

LyXの数式詳細説明書

LyX プロジェクトチーム*

第 2.3.x 版

2023 年 5 月 5 日

*コメントや誤りの修正などがございましたら、LyX 文書化メーリングリスト
lyx-docs@lists.lyx.org までお知らせください。

目次

1. はじめに	1
2. 一般的な説明	1
3. 基礎的な関数	4
3.1. 指数および添字	4
3.2. 分数	4
3.3. 根号	5
3.4. 二項係数	6
3.5. 場合分け	6
3.6. 否定	7
3.7. 埋め草	7
3.8. 横線	8
3.9. 省略符号	8
4. 行列	10
5. 括弧と区分記号	11
5.1. 垂直括弧と区分記号	11
5.1.1. 手動の括弧高	11
5.1.2. 自動の括弧高	12
5.2. 水平括弧	13
6. 矢印	14
6.1. 水平矢印	15
6.2. 垂直矢印および対角矢印	16
7. アクセント	16
7.1. 一文字に付けるアクセント ¹	17
7.2. 複数の文字に付けるアクセント	18
8. 空白	18
8.1. 定義済みの空白	18
8.2. 可変長の空白 ²	19
8.3. 行内数式周りの空白	20
9. ボックスと枠	20
9.1. 枠付きボックス	20
9.2. 枠なしボックス	22
9.3. 色付きボックス	22
9.4. 段落ボックス	24
10. 演算子	25
10.1. 大演算子	25
10.2. 演算子の範囲	26

¹本文中のアクセントについては、第 16.2 節を参照。

²数式中の垂直方向の空白については、第 18.1.1 節をご覧ください。

10.3.	演算子の修飾	28
10.4.	二項演算子	29
10.5.	自己定義演算子	29
11.	書体	30
11.1.	書体様式	30
11.2.	ボールド体の数式	31
11.3.	色付きの数式	31
11.4.	書体寸法	32
12.	ギリシャ文字	33
12.1.	小文字	33
12.2.	大文字	33
12.3.	ボールド体	34
13.	記号³	34
13.1.	数学記号	34
13.2.	その他の記号	35
13.3.	ユーロ通貨記号	35
14.	関係子	35
15.	関数	36
15.1.	定義済み関数	36
15.2.	自己定義関数	37
15.3.	極限	37
15.4.	剰余関数	38
16.	特殊文字	38
16.1.	数式テキストにおける特殊文字	39
16.2.	文章中のアクセント	39
16.3.	古式数字	40
17.	数式様式	40
18.	多行数式	40
18.1.	概要	40
18.1.1.	行間	41
18.1.2.	列間	42
18.1.3.	長い数式	42
18.1.4.	多行にわたる分数	43
18.1.5.	多行にわたる括弧	43
18.2.	align 環境	44
18.2.1.	標準 align 環境	44
18.2.2.	alignat 環境	44
18.2.3.	flalign 環境	45
18.3.	eqnarray 環境	45
18.4.	gather 環境	45

³各 L^AT_EX パッケージに含まれる全記号をほとんど網羅した一覧が、[4] にあります。

18.5. multiline 環境	46
18.6. 数式の一部の多行化	46
18.7. 多行数式中のテキスト	47
19. 数式番号	48
19.1. 概要	48
19.2. 相互参照	48
19.3. 細目番号	48
19.4. ローマ数字や文字を使った付番	49
19.5. 自己定義番号	51
20. 化学記号と化学式	53
21. 図解	54
21.1. amscd 図解	54
21.2. xymatrix 図解	55
21.3. ファインマン図	55
22. 自己定義コマンド	55
22.1. \newcommand コマンド	55
22.2. 数式マクロ	56
23. コンピュータ代数システム	58
23.1. 使用法	59
23.2. 捷径	59
24. 補遺	59
24.1. 負の数	59
24.2. 位区切りとしてのコンマ	60
24.3. 物理ベクトル	60
24.4. 自己定義の分数	60
24.5. 数式の消去	61
24.6. 節見出し中の数式	62
24.6.1. 目次中では数式を使わない見出し	63
24.6.2. 目次中で数式を使う見出し $\sqrt{-1} = i$	63
24.7. 多段組文中の数式	63
24.8. 変数の説明付き数式	64
24.9. アップライト体のギリシャ小文字	64
24.10. 数式中のテキスト文字	65
24.11. 数式中の L ^A T _E X コメント	65
A. 組版上の助言	66
B. 同義語	67
参考文献	68
索引	69

1. はじめに

この文書は、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ の数式機能の説明書であると同時に、なによりも数式記号および数式要素に使用される $\text{I}^{\text{A}}\text{T}_{\text{E}}\text{X}$ コマンドのコレクションでもあります。説明は、コマンドの使用を念頭に置いています。したがって、ユーザーの手引きの数式の節をすでにお読みになっていることを前提としています。

この説明書で説明されている、ほとんどの数式記号と、数式要素の多くは、挿入▷数式メニューか数式ツールバーからアクセスすることが可能です。しかし、たくさんの数式を書かなくてはならない人はみな、数式ツールバーを使うよりもコマンドを使った方がずっと速いことに気付くことになるのです。したがって、この説明書はコマンドに焦点を当てますが、対応するツールバーボタンが利用可能なときには、それにも言及することになります。

とくに断らなければ、コマンドは数式内からのみ利用可能です。この文書で説明されているすべてのコマンドを利用できるようにするためには、文書設定 (文書▷設定▷数式オプションメニュー) で AMS math パッケージを使うオプションを有効にしなくてはなりません⁴。

説明を明瞭にするために、この文書はすべての $\mathcal{A}\mathcal{M}\mathcal{S}$ -math コマンド⁵を列挙はしません。

2. 一般的な説明

本文に埋め込まれた行内数式を作成するには、^{ショートカット} 捷 径 CommandM, ControlC M, ControlM M のうちのいずれか、あるいはツールバーボタン^{alt}を使用してください。

大きく別の段落として表示される別行立て数式を作成するには、ShiftCommandM, ControlM D のうちいずれかの捷徑を使用して下さい。

別行立て様式の数式を行内数式に変更するには、カーソルを数式内に合わせて CommandM, ControlC M, ControlM M のいずれかの捷徑か、編集▷数式▷数式の表記を変更メニューを使用して下さい。同じ方法が、行内数式を別行立て数式に変更するのにも使用できます。

行内数式の一部を別行立て数式の大きさで表示するには、`\displaystyle` を数式に入力して下さい。すると、青いボックスが新規に現れて、希望する数式の箇所を挿入することができます。

表の中では、行内数式のみが使用が許されています。

数式ツールバーは、表示▷ツールバーメニューで表示することができます。そのメニューで「数式」をクリックすると、ツールバーが下部に永続的に表示されます。この状態は、ツールバーメニューの中ではチェック印で表されます。この状態から、ツールバーメニューの「数式」をもう一度クリックすると、数式ツールバーは、カーソルが数式内部にあるときのみ表示されるようになります。この状態は、メニュー項目が「数式」から「数式 (自動)」に変わることによって表されます。

$\text{T}_{\text{E}}\text{X}$ モードは、ツールバーボタン^{tex}を押すか、挿入▷ $\text{T}_{\text{E}}\text{X}$ コード (捷徑: CommandL) メニューを使うことで、起動できます。

⁴AMS math パッケージを自動的に使うオプションは、 $\text{L}^{\text{A}}\text{X}$ でサポートされている数式要素が見つかったときのみ、 $\mathcal{A}\mathcal{M}\mathcal{S}$ -math パッケージを使用します。

⁵すべての $\mathcal{A}\mathcal{M}\mathcal{S}$ -math コマンドの一覧は、[amsguide.pdf](#) ファイルに収録されています。このファイルは、すべての $\text{I}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 標準頒布版に含まれています。

L^AT_EX プリアンブルを変更するには、文書▷設定▷LaTeX プリアンブルメニューを使用してください。

行列や場合分け、多行数式を続けて編集するには、編集▷数式メニューと編集▷行と列メニューを使うか、表ツールバーを使用することができます。メニューから行や列を交換するように指定されたときには、カーソルのある列や行は、それぞれ右側の列や下の行と交換されます。カーソルが最後の列や行にあるときには、左の列や上の行と交換されることになります。

数式内で文章を書く⁶には、**数式テキスト**が使用されます。このモードには、捷徑 CommandM を使うか、`\text` コマンドを挿入することで入ることができます。テキストは、L_YX 中では黒字で表示されるので、青字で表示される他の数式部分とは区別することができます。出力においては、数式テキストは、他の数式部分とは違って、アップライト体に組まれます。

コマンドの構成

数式要素に使われるほとんどの L^AT_EX コマンドは、以下のような構成になっています。

\コマンド名 [非必須引数]{必須引数}

コマンドは、つねにバックスラッシュ「\」で始まります。非必須の引数を省略するときには、随伴する括弧も省略しなくてはなりません。必須引数の前後の括弧は、この文書中では、T_EX 括弧と呼ぶことにします。数式中でコマンド名に左括弧を付けると、L_YX は自動的に T_EX 括弧を生成します。数式中ではそれ以外に、`\{` コマンドを使えば、つねに T_EX 括弧を生成することができます。L_YX 中で、青字で表示される通常の括弧とは違って、T_EX 括弧は赤字で表示されます。T_EX モード中では、T_EX 括弧を得るのに、とくにコマンドは必要としません。また、T_EX 括弧は出力中では表示されません。

記号のコマンドのように引数のないコマンドを T_EX モードに入力するときには、コマンドの終わりを表すために、コマンドの後に空白が**かならず**入力されなくてはなりません。この空白は出力中には現れません。空白を出力中に表示したいときには、空白の後に、通常テキストモードの保護された空白が来なくてはなりません。

保護された空白は、未定義^{キー} 鍵で入力できます。

⁶多行数式では、`\intertext` コマンドが使用されます。18.7 を参照のこと。

文法の説明

- 記号 `␣` は、空白文字を入力することを表します。
- `→` のような矢印は、キーボードから対応する矢印^{キー}を押すことを表します。



使用できる単位

表 1: 使用できる単位

単位	名称／摘要
mm	ミリメートル
cm	センチメートル
in	インチ
pt	ポイント (72.27 pt = 1 in)
pc	パイカ (1 pc = 12 pt)
sp	スケールポイント (65536 sp = 1 pt)
bp	ビッグポイント (72 bp = 1 in)
dd	ディドー (72 dd \approx 37.6 mm)
cc	シセロ (1 cc = 12 dd)
ex	現在のフォントの文字「x」の高さ
em	現在のフォントの文字「M」の幅
mu	数式単位 (1 mu = $1/18$ em)

3. 基礎的な関数



3.1. 指数および添字

添字は、アンダースコア「_」を打鍵するか、数式ツールバーボタンを使って入力することができ、指数は、キャレット「^」を打鍵するか、数式ツールバーボタンを使って入力することができます。

コマンド	出力
B_V	B_V
B^V	B^V
B^A	B^A

キャレットは、言語によってはアクセント記号として使用されているので、そのような場合には、母音字の後でキャレットを押すと、指数にならずにアクセントをつけることになってしまいます⁷。この場合に指数を作るには、上記の最後の例のように、キャレットの後に Space を押してください。

3.2. 分数

分数は、コマンド `\frac` か数式ツールバーボタンで作ることができます。フォント寸法は、分数が行内数式にあるか別行立て数式にあるかに応じて、自動的に調整されます。数式ツールバーボタンを使えば、分数の種類を選ぶことができます。

コマンド `\dfrac` を使えば、つねに別行立て数式の大きさを持つ分数を作成することができます。また、コマンド `\tfrac` では、つねに行内数式の大きさに分数が表示されます。以下はこれらの例です。

これは、コマンド `\frac` を使用して作った分数 $\frac{1}{2}$ を含む行です。

これは、コマンド `\dfrac` を使用して作った分数 $\frac{1}{2}$ を含む行です。

コマンド	出力
$\frac{A}{B}$	$\frac{A}{B}$
$\dfrac{A}{B}$	$\frac{A}{B}$
$\dfrac{e^{\frac{1}{2}}}{3}$	$\frac{e^{\frac{1}{2}}}{3}$

⁷使用しているキーボード設定によっては、同様のことが母音以外の文字でも起こることがあります。

入れ子の分数を作るには、コマンド`\cfrac`が使えます。以下がその例です。

`\frac` を使用して作成

$$\frac{A}{B + \frac{C + \frac{E}{F}}{D}}$$

`\cfrac` を使用して作成

$$\cfrac{A}{B + \cfrac{C + \cfrac{E}{F}}{D}}$$

上記の例で使用したコマンドは、

`\cfrac A\downarrow B+\cfrac C+\cfrac E\downarrow F\downarrow D`

です。

`\cfrac` は、他の分数中に入れ子になっている場合も含め、分数をつねに別行立て数式の大きさに設定します。

分子の揃え方は、指定することができます。`\cfracleft` コマンドは左揃えにし、`\cfracright` は右揃えにします。`\cfrac` は中央揃えです。以下の各分数は、それぞれの揃え位置を示しています。

$$\frac{A}{B+C}, \frac{A}{B+C}, \frac{A}{B+C}$$

【註】 `\cfracleft` と `\cfracright` は、生粋の \LaTeX コマンドではなく、実体は、コマンド `\cfrac[揃え位置]{分子}{分母}` です。したがって、これらを \TeX モードで使うことはできません。

ときに、以下のように `\cfrac` と `\frac` を組み合わせて使うと便利です。

$$\frac{A}{B + \cfrac{C + \frac{E}{F}}{D}}$$



斜めの分数線を持つ行内分数を作るには、コマンド `\nicefrac` (例: $\frac{5}{31}$) を使うか、コマンド `\unitfrac` (例: $\frac{5}{31}$) を使います。さらに、 $2\frac{1}{3}$ のような帯分数を作るコマンド `\unitfracthree` もあります。

【註】 実は、`\unitfracthree` は生粋の \LaTeX コマンドではなく、実体は `\unitfrac[自然数]{分子}{分母}` というコマンドなので、 \TeX コードでは使用できません。

分数線を変更できるような独自の分数の定義のしかたは、第 24.4 節に説明があります。

多行分数の説明は、第 18.1.4 節にあります。

3.3. 根号

平方根は、`\sqrt` か数式ツールバーボタン  で作成することができ、他のすべての根号は、コマンド `\root` か数式ツールバーボタン  で作成することができます。

コマンド	出力
<code>\sqrt{A-B}</code>	$\sqrt{A-B}$
<code>\root{3}\sqrt[3]{A-B}</code>	$\sqrt[3]{A-B}$


平方根は、根号指数フィールドを空白のままにしておけば、`\root` でも作成することができます。

$\sqrt[\beta]{B}$ の例のように、指数のとり値によっては、根号への距離が近すぎることがあります。この場合には、 β が根号に触れてしまいます。これを避けるためには、以下のようなコマンド書式で、コマンド `\leftroot` と `\uproot` を使います。

`\leftroot{距離}` および `\uproot{距離}`

ここで「距離」は、指数を左あるいは上に動かす、Big Point(単位 bp; 72 bp = 1 インチ)での数値です。これらのコマンドは、指数に書き込みます。このようにして、コマンド `\root\leftroot{-1}\uproot{2}\beta\rightarrow B` は、正しく組版された数式 $\sqrt[\beta]{B}$ を生成します。



3.4. 二項係数

二項係数は、コマンド `\binom` か数式ツールバーボタン  の小メニューを使って挿入することができます。分数 (`\frac`) と同様に、`\binom` の他に、コマンド `\dbinom` および `\tbinom` があります。二項係数のまわりの括弧に、他の括弧を使うには、コマンド `\brack` と `\brack` があります。

コマンド	出力
<code>\binom{A}{B}</code>	$\binom{A}{B}$
<code>\dbinom{A}{B}</code>	$\dbinom{A}{B}$
<code>\tbinom{A}{B}</code>	$\tbinom{A}{B}$
<code>\brack{A}{B}</code>	$\brack{A}{B}$
<code>\brace{A}{B}</code>	$\brace{A}{B}$

3.5. 場合分け

コマンド	出力
<code>\cases{A \rightarrow B > 0}</code>	$\left\{ \begin{array}{l} A \quad B > 0 \end{array} \right.$
<code>\cases{CommandReturn}</code>	$\left\{ \begin{array}{ll} A & \text{for } x > 0 \\ B & \text{for } x = 0 \end{array} \right.$

`\cases` を挿入するか数式ツールバーボタン  を使用した後では、ショートカット 捷徑 `CommandReturn` か表ツールバーボタン  を使えば、新しい行を作ることができます。

コマンド `\cases` は、挿入 > 数式 > Cases 環境メニューで挿入することもできます。

3.6. 否定

`\not` を挿入することで、すべての文字を取り消し形で表示できます。文字はスラッシュを上書きされた形になります。


コマンド	出力
<code>\not=</code>	\neq
<code>\not\leq</code>	\nless
<code>\not\parallel</code>	\nparallel

最後の例が示すように、すべての否定形がきれいに出力されるわけではありません。このことから、否定形に専用のコマンドを持つものもあります (第 13.1 節および第 14 節を参照)。



3.7. 埋め草

たとえば同位体⁸を表示しようとする、次のような問題が起こります。

上付き文字と下付き文字を使用して作った指数： $^{19}_9\text{F}$
正しい指数： $^{19}_9\text{F}$


短い方の指数は、既定で、長い方の指数の一文字目の下ないし上に配置されてしまいます。これを避けるには、一文字ないし複数の空の文字を生成するコマンド `\phantom` や数式ツールバーボタン⁹  があります。 `\phantom` を挿入すると、二つの赤い矢印が重なった青枠が表示されます。矢印は、箱の中身の幅と高さの両方が、埋め草 (指定した文字と同じ大きさの余白を確保するために使われる空打ち文字) として適用されることを示しています。したがって、 `\phantom` の作る文字は、箱の中身の文字の大きさを持つ埋め草となります。

コマンド	出力
<code>^{19}_9\phantom_1\rightarrow_9\text{F}</code>	$^{19}_9\text{F}$
<code>^{235}_9\phantom_23\rightarrow_9\text{F}</code>	$^{235}_9\text{F}$
<code>\Lambda^t\phantom_{ii}\rightarrow t_{MMt}</code>	Λ_{MMt}^t

さらに、 `\vphantom` (ツールバーボタン ) および `\hphantom` (ツールバーボタン ) というコマンドもあります。 `\vphantom` は、枠内部の文字の最大高のみの空白を作り、幅は考慮しません。 `\hphantom` は、枠の内容の幅のみの空白を作ります。このことから、これらの枠は一本の赤矢印のみで表示されます。

たとえば、 `\vphantom_a\int` は、積分記号¹⁰が最大高の文字なので、積分記号の高さを持つ空白を作ります。実際の適用例については、第 18.1.5 節を参照してください。

⁸同位体と化学記号の組版に関しては、第 20 節に記述があります。

⁹ツールバーボタン  の小メニューに入っています。

¹⁰`\int` コマンドは、積分記号を生成します。第 10.1 節を参照してください。

埋め草は、メニュー挿入▷整形▷埋め草を使えば、以下のように本文中でも使用することができます。

これは本文です。
本文です。

3.8. 横線

コマンド	出力
<code>\overline{A+B}</code>	$\overline{A+B}$
<code>\underline{A+B}</code>	$\underline{A+B}$
<code>\overline{\underline{A+B}}</code>	$\overline{\underline{A+B}}$

上記最後の例では、先に`\overline`が来ようが`\underline`が来ようが、関係ありません。

二重下線を引くには、`\underline`を二回使います。

文字の上下6本の線まで引くことができます。


自己定義の線は、以下の書式を持つ`\rule` コマンドで作成することができます。

`\rule[垂直オフセット幅]{長さ}{厚み}`


オプションの「垂直オフセット幅」は、行を上方に(値が負であれば下方に)移動させます。値としては、第1表に掲げてある単位を用いることができます。以下に、

`\rule[-2ex]{3cm}{2pt}` および `\rule{2cm}{1pt}`

というコマンドを用いて作成したふたつの例を例示します。


この行には、 二本の線があります。

`\rule` は、メニュー挿入▷整形▷水平線を使っても、本文に挿入することができます。

これは一行の文章です。

3.9. 省略符号

省略符号には、いくつかの種類が使用できます¹¹。列挙のためには、ベースラインの点々(`\ldots`)を使用しますが、演算子の場合は、演算子と同じ高さの点々(`\cdots`)が必要です。`\dots` コマンドを使うと、 \LaTeX は次に来る文字がどのような種類の文字であるかによって、自動的にどの種類を使うかを選択します。

