L'extension pour TEX et LATEX

systeme

v0.2beta

6 mars 2011

Manuel de l'utilisateur

Christian TELLECHEA
unbonpetit@gmail.com

Résumé

Cette petite extension met en forme des systèmes d'équations ou d'inéquations où les termes et les signes sont alignés verticalement, tout en permettant une saisie quasi naturelle.

Table des matières

0.1 Avant propos

Tout a recommencé, comme chaque année lorsque j'enseigne les systèmes d'équations, par un (petit) énervement concernant la difficulté de la saisie pour avoir une mise en forme acceptable. C'est à chaque fois un casse tête et une perte de temps conséquente de se battre avec les tableaux LaTeX pour obtenir *in fine* des systèmes avec un alignement correct, d'où le petit énervement, surtout lorsque, insouciant, on commence à taper ses sujets vers 23h pour le lendemain.

Fort de ce constat, je me suis dit qu'il allait falloir écrire des macros pour être débarrassé de la difficulté de la saisie. Et tant qu'à faire, autant écrire des macros en plain εTeX, que tout le monde puisse en profiter [1]. Ces macros sont réunies dans cette petite extension maintenant à peu près fonctionnelle.

1 Fonctionnalités de l'extension

1.1 La commande \systeme

Pour l'utiliser l'extension « systeme » , il faut écrire :

- `input systeme.tex` lorsqu'on utilise TeX ou pdfTeX ;
- `\usepackage{systeme}` dans le préambule lorsqu'on utilise LaTeX.

L'extension **xstring** est requise et est chargée si cela n'a pas été le cas.

La commande principale est `\systeme` dont l'argument obligatoire contient les équations séparées par une virgule :

La commande \systeme

```
Résoudre \systeme{2a-3b+4c=2,
a+8b+5c=8,
-a+2b+c=-5}
```

Résoudre $\begin{cases} 2a-3b+4c=2 \\ a+8b+5c=8 \\ -a+2b+c=-5 \end{cases}$

La virgule, qui est le séparateur par défaut, peut être changée en tout autre caractère. Il suffit de placer ce nouveau caractère dans l'argument de la commande `\syseqsep`.

La commande `\systeme` fonctionnera en mode math ou non et donnera un résultat correct si toutes les inconnues se trouvent dans le membre de gauche, le membre de droite étant celui des constantes. Les équations doivent être *développées*, c'est-à-dire que chaque terme est séparé de son voisin par un "+" ou un "-". De plus, les inconnues doivent être des lettres minuscules non accentuées, c'est-à-dire tout caractère de « a » à « z ».

L'alignement construit sera un tableau précédé d'une accolade et aura les spécificités suivantes :

- les signes d'égalité ou d'inégalité séparant les deux membres sont alignés ;
- les signes "+" ou "-" séparant chaque terme du membre gauche sont alignés ;
- chaque terme du membre de gauche se trouve dans une colonne au fer à droite ;
- le membre de droite se trouve dans une colonne au fer à gauche ;
- les espacements mathématiques entre colonnes seront corrects.

Les inconnues peuvent avoir un indice, sous réserve que celui-ci soit un nombre entier *positif ou nul* [2] :

Inconnues indicées

```
\systeme{4x_1-x_2=3,
-x_1+5x_2=-1}
```

$\begin{cases} 4x_1-x_2=3 \\ -x_1+5x_2=-1 \end{cases}$

Si une inconnue est manquante dans une équation, la colonne du tableau reste vide :

1. Enfin, tout le monde, c'est vite dit ! Cette extension n'est pas compatible avec ConTeXt car, pour une raison que je ne m'explique pas, **xstring** n'est pas utilisable avec ConTeXt. Si quelqu'un a une explication (et éventuellement un remède), je lui serais très reconnaissant de me contacter par **email** !
2. L'indice -1 correspond en interne à une inconnue non indicée. Par conséquent, la présence simultanée dans une même équation d'une inconnue non indicée et de cette inconnue avec l'indice -1 provoquera une erreur.

