplibsetformat{metafun}%
\begin{mplibcode}
  beginfig(0) ;
    path p ;
    p:= fullcircle scaled 2cm yshifted .5cm ;
    fill p withcolor
      transparent("normal", 0.5, red) ;
    fill p rotated 120 withcolor
      transparent("normal",0.5,green) ;
    fill p rotated 240 withcolor
      transparent("normal",0.5,blue) ;
  endfig ;
\end{mplibcode}%
```

5.3. `verbatimtex ... tex`

Lorsqu'on ne fait pas appel à un fichier `.mp` externe, les éléments `verbatimtex ... tex` qui sont avant « `beginfig()` » ne sont pas ignorés. Les commandes LaTeX qui se trouvent entre « `verbatimtex` » et « `etex` » sont insérées juste avant la boîte horizontale produite par `mplib`. Ainsi, on peut, par exemple, déplacer verticalement ou horizontalement chaque boîte contenant la figure produite. Dans les sections suivantes, nous allons voir quelques utilisations de ce mécanisme.

5.4. Factoriser des commandes

On peut grâce aux commandes LaTeX `\everymplib{...}` et `\everyendmplib{...}` factoriser du code METAPOST à insérer au début et à la fin de l'environnement `mplibcode`. Par exemple, il est souvent bien plus pratique de factoriser les « `beginfig(0) ;` » et « `endfig ;` », et ceci se fait grâce aux lignes suivantes :

```
\everymplib{ beginfig(0) ; }
\everyendmplib{ endfig ; }
```

Pour réinitialiser, il suffit d'invoquer ces même commandes avec un argument vide :

```
\everymplib{}\everyendmplib{}
```

En utilisant la fonctionnalité de la section précédente, on peut forcer le fait de quitter le mode vertical. En écrivant :

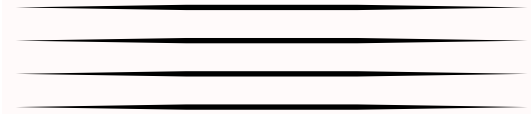
```
\everymplib{verbatimtex \leavevmode etex ;
            beginfig(0) ; }
\everyendmplib{ endfig ; }
```

`luamplib` insérera la commande `\leavevmode` avant la boîte horizontale produite par `mplib` et ainsi on aura un comportement plus intuitif concernant la réponse aux commandes comme `\centering`.

5.5. Sauvegarde de figures

On peut utiliser le mécanisme de `verbatimtex ... tex` pour sauvegarder des figures produites par un environnement `mplibcode` et les réutiliser ensuite comme dans l'exemple suivant :

```
\newbox\mympbox
\begin{mplibcode}
  verbatimtex \global\setbox\mympbox etex
  beginfig(0) ;
    breadth=.667\mpdim\linewidth;
    height=2pt ;
    x1=0 ; x2=x6=.333x4 ; x5=x3=.667x4 ; x4=breadth ;
    y1=y4=height/2 ; y2=y3=height ; y5=y6=0 ;
    fill z1--z2--z3--z4--z5--z6--cycle ;
  endfig ;
\end{mplibcode}%
\copy\mympbox \copy\mympbox
\copy\mympbox \copy\mympbox
```



5.6. Précision

METAPOST gère les quantités numériques dans quatre formats de représentation ou systèmes de numération. Lors de l'exécution du programme METAPOST en ligne de commande, cela se décide grâce à l'option `-numbersystem` dont l'argument est `scaled`, `double`, `binary` ou `decimal` (voir [9], annexe A). La représentation interne par défaut est `scaled`. L'extension `luamplib` permet d'utiliser la double précision (pour les machines 64 bits) en précisant :

```
\mplibnumbersystem{double}
```

5.7. Fichier de configuration

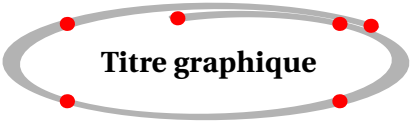
À la fin du chargement de l'extension `luamplib`, si le fichier `luamplib.cfg` existe, il est alors lu automatiquement. Les paramètres « globaux » de l'extension tels que `\everymplib` ont donc vocation à être définis dans ce fichier.

6. Des exemples

Dans cette section, nous présentons un panel d'exemples illustrant les possibilités que nous offrent LuaTEX, `mplib` et `luamplib` avec LaTeX.

6.1. Un exemple de titre

Comme nous l'avons dit en introduction, cela fait maintenant longtemps que MetaPost est intégré à ConTEXt, et il y a un exemple bien connu de style de titre de Taco Hoekwater[10] que nous pouvons maintenant reproduire avec `luamplib` et LaTeX. L'objectif est d'obtenir :



Pour cela, nous définissons une commande LaTeX qui va, grâce à MetaPost, tracer ce qui entoure le titre avec deux des points de contrôle qui seront placés de façon aléatoire (les quatre autres seront reliés directement à la *bounding box* du titre). Le code pour une telle fonction peut être le suivant[11] :

