Tableau de qualité pour la publication avec LaTeX*†

Simon Fear
University of Liverpool

Printed 31 octobre 2001

Résumé

Ce document décrit quelques commandes supplémentaires pour améliorer la qualité des tables en LaTeX. Des conseils sont donnés ici pour dire ce qui constitue une « bonne [1] » table dans ce contexte.

1 Introduction

Les commandes décrites ci-dessous doivent faciliter la production de tables telles qu'elles apparaissent (devraient apparaître) dans les livres et journaux scientifiques publiés. Ce qui distingue ces tables de celles que le LaTeX normal est capable de produire est un espacement au dessus et au dessous des filets et des filets d'épaisseur variable. Ce qui les distingue encore plus des tables que beaucoup de gens produisent *en fait* en utilisant LaTeX est l'absence de filets verticaux et de filets doubles.

Je dois faire une claire distinction entre ce que j'entends par une *table formelle*, qui est un ensemble de valeurs et de labels dans des colonnes, et ce que j'appelle un *tableau*, qui est le genre de choses présentées dans le manuel LaTeX, et qui est de plus en plus commun comme sortie des systèmes de gestion ou de bases de données, avec en plus des icônes en abondance, et probablement de la couleur par dessus le marché. La mise en page d'un *tableau* est déterminée (heureusement) de manière unique, étant donné un paquet de matériel que le concepteur essaie de combiner en une configuration significative. Mais la mise en page d'une *table* a été définie littéralement depuis des siècles d'expérience et ne devrait être altérée que dans des circonstances vraiment extraordinaires.

Pour illustrer mon propos, considérons ce tableau extrait du manuel LaTeX (page 64 de l'ancienne édition) :

---

*Ce fichier est la version v1.00, dernière révision du 6 novembre 1995.

†Traduit en français par Jean-Pierre Drucbert et Mathieu Goutelle le 2 mai 2001. Titre original « Publication quality tables in LaTeX ».

[1] NdT : selon des règles d'esthétiques particulières, que Simon C. Fear préfère.

gnats	gram	\$13.65
	each	.01
gnu	stuffed	92.50
emu		33.33
armadillo	frozen	8.99

C'est un fatras d'informations qui est probablement présenté raisonnablement clairement ainsi (mais l'émeu est-il farci ou non ?), Cependant, pour une table publiée, elle devrait certainement suivre les conseils donné plus bas sur cette page du manuel :

Item		
Animal	Description	Price (\$)
Gnat	per gram	13.65
	each	0.01
Gnu	stuffed	92.50
Emu	stuffed	33.33
Armadillo	frozen	8.99

Vous pouvez notez que ceci en fait demande moins de travail pour mettre ceci en page sous forme de table formelle ; nous n'avons plus à construire une nouvelle mise en page pour chaque table que nous faisons. Bien plus, nous pouvons être presque certains que les données ne pourront pas être mal lues (car le lecteur n'a pas à apprendre comment lire chacun parmi l'infinie variété des tableaux possibles).

La table ci-dessus ne peut pas être produite en LaTeX pur, malheureusement. Elle peut être mise en page comme elle devrait l'être, mais malgré tous nos efforts, l'utilisation de simples commandes `\hline` donne

Item		
Animal	Description	Price (\$)
Gnat	per gram	13.65
	each	0.01
Gnu	stuffed	92.50
Emu	stuffed	33.33
Armadillo	frozen	8.99

Notez (si ce n'est pas déjà évident) qu'il n'y a pas assez d'espace entre la ligne du haut et le I majuscule de « Item », et de même pour toutes les lignes : comparez avec la version précédente. De plus, les filets (c'est-à-dire les lignes) du haut et du bas dans la première version sont plus gras que le filet du milieu, qui à son tour est plus gras que le filet mineur en dessous de « Item ». Oui, je *sais* que vous pouvez redéfinir `\doublerulesep` et ensuite utiliser `\hline\hline` pour obtenir quelque chose donnant presque le même effet, et vous pouvez utiliser des « struts » pour améliorer l'espacement. Mais ce sont des astuces et vous ne devriez pas avoir à vous soucier de telles choses. Le paquetage `booktabs` définit ses propres commandes pour que ces problèmes soient pris en compte automatiquement.