Inconnues manquantes

```
\systeme{a-2b=3,
b-3c=4,
-a+4c=-1}
```

$$\begin{cases} a-2b=3 \\ b-3c=4 \\ -a+4c=-1 \end{cases}$$

On peut également avoir une, plusieurs, ou toutes les équations sans second membre :

Équation sans second membre

```
\systeme{2a+3b-c=4,
b-2c,-a+2b+3c}
```

$$\begin{cases} 2a+3b-c=4 \\ b-2c \\ -a+2b+3c \end{cases}$$

Dans l'argument de la commande \systeme, lorsque deux virgules se suivent, une équation vide, c'est-à-dire une ligne vide est insérée. Malgré cette facilité, pour augmenter l'espacement vertical entre les équations, il vaut mieux utiliser la commande \syslineskipcoeff, voir page 5.

Ligne vide

```
\systeme{a-2b=3,,2a+5b=7}
```

$$\begin{cases} a-2b=3 \\ \\ 2a+5b=7 \end{cases}$$

La commande \systeme permet d'utiliser les commandes « \+ » et « \- » pour échapper « + » et « - », au cas où les coefficients doivent contenir ces signes, pour éviter qu'ils ne soient compris comme la fin d'un terme :

Coefficient contenant une addition ou soustraction

```
\systeme{(2\+\sqrt2)x-(1\-\sqrt2)y=1,
x+(1\+\sqrt2)y=-1}
```

$$\begin{cases} (2+\sqrt{2})x-(1-\sqrt{2})y=1 \\ x+(1+\sqrt{2})y=-1 \end{cases}$$

Lorsqu'elle est suivie d'une « * », la commande \systeme est dégradée et n'aligne que les signes d'égalité :

La commande \systeme*

```
\systeme{3x-4y+z=5,
5x-3z=1,
y+z=0}

\systeme*{3x-4y+z=5,
5x-3z=1,
y+z=0}
```

$$\begin{cases} 3x-4y+z=5 \\ 5x-3z=1 \\ y+z=0 \end{cases}$$

$$\begin{cases} 3x-4y+z=5 \\ 5x-3z=1 \\ y+z=0 \end{cases}$$

1.2 Tri des inconnues

Quel que soit l'ordre dans lequel sont entrées les inconnues lors de la saisie, elles seront triées par ordre alphabétique à l'affichage, en tenant compte de leur éventuel indice. Un signe "+" ne sera pas affiché lorsqu'il précède le premier terme d'une équation :

Tri des inconnues

```
\systeme{2y+x-3z=4,
z-y+2x=-1,
-2x+3z-4y=0}

\systeme{y_2+5y_1-x_1=0,
y_1-x_1+3y_2=4,
2x_1-y_2-y_1=-1}
```

$$\begin{cases} x+2y-3z=4 \\ 2x-y+z=-1 \\ -2x-4y+3z=0 \end{cases}$$

$$\begin{cases} -x_1+5y_1+y_2=0 \\ -x_1+y_1+3y_2=4 \\ 2x_1-y_1-y_2=-1 \end{cases}$$

Lorque les inconnues sont indicées, elles sont classés alphabétiquement puis dans l'ordre croissant de leur indice.

Le tri alphabétique est une facilité mais il peut s'avérer gênant surtout dans les systèmes 4×4 où, bien souvent, la 4e inconnue est « t » :

Tri alphabétique indésirable

```
\systeme{x+2y-3z+t=0,
2x-y-z+3t=4,
2y+3z+4t=-1,
3x-2z-2t=3}
```

$$\begin{cases} t+ & x+2y-3z=0 \\ 3t+2x- & y- & z=4 \\ 4t & +2y+3z=-1 \\ -2t+3x & -2z=3 \end{cases}$$

On aimerait bien que l'inconnue t soit en 4e position dans toutes les équations. Pour cela, il faut forcer un tri différent du tri alphabétique avec l'argument optionnel de la commande `\systeme`. Cet argument optionnel doit contenir la liste des inconnues, éventuellement indicées, sans aucun espace entre elles, et dans l'ordre où l'on souhaite les voir affichées dans chaque équation.