¹¹数式ツールバー中のボタンで表示されている小メニューです。

コマンド	出力
A_1, \dots, A_n	A_1, \dots, A_n
$A_1 + \dots + A_n$	$A_1 + \dots + A_n$
A_1, \ldots, A_n	A_1, \dots, A_n
$A_1 + \cdots + A_n$	$A_1 + \dots + A_n$
\vdots	\vdots
\ddots	\ddots
\iddots	\iddots
	$A_{11} \cdots A_{1m}$
いろいろな点々を使った 3×3 行列	$\begin{matrix} \vdots & \ddots & \vdots \\ A_{n1} & \cdots & A_{nm} \end{matrix}$

挿入▷省略符号メニューで挿入される省略符号は`\ldots`です。

`\iddots`を使うには、文書設定の数式オプションにある `mathdots` パッケージを (自動的に) 使うオプションのうちいずれかを有効にしなくてはなりません。

`mathdots` パッケージを使うオプションを使用すると、文書中のフォント様式や寸法が既定値でないときのあらゆるドットの表示が改善されます。

とくに行列には、複数列にわたることのできる省略符号があります。これは、以下の書式を持つ `\hdotsfor` コマンドで作ることができます。

`\hdotsfor[距離]{列数}`

ここで「列数」は、何列に広げるかを指定します。「距離」は、点々のあいだの距離を示す因子です。

以下の行列では、2行目の1つ目の枠に`\hdotsfor[2]{4}`を挿入して、`\dots` コマンドの2倍の点間距離を持つ省略符号を挿入しています。

$$\begin{pmatrix} A & B & C & D \\ \hdotsfor[2]{4} & & & \\ q & w & e & r \end{pmatrix}$$

省略符号を広げる対象となる行列フィールドは空白にしておく必要があることに注意して下さい。さもないと \LaTeX エラーが発生します。

さらに、`\dotfill` コマンドを使えば、行の残りを点々で埋めることもできます。このコマンドの働きは、`\hfill` と同様のものです。第8.2節をご参照下さい。

たとえば、`A\dotfill_B` コマンドは、


$A \dots\dots\dots B$

のようになります。点々を使う`\dotfill`の直線版として、`\hrulefill`

$A \hrulefill B$

があります。これらのコマンドを本文で使用するには、これらのコマンドは \TeX モードで挿入される必要があります。

4. 行列

行列は、数式ツールバーボタンの  か挿入▷数式▷行列メニューで挿入することができます。すると、行列の行数・列数・配置方法・装飾を尋ねられます。ここで垂直配置は、行内数式内の行列でのみ意味を持ちます。



最初の行列は「上」配置
$$\begin{matrix} A & D & G & J \\ B & E & H & K \\ C & F & I & L \end{matrix}$$
 で、二番目は「中央」配置
$$\begin{matrix} A & D & G & J \\ B & E & H & K \\ C & F & I & L \end{matrix}$$

三番目は「下」配置
$$\begin{matrix} A & D & G & J \\ B & E & H & K \\ C & F & I & L \end{matrix}$$
 です。

水平配置は、各列がどのように配置されるべきかを指定します。これは、各列に対応した文字を一つずつ入力することによって設定します。*l*は左寄せ、*c*は中央揃え、*r*は右寄せを意味します。たとえば、第1列が左寄せで第2列と第3列が中央揃え、第4列が右揃えの4×4行列を作成するには、水平配置のところに **lccr** と入力します。通常、行列では各列は中央揃えですから、各列の既定値は **c** です。

水平行列の例です。

10000 D G 10000 D G 10000 D G
 III: B 10000 H , **ccc**: B 10000 H , **rrr**: B 10000 H
 C F 10000 C F 10000 C F 10000

つづいて行や列を追加したり削除したりするには、数式ツールバーボタンの  や  などや編集▷行と列メニューを使用することができます。また、行は **CommandReturn** で作成することもできます。

装飾は、行列の前後に選択した様式の括弧を加えます。他にも括弧は、**\left** コマンドや **\right** コマンドで作成することもできます (ショートカット 捷 径 : Alt+M 括弧)。第 5.1.2 節を参照してください。あるいは、以下のコマンドを使うこともできます。

コマンド	出力	コマンド	出力
\bmatrix _{2 × 2 行列}	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	\vmatrix _{2 × 2 行列}	$\begin{vmatrix} 0 & -i \\ i & 0 \end{vmatrix}$
\Bmatrix _{2 × 2 行列}	$\left\{ \begin{matrix} 0 & -i \\ i & 0 \end{matrix} \right\}$	\Vmatrix _{2 × 2 行列}	$\left\ \begin{matrix} 0 & -i \\ i & 0 \end{matrix} \right\ $
\pmatrix _{2 × 2 行列}	$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$	\matrix _{2 × 2 行列}	$\begin{matrix} 0 & -i \\ i & 0 \end{matrix}$

たとえば **\vmatrix** などを入力すると、青枠が二つの垂直線のあいだに現れるので、そこに行列を挿入することができます。

じつは多行数式はすべて行列なので、行列の各列の間隔を変更するには、第 18.1.2 節に説明されている距離 **\arraycolsep** をここでも使用することができます。

行間隔を変更するには、`\arraystretch` コマンドを使用します。以下のようにして使用します。

`\renewcommand{\arraystretch}{伸長因子}`

`\renewcommand` コマンドは、伸長因子を定義済みの `\arraystretch` コマンドに割り当てます。たとえば行間隔を2倍にするには、因子として2を指定して下さい。すると、以降の行列すべてにこれが使用されるようになります。元の間隔に戻すには、`\arraystretch` に因子1を割り当てて下さい。

本文行中に行列を入れるには、`\smallmatrix` コマンドを使います。これを挿入すると、二つの点線に囲まれた青枠が現れます。この枠のなかに行列を入れることができます。

これは、本文行中の行列 $\begin{pmatrix} A & B \\ C & D \end{pmatrix}$ です。

5. 括弧と区分記号

5.1. 垂直括弧と区分記号

コマンド	出力	コマンド	出力
<code>(</code>	$($	<code>)</code>	$)$
<code>{</code>	$\{$	<code>}</code>	$\}$
<code>[</code>	$[$	<code>]</code>	$]$
<code>\langle</code>	\langle	<code>\rangle</code>	\rangle
<code>\lceil</code>	\lceil	<code>\rceil</code>	\rceil
<code>\lfloor</code>	\lfloor	<code>\rfloor</code>	\rfloor
<code>/</code>	$/$	<code>\</code>	\backslash
<code> </code>	$ $	<code>\ </code>	$\ $

【註】 \TeX モードでは、`\` コマンドはその場所に改行を入れてしまうので、バックスラッシュを入力するには `\textbackslash` を使わなくてはなりません。

上に列挙した文字すべてについて、以下の二小節で説明されているコマンドを使って、大きさを調整することができます。これらのコマンドを使用するにあたっては、`\langle` や `\rangle` コマンドを使用せずに `<` や `>` の文字を直接使用することができます。

5.1.1. 手動の括弧高

括弧の丈は、 \LaTeX コマンドの `\big`, `\Big`, `\bigg` および `\Bigg` を使って、手動で指定することができます。`\big` が最小の大きさであり、`\Bigg` が最大の括弧高になります。

これらのコマンドは、括弧の階層を強調するのに使われます。

すべての括弧が同じ大きさ： $((A+B)(A-B))^C$

こちらの方が良い： $\left((A+B)(A-B)\right)^C$

二つ目の数式では、`\Big((A+B)(A-B)\Big)^C` というコマンドが使われています。

以下は、すべての括弧高の羅列です。

`\Bigg(\exp\bigg<\Big[\big\{\ln(3x)\big\}^2\sin(x)\Big]^A\bigg>\Bigg)^{0,5}`


$$\left(\exp\left\langle\left[\{\ln(3x)\}^2\sin(x)\right]^A\right\rangle\right)^{0,5}$$

`\big` 型コマンドの他に、括弧と中身のあいだにもう少し空白を加える `\bigm` という派生型と、空白を追加しない `\bigl`-`\bigr` 派生型があります。 `\bigl` コマンドの最後の `l` は、左括弧であることを示し、右括弧の場合には、`l` の代わりに `r` を用います。左括弧と右括弧は、それぞれ括弧の開始と終了に用いられます。

以下の表は、これらの派生型の比較です。

コマンド	出力
<code>\Bigm(\bigm(\ln(3x)\bigm)^2\Bigm)</code>	$\left((\ln(3x))^2\right)$
<code>\Big(\big(\ln(3x)\big)^2\Big)</code>	$\left((\ln(3x))^2\right)$
<code>\Bigl(\bigl(\ln(3x)\bigr)^2\Bigr)</code>	$\left((\ln(3x))^2\right)$
<code>\bigl)\ln(3x)\bigr(</code>	$)\ln(3x)($

5.1.2. 自動の括弧高

可変の丈を持つ括弧は、`\left` コマンドおよび `\right` コマンド、あるいは数式ツールバーボタンの  で挿入することができます。 `\left` および `\right` の直後には、必要とする括弧を挿入しなくてはなりません。すると、括弧高は出力時に自動的に計算されます。

通常の括弧：`\ln(\frac{A}{C})` というコマンドは

$$\ln\left(\frac{A}{C}\right)$$

を生成します。

複数行の括弧：`\ln\left(\frac{A}{C}\right)` というコマンドは

$$\ln\left(\frac{A}{C}\right)$$

を生成します。

`\left` や `\right` の代わりに、^{ショートカット} 捷 徑 `Alt+M` 括弧を使うこともできます。これを使うと、 $\text{L}_\text{Y}\text{X}$ 中で即座に実際の括弧高を確認することができるという利点と、対応する右括弧も生成されるという利点があります。

すると、先ほどの例を作るコマンドは `\ln Alt+M (\frac{A}{C}` となります。

左括弧あるいは右括弧を省略するには、ドットを挿入します。たとえば、`\left.\frac{A}{B}\right\}` というコマンドは

$$\left.\frac{A}{B}\right\}$$

を生成します。`\left` コマンドおよび `\right` コマンドは、文書が再度読み込まれたときには、 $\mathrm{L}_\mathrm{T}_\mathrm{X}$ によって正しい丈の括弧に変換され、省略された括弧は、点線として表示されます。

著名な $\mathrm{L}_\mathrm{T}_\mathrm{E}_\mathrm{X}$ 頒布版は、すべて $\mathrm{L}_\mathrm{T}_\mathrm{E}_\mathrm{X}$ の拡張である $\mathrm{e}_\mathrm{T}_\mathrm{E}_\mathrm{X}$ を使用しているので、これらの頒布版では、すべての括弧および極限に対して `\middle` コマンドも使用することができます¹²。このコマンドでは、物理ベクトル

$$\left\langle \phi \left| J = \frac{3}{2}, M_J \right. \right\rangle$$

で必要とされるように、次に続く文字の高さは、囲まれる括弧の高さに調節されます。物理ベクトルに関しては、第 24.3 節に説明されているように特殊な $\mathrm{L}_\mathrm{T}_\mathrm{E}_\mathrm{X}$ パッケージがあります。

5.2. 水平括弧

コマンド	出力
<code>\overbrace{A+B}_3</code>	$\overbrace{A+B}^3$
<code>\underbrace{A+B}_5</code>	$\underbrace{A+B}_5$
<code>\overbrace{\underbrace{A+B_w}_7}_C</code>	$\overbrace{\underbrace{A+B_w}_7}^C$

最後の例では、`\overbrace` が先に挿入されようが `\underbrace` が先に挿入されようが代わりはありません。

文書設定の数式オプションで、`mathtools` パッケージに常に読み込みオプションを設定すると、以下のような角括弧が利用可能になります。

コマンド	出力
<code>\overbracket{A+B}_3</code>	$\overbracket{A+B}^3$
<code>\underbracket{A+B}_5</code>	$\underbracket{A+B}_5$
<code>\overbracket{\underbracket{A+B_w}_7}_C</code>	$\overbracket{\underbracket{A+B_w}_7}^C$

¹² 【訳註】 $\mathrm{p}_\mathrm{L}_\mathrm{T}_\mathrm{E}_\mathrm{X}$ では、標準では `\middle` コマンドは使えません。よって、以下の例では「`\middle|`」の代わりに「`\biggm|`」を用いています。

`\overbracket` と `\underbracket` に関しては、コマンドに続く角括弧中に、希望する厚みを指定すれば、以下のように、角括弧の筆跡の厚みに変更を加えることができます。

コマンド	出力
<code>\overbracket[3pt]{A+B}_3</code>	$\overbrace{A+B}^3$
<code>\underbracket[1pt]{A+B}_5</code>	$\underbrace{A+B}_5$

括弧をお互いに重ねる必要がある場合には、第 18 節に説明されているように、次のような多行数式を使わなくてはなりません。

$$A = \underbrace{\underbrace{gggg+bbqq}_r + dddd}_s$$


一行目には、数式が一つめの括弧とともに挿入されています。ここで、空白コマンド¹³`\:`を最初の d の前に挿入しておくことが重要です。さもないと、 q の後ろで終わる括弧のせいで、直後の「+」の周りに正しく空白が入ることが妨げられてしまう¹⁴ためです。二行目には、二つめの括弧が挿入されています。 b の直前から始まるようにするために、まず `\hphantom{gggg+}` というコマンド¹⁵が挿入されています。この数式中の「+」も空白で囲まれるようにするために、この空白コマンドが必要になっています。二つめの括弧は、`\hphantom{bbqq+dddd}` コマンドの下に置きます。

以下の例のように、括弧が反対側に重なる場合には、もっと複雑になります。

$$A = \underbrace{gggg + \overbrace{bbqq}^s + dddd}_r$$

最初の数式行は、括弧が上に来ていること以外は、先の例の第二行と同じです。二行目には、二つめの括弧と一緒に数式が入っています。一行目の括弧と数式のあいだに余白が入ることを防ぐために、行間を減らさなくてはならないのですが、これは $\text{L}_\text{Y}\text{X}$ のバグ¹⁶のせいで簡単にはできません。この問題を回避するためには、数式直前に $\text{T}_\text{E}\text{X}$ モードで `setlength{\jot}{-6pt}` というコマンドを入れて、大域的な数式行間 `\jot` を -6pt に変更しなくてはなりません。`\jot` は、数式直後に同様のコマンドを使って標準値の 3pt に戻します。数式中の行間について、詳しくは第 18.1.1 章に説明があります。

6. 矢印

矢印は、数式ツールバーボタンのか、以下の各小節に列挙してあるコマンドで挿入することができます。

¹³空白コマンドは第 8.1 章に説明があります。

¹⁴これは、括弧が文字として取り扱われないためです。第 10.4 章参照。

¹⁵`\hphantom` に関する詳細は、第 3.7 章を参照してください。

¹⁶[LyX-bug #1505](#)

6.1. 水平矢印

コマンド	出力	コマンド	出力
<code>\gets</code>	\leftarrow	<code>\to</code>	\rightarrow
<code>\Leftarrow</code>	\Leftarrow	<code>\Rightarrow</code>	\Rightarrow
<code>\longleftarrow</code>	\longleftarrow	<code>\longrightarrow</code>	\longrightarrow
<code>\Longleftarrow</code>	\Longleftarrow	<code>\Longrightarrow</code>	\Longrightarrow
<code>\leftharpoonup</code>	\leftharpoonup	<code>\rightharpoonup</code>	\rightharpoonup
<code>\leftharpoondown</code>	\leftharpoondown	<code>\rightharpoondown</code>	\rightharpoondown
<code>\hookleftarrow</code>	\hookleftarrow	<code>\hookrightarrow</code>	\hookrightarrow

コマンド	出力	コマンド	出力
<code>\leftrightarrow</code>	\leftrightarrow	<code>\mapsto</code>	\mapsto
<code>\Leftrightarrow</code>	\Leftrightarrow	<code>\longmapsto</code>	\longmapsto
<code>\longleftrightarrow</code>	\longleftrightarrow	<code>\leadsto</code>	\leadsto
<code>\Longleftrightarrow</code>	\Longleftrightarrow	<code>\dashrightarrow</code>	\dashrightarrow
<code>\rightleftarrows</code>	\rightleftarrows		

たとえばベクトル記号の矢印のようにアクセントとして使用される矢印は，第7節に一覧があります。

さらに，ラベル付き矢印として，`\xleftarrow` と `\xrightarrow` があります。これらのコマンドを数式に挿入すると，二つの青枠のついた矢印が現れるので，そこにラベルを入れることができます。矢印の長さは，ラベルの幅に応じて調整されます。

コマンド	出力
$F(a) \xleftarrow{x=a \downarrow x>0} F(x)$	$F(a) \xleftarrow{x=a \downarrow x>0} F(x)$
$F(x) \xrightarrow{x=a \downarrow x>0} F(a)$	$F(x) \xrightarrow{x=a \downarrow x>0} F(a)$

文書設定の数式オプションで，**mathtools** パッケージに常に読み込みオプションを設定すると，以下のようなラベル付き矢印が利用可能になります。

コマンド	Example	コマンド	Example
$\backslash xleftarrow$	$\xrightarrow{x>0}{x=a}$	$\backslash xleftarrow$	$\xrightarrow{x>0}{x=a}$
$\backslash xLeftarrow$	$\xrightarrow{x>0}{x=a}$	$\backslash xleftarrow$	$\xrightarrow{x>0}{x=a}$
$\backslash xrightarrow$	$\xrightarrow{x>0}{x=a}$	$\backslash xrightarrow$	$\xrightarrow{x>0}{x=a}$
$\backslash xleftrightarrow$	$\xleftrightarrow{x>0}{x=a}$	$\backslash xrightarrow$	$\xrightarrow{x>0}{x=a}$
$\backslash xhookrightarrow$	$\xrightarrow{x>0}{x=a}$	$\backslash xleftrightarrow$	$\xleftrightarrow{x>0}{x=a}$
$\backslash xhookleftarrow$	$\xrightarrow{x>0}{x=a}$	$\backslash xleftrightarrow$	$\xleftrightarrow{x>0}{x=a}$

これらの矢印は、すべて以下のようなコマンドスキームを取ります。

コマンド	出力
$F(a)\backslash xleftarrow[x=a]{\{x>0\}}F(x)$	$F(a)\xrightarrow[x=a]{x>0}F(x)$

6.2. 垂直矢印および対角矢印

コマンド	出力	コマンド	出力
$\backslash uparrow$	\uparrow	$\backslash nearrow$	\nearrow
$\backslash Uparrow$	\Uparrow	$\backslash searrow$	\searrow
$\backslash updownarrow$	\updownarrow	$\backslash swarrow$	\swarrow
$\backslash Updownarrow$	\Updownarrow	$\backslash nwarrow$	\nwarrow
$\backslash Downarrow$	\Downarrow		
$\backslash downarrow$	\downarrow		

垂直矢印は、第 5.1.1 節および第 5.1.2 節に述べられているコマンドを使うと、区分記号として使用することもできます。

7. アクセント

アクセントは、数式ツールバーボタンの  か、以下の各小節に列挙してあるコマンドで入力することができます。

7.1. 一文字に付けるアクセント¹⁷

コマンド	出力	コマンド	出力
<code>\dot_A</code>	\dot{A}	<code>\tilde_A</code>	\tilde{A}
<code>\ddot_A</code>	\ddot{A}	<code>\hat_A</code>	\hat{A}
<code>\dddot_A</code>	\dddot{A}	<code>\check_A</code>	\check{A}
<code>\ddddot_A</code>	\ddddot{A}	<code>\acute_A</code>	\acute{A}
<code>\vec_A</code>	\vec{A}	<code>\grave_A</code>	\grave{A}
<code>\bar_A</code>	\bar{A}	<code>\breve_A</code>	\breve{A}
<code>\mathring_A</code>	\mathring{A}		

éのようなアクセントは、数式に直接入れることができます。L^AT_EX は、それを対応するアクセントコマンドに変換します。ウムラウトに関しては、母音の前に引用符を挿入する方法の方がよいでしょう。ウムラウトのある数式部分がドイツ語に指定してあれば、L^AT_EX は、引用符と母音をまとめて一つの文字として取り扱います。`\ddot`と違い、この方法では、以下の例に示すように「本物の」ウムラウトが作られます。

コマンド	出力
<code>"i</code>	\ddot{i}
<code>\ddot_i</code>	\ddot{i}

`\ddot` に比べて良いもう一つの利点は、上記のアクセントコマンドが**数式中テキストでは使用できない**のに対し、ウムラウトは直接数式中テキストに変換されることです。(アクセントコマンドによる)アクセント付き文字を数式中テキストに変換すると、アクセントの下にある文字しか変換されません。これは、たとえばイタリック体やボールド体への変換など、他のすべての変換に関しても言えることです。