En général, je voudrais dire que ce paquetage est sans intérêt pour ceux qui cherche une alternative à PicTeX pour traiter des tableaux sophistiqués. C'est plutôt un guide de style pour les auteurs d'articles et de livres scientifiques en ce qui concerne la mise en page des tables. Il n'est pas exagéré de dire que si vous ne pouvez pas créer une table en utilisant les commandes de ce paquetage, c'est que vous l'avez mal conçue.

1.1 Une note sur la terminologie

En typographie britannique, une « line » s'appelle toujours une « rule » (NdT : en typographie française, une « ligne » s'appelle toujours un « filet »). Peut-être en portant à confusion (pour des raisons historiques), l'« épaisseur » d'un filet est souvent appelée sa « largeur » (alors que n'importe qui d'autre l'appellerait sa « profondeur » ou « hauteur », s'il pensait à un filet horizontal). Une « ligne noire épaisse » est appelée un « filet gras » (« heavy rule »). Cette terminologie (britannique) est utilisée dans la plupart des nouvelles commandes décrites ci-dessous. Au moins cela évite la confusion avec `\hline`.

1.2 La mise en page des tables formelles

Vous ne ferez pas de graves erreurs si vous vous rappelez à tout moment de deux simples commandements :

1. Ne jamais, au grand jamais, utiliser de filets verticaux.
2. Ne jamais utiliser de filets doubles.

Ces commandements peuvent sembler extrêmes mais en des années d'expérience je n'ai jamais trouvé un bon argument pour passer outre. Par exemple, si vous sentez que les informations dans la moitié gauche d'une table sont si différentes de celles de la droite qu'il faut les séparer par une ligne verticale, alors vous devriez plutôt utiliser deux tables. Le second commandement est très, très occasionnellement violé : j'ai travaillé pour un éditeur qui insistait pour placer un filet fin double au dessus d'une rangée de totaux. Mais ce n'aurait pas été mon choix.

Il y a trois autres conseils que je pourrai citer ici car ils sont si peu connus en dehors des cercles des typographes et éditeurs professionnels :

1. Placez les unités dans l'en-tête de la colonne (pas dans le corps de la table).
2. Faites toujours précéder un point décimal (une virgule décimale en français) par un chiffre ; donc 0.1 (ou 0,1) et *pas* simplement .1 (,1).
3. N'utilisez pas de signes « ditto » ou toute convention analogue pour répéter une valeur précédente. Dans la plupart des cas, un blanc fait aussi bien l'affaire. Sinon, répétez la valeur.

Est-ce que c'est moi qui suis pédant ? Ces derniers conseils sont de plus en plus souvent ignorés dans les travaux publiés. Pour moi, ceci montre simplement que la typographie est celle d'un amateur.

De toute façon, que vous vouliez ou non suivre ces embellissements mineurs, si vous n'utilisez que les commandes suivantes dans vos tables formelles, votre lecteur sera reconnaissant. (Je répète que ces conseils ne sont pas seulement pour faire plaisir au pédant. Une structure de présentation améliorée commence par améliorer la pensée structurée).

2 Utilisation des nouvelles commandes

\toprule
\midrule
\bottomrule

Dans les cas les plus simples une table commence par une **\toprule**, a une rangée simple d'en-têtes de colonnes, puis un filet de séparation appelé ici **\midrule** ; après les colonnes de données nous terminons par une **\bottomrule**. La plupart des éditeurs de livres rendent les filets **\toprule** et **\bottomrule** plus gras (c'est-à-dire plus larges, ou plus sombres ; c'est une question de notation) que le filet intermédiaire **\midrule**. Cependant, lorsque les tables sont en très petits caractères, il est parfois impossible de faire cette distinction, et de plus quelques journaux utilisent des filets qui sont tous de même épaisseur. Les commandes de filet de ce paquetage ont toutes une épaisseur par défaut qui peut être modifiée à l'intérieur du document (de préférence, mais pas obligatoirement, dans le préambule). Pour les filets du haut et du bas, c'est **\heavyrulewidth** et pour les filets intermédiaires c'est **\lightrulewidth** (ces commandes sont décrites complètement plus loin). Dans de très rares cas, vous pouvez utiliser les arguments optionnels des commandes de filet qui ont la syntaxe formelle suivante :