Ici, on affiche deux fois le même système avec deux ordres différents :

Tri forcé

```
\systeme[xyzt]{x+2y-3z+t=0,
2x-y-z+3t=4,
2y+3z+4t=-1,
3x-2z-2t=3}

\systeme[ztyx]{x+2y-3z+t=0,
2x-y-z+3t=4,
2y+3z+4t=-1,
3x-2z-2t=3}
```

$$\begin{cases} x+2y-3z+ & t=0 \\ 2x- & y- & z+3t=4 \\ & 2y+3z+4t=-1 \\ 3x & -2z-2t=3 \end{cases}$$

$$\begin{cases} -3z+ & t+2y+ & x=0 \\ -z+3t- & y+2x=4 \\ 3z+4t+2y & =-1 \\ -2z-2t & +3x=3 \end{cases}$$

Cet argument optionnel implique d'autres fonctionnalités, qui, avec le tri forcé vu juste au dessus, se cumulent toutes. Ainsi, lorsque cet argument optionnel existe et n'est pas vide :

1. les inconnues qui sont contenues dans l'argument optionnel peuvent être autre chose que des lettres minuscules (par exemple `A`, `B`, `\alpha`, etc.) éventuellement indicées :

   Autres inconnues

   ```
   \systeme[A\alpha\beta]{%
   A-\alpha+3\beta=4,
   2A+\alpha-\beta=0,
   -A+3\alpha-2\beta=5}
   ```

   $$\begin{cases} A- & \alpha+3\beta=4 \\ 2A+ & \alpha- & \beta=0 \\ -A+3\alpha-2\beta=5 \end{cases}$$

2. seules les inconnues figurant dans l'argument optionnel sont reconnues dans les équations. Dans l'exemple ci dessous, la lettre m n'est pas reconnue comme inconnue :

   Iconnues forcées

   ```
   \systeme[xy]{mx-y=3,
   x-m^2y=-1}
   ```

   $$\begin{cases} mx- & y=3 \\ x-m^2y=-1 \end{cases}$$

1.3 Inéquations

Dans chaque ligne, le signe susceptible de séparer les deux membres d'une équation est l'un de ceux-ci : `=`, `<`, `>`, `<=`, `>=`, `\leq`, `\geq`, `\leqslant` et `\geqslant`. Les deux derniers ne sont utilisables que si l'extension **amssymb** a été chargée.

Les signes `<=` et `>=` sont remplacés à l'affichage par `\leq`, `\geq`, qui donnent $\leq$ ou $\geq$.

Inéquations

```
\systeme{x+y-2z>4,
2x-y+z\geq-1,
3x-2y+z<=3}
```

$$\begin{cases} x+ \ y-2z>4 \\ 2x- \ y+ \ z\geq -1 \\ 3x-2y+ \ z\leq 3 \end{cases}$$

Pour choisir une autre substitution à « >= » ou « <= » ou pour en créer une pour tout autre signe d'égalité, on doit utiliser la commande :

```
\sysequivsign{signe}{substitution}
```

Voici le même système où l'on définit la substitution de « <= » avec « \leqslant » comme on l'observe à la troisième équation :

Redéfinir une substitution

```
\sysequivsign{<=}{\leqslant}
\systeme{x+y-2z>4,
2x-y+z\geq-1,
3x-2y+z<=3}
```

$$\begin{cases} x+ \ y-2z>4 \\ 2x- \ y+ \ z\geq -1 \\ 3x-2y+ \ z\leqslant 3 \end{cases}$$

1.4 Nouveaux signes d'égalité

Avec la commande \sysaddeqsign, on peut créer un nouveau signe susceptible de séparer les deux membres des équations. Il faut écrire :

```
\sysaddeqsign{<nouveau signe>}
```

Mettons ici que l'on créé le nouveau signe [3] d'égalité « ~ » en écrivant :

```
\sysaddeqsign{~}
```

