`diffcoeff`

a LaTeX package to ease the writing of differential coefficients

Version 5.3

Andrew Parsloe
(ajparsloe@gmail.com)

April 12, 2023

Abstract

`diffcoeff` is a LaTeX package to ease the writing of ordinary, partial and other derivatives of arbitrary algebraic or numeric order. For mixed partial derivatives, the total order of differentiation is calculated by the package. Optional arguments allow for points of evaluation (ordinary derivatives), or variables held constant (partial derivatives), and the placement of the differentiand in numerator or appended. Besides $\frac{\mathrm{d}y}{\mathrm{d}x}$, forms like $\mathrm{d}y/\mathrm{d}x$ and $\partial_x y$ are also available, as well as derivatives built from D, Δ, δ, and configurable jacobians and differentials. Other notations like line elements ($\mathrm{d}x^2 + \mathrm{d}y^2 + \mathrm{d}z^2$) and bra-kets are easily produced.

Version 5 of `diffcoeff` more fully embraces the configurability offered by `xtemplate` than earlier versions. Some incompatibilities have arisen, but all is not lost: version 4 is still available with the command

`\usepackage[<options>]{diffcoeff}[=v4]`

For users of version 4

The `\diff` and `\diffp` commands of version 4 of `diffcoeff` remain, but lack the 'spacing switch' `!` (which on reflection was a mistake) and, more noticeably, the 'slash switch' `/`. Slash-fraction derivatives are now created with the `\difs` and `\difsp` commands. New commands `\difc` and `\difcp` produce derivatives in 'compact notation' like $\mathrm{d}_x y$ and $\partial_x y$. The order-override option (for mixed partial derivatives) has been changed to use angle brackets (for clarity) or a command `\difoverride`. The sequential order of differentiand and variable(s) of differentiation can now be reversed, when the differentiand is appended, by using a second star, e.g., `\diffp**`. The two-argument `\diffdef` command of earlier versions has been replaced by the three-argument command `\difdef`, the additional argument determining which one or more of the `f`, `s`, `c`, `fp`, `sp` or `cp` forms the defined variant applies to. The differential command `\dl` has been rewritten and is now fully template-configurable (allowing easy writing of line elements like $\mathrm{d}x^2 + \mathrm{d}y^2 + \mathrm{d}z^2$), and the jacobian command `\jacob` is also configurable. Indeed version 5 of `diffcoeff` more fully embraces the configurability offered by the `xtemplate` package than previous versions, bringing other notations – like those of the `braket` package – within its compass.

ISO defaults

The `ISO` package option is redundant. Unlike the default set-up in version 4, the defaults in version 5 of `diffcoeff` are chosen to reflect ISO recommendations (see the standard ISO 80000-2). In particular this means upright 'd's and subscripted parentheses enclosing a derivative to indicate a point of evaluation. This document is written with those defaults. For those (like the author) who prefer math-italic 'd's and a subscripted vertical rule to indicate a point of evaluation, the means of creating 'variant forms' or changing the defaults is readily available; see §§3.3, 3.4.

Contents

Chapter 1

Introduction

`diffcoeff.sty` is written in the `expl3` language of LaTeX3, now part of standard LaTeX since February 2020. A LaTeX distribution from or later than that date is assumed. The package also requires the packages `xtemplate` (part of the `l3packages` bundle) and `mleftright`. The package is loaded in the usual way by entering

```
\usepackage{diffcoeff}
```

in the preamble of your document or, if package options are being used,[1]

```
\usepackage[<options>]{diffcoeff}
```

The interface of `diffcoeff` with version 5 has changed from earlier versions. If you want the familiar behaviour of version 4, add to these commands a trailing optional argument like this,

```
\usepackage[<options>]{diffcoeff}[=v4]
```

(with no space after the '=' sign!). Of course you will not get the new features of version 5. Working with version 4 is described in the document `diffcoeff4.pdf`.

1.1 Package options

There are four package options, which are entered in a comma-separated list in the optional argument of the `\usepackage` command. The *second* and *fourth* are new with version 5 of `diffcoeff`.