```
%arguments:
%#1 = largeur désirée
%#2 = hauteur désirée
%#3 = épaisseur du trait (relatif aux largeur/hauteur)
%#4 = couleur de la ligne
%#5 = couleur des points
\def\MPclipOne#1#2#3#4#5%
 {\begin{mplibcode}
    beginfig(0) ;
      w := #1 ; width := 100 ; wfactor := w/width ;
      h := #2 ; height := 100 ; hfactor := h/height ;
      color lightred ; lightred := (.90,.50,.50) ;
      color lightgray ; lightgray := (.95,.95,.95) ;
      color gray ; gray := (.50,.50,.50) ;
      def random_delta (expr d) = d-(uniformdeviate 2d)
```

10. Voir [10].

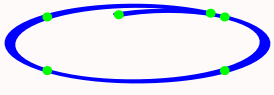
11. Le code que nous reproduisons ici est issu de la documentation de l'extension `gmp` dont nous avons déjà parlé.

```
    enddef ;
    z1 = (0,height) ; z2 = (0,0) ; z3 = (width,0) ;
    z4 = (width,height) ;
    z5 = (width+random_delta(.2width),
          height+random_delta(.2height)) ;
    z6 = (.5width+random_delta(.1width),
          height+random_delta(.1height)) ;
    pickup pencircle xscaled (#3/wfactor)
      yscaled (#3/(2*hfactor)) rotated 30 ;
    draw z5..z1..z2..z3..z4..z6 withcolor #4 ;
    pickup pencircle xscaled (#3/wfactor)
      yscaled (#3/hfactor) ;
    draw z1 withcolor #5 ; draw z2 withcolor #5 ;
    draw z3 withcolor #5 ; draw z4 withcolor #5 ;
    draw z5 withcolor #5 ; draw z6 withcolor #5 ;
    newwidth := (xpart (urcorner currentpicture)) -
                (xpart (llcorner currentpicture)) ;
    newheight := (ypart (urcorner currentpicture)) -
                 (ypart (llcorner currentpicture)) ;
    currentpicture :=
      currentpicture xscaled (w/newwidth)
        yscaled (h/newheight) ;
  endfig ;
\end{mplibcode}}
```

Listing 2. Macro TEX pour la production de ce qui entourera le titre.

Cette macro s'emploie de la manière suivante :

```
\MPclipOne{100pt}{30pt}{5}{blue}{green}
```



On peut alors utiliser cette macro avec une commande qui prend en argument un texte, construit la boite qui le contient (en mettant en `\Large` par exemple), et place le texte produit entouré par le dessin METAPOST. Ceci peut s'implémenter de la façon suivante :

```
% On déclare la boite
\newsavebox{\tacochapterbox}

% déclaration de la commande
\newcommand{\tacochapterhead}[1]{%
 % la boite a les dimensions de l'argument en Large et
 % gras
 \sbox\tacochapterbox{\Large\bfseries #1}
 {\ooalign{%
    % on récupère la largeur avec une marge et on
    % transmet grâce à \mpdim
    \MPclipOne{\mpdim{\wd\tacochapterbox+6pc}}
    % idem pour la hauteur
    {\mpdim{\ht\tacochapterbox+3pc}}
    {8} % l'épaisseur relative
    {(.7,.7,.7)} % gris pour le trait
    {red} % rouge pour les points
    % on décale verticalement de la moitié de la
    % hauteur + marge
    \cr\hfill%
    \raisebox{\dimexpr.5\ht\tacochapterbox+1.5pc\relax}
    % on place notre dessin
    {\box\tacochapterbox}\hfill}}}
```

Listing 3. Commande pour construire un titre entouré.

On peut alors construire notre titre de la façon suivante :