Puis, mettons que l'on veuille ensuite remplacer ce nouveau signe par « \approx » dans l'affichage final. On devra écrire :

```
\sysequivsign{~}{\approx}
```

En voici l'illustration dans cet exemple :

Nouveau signe

```
\sysaddeqsign{~}
\sysequivsign{~}{\approx}
\systeme{2a+b-c~6,a-4b~4}
```

$$\begin{cases} 2a+ \ b-c\approx 6 \\ \ a-4b \quad \approx 4 \end{cases}$$

Par la suite, on peut supprimer ce signe ou n'importe quel autre déjà existant il faut utiliser la commande \sysremoveeqsign et écrire :

```
\sysremoveeqsign{~}
```

1.5 Coefficients décimaux

À première vue, la virgule étant utilisée pour séparer les équations, il n'est pas possible d'écrire des coefficients décimaux. On peut spécifier un autre caractère pour séparer les différentes équations avec le second argument optionnel de la commande \système. Ici, on prend « : » [4] ce qui permet d'écrire des coefficients décimaux. Le comportement de la virgule est redéfini à l'intérieur de la commande \systeme de telle sorte qu'elle ne soit pas suivie d'une espace, comme c'est le cas en mode mathématique.

Coefficients décimaux

```
\systeme[][:]{1,5x-0,45y=0,7:x-0,8y=1,4}
```

$$\begin{cases} 1{,}5x-0{,}45y=0{,}7 \\ \ x- \ 0{,}8y=1{,}4 \end{cases}$$

3. La création d'un nouveau signe est possible même si son code de catégorie est actif.

4. Ici encore, il est possible de choisir un caractère de code de catégorie actif, comme c'est le cas de « : » lorsque l'option « frenchb » est spécifiée au package babel.

Pour qui veut utiliser la virgule régulièrement dans les coefficients, on peut changer définitivement le séparateur par défaut pour n'importe quel autre caractère que l'on placera dans l'argument de la commande `\syseqsep` et écrire par exemple « `\syseqsep{:}` » pour choisir « : ».

1.6 Espacement des lignes

On peut faire varier l'espacement entre les lignes avec la commande `\syslineskipcoeff` dont l'argument est un nombre qui viendra multiplier la valeur de `\baselinskip`. Par défaut, l'argument vaut `1.25`.

Espacement variable

```
\systeme{x+2y-z=0,2x-y+z=1,x-3y+2z=1}
\syslineskipcoeff{1.75}\quad
\systeme{x+2y-z=0,2x-y+z=1,x-3y+2z=1}
```

$$\begin{cases} x+2y-z=0 \\ 2x-y+z=1 \\ x-3y+2z=1 \end{cases} \quad \begin{cases} x+2y-z=0 \\ 2x-y+z=1 \\ x-3y+2z=1 \end{cases}$$

1.7 Colonne supplémentaire

Lorsqu'une équation comporte le signe « `@` », tout ce qui se trouve à droite de ce caractère sera mis dans une colonne supplementaire au fer à gauche qui se trouvera à droite du système, en dernière position.

Colonne supplémentaire

```
\systeme{x+y=125@L_1,
x-y=12@L_2}
```

$$\begin{cases} x+y=125 & L_1 \\ x-y=12 & L_2 \end{cases}$$

Le signe « `@` » peut être changé en un autre avec la commande `\sysextracolsign` et on peut écrire par exemple :

```
\sysextracolsign{|}
```

Cette colonne supplémentaire n'est *pas* composée en mode mathématique, mais deux codes sont insérées au début et à la fin de cette colonne. Ils sont définis par :

```
\syscodeextracol{<code début>}{<code fin>}
```

Par défaut, on a `\syscodeextracol{\kern1.5em$}{$}` ce qui signifie qu'un espace horizontal d'1.5em est inséré et le `$` débute le mode mathématique au début de la colonne. Enfin, `$` termine le mode math à la fin.

