Le fichier variations.mod (0.9)

Pour TeXgraph 1.99

18 juillet 2021

Résumé

Description des macros de variations.mac et variations.mod.

Table des matières

1 Introduction

Le fichier modèle *variations.mod* permet de charger en mémoire les macros du fichier *variations.mac*, celles-ci permettent le dessin de tableaux de variations ou de signes. Lorsque ce modèle est chargé dans l'interface graphique, on voit apparaître un menu sous forme de boutons dont la plupart correspondent aux différentes macros. Une bulle d'aide s'affiche au passage de la souris sur les boutons ce qui rend l'usage assez intuitif.

2 Initialisation et dessin du tableau

De manière à pouvoir utiliser ce modèle dans un document LaTeX, un certain nombre de macros permettent de dessiner un tableau de variations « à la main ».

Premier exemple

```
\begin{texgraph}[name=tableau1,file]
Include "variations.mod";
Graph image = [
```

```
InitLig1("x","-\infty","0","\sqrt{2}","+\infty"),
InitCol1("f'(x)",+1,"f(x)",+1), traitH(1), traitV(1),
valeursLig(2,+1,"-","0","+",+1,"+"), traitH(2),
variationsLig(3,"+\infty",
        -2,"0", ["2",labelstyle :=framed,width :=8,linestyle :=dotted, arrows :=0],
        +2,"+\infty")
];
\end{texgraph}
```

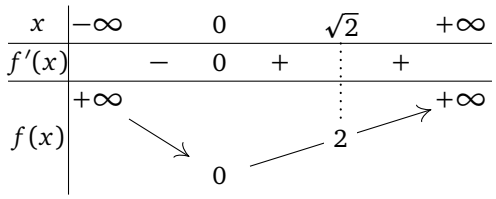


FIGURE 1 – *Un premier exemple*

Ces macros sont également utilisables dans l'interface graphique (dans un élément Utilisateur), ce qui permet d'utiliser le bouton *Grille* pour mieux visualiser le tableau et la position relative de ses éléments :

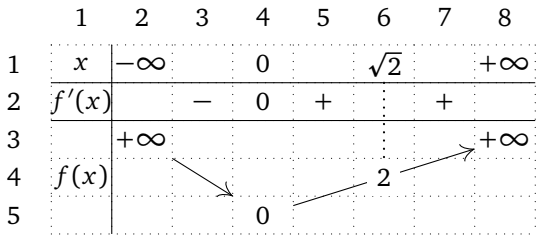


FIGURE 2 – Visualisation avec la grille dans l'interface

Afin que le graphique soit correct il y a un ordre à respecter dans la conception :

1. Initialiser la ligne 1 : c'est la macro **InitLig1**.
2. Initialiser la colonne 1 : c'est la macro **InitCol1**.
3. Remplir une ligne : c'est la macro **valeursLig**.
4. Dessiner les variations : c'est la macro **variationsLig**.

2.1 InitLig1

- **InitLig1(<["texte1", options]>, <argument2>, <argument3>, ...)**
- Cette macro remplit la ligne 1 et calcule le nombre de colonnes. Le *texte1* est placé en colonne 1. Les arguments suivants peuvent être :
  - un texte avec des options : *["texte", options]*, après l'affichage du texte on avance de 2 colonnes. si *argument2* est un texte il sera placé en colonne 2 (le suivant en colonne 4, etc).
  - un entier positif : il représente alors un incrément de colonne, par exemple +1 pour avancer d'une colonne.
- Les options sont des options d'attributs, par exemple : *["texte", Color :=blue]*. Attention : les textes suivants sur la ligne 1 seront également en bleu, mais à la fin de la macro l'attribut *Color* reprend sa valeur d'origine (idem pour les autres attributs).
- Cas particulier : pour peindre la ligne 1 on joue sur les options du premier texte, par exemple : *["texte1", FillColor :=yellow, FillStyle :=full]*.

2.2 InitCol1

- **InitCol1(<argument1>, <argument2>, ...)**
- Cette macro remplit la colonne 1 et calcule le nombre de lignes. Chaque *argument* peut-être :
  - un texte avec des options : *["texte", options]*, cela fonctionne comme dans la macro précédente sauf qu'après l'affichage d'un texte on passe à la ligne suivante. Si le premier argument est un texte, il sera placé dans la ligne 2.

  - un entier positif : il représente alors un incrément de ligne, par exemple +1 pour descendre d'une ligne.
- Cas particulier : pour peindre la colonne 1 (à partir de la ligne 2) on joue sur les options du premier argument, par exemple : *[<argument 1>, FillColor :=lightgreen, FillStyle :=full]*.

