L'extension intexgral*

Valentin Dao†

Publiée le 26-07-2025

Résumé

Tandis que la composition d'intégrales est très fréquente en LaTeX, cette dernière se révèle souvent peu pratique. Lorsque l'expression devient complexe, il devient alors difficile d'en modifier les différents éléments dans un code source peu lisible. Pour répondre à ce problème, le package intexgral met à disposition une macro centrale dont le seul argument est l'intégrande. Tout le reste (les bornes, les différentielles...) pourra aisément être modifié grâce à une interface ⟨*clés = valeur*⟩. Contrairement à la méthode classique, où l'utilisateur choisit l'ordre des bornes avec les caractères actifs _ et ^, le package a dû fixer une convention. Ainsi, pour les deux clés gérant les bornes qui seront présentées, l'entrée supposée sera _⟨*borne inférieure*⟩^⟨*borne supérieure*⟩. Ce package étant écrit en expl3, il dépend donc des packages LaTeX3 l3kernel et l3package. De plus, intexgral utilise le package derivative afin de faciliter la manipulation de différentielles.

Table des matières

*Ce fichier décrit la version v2.0.0, mise à jour le 2025-09-09

†E-mail : vdao.texdev@gmail.com

Licence

© 2025 Valentin Dao, publié sous la LaTeX Project Public License (LPPL) 1.3c

Polices

Chronicle Text G3 Roman
© 2002, 2007 Hoefler & Frere-Jones.

Whitney-Medium
© 1996, 2009 Hoefler & Frere-Jones.

Dépôt

Voir dépôt GitHub.

Remerciements

Un grand merci à Plante et Slurpy de m'avoir accompagné dans cette aventure que fut l'apprentissage de TeX, vos conseils ont été précieux. Je tiens également à remercier Matty qui a gentiment révisé ce package et m'a donné des idées pour les versions à venir.

1 Options de package

Ces options peuvent être utilisées avec la syntaxe suivante lorsque le package est chargé dans le préambule :

```
\usepackage[⟨clés⟩]{intexgral}
```

`invert-limits` ⟨*boolean*⟩ (défaut: false)

Cette clé permet d'inverser la convention d'ordre des limites. L'entièreté du document ne peut suivre qu'une seule convention.

`invert-differentials` ⟨*boolean*⟩ (défaut: false)

Il est courant de voir les différentielles placées avant l'intégrande dans les articles de physique. Cette clé intervertit donc leurs positionnements.

`hide-differentials` ⟨*boolean*⟩ (défaut: false)

Si l'utilisateur souhaite que les différentielles ne soient pas affichées tout au long du document, cette clé désactive complètement la composition automatique de celles-ci.

`italic` ⟨*boolean*⟩ (défaut: false)
`upright` ⟨*boolean*⟩ (défaut: true)

Ce sont les deux clés du package derivative.[1]

2 Présentation de la macro principale

`\integral` `[⟨liste de clés⟩]{⟨intégrande⟩}`

Cette macro est celle qui permet de composer les intégrales. Elle doit naturellement être utilisée en mode mathématique. En voici un premier exemple :

```
\integral{f(x)}
```

$$\int f(x)\,\mathrm{d}x \tag{1}$$

Puisque cette macro est axée autour de l'utilisation de clés, la première partie de cette documentation les présentera en les regroupant par domaines. La seconde partie quant à elle documentera les macros auxiliaires qui viennent compléter les fonctionnalités de certaines clés.