ウムラウトと他のアクセント付き文字は、数式中テキストに直接入れることができます。

L^AT_EX は、L^AT_EX パッケージ **undertilde** がシステム上に導入されていれば、文字下のティルダもサポートします。

この小節のすべてを出力で見するには、**undertilde** L^AT_EX パッケージを導入する必要があります。

¹⁷本文中のアクセントについては、第 16.2 節を参照。

7.2. 複数の文字に付けるアクセント

コマンド	出力	コマンド	出力
<code>\overleftarrow{A=B}</code>	$\overleftarrow{A=B}$	<code>\overrightarrow{A=B}</code>	$\overrightarrow{A=B}$
<code>\underleftarrow{A=B}</code>	$\underleftarrow{A=B}$	<code>\underrightarrow{A=B}</code>	$\underrightarrow{A=B}$
<code>\overleftrightharrow{A=B}</code>	$\overleftrightharrow{A=B}$	<code>\widetilde{A=B}</code>	$\widetilde{A=B}$
<code>\underleftrightharrow{A=B}</code>	$\underleftrightharrow{A=B}$	<code>\widehat{A=B}</code>	$\widehat{A=B}$

これらのコマンドでは、好きなだけ多くの文字にアクセントを付けることができます。しかし、`\widetilde` および `\widehat` のアクセントは、以下の例のように、出力では3文字分の長さにしかありません。

$$A + \widetilde{B} = C - D$$


前小節で述べた `\overset` コマンドと `\underset` コマンドを使っても、複数の文字にアクセントを付けることができます。 `\underset{A=B}{***}` というコマンドは、

$$\underset{***}{A=B}$$

のようになります。

8. 空白

8.1. 定義済みの空白

数式に水平方向の空白を挿入することが、必要になることがあります。これは、保護された空白 (ショートカット 捷 徑 : 未定義) を挿入することで実現できます。「`\`」が現れるので、Space を何回か押すことによって、8 種の異なる長さの空白のうち一つを選択することができます。空白は、数式ツールバーボタンの  を押すか、特定のコマンドを入力することで、挿入することができます。挿入したコマンド如何に関わらず、直後に Space を押すことによって、長さを変更することができます。

コマンド	保護された空白を挿入したのち、Space を叩く回数	出力	コマンド	保護された空白を挿入したのち、Space を叩く回数	出力
<code>\,</code>	0	$A B$	<code>\hfill</code>	5	$A \quad B$
<code>\:</code>	1	$A : B$	<code>\hspace*{1em}</code>	6	$A \quad B$
<code>\;</code>	2	$A ; B$	<code>\hspace{1em}</code>	7	$A \quad B$
<code>\quad</code>	3	$A \quad B$	<code>_</code>	8	$A \quad B$
<code>\qquad</code>	4	$A \quad B$	<code>\!</code>	9	$A B$

5–7 回 Space を押した場合は、第 8.2 節に説明されているように、可変空白となります。9 回押した場合は、一見、空白を生まないように見えます。実はこれは負の長さなので、他の長さと異なり、 \LaTeX 中では赤で表示されます。以下のように、他にもう二つ、負の長さの空白があります。

コマンド	$\backslash\text{negmedspace}$	$\backslash\text{negthickspace}$
保護された空白を挿入したのち、Space を叩く回数	10	11
出力	AB	AB

負の空白を使うと、文字が重なってしまうことがあります。これを利用して、合字処理を強制することができます。これは、たとえば以下のように、和演算子に使えます。

コマンド	出力
$\backslash\text{sum}\backslash\text{sum_f_kl}$	$\sum\sum f_{kl}$
$\backslash\text{sum}\backslash\text{negmedspace}\backslash\text{sum_f_kl}$	$\sum\sum f_{kl}$

イコール記号などの関係子は、つねに空白を前後に伴うようになっていますが、これを抑制するには、イコール記号を \TeX 括弧で囲みます。以下の例にこれを示します。

$$\begin{array}{ll} \text{通常の数式} & A = B \\ \text{空白なしの数式} & A=B \end{array}$$

二行目の数式を作るコマンドは、 $A\{\Rightarrow B$ です。

物理単位には、値と単位のあいだに通常空白ではなく、最小の空白を入れる必要があるために、それに適した空白が必要です。本文中の単位には、挿入▷整形▷小空白メニューショートカット (捷 徑 : OptionShiftSpace) で、最小の空白を挿入することができます。

違いを示す例を以下に掲げます。

24 kW·h 値と単位のあいだに通常空白を入れた例
24 kW·h 値と単位のあいだに最小の空白を入れた例

8.2. 可変長の空白¹⁸

指定した長さの空白が、 $\backslash\text{hspace}$ コマンドで入力することができます。すると、ながい「 $\rule{1cm}{0.4pt}$ 」が現れます。長さは、「 $\rule{1cm}{0.4pt}$ 」を左クリックすることによって指定することができます。長さは負の値でも構いません。空白が行冒頭の文字である場合、それは無視されます。この場合でも、空白出力を強制させたいときには、 $\backslash\text{hspace}$ コマンドの代わりに $\backslash\text{hspace*}$ コマンドを使用するか、「 $\rule{1cm}{0.4pt}$ 」の上を左クリックして、保護オプションをチェックしてください。

数式が使用できる空白をすべて使い尽くすだけの空白を挿入するには、 $\backslash\text{hfill}$ コマンドを使用します。

¹⁸数式中の垂直方向の空白については、第 18.1.1 節をご覧ください。

コマンド (<code>\hspace</code> 長さ)	出力	
$A=B\hspace{A}\not=C$ (3 cm)	$A = B$	$A \neq C$
$A\hspace{A}\not=A$ (-1 mm)	$AA \neq A$	
$A=A\hfill B=B$	$A = A$	$B = B$

上記の最後の例では、使用できる空白は、表の列中もっとも長い要素によって規定されます。行内数式では、空白は、`\hfill` が挿入された行の長さに依存します。つまり、その行が全幅を使用している場合、空白はまったく作られません。また `\hfill` は、別行立て数式中では、**行頭下げ**数式スタイルが使われているときのみ、意味を持ちます(数式スタイルは第 17 節で説明されています)。

`\hfill` の他にも、空白を模様で埋める `\dotfill` や `\hrulefill` といったコマンドがあります。用例については第 3.9 節をご参照下さい。

本文中では、可変長空白は、挿入▷整形▷水平方向の空白メニューで挿入することができます。

(例)

この行には、2 cm の空白が入っています。

この行には、最大の空白が入っています。

8.3. 行内数式周りの空白

行内数式前後の空白は、長さ `\mathsurround` を使って調節することができます。長さの値は、以下の書式を持つ `\setlength` コマンドを使って設定することができます。

`\setlength{長さ名}{値}`

`\mathsurround` を 5 mm の値に設定するには、以下のコマンド

`\setlength{\mathsurround}{5mm}`

を TeX モードで挿入します。すると、5 mm の空白がすべての行内数式の前後に設定されることになります。

この行には、周囲に 5 mm の余白を設定した行内数式 $A = B$ があります。

既定値に戻すには、`\mathsurround` を 0 pt の値に戻して下さい。

9. ボックスと枠

本文中のボックスについては、取扱説明書埋め込みオブジェクト篇のボックスの章に述べられています。

9.1. 枠付きボックス

`\fbox` コマンドや `\boxed` コマンドを使えば、数式やその一部を枠の中に入れることができます。

どちらかのコマンドを数式に挿入すると、枠の中に青枠が現れ、数式の断片を入れることができます。 `\fbox` の場合には、そのままではボックスの中身が数式テキストとして取り扱われてしまうので、`Ctrl+M` を使って、このボックスの中にもう一度数式を作らなくてはなりません。 `\boxed` を使った場合には、新しい数式が自動的に枠内に作られます。

`\fbox` コマンドは、数式がつねに本文の大きさに設定されてしまうので、別行立て数式に枠を付けるのには適していません。逆に `\boxed` は、数式がつねに別行立て数式の大きさに設定されてしまうので、行内数式に枠をつけるのには適していません。

`\fbox` の拡張として、枠幅と配置も指定することができる `\framebox` コマンドがあります。 `\framebox` は、以下の書式を持ちます。

`\framebox[枠幅][位置]{ボックスの内容}`

「位置」は、*l* か *r* の値をとります。 *l* は、ボックス中で数式を左寄せ、 *r* は右寄せにします。位置を指定しない時には、数式は中央揃えになります。

「枠幅」を指定しない時には、位置を指定することができません。この場合には、 `\fbox` と同様、枠幅がボックスの内容に応じて調節されるのです。

`\framebox` コマンドを挿入すると、三つの青枠を含むボックスが現れます。最初の二つの枠は括弧で囲まれており、二つとも非必須の変数であることを意味します。三つ目の枠は、 `\fbox` と同様、数式の断片を入れるためのものです。

コマンド	出力
<code>\fbox_Ctrl+M \int_A=B</code>	$\int A = B$
<code>\boxed_\int_A=B</code>	$\int A = B$
<code>A+\fbox_B</code>	$A + B$
<code>\framebox_20mm→→Ctrl+M \frac_AB</code>	$\frac{A}{B}$

枠の厚みも調節可能です。そのためには、以下のコマンドを数式の前に \TeX モードで挿入しなくてはなりません。

`\fboxrule` “厚み” `\fboxsep` “距離”

“距離”は、枠とボックス内の一文字目との間の距離を示します。これを使った例として、以下の枠付き数式をご覧ください。

$$A + B = C$$

この数式の直前には、

`\fboxrule 2mm \fboxsep 3mm`

というコマンドが、 \TeX モードで挿入されています。ここで与えられた値は、以後のすべてのボックスに適用されます。

標準の枠寸法に戻すには、

`\fboxrule 0.4pt \fboxsep 3pt`

というコマンドを、次の数式が始まる前に \TeX モードで挿入しておきます。

9.2. 枠なしボックス

枠のないボックスを作るには、`\mbox`・`\makebox`・`\raisebox` の三つのコマンドがあります。

`\raisebox` を使うと、ボックスを上付きにしたり下付きにしたりすることができます。しかし、通常の上付き文字・下付き文字とは違い、ボックス内の文字寸法はそのまま保たれます。`\raisebox` は、以下の書式で用いられます。

`\raisebox{高さ}{ボックスの内容}`

`\fbox` と同様、ボックスに数式を入れる際には、明示的に数式として入れる必要があります。**【註】** 下の最後の `\raisebox` のところで、`Ctrl+M` を一回でなく二回押すことによって、もう一段数式をいれています。これは、`LyX` が `\raisebox` を直接サポートしていないためです。

コマンド	出力
<code>H\raisebox{2mm}{\{al\rightarrow lo}</code>	$H^{al}lo$
<code>H\raisebox{-2mm}{\{al\rightarrow lo}</code>	$H_{al}lo$
<code>A=\raisebox{-2mm}{\{Ctrl+M Ctrl+M \sqrt{B}</code>	$A = \sqrt{B}$

枠がないことを除けば、`\mbox` コマンドは `\fbox` と同じであり、`\makebox` は `\framebox` と同じです。

9.3. 色付きボックス

本節で説明されているコマンドをすべて使えるようにするためには、`LaTeX` プリアンプルに

`\usepackage{color}`

という行¹⁹を書き加えて、`LaTeX` パッケージの `color`²⁰を読み込む必要があります。

ボックスに色を付けるには、`\colorbox` コマンドを以下の書式で使います。

`\colorbox{色}{ボックスの内容}`

ボックスの内容には、別のボックスが含まれても構いませんし、`\colorbox` 自体も、別のボックスに入っている場合でも構いません (以下の二番目と三番目の例を参照してください)。ボックスに数式を含める場合には、`\raisebox` と同様、明示的に数式を作らなくてはなりません²¹。

選択できる定義済みの色としては、

`black`(黒)・`blue`(青)・`cyan`(シアン)・`green`(緑)・`magenta`(マゼンタ)・`red`(赤)・`white`(白)・`yellow`(黄)


があります。

¹⁹定義済みの色を使って、文書中のどこかで文章に色を付けてある場合、`LyX` は、自動的に `LaTeX` パッケージ `color` を読み込みます。したがって、本パッケージが二度読み込まれる可能性があるわけですが、そうなったとしても問題は生じません。

²⁰`LaTeX` パッケージ `color` は、すべての標準的な `LaTeX` 頒布版に含まれています。

²¹これは、`\fcolorbox` コマンドにも当てはまります。

コマンド	出力
<code>\colorbox{yellow}\{A=B</code>	$A=B$
<code>\colorbox{green}\{\fbox{A=B</code>	$A=B$
<code>\fbox{\colorbox{green}\{\Ctrl+M Ctrl+M \int C=D</code>	$\int C = D$

`\colorbox` は、ボックスに色をつけるだけで、ボックス内の文字には色付けをしません。すべての文字に色付けするには、数式全体を選択し、文字様式ダイアログで欲しい色を選択します。このダイアログは、ツールバーボタン  か編集▷文字様式▷任意設定メニューで開くことができます。すると、数式番号も数式と同じ色になります。数式番号が数式の文字とは別の色になるようにするには、数式内部で色を変えなくてはなりません。

たとえば、

$$\int A = B \tag{1}$$

$$\int A = B \tag{2}$$

数式 (1) は、全体が赤色です。

数式 (2) は、数式番号を緑色にするために、まず全体を緑色にします。その後、数式内の文字を赤色にします。

ボックスの枠だけ別の色にするには、`\fcolorbox` コマンドを以下の書式で使します。

`\fcolorbox{枠の色}{色}{ボックスの内容}`

つまり、`\fcolorbox` は `\colorbox` コマンドの拡張です。`\framebox` と同様に、枠の厚みは `\fboxrule` と `\fboxsep` で設定します。たとえば、

$$A=B$$

のようにします。

上記の数式は、以下のコマンドで作成されています。

`\fcolorbox{cyan}\{magenta\{A=B.`

定義済みの色以外の色を使いたい場合には、まずその色を定義しなくてはなりません。

たとえば、「darkgreen」という色を定義するには、 \LaTeX プリアンブルに

`\definecolor{darkgreen}{cmyk}{0.5, 0, 1, 0.5}`

という行を書き加えます。

cmyk とは、**cyan**(シアン)・**magenta**(マゼンタ)・**yellow**(黄)・**black**(黒)の各色を表す色空間です。コンマで区切られた四つの数字は、この色空間における各色の出力強度です。強度は、0-1の範囲をとることができます。定義には、**cmyk**の他に、**rgb**という色空間を使うこともできます。**rgb**とは、**red**(赤)・**green**(緑)・**blue**(青)の各色を意味

し、この場合には、各色に対応した三つの出力強度を指定します。さらに、灰色の出力強度のみをとる **gray** という色空間もあります。

例として、文字が **yellow** に色付けされ、新しく定義した **darkgreen** という色を持つ枠付きボックスを挙げておきます。


$$\int A dx = \frac{\sqrt[5]{B}}{\ln\left(\frac{1}{3}\right)} \quad (3)$$

`\textcolor` コマンドを使うと、以下のように、自前で定義した色をテキスト中でも使用することができます。

この文は「darkgreen」です。

`\textcolor` は、`\textcolor{色}{色付けをする文}` という書式で 사용할ことができます。

9.4. 段落ボックス

いくつかの行や段落を含む、いわゆる段落ボックス (parbox) は、挿入 ▢ ボックスメニュー か ツールバー ボタン  で作成することができます。

以下の例は、行中の枠付き parbox を示したものです。

この行は、

これは段落ボックスです。これはちょうど 5 cm の幅になっており、以下のように数式を含めることもできます。 $\int A ds = C$

 parbox の入った行です。

このようなボックスは、灰色のボックス挿入枠を右クリックすることによって作ることができます。すると、ボックスの特性を表示したダイアログが現れます。上の例では、**装飾**：簡素な長方形の箱型、**内部ボックス**：parbox コマンド、**幅**：5 cm、**垂直ボックス配置**：中央、に設定されています。

L^AT_EX では、parbox は、以下の書式を持つ `\parbox` コマンドによって作られます。

`\parbox[位置]{幅}{ボックスの内容}`

「位置」は、*b* と *t* の値をとることができます。下揃えを意味する *b*(bottom) は、ボックスを、周囲の本文中の最後の行と合わせることを意味します。上揃えを意味する *t*(top) は、これを最初の行に合わせます。位置を指定しない時には、ボックスは事実上中央揃えになります。用例については、取扱説明書埋込オブジェクト篇の **ボックス** の節をご参照下さい。

数式番号を含めて、数式を完全に枠で囲むためには、数式を parbox 内に収めなくてはなりません。こうするには、数式前に T_EX モードで `\parbox{\linewidth-2\fbboxsep-2\fbboxrule}` というコマンドを挿入します。ここで `\linewidth` は、使用中の文書に設定されている行幅です。枠は、parbox の外側にあるので、枠余白と枠幅の 2 倍を行幅から差し引かなくてはなりません。引数中で掛け算や引き算を行うためには、L^AT_EX パッケージの **calc**²² を、L^AT_EX プリアンブル中で

²²calc は、標準的 L^AT_EX 頒布版のすべてに含まれています。

`\usepackage{calc}`

のように読み込んでおく必要があります。数式の後では、 $\text{T}_{\text{E}}\text{X}$ モードで`}}`を入力して、二つのボックスを閉じておかななくてはなりません。以下に例を挙げます。

$$\int A \, dx = \frac{\sqrt[5]{B}}{\ln\left(\frac{1}{3}\right)} \quad (4)$$

`\fbox` の引数として `parbox` が使われているので、この場合には、`\fbox` を使おうが `\boxed` を使おうが、差は生じません。

段落ボックスは、数式にじかにコメントを付けるのにたいへん便利です。これを行うには、`\parbox` を `\tag` コマンドといっしょに使います (`\tag` についての詳細は、第 19.5 節をご参照下さい)

以下は、`\parbox` を使ってコメントを付けた数式の例です。

$$5x - 7b = 3b$$

これは説明です。数式や多行数式本体からはっきりと離れています。

$\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ は、まだ数式中での `\parbox` コマンドをサポートしていないので、上のような数式は、完全に $\text{T}_{\text{E}}\text{X}$ モードで挿入しなくてはなりません。この数式は、以下のようなコマンド列を使って作ってあります。

まず、`\[5x-7b=3b\tag*{\{\parbox{5cm}\}}` というコマンドを $\text{T}_{\text{E}}\text{X}$ モードで挿入します²³。それから、説明を通常のテキストとして入れ、最後に`}}\]` を $\text{T}_{\text{E}}\text{X}$ モードで挿入します。ここで `\[` および `\]` コマンドは別行立て数式を作るためのものです。

`\parbox` を使う利点は、数式テキストモードを使用して「コメントを付けた」以下の例と比較すると、よくわかるでしょう。

$$5x - 7b = 3b \text{ これは説明です。数式本体から離れていません...}$$

10. 演算子

10.1. 大演算子

ここに挙げた積分演算子をすべて使えるようにするには、文書設定の数式オプションの面にある `esint` パッケージを自動的に使うオプションを有効にしなくてはなりません。

²³行頭下げ数式様式を使用している時には、`\tag*{\}` の代わりに `\hfill` を用いることもできます (数式様式に関しては、第 17 節をご参照下さい)。

コマンド	出力	コマンド	出力
<code>\int</code>	\int	<code>\sum</code>	Σ
<code>\oint</code>	\oint	<code>\prod</code>	\prod
<code>\ointctrclockwise</code>	\oint	<code>\coprod</code>	\coprod
<code>\ointclockwise</code>	\oint	<code>\bigodot</code>	\odot
<code>\sqint</code>	\int	<code>\bigotimes</code>	\otimes
<code>\fint</code>	\int	<code>\bigoplus</code>	\oplus
<code>\landupint</code>	\int	<code>\bigwedge</code>	\wedge
<code>\landdownint</code>	\int	<code>\bigvee</code>	\vee
<code>\bigcap</code>	\bigcap	<code>\bigsqcup</code>	\sqcup
<code>\bigcup</code>	\bigcup	<code>\biguplus</code>	\uplus

すべての大演算子は、数式ツールバーボタンの \int でも挿入することができます。

これらの演算子は、よく見ないと同じように見える二項演算子よりも大きいので、大演算子と呼ばれます。大演算子はすべて、次小節で説明する「範囲」をとることができます。

積分演算子はすべて、`\intop`や`\ointop`のように、`op`で終わる別バージョンがあります。これらの演算子は、`\int`とは範囲の表示のしかたが異なります。第 10.2 節をご参照下さい。