```
\tacochapterhead{Un titre entouré}
```

Ceci peut aussi s'utiliser avec l'extension `titlesec` pour modifier les commandes `\chapter` ou `\section` du document.

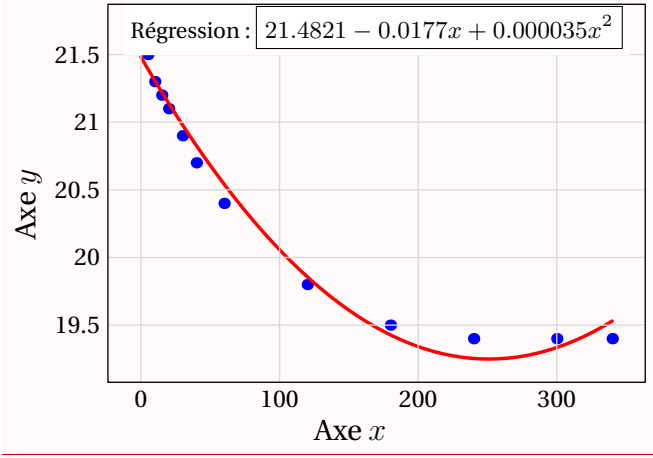
6.2. Utilisation de graph.mp

L'extension ou *package* METAPOST graph.mp, bien connue des utilisateurs et utilisatrices de METAPOST, peut, elle aussi, être utilisée avec luamplib. Ici, nous montrons un exemple qui trace un ensemble de points et leur régression d'ordre 2, le tout étant stocké dans deux fichiers externes tab.dat et regrtab.dat[12]. Nous renvoyons à la documentation de l'extension graph.mp [8] pour le détail du code utilisé.

```
\begin{mplibcode}
  picture bullet ;
  bullet = image(draw origin withpen pencircle
                   scaled 1.5mm withcolor blue) ;
  input graph ;

  beginfig(1) ;
    draw
      begingraph(\mpdim{0.7\linewidth},5cm) ;
        glabel.lft(btex \Large Axe $y$ etex
                     rotated 90,
                   OUT) ;
        glabel.bot(btex \Large Axe $x$ etex,OUT) ;
        gdraw "tab.dat" plot bullet ;
        gdraw "regrtab.dat"
          withcolor red
          withpen pencircle scaled 1.3pt ;
        autogrid(grid.bot,grid.lft)
          withcolor .85white ;
        setrange(whatever,whatever,whatever,
                 whatever) ;
        picture legende ;
        legende =
          btex \small Régression :
```

12. Cet exemple est tiré de http://melusine.eu.org/syracuse/journal/2008/07/21/perl/ où c'est un *script* Perl qui calcule la régression. On peut bien évidemment faire calculer la régression par Lua, comme il a été fait dans les *Cahiers GUTenberg n^os 54-55, voir* [4].

```
            $\boxed{21.4821-0.0177 x+0.000035x^{2}}$
          etex ;
        glabel(image(unfill bbox legende ;
                     draw legende),
               (170,21.7)) ;
        frame ;
      endgraph ;
  endfig ;
\end{mplibcode}
```



6.3. `mp-solid`

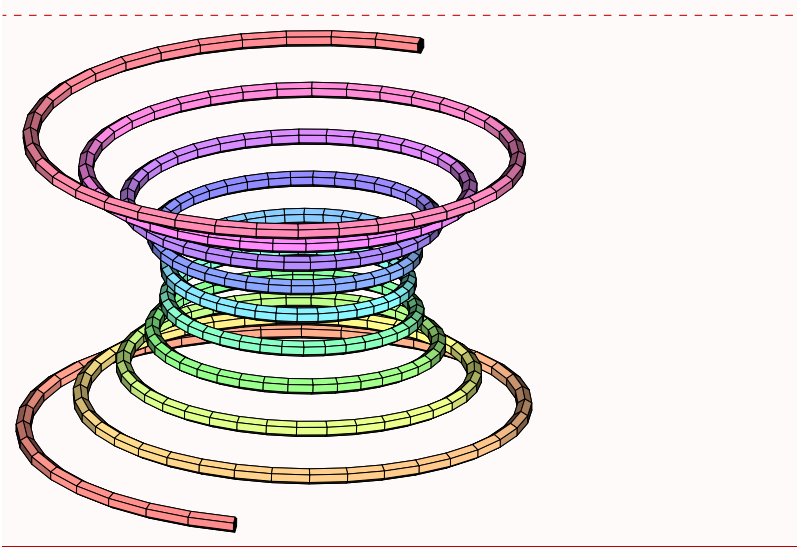
L'utilisation de l'extension `mp-solid` [13] de Christophe Poulain se fait aussi sans problème, si ce n'est le temps de compilation. En effet, les calculs pour la représentation 3D étant nombreux, on s'expose à une attente qui peut être conséquente. Notons tout de même que grâce au moteur LuaTEX, nous n'avons pas de problèmes de taille de mémoire (si fréquents avec l'utilisation de pdfTEX ou du programme METAPOST). Voici un exemple tiré du site Syracuse à l'adresse http://melusine.eu.org/lab/bmp/a_mp-solid/s_psurfaces/ex35.mpf?fig=1.

Comme il a été dit en section 4.5, l'héritage entre environnement `mplibcode`, autorisé avec la commande :