1. The first package option is the `spaced` option which takes three values:
   (a) `spaced=1` inserts a small space before the differentiand; this is the default so that entering `spaced` is equivalent to `spaced=1`;

[1] Angle brackets indicate possible user input (*without* the angle brackets).

(b) `spaced=0` inserts no space before the differentiand; `diffcoeff` is initialized to `spaced=0` so that if the spaced option is not used `spaced=0` is assumed;

(c) `spaced=-1` inserts a small space before the differentiand *if it contains more than one token*, and no space otherwise. The present document uses `spaced=-1`.

This option is discussed in §2.2.1.

2. For the second package option, by entering `mleftright` in the options list the command `\mleftright` is automatically inserted in the preamble. The effect is to change all occurrences of `\left`, `\right` in the document to `\mleft`, `\mright` so that the spacing around scalable delimiters modified by `\left`, `\right` is reduced; see the box below and §2.2.4.3. The present document does not use this package option.

3. The third package option requires the `<filename>` of a file with extension `.def`, `<filename>.def`, containing definitions of variant forms of derivative (see §3.3):

```
\usepackage[def-file=<filename>]{diffcoeff}
```

This is discussed in §3.3.1. The present document uses the package option `def-file=diffcoeff5`.

4. The fourth package option `DIF` is a comma list of *key=value* statements amending the built-in defaults for the 'grandparent' template `DIF`; see §3.4.1. The present document does not use this package option.

To see the effect of the `mleftright` package, consider the expression

```
\[ \ln \left(\frac xy\right),\quad\sin\left(x^2\right). \] ⟹
```

$$\ln\left(\frac{x}{y}\right),\quad \sin\left(x^2\right).$$

in which there is significant whitespace before and after the parentheses. The package `mleftright` enables this whitespace to be reduced by using `\mleft`, `\mright` in place of `\left`, `\right`:

```
\[ \ln \mleft(\frac xy\mright),\quad\sin\mleft(x^2\mright). \] ⟹
```

$$\ln\left(\frac{x}{y}\right),\quad \sin\left(x^2\right).$$

If you put `\mleftright` in the preamble, which is what the `mleftright` package option does, then all occurrences of `\left`, `\right` in the document will be affected. `\left`, `\right` can be restored to their normal behaviour by the command `\mleftrightrestore`. Rather than use `\mleft`, `\mright` explicitly, as in the example, the same effect can be obtained by using `\left`, `\right` and preceding the expression with the command `\mleftright`:

```
\mleftright
\[ \ln \left(\frac xy\right),\quad\sin \left(x^2\right). \]
\mleftrightrestore
```

$\Longrightarrow$

$$\ln\left(\frac{x}{y}\right), \quad \sin\left(x^2\right).$$

For the present document, the call is

```
\usepackage[def-file=diffcoeff5,spaced=-1]{diffcoeff}
```

1.2 A Rogues' Gallery of derivatives

Browsing through some (rather old) calculus textbooks and texts on statistical mechanics, relativity and classical mechanics I find the following choice examples of derivatives 'disporting every which way'.

- Multi-character variables of differentiation un-parenthesized:

$$\frac{\partial\, \frac{\psi}{\Theta}}{\partial \frac{1}{\Theta}}, \quad \frac{\partial\, E/T}{\partial 1/T}, \quad \frac{\mathrm{d}\, \ln f}{\mathrm{d}\ln x_0}, \quad \frac{\partial^2 \psi}{\partial a_i\, \partial \frac{1}{\Theta}}, \quad \frac{\partial\, \mathcal{L}}{\partial \eta_{,i}^{(r)}}. \tag{1.1}$$

- Multi-character variables of differentiation parenthesized in *higher-order* derivatives, where the parentheses do not or (sometimes) do include the operator:

$$\frac{\partial^2 q}{\partial\left(\frac{1}{\Theta}\right)^2}, \quad \frac{\partial^2 q}{\partial(1/\Theta)^2}, \quad \frac{\partial^2 \varepsilon}{\partial(a_i)^2}, \quad \frac{\mathrm{d}^2\, \phi^i(x^i)}{(\mathrm{d}x^i)^2}. \tag{1.2}$$

  Should the d or ∂ be included within the parentheses, as in the last of (1.2), or not, as in the others? Logic says 'yes'; practice suggests (generally) 'no'.