> **\toprule[**⟨*largeur*⟩**]**
> **\midrule[**⟨*largeur*⟩**]**
> **\bottomrule[**⟨*largeur*⟩**]**

où ⟨*largeur*⟩ est une dimension TeX (par exemple 1pt, .4em, etc.).

Toutes les commandes de filets décrites ici se placent immédiatement après la commande \\ qui termine la rangée précédente (sauf bien sûr pour **\toprule**, qui se place juste après le début de l'environnement **tabular**) ; en d'autres termes, exactement là où le LaTeX standard permet **\hline** ou **\cline**.

\cmidrule

Bien sûr plus souvent que nous le voudrions nous avons besoin d'un filet qui ne s'étend que sur certaines des colonnes, filet pour lequel nos utiliserons **\cmidrule** (analogue à la commande **\cline** de LaTeX). En général, ce filet ne devrait pas recouvrir complètement les colonnes terminales, est ceci est en particulier le cas lorsque nous devons commencer une **\cmidrule** immédiatement après la fin d'une autre (les **\cline**'s de LaTeX se touchent si vous n'êtes pas extrêmement attentifs à **\extracolsep**). Donc vous voudrez en général utiliser les commandes optionnelles de raccourcissement (« trimming »), qui sont **(r)**, **(l)** et **(rl)** ou **(lr)**, qui indiquent si les extrémités droite et/ou gauche du filet doivent être rognées. Notez l'utilisation exceptionnelle de *parenthèses* au lieu d'accolades ou de crochets pour cette commande, dont la syntaxe complète est

> **\cmidrule[**⟨*largeur*⟩**](**⟨*rognage*⟩**){**a–b**}**

où ⟨*largeur*⟩ est encore une épaisseur optionnelle de filet (dont la valeur par défaut est ici **\cmidrulewidth**) et le dernier argument, qui n'est pas optionnel, donne les numéros des colonnes à englober.

Un exemple d'utilisation de ces commandes est donné par le code utilisé pour produire l'exemple de table ci-dessus :

```
\begin{tabular}{@{}llr@{}} \toprule
\multicolumn{2}{c}{Item} \\ \cmidrule(r){1-2}
Animal & Description & Price (\$)\\ \midrule
Gnat  & per gram  & 13.65 \\
```

```
    & each      & 0.01 \\
Gnu   & stuffed   & 92.50 \\
Emu   & stuffed   & 33.33 \\
Armadillo & frozen & 8.99 \\ \bottomrule
\end{tabular}
```

`\addlinespace` Occasionnellement nous désirons mettre un espacement supplémentaire entre certaines rangées d'une table ; par exemple, avant la dernière rangée, si c'est un total (un espacement est ici préférable à une autre `\midrule`, selon mon opinion). Il suffit d'insérer

`\addlinespace[`⟨*largeur*⟩`]`

après le marqueur d'alignement `\\`. Il n'est pas mauvais de voir `\addlinespace` comme étant un filet blanc d'épaisseur ⟨*largeur*⟩. L'espacement par défaut est `\defaultaddspace` qui donne bien moins qu'un interligne complet (comme ce que vous obtiendriez en utilisant `\\ \\` à la fin de la ligne ; ceci donne vraiment trop d'espace dans la plupart des cas).

3 Abus des nouvelles commandes

Il faut le reconnaître, cela ne marche pas tout seul, et il y a donc quelques conseils et des commandes supplémentaires pour les TEXperts et les touche-à-tout.