```
\mplibcodeinherit{enable}
```

pose problème dès que les variables contiennent des éléments `btex ... etex`. C'est le cas pour les extensions que nous venons d'utiliser : `graph.mp` et `mp-solid.mp`. Pour la compilation de ce document en particulier, il a fallu veiller à désactiver l'héritage au moment de l'utilisation de ces extensions.

```
\begin{mplibcode}
  input mp-solid ;
  figureespace(-10u,-10u,10u,10u) ;
  % ok ressort-siège
  nb:=6 ;
  % ok nb=6 10''
  invnormale:=-1 ;
  Initialisation(1000,20,20,15) ;
  arcenciel:=true ;
  draw
  Tuben("((t**2+3)*sin(15*t),(t**2+3)*cos(15*t),2*t)
        ","(2*t*sin(15*t)+15*((t**2)+3)*cos(15*t),2*t
        *cos(15*t)-15*((t**2)+3)*sin(15*t),2)
        ",0.2,-2,360,1/90) ;
  finespace;
\end{mplibcode}
```



6.4. Une animation avec `animate`

En guise de conclusion de cet article, vu la rapidité de calcul due au fait que le programme METAPOST est intégré à LuaTEX, il nous a semblé intéressant de montrer qu'on peut produire très rapidement et facilement des animations grâce à l'extension `animate` (voir [1]). Bien évidemment, l'intérêt pour les *Cahiers GUTenberg* en version papier est limité, mais vous pourrez observer le résultat en ligne sur le futur site des *Cahiers GUTenberg*.

L'extension `animate` permet de produire des animations lisibles dans un PDF par le lecteur *Acrobat Reader*[13]. Plusieurs options sont envisageables : soit produire l'animation à partir d'une séquence d'images pré-construites (sous différents formats, JPEG, PS, PDF, *etc.*), soit à partir d'images générées par du code LaTeX comme `tikz`, `pstricks`, *etc.*., et désormais… METAPOST ! C'est ce que nous présentons ici à partir d'un exemple d'animation qu'on peut trouver sur le site Syracuse à l'adresse suivante : http://melusine.eu.org/syracuse/metapost/animations/gerono/.

13. Malheureusement, à notre connaissance, il n'existe pas de lecteur PDF libre permettant de lire les animations produites.

Dans cet exemple, on paramétrise un certain point $P = (\cos\theta, \sin\theta)$ à l'aide du paramètre d'angle θ qui permet de construire le *lemniscate* de Gérono grâce à la recette suivante :

> *Soit P un point décrivant un cercle de centre O et de rayon a. On projette P en Q sur l'axe (Ox), puis Q en R sur le segment $[OP]$. Le* lemniscate *de Gerono est alors le lieu du point M de $[PQ]$ tel que $QM = QR$.*

À partir de là, on construit la commande qui prend pour argument la valeur de l'angle, et qui construit l'image correspondante. On va ici utiliser l'héritage qu'on a vu à la section 4.5 pour garder en mémoire les unités et la courbe construite au fur et à mesure, à partir des étapes précédentes.

