

# VarTable

VarTable est un paquet pour rendre la réalisation  
des tableaux de signe plus simple

Ce paquet est construit sur [Cetz](#)

(version : 0.2.1)

## Table des Matières

1 - Introduction .....	2
2 - Tabvar .....	3
2.1 - description générale .....	3
tabvar .....	3
Parameters .....	3
variable .....	3
label .....	3
domain .....	4
contents .....	4
table-style .....	4
nocadre .....	4
arrow-mark .....	4
arrow-style .....	4
line-0 .....	5
line-style .....	5
hatching-style .....	5
first-column-width .....	5
first-line-height .....	5
element-distance .....	5
values .....	6
add .....	6
2.2 - Le paramètre de contenus .....	7
2.2.1 - Le format pour les Signes .....	7
2.2.1.1 - Un array classique pour les signes .....	7
2.2.1.2 - Une bar de séparation customisé .....	8
2.2.1.2.1 - Le style de la bar .....	8
2.2.1.2.2 - Le type de la bar .....	8
2.2.1.3 - Un même signe pour plus d'une seule valeur .....	9
2.2.1.4 - Hachurage pour une zone non définie pour les sous tableaux de signes ..	9

## 1 - Introduction

Ce paquet a été réalisé pour rendre la création de tableau de signe plus simple. Pour cela, ce paquet fournis la fonction « `tabvar` », dont les arguments sont décrit dans cette documentation.

Si vous rencontrez un bug, merci de me prévenir via mon [GitHub](#).

## 2 - Tabvar

### 2.1 - description générale

#### tabvar

Retourne un tableau de variation

#### Parameters

```
tabvar(  
  variable: content,  
  label: array,  
  domain: array,  
  contents: array,  
  table-style: style,  
  nocadre: bool,  
  arrow-mark: mark,  
  arrow-style: style,  
  line-0: bool,  
  line-style: style,  
  hatching-style: tiling,  
  first-column-width: length,  
  first-line-height: length,  
  element-distance: length,  
  values: array,  
  add: content  
)
```

**variable**    content

variable est la variable qui contient la variable du tableau (comme  $x$  ou  $t$ )

**Exemple:** si la variable de la fonction est  $t$ , alors:

variable : \$ t \$

Default: \$ x \$

**label**    array

label est un array qui contient des array de longueur 2, une pour chaque ligne du tableau, dont le premier élément est le titre de la ligne et le second est le type de la ligne: signe (s) ou variation (v)

**Exemple:** pour le tableau de variation de la fonction  $f$ , vous devriez écrire:

```
label : (  
  ([Signe de $f$], "s"), // la première ligne est un tableau de signe  
  ([Variation de $f$], "v") // la seconde ligne est un tableaux de variation  
)
```

Default: ()

**domain**    array

les valeurs prises par la variable

par exemple, si votre fonction change de signe ou atteint un extremum pour  $x \in \{0, 1, 2, 3\}$  vous devriez écrire:

domain: ( $\$0$ ,  $\$1$ ,  $\$2$ ,  $\$3$ )

Default: ()

**contents**    array

le contenu de la table

voir 2.2 pour plus de détaille

Default: ((),)

**table-style**    style

### Optionelle

Le style de la table

le type style est définis par Cetz, ainsi je vous recommande de vous référer au [manuelle de Cetz](#).

**Attention :** Si vous ne mettez pas le paramètre de style: mark a none, alors toute les lignes du tableau aurons une tête en flèche

Default: (stroke: 1pt + black, mark: (symbol: none))

**nocadre**    bool

Pour cacher le cadre externe du tableau

Default: false

**arrow-mark**    mark

Le style de la tête de flèche.

**N.B.** le type mark est définis par Cetz

Default: (end: "straight")

**arrow-style**    style

### Optionelle :

Le style des flèches.