- Indicating a point of evaluation is similarly varied:

$$\left.\frac{\partial \phi}{\partial \varepsilon}\right|_{\varepsilon=\varepsilon_0}, \quad \left.\frac{\mathrm{d}^2 \phi}{\mathrm{d}\varepsilon^2}\right|_{\varepsilon=\varepsilon_0}, \quad \left[\frac{\partial\, b^\beta}{\partial a^\alpha}\right]_{b=0}, \quad \left(\frac{\mathrm{d}u}{\mathrm{d}v}\right)_{v=0}. \tag{1.3}$$

ISO 80000-2 (item 2.11.13) favours the last of these – parentheses – for ordinary derivatives. Presumably, partial derivatives should follow suit, although parentheses are also used to indicate variables held constant:

$$\left(\frac{\partial}{\partial U}\,\frac{P}{T}\right)_V, \quad \left(\frac{\partial S}{\partial N_2}\right)_{U,V,N_1}, \quad (\partial S/\partial T)_V\,. \tag{1.4}$$

- Other symbols besides d and ∂ are used to denote derivative-like quantities. From introductory calculus and from classical mechanics and thermodynamics come δ and Δ, from fluid mechanics comes D:

$$\frac{\delta y}{\delta x}, \quad \frac{D\rho}{Dt}, \quad \left(\frac{\Delta U}{\Delta T}\right)_V, \quad \Delta U/\Delta T, \quad \frac{\delta\,\mathcal{L}}{\delta\eta^{(r)}}. \tag{1.5}$$

- There are those, like the International Organization for Standardization (ISO), who stipulate (or prefer) an upright d for their derivatives, and there are those (like the author, through sixty years of habit) who prefer a math-italic d:

$$\frac{\mathrm{d}y}{\mathrm{d}x}, \qquad \frac{dy}{dx}, \tag{1.6}$$

and of course also in slash-fraction form $\mathrm{d}y/\mathrm{d}x$, dy/dx. Subscripted forms of derivative are also used – for example, $\partial_x F$, or in the discussion of differential equations one sometimes comes across expressions like

$$\mathbf{D}_x^2 y + 2\mathbf{D}_x y - 4 = 0.$$

- When the differentiand is too big or awkward to sit in the numerator and is appended to the operator, the d or ∂ in the numerator is generally centred – but not always. In texts prior to the age of computerised typesetting one will sometimes find the symbol pushed to the *left*:

$$\frac{\partial}{\partial x^{l^*}}\,\frac{\partial\, x^{i^*}}{\partial x^{k^*}}, \quad \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{m\mathbf{q}_x}{\sqrt{1-q^2}}\right). \tag{1.7}$$

The keen-eyed will note an italic adjustment with the first expression, so that the ∂s in the numerators are indented a little (to line up – more or less – in a slanting column with the ∂s in the denominators).

- Then there is the case when the operator in the numerator differs from that in the denominator. For instance, in tensor calculus acceleration is sometimes written

$$\frac{\nabla\, v^i}{\mathrm{d}t} = \frac{\mathrm{d}\, v^i}{\mathrm{d}t} + \Gamma_k{}^i{}_h v^h \frac{\mathrm{d}\, y^k}{\mathrm{d}t}$$

where ∇v^i is the 'absolute differential' of the velocity v^i.

The `diffcoeff` package has the generative power to cope with all these variations – see §3.3 – although it is unlikely an author should need to call on this capacity to anything like the extent required for this Rogues' Gallery.