Les nouvelles commandes de filet ne sont pas garanties pour fonctionner avec `\hline` ou `\cline`, bien que celles-ci restent disponibles et inchangées. Je ne peut prévoir aucune raison pour vouloir les mélanger.

Plus sérieusement, les filets engendrés par les nouvelles commandes ne sont pas garantis pour se connecter aux filets verticaux engendrés par des caractères `|` dans le préambule de la table. Ceci est une particularité (voir plus haut). Vous ne devriez pas utiliser de filets verticaux dans les tables, point final.

`\morecmidrules` Si vous ne pouvez pas vous empêcher d'utiliser un filet double, même une construction aussi bizarre que `\toprule\bottomrule\midrule` fonctionnera sans provoquer de message d'erreur (tout comme vous pouviez utiliser une double `\hline`). Ces filets seront séparées de l'intervalle `\doublerulesep` du LATEX normal. Cependant si votre perversion va jusqu'à vouloir des `\cmidrule` doubles, vous aurez besoin de la commande supplémentaire `\morecmidrules` pour le faire correctement, car normalement deux commandes `\cmidrule` de suite est une construction parfaitement correcte demandant deux filets sur le même « filet de rangée ». Donc dans

```
\cmidrule{1-2}\cmidrule{1-2}
```

la seconde commande écrit un filet qui vient se placer exactement sur le premier ; je suppose que vous vouliez

```
\cmidrule{1-2}\morecmidrules\cmidrule{1-2}
```

qui donne un filet double entre les colonne une et deux, séparés de `\cmidrulesep` (note : puisque qu'une `\cmidrule` donne un filet très fin, la valeur ordinaire `\doublerulesep` donnerait probablement un espacement trop grand). Il faut terminer une rangée complète de filets avant de mettre la commande `\morecmidrules`. Notez que `\morecmidrules` n'a aucun effet si elle ne suit pas immédiatement une `\cmidrule` (ie n'est donc pas une commande générale d'espacement).

`\specialrule` Parler de ceci nous amène à l'abus de `\addlinespace` pour engendrer un espacement supplémentaire curieux entre des filets. Ne le faites pas (ce n'est cependant pas vraiment illégal). Utilisez plutôt

`\specialrule{`⟨*largeur*⟩`}{`⟨*espace au dessus*⟩`}{`⟨*espace au dessous*⟩`}`

où il faut noter que les trois arguments sont obligatoires (je ne me suis pas soucié d'établir des valeurs par défaut). Si vous utilisez ceci fréquemment, vous n'avez pas compris le but et le contenu des conseils donnés plus haut. Note technique : aucun espacement n'est ajouté après un filet précédent, mais un filet suivant engendrera un espacement `\doublerulesep` au dessus de lui. Pourquoi lisez-vous cette section ?

4 Sommaire technique des commandes

Les nouvelles commandes de filets sont valides à l'intérieur de l'environnement `tabular` (et en fait aussi à l'intérieur de l'environnement `array` et de ses dérivés), dans toutes les versions de LaTeX (vous devrez retirer la ligne

`\ProvidesPackage{booktabs}`

de `booktabs.sty` pour fonctionner sous LaTeX2.09 et les versions antérieures), et entièrement compatible avec l'environnement `array` du paquetage `array` fourni avec LaTeX 2_ε.

Les commandes suivent la syntaxe de placement standard de `\hline`. Si des commandes de filets sont doublées, il est plus prudent de vérifier qu'il n'y a pas d'espace entre les commandes. (Dans de nombreux cas, l'oubli de cette règle de sécurité donnera le message étrange « `misplaced noalign{` ».) N'utilisez pas de filets doubles !

Dans ce qui suit, un « filet » est l'une des commandes `\toprule`, `\midrule`, `\bottomrule`, `\cmidrule`, `\specialrule` ou `\addlinespace`. Notez en particulier la présence de la commande `\addlinespace` dans cette liste ; et l'exclusion de `\hline` et `\cline`, qui donneront des résultats imprévisibles si vous les mélanger avec les premières.