積分の子細

積分中で用いられる文字 d は演算子なので、アップライト体で組まれなくてはなりません。これを行うには d を選択して、^{ショートカット} 捷 径 Control C R を用います²⁴。最後に、演算子の慣例に倣って、 d の前に最小空白を挿入しなくてはなりません。たとえば、

正しくない例： $\int A(x)dx$

正しい 例： $\int A(x) \, dx$

多重積分に関しては、以下のコマンドがあります。

コマンド	出力	コマンド	出力
<code>\iint</code>	\iint	<code>\iiint</code>	\iiint
<code>\oiint</code>	\oiint	<code>\iiiint</code>	\iiiint
<code>\sqiint</code>	\sqiint	<code>\dotsint</code>	$\int \cdots \int$

10.2. 演算子の範囲

範囲は、上付き文字と下付き文字とで作成することができます。

コマンド	出力
<code>\prod^{\infty}_{0 \rightarrow} A(x)</code>	$\prod_0^\infty A(x)$

²⁴文字様式については、第 11.1 節参照。

行内数式では、範囲は演算子の右横に表示されます。別行立て数式での範囲は、積分範囲を除き、演算子の上と下に表示されます。

範囲が演算子の横に表示されるように強制するには、カーソルを当該演算子の直後において、編集▷数式▷範囲の表記を変更メニューで**行内形式** (ショートカット 捷徑 : Control M L) を選択することで範囲形式を変更することができます。以下はその用例です。

既定の範囲形式は、以下のようになっています。

$$\sum_{x=0}^{\infty} \frac{1}{x^2}$$

以下は、範囲形式を**行内形式**に変更したときの表示です。

$$\sum_{x=0}^{\infty} \frac{1}{x^2}$$

`\intop` や `\ointop` などのように `op` で終わるもの以外の積分記号では、範囲は、既定で演算子の横に設定されます。しかし、多重積分においては、範囲を演算子の下に置くべきときがあります。このことから、以下の例では、範囲形式を**別行立て形式**にして積分記号の下に置くようにしています。

$$\iiint_V X \, dV = U \quad (5)$$

範囲に条件を指定したい場合には、`\subarray` コマンドや `\substack` コマンドを使用します。たとえば、以下の表記

$$\sum_{\substack{0 < k < 1000 \\ k \in \mathbb{N}}}^n k^{-2} \quad (6)$$

を作成するには、以下のようになくってはなりません。

まず、`\sum^n_` というコマンドを入力します。すると、和演算子の下の青枠に移動するので、ここに `\subarray_` コマンドを挿入します。すると、青枠が紫枠の中に入って、ここに複数の行を書き込むことができるようになります。新しい行は、改行 (CommandReturn) を挿入することで作ることができます。ここに

0 < k < 1000 Ctrl+Return

と入力すると、新規行のための新しい枠が現れます。

各行の揃え方は、表ツールバーか編集▷行と列メニューで変更することができますが、右揃えにするには、行頭に `\hfill_` を挿入しなくてはなりません。

`\substack` コマンドは、各行がつねに中央揃えになることを除いては、`\subarray` と同じです。

演算子の後に来る文字は、範囲の横に来るので、(6) 式のように、演算子の横の余白が大きくなりすぎることがあります。これを防ぐには、コマンド `\smashoperator` を使うことができます。これを有効化するには、文書設定の数式オプションで、**mathtools** パッケージの常に読み込みオプションを指定してください。`\smashoperator` は、内容物の幅を 0pt に設定します。

これを (6) 式に応用してみると、コマンド

`\smashoperator{`

の波括弧の中に、範囲付きの演算子を挿入することになります。波括弧を閉じた後に、数式を続けます。例：

$$\sum_{\substack{0 < k < 1000 \\ k \in \mathbb{N}}}^n k^{-2}$$

演算子の片側だけ、空白を 0pt にすることも可能です。そのためには、`\smashoperator` と付属する波括弧の間に、`[l]` または `[r]` と書き込みます。下記は、`\smashoperator` のとりうる形の例です。

$$Y \sum_{1 \leq i \leq j \leq n}^{n=3456} X_{ij} = Y \sum_{1 \leq i \leq j \leq n}^{n=3456} X_{ij} = Y \sum_{1 \leq i \leq j \leq n}^{n=3456} X_{ij}$$

範囲を持つ演算子が続くとき、出力での、範囲の組版は出来が良くないものになることがあります。下記をご覧ください。

$$\text{a) } \lim_{n \rightarrow \infty} \max_{p \geq n} \quad \text{b) } \lim_{n \rightarrow \infty} \max_{p^2 \geq n} \quad \text{c) } \lim_{n \rightarrow \infty} \sup_{p^2 \geq nK} \quad \text{d) } \limsup_{n \rightarrow \infty} \max_{p \geq n}$$

組版を改善するには、文書設定の数式オプションで、`mathtools` パッケージの常に読み込みオプションを指定してください。その上で、数式中の最初の演算子の前に直接、`\adjustlimits` コマンドを置いてください。そうすると、上記の例はこのようなになります。

$$\text{a) } \lim_{n \rightarrow \infty} \max_{p \geq n} \quad \text{b) } \lim_{n \rightarrow \infty} \max_{p^2 \geq n} \quad \text{c) } \lim_{n \rightarrow \infty} \sup_{p^2 \geq nK} \quad \text{d) } \limsup_{n \rightarrow \infty} \max_{p \geq n}$$

一つの範囲を複数の演算子に用いる方法が、第 10.5 節に述べられています。

10.3. 演算子の修飾

`\overset` コマンドや `\underset` コマンドを使うと、それぞれ演算子の上や下に、文字をアクセントとして付けることができます。また、`\sideset` コマンドを使うと、文字を演算子の前や後ろに付けることができます。

`\sideset` には 4 つの派生版があります。

- `\sideset` は、演算子の隅に文字を置くのに用います。
- `\sidesetn` は、演算子の前後に文字を置くのに用います (この派生版は、 \LaTeX の元々の `\sideset` コマンドを表します)。
- `\sidesetl` は、演算子の左隅と後ろに文字を置くのに用います。
- `\sidesetr` は、演算子の右隅と前に文字を置くのに用います。

たとえば、`\sidesetn{\sum}\rightarrow_{k=1}^n` というコマンドを入力すると、

$$\sum_{k=1}^n$$

のようになります。

【註】`\sideset` は、大演算子を修飾するためだけに用いることができ、二項演算子には用いることができません。


のようになります。最後の例からわかるように、`\overset` や `\underset` では、記号や文字にアクセントをつけることもできます。一方、また、`\overset{_}{\maltese}\uparrow a` というコマンドならば、



10.4. 二項演算子

二項演算子は、前後に文字がある場合、周囲に余白が入ります。

コマンド	出力	コマンド	出力	コマンド	出力
<code>+</code>	+	<code>\nabla</code>	∇	<code>\oplus</code>	\oplus
<code>-</code>	-	<code>\bigtriangledown</code>	\bigtriangledown	<code>\ominus</code>	\ominus
<code>\pm</code>	\pm	<code>\bigtriangleup</code>	\bigtriangleup	<code>\otimes</code>	\otimes
<code>\mp</code>	\mp	<code>\Box</code>	\Box	<code>\oslash</code>	\oslash
<code>\cdot</code>	\cdot	<code>\cap</code>	\cap	<code>\odot</code>	\odot
<code>\times</code>	\times	<code>\cup</code>	\cup	<code>\amalg</code>	\amalg
<code>\div</code>	\div	<code>\dagger</code>	\dagger	<code>\uplus</code>	\uplus
<code>*</code>	$*$	<code>\ddagger</code>	\ddagger	<code>\setminus</code>	\setminus
<code>\star</code>	\star	<code>\wr</code>	\wr	<code>\sqcap</code>	\sqcap
<code>\circ</code>	\circ	<code>\bigcirc</code>	\bigcirc	<code>\sqcup</code>	\sqcup
<code>\diamond</code>	\diamond	<code>\wedge</code>	\wedge	<code>\triangleleft</code>	\triangleleft
<code>\bullet</code>	\bullet	<code>\vee</code>	\vee	<code>\triangleright</code>	\triangleright

二項演算子は、すべて数式ツールバーボタンの  から挿入することもできます。

ラプラス演算子を組版するには、`\bigtriangleup` 以外に、`\Delta` や `\nabla^2` (∇^2) を使用することもできます。

挿入▷特殊文字メニューのメニュー区切りで入力される文字は、`\triangleright` 演算子です。

10.5. 自己定義演算子

L^AT_EX プリアンブルで `\DeclareMathOperator` コマンドを使用すると、自己定義演算子を定義することができます。このコマンドの書式は

`\DeclareMathOperator{新規コマンド}{表示}`

です。「表示」は、出力での演算子の表示され方を定義する文字や記号です。大演算子を定義するには、コマンドの後に「*」を置きます。自己定義の大演算子は、すべて第 10.2 節で述べられた範囲を指定することができます。

たとえば、以下のような L^AT_EX プリアンブル行

`\DeclareMathOperator*{\Lozenge}{\blacklozenge}`

は、第13.2節にある菱形記号を使った大演算子を挿入する、以下のようなコマンド`\Lozenge`を定義します。

$$\sum_{n=1}^{\infty}$$

上記の数式を作るコマンドは、`\Lozenge^{\infty}\rightarrow _n=1` です。

自己定義演算子を、同一文書内で複数回用いない時には、以下の書式を持つ`\mathop` コマンドおよび`\mathbin` コマンドを用いて定義を行うこともできます。

(書式)`\mathop{表示}`および`\mathbin{表示}`

`\mathop` は大演算子を定義し、`\mathbin` は二項演算子を定義します。

たとえば`\mathop` は、以下のように、複数の演算子に共通の範囲指定を行うのに用いることができます。

$$\sum_{i,j=1}^N$$

上記の数式では

`\mathop{\sum\limits_{i,j=1}^N}` というコマンドを用いています。

11. 書体

11.1. 書体様式

数式中のラテン文字は、以下の書体様式のうちいずれかに設定することができます。

コマンド	出力	Shortcut
<code>\mathbb{ABC}</code>	\mathbb{ABC}	-
<code>\mathbf{AbC}</code>	\mathbf{AbC}	CommandB
<code>\boldsymbol{AbC}</code>	\boldsymbol{AbC}	ControlCommandB
<code>\mathcal{ABC}</code>	\mathcal{ABC}	-
<code>\mathfrak{AbC}</code>	\mathfrak{AbC}	-

コマンド	出力	Shortcut
<code>\mathit{AbC}</code>	AbC	-
<code>\mathrm{AbC}</code>	AbC	ControlC R
<code>\mathsf{AbC}</code>	AbC	ControlC ControlS
<code>\mathtt{AbC}</code>	\mathtt{AbC}	ShiftCommandP
<code>\mathscr{ABC}</code>	\mathscr{ABC}	-

【註】`\mathbb`, `mathcal` 様式と`\mathscr` 様式は、大文字にのみ使用することができます。

既定では、`\mathnormal` 様式に設定されています。

書体様式コマンドは、以下のように数式構成要素内の文字に対しても機能します。

$$\mathfrak{a} = \frac{b}{c}$$

数式テキストに含まれる文字に対しては、数式書体様式は反映せず、`\textrm` 様式で表示されます。数式テキストの様式を文字様式ダイアログで設定することができないのは、`LyX` のバグです²⁵。

書体様式コマンドの代わりに、編集▷数学▷文字様式ダイアログや、`\AA`を使用することもできます。

11.2. ボールド体の数式

数式全体をボールド体にしようとする、前節の`\mathbf` コマンドは、ギリシャ文字の小文字に対しては機能しないので、使用することができません。さらにこのコマンドは、以下の式のように、ラテン文字をつねにアップライト体に印字してしまいます。

$$\int_n^2 \mathbf{f}(\theta) = \Gamma \quad \text{\code{\mathbfbf}を使用した数式}$$

この数式を正しく表示するには、以下のように、`\boldsymbol` コマンドを使用します。

$$\int_n^2 \boldsymbol{f}(\theta) = \Gamma \quad \text{\code{\boldsymbol}を使用した数式}$$

また、数式を`boldmath`環境に設定する方法もあります。この環境は、`TeX` モードで`\boldmath` コマンドを挿入することによって作ることができます。環境を閉じるには、`\unboldmath` コマンドを `TeX` モードで挿入します。

$$\int_n^2 \boldsymbol{f}(\theta) = \Gamma \quad \text{boldmath 環境に置いた数式}$$

11.3. 色付きの数式

数式も、通常の本文と同様、色を付けることができます。数式あるいは数式の一部を選択して、文字様式ダイアログを使用して下さい。下記は、マゼンタ色にした数式です。

$$\int A \, dx = \frac{\sqrt[5]{B}}{\ln\left(\frac{1}{3}\right)}$$

第 9.3 節に述べられているように、自己定義の色を定義することもできます。自己定義の色は、以下の書式を持つ`\textcolor``TeX` コードコマンドで適用することができます。

²⁵`LyX`-bug #4629

`\textcolor{色}{文字ないし数式}`

下記の例は、全体を濃緑にし、一部を赤にしています。

$$\int A dx = \frac{\sqrt[5]{B}}{\ln\left(\frac{1}{3}\right)}$$

LyX のバグのため、自己定義色は数式全体に対してしか使用することができません²⁶。


11.4. 書体寸法

数式内の文字については、本文中の文字同様、以下の書体寸法設定コマンドがあります。

`\Huge`, `\huge`, `\LARGE`, `\Large`, `\large`, `\normalsize`, `\small`,
`\footnotesize`, `\scriptsize`, および `\tiny`

これらのコマンドによって生成される実際の書体寸法は、文書の書体寸法に依存し、文書の書体寸法が `\normalsize` コマンドに設定されます。他のコマンドは、`\normalsize` を基準として拡大ないし縮小されます。しかしながら、書体寸法は一定の値を越えることができないようになっています。たとえば、文書書体寸法が 12pt であるならば、`\Huge` コマンドは `\huge` コマンドと同じ大きさに落とされます。

ある場所以降のすべての数式と本文文字を変更するには、書体寸法コマンドを T_EX モードで挿入します。元の書体寸法に戻すには、数式の後に T_EX モードで `\normalsize` コマンドを挿入します。

数式内では、以下の寸法コマンドまたはツールバーボタンを使用して、寸法を変更することができます。

コマンド	出力
<code>\displaystyle</code>	$E_{\text{pot}1} = \frac{K}{l + \frac{m}{n_2}}$
<code>\textstyle</code>	$E_{\text{pot}1} = \frac{K}{l + \frac{m}{n_2}}$
<code>\scriptstyle</code>	$E_{\text{pot}1} = \frac{K}{l + \frac{m}{n_2}}$
<code>\scriptscriptstyle</code>	$E_{\text{pot}1} = \frac{K}{l + \frac{m}{n_2}}$

これらのコマンドを入力すると、青いボックスが現れるので、そこに数式のパーツを入れることができます。

フォント寸法を変更するにはもう一つの方法がありますが、これは記号と数式内テキストのみに使うことができます。これを使うには、書体寸法コマンドを数式テキスト内に挿入します。数式テキストの終わりか、別の書体寸法コマンドが現れるまでの文字すべてが、選択した寸法になります。以下に二つの例を挙げます。

$$A = \frac{B}{c} \cdot \text{✠}$$

²⁶[LyX-bug #5269](#)




二つの式の前には、`\huge` コマンドが挿入されています。二つ目の数式を入力するコマンドは、

```
\maltese_A Alt+M M \Large_\maltese_\textit_A→→
      Alt+M M \tiny_\maltese_\textit_A
```

のようになります。

ある記号を別の寸法で表示することができないときには、その記号はつねに既定寸法で表示されます。

12. ギリシャ文字

すべてのギリシャ文字は、ツールバーボタンの  からでも挿入することができます。各国の組版規則では、数式内のギリシャ文字はどれもイタリック体か斜体で組版されなくてはならないことになっていますが、フランス語やロシア語などいくつかの言語では、それにもかかわらず立体で組版されることがあります。

12.1. 小文字

コマンド	出力	コマンド	出力	コマンド	出力
<code>\alpha</code>	α	<code>\iota</code>	ι	<code>\varrho</code>	ϱ
<code>\beta</code>	β	<code>\kappa</code>	κ	<code>\sigma</code>	σ
<code>\gamma</code>	γ	<code>\varkappa</code>	\varkappa	<code>\varsigma</code>	ς
<code>\delta</code>	δ	<code>\lambda</code>	λ	<code>\tau</code>	τ
<code>\epsilon</code>	ϵ	<code>\mu</code>	μ	<code>\upsilon</code>	υ
<code>\varepsilon</code>	ε	<code>\nu</code>	ν	<code>\phi</code>	ϕ
<code>\zeta</code>	ζ	<code>\xi</code>	ξ	<code>\varphi</code>	φ
<code>\eta</code>	η	<code>o</code>	o	<code>\chi</code>	χ
<code>\theta</code>	θ	<code>\pi</code>	π	<code>\psi</code>	ψ
<code>\vartheta</code>	ϑ	<code>\varpi</code>	ϖ	<code>\omega</code>	ω
		<code>\rho</code>	ρ		

アップライト体のギリシャ文字を作成する方法は、第 24.9 節に説明されています。

12.2. 大文字

コマンド	出力	コマンド	出力
<code>\Gamma</code>	Γ	<code>\Sigma</code>	Σ
<code>\Delta</code>	Δ	<code>\Upsilon</code>	Υ
<code>\Theta</code>	Θ	<code>\Phi</code>	Φ
<code>\Lambda</code>	Λ	<code>\Psi</code>	Ψ
<code>\Xi</code>	Ξ	<code>\Omega</code>	Ω
<code>\Pi</code>	Π		

大文字のギリシャ文字が立体で表示されるのは、 $\text{T}_{\text{E}}\text{X}$ の開発途上に生じたデザイン上のバグによるものです。正しいイタリック体の大文字を得るためには、各コマンドの頭に `\var` を付けてください。たとえば、`\varGamma` コマンドは、 Γ を生成します。もう一つの方法は、パッケージ `fixmath`²⁷ を $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ プリアンブル行に

```
\usepackage{fixmath}
```



と書いて読み込む方法です。すると、文書中の大きなギリシャ文字は、すべて自動的にイタリック体として組版されます。

12.3. ボールド体

ギリシャ文字は、ラテン文字のように、多様な書体様式に設定することができません。ギリシャ文字をボールド体にできるのは、`\boldsymbol` コマンドのみです。

コマンド	出力
<code>\Upsilon\boldsymbol{\Upsilon}</code>	$\Upsilon\Upsilon$
<code>\theta\boldsymbol{\theta}</code>	$\theta\theta$

13. 記号²⁸

本節に掲げてある各記号の多くは、ツールバーボタンの  や  でも挿入することができます。

13.1. 数学記号

コマンド	出力	コマンド	出力	コマンド	出力
<code>\neg</code>	\neg	<code>\forall</code>	\forall	<code>\prime</code>	$'$
<code>\Im</code>	\Im	<code>\exists</code>	\exists	<code>\backprime</code>	\backprime
<code>\Re</code>	\Re	<code>\nexists</code>	\nexists	<code>\mho</code>	\mho
<code>\aleph</code>	\aleph	<code>\emptyset</code>	\emptyset	<code>\triangle</code>	\triangle
<code>\partial</code>	∂	<code>\varnothing</code>	\varnothing	<code>\angle</code>	\angle
<code>\infty</code>	∞	<code>\dagger</code>	\dagger	<code>\measuredangle</code>	\measuredangle
<code>\wp</code>	\wp	<code>\ddagger</code>	\ddagger	<code>\sphericalangle</code>	\sphericalangle
<code>\imath</code>	\imath	<code>\complement</code>	\complement	<code>\top</code>	\top
<code>\jmath</code>	\jmath	<code>\Bbbk</code>	\Bbbk	<code>\bot</code>	\bot

²⁷`fixmath` は、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ パッケージ `was` に含まれています。

²⁸各 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ パッケージに含まれる全記号をほとんど網羅した一覧が、[4] にあります。