On va redéfinir ces deux codes pour ne pas se mettre en mode math et mettre un espace de 2.5em :

Personnalisation de la colonne supplémentaire

```
\syscodeextracol{\kern2.5em }{}
\sysextracolsign{|}
\systeme{x+y=125|somme des deux nombres,
x-y=12|différence des deux nombres}
```

$$\begin{cases} x+y=125 & \text{somme des deux nombres} \\ x-y=12 & \text{différence des deux nombres} \end{cases}$$

1.8 Numérotation automatique

La colonne supplémentaire peut être utilisée pour y mettre une numérotation automatique. La première façon de procéder est d'indiquer le schéma que doit prendre cette numérotation automatique à la première ligne, sachant que le caractère « * » déclenche la lise en place de la numérotation automatique et sera remplacé par le numéro de la ligne dans le système. Les autres lignes, même si elles n'ont pas de colonne supplémentaire, auront un numéro :

Numérotation automatique

```
\systeme{x+y-z=3@L_{*},
2x+y+z=4,
x-y+2z=0}
```

$$\begin{cases} x+y-z=3 & L_1 \\ 2x+y+z=4 & L_2 \\ x-y+2z=0 & L_3 \end{cases}$$

Si on souhaite spécifier explicitement des colonnes supplémentaires aux lignes qui n'en ont pas, la numérotation automatique vient *avant* le contenu explicite des colonnes supplémentaires :

Numérotation automatique

```
\systeme{x+y-z=3@L_{*}\quad,
2x+y+z=4,
x-y+2z=0}

\systeme{x+y-z=3@L'_{*},
3x+2y=7@=L_1+L_2,
3x+y=6@=2L_1+L_3}
```

$$\begin{cases} x+y-z=3 & L_1 \\ 2x+y+z=4 & L_2 \\ x-y+2z=0 & L_3 \end{cases}$$

$$\begin{cases} x+y-z=3 & L'_1 \\ 3x+2y=7 & L'_2=L_1+L_2 \\ 3x+y=6 & L'_3=2L_1+L_3 \end{cases}$$

Il est ennuyeux que l'on ne puisse mettre « `L'_{*}=L_1` » à la première ligne car la totalité de ce schéma et notamment la partie indésirable « `L_1` » serait reproduit aux lignes suivantes. On peut donc spécifier le schéma de numérotation d'une deuxième façon, *avant* d'écrire le système. On utilise la commande `\sysautonum` dont l'argument contient ce schéma.

Numérotation automatique

```
\sysautonum{L_{*}}
\systeme{x+y-z=3,
2x+y+z=4,
x-y+2z=0}

\sysautonum{L'_{*}\longleftarrow}
\systeme{x+y-z=3@L_1,
3x+2y=7@L_1+L_2,
3x+y=6@2L_1+L_3}
```

$$\begin{cases} x+y-z=3 & L_1 \\ 2x+y+z=4 & L_2 \\ x-y+2z=0 & L_3 \end{cases}$$

$$\begin{cases} x+y-z=3 & L'_1 \longleftarrow L_1 \\ 3x+2y=7 & L'_2 \longleftarrow L_1+L_2 \\ 3x+y=6 & L'_3 \longleftarrow 2L_1+L_3 \end{cases}$$

Le schéma spécifié avec la commande `\sysautonum` n'est valable *que pour le prochain système* et sera effacé ensuite, sauf si on a écrit une étoile juste après la commande `\sysautonum`, auquel cas le schéma de numérotation se poursuit pour tous les systèmes à venir :

Numérotation automatique persistante

```
\sysautonum*{L_{*}}
\systeme{a+b=4,2a-b=5}
\quad
\systeme{x-3y=0,2x+y=1}
```

$$\begin{cases} a+b=4 & L_1 \\ 2a-b=5 & L_2 \end{cases} \quad \begin{cases} x-3y=0 & L_1 \\ 2x+y=1 & L_2 \end{cases}$$

On peut effacer le schéma de numérotation automatique en entrant un arguemnt vide : `\sysautonum{}`.