`\toprule[`⟨*largeur*⟩`]`

Un filet de largeur ⟨*largeur*⟩ (défaut `\heavyrulewidth`) avec `\belowrulesep` d'espacement vertical supplémentaire inséré en dessous de lui (sauf si il est suivi d'une autre commande de filet, auquel cas un espacement vertical `\doublerulesep` suit).

`\midrule[`⟨*largeur*⟩`]`

Un filet de largeur ⟨*largeur*⟩ (défaut `\lightrulewidth`) avec un espacement `\aboverulesep` au dessus de lui (sauf si précédé d'un autre filet, auquel cas il en sera séparé de la quantité `\doublerulesep`) et avec un espacement `\belowrulesep` en dessous de lui (à moins qu'un autre filet suive).

`\bottomrule[`⟨*largeur*⟩`]`

Un filet de largeur ⟨*largeur*⟩ (défaut `\heavyrulewidth`) avec un espacement `\aboverulesep` au dessus de lui (sauf s'il est précédé d'un autre filet, duquel il sera séparé de la quantité `\doublerulesep`) et avec un espacement `\belowrulesep` en dessous de lui (à moins qu'un autre filet le suive). L'espacement supplémentaire en dessous permet de laisser de l'espace pour des notes en bas de table.

`\cmidrule[`⟨*largeur*⟩`](`⟨*rognage*⟩`){`*a–b*`}`

Un filet de largeur ⟨*largeur*⟩ (défaut `\cmidrulewidth`) avec un espacement `\aboverulesep` au dessus de lui (sauf si il suit un autre `\cmidrule`, auquel cas il est sur le même alignement vertical ; ou si il suit tout autre filet, séparé de `\doublerulesep` ; ou si il suit `\morecmidrules`, séparé de `\cmidrulesep`), et avec l'espacement `\belowrulesep` en dessous de lui (à moins d'être suivi d'une autre `\cmidrule`, auquel cas le filet suivant est sur le même alignement vertical ; ou si suivi de `\morecmdirules`, avec l'espacement `\cmidrulesep` en dessous).

Le filet s'étend sur les colonnes a à b. L'argument optionnel ⟨*trim*⟩ qui se place entre parenthèses s'il est présent, peut être `r` pour rogner sur la droite, `l` pour rogner sur la gauche, ou les deux.

`\addlinespace[`⟨*largeur*⟩`]`

En fait considéré comme un filet de largeur nulle (donc invisible) sans espacement supplémentaire au dessus et un espacement ⟨*largeur*⟩ (qui est par défaut `\defaultaddspace`) au dessous (si un autre filet suit, celui-ci sera séparé en plus de `\doublerulesep`). En pratique, n'utilisez cette commande que pour ajouter de l'espace entre des rangées dans le corps de la table.

`\specialrule{`⟨*largeur*⟩`}{`⟨*espace au dessus*⟩`}{`⟨*espace au dessous*⟩`}`

Un filet de largeur ⟨*largeur*⟩ (notez l'argument obligatoire) avec un ⟨*espace au dessus*⟩ et un ⟨*espace au dessous*⟩ (sauf si un autre filet suit, auquel cas la séparation sera encore augmentée de `\doublerulesep`).

`\morecmidrules`

Avertit LaTeX qu'il faut commencer une nouvelle rangée de `\cmidrule`, séparée de la dernière par `\cmidrulesep`. N'a aucun effet en dehors de ce contexte.

Les dimensions par défaut sont

```
\heavyrulewidth=.08em
\lightrulewidth=0.5em
\cmidrulewidth=0.3em
\belowrulesep=.65ex
\aboverulesep=.4ex
\defaultaddspace=.5em
\cmidrulekern=.25em
```

La dernière d'entre elles, `\cmidrulekern`, est la quantité dont une `\cmidrule` est rognée à chaque bout indiqué dans les options `()`. Dans la construction

```
\cmidrule(r){1-2}\cmidrule(l){3-4}
```

il y a un total de 0.5 em séparant les deux filets. Actuellement le seul moyen pour obtenir des effets spéciaux est de modifier comme il convient la valeur de `\cmidrulekern` ; la valeur du raccourcissement n'est pas disponible sous forme d'argument dans le codage actuel de `\cmidrule`.