13.2. その他の記号

コマンド	出力	コマンド	出力	コマンド	出力
<code>\flat</code>	♭	<code>\hbar</code>	ℏ	<code>\diamondsuit</code>	◇
<code>\natural</code>	♮	<code>\hslash</code>	ℏ	<code>\Diamond</code>	◇
<code>\sharp</code>	♯	<code>\clubsuit</code>	♣	<code>\heartsuit</code>	♥
<code>\surd</code>	✓	<code>\spadesuit</code>	♠	<code>\P</code>	¶
<code>\checkmark</code>	✓	<code>\bigstar</code>	★	<code>\copyright</code>	©
<code>\yen</code>	¥	<code>\blacklozenge</code>	◆	<code>\circledR</code>	®
<code>\pounds</code>	£	<code>\blacktriangle</code>	▲	<code>\maltese</code>	✠
<code>\$</code>	\$	<code>\blacktriangledown</code>	▼	<code>\diagup</code>	／
<code>§</code>	§	<code>\bullet</code>	•	<code>\diagdown</code>	＼

寸法を変えて表示することのできる記号もあります。第 11.4 節をご参照下さい。

13.3. ユーロ通貨記号

ユーロ通貨記号を数式で使用するには、 \LaTeX パッケージ `eurosym` が導入されていて、以下のような \LaTeX プリアンブル行によって読み込まれていなくてはなりません。

```
\usepackage[gennarrow]{eurosym}
```


すると、ユーロ通貨記号を `\euro` コマンドで挿入することができるようになります。

数式テキストには、`eurosym` が導入されていなくても、ユーロ通貨記号を直接キーボードを使って (あれば) 挿入することができます。 `eurosym` が導入されていれば、`\euro` は \TeX モードでも挿入することができます。また、正式な通貨記号を `\officialeuro` コマンド (これは \TeX モードでのみ使用することができます) で挿入することができます。

以下は、各ユーロ通貨記号のまとめです。


	コマンド	出力
数式	<code>\euro</code>	€
数式テキスト	(ユーロ記号)	(ユーロ記号)
\TeX モード	<code>\officialeuro</code>	€

14. 関係子

関係子はすべて、ツールバーボタンの  でも挿入することができます。

コマンド	出力	コマンド	出力	コマンド	出力
<	<	=	=	>	>
\le	\leq	\not=	\neq	\ge	\geq
\ll	\ll	\equiv	\equiv	\gg	\gg
\prec	\prec	\sim	\sim	\succ	\succ
\preceq	\preceq	\simeq	\simeq	\succeq	\succeq
\subset	\subset	\approx	\approx	\supset	\supset
\subseteq	\subseteq	\cong	\cong	\supseteq	\supseteq
\sqsubseteq	\sqsubseteq	\bowtie	\bowtie	\sqsupseteq	\sqsupseteq
\in	\in	\notin	\notin	\ni	\ni
\vdash	\vdash	\perp	\perp	\dashv	\dashv
\smile	\smile	\propto	\propto	\frown	\frown
\lhd	\triangleleft	\asymp	\asymp	\rhd	\triangleleft
\unlhd	\triangleleft	\doteq	\doteq	\unrhd	\triangleleft
\gtrless	\gtrless	\circeq	\circeq	\lessgtr	\lessgtr
\mid	$ $	\models	\models	\parallel	\parallel
\nmid	\nmid	\widehat{=}	$\widehat{=}$	\nparallel	\nparallel

\lhd と \rhd の文字は、同じように見える演算子 \triangleleft および \triangleright よりも大きくなっています。

L_AT_EX は、特殊な関係子を多くサポートしています。これらの関係子一覧を見るには、ツールバーボタン  をご覧ください。

関係子は、記号とは違って、つねに前後に余白が置かれます。

\stackrel コマンドを使うと、以下のように、ラベル付きの関係子を作ることができます。

コマンド	出力
$A(r)\stackrel{r\rightarrow\infty}{\approx}B$	$A(r)\stackrel{r\rightarrow\infty}{\approx}B$

15. 関数

15.1. 定義済み関数

一般的に、数式表現では変数はイタリック体に設定されますが、関数名はイタリック体にしません。なぜなら、*sin* は *s*・*i*・*n* であるかのように誤解させる恐れがあるためです。そのために、定義済み関数が存在し、これらは先行する要素よりも少し離れて配置されます。定義済み関数は、関数名の前にバックスラッシュを加えたコマンドとして挿入します。

コマンド	出力	コマンド	出力
$A\sin(x)+B$	$A\sin(x)+B$	$A\sin(x)+B$	$A\sin(x)+B$

以下の関数が定義済みです。

コマンド	コマンド	コマンド	コマンド
\sin	\sinh	\arcsin	\sup
\cos	\cosh	\arccos	\inf
\tan	\tanh	\arctan	\lim
\cot	\coth	\arg	\liminf
\sec	\min	\deg	\limsup
\csc	\max	\det	\Pr
\ln	\exp	\dim	\hom
\lg	\log	\ker	\gcd

上記は、数式ツールバーボタンの^{exp}_{tan}でも挿入することができます。

15.2. 自己定義関数

たとえば符号関数 $\operatorname{sgn}(x)$ のように、定義済みでない関数を使うには、二つの方法があります。

- 以下の行を \LaTeX プリアンブルに加えることによって関数を定義します。²⁹

```
\DeclareMathOperator{\sgn}{sgn}
```

これによって、新しく定義された関数を $\text{\textbackslash sg n}$ コマンドで呼び出すことができるようになります。

- 数式を普通書き下し、関数名を選択して (上記の例では sgn の文字)、それを数式テキストに変更します。最後に、空白を先行する要素と関数の間に入れます。

双方とも定義済み関数と同等の出力をもたらします³⁰。

コマンド	出力
$A \backslash \operatorname{sgn}(x) + B$	$A \operatorname{sgn}(x) + B$
$A \backslash, \underbrace{\operatorname{sgn}}_{\text{Alt+MM}}(x) + B$	$A \operatorname{sgn}(x) + B$

自己定義関数を何回か使用する場合には、一番目の方法の方が適切です。

15.3. 極限

極限用には、 $\text{\textbackslash lim}$ ・ $\text{\textbackslash liminf}$ ・ $\text{\textbackslash limsup}$ の他に、以下の関数があります。

²⁹ $\text{\textbackslash DeclareMathOperator}$ についての詳細は、第 10.5 節をご参照下さい。

³⁰ $\text{\textbackslash LyX}$ 上では、自己定義関数は赤で表示され、定義済み関数は黒で表示されます。

コマンド	出力
<code>\varliminf</code>	\varliminf
<code>\varlimsup</code>	\varlimsup
<code>\varprojlim</code>	\varprojlim
<code>\varinjlim</code>	\varinjlim

極限は、下付き文字を挿入することによって示されます。行内数式では、極限は、以下のように関数の横に置かれます。

コマンド	出力
<code>\lim_x\to A_x=B</code>	$\lim_{x \rightarrow A} x = B$

別行立て数式では、極限は、以下のように通常どおり下に置かれます。

$$\lim_{x \rightarrow A} x = B$$

極限の組版を調整する仕方については、第 10.2 節をご覧ください。

15.4. 剰余関数

剰余関数は、特別に 4 つの派生型があります。

以下は、別行立て数式での派生型です。

コマンド	出力
<code>a\mod b</code>	$a \bmod b$
<code>a\pmod b</code>	$a \pmod b$
<code>a\bmod b</code>	$a \bmod b$
<code>a\pod b</code>	$a \pod b$

行内数式では、すべての派生型で、関数名の前の余白がすこし小さく設定されます。既定では、剰余関数は、続く一文字目だけを考慮します。二文字以上を使いたい時には、下記のように文字を TeX 括弧で囲みます。

コマンド	出力
<code>a\podbcd</code>	$a \pod (b)cd$
<code>a\pod\{bc\}d</code>	$a \pod (bc)d$

16. 特殊文字


`^` および `_` の各文字は、通常、上付き文字や下付き文字を生成してしまいます。これらの文字自体を得るには、その前に `\` を付けなくてはなりません。

コマンド	出力
<code>\^</code>	\wedge
<code>_</code>	$_$

16.1. 数式テキストにおける特殊文字

以下の各コマンドは、数式テキストか $\text{T}_{\text{E}}\text{X}$ モード中でのみ使用することができます。

コマンド	出力	コマンド	出力
$\backslash\text{oe}$	œ	$\backslash\text{o}$	ø
$\backslash\text{OE}$	Œ	$\backslash\text{O}$	Ø
$\backslash\text{ae}$	æ	$\backslash\text{l}$	ł
$\backslash\text{AE}$	Æ	$\backslash\text{L}$	Ł
$\backslash\text{aa}$	å	$\backslash\text{!}\text{'}\text{}$	ı
$\backslash\text{AA}$	Å	$\backslash\text{?}\text{'}\text{}$	İ
$\backslash\text{i}$	ı	$\backslash\text{j}$	Ј

Å と Ø の各文字は、数式ツールバーボタンの  から挿入することができます。

例外は、 $\backslash\text{!}\text{'}$ と $\backslash\text{?}\text{'}$ の各コマンドで、これらは直接 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 中の本文に入れることができます。

16.2. 文章中のアクセント

以下に挙げる各コマンドを使えば、すべての文字にアクセントを付けることができます。これらのコマンドは、 $\text{T}_{\text{E}}\text{X}$ モードで入れなくてはなりません。

コマンド	出力	コマンド	出力
$\backslash\text{'e}$	ë	$\backslash\text{H}_\text{e}$	ě
$\backslash\text{'e}$	è	$\backslash\text{'e}$	é
$\backslash\text{^}_\text{e}$	ê	$\backslash\text{~}_\text{e}$	ẽ
$\backslash\text{=}_\text{e}$	ē	$\backslash\text{.}_\text{e}$	ė
$\backslash\text{u}_\text{e}$	ě	$\backslash\text{v}_\text{e}$	ė
$\backslash\text{b}_\text{e}$	ę	$\backslash\text{d}_\text{e}$	ę
$\backslash\text{t}_\text{ee}$	êe	$\backslash\text{c}_\text{e}$	ę

$\backslash\text{t}$ コマンドは異なる二つの文字にアクセントを付けることもできます。たとえば、コマンド $\backslash\text{t}_\text{sz}$ は、sz となります。

‘・’・^ の各アクセントは、 $\text{T}_{\text{E}}\text{X}$ モードを使わなくても、母音といっしょに直接キーボードから入力することもできます。チルダ³¹を、*a* や *n* や *o* といっしょに使うときも同様です。

$\backslash\text{b}\cdot\backslash\text{c}\cdot\backslash\text{d}\cdot\backslash\text{H}\cdot\backslash\text{t}\cdot\backslash\text{u}\cdot\backslash\text{v}$ の各コマンドと、キーボードから直接挿入するアクセントは、数式テキスト中でも使うことができます。他のアクセントについては、数式内向けの特別な数式コマンドがあります。第 7.1 節をご参照下さい。

さらに、 $\backslash\text{textcircled}$ コマンドを使えば、著作権マークのように、あらゆる数字や文字を丸で囲む—敢えて言えば、丸囲みでアクセントを付けるようなものといえるでしょう—ことができます。

³¹ これは、チルダがアクセントとして定義されているキーボードのみに適用されます。

コマンド	出力
<code>\textcircled{w}</code>	Ⓜ
<code>\Large \textcircled{\normalsize\protect\raisebox{-1.5pt}{W}}</code>	Ⓜ

ここではユーザーが、文字が丸のなかに収まるように調整してやらなくてはなりません。ここでは、`\Large`³²で丸の大きさを指定しています。そして`\raisebox`³³を使って、文字が真ん中にくるようにしています。

16.3. 古式数字

古式数字は、`\oldstylenums` コマンドで作成することができます。このコマンドは、数式中でも $\text{T}_{\text{E}}\text{X}$ モード中でも使うことができます。コマンド書式は、

`\oldstylenums{数字}`

です。`\oldstylenums{0123456789}` というコマンドは、0123456789 のようになります。

17. 数式様式

- 以下の二つの配置様式があります。

中央揃え 事前に定義された標準です。

行頭下げ これは文書▷設定メニューの数式オプションで設定することができます。

数式を字下げを指定すると下げ幅を指定することができます。既定値は使用している文書苦楽に依存して変わります。

- また、以下の二つの連番様式があり、文書▷設定メニューの数式オプションで設定することができます。

右 これが多くの場合既定値です。

左

これで選択した様式は、文書中のすべての別行立て数式に用いられます。もし、中央揃えと行頭下げの両様式を同一文書中で用いたい場合には、**中央揃え**様式を指定するようにします。そして、行頭下げにしたい数式は、`flalign` 環境に指定するようにします。第 18.2.3 節をご覧ください。

18. 多行数式

18.1. 概要



$\text{L}_{\text{Y}}\text{X}$ では、多行数式は、数式中で `CommandReturn` を押すことで作られます。この操作によって、第 18.3 節に述べられている `eqnarray` 環境が作り出されるか、あるいは文書設

³²第 11.4 節参照のこと。

³³第 9.2 節参照のこと。

定で AMS math パッケージを使うオプションが選択されている場合には、第 18.2.1 節に述べられている **align 環境** が作り出されることになります。

他にも、挿入▷数式メニューで作ることのできる多行数式環境があります。これらの環境は、以下の各節で説明します。

すべての多行数式において、新規行は、CommandReturn を押すことによって作られます。行を足したり削ったりするには、数式ツールバーボタンの  や  を使うか、編集▷行と列メニューを使うことができます。

18.1.1. 行間

以下のように、多行数式において行のあいだの空白が足りないことが、ときどき起こります。

$$\begin{aligned} B^2(B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) + 4x_0^2x^2 + 4x_0xD &= -4x^2B^2 + 4x_0xB^2 \\ 4x^2(B^2 + x_0^2) + 4x_0x(D - B^2) + B^2(B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) &= 0 \end{aligned}$$

L^AT_EX において行間を付け加えるには、新規行コマンドに非必須の引数をとらせて指定しますが、これはまだ L_YX には実装されていない³⁴ので、数式全体を T_EX モードで入れなくてはなりません。上記の例の行間を大きくするには、最初の行の最後に `\[3mm]` というコマンドを入れます。すると、次のようになります。

$$\begin{aligned} B^2(B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) + 4x_0^2x^2 + 4x_0xD &= -4x^2B^2 + 4x_0xB^2 \\ 4x^2(B^2 + x_0^2) + 4x_0x(D - B^2) + B^2(B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) &= 0 \end{aligned}$$

同一数式内のすべての行の行間を一律に指定するには、`\jot` 変数を変更します。定義は、`行間 = 6 pt + \jot` となっています。`\jot` の既定値は、3 pt です。上記の例のように、行間を 3 mm 追加するには、

`\setlength{\jot}{3mm+3pt}`

というコマンドを、数式直前に T_EX モードに入れておきます。これを行うには、L^AT_EX プリアンプルに

`\usepackage{calc}`

という行を入れて、L^AT_EX パッケージ **calc**³⁵ を読み込んでおく必要があります。すると、

$$\begin{aligned} B^2(B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) + 4x_0^2x^2 + 4x_0xD &= -4x^2B^2 + 4x_0xB^2 \\ 4x^2(B^2 + x_0^2) + 4x_0x(D - B^2) + B^2(B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) &= 0 \end{aligned}$$

のような結果を得ます。行間を既定値に戻すには、`\jot` をふたたび 3 pt に戻します。

³⁴[LyX-bug #1505](#) を参照。

³⁵**calc** は標準的な L^AT_EX 頒布版のすべてに付属しています。

18.1.2. 列間

多行数式は、行列を形成します。たとえば、`eqnarray` 環境の数式は、3 列からなる行列です。この環境で列間を変更すれば、関係子周辺の余白を変更することができます。

列間は、`\arraycolsep` 変数を使って指定し、

列間 = `2\arraycolsep`

という関係があります。したがって、

`\setlength{\arraycolsep}{1cm}`

というコマンドを $\text{T}_\text{E}\text{X}$ モードで入れると、ここから後のすべての数式の列間が 2 cm になります。これを既定値に戻すには、`\arraycolsep` を 5 pt に戻して下さい。

以下は、2 cm の列間を持つ数式です。

$$\begin{array}{ccc} A & = & B \\ C & \neq & A \end{array}$$

行列の既定の列間 10 pt を持つ数式です。

$$\begin{array}{ccc} A & = & B \\ C & \neq & A \end{array}$$

18.1.3. 長い数式

長い数式は、以下の方法を使って組版することができます。

- 左辺ないし右辺の一方が、行幅よりもかなり短いときには、以下のように、短い方を左辺に置き、右辺を二行に分けて組版します。

$$\begin{aligned} H = & W_{SB} + W_{mv} + W_D - \frac{\hbar^2}{2m_0}\Delta - \frac{\hbar^2}{2m_1}\Delta_1 - \frac{\hbar^2}{2m_2}\Delta_2 - \frac{e^2}{4\pi\epsilon_0|\mathbf{r} - \mathbf{R}_1|} \\ & - \frac{e^2}{4\pi\epsilon_0|\mathbf{r} - \mathbf{R}_2|} + \frac{e^2}{4\pi\epsilon_0|\mathbf{R}_1 - \mathbf{R}_2|} \end{aligned} \quad (7)$$

二行目の最初のマイナス記号は、行頭の文字になってしまうため、通常、表示の上で演算子としては取り扱われません。前後に余白が置かれることもなく、分数線からも離れて表示されません。これを避けるために、マイナス記号の後に`\hspace`³⁶コマンドを使って 3 pt 空白を入れてあります。

- 数式の両辺がともに長すぎるときには、`\lefteqn` コマンドを使います。これを最初の行の第一列に入れると、以下のように、続きの内容が他の列にかかって表示されます。

$$\begin{aligned} & 4x^2(B^2 + x_0^2) + 4x_0x(D - B^2) + B^2(B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) + D^2 \\ & - B^2 - 2B\sqrt{r_g^2 - x^2 + 2x_0x - x_0^2 + r_g^2 - x^2 + 2x_0x - x_0^2} \\ & = B^2 + 2(r_g^2 + 2x_0x - x_0^2 - r_k^2) + \frac{(r_g^2 + 2x_0x - x_0^2 - r_k^2)^2}{B^2} \end{aligned} \quad (8)$$

³⁶`\hspace` に関する詳細は、第 8.2 節をご覧ください。

`\lefteqn`を入力すると、青枠から少し左にずれたところに現れる紫枠にカーソルが移るので、ここに数式を入力します。二行め以降の行の内容は、二列め以降の列に挿入します。挿入する列が右になるほど、字下げの量が大きくなります。

`\lefteqn`を使用する際には、以下のことに注意下さい。

- 数式では、ページ幅全部を使うことはしません。たとえば、上記の例で、最初の行に $-B^2$ という項を置いたとすると、ページ余白の領域に出てしまいますが、これはよくありません。幅をうまく使うには、最初の行の行頭に負の空白を入れる方法もあります。
- \LaTeX のバグによって、最初の行にマウスでカーソルを入れることはできません³⁷。カーソルを行頭に合わせて、矢印^{キー}で移動するしかありません。
- 長い数式を組む他の方法として、第 18.5 節と第 18.6 節で述べられている環境を用いる方法があります。

18.1.4. 多行にわたる分数

分数の分母もしくは分子が長すぎて、一行に入りきらないことがあります。この場合には、分数内で改行をしなくてはなりません。これを行うには、`\splitfrac` コマンドを使うことができます。これを有効化するには、文書設定の数式オプションで、`mathtools` パッケージの常に読み込みオプションを指定してください。`\splitfrac` のスキームは

`\splitfrac{1 行目}{2 行目}`

となっており、分母にも分子にも使うことができます。

$$a = \frac{xy + xy + xy + xy + xy + wy + wy + wy + wy}{z} \neq \frac{xy + \frac{xy}{z} + xy + xy + xy + xy + xy + xy}{z}$$

上記の最初の分子に使われているコマンドは、

`\splitfrac{xy+xy+xy+xy+xy}{wy+wy+wy+wy}` です。

18.1.5. 多行にわたる括弧

多行にわたる括弧を作ろうとすると、以下のような問題が生じます。

$$A = \sin(x) \left[\prod_{R=1}^{\infty} \frac{1}{R} + \cdots \right. \\ \left. \cdots + B - D \right]$$

可変寸法の括弧は多行にわたることができないので、閉じ括弧が初めの括弧よりも小さくなっています。

³⁷[LyX-bug #1429](#)

二行めの括弧の大きさを正しく設定するには、最初の行の終わりを`\right.`とし、二行めの始めを`\left.`³⁸とします。一行めにおいては、範囲付き積演算子`\prod`がもっとも大きな記号であり、これが二行めの括弧の大きさにならなくてはならないので、`\left.`の後に、`\vphantom{\prod^{\infty}_{R=1}}`というコマンドを挿入します。