Un peu de couleurs

```
\begin{texgraph}[name=couleurs, file]
Include "variations.mod";
Graph image = [
InitLig1(["x", FillColor :=yellow, FillStyle :=full, Color :=blue],
      "-\infty","0","\sqrt{2}","+\infty"),
InitCol1(["f'(x)", FillColor :=lightgreen, FillStyle :=full], +1,"f(x)",+1),
FillColor :=lightcyan, Cadre(2,2,NbLig,NbCol, [FillStyle :=full, LineStyle :=noline]),
valeursLig([2,FillStyle :=full,FillColor :=firebrick,LineStyle :=noline],
       +1,"-","0","+",+1,"+"),
traitH(2),traitH(1), traitV(1),
variationsLig(3,"+\infty",
          -2,"0", ["2",labelstyle :=framed,width :=8,linestyle :=dotted, arrows :=0],
          +2,"+\infty")
];
\end{texgraph}
```

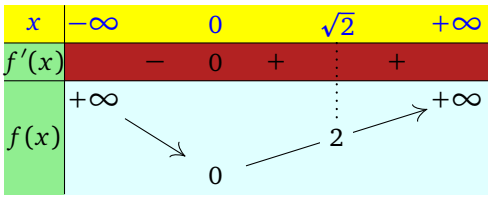


FIGURE 3 – *Un peu de couleurs*

Après cette deuxième étape, le tableau est dimensionné, mais si on veut une taille supérieure on peut commencer par fixer la taille du tableau avec la macro **dimTab(<nb lig>, <nb col>)** avant d'appeler les macros *InitLig1* et *InitCol1*. Celles-ci pourront ensuite augmenter la taille fixée si nécessaire, mais pas la diminuer.

```
\begin{texgraph}[name=avant, file]
Include "variations.mod";
Graph image = [
dimTab(6,8),
InitLig1("x","-\infty","0","+\infty"), traitH(1),
InitCol1(+1,"f",+1), traitV(1),
variationsLig(2,"1",-2,"0",+2,"1")
];
\end{texgraph}
```

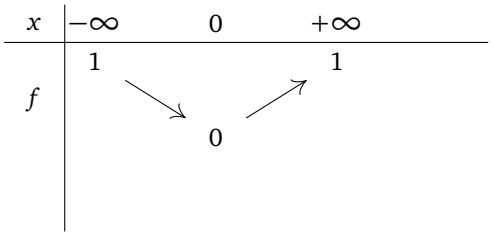


FIGURE 4 – *Avec dimTab() en premier*

```
\begin{texgraph}[name=apres, file]
Include "variations.mod";
Graph image = [
InitLig1("x","-\infty","0","+\infty"), traitH(1),
InitCol1(+1,"f",+1), traitV(1),
variationsLig(2,"1",-2,"0",+2,"1"),
dimTab(6,8), grille()
];
\end{texgraph}
```

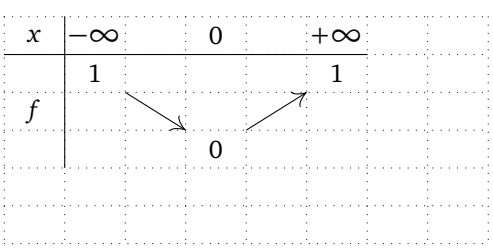


FIGURE 5 – *Avec dimTab() en dernier*

S'il est nécessaire de modifier la hauteur d'une ou plusieurs lignes ou la largeur d'une ou plusieurs colonnes, cela peut se faire avant l'initialisation de la ligne 1 et de la colonne 1. avec les macros suivantes :

- **defLargeCol(<[numéros]>, <valeur>, <[numéros]>, <valeur>, ...)**
- Les arguments *[numéros]* sont les listes des numéros des colonnes dont la largeur va être modifiée, par convention si cette liste est réduite à la valeur 0 cela signifie **toutes les colonnes**. L'argument *valeur* qui suit, représente la nouvelle largeur en unité graphique, par défaut, la largeur d'une colonne est de 1. (Voir l'exemple 2.3).
- **defLargeLig(<[numéros]>, <valeur>, <[numéros]>, <valeur>, ...)**
- Les arguments *[numéros]* sont les listes des numéros des lignes dont la hauteur va être modifiée, par convention si cette liste est réduite à la valeur 0 cela signifie **toutes les lignes**. L'argument *valeur* qui suit, représente la nouvelle hauteur en unité graphique, par défaut, la hauteur d'une ligne est de 1. (Voir l'exemple 2.3).