L'usager peut modifier ces valeurs par défaut au vol en insérant simplement une commande exactement sous le format ci-dessus ; la redéfinition restera effective pour le reste du document ou jusqu'à la prochaine redéfinition.

5 Support de `firsthline` et `lasthline`

Oui, d'une certaine manière, mais essentiellement non. Ceci ne s'applique pas en fait. Les commandes de ce paquetage ne sont pas faites pour faire de belles choses avec des boîtes ; celles-ci sont des tableaux et non des tables. Vous ne voudrez jamais, jamais, placer une table formelle au milieu d'une ligne de texte.

Cependant, si vous utilisez le paquetage `array`, les commandes ne sont pas altérées par le code présent (par exemple `\hline` et `\cline` restent actives).

6 Remerciements

Je[2] suis grandement redevable bien sûr à DEK et Lamport ; l'argument optionnel et le code de `\cmidrule` notamment est copié et adapté de `latex.sty`. La documentation est massivement tirée de la description du package `dcolumn.dtx` de David Carlisle.

Pour les tests et les encouragements...

7 Le code

La version actuelle est définie au début du fichier par quelquechose ressemblant à ça :

```
⟨*package⟩
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{booktabs}
  [\filedate\space version\fileversion]
```

D'abord nous définissons les nouvelles dimensions décrites plus haut :

```
\newdimen\heavyrulewidth
\newdimen\lightrulewidth
\newdimen\cmidrulewidth
\newdimen\belowrulesep
\newdimen\aboverulesep
\newdimen\cmidrulesep
\newdimen\cmidrulekern
\newdimen\defaultaddspace
\heavyrulewidth=.08em
```

[2]NdT : Simon Fear

```
\lightrulewidth=.05em
\cmidrulewidth=.03em
\belowrulesep=.65ex
\aboverulesep=.4ex
\cmidrulesep=\doublerulesep
\cmidrulekern=.25em
\defaultaddspace=.5em
```

et quelques compteurs internes, sans intérêt pour l'utilisateur :

```
\newcount\rulesflag
\newdimen\@cmidrulewidth
\newcount\@cmidla
\newcount\@cmidlb
\rulesflag=0
```

qui seront décrits plus bas si besoin.

7.1 Filets en pleine largeur

Nous plaçons le filet en pleine largeur dans un groupe `\noalign{}`, en utilisant une grosse astuce (`\ifnum=0`}`) pour faire croire à l'analyseur que le nombre d'accolades est correct. L'accolade sera réellement fermée après tout le traitement à la fin de la macro `\@endrule`.

\toprule

```
\def\toprule{\noalign{\ifnum0=`}\fi
 \@ifnextchar[{\@toprule}{\@toprule[\heavyrulewidth]}}
```

Cela tient compte de l'argument optionnel de `\toprule` : s'il y en a un, il est passé à `\@toprule`, sinon l'appel est fait avec la taille par défaut `\heavyrulewidth`.

Dans la suite, si `\rulesflag` a été défini (avec la valeur 1), nous venons juste de créer un précédent filet qui avait été modifié exceptionellement pour ne pas avoir un espacement normal après, donc nous devons placer `\doublerulesep` avant ce `\toprule` ; ensuite, nous réinitialisons `\rulesflag` à zéro. Note : nous ne pouvons pas juste ajouter toujours `\belowrulesep` après un `\toprule`, parce que il pourrait y avoir un `\doublerulesep` entre deux filets successifs. En revanche, nous pouvons interdire tout simplement les files doubles !

```
\def\@toprule[#1]{\ifnum\rulesflag=1\vskip
 \doublerulesep\global\rulesflag=0\fi
 \hrule \@height#1\futurelet\@tempa\@endrule}
```