その結果が以下の数式です。



$$A = \sin(x) \left[\prod_{R=1}^{\infty} \frac{1}{R} + \cdots \right. \\ \left. \cdots + B - D \right]$$

18.2. align 環境

`align` 環境は、すべての型の多行数式に使うことができ、とくに、いくつかの数式を並べて表示させるのに便利です。

`align` 環境には列があり、奇数列は右揃え、偶数列は左揃えに設定されます。`align` 環境の行にはすべて、付番することができます。

`align` 環境は、挿入▷数式メニューから作ることができます。編集▷数式▷数式の表記を変更メニューを使えば、既存の数式を `align` 環境に変更することができます。

列を追加したり削除したりするには、数式ツールバーボタンの  や  を使うか、編集▷行と列メニューを使います。

18.2.1. 標準 align 環境

この `align` 環境は、数式中で `CommandReturn` を押すか、挿入▷数式▷AMS `align` 環境メニューで作ることができます。

以下は、二つの数式を横に並べた例ですが、これは 4 列からなる `align` 環境として作ります。

$$\begin{array}{ll} A = \sin(B) & C = D \\ C \neq A & B \neq D \end{array}$$

ご覧になって分かるように、この環境の数式は、一列めの前と偶数列の後に `\hfill`³⁹があるかのように配置されます。数式様式として行頭下げ⁴⁰を使う場合には、第一列の前の `\hfill` はない形で数式が設定されます。

18.2.2. alignat 環境

`alignat` 環境には、事前に設定された列間が存在しません。列間は、必要ならば、第 8 節に述べられている空白を使用して手動で入れます。

³⁸`\left` と `\right` に関する詳細は、第 5.1.2 節をご覧ください。

³⁹`\hfill` に関する詳細は、第 8.2 節をご覧ください。

⁴⁰数式様式については、第 17 節をご覧ください。

以下は、上記の例を alignat 環境に設定し、二つめの数式の頭に 1 cm の空白を入れたものです。

$$\begin{array}{cc} A = \sin(B) & C = D \\ C \neq A & B \neq D \end{array}$$

列間を各列ごとに設定することができるので、この環境は、とくに三つないし四つの数式を横に並べるのに向いています。

18.2.3. flalign 環境

この環境では、つねに、最初の二列をできるだけ左寄せにし、最後の二列をできるだけ右寄せにするように設定されます。以下がその例です。

$$\begin{array}{ccc} A = 1 & B = 2 & C = 3 \\ X = -1 & Y = -2 & Z = 4 \end{array}$$

奇数列の flalign 環境を作成し、最後の列に空の $\mathrm{T}_\mathrm{E}\mathrm{X}$ 括弧を入れておくと、数式様式として**中央揃え**が用いられているときでも、一部の数式を左寄せにすることができます。以下は、例として (5) 式を行頭下げにしたものです。

$$\iiint_V X \, dV = U \tag{9}$$

ここで、最初の二列には数式が入れられており、行頭下げ数式様式と同等の字下げを行うために、30 pt の空白が第 1 列の頭に入れてあります。

18.3. eqnarray 環境

この環境を作成すると、三つの青枠が現れます。最初の枠の内容は右寄せに設定され、最後の枠の内容は左寄せに設定されます。中央の枠は、関係子のみを入れることを想定しているので、その内容は中央揃えで少し小さく設定されます。

$$\begin{array}{ccc} \frac{ABC}{D} & \frac{ABC}{D} & \frac{ABC}{D} \\ AB & AB & AB \\ A & = & A \end{array}$$

18.4. gather 環境

この環境には、中央揃えに設定された列一つしかありません。行はすべて付番することができます。

$$A = 1 \tag{10}$$

$$X = -1 \tag{11}$$

18.5. multiline 環境

gather 環境同様、multiline 環境には、中央揃えに設定された列一つしかありません。ただし、一行めが左揃えに設定され、最終行が右揃えに設定されるのです。他の行はすべて中央揃えになります。このことから、この環境は、長い数式に使うのに向いています。用例として、(8) 式を multiline 環境に置いたものを示します。

$$\begin{aligned} & 4x^2 (B^2 + x_0^2) + 4x_0x (D - B^2) + B^2 (B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) + D^2 \\ & - B^2 - 2B\sqrt{r_g^2 - x^2 + 2x_0x - x_0^2 + r_g^2 - x^2 + 2x_0x - x_0^2} \\ & = B^2 + 2(r_g^2 + 2x_0x - x_0^2 - r_k^2) + \frac{(r_g^2 + 2x_0x - x_0^2 - r_k^2)^2}{B^2} \quad (12) \end{aligned}$$

文書の付番設定が右寄せ (左寄せ) になっているときには、出力では、multiline 環境の最後 (最初) の行だけが付番されます⁴¹。

`\shoveright` コマンドや `\shoveleft` コマンドを使えば、中央揃えの行を右寄せや左寄せにすることができます。これらのコマンドは、以下のように使います。

`\shoveright{行の内容}` あるいは `\shoveleft{行の内容}`

`\multlinegap` 長は、一行めの左ページ余白からの距離を指定します。既定値は 0 pt の長さです。

以下は、上記の数式に

`\setlength{\multlinegap}{2cm}`

というコマンドを、 \TeX モードで直前に挿入した例です。

$$\begin{aligned} & 4x^2 (B^2 + x_0^2) + 4x_0x (D - B^2) + B^2 (B^2 - 2r_g^2 + 2x_0^2 - 2r_k^2) + D^2 \\ & - B^2 - 2B\sqrt{r_g^2 - x^2 + 2x_0x - x_0^2 + r_g^2 - x^2 + 2x_0x - x_0^2} \\ & = B^2 + 2(r_g^2 + 2x_0x - x_0^2 - r_k^2) + \frac{(r_g^2 + 2x_0x - x_0^2 - r_k^2)^2}{B^2} \quad (13) \end{aligned}$$

二行めは、`\shoveleft` を使って左揃えにしています。

18.6. 数式の一部の多行化

数式の一部のみを多行表示したい場合には、`aligned`・`alignedat`・`gathered`・`split` のうちのいずれかの環境を使用します。これらは、挿入▷数式メニューか、本節で解説している各コマンドを使用して挿入することができます。

最初の三つの環境は、環境名から `ed` を省いた同名の多行数式環境と同じ性格を持ちますが、環境の前後に数式を続けることが可能です。たとえば、

$$\left. \begin{aligned} \Delta x \Delta p &\geq \frac{\hbar}{2} \\ \Delta E \Delta t &\geq \frac{\hbar}{2} \end{aligned} \right\} \text{不確定性原理}$$

⁴¹付番様式については、第 17 節を参照。

この数式を作るには、別行立て数式をまず作っておいて、そこに`\aligned` コマンドを挿入しています。紫枠の中に青枠が現れるので、そこに必要に応じて、列や行を加えていきます。この多行環境の外には、括弧などの他の数式要素を入れることができます。

`aligned` 環境は、長い数式を水平方向を揃えて表示するのにも向いています。別行立て数式内で `aligned` を用いるようにすると、数式番号を行末の、数式全高の中心に配置できる利点があります。以下に例として、(7) 式に `aligned` 環境を適用したものを示します。

$$H = W_{SB} + W_{mv} + W_D - \frac{\hbar^2}{2m_0}\Delta - \frac{\hbar^2}{2m_1}\Delta_1 - \frac{\hbar^2}{2m_2}\Delta_2 - \frac{e^2}{4\pi\epsilon_0|\mathbf{r} - \mathbf{R}_1|} - \frac{e^2}{4\pi\epsilon_0|\mathbf{r} - \mathbf{R}_2|} + \frac{e^2}{4\pi\epsilon_0|\mathbf{R}_1 - \mathbf{R}_2|} \quad (14)$$

`alignedat`・`gathered`・`split` の各環境を使うには、それぞれ`\alignedat`・`\gathered`・`\split` の各コマンドを挿入します。`split` 環境は、`aligned` 環境と同じ性格を持ちますが、二つの列しか作ることができません。

18.7. 多行数式中のテキスト

各 `align` 系環境および `multline`・`gather` 環境では、独立した行に列揃えの影響を受けない形でテキストを挿入することができます。これを行うには、以下の書式を持つ`\intertext` コマンドを使います。

`\intertext{テキスト}`

テキストのハイフネーションを行うことはできないので、テキストは一行に収めなくてはなりません。L^AT_EX は、現時点では`\intertext` を直接サポートしていないので、テキストは数式テキストとして書き入れます。ここで、`\intertext` は行頭になくなくてはならず、当該行の上に出力されます。以下は、二行めの行頭にテキストを入れた例です。

$$I = a\sqrt{2} \int_0^{2\pi} \sqrt{1 + \cos(\phi)} \, d\phi \quad (15)$$

被積分関数は $\phi = \pi$ において対称なので、

$$= 2a\sqrt{2} \int_0^{\pi} \sqrt{1 + \cos(\phi)} \, d\phi \quad (16)$$

`\intertext` は、文と数式行の間に、常に縦空白を生み出します。この空白を抑制するには、文書設定の数式オプションで `mathtools` パッケージを常に読み込みに設定してください。そうすると、以下のように、`\intertext` の代わりに`\shortintertext` コマンドを使うことができます。

$$I = a\sqrt{2} \int_0^{2\pi} \sqrt{1 + \cos(\phi)} \, d\phi \quad (17)$$

被積分関数は $\phi = \pi$ において対称なので、

$$= 2a\sqrt{2} \int_0^{\pi} \sqrt{1 + \cos(\phi)} \, d\phi \quad (18)$$

19. 数式番号


19.1. 概要

付番数式は、挿入▷数式▷付番数式メニュー (ショートカット 捷 径 : Ctrl+Alt N) で作ることができます。既存の数式に番号を振るには、編集▷数式▷数式全体を付番メニュー (捷徑: ControlM N) を使います。L_YX 中において数式番号は、数式の後に括弧に囲まれた「#」で表されます。「#」は、実際の出力では数式番号に置き換えられます。


多行数式で付番が有効になっているときには、すべての行に番号が振られます。ただし、編集▷数式▷この行を付番メニュー (捷徑: ControlM ShiftN) を使用すれば、各行毎に付番するかどうか指定することができます。

行内数式を除いて、すべての数式は二通りの様式で番号を振ることができます。第 17 節をご覧ください。

19.2. 相互参照

ラベルを付けた数式は、すべて相互参照することができます。ラベルは、挿入▷ラベルメニューか、ツールバーボタンので付けることができます。このとき、カーソルは別行立て数式の中になくても構いません。すると、テキストフィールドの中に **eq:** という接頭語の入ったダイアログが現れるので、接頭語の後にラベルを挿入します。この既定の接頭辞は「equation(数式)」を意味し、こうして数式ラベルであるとの標を付けることによって、節ラベルなどから区別し、大きな文書の中でラベルを見つけるのを容易にします。ラベルを変更するには、挿入▷ラベルメニューをもういちど使って下さい。

L_YX 中でラベル名は、数式の後ろに、二つの括弧に囲まれて表示されます。ラベル付きの数式はつねに付番されます。

相互参照は、挿入▷相互参照メニューかツールバーボタンのを使って挿入します。数式相互参照は、出力では数式番号として表示されます。相互参照ダイアログで「(<参照>)」書式を選択した場合には、出力での相互参照は、括弧に囲まれた数式番号として表示されます。

L_YX 中で相互参照を右クリックすると、参照先の数式に移動することができます。

以下は、後の各小節に現れる数式への相互参照を含む例です。

(何とかかんとか) 式と (20b) 式は、等価です。(XXIII) 式とは異なり、(Y) 式では、付番にラテン数字を使用しています。

`\tag42`の引数が、第 9.4 節で述べたボックスを含んでいるときには、その数式を参照することはできません。

19.3. 細目番号

Formulas can be numbered as subequations. To activate this feature, add the module Subequations to the document in the menu Document▷Settings▷Modules. To insert subequations use the menu Insert▷Custom Insets▷Subequations.

⁴²`\tag` は、第 19.5 節に説明があります。

The equations to be subnumbered are put into the **Subequations** inset. たとえば,

$$A = C - B \quad (19)$$

$$B = C - A \quad (20a)$$

$$C = A + B \quad (20b)$$

Every formula in the **Subequations** inset will be subnumbered with $a \cdot b \cdot c \dots$ のように細目番号が振られます.

細目番号への参照は, (20a) や (20b). If you want to reference the subequations collectively, insert a label as first thing in the **Subequations** inset. 数式 (21) はその例です.

By inserting subequations vertical space is added before it. これを戻すために, -5 mm vertical space right before the **Subequations** inset. 数式様式として行頭下げ⁴³を用いているときには, これを-7 mm 空白にしてください.

To get another subnumber instead of small Latin letters, insert this command as **T_EX** code before the equations in the **Subequations** inset:

`\renewcommand{\theequation}{\theparentequation - \roman{equation}}`

ここで, `\theparentequation` は親番号を生成し, `\roman{equation}` は細目番号として小文字ローマ数字を生成します. 番号付けの調整について, 詳しくは第 19.4 節を参照してください.

下記は, 番号付けを調整済みで, 2 行目の付番を消した多行数式の例です.

$$A = (B - Z)^2 = (B - Z)(B - Z) \quad (21-i)$$

$$\begin{aligned} &= B^2 - ZB - BZ + Z^2 \\ &= B^2 - 2BZ + Z^2 \end{aligned} \quad (21-ii)$$

19.4. ローマ数字や文字を使った付番

数式は, ローマ数字やラテン文字を使って付番することもできます. たとえば, 小文字のローマ数字を使って付番するには, 数式の前に **T_EX** モードで

`\renewcommand{\theequation}{\roman{equation}}`

というコマンドを入れます. `\renewcommand` は, 定義済みのコマンド `\theequation` をコマンド `\roman{equation}` に再定義します⁴⁴. ここで, **equation** は数式カウンタです. コマンド `\the` をカウンタの接頭辞として使用すると, カウンタの値がアラビア数字として出力されます. 数式に番号を振ると, **L^AT_EX** は, 内部的に `\theequation` コマンドを数式の後ろに置くのです. `\roman{equation}` は, カウンタを小文字のローマ数字として出力します.

こうして, `\renewcommand` コマンド以降の数式はすべて, ローマ数字で付番されるようになります. 大文字のローマ数字での付番に切り替えたいときは, 同じコマンドの `\roman` の部分を `\Roman` に変えて挿入します. また, 小文字ラテン文字を使って

⁴³数式様式に関しては, 第 17 節を参照.

⁴⁴`\renewcommand` コマンドは, 第 22.1 節に述べられている `\newcommand` コマンドと同じ書式を持ちます.

「付番」したいときのために、`\alph` コマンドがあり、大文字ラテン文字を使って付番したいときのためには、`\Alph` コマンドがあります。

【註】ラテン文字を使うと、一つの文書の中で、最大 26 個の数式しか番号を振ることができません。

$$A = \text{小文字ローマ数字} \quad (\text{xxii})$$

$$B = \text{大文字ローマ数字} \quad (\text{XXIII})$$

$$C = \text{小文字ラテン文字} \quad (\text{x})$$

$$D = \text{大文字ラテン文字} \quad (\text{Y})$$

既定の付番方式に戻すには、以下のコマンドを挿入してください。

```
\renewcommand{\theequation}{\arabic{equation}}
```

$$E = \text{アラビア数字} \quad (26)$$

以上からわかるとおり、付番様式の違いに関わらず、数式番号は連番で振られます。様式変更時に「1」から番号が始まるようにするためには、新しい数式カウンタを定義しなくてはなりません。この点に関する説明は、ファイル [Formula-numbering.lyx](#) にあります。

19.5. 自己定義番号

【註】 If the document language is a right-to-left language such as Arabic, you must add this to the \LaTeX -preamble to be able to use the commands described in this section:

```
\AtBeginDocument{
\def\tagform@#1{\maketag@@@{\ignorespaces#1\unskip}} }
```

標準の付番では、数式番号の周りに括弧が表示されます。括弧をたとえば縦棒に置き換えるには、 \LaTeX プリアンブルに以下の行を付け加えます。

```
\AtBeginDocument{
\def\tagform@#1{\maketag@@@{|#1|}} }
```

他の記号を使いたいときには、`#1` 脇の縦棒を一つないし複数の文字で置き換えて下さい。数式番号だけで良い時は、縦棒を削除して下さい。

数式の後ろに、括弧に囲まれた連番の代わりに、何かしらの表現が欲しいときには、以下のように `\tag` コマンドを使います。

$$A + B = C \quad (\text{何とかかんとか})$$

上記の例では、`\tag_何とかかんとか` というコマンドを数式に打ち込んでいます。

代わりに `\tag*_何とかかんとか` というコマンドを使うと、星印は表現の周りの括弧を抑制するので、以下ようになります。

$$A + B = C \quad \text{something}$$

数式番号を、文書中の新しい部門や節ごとに振りなおしたいときには、部に関しては

`\@addtoreset{equation}{part}`

節に関しては

`\@addtoreset{equation}{section}`

というコマンドを使います。

これらのコマンドを \TeX モードで使えるようにするためには、`\makeatletter` コマンドで「@」字を \LaTeX 中で「有効」にしてやらなくてはなりません。一方、`\makeatother` コマンドはこれを無効にします。したがって、 \TeX モード中での上記コマンド列は、

`\makeatletter`

`\@addtoreset{equation}{section}`

`\makeatother`

のようにならなくてはなりません。

\LaTeX プリアンブル中では、`\makeatletter` と `\makeatother` は、 \LaTeX が内部的に自動で挿入するので省略してかまいません。

`\@addtoreset` を戻すには、まず \LaTeX プリアンブル中に

`\usepackage{remreset}`

という行を入れて、`remreset.sty`⁴⁵ ファイルを読み込んでおかななくてはなりません。その後

`\@removefromreset` コマンドを `\@addtoreset` と同じ書式で使用すると、`\@addtoreset` を戻すことができます。

ときには、数式を

(節番号. 数式番号)

のような形で付番し、節ごとに数式番号を「1」から始めさせなくてはならないときがあります。

このような場合のために、`\numberwithin` というコマンドがあり、

`\numberwithin{カウンタ}{節階層}`

という書式で用います。「カウンタ」は、どの番号を制御するかを表し、「節階層」は点の前に何の番号を振るのかを表します。

したがって、ここでは \LaTeX プリアンブルか \TeX コードで

`\numberwithin{equation}{section}`

という行を用いることにしましょう。その結果がこれです。

$$A + B = C \quad (19.27)$$

たとえば、部番号を節階層として使用して、表に付番を施すときには

`\numberwithin{table}{part}` を用います。

標準の付番方式に戻したいときや、この種の付番が文書クラスで定義されているときに、それを止めさせたい場合には、

⁴⁵`remreset` は、 \LaTeX パッケージの `carlisle` の一部として含まれており、 \LaTeX 標準頒布版には含まれていません。

`\renewcommand{\theequation}{\arabic{equation}}`

あるいは

`\renewcommand{\thetable}{\arabic{table}}`

というコマンドを $\text{T}_{\text{E}}\text{X}$ コードとして入れるか、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ プリアンブルに入れます。 `\numberwithin` は、内部的に上記で述べた `\@addtoreset` コマンドを使用しているので、これも使用後は戻しておかなくてはなりません。

20. 化学記号と化学式

以下は、化学関係の文章の例です。

SO_4^{2-} イオンは、2つの Na^+ イオンと反応して、硫酸化塩 (Na_2SO_4) を形成します。この化学式は以下のようになります。



この化学式は、直接数式として作成することができます。記号がイタリック体として表示されることを防ぐには、全体を選択してから ショートカット 捷 徑 Control C R を押せば、アップライトフォント様式に変更することができます⁴⁶。

化学式を組版するのにもう少し便利な方法は、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ パッケージ **mhchem** が導入されているときに使用することができる `\ce` コマンドを使用することです。`\ce` を数式に入力すると、新しい青いボックスが現れ、直感的に化学式を入力することができます。