D'une manière générale, on se souviendra que la taille du tableau doit être fixée avant la première instruction de dessin. Lorsque cette taille est fixée, on peut utiliser les variables **NbLig** et **NbCol** qui représentent respectivement le nombre de lignes et le nombre de colonnes.

2.3 valeursLig

- **valeursLig(<lig>, <argument 1>, <argument 2, ...)**
- Cette macro remplit la ligne numéro *lig* en commençant par la colonne 2. Chaque *argument* peut-être :
  - un texte avec des options : *["texte", options]*, après l'affichage du texte on passe à la colonne suivante. Si *argument 1* est un texte, il sera placé dans la colonne 2. Les options sont :
    - les options d'attributs comme *Color*, *FillColor*, ... **sauf *LineStyle* et LabelStyle**, les changements s'appliquent jusqu'à la fin de la ligne.
    - linestyle := < style de ligne > : réglée à *noline* par défaut, cette option permet de tracer ou non un trait vertical dans la ligne sous le label. Le changement de cette option est local.
    - width := < épaisseur > : réglée à la valeur de **Width** par défaut, cette option permet de modifier l'épaisseur du trait vertical dans la ligne sous le label. Le changement de cette option est local.
    - labelstyle := < style de label > : réglée à *center* par défaut, cette option permet de modifier le style du label. Le changement de cette option est local. Lorsque le style est *framed*, le cadre est peint mais le bord n'est pas dessiné.
  - un entier positif : il représente alors un incrément de colonne, par exemple *+1* pour avancer d'une colonne.
- Cas particulier : pour peindre la ligne (à partir de la colonne 2) on joue sur les options du premier argument, par exemple : *[<lig>, FillColor :=lightgreen, FillStyle :=full]*.

La macro valeursLig

```
\begin{texgraph}[name=valeurLig, file]
Include "variations.mod";
Graph image = [
defLargeCol(1,2), defHautLig(0,2),
InitLig1(["x",FillStyle :=full, FillColor :=lightblue],+1,"-2","-1","0","1",+1),
InitCol1(["x^2-1",FillStyle :=full, FillColor :=lightblue], "x(x+2)","\frac{x^2-1}{x(x+2)}"),
valeursLig(2,"+",+1,"+",["0",linestyle :=solid,labelstyle :=framed],
  "-",+1,"-",["0",linestyle :=solid, labelstyle :=framed],"+"),
valeursLig(3,"+",["0",linestyle :=solid, labelstyle :=framed],
  "-",+1,"-",["0",linestyle :=solid, labelstyle :=framed],"+",+1,"+"),
valeursLig(4,"+",+1,"-",["0",linestyle :=solid, labelstyle :=framed],
  "+",+1,"-",["0",linestyle :=solid, labelstyle :=framed],"+",+1,"+"),
doubleB(3,4,4), doubleB(7,4,4),
grille([LineStyle :=solid,Color :=gray]),traitH(1),traitV(1)
];
\end{texgraph}
```

x		-2		-1		0		1	
x^2-1	$+$		$+$	0	$-$		$-$	0	$+$
$x(x+2)$	$+$	0	$-$		$-$	0	$+$		$+$
$\frac{x^2-1}{x(x+2)}$	$+$	‖	$-$	0	$+$	‖	$-$	0	$+$

FIGURE 6 – *La macro valeursLig*

2.4 variationsLig

- **variationsLig(<lig>, <argument2>, <argument3>, ...).**
- Cette macro permet de « dessiner » les variations d'une fonction, la ligne de référence est celle dont le numéro est le premier argument *lig*. Si l'argument suivant est un texte il sera affiché en colonne 2 sur la ligne de référence. Les arguments peuvent être :
  - **un entier** : celui-ci représente alors un changement de ligne, par exemple -2 pour descendre de 2 lignes. S'il est nécessaire de changer également de colonne, alors on peut ajouter un partie imaginaire à l'argument, par exemple : $-2+i$ pour descendre de 2 lignes avancer d'une colonne.
  - **jump** : la présence de cette constante fait qu'il n'y aura pas de flèche entre le dernier label et le prochain.
  - **["texte", options]** : affiche le *"texte"* dans la ligne courante et la colonne courante, le numéro de colonne sera ensuite incrémenté de 2 pour le texte suivant. Les options peuvent être :
    - les options d'attributs comme *Color*, *FillColor*, ... **sauf *LineStyle*, LabelStyle et Width**, les changements s'appliquent jusqu'à la fin de la ligne.
    - linestyle := < style de ligne > : réglée à *noline* par défaut, cette option permet de tracer ou non un trait vertical dans la colonne du label. Le changement de cette option est local.
    - width := < épaisseur > : réglée à la valeur de **Width** par défaut, cette option permet de modifier l'épaisseur du trait vertical. Le changement de cette option est local.
    - labelstyle := < style de label > : réglée à *center* par défaut, cette option permet de modifier le style du label. Lorsque le style est *framed*, le cadre est peint mais le bord n'est pas dessiné. Le changement de cette option est local.
    - arrows := < 0/1 > : réglée à 1 par défaut, cette option permet de dire si ce label est ajouté par dessus une flèche (cas où *arrows :=0*) ou si une flèche arrive sur ce label et en repart. Le changement de cette option est local.
    - ligdep := < entier>0 > : initialisée à 2, cette option représente la ligne de départ lorsqu'un trait vertical doit être dessiné. Un trait est dessiné lorsque l'option *linestyle* est différente de *noline*, le trait part du haut de la ligne. Le changement de cette option est local.
    - ligarr := < entier>0 > : initialisée à la ligne courante, cette option représente la ligne d'arrivée lorsqu'un trait vertical doit être dessiné. Un trait est dessiné lorsque l'option *linestyle* est différente de *noline*, le trait arrive de la ligne. Le changement de cette option est local.