[1] Disponible sur CTAN sous : https://ctan.org/pkg/derivative

3 Liste des clés

3.1 Bornes d'intégration

3.1.1 Définir les bornes

`limits` {⟨*pseudo-clist*⟩}, ⟨*mot-clé*⟩
`limits*` {⟨*pseudo-clist*⟩}, ⟨*mot-clé*⟩

Cette clé détermine les bornes d'intégration à utiliser en suivant la convention édictée dans le résumé.

```
\integral[
  limits={1, 10}
  ]{f(x)}
```

$$\int_{1}^{10} f(x)\,\mathrm{d}x \tag{2}$$

La variante étoilée de la clé exprime les bornes sous la forme d'un intervalle. Le sens des crochets s'adapte automatiquement à la présence de $\pm$`\infty`.

```
\integral[
  limits*={1, 10}
  ]{f(x)}
```

$$\int_{[1,10]} f(x)\,\mathrm{d}x \tag{3}$$

La grande force de `limits` est qu'elle permet de spécifier les bornes de plusieurs intégrales et se charge de les composer pour vous. Pour ce faire, il faut séparer les bornes de points-virgules pour permettre au package de correctement évaluer à quelle intégrale elles appartiennent.

```
\integral[
  limits={0, R; 0, H; 0, 2\pi},
  variable={\rho, z, \phi}
  ]{\rho z \phi}
```

$$\int_{0}^{R}\int_{0}^{H}\int_{0}^{2\pi} \rho z \phi\,\mathrm{d}\rho\,\mathrm{d}z\,\mathrm{d}\phi \tag{4}$$

3.1.2 Séparer l'intégrale

`int-split` ⟨*boolean*⟩ (défaut: `false`)

Cette clé sépare les différentes variables de l'intégrale et les isole. Il est donc nécessaire de l'utiliser *avant* `limits` afin que celle-ci sache dans quel mode opérer. De plus, cette clé requiert que l'on indique comment séparer l'intégrande. Cette étape supplémentaire se réalise en suivant la même logique que pour les bornes, c'est-à-dire en utilisant le point-virgule.

```
\integral[
  int-split=true,
  limits={0, R; 0, H; 0, 2\pi},
  variable={\rho, z, \phi}
  ]{\rho; z; \phi}
```

$$\int_0^R \rho \,\mathrm{d}\rho \int_0^H z \,\mathrm{d}z \int_0^{2\pi} \phi \,\mathrm{d}\phi \tag{5}$$

Pour bien fonctionner, il est évident que les clés `limits` et `variable`[2], ainsi que l'intégrande, doivent avoir la même dimension. Si ce n'est pas le cas, la compilation n'échouera pas, mais l'expression de l'intégrale risque de ne plus avoir aucun sens. De nombreux messages d'avertissement seront là pour vous en informer et vous dire ce qui ne va pas. N'oubliez donc pas les virgules et/ou points-virgules même si un élément reste vide !

3.1.3 Choisir le positionnement des bornes

`limits-mode` `limits, nolimits` (défaut: `nolimits`)

Équivalent à employer `\limits` ou `\nolimits`.

```
\integral[
  limits-mode=limits,
  limits={a, b},
  ]{f(x)}
```

$$\int_a^b f(x) \,\mathrm{d}x \tag{6}$$

3.2 Symbole (et encore des bornes)

Les clés `limits` et `limits*` se révèlent très utiles lorsque l'expression finale d'une intégrale est déterminée (autrement dit, lorsque les bornes de chaque variable ont été définies). Cependant, cela couvre seulement... l'expression finale ! Pour des cas plus généraux, elles sont

[2] à découvrir d'ici peu.

relativement malcommodes. Pour composer une intégrale double sur une surface S, on devrait écrire `limits={,;,S}`. Il va sans dire que c'est assez mal adapté pour l'utilisateur et peu optimal pour le package. Par ailleurs, le glyphe ne serait pas correct. L'ensemble des clés à suivre propose donc une façon de modifier facilement le symbole intégrale et les bornes.