À la troisième ligne au dessus, nous avons ajouter le filet et nous appelons la routine de fin `\@endrules` avec `\@tempa` égal au *token* suivant le filet dans le document.

```
\def\@endrule{\ifx\@tempa\toprule\global\rulesflag=1%
 \else\ifx\@tempa\midrule\global\rulesflag=1%
 \else\ifx\@tempa\bottomrule\global\rulesflag=1%
 \else\ifx\@tempa\cmidrule\global\rulesflag=1%
 \else\ifx\@tempa\specialrule\global\rulesflag=1%
 \else\ifx\@tempa\addlinespace\global\rulesflag=1%
 \else\vskip \belowrulesep\fi\fi\fi\fi\fi\fi\ifnum0=`{\fi}}
```

Ici, si la commande suivante est un autre filet ou un interligne (la honte sur l'utilisateur !), nous avons initialisé `\rulesflag` à 1 sans ajouter d'espace. Sinon, nous avons placé, après, l'espacement approrié.

`\midrule` `\bottomrule` Le code est presque le même que pour `\topriule`, sauf pour l'ajout de l'espace suivant le filet.

Notons qu'en ce qui concerne la programmation, un `\bottomrule` est simplement un `\midrule` large (mais l'utilisateur ne doit pas penser de cette façon.

```
\def\midrule{\noalign{\ifnum0=`}\fi
 \@ifnextchar[{\@midrule}{\@midrule[\lightrulewidth]}}
\def\@midrule[#1]{\ifnum\rulesflag=1\vskip
 \doublerulesep\global\rulesflag=0
 \else\vskip \aboverulesep\fi
 \hrule \@height#1\futurelet\@tempa\@endrule}
\def\bottomrule{\noalign{\ifnum0=`}\fi
 \@ifnextchar[{\@midrule}{\@midrule[\heavyrulewidth]}}
```

`\addlinespace` Un `\addlinespace` est traité comme une ligne de largeur nulle, sans espace avant et avec après l'espacement donné en argument (ou par défaut). Notez que la ligne suivante sera ajouté après un espace *supplémentaire* de `\doublerulesep`. L'utilisateur n'est pas encouragé à ajouter de l'espace avant/après les filets avec `\addlinespace`. Si besoin, il devra utiliser `\specialrule`.

```
\def\addlinespace{\noalign{\ifnum0=`}\fi
 \@ifnextchar[{\@addspace}{\@addspace[\defaultaddspace]}}
\def\@addspace[#1]{\ifnum\rulesflag=1\global\rulesflag=0\fi
 \vskip #1\futurelet\@tempa\@endrule}
```

`\specialrule` Ceci est inclus avec une certaine appréhension puisque cela permet à l'utilisateur de faire des choses stupides. Mais, un *designer* pourrait avoir besoin de ceci (ou d'une adaptation).

Notez qu'un `\toprule` suivant se comportera comme attendu (pas d'espace supplémentaire avant), mais un `\midrule` ou un `\bottomrule` ajoutera `\aboverulespace`, pendant que `\cmidrule` ajoutera `\doublerulesep`. Pourquoi voudriez vous faire ceci de toute façon ?

```
\def\specialrule#1#2#3{\noalign{
 \ifnum\rulesflag=1\global\rulesflag=0
 \else\vskip #2\fi\hrule \@height#1\vskip #3}}
```

7.2 Filets spéciaux

`\cmidrule` `\cmidrule` utilise `\rulesflag` d'une manière un peu différente. Il est mis (ou laissé) à 1 si vous êtes au milieu d'une ligne de `\cmidrule` ou si vous en commencez une nouvelle (avec `\morecmidrules`). Sinon, si `\rulesflag` vaut zéro, nous faisons précéder le filet par `\aboverulesep`.

```
\def\cmidrule{\noalign{\ifnum0=`}\fi
 \@ifnextchar[{\@cmidrule}{\@cmidrule[\cmidrulewidth]}}
\def\@cmidrule[#1]{\@ifnextchar({\@@cmidrule[#1]}{\@@cmidrule[#1]()}}
\def\@@cmidrule[#1](#2)#3{\@@@cmidrule[#3]{#1}{#2}}
```