La macro variationsLig

```
\begin{texgraph}[name=variationsLig, file]
Include "variations.mod";
Graph image = [
 defLargeCol(1,1.5), defHautLig(2,1.5),
 InitLig1("x","-\infty","0","1","\sqrt{2}","\sqrt{3}","+\infty"),
 InitCol1("f(x)",+1,"f(x)",+1), exclusion(2,4,5,6), traitH(1), traitV(1),
 valeursLig(2,+1,"-",+3,"+",
        ["0", labelstyle :=framed, linestyle :=solid],
         "+"), traitH(2),
 variationsLig(3,"+\infty",
           -2,["0",labelstyle :=right],
           jump,
```

```
        ["\,2", labelstyle :=left],
        ["3",  labelstyle :=framed, linestyle :=solid, arrows :=0, ligdep :=3],
        ["4",  labelstyle :=framed, width :=8, linestyle :=dotted, arrows :=0],
        +2,["+\infty", width :=4])
];
\end{texgraph}
```

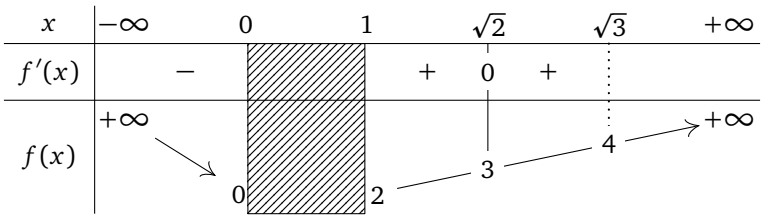
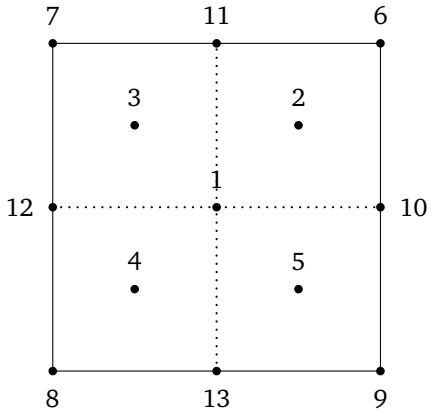


FIGURE 7 – *La macro variationsLig*

3 Repérage

Les colonnes sont numérotées de gauche à droite de 1 à *NbCol* et les lignes de haut en bas de 1 à *NbLig*. Chaque case est donc repèrée par un couple *(lig,col)*. Dans une telle case, les affixes des points numérotés de 1 à 13 sur la figure suivante, sont données par les macros suivantes :



1. **pos(<lig>, <col>)**
2. **posNE(<lig>, <col>)**
3. **posNO(<lig>, <col>)**
4. **posSO(<lig>, <col>)**
5. **posSE(<lig>, <col>)**
6. **coinNE(<lig>, <col>)**
7. **coinNO(<lig>, <col>)**
8. **coinSO(<lig>, <col>)**
9. **coinSE(<lig>, <col>)**
10. **posE(<lig>, <col>)**
11. **posN(<lig>, <col>)**
12. **posO(<lig>, <col>)**
13. **posS(<lig>, <col>)**

4 Autres macros de dessin

4.1 Cadre

- **Cadre(<lig1>, <col1>, <lig2>, <col2>, <[options]>)**
- Cette macro dessine le contour du rectangle défini par les deux cases *(lig1,col1)* et *(lig2,col2)* incluses.
- L'argument *[options]* (facultatif) contient une liste d'instructions concernant les attributs. Le changement des attributs est local.
- Exemple : *Cadre(1,1,NbLig,NbCol,[FillStyle :=full, FillColor :=pink])* (à placer après le dimensionnement du tableau et avant les macros qui écrivent dans le tableau).