Il existe un compteur global[5] d'équations accessible de la même façon que le compteur local mais avec « ** ». Voici un exemple de numérotation persistante, où l'on voit que 84 équations ont été écrites jusqu'ici :

Numérotation globale

```
\sysautonum*{\hbox{eq }(**)}
\systeme{x-y+z=3,
2x+y+z=1,
x-z=8}

\systeme{u+w=9,
v+2w=0,
u-v=1}
```

$$\begin{cases} x-y+z=3 & \text{eq } (85) \\ 2x+y+z=1 & \text{eq } (86) \\ x-z=8 & \text{eq } (87) \end{cases}$$

$$\begin{cases} u+w=9 & \text{eq } (88) \\ v+2w=0 & \text{eq } (89) \\ u-v=1 & \text{eq } (90) \end{cases}$$

On peut à tout moment réinitialiser le compteur global d'équations à l'aide de « `\sysreseteqnum` ».

5. Il s'agit d'un compteur TEX portant le doux nom de `\SYSeqnum`, accessible via les commandes TEX habituelles, c'est-à-dire qu'on peut le faire précéder de `\number`, `\romannumeral` pour l'afficher.

1.9 Substitution post taitement

Juste avant l'affichage du système, il est encore possible de substituter dans le code du système tout caractère par un autre [6]. Pour cela, la commande `\syssubstitute` agit pour tous les systèmes à venir et son argument est fait de caractères allant par paires ; le premier étant le caractère à substituer et le second étant ce par quoi il le sera. Les subtitutions définies par `\syssubstitute` viennent *s'ajouter* à celles déjà définies.

Dans cet exemple, on entre des coefficients décimaux où le point est le séparateur décimal. On va demander à ce que tous les « `-.` » soient remplacés par des « `,` ». On remplacera aussi l'inconnue a_1 par x_n, et a_2 par x_{n+1}, ces 3 substitutions étant faites avec `\syssubstitute{.,{a_1}{x_n}{a_2}{x_{n+1}}}` :

Substitution post traitement

```
\syssubstitute{.,{a_1}{x_n}{a_2}{x_{n+1}}}
\systeme{1.5a_1-0.5a_2=2,1.6a_1-2a_2=0.4}
```

$$\begin{cases} 1{,}5x_n - 0{,}5x_{n+1} = 2 \\ 1{,}6x_n - \quad 2x_{n+1} = 0{,}4 \end{cases}$$

La commande `\sysnosubstitute` annule toutes les substitutions précédemment définies.

2 Liste des commandes

Voici la liste de toutes les commandes définies par cette extension :

Commandes	Description
`\systeme{<code système>}`	compose un système d'équations ou d'inéquations
`\syseqsep{<caractère>}`	définit le séparateur par défaut des équations
`\syslineskipcoeff{<coeff>}`	définit le coefficient multiplicateur de `\baselinskip` pour modifier l'espacement vertical des équations
`\+` et `\-`	remplace "+" et "-" lorsqu'on veut éviter d'indiquer un nouveau terme
`\sysequivsign{<signe>}{<substitution>}`	définit ce par quoi un `<signe>` d'égalité doit être remplacé à l'affichage
`\sysaddeqsign{<signe>}`	définit un nouveau signe d'égalité
`\sysremoveeqsign{<signe>}`	supprime un signe d'égalité
`\sysextracolsign{<signe>}`	définit le caractère délimitant la colonne supplémentaire
`\syscodeextracol{<code>}{<code>}`	définit les `<code>` qui seront inséré au début et à la fin de la colonne supplémentaire
`*`	déclenche la numérotation automatique dans la colonne supplémentaire et est remplacé par le numéro de la ligne du système
`**`	dans la colonne supplémentaire, est remplacé par le numéro global de l'équation
`\sysautonum{<code>}`	définit le schéma de la numérotation automatique
`\syssubstitute{<paires de caractères>}`	ajoute des substitutions à faire dans le code du système avant que celui-ci ne soit affiché
`\sysnosubstitute`	supprime toutes les substitutions jusqu'à présent définies

3 Algorithme

Voici les notations utilisées dans l'algorithme :

- les principales variables utilisées sont en `bleu` ;
- les ⟨`constantes`⟩ sont entre chevrons ;
- $\texttt{car}_{\texttt{n}}$(`variable`) est le caractère n de la `variable` ;
- `x` ⟵ `y` est une assignation qui signifie que `x` reçoit `y` ;
- `x` $=\!\!\oplus$ `y` est une concaténation qui signifie que la chaîne `y` est ajoutée à la fin de `x`.