3.2.1 Sélectionner le symbole

`int-symb` ⟨*séquence de contrôle*⟩

Cette clé accepte une macro désignant un symbole intégrale. Par exemple, `int-symb=\oint` revient à utiliser `\oint_{...}^{...}`. Par défaut, le package intexgral tente de charger tous les symboles d'unicode-math[3], étant donné qu'il en offre le plus grand nombre. Si un symbole n'est pas défini, un message d'avertissement sera émis et la commande manquante sera substituée à `\int`.

```
\integral[
  int-symb=\oint,
  lower-lim=\mathcal{C},
  diff-vec=true
  ]{f(\vec{r})}
```

$$\oint_{\mathcal{C}} f(\vec{r}) \cdot \mathrm{d}\vec{r} \tag{7}$$

3.2.2 Générer beaucoup intégrales

`nint` ⟨*entier*⟩

Cette clé accepte un entier n qui permet de composer n intégrales. Il est conseillé d'avoir recours à cette clé seulement si le nombre de symboles dépasse 4 afin de privilégier les glyphes définis par la police mathématique utilisée.

```
\integral[
  nint=5,
  lower-lim=\Omega,
  variable={x_1, x_2, x_3, x_4, x_5}
  ]{f(x_1, x_2, x_3, x_4, x_5)}
```

$$\idotsint_{\Omega} f(x_1, x_2, x_3, x_4, x_5)\, \mathrm{d}x_1\, \mathrm{d}x_2\, \mathrm{d}x_3\, \mathrm{d}x_4\, \mathrm{d}x_5 \tag{8}$$

[3] Voir la documentation pour la liste complète des symboles.

3.2.3 Définir les bornes (lorsqu'il n'y a qu'un seul symbole)

`lower-lim` {⟨*borne inférieure*⟩}
`upper-lim` {⟨*borne supérieure*⟩}

Ces deux clés permettent de spécifier les bornes inférieures et supérieures respectivement. Elles ne sont adaptées que si un seul symbole est affiché. Si vous avez besoin de spécifier les deux limites, la clé `limits` devra être privilégiée.

```
\integral[
  lower-lim={x^2 + y^2 \leq 1},
  variable={x, y}
  ]{f(x, y)}
```

$$\int_{x^2+y^2\leq 1} f(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{9}$$

Les clés précédentes (sans compter `nint`) couvrent un usage — je l'espère — assez général. Toutefois, on peut optimiser davantage pour un gain de temps supplémentaire. C'est pourquoi en plus de `int-symb`, `lower-lim` et `upper-lim`, intexgral propose des clés *raccourci* :

3.2.4 Combiner symbole et borne

`single`	`*{`⟨*borne*⟩`}`	`(symbole: \int)`
`double`	`*{`⟨*borne*⟩`}`	`(symbole: \iint)`
`triple`	`*{`⟨*borne*⟩`}`	`(symbole: \iiint)`
`quadruple`	`*{`⟨*borne*⟩`}`	`(symbole: \iiiint)`
`contour`	`*{`⟨*borne*⟩`}`	`(symbole: \oint)`
`surface`	`*{`⟨*borne*⟩`}`	`(symbole: \oiint)`
`volume`	`*{`⟨*borne*⟩`}`	`(symbole: \oiiint)`

Mis à jour: v1.1.0

Toutes ces clés permettent de regrouper l'action de `int-symb` et `lower-lim`/`upper-lim` ensemble, facilitant le choix du symbole et de la borne en même temps. La valeur de la clé est optionnelle, auquel cas le symbole changera mais la borne restera vide. Les clés sans astérisques modifient la borne inférieure, tandis que leurs variantes étoilées affectent la borne supérieure.

```
\integral[
  double=S,
  variable={x, y}
  ]{xy}
```

$$\iint_{S} xy\,\mathrm{d}x\,\mathrm{d}y \tag{10}$$

3.2.5 Motifs récurrents de bornes

Tout comme pour les symboles, les bornes suivent parfois des schémas courants qui peuvent se généraliser avec des clés, évitant de surcharger l'expression de `lower-lim`.