コマンド	出力
<code>\ce{H2CO3}</code>	H_2CO_3
<code>\ce{SO4^2-}</code>	SO_4^{2-}
<code>\ce{(NH4)2S}</code>	$(\text{NH}_4)_2\text{S}$
<code>\ce{KCr(SO4)2.12H2O}</code>	$\text{KCr}(\text{SO}_4)_2 \cdot 12\text{H}_2\text{O}$
<code>\ce{A-B\!-\!C\equiv D}</code>	$\text{A}-\text{B}=\text{C}\equiv\text{D}$
<code>\ce{^227\downarrow_{90}\rightarrow Th+}</code>	$^{227}_{90}\text{Th}^+$
<code>\ce{\mu\text{-}Cl}</code>	$\mu\text{-Cl}$
<code>\ce{CO2 + C <=> 2CO}</code>	$\text{CO}_2 + \text{C} \rightleftharpoons 2\text{CO}$
<code>\ce{CO2 + C ->[\alpha][\beta] 2CO}</code>	$\text{CO}_2 + \text{C} \xrightarrow[\beta]{\alpha} 2\text{CO}$

【註】 $\text{L}^{\text{Y}}\text{X}$ では、**mhchem** の説明書通りに上矢印を得ようと `^` 文字を使用しても、うまく行きません。代わりに、次のように、負の小空白の後に `\uparrow` コマンドを続けます。 $\text{Fe} + 2\text{H}^+ \longrightarrow \text{Fe}^{2+} + \text{H}_2\uparrow$

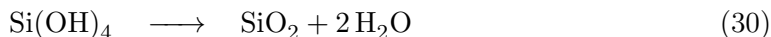
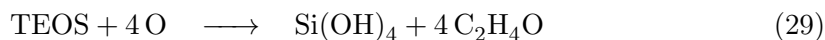
`\ce` を使うと (28) 式のコマンドは

`\ce{2Na+SO4^2- -> Na2SO4}`

のようになります。

⁴⁶フォント様式に関しては、第 11.1 節を参照のこと。

複数行の化学式を作るには、第 18 節に述べられている方法で、多行数式をまず作ります。その後、数式の小さな青いボックスそれぞれに `\ce` コマンドを使用します。(29) 式と (30) 式は、多段化学反応式の例で、一つの式毎に番号が振られています。



`mhchem` パッケージは、`\ce` の他に、特殊ケースに使用する `\cf` コマンドを提供しています。`\cf` の詳しい情報と例示については、`mhchem[7]` の取扱説明書をご覧ください。

21. 図解

$\text{L}_\text{A}\text{T}_\text{E}_\text{X}$ は、二つの型の可換図 `amscd` および `xymatrix` をサポートしており、以下でこれらの説明をします。

21.1. `amscd` 図解

この型の図解は、以下のように、関係を縦横の線や矢印で図示します。

$$\begin{array}{ccccc} A & \longrightarrow & B & \longrightarrow & C \\ & \uparrow & & & \downarrow \\ F & \longleftarrow & E & \longleftarrow & D \end{array}$$

これを作るには、数式に `\CD` コマンドを挿入します。二つの点線に囲まれた青枠が現れるので、ここにコマンドを入れていきます。`CommandReturn` を押すと、新しい行が作られます。水平方向の関係は奇数行に入れ、垂直方向の関係は偶数行に入れます。

関係を作るには、以下のコマンドがあります。

- `@<<<` は左矢印、`@>>>` は右矢印、`@=` は長い等号を生成します。
- `@AAA` は上矢印、`@VVV` は下矢印、`@|` は縦向きの等号を生成します。
- `@.` は関係が存在しない部分に置きます。

矢印はすべて、以下のようにラベル付けをすることができます。

- 文章を、第 1 と第 2 の「<」ないし「>」のあいだに入れると、この文章は矢印の上に表示されます。第 2・第 3 の「<」ないし「>」のあいだに入れると、矢印の下に表示されます。
- 縦矢印に付ける文章を、第 1・第 2 の「A」ないし「V」のあいだに入れると、この文章は矢印の左に表示されます。第 2・第 3 のもののあいだに入れると、矢印の右に表示されます。文章中に「A」や「V」の文字があるときには、これらは $\text{T}_\text{E}_\text{X}$ 括弧の中に入れなくてはなりません。

以下は、上記のすべての関係を使った例です。

$$\begin{array}{ccccccc} A & \xrightarrow{j} & B & \xrightarrow{k} & C & \xlongequal{\quad} & F \\ m \uparrow & & & & \downarrow V & & \parallel \\ D & \xleftarrow{j} & E & \xrightarrow{k} & F & \xlongequal{\quad} & C \end{array}$$

これを作るコマンドは、以下のとおりです。

```
\CD_A@>j>>B@>>k>C@=F Ctrl+Return
@AmAA@.@VV\{V→V@| Ctrl+Return
D@<<j<E@>k>>F@=C
```

21.2. xymatrix 図解

xymatrices を使うには、 \LaTeX パッケージの **xypic** が導入済みである必要があります。xymatrix は、数式中に `\xymatrix` コマンドを入れることで作ることができます。すると、通常の行列と同じようにして、列や行を付け加えることができます。第4節をご参照下さい。

amscd 図解とは異なり、xymatrices は、対角矢印や曲がった矢印など多様なサポートをしています。作ることのできる可換図と装飾は、ヘルプ▶用途別説明書▶XY-pic 説明書メニューにある *XY-pic 説明書* で詳しく網羅しています。

21.3. ファインマン図

ファインマン図を使うには、 \LaTeX パッケージ **feyn** を導入しておかなくてはなりません。すると、ファインマン図は、数式中で `\Diagram` コマンドを挿入すれば生成されます。通常の行列で行うのと同じようにして新規行や新規列を加えることができます(第4節参照)。

\LaTeX でのファインマン図の作り方は、メニューヘルプ▶用途別説明書内の『**ファインマン図**』にあります。

22. 自己定義コマンド

【註】自己定義コマンド名及びマクロ名には、ラテン文字しか使用することができません。

22.1. \newcommand コマンド

頻繁に用いるには、長すぎる \LaTeX コマンドはたくさんありますが、`\newcommand` コマンドを使えば、新しい短縮コマンドを定義することが可能です。

`\newcommand` コマンドの書式は、

```
\newcommand{新コマンド名}[引数の数][オプションの値]
{コマンド定義}
```

です。

【註】新コマンド名が、使用中の文書や呼び出している \LaTeX パッケージで、既に使用されていないことを確認して下さい。たとえば、`\Leftarrow` の短縮のつもりで `\le` というコマンドを定義したとすると、`\le` は既に「 \leq 」を表すコマンドとして定義されているので、エラーメッセージが表示されます。

「引数の数」は、0-9 の範囲の整数であり、新コマンドがいくつの引数をとるかを指定するものです。「オプションの値」では、非必須の引数の既定値を定義できます。これを指定すると、新コマンドの**最初の**引数は、自動的に非必須の引数になります。

以下にいくつかの例を挙げます。

- `\Longrightarrow` の短縮形として `\gr` というコマンドを定義するには、 \LaTeX プリアンブルに以下の行を加えます。

```
\newcommand{\gr}{\Longrightarrow}
```

- `\underline` の短縮形として `\us` というコマンドを定義するには、(下線を引くべき文字列を示す) 引数を考慮に入れなくてはなりません。このためには、以下のようなプリアンブル行を入れます。

```
\newcommand{\us}[1]{\underline{#1}}
```

「#」という文字は、引数の入る場所を示し、その後ろの「1」は、これが第1引数の入る場所であることを示します。

- `\framebox` の短縮形として、たとえば `\fb` というコマンドを定義するには、

```
\newcommand{\fb}[3]{\framebox#1#2{${\#3}$}}
```

二つのドルマークは、`\framebox` が必要とする内部の数式を作り出します。第9.1節をご参照下さい。

- ボックスの色を指定する必要がない `\fcolorbox` 用の新コマンドを作るには、以下のように、色を示す引数を非必須として定義します。

```
\newcommand{\cb}[3][white]{\fcolorbox{#2}{#1}{${\#3}$}}
```

`\cb` を使うときに色が指定されなければ、事前に定義された色である `white` が使用されます。

以下は、上で定義したコマンドの動作テストです。

コマンド	出力
$A \backslash \text{gr} B$	$A \Longrightarrow B$
$\backslash \text{us}\{ABcd\}$	\underline{ABcd}
$\backslash \text{fb}\{[2cm] \rightarrow \backslash \{ \int \int A=B\}$	$\int A = B$
$\backslash \text{cb}\{\text{red} \rightarrow \backslash \{ \int \int A=B\}$	$\int A = B$
$\backslash \text{cb}[\text{green}]\backslash \{\text{red} \rightarrow \backslash \{ \int \int A=B\}$	$\int A = B$

22.2. 数式マクロ

自己定義コマンドは、複雑な表現を使うときに特に便利です。たとえば、文書中で二次方程式を扱っているとすると、同じような解の形が何度も出てきます。二次方程式の一般型は、

$$0 = \lambda^2 + p\lambda + q$$

であり、その解の一般型は

$$\lambda_{1,2} = -\frac{p}{2} \pm \sqrt{\frac{p^2}{4} - q}$$

です。 $\lambda \cdot p \cdot q$ の3つのパラメータを指定することが必須であり、 λ の指数をオプションとして与えることができるような、解の公式のコマンドを定義するには、以下のような


L^AT_EX プリアンブル行を加えます.

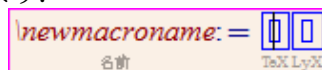
```
\newcommand{\qG}[4][1,\,2]{\#2_{\#1}=-\frac{\#3}{\#2}\pm
\sqrt{\frac{\#3^{\#2}}{\#4}-\#4}}
```




これを使って解の公式を作るには,

`\qG{\lambda\rightarrow\{p\rightarrow\{q` というコマンドを数式に入れます.

新コマンドを定義する方法は, たとえば L^AT_EX 中で分数は `\frac{分子}{分母}` の形で入れなくてはならないことなど, 使用するすべての L^AT_EX コマンドの書式を知っている必要があるのですが, 直感的ではありません. さらに, 定義中で中括弧を入れ忘れることはよくあり, それをやってしまうと, L_YX からは新コマンドが何をやらかしているか確認しにくくなってしまいます. これらの問題を回避するために, L_YX は, `\newcommand` コマンドの代わりに, 数式マクロを使う方法を提供しています.

数式マクロは, 挿入▷数式▷マクロメニューか, ツールバーボタンの  で作ることができます. すると, 数式マクロツールバーが表示されるとともに, マクロを定義した箇所に以下のようなボックスが現れます.



`\newmacroname` が既定のマクロ名として現れますが, 意味のある名称に変更するべきでしょう. 欲しい数式は一つ目の青枠の中に入れます. 引数を置く場所は, `\#1` のように `\#引数番号` というコマンドで入力するか, マクロツールバーボタンの  を使用します. 引数位置は赤で表示されます. 引数は, 最大で9つまでとることができます. 非必須引数は, ツールバーボタンの  で作ることができます. 最初の必須引数は, ツールバーボタンの  を使って, 非必須引数にすることができます. 二つ目の青枠には, L_YX 中でのマクロの表示のしかたを定義することができます. 通常は, 定義したとおりに表示された方が便利なので, この枠は空白にしておきます. しかし, 画面の過半を占拠してしまうようなマクロを作ってしまった場合には, たとえばこの枠に

qG: `\#1`, `\#2`, `\#3`, `\#4`

のように入れることができます. このようにすると, マクロ名と引数のみが L_YX 上に表示され, 見通しが良くなります. 一方, 出力での数式は, 最初の枠で定義したように表示されます.

さらに, 数式中のマクロ表示は, マクロの中にカーソルを置いて, 表示▷数式マクロを展開 (畳む) メニューを使うことで, マクロ毎に変えることができます.

マクロを使うには, 数式中にマクロ名をコマンドとして入れます. 上記の例では, `\qG` とします. このマクロは, L_YX 中では以下のように表示されます.

以下は, 上記の例に, 引数 $x \cdot \ln(x) \cdot B$ を指定したものです.

$$x_{1,2} = -\frac{\ln(x)}{2} \pm \sqrt{\frac{\ln(x)^2}{4} - B}$$

L_YX は, ツール▷設定▷編集▷制御メニューで, マクロを編集するのに複数の様式を用意しています. あなたに最も合った様式を見つけるには, 様式を選択してから, 違いを見るために数式マクロにカーソルを合わせてみてください.

数式マクロは, 文書書き出し時に, 内部的に `\newcommand` コマンドに変換されます. こうして生成された `\newcommand` コマンドは, L^AT_EX プリアンブルには置かれませ

るので、マクロは、文書中、マクロ定義ボックスよりも後の数式でのみ使うことができます。

数式マクロは、`\newcommand` コマンドから直接作ることもできます。たとえば、 LyX 中に通常の文章として、

`\newcommand{\larrow}[2]{\xleftarrow[#2]{#1}}`










というコマンドを書き入れ、この全体を選択して、ショートカット 捷 径 `Ctrl+M` を押すと、このコマンドは数式マクロに変換されます。この方法を使うに当たっては、`\newcommand` コマンドが正しく入力されていることに気をつけなくてはなりません。さもないと、間違ったマクロが作られてしまって、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ エラーが発生します。

数式マクロには、まだ、マクロ定義中に再帰的に数式を入れてしまうと、正しく処理されないという問題が残っています。したがって、第 22.1 節で例として作った `\fb` は、マクロとしては作ることができません。

カーソルがマクロ定義ボックスの中にあるとき、 LyX 中に以下のようなマクロツールバーが表示されます。



マクロツールバーは、左から右に、以下の各ボタンがあります。

-  編集▷数式▷マクロ定義▷最後の引数を削除
-  編集▷数式▷マクロ定義▷引数を追加
-  編集▷数式▷マクロ定義▷最初の必須引数を
非必須引数にする
-  編集▷数式▷マクロ定義▷最後の非必須引数を
必須引数にする
-  編集▷数式▷マクロ定義▷非必須引数を削除
-  編集▷数式▷マクロ定義▷非必須引数を挿入
-  編集▷数式▷マクロ定義▷右に吐き出す形で
最後の引数を削除
-  編集▷数式▷マクロ定義▷右から喰う形で
引数を追加
-  編集▷数式▷マクロ定義▷右から喰う形で
非必須引数を追加

23. コンピュータ代数システム

LyX では、 LyX の数式エディタで書かれた数式表現と、外部プログラムやユーザ定義スクリプトとの間のやり取りが可能です。現在サポートされているプログラムは、 Maple ・ Mathematica ・ Maxima ・ Octave です。サポートされている数式表現は、非常に単純なものに限られていますので、ご注意ください。

23.1. 使用法

数式表現を書いた後、編集▷数式▷コンピュータ代数システムを使用メニューで、コンピュータに導入済みのプログラムを選択します。その後、計算結果が、数式表現の右に「=」演算子とともに表示されます。

下記は、どのような計算が可能かをいくつか例示したものです (出力は Maxima です)。

- $\frac{37}{3} * 2 - \sum_{i=1}^3 i^i = -\frac{22}{3}$
- $\frac{37.0}{3} = 12.333333333333333$
- $\int_1^2 \sin(x) dx = \cos 1 - \cos 2$
- $\int \left(\frac{1}{1+x^3} \right) dx = -\frac{\log(x^2-x+1)}{6} + \frac{\arctan\left(\frac{2x-1}{\sqrt{3}}\right)}{\sqrt{3}} + \frac{\log(x+1)}{3}$
【註】単純な「(」 「)」文字ではなく、適切な区切り差込棒 () を使わなくてはなりません。
- $\det \begin{bmatrix} 1 & 6 & 7 \\ 2 & 5 & 8 \\ 3 & 4 & 17 \end{bmatrix} = -56$
- $\lim_{x \rightarrow 0} \left(\frac{\sin(x)}{x} \right) = 1$
- $\text{powerseries}(-\log(5-x), x, 1) = \sum_{i_2=0}^{\infty} \frac{4^{-i_2-1} (x-1)^{i_2+1}}{i_2+1} - \log 4$
- $\text{solve}(x_1 + y_1^3 = y_1 + x_1^2, x_1) = \left[x_1 = -\frac{\sqrt{4y_1^3-4y_1+1}-1}{2}, x_1 = \frac{\sqrt{4y_1^3-4y_1+1}+1}{2} \right]$

23.2. 捷徑

現在のところ、代数プログラムを呼び出すための ショートカット 捷徑はありません。しかし、捷徑を `math-extern LATEX` 関数にバインドすることは容易です (取扱説明書『`LATEX` 関数篇』参照)。例えば、数式エディタ中でリターン^{キー}鍵をバインドさせることもできます。それを Maxima にバインドするには、下記の捷徑定義コマンドを使用します。

`command-alternatives paragraph-break;math-extern maxima`

24. 補遺

24.1. 負の数

数式中の負の数は、数の前の負符号が、差演算子記号と同じ長さに設定されてしまうために、汚く見えてしまうことがあります。負の数を通常の文章として書くと、負符号は正しく表示されます。

したがって、この問題は、負符号を数式テキストに変換することによって、解消されます。以下は、この問題を示す例です。

通常の文章：	$x = -2$
数式：	$x = -2$
解決策：	$x = -2$

24.2. 位区切りとしてのコンマ

L^AT_EX では、英語の慣習にしたがい、数式中のコンマを数字の位区切りに使用します。よって、数式中のコンマの後ろには、つねに空白が加わります。

これを回避するためには、コンマを選択して、数式テキストに変更して下さい (ショートカット 捷 径 : CommandM)。

文書中の数式コンマを、すべて小数点として使うには、L^AT_EX プリアンブルに

```
\usepackage{icomma}
```

という行を加えて、icomma.sty⁴⁷ ファイルを読み込みます。

24.3. 物理ベクトル

L^AT_EX パッケージ braket⁴⁸には、定義済みのベクトルが提供されており、

```
\usepackage{braket}
```

という L^AT_EX プリアンブル行で読み込むことができます。

以下のコマンドが定義されています。

コマンド	出力
<code>\Bra{\psi}</code>	$\langle\psi $
<code>\Ket{\psi}</code>	$ \psi\rangle$
<code>\Braket{\psi \phi}</code>	$\langle\psi \phi\rangle$

`\Braket` コマンドを使うと、以下のように、すべての縦棒がそれを囲む括弧と同じ大きさに設定されます。

$$\left\langle\phi\left|J=\frac{3}{2},M_J\right.\right\rangle$$

`\Braket` と同じ効果は、第 5.1.2 節に説明されているとおり、`\middle` コマンドを用いることによっても実現できます。

24.4. 自己定義の分数

分数用の自己定義コマンドを定義するには、以下の書式を持つ `\genfrac` コマンドを使います。

```
\genfrac{左括弧}{右括弧}{分数線の厚み}{様式}
          {分子}{分母}
```

ここで「様式」は、0-3 の範囲の数字です。

⁴⁷icomma は、L^AT_EX パッケージ was に含まれています。

⁴⁸braket は標準的 L^AT_EX 頒布版のすべてに含まれています。

数字	様式 (大きさ)
0	別行立て様式の数式
1	行内数式
2	やや小 (small)
3	最小 (tiny)

「様式」を指定しないときには、`\frac` コマンドのように、大きさは周囲の環境に合わせて調節されます。

「分数線の厚み」を指定しないときには、既定値である 0.4pt が用いられます。

たとえば、第 3.2 節の `\dfrac` コマンドおよび `\tbinom` コマンドは、

```
\newcommand{\dfrac}[2]{\genfrac{}{}{0}{#1}{#2}}
```

あるいは

```
\newcommand{\tbinom}[2]{\genfrac{({}{)}{0pt}{1}{#1}{#2}}
```

というコマンドで定義できます。


分数線の厚みを非必須の引数として与えることのできる分数を定義するには、 \LaTeX プリアンブルに

```
\newcommand{\fracS}[3]{\genfrac{}{}{#1}{#2}{#3}}
```

という行を入れます。

以下は、そのテストです。

コマンド	<code>\fracS[1mm]\{A \rightarrow \{B</code>	<code>\fracS[5mm]\{A \rightarrow \{B</code>
		A

出力	$\frac{A}{B}$	
		B

ご覧になってわかるように、分子や分母から分数線までの距離は、分数線の厚みの約 3 倍になります。

24.5. 数式の消去

数式の全体もしくは一部を消去するには、4つの方法があります。

コマンド	出力
<code>\cancel_\int A=B</code>	$\int A=\cancel{B}$
<code>\bcancel_\int A=B</code>	$\int A=\cancel{B}$
<code>\xcancel_\int A=B</code>	$\int A=\cancel{B}$
<code>\cancelto_\int A=B\rightarrow 1</code>	$\int A=\cancel{B}^1$

`\cancelto` は、以下のように、とくに数式中の分数の約分を表示するのに適しています。

$$\frac{(x_0 + bB)^2}{(1 + b^2)^2} = \frac{x_0^2 + B^2 - r_g^2}{1 + b^2}$$

`\cancelto` の「下付き文字」寸法の大きさを変えるには、

`\PassOptionsToPackage{オプション}{cancel}`

という行を \LaTeX プリアンブルに書き加えてください。ここで、**オプション**を **samesize** にすると、取り消した部分と同じ寸法になり、**Smaller** にすると、通常よりも少し小さくなります。

取り消し線に色を付けるには、 \TeX コードで以下のコマンドを書き加えてください。

`\renewcommand{\CancelColor}{\color{red}}`

ここで **red** は、お好みの色に変更することができます。

$$\frac{(x_0 + bB)^2}{(1 + b^2)^2} = \frac{x_0^2 + B^2 - r_g^2}{1 + b^2}$$

他の微調整については、**cancel** パッケージの取扱説明書 [5] をご覧ください。

24.6. 節見出し中の数式

数式を節見出し中で使う際には、以下のことに留意しなくてはなりません。

文書設定ダイアログの PDF 特性で **hyperref** サポートが有効になっている場合、PDF のしおりが、目次にある節見出しすべてに関して生成されます。しおり中に数式を入れることは PDF の慣習に違反しているため、節見出しに数式が含まれている場合、数式はしおり中に誤った文字列として表示されます。

これらの問題は、挿入▷短縮タイトルメニューを使って、問題となる節見出しの最後に短縮タイトルを入れることで解決することができます。短縮タイトルは、目次が美しく整形されるように、多行にわたる節見出しに別名を付けるものです。目次中には、短縮タイトルのみが表示され、したがって PDF しおり中にも短縮タイトルのみが表示されます。

数式を目次中でも使わなくてもならないが、**hyperref** も使用しなくてはならないときには、

`\texorpdfstring{部分}{代替文字列}`

というコマンドを $\text{T}_{\text{E}}\text{X}$ モードで使う方法があります.