**Attention :** le paramètre mark est supplenté par le paramètre arrow-mark

Default: (stroke: black + 1pt)

**line-0**    `bool`

**Optionelle**

si vous voulez changer la bar par défaut dans les tableaux de signe, pour une bar avec un zéro en sont centre

Default: `false`

**line-style**    `style`

**Optionelle**

Si vous voulez le style de toutes les bars de séparation entre les signes

Default: `(stroke: black + 1pt)`

**hatching-style**    `tiling`

**Optionelle**

le style des hachures s'il y a des zones hachurées

Default: `tiling(size: (30pt, 30pt))`

```
#place(line(start: (0%, 100%), end: (100%, 0%), stroke: 2pt))
#place(line(start: (-100%, 100%), end: (100%, -100%), stroke: 2pt))
#place(line(start: (0%, 200%), end: (200%, 0%), stroke: 2pt))
]
```

**first-column-width**    `length`

**Optionelle**

change la largeur de la première colonne

Default: `none`

**first-line-height**    `length`

change la hauteur de la première ligne (celle du domaine et de la variable)

Default: `none`

**element-distance**    `length`

**Optionelle**

change la distance entre deux éléments

Default: `none`

**values** `array`

pour ajouter des valeurs entre deux valeurs prè-définis

Default: `(( , )`

**add** `content`

Pour ajouter plus d'éléments via Cetz

Default: `()`

## 2.2 - Le paramètre de contenus

Le paramètre contenu est un array avec un élément par ligne (par label).

Chaque éléments sont eux même des array avec un élément pour chaque colonne, avec un format différents pour les signes et les variations qui seront détaillés ci-dessous.

### 2.2.1 - Le format pour les Signes

Il doit être positionné au même index dans l'array contents que un label possédant le string "s", ce qui indique que la ligne doit être considéré comme un tableau de signe

De plus, il doit contenir autemps d'éléments que le domaine moins un (un par interval), plus un argument optionelle pour pour le sytle de la bar

Chaque éléments doits être d'une de ces forme, différentes formes peuvent être utilisées sur une même ligne :

() - Vide : pour étendre le dernier signe en partant de la gauche sur les intervalles marqués vides

body - Le cas basique, constitué du type body de typst, comme  $\$ + \$$  ou  $\$ - \$$

(style de la bar, body) - Pour spécifier un style particulier à la bar de **devant** le signe, ce style peut être : "|" la bar simple, "||" une double bar ou "0" pour une bar avec un zéro en sont centre

**NB** : le paramètre line-0 change la bar par défaut pour la bar avec un zéro "0".

Vous pouvez mettre en plus à la fin le string "||", pour rajouter un double bar à la toute fin

#### 2.2.1.1 - Un array classique pour les signes

Un tableau de signe classique :

```
#tabvar(  
  init: (  
    variable: $t$,  
    label: ([signe], "s"),  
  ),  
  domain: ($2$, $4$, $6$, $8$),  
  contents: (($+$, $-$, $ + $)),  
)
```

$t$	2	4	6	8
signe	+	-	+	

Un exemple plus complexe :

```
#tabvar(  
  variable: $t$,  
  label: (  
    [signe], "s"),  
  ),  
  domain: ($ 2 $, $4$, $6$, $8$),  
  contents: (  
    ("Hello world !", $-$, $ 3 / 2 $),  
  ),  
)
```

$t$	2	4	6	8
signe	Hello world !	-	$\frac{3}{2}$	

**Note** : Sur le second exemple, le tableau est comprimé à l'aide de la fonction scale

### 2.2.1.2 - Une bar de séparation customisé

#### 2.2.1.2.1 - Le style de la bar

Vous pouvez modifier le style de la bar

Le style de la bar est un dictionary, du type "style" définis par Cetz.

Pour fair simple, si vous voulez changer uniquement le stroke des bars, vous avez juste a mettre (stroke: votre stroke).

Pour des usages plus complexe référer vous au manuel de Cetz.

**Exemple :**

```
#tabvar(  
  line-style: (  
    stroke: (paint: red, dash: "dashed")  
  ),  
  variable: $t$,  
  label: ([[signe], "s"),),  
  domain: ($2$, $4$, $6$,),  
  contents: (  
    ($+$, $-$),  
  ),  
)
```

$t$	2	4	6
signe	+	—	—

#### 2.2.1.2.2 - Le type de la bar

Pour tout les signes sauf le premier, au lieu de placé directement un signe, vous pouvez mettre un couple, dont le premier éléments définis le type de la bar placée avant le signe.