`domain` {⟨**-list*⟩}

Cette clé accepte une liste dont le délimiteur est un astérisque. Ensuite, chaque élément de la liste est analysée de la façon suivante :

- Le premier token de l'item se voit passer `\mathbb` (précédé d'`\uppercase` au cas où la touche SHIFT serait trop loin pour vos doigts).
- Les tokens restants — qui peuvent être vides — sont interprétés comme la puissance cartésienne de l'ensemble.

```
\integral[
  double,
  domain={r*r},
  variable={x, y}
  ]{xy}
```

$$\iint_{\mathbb{R}\times\mathbb{R}} xy \,\mathrm{d}x\,\mathrm{d}y \tag{11}$$

`boundary` {⟨*tokens*⟩}

Cette clé place simplement le symbol ∂ avant la borne inférieure.

```
\integral[
  contour,
  variable=r,
  diff-vec=true,
  boundary=S
  ]{G(\vec{r})}
```

$$\oint_{\partial S} G(\vec{r}) \cdot \mathrm{d}\vec{r} \tag{12}$$

3.3 Différentielles

3.3.1 Spécifier les variables

`variable` {⟨*clist*⟩}, ⟨*mot-clé*⟩, `none`

Cette clé permet de spécifier les paramètres d'intégration sous forme de liste d'éléments séparés par des virgules (clist). Si cette clé n'est pas renseignée, le package affichera par défaut $\mathrm{d}x$ (ou $\mathrm{d}\vec{r}$ si `diff-vec` est activé). La section 4.6 vous montre comment modifier ces choix. Si `none` est passé comme valeur, aucune différentielle ne sera affichée.

```
\integral[
  triple=V,
  variable={x, y, z}
  ]{f(x, y, z)}
```

$$\iiint_V f(x,y,z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \tag{13}$$

3.3.2 Inclure le jacobien

`jacobian` ⟨*boolean*⟩ `(défaut: false)`

Cette clé active/désactive l'affichage du jacobien lorsque défini par `\NewDifferentialKeyword` (voir section 4.4)

```
V_\textrm{sphère} =
\integral[
  int-split=true,
  limits={0, R; 0, 2\pi; 0, \pi},
  variable=spherical,
  jacobian=true
  ]{;;}
```

$$V_{\text{sphère}} = \int_0^R r^2\,\mathrm{d}r \int_0^{2\pi} \sin\theta\,\mathrm{d}\theta \int_0^{\pi} \mathrm{d}\phi \tag{14}$$

3.3.3 Modifier le symbole

`diff-symb` ⟨*séquence de contrôle*⟩

Comme évoqué plus tôt, le package derivative est utilisé pour la mise en forme des différentielles. Cette clé permet de changer le style des différentielles parmi ceux définis par `\NewDifferential`.

```
\NewDifferential{\feynmandiff}{\mathcal{D}}
\integral[
  variable={q(t)},
  diff-symb=\feynmandiff
  ]{e^{+\dfrac{\imath S[q(t)]}{\hbar}}}
```

$$\int e^{+\frac{\imath S[q(t)]}{\hbar}} \mathcal{D}q(t) \tag{15}$$

3.3.4 Opter pour la variante étoilée

`diff-star` ⟨*boolean*⟩ (défaut: false)

Si cette clé est vraie, cela revient à utiliser `\odif*` ou ses variantes.

```
\integral[
  double,
  variable={x, y},
  diff-star=true,
  ]{x^2 \exp(y)}
```

$$\iint x^2 \exp(y) \, \mathrm{d}_{x,y} \tag{16}$$

3.3.5 Désigner des options

`diff-options` {⟨*clés*⟩}

Cette clé accepte une liste de clés telle qu'elle aurait été écrite comme argument optionnel de `\odif` ou ses variantes. Pour illustrer, `diff-options={order={2, 3}, var=all}` sera interprété comme `\odif[order={2, 3}, var=all]{...}`.

```
\integral[
  diff-options={order=4}
  ]{\bar{\psi}(x)(\imath\gamma^\mu\partial_\mu-m)\psi(x)}
```

$$\int \bar{\psi}(x)(\imath\gamma^\mu\partial_\mu - m)\psi(x) \, \mathrm{d}^4x \tag{17}$$