「部分」は、見出し中、PDF しおりに表示したくない部分です. これは、文字・数式・脚註のほかに相互参照をとることもできます. しおりには、この部分の代わりに、「代替文字列」が用いられます.

以下の二つは、見出しの例です.

24.6.1. 目次中では数式を使わない見出し $\sqrt{-1} = i$

24.6.2. 目次中で数式を使う見出し $\sqrt{-1} = i$

一つめの見出しでは短縮タイトルが使われており、二つめの見出しでは `\texorpdfstring` が使われています.

他の節見出しと同じ書式を得るために、上の見出し全体は `boldmath` 環境に設定してあります⁴⁹.

24.7. 多段組文中の数式

多段組文中に数式を作ると、段の中に収まりきらないことも多く、ページ幅全体に広がるようにする必要があることがあります. これは、`multicol`⁵⁰ $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ パッケージを、

```
\usepackage{multicol}
```

という $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ プリアンブル行を書いて読み込むことで、実現できます.

ここで、文書▷設定メニューの本文レイアウトで、二段組文書の設定を有効にしてはならないことに注意してください.

多段組文の前に

```
\begin{multicols}{段数}
```

というコマンドを $\text{T}_{\text{E}}\text{X}$ モードで書き入れます. 「段数」は、2-10 のあいだの数字です. 多段組文の終わる数式の前には、

```
\end{multicols}
```

というコマンドを $\text{T}_{\text{E}}\text{X}$ モードで入れます.

このコマンドによって、数式の前にいくらかの余白が、自動的に作られます. これをなくすには、数式の前に-6 mm の垂直空白を入れて下さい. 数式様式⁵¹として行頭下げを使用している場合には、代わりに-9 mm の垂直空白を入れて下さい.

以下は、別行立て数式を含む、多段組文の例です.

Das Spektrum wird fouriertransformiert. die Phasenverschiebung bestimmen kann-
Die Fouriertransformation wird verwendet, ten, interessiert uns nun das Aussehen
um die überlagerten Signale (Netzwerk, des Ausgangssignals. Im Experiment ha-
Lösungsmittel) zu trennen. Nachdem wir ben wir es mit sehr vielen Teilchen zu tun,

⁴⁹第 11.2 節参照.

⁵⁰`multicol` は、標準的 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 頒布版のすべてに含まれています.

⁵¹数式様式に関しては、第 17 節をご覧ください.

so das man über alle Phasen integrieren muss. Sei nun S unser normiertes Ausgangssignal und P die Phasenverteilungsfunktion, so ergibt sich die Beziehung

$$S(t) = S_0(t) \int_{-\infty}^{\infty} P(\phi, t) e^{i\phi} d\phi \quad (31)$$

wobei S_0 das Signal ohne Gradient beginnt sich die Magnetisierung zu entwickeln ist und die Normierungsbedingung $\int_{-\infty}^{\infty} P(\phi, t) d\phi = 1$ gilt. Nun dürfen wir fokussieren, wodurch sich das Signal zusätzlich abschwächt. Diese Abschwächung aber nicht den Relaxationsprozess außer verläuft exponentiell in Abhängigkeit der Acht lassen. Direkt nach dem $\pi/2$ -rf-Puls so genannten T_2 -Zeit.

24.8. 変数の説明付き数式

(32) 式のように, 数式内で変数の説明をするには, n 個の変数が使われている場合, 左寄せの列を持つ $2 \times n$ 行列を使用します⁵². 説明を小さな文字にするには, 行列の前に, たとえば `\footnotesize` コマンドを挿入します⁵³.

数式様式に **行頭下げ**⁵⁴を使っている場合, 行列を数式とページ余白から等距離に置くために, 行列の前後に `\hfill`⁵⁵を入れます.

数式様式に **中央揃え**を使っている場合, 数式を字下げするには, 第 18.2.3 節で述べた方法を使用します. (32) 式には 5 列があり, 最初の 2 列には数式, 3 列めには行列, 最終列には空の $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 括弧が入っています.

$$F_A = \rho \cdot V \cdot g \quad \begin{array}{ll} \rho & \text{density} \\ V & \text{volume} \\ g & \text{gravitational acceleration} \end{array} \quad (32)$$

24.9. アップライト体のギリシャ小文字

ほとんどの数式書体は, イタリック体のギリシャ小文字しか提供していません. しかし, π 中間子やニュートリノのような素粒子の記号には, アップライト体のギリシャ文字が必要とされます. `upgreek.sty`⁵⁶ ファイルを

`\usepackage{upgreek}`

という $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ プリアンブル行で読み込めば, これらが提供されるようになります. アップライト体のギリシャ小文字は, ギリシャ小文字のコマンド名の前に **up** を付けると作ることができます. たとえば `\uptau` コマンドは, τ のようになります.

これらのコマンドを使えば, 以下のような素粒子の反応を組版することができるようになります.

$$\pi^+ \rightarrow \mu^+ + \nu_\mu$$

アップライト体の文字は, イタリック体のものよりも太く幅広です. したがって, これらを「 $\mu\mathrm{m}$ 」のような単位に使うべきではありません.

⁵²行列に関しては, 第 4 節参照.

⁵³フォント寸法に関しては, 第 11.4 節参照.

⁵⁴数式様式に関しては, 第 17 節参照.

⁵⁵`\hfill` は, **行頭下げ**様式のときのみ機能します. 第 8.2 節をご覧ください.

⁵⁶`upgreek` は, **was** $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ パッケージの一部です.

24.10. 数式中のテキスト文字

折にふれて、テキスト文字を直接数式中に入れたいときがあるでしょう。たとえば、中黒「・」を $\nu = 5 \cdot 10^5 \text{ Hz}$ のように数式中で頻繁に用いようとするとき、この中黒はすべてのエンコーディングでテキスト文字として定義されているために、代わりに `\cdot`⁵⁷ コマンドを挿入しなくてはならなくなることでしょう。しかし、

```
\DeclareInputtext{183}{\ifmmode\cdot\else\textperiodcentered\fi}
```

という L^AT_EX プリアンブル行を使えば、エンコーディングに変更を加えることができます。

文字エンコーディング (文書▷設定▷言語メニュー) は、キーボード上の^{キー}鍵が押されたときにどの文字が表示されるかを指定します。「・」文字に対応する鍵が押されると、内部的には `\textperiodcentered` コマンドが使用されます。しかし、このコマンドは数式中では使えないので、L^AT_EX エラーが発生するのです。変更後のエンコーディングでは、文字が数式中に挿入されたか否かによって、正しいコマンドが自動的に選択されます。

定義ファイル中には、複数の文字のエンコーディングが保管されています。たとえば、**latin9** エンコーディングは、L^AT_EX がインストールされたフォルダにある **latin9.def** ファイルに定義されています。エンコーディングは、L^AT_EX プリアンブルで変更するべきであって、定義ファイルを変更してはなりません。さもないと、自分の作成した文書は、他のコンピューターで作業をしている他のユーザーによっては編集することができなくなってしまう。

中黒の他にこの文書では、角度記号「°」が、数式に直接入れることができるよう、以下のような L^AT_EX プリアンブル行で定義されています⁵⁸。

```
\DeclareInputtext{176}{\ifmmode^\circ\else\textdegree\fi}
```

24.11. 数式中の L^AT_EX コメント

本文中では、ファイルの L^AT_EX ソースコードでのみ見ることのできるコメントを、挿入▷註釈▷コメントメニューで挿入することができます。これと同じことは、数式中ではできませんが、

```
\%
```

というコマンドを使えば、L^AT_EX コメントを挿入することができます。このコマンドを打ち込むと、コメントを書き込むことのできるボックスが生成されます。L^AT_EX のバグのせいで⁵⁹、コメント中に、通常の本文や数式内本文そのものを書くことはできませんが、L^AT_EX 出力中に、L^AT_EX コマンドの形では見ることができます。

下記は、L^AT_EX コメントを持つ数式の例です。

$$A = B$$

⁵⁷第 10.4 節参照。

⁵⁸【訳註】pL^AT_EX では、これらの定義は必要ないので、コメントアウトして無効にしています。

⁵⁹[LyX-bug #9002](#)

A. 組版上の助言

この節は、ISO 規範に掲げてある、もっとも重要な組版ルールの要約です⁶⁰。

- 物理単位は、つねに (イタリック文中にあるときも) アップライト体にします⁶¹：
30 km/h
値と単位の間には、最小空白を入れます。第 8.1 節を参照。
この慣習は、`\unittwo` コマンドを使用すると、つねに満たされます。このコマンドを数式に入れると、二つの枠が現れます。最初の枠には値をいれ、第二の枠に単位を入れると、上記と同じような結果が得られます：30 km/h。実は、`\unittwo` は、 \LaTeX コマンドの実体ではなく、`\unit[値]{単位}` というコマンドです。したがって、これを \TeX コード中で使用することはできません。
- 百分率記号と千分率記号は、物理単位と同様に組みます：
血中アルコール 1,2 ‰
- 角度記号は値の直後に置きます：15°。しかし、単位として用いられるときは別です：15 °C
- 4桁以上の数は、3桁ごとに最小空白を直前に挿入して、グループ化します：18 473 588
- 120 × 90 × 40 cm のような寸法には、積記号「×」を用います。これは、`\times` コマンドか、挿入▷特殊文字▷記号メニューから入れることができます。
- いくつかの文字を含む関数名は、混乱を防ぐためにアップライト体にします。第 15.1 節を参照。
- 複数の文字を含む指数は、アップライト体にします： E_{kin}
行列要素はイタリック体にします： \hat{H}_{kl}
- 微分作用素・積分作用素「d」、オイラー数「e」、虚数単位「i」は、他の変数と間違えることを避けるために、アップライト体にします。
- フーリエ変換を表す文字は、`\mathscr{F}` コマンドか挿入▷特殊文字▷記号▷文字様記号メニューの \mathscr{F} で入れることができます。

⁶⁰ この要約の一部は、ISO 規則を取り上げている「Duden」[8] と呼ばれるドイツの半公的辞書から採られています。

⁶¹ 書体様式で指定します。第 11.1 節を参照。

B. 同義語

いくつかの文字や記号は、複数のコマンドから作ることができます。以下は、同義のコマンドの一覧です。

コマンド	同義のコマンド
<code>\ast</code>	<code>*</code>
<code>\choose</code>	<code>\binom</code>
<code>\geq</code>	<code>\ge</code>
<code>\lbrace</code>	<code>{</code>
<code>\lbracket</code>	<code>[</code>
<code>\leftarrow</code>	<code>\gets</code>
<code>\leq</code>	<code>\le</code>
<code>\lor</code>	<code>\vee</code>
<code>\neq</code>	<code>\not=</code>
<code>\slash</code>	<code>/</code>
<code>\vert</code>	<code> </code>

コマンド	同義のコマンド
<code>\backslash</code>	<code>\\</code>
<code>\dasharrow</code>	<code>\dashrightarrow</code>
<code>\land</code>	<code>\wedge</code>
<code>\rbrace</code>	<code>}</code>
<code>\rbracket</code>	<code>]</code>
<code>\rightarrow</code>	<code>\to</code>
<code>\lnot</code>	<code>\neg</code>
<code>\ne</code>	<code>\not=</code>
<code>\owns</code>	<code>\ni</code>
<code>\square</code>	<code>\Box</code>
<code>\Vert</code>	<code> </code>

参考文献

- [1] F. MITTELBACH; M. GOOSSENS: *The L^AT_EX Companion*, 2nd ed. Addison Wesley, 2004
- [2] L^AT_EX の数式能力の[説明](#)
- [3] $\mathcal{A}\mathcal{M}\mathcal{S}$ -L^AT_EX の[説明](#)
- [4] L^AT_EX パッケージで利用できる記号の[全覧](#)
- [5] L^AT_EX **cancel** パッケージの[取扱説明書](#)
- [6] L^AT_EX **hyperref** パッケージの[取扱説明書](#)
- [7] L^AT_EX **mhchem** パッケージの[取扱説明書](#)
- [8] *Duden Band 1*. 22. Auflage, Dudenverlag, 2000
- [9] 原稿見直しの[チェックリスト](#)

索引

€, 35

L^AT_EX コメント, 65

L^AT_EX プリアンブル, 2

T_EX 括弧, 2

T_EX モード, 1

アイソトープ, *see* 同位体

アクセント, 16

一文字に付ける, 17

複数の文字に付ける, 18

文章中の, 39

上付き文字, *see* 指数

ウムラウト, 17

埋め草, 7

演算子, 25

自己定義 ——, 29

修飾, 28

大 ——, 25

二項 ——, 29

範囲, 26

化学記号

記号, 53

同位体, 7

化学式, 53

数

負の, 59

括弧, 11

垂直 ——, 11

水平 ——, 13

多行数式用, 43

括弧高

自動, 12

手動, 11

関係子, 35

関数

自己定義, 37

剰余, 38

定義済み, 36

モジュラス, *see* 剰余

カンマ, *see* コンマ

記号, 34

化学, 53

数学, 34

その他, 35

ユーロ通貨記号, 35

行列, 10

極限, 37

ギリシャ文字, 33

アップライト体, 64

大文字, 33

小文字, 33

ボールド体, 34

空白

行内数式周り, 20

水平 ——, 18

可変長, 19

定義済み, 18

区分記号, 11

組版上の助言, 66

古式数字, 40

コマンド

記号

\@addtoreset, 52

\@removefromreset, 52

A

\Alph, 50

\adjustlimits, 28

\aligned, 47

\alignedat, 47

\alph, 50

\arabic, 51, 53

\arraycolsep, 10, 42

\arraystretch, 11

B

\big, 11

\bigl - \bigr, 12

\bigm, 12

\binom, 6

\boldmath, 31

\boldsymbol, 34

\boxed, 20

\brace, 6

\brack, 6

C

\cases, 6

\backslash CD, 54	\backslash mbox, 22
\backslash cdots, 8	\backslash middle, 13
\backslash ce, 53	\backslash multlinegap, 46
\backslash cf, 54	N
\backslash cfrac, 5	\backslash newcommand, 55
\backslash colorbox, 22	\backslash nicefrac, 5
D	\backslash not, 7
\backslash DeclareMathOperator, 29, 37	\backslash numberwithin, 52
\backslash dbinom, 6	O
\backslash definecolor, 23	\backslash officialeuro, 35
\backslash dfrac, 4	\backslash oldstylenums, 40
\backslash displaystyle, 1, 32	\backslash overbrace, 13
\backslash dotfill, 9	\backslash overbracket, 13
\backslash dots, 8	\backslash overline, 8
E	\backslash overset, 18, 28
\backslash euro, 35	P
F	\backslash parbox, 24
\backslash fbox, 20	\backslash phantom, 7
\backslash colorbox, 23	\backslash prod, 26
\backslash frac, 4	R
\backslash framebox, 21	\backslash Roman, 49
G	\backslash raisebox, 22
\backslash gathered, 47	\backslash renewcommand, 11, 49
\backslash genfrac, 60	\backslash right, 10, 12, 44
H	\backslash roman, 49
\backslash hdotsfor, 9	\backslash root, 5
\backslash hfill, 19	\backslash rule, 8
\backslash hphantom, 7	S
\backslash hrulefill, 9	\backslash setlength, 20
\backslash hspace, 19, 42	\backslash shortintertext, 47
I	\backslash shoveleft, 46
\backslash int, 26	\backslash shoveright, 46
\backslash intertext, 47	\backslash sideset, 28
J	\backslash smallmatrix, 11
\backslash jot, 14, 41	\backslash smashoperator, 27
L	\backslash split, 47
\backslash ldots, 8	\backslash splitfrac, 43
\backslash left, 10, 12, 44	\backslash sqrt, 5
\backslash lefteqn, 42	\backslash stackrel, 36
\backslash leftroot, 6	\backslash subarray, 27
\backslash lim, 37	\backslash substack, 27
\backslash linewidth, 24	\backslash sum, 26
M	T
\backslash makebox, 22	\backslash tag, 51
\backslash mathbin, 30	\backslash tbinom, 6
\backslash mathop, 30	\backslash texorpdfstring, 62
\backslash mathscr, 66	\backslash text, 2
\backslash mathsurround, 20	\backslash textbackslash, 11

- `\textcircled`, 39
- `\textcolor`, 24, 31
- `\tfrac`, 4
- U
 - `\unboldmath`, 31
 - `\underbrace`, 13
 - `\underbracket`, 13
 - `\underline`, 8
 - `\underset`, 18, 28
 - `\unitfrac`, 5
 - `\uproot`, 6
- V
 - `\vphantom`, 7, 44
- X
 - `\xleftarrow`, 15
 - `\xrightarrow`, 15

根号, 5
コンマ, 60

自己定義コマンド, 55
`\newcommand`, 55
数式マクロ, 56

指数, 4
下付き文字, *see* 添字
省略符号, 8
書体, 30
寸法, 32
様式, 30

数式
色付き, 31
下線, 8
行内, 1
消去, 61
節見出し中の, 62
多行 —, 40
alignat 環境, 44
align 環境, 44
eqnarray 環境, 45
flalign 環境, 45
gather 環境, 45
multpline 環境, 46
行間, 41
数式の一部, 46
テキスト, 47
列間, 42
多段組文中の, 63
長い, 42

番号, *see* 数式番号
別行立て様式, 1
変数の説明付き, 64
ボード体, 31
様式, 40
数式テキスト, 2
数式番号, 48
細目番号, 48
自己定義, 51
自己定義区切り, 51
文字を使った, 49
ローマ数字を使った, 49
図解
amscd, 54
xymatrix, 55
ファインマン, 55

積分記号, 25

相互参照
数式への, 48
添字, 4

チルダ, 39

テキスト
色付き, 24
数式中の, 2, 47, 65

同位体, *see* 化学記号
同義語, 67
特殊文字, 38

二項係数, 6

場合分け, 6

パッケージ
braket, 60
calc, 24, 41
cancel, 62, 68
carlisle, 52
color, 22
eurosym, 35
fixmath, 34
hyperref, 62, 68
icomma, 60
mathtools, 13, 15, 27, 28, 43, 47
mhchem, 53, 68
multicol, 63
remreset, 52

upgreek, 17, 64
was, 34, 60, 64

比較子, *see* 関係子
否定, 7

フォント, *see* 書体
分数, 4
 自己定義の, 60
 多行にわたる, 43

ベクトル, 17
 物理——, 60

補遺, 59
棒線, *see* 横線
ボックス, 20
 色付き——, 22
 段落——, 24
 枠付き——, 20
 枠なし——, 22

マクロ, 56
 ツールバー, 58

矢印, 14
 垂直, 16
 水平, 15
 対角, 16
 ラベル付き, 15

横線, 8

ルート, *see* 根号

和, 25
枠, *see* ボックス