4.2 Diag

- **Diag(<lig1>, <col1>, <lig2>, <col2>, <type (1/2)>, [options])**

- Cette macro dessine une diagonale du rectangle défini par les deux cases *(lig1,col1)* et *(lig2,col2)* incluses. La diagonale est descendante lorsque l'argument *type* vaut 1, et montante lorsque le *type* vaut 2.
- L'argument *[options]* (facultatif) contient une liste d'instructions concernant les attributs. Le changement des attributs est local.
- Exemple : *Diag(1,1,NbLig,NbCol, 2, [LineStyle :=dashed, Color :=blue])* (à placer après le dimensionnement du tableau).

4.3 doubleB

- **doubleB(<col>, <lig1>, <lig2>, [options])**
- Cette macro dessine une double barre dans la colonne *col* allant du haut de la ligne *lig1* au bas de la ligne *lig2*.
- L'argument *[options]* (facultatif) contient une liste d'instructions concernant les attributs. Le changement des attributs est local.
- L'écartement entre les deux barres peut être réglé en modifiant la variable **decal**, celle-ci vaut 1/15 par défaut.

4.4 exclusion

- **exclusion(<lig1>, <col1>, <lig2>, <col2>, <[options]>)**
- Cette macro dessine et remplit le rectangle défini par les deux cases *(lig1,col1)* et *(lig2,col2)* incluses à moitié seulement.
- L'argument *[options]* (facultatif) contient une liste d'instructions concernant les attributs. Le changement des attributs est local. Par défaut l'attribut *FillStyle* vaut *bdiag* et l'attribut *FillColor* vaut *black*.

4.5 fleche

- **fleche(<lig1>, <col1>, <lig2>, <col2>, ...)**
- Cette macro dessine une série de flèches dans les attributs courants, allant de la case *(lig1,col1)* à *(lig2,col2)*, puis de *(lig2,col2)* à *(lig3,col3)*, etc.
- Les lignes dessinées ne traversent pas les cases mentionnées, mais s'arrêtent à leur frontière.
- Il est possible de modifier localement les attributs avec le premier (ou deuxième) argument : *[<lig1>, options]*.

4.6 grille

- **grille(<[options]>)**
- Cette macro dessine la grille de repèrage avec les attributs définis par les *[options]* (facultatives). Par défaut, l'attribut *LineStyle* vaut *dotted*.

4.7 simpleB

- **simpleB(<col>, <lig1>, <lig2>, [options])**
- Cette macro dessine une barre dans la colonne *col* allant du haut de la ligne *lig1* au bas de la ligne *lig2*.
- L'argument *[options]* (facultatif) contient une liste d'instructions concernant les attributs. Le changement des attributs est local.

4.8 traitH

- **traitH(<lig> [, <col1>, <col2>, <position>, <double>])**

- Cette macro dessine un trait horizontal dans la ligne *lig* allant de la gauche de la colonne *col1* à la droite de la colonne *col2* (de 1 à *NbCol* par défaut). L'argument *<position>* est un nombre entre 0 et 1 qui indique à quel endroit de la ligne est tracé le trait : 0 pour le haut de la ligne jusqu'à 1 pour le bas de la ligne (valeur par défaut). L'argument *<double>* vaut 0 ou 1 et permet de dire si le trait est simple (valeur 0) ou double (valeur 1), par défaut le trait est simple.
- Cas particulier : bien que formellement il n'y ait pas de ligne d'indice 0, la commande **traitH(0)** tracera un trait horizontal en haut de la ligne 1.

4.9 traitV

- **traitV(<col> [, <lig1>, <lig2>, <position>, <double>])**
- Cette macro dessine une barre dans la colonne *col* allant du haut de la ligne *lig1* au bas de la ligne *lig2*. (de 1 à *NbLig* par défaut). L'argument *<position>* est un nombre entre 0 et 1 qui indique à quel endroit de la colonne est tracé le trait : 0 pour le côté gauche de la colonne jusqu'à 1 pour le côyé droit de la colonne (valeur par défaut). L'argument *<double>* vaut 0 ou 1 et permet de dire si le trait est simple (valeur 0) ou double (valeur 1), par défaut le trait est simple.
- Cas particulier : bien que formellement il n'y ait pas de colonne d'indice 0, la commande **traitV(0)** tracera un trait vertical à gauche de la colonne 1.

Index