3.3.6 Réaliser une intégrale curviligne

`diff-vec` ⟨*boolean*⟩

Cette clé applique `\vec` à chaque paramètre d'intégration et place un `\cdot` comme séparation.

```
\integral[
  double=S,
  variable=S,
  diff-vec=true
  ]{\vec{F}}
```

$$\iint_S \vec{F} \cdot \mathrm{d}\vec{S} \tag{18}$$

4 Macros auxiliaires

4.1 Options de package

`\intexgralsetup` {⟨*options de package*⟩}

Nouveau: v2.0.0

Cette macro offre un moyen alternatif de charger les options du package. Elle doit bien évidemment être utilisée dans le préambule seulement.

4.2 Charger des symboles

`\NewIntegralSymbol` {⟨*csname*⟩}

Si vous utilisez un package définissant des symboles ne faisant pas partie d'unicode-math, vous pouvez les charger en utilisant la macro `\NewIntegralSymbol` dans le préambule. Ils seront dès lors accessible avec la clé `int-symb`. L'argument devra prendre la forme d'une commande *sans antislash* : par exemple `\NewIntegralSymbol{myint}`. Veuillez noter que `\NewIntegralSymbol` n'est qu'un moyen pour le package de reconnaître les symboles préexistants et leurs macros de façon interne. Ce n'est en rien une manière de créer un nouveau glyphe. Il en va de la responsabilité de l'utilisateur de charger les packages nécessaires pour que les symboles voulus soient définis. L'ordre de chargement n'a cependant pas d'importance.

4.3 Retour sur la clé `limits`

`\NewLimitsKeyword` {⟨*mot-clé*⟩}{⟨*bornes*⟩}
`\RenewLimitsKeyword`
`\ProvideLimitsKeyword`
`\DeclareLimitsKeyword`

Il est courant de devoir renseigner les mêmes bornes dans une intégrale. Pour faciliter leur écriture, les clé(s) `limits(*)` accepte également un mot-clé désignant des bornes définies par l'une de ces quatre macros :

- `\NewLimitsKeyword` crée un nouveau mot-clé et émet une erreur s'il existe déjà.
- `\RenewLimitsKeyword` redéfinit un mot-clé et émet une erreur s'il n'existe pas déjà.
- `\ProvideLimitsKeyword` ne crée un nouveau mot-clé que s'il n'existe pas déjà. Aucun message ne sera émis dans le cas échéant.
- `\DeclareLimitsKeyword` crée un nouveau mot-clé quoi qu'il en soit, écrasant toute définition préalable.

En reprenant l'un des exemples précédents, on pourrait modifier l'expression de l'intégrale de la façon suivante :

```
\integral[
  limits={radius; height; circle},
  variable={\rho, z, \phi}
  ]{\rho z \phi}
```

$$\int_0^R \int_0^H \int_0^{2\pi} \rho z \phi \,\mathrm{d}\rho \,\mathrm{d}z \,\mathrm{d}\phi \tag{19}$$

Important : dû à l'implémentation de `\NewLimitsKeyword` et ses variantes, l'utilisateur *doit* suivre la convention d'ordre des bornes du package, même si `invert-limits` est fixé à `true`.

4.4 Retour sur la clé `variable`

`\NewDifferentialKeyword` {⟨*mot-clé*⟩}{⟨*différentielles*⟩}[⟨*jacobien*⟩]
`\RenewDifferentialKeyword`
`\ProvideDifferentialKeyword`
`\DeclareDifferentialKeyword`

De façon similaire, les mêmes groupes de différentielles réapparaissent souvent. On peut donc également définir des mots-clés dont la liste est déjà prédéfinie (les variantes ont le même comportement). La seule différence est que l'on peut en plus spécifier un jacobien. Ce dernier devra prendre la forme d'une clist. Ceci permet de pouvoir séparer le jacobien lorsque le mode `int-split` est activé. Son affichage est contrôlé par la clé `jacobian` expliquée précédemment.