6. Plus exactement, toute *suite de tokens* par une autre.

De la même façon, x $\oplus=$ y ajoute la chaîne y au début de x ;
- $\exists$ x signifie que la variable x existe ;
- gauche(var1,var2) est dans var1 ce qui se trouve à gauche de la première occurrence de var2. Même chose pour droite(var1,var2) sauf que c'est ce qui est à droite.

Dans les grandes lignes, voici l'algorithme qui est utilisé pour parcourir, analyser, découper, trier et reconstruire un système avec la commande

```
\systeme<*>[arg_opt#1][arg_opt#2]{argument obligatoire}
```

1. insérer un ⟨\begingroup⟩
2. ⟨\mathcode`\,⟩ ⟵ ⟨"013B⟩
3. si ⟨*⟩ présente
   étoile ⟵ ⟨vrai⟩
   sinon
   étoile ⟵ ⟨faux⟩
4. si ⟨arg_opt#1⟩ = ⟨vide⟩
   tri_auto ⟵ ⟨vrai⟩
   list_inconnues ⟵ ⟨vide⟩
   sinon
   tri_auto ⟵ ⟨faux⟩
   list_inconnues ⟵ ⟨arg_opt#1⟩
5. si ⟨arg_opt#2⟩ = ⟨vide⟩
   séparateur ⟵ ⟨séparateur par défaut⟩
   sinon
   séparateur ⟵ ⟨arg_opt#2⟩
6. numligne ⟵ ⟨1⟩
   arg_restant ⟵ ⟨argument obligatoire⟩
   extra_col ⟵ ⟨faux⟩
7. si séparateur ∈ arg_restant
   éq_actuelle ⟵ gauche(arg_restant,séparateur)
   arg_restant ⟵ droite(arg_restant,séparateur)
   sinon
   éq_actuelle ⟵ arg_restant
   arg_restant ⟵ ⟨vide⟩
8. si l'eq_actuelle contient le ⟨signe d'extra_col⟩ (qui est ⟨@⟩ par défaut)
   extra_col ⟵ ⟨vrai⟩
   excol[numligne] ⟵ droite(éq_actuelle,⟨signe d'extra_col⟩)
   éq_actuelle ⟵ gauche(éq_actuelle,⟨signe d'extra_col⟩)
9. si l'éq_actuelle contient un signe contenu dans la liste des signes d'égalité
   signe[numligne] ⟵ signe
   membre_G ⟵ gauche(éq_actuelle,signe)
   membre_D[numligne] ⟵ droite(éq_actuelle,signe)
   sinon
   membre_G ⟵ éq_actuelle
10. si car_1(membre_G) ∉ {⟨+⟩, ⟨-⟩}
    membre_G $\oplus=$ ⟨+⟩
11. signe_actuel ⟵ car_1(membre_G)
    membre_G ⟵ droite(membre_G,signe_actuel)
    (a) si membre_G contient ⟨+⟩ ou ⟨-⟩
        signe ⟵ première occurrence de ⟨+⟩ ou ⟨-⟩ dans membre_G
        terme_actuel ⟵ gauche(membre_G,signe)
        membre_G ⟵ droite(membre_G,signe)
      sinon
        terme_actuel ⟵ membre_G
        membre_G ⟵ ⟨vide⟩

(b) chercher alpha, la lettre représentant l'inconnue dans le terme_actuel

(c) signe[numligne,alpha] ⟵ signe_actuel

(d) terme[numligne,alpha] ⟵ terme_actuel

(e) si tri_auto = ⟨vrai⟩ et alpha ∉ list_inconnues
insérer alpha à sa place alphabétique dans list_inconnues