```
\mplibcodeinherit{enable}
\newcommand{\lemniscate}[1]{%
 % #1: l'angle en degrés
 \begin{mplibcode}
   if (#1=0) :
     u := 3cm ; a := 2u ; h:=a/4 ; s := 1.3 ;
     path lemn ; path carre ;
     carre = (0,0)--(1,0)--(1,1)--(0,1)--cycle ;
   fi ;
   beginfig(#1) ;
     pickup pencircle scaled 0.6pt ;
     drawarrow (0,-s*u)--(0,s*u) ;
     drawarrow (-s*u,0)--(s*u,0) ;
     path cercle,hori,verti ; pair O,P,Q,R,M ;
     O := (0,0) ; cercle := fullcircle scaled (a) ;
     pickup pencircle scaled 0.4pt ;
     draw cercle dashed evenly withcolor blue ;
     % chemin qui parcourt le cercle
     P := (a/2*cosd(#1),a/2*sind(#1)) ;
     % projection sur (Ox)
     Q := (a/2*cosd(#1),0) ;
     % projection de Q sur [O,P]
     R =
       cosd(#1)*cosd(#1)*(a/2*cosd(#1),a/2*sind(#1)) ;
```

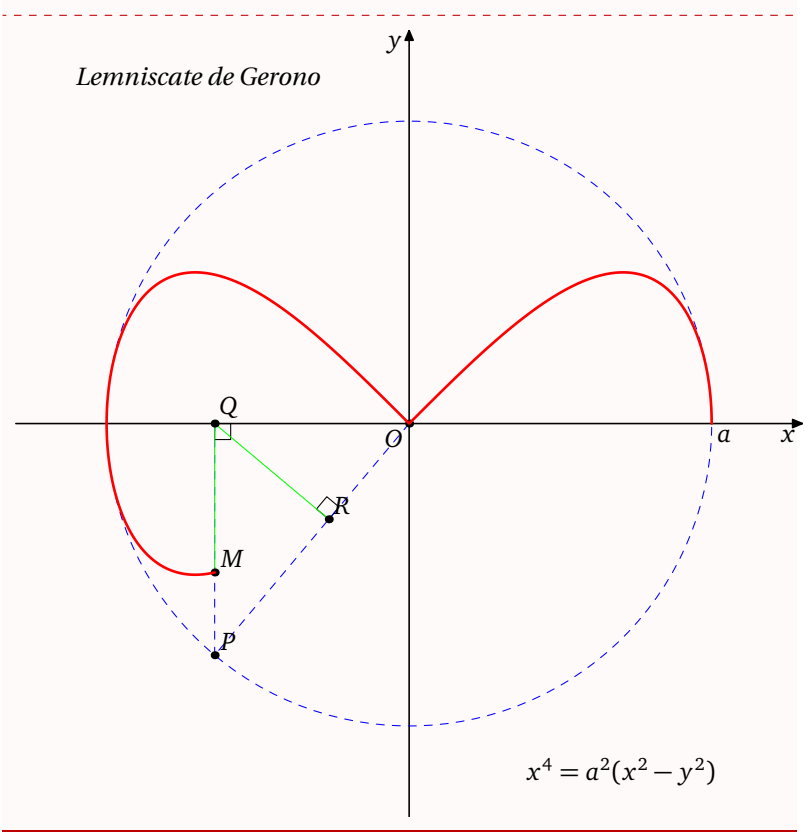
```
% M sur [PQ] tel que QM = QR
QR := abs(R-Q) ;
if (sind(#1)>0.0) :
  M = Q + (0,QR) ;
else :
  M = Q - (0,QR) ;
fi ;

if #1=0 :
  lemn:=M ;
else :
  lemn := lemn--M ;
fi ;

if ((#1>0) and (#1<90)) or
     ((#1>180) and (#1<270)) :
  draw carre scaled 6 rotated (angle(Q-P)+180)
    shifted Q ;
  draw carre scaled 6 rotated (angle(R-P))
    shifted R ;
fi ;
if ((#1>=90) and (#1<180)) or
     ((#1>=270) and (#1<360)) :
  draw carre scaled 6 rotated (angle(Q-P)+180)
    shifted Q ;
  draw carre scaled 6 rotated (angle(R-P)+180)
    shifted R ;
fi ;

draw O--P dashed evenly withcolor blue ;
draw P--Q dashed evenly withcolor blue ;
draw R--Q--M withcolor green ;

label.lrt(btex $a$ etex,(a/2,0)) ;
label.llft(btex $x$ etex,(s*u,0)) ;
label.llft(btex $y$ etex,(0,s*u)) ;
label(btex \itshape Lemniscate \par de Gerono
      etex,
```