Ce qui précède est du bidouillage pour initialiser les valeurs par défaut des arguments optionnels manquants. Nous passons également à `\@@@cmidrule` les arguments dans un ordre différent, c'est-à-dire `[a-b]{largeur demandée}` `{commandes de rognage}` (je ne parviens pas à me souvenir pourquoi j'ai fait ça).

```
\def\@@@cmidrule[#1-#2]#3#4{\global\@cmidla#1\relax
\global\advance\@cmidla\m@ne
\ifnum\@cmidla>0\global\let\@gtempa\@cmidrulea\else
\global\let\@gtempa\@cmidruleb\fi
\global\@cmidlb#2\relax
\global\advance\@cmidlb-\@cmidla
```

Cela a créé un un branchement conditionnel pour appeler la routine appropriée, `\@cmidrulea` ou `\@cmidruleb`, selon que nous commençons dans la première colonne ou non (quelle perte de temps !).

```
\global\@cmidrulewidth=#3
```

Il s'agit soit de la valeur par défaut ou de la valeur passée en argument.

Maintenant, nous analysons les arguments (le cas échéant) :

```
\global\let\cmlkern@l\z@ \global\let\cmlkern@r\z@
\@tfor\@tempa :=#4\do{\global\expandafter\let
 \csname cmlkern@\@tempa\endcsname\cmidrulekern}%
```

Maintenant, il faut insérer l'espace au dessus si besoin, fermer le `\noalign`, puis passer à la bonne routine de dessin du filet, définie plus haut (`\let\@gtempa`) :

```
\ifnum\rulesflag=0\vskip \aboverulesep\fi\ifnum0=`{\fi}\@gtempa
```

Maintenant, on ouvre un autre `\noalign` et on appelle la routine de fin.

```
\noalign{\ifnum0=`}\fi\futurelet\@tempa\@xcmidrule}
```

Ce code (appelé plus haut) dessine effectivement les filets :

```
\def\@cmidrulea{\multispan\@cmidla&\multispan\@cmidlb
\unskip\hskip \cmlkern@l\leaders\hrule \@height\@cmidrulewidth\hfill
\hskip \cmlkern@r\cr}
\def\@cmidruleb{\multispan\@cmidlb
\unskip\hskip \cmlkern@l\leaders\hrule \@height\@cmidrulewidth\hfill
\hskip \cmlkern@r\cr}
```

Pour finir, la routine de fin. Si un autre `\cmidrule` suit, on supprime l'espace vertical pour qu'il se chevauche et `\rulesflag=1` empêchera d'ajouter de l'espace au dessus du suivant. Si `\morecimdrules` suit, on ajoute `\cmidrulesep` (et on met encore `\rulesflag` à 1). Dans les autres cas, il s'agit du dernier filet du groupe actuel et nous devons juste ajouter l'espacement `\belowrulesep`.

Finalement, nous fermons le `\noalign`.

```
\def\@xcmidrule{\ifx\@tempa\cmidrule\vskip-\@cmidrulewidth
\global\rulesflag=1\else
\ifx\@tempa\morecmidrules\vskip \cmidrulesep
\global\rulesflag=1\else
\vskip \belowrulesep\global\rulesflag=0\fi\fi
\ifnum0=`{\fi}}
```

`\morecmidrules` Il s'agit vraiment d'une fausse commande ; tout le travail a été fait plus haut, dans la routine `\cmidrule`. Nous regardons un pas en avant pour voir si un `\morecmidrules` suit l'actuel `\cmidrule`, et le cas échéant, on active l'indicateur `\rulesflag`. Autrement, la commande `\morecmidrules` en elle même ne fait rien.

```
\def\morecmidrules{\noalign{\relax}}
⟨/package⟩
```