(f) si membre_G ≠ ⟨vide⟩
signe_actuel ⟵ signe
retourner en 9a

12. si arg_restant ≠ ⟨vide⟩
numligne ⟵ numligne + 1
aller en 6
13. nb_inconnues ⟵ nombre d'inconnues dans list_inconnues
14. nb_lignes ⟵ numligne
15. fabriquer le préambule du \halign :

(a) code_préambule ⟵ ⟨vide⟩

(b) si étoile = ⟨faux⟩
code_préambule =⊕ ⟨\hfil#&\hfil#&⟩ × (nb_inconnues − 1)

(c) code_préambule =⊕ ⟨\hfil#&#&#\hfil\null⟩

(d) si extra_col = ⟨vrai⟩
code_préambule =⊕ ⟨&#\hfil\null⟩

(e) code_préambule =⊕ ⟨\cr⟩

16. numligne ⟵ ⟨1⟩ code_système ⟵ ⟨vide⟩
17. numlettre ⟵ ⟨1⟩ premier_terme ⟵ ⟨vrai⟩
18. alpha ⟵ inconnue n° numlettre dans list_inconnues

(a) si ∃ signe[numligne,alpha]
si signe[numligne,alpha] = ⟨+⟩
si numlettre ≠ 1 et premier_terme = ⟨faux⟩
code_systeme =⊕ signe[numligne,alpha]
sinon
si numlettre = 1
terme[numligne,alpha] ⊕= signe[numligne,alpha]
sinon
code_systeme =⊕ signe[numligne,alpha]
premier_terme ⟵ ⟨faux⟩

(b) si numlettre ≠ 1 et si étoile = ⟨faux⟩
code_systeme =⊕ ⟨&⟩

(c) si ∃ terme[numligne,alpha]
code_systeme =⊕ terme[numligne,alpha]

(d) si étoile = ⟨faux⟩
code_systeme =⊕ ⟨&⟩

(e) si numlettre < nb_inconnues
numlettre ⟵ numlettre+⟨1⟩
aller en 16

(f) si ∃ signe[numligne]
code_systeme =⊕ signe[numligne]

(g) code_systeme =⊕ ⟨&⟩

(h) si ∃ membre_D[numligne]
code_systeme =⊕ membre_D[numligne]

(i) si extra_col = ⟨vrai⟩
code_systeme =⊕ ⟨&⟩

si $\exists$ `excol[numligne]`
`code_systeme` $=\oplus$ `excol[numligne]`

(j) si `numligne` $<$ `nb_lignes`
`code_systeme` $=\oplus \langle$`\cr`$\rangle$
`numligne` $\longleftarrow$ `numligne` $+ \langle$`1`$\rangle$
aller en 15

19. `code_systeme` $=\oplus \langle$`\cr`$\rangle$
20. si $\exists$ `liste des substitutions` et si `liste des substitutions` $\neq \langle$`vide`$\rangle$
effecteur les substitutions dans `code_systeme`
21. se mettre en mode mathématique s'il y a besoin et insérer
$\langle$`\left{\vcenter{\halign{code_préambulecode_systeme}}\right.`$\rangle$
22. insérer $\langle$`\endgroup`$\rangle$.

Pour finir et rendre les choses compréhensibles, voici un système où les frontières des 7 colonnes (3 inconnues et donc $2 \times 3 + 1 = 7$ colonnes) sont visibles :

$$\left\{\begin{array}{|r|r|r|l|}\hline -10x & +4y & -5z & =-1 \\ \hline x & & -7z & =3 \\ \hline 2x & -y & & =0 \\ \hline \end{array}\right.$$

$\star$
$\star \quad \star$

J'espère que cette extension vous sera utile et surtout que le code ne comporte pas trop de bugs… Un **email** pour me signaler tout dysfonctionnement, toute proposition d'amélioration ou même tout commentaire sur cette extension sera le bienvenu.

Christian Tellechea