Il y a trois type différents de bar :

- "|": une bar simple
- 0: une bar avec un zéro en sont centre
- || une double bar, pour les valeurs non-définis

**Exemple**

```
#tabvar(  
  variable: $t$,  
  label: ([[signe], "s"),),  
  domain: ($2$, $4$, $6$, $8$, $10$,),  
  contents: (  
    (  
      $+$,  
      ("|", $-$),  
      ("0", $-$),  
      ("||", $+$)  
    ),  
  ),  
)
```

$t$	2	4	6	8	10
signe	+	—	0	—	+



Si vous voulez avoir une double bar avant le premier signe, vous pouvez utiliser le couple avec en premier éléments "| |", à la place du premier signe ; pour mettre une double bar à la fin, ajoutez à la fin de l'array le string "| |".

**Exemple :**

```
#tabvar(
  variable: $t$,
  label: ([[signe], "s"),),
  domain: ($2$, $4$, $6$),
  contents: (
    (
      ("| |", $+$),
      $-$,
      "| |"
    ),
  ),
)
```

$t$	2	4	6
signe	+	−	

### 2.2.1.3 - Un même signe pour plus d'une seule valeur

Quand votre tableau de signe possède plus d'un sous tableau, alors vous seriez tenté de vouloir mettre un même signe pour plusieurs valeurs du domaine.

Pour cela c'est assez simple, au lieu de mettre un signe directement, mettez simplement un couple vide ()

**Exemple :**

```
#tabvar(
  line-0: true,
  variable: $t$,
  label: (
    ([signe], "s"),
  ),
  domain: ($2$, $4$, $6$, $8$),
  contents: (
    ($+$, (), $-$),
  ),
)
```

$t$	2	4	6	8
signe		+	0	−

### 2.2.1.4 - Hachurage pour une zone non définie pour les sous tableaux de signes

Il se peut que vos fonctions ne soient pas définies sur un ou plusieurs interval malheureusement présent dans le domaine du tableau de signe, pour cela la convention veut que l'on hache la zone en question.

Étant donnée que les signes porte sur les interval du domaine, il en résulte une syntaxe relativement simple d'usage, dont on pourrait distinguer 4 cas :

- le premier cas et le plus courant, celui où les deux bornes de l'intervalle indéfini le sont également, ainsi à la place où vous auriez mis votre signe (ou tout autres éléments), vous renseignerez l'élément suivant : "| h |"
- le second cas, également relativement présent, est celui où les deux bornes elle définies contrairement cette fois à l'intervalle, ainsi vous omettez les deux bar «|» de l'élément présenté ci-dessus, i.e. vous renseignerez "h"

- les deux autres cas, moins courant mais pouvant tout de même apparaître, est celui où seul l'une des deux bornes est définie, ainsi, comme vous l'auriez sans doute compris, retirer (resp. rajouter) la barre pour le côté où l'élément est défini (resp. indéfini), soit : pour une valeur définie à gauche " $h|$ "; pour une valeur définie à droite " $|h$ "

**Remarque :** Vous avez sans doute compris que la barre « $|$ » symbolise les doubles barres indéfinies, de même que le « $h$ » représente le « $h$ » de hachurage, ainsi il est naturel de mettre ou non les barres au besoin

Pour étendre le hachurage sur plus d'un des intervalles du domaine, il vous suffit de sauter l'élément suivant avec toujours la même notation, à savoir ( )

**Exemple :**

```
#tabvar(
  variable: $t$,
  label: (
    ([Pour `|h|`], "s"),
    ([Pour `h|`], "s"),
    ([Pour `|h`], "s"),
    ([Pour `h`], "s"),
  ),
  domain: ($ 2 $, $4$, $5$, $8$,
$9$),
  contents: (
    ($+$, "|h|", (), $-$),
    ($-$, "h|", $-$, $+$),
    ($+$, "|h", (), $-$),
    ($-$, "h", $-$, $+$),
  ),
)
```

$t$	2	4	5	8	9
Pour $ h $	+				-
Pour $h $	-			-	+
Pour $ h$	+				-
Pour $h$	-			-	+