```
        (-0.7u,1.15u)) ;
    label(btex $x^4=a^2(x^2-y^2)$ etex,
        (0.7u,-1.15u)) ;

    dotlabel.llft(btex $O$ etex,O) ;
    dotlabel.urt(btex $M$ etex,M) ;
    dotlabel.urt(btex $P$ etex,P) ;
    dotlabel.urt(btex $Q$ etex,Q) ;
    dotlabel.urt(btex $R$ etex,R) ;

    pickup pencircle scaled 1pt ;
    draw lemn withcolor red ;

    clip currentpicture to
      (-s*u,-s*u)--(s*u,-s*u)--(s*u,s*u)--(-s*u,s*u)--
      cycle ;
  endfig ;
\end{mplibcode}
}
```

Listing 4. Macro pour la construction des images de l'animation du tracé du *lemniscate* de Gerono.

Il suffit alors d'animer les différentes images avec l'extension `animate` et son environnement `animateinline`. Le résultat de l'animation — telle qu'elle est programmée dans le texte source ci-après — est visible sur la version en ligne du présent article, il est remplacé par la seule situation avec un angle de 230° dans la version imprimée sur papier.

```
\begin{animateinline}[poster=230,controls=on]{24}
 \multiframe{360}{ii=0+1}{\lemniscate{\ii}}
\end{animateinline}
```



Bibliographie

[1] Grahn Alexander: *The* animate *Package*. 2020. https://ctan.org/pkg/animate.

[2] Jacques André and Jean-Côme Charpentier: "Lexique anglo-français du *Compagnion*". *Cahiers GUTenberg*, vol. 49, pp. 19–45. http://www.numdam.org/item/CG_2007___49_19_0/. 2007.

[3] Association GUTenberg: *Cahiers n^os 54-55 : Introduction à LuaTEX*. http://www.numdam.org/issues/CG_2010___54-55/. October 2010.

[4] Maxime Chupin: "LuaLaTeX pour les non-sorciers, deux exemples". *Cahiers GUTenberg*, vol. 54-55, pp. 37–56. http://www.numdam.org/item/CG_2010__54-55_37_0/. 2010.

[5] Enrico Gregorio: *The* gmp *Package. Enable Integration between* METAPOST *Pictures and* LaTeX. Version 1.0. 4 July 2021. https://ctan.org/pkg/gmp.
[6] Hans Hagen: *Metafun.* V. 2.11.3. 2020. http://www.pragma-ade.com/general/manuals/metafun-p.pdf.
[7] Hans Hagen, Taco Hoekwater, Elie Roux, Philipp Gesang and Kime Dohyun: *The* luamplib *package.* V. 2.11.3. 2020. https://ctan.org/pkg/luamplib.
[8] John D. Hobby: *Drawing Graphs with* METAPOST. https://www.tug.org/docs/metapost/mpgraph.pdf. TUG.
[9] John D. Hobby and METAPOST development team: *METAPOST, a user's manual.* V. 2.0. 2019. https://ctan.org/pkg/metapost.
[10] Tom Hotten and Hans Hagen: *ConTeXt, an Excursion.* Version 990527. 27 May 1999. PRAGMA, http://www.pragma-ade.com/general/manuals/ms-cb-en.pdf.
[11] Roberto Ierusalimschy: *Programming in Lua.* Lua.org. 2016.
[12] LuaTeX development team: *LuaTeX Reference Manual.* Dernière version en ligne v. 1.12 : http://www.luatex.org/svn/trunk/manual/luatex.pdf. March 2020.
[13] Christophe Poulain: mp-solid. V. 1. 2011. http://melusine.eu.org/syracuse/metapost/mp-solid/mp-solid-doc.pdf.

◖ Maxime Chupin
Docteur en mathématiques appliquées, ingénieur de recherche cnrs au ceremade à l'Université Paris-Dauphine, ses activités de recherche concernent l'analyse numérique, l'optimisation, et le contrôle optimal. Membre de GUTenberg depuis 10 ans, il a rejoint le Conseil d'Administration très rapidement. Il s'est impliqué dans le partage de ressources sur le site Syracuse https://melusine.eu.org, notamment autour de METAPOST avec la production de nombreuses animations. Il est aussi l'auteur de deux *packages*, `bclogo` et `luamesh`, ainsi que de la classe `matapli` (https://www.ctan.org/pkg/{bclogo,luamesh,matapli}), disponibles sur le ctan. Il dispense aussi des formations à LaTeX au stage de Dunkerque, dans des laboratoires de recherche en mathématiques parisiens, ainsi que dans le *cursus* universitaire mathématique à Paris-Dauphine, avec l'écriture d'un polycopié *Très courte initiation à LaTeX* (https://plmlab.math.cnrs.fr/mchupin/initiation-latex).
29, rue Pierre et Marie Curie
91400 Orsay
chupin (at) ceremade (dot) dauphine (dot) fr
http://fougeriens.org/~mc