Mot-clé	Bornes
ab	a, b
real	$-\infty, +\infty$
positive	$0, +\infty$
negative	$-\infty, 0$
unit	$0, 1$
circle	$0, 2\pi$
scircle	$0, \pi$
qcircle	$0, \frac{pi}{2}$
height	$0, H$
radius	$0, R$
length	$0, L$
time	$0, T$

TAB. 1 : Liste des mots-clés pour les bornes.

```
\integral[
  triple={V},
  variable=cartesian
  ]{f(x, y, z)}
```

$$\iiint_V f(x,y,z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \tag{20}$$

Mot-clé	Différentielles	Jacobien
cartesian	x, y, z	–
polar	r, θ	r
cylindrical	r, θ, z	r
spherical	r, θ, ϕ	$r, \sin\theta$

TAB. 2 : Liste des mots-clés pour les différentielles.

4.5 Emplacements personnalisés des différentielles

`\differentials` Bien que l'option `invert-differentials` existe, il est souvent souhaitable de pouvoir placer les différentielles où l'on veut. Par exemple, il est fréquent de les voir au numérateur d'une fraction lorsque celui-ci vaut 1. Pour répondre à ce besoin, le package met à disposition la macro `\differentials`. Lorsque `int-split=false`, `\differentials` compose toutes les différentielles au même endroit. Si `int-split=true`, il faudra réécrire `\differentials` entre chaque point-virgule. Dans les deux cas, le jacobien sera rattaché aux différentielles.

```
\integral[
  variable={a},
  ]{\frac{\differentials}{a}} = \ln(a) + C
```

$$\int \frac{\mathrm{d}a}{a} = \ln(a) + C \tag{21}$$

4.6 Configuration par défaut des différentielles

`\defaultdiff` {⟨*clist*⟩}, ⟨*token*⟩, ⟨*mot-clé*⟩ (défaut: x)
`\defaultvdiff` {⟨*clist*⟩}, ⟨*token*⟩, ⟨*mot-clé*⟩ (défaut: r)

Nouveau: v2.0.0

Comme évoqué avant, le package place automatiquement x (ou $\vec{r}$) comme paramètre d'intégration si aucune variable n'est donnée. Ceci peut être modifié et ainsi éviter de répéter la clé `variable` lorsqu'une grande partie de votre document utilise une même différentielle. À l'image de la clé `variable`, différents arguments sont possibles. L'assignation des deux macros est locale et elle peuvent donc être placées dans un groupe si besoin.

`\vdiffstyle` {⟨*séquence de contrôle*⟩} (défaut: \vec)

Nouveau: v2.0.0

La clé `diff-vec` utilise `\vec` afin d'appliquer un vecteur sur les différentielles. Il existe pourtant de nombreux styles de vecteurs différents en LaTeX. Ainsi, la macro ici présente utilise la séquence de contrôle donnée *sans argument* pour formater le vecteur. Par exemple, avec le package esvect chargé, on pourrait définir le style à utiliser de la façon suivante : `\vdiffstyle{\vv}`. L'assignation est elle aussi locale.

Historique des modifications

2.0.0 (2025-09-09)

Ajouté

- Macro `\intexgralsetup`.
- Macro `\defaultdiff`.
- Macro `\defaultvdiff`.
- Macro `\vdiffstyle`.

Modifié

- Messages d'avertissement liés aux symboles non-existants. Ils ne sont maintenant provoqués que lorsque l'intégrale est composée.

Supprimé

- Clé `diff-vec-style`.
- Message d'avertissement lié à la mauvaise utilisation des points-virgules en conjonction de la clé `int-split`.

Corrigé

- Bug où l'intégrande n'était pas réinitialisée lorsque `\integral` était employée successivement (détails ici).
- Des restes de log expl3 qui n'auraient pas dû être là.

1.1.0 (2025-07-29)

Ajouté

- Variantes étoilées pour les clés controlant symbole et borne (section 3.2.4).

1.0.0 (2025-07-26) — Version initiale.

Index