Chapter 2

Syntax and use

`diffcoeff` aims to ease the writing of derivatives (sometimes also called differential coefficients). There are long-established shorthands available in a few cases: $\dot{x}$ and $\ddot{x}$ for the time derivatives of a function x of time t; y' and y'' for the derivatives of a function y (usually) of x. But mostly derivatives are expressed in fraction form and require more keystrokes to compose. It is here that `diffcoeff` is aimed. It uses three pairs of commands: `\diff` and `\diffp` to write (upright) fraction forms of ordinary and partial derivatives like $\frac{\mathrm{d}y}{\mathrm{d}x}$, $\frac{\partial y}{\partial x}$, generally intended for display-style environments; `\difs` and `\difsp` for slash-fraction forms of ordinary and partial derivatives like $\mathrm{d}y/\mathrm{d}x$, $\partial y/\partial x$, generally intended for text-style environments; and `\difc` and `\difcp` to write compact forms of ordinary and partial derivatives like $\mathrm{d}_x y$ and $\partial_x y$. (Of these, the 's' forms replace the slash argument for the `\diff`, `\diffp` commands in version 4 of `diffcoeff`, and the 'c' form is new to version 5.[1])

Note

I refer throughout to the quantity or function being differentiated as the *differentiand* or *derivand* (in line with *integrand*, *operand*, etc.) and shall sometimes use `\difx` (resp. `\difxp`) to make general statements about any or all of `\diff`, `\difs` or `\difc` (resp.`\diffp`, `\difsp`, `\difcp`).

2.1 Syntax

All commands, `\difx`, `\difxp`, share the same syntax. With options present the syntax is

```
\difx.name.*[order-spec]<override>{differentiand}
    {variable(s)}[pt of eval]
```

[1] Suggested by a question on TEX StackExchange: https://tex.stackexchange.com/questions/652223/write-a-derivative-operator-without-denominator-using-diffcoef/652298#652298

```
\difx.name.**[order-spec]<override>{variable(s)}
    {differentiand}[pt of eval]
```

The syntax is identical for `\difxp`. The seven arguments have the following meanings:

- `name` (optional) A dot-delimited name to distinguish a variant form (non-default form) of derivative; not discussed further until §3 below, and specifically, §3.3.
- `*` (optional) The presence of a star (asterisk) signals: *append* the differentiand; its absence means the differentiand appears in the numerator of an upright- or slash-fraction form derivative; no effect for compact-form derivatives unless (see next) a second `*` is present; see §2.2.3.
- `*` (optional) The presence of a *second* star signals that the argument specifying the variable(s) of differentiation comes *before* the argument specifying the differentiand; this is sometimes convenient when a complicated or lengthy differentiand is appended; see §2.2.3.1.
- `order-spec` (optional) The order of differentiation when differentiating in a single variable, or a comma list of orders of differentiation for a mixed partial derivative; see §2.2.2 and §2.2.5.
- `override` (optional) The total order of differentiation when it cannot be calculated by `diffcoeff` or is wanted in a different form from that calculated by `diffcoeff`; see §2.2.5.3.
- `differentiand` (mandatory) The function being differentiated.
- `variable(s)` (mandatory) The variable of differentiation or a comma list of variables of differentiation (for a mixed partial derivative).
- `pt of eval` (optional) Point of evaluation or, for partial derivatives, variable or variables held constant; *no space* before the left square bracket; see §2.2.4.

Both mandatory arguments may be empty, but require empty brace pairs to indicate as much. (Omitting the differentiand makes sense for all forms of derivative, `\difx`, `\difxp`, but omitting the variable or variables of differentiation is sensible only for the compact forms, `\difc`, `\difcp` – see §3.3.2.6.)

2.2 General use

Writing `\diff{y}{x}` will produce $\frac{\mathrm{d}y}{\mathrm{d}x}$ in an inline math environment (i.e., placed between `\( \)` or `$ $`) or

$$\frac{\mathrm{d}y}{\mathrm{d}x}$$

in display style (placed, for instance, between `\[ \]`). In fact `\diff yx` (omitting the braces) will produce these results, with a saving on keystrokes. The braces are needed only when an argument – the variable of differentiation, or the differentiand – is multi-token:

`\[ \diff{\ln x}x \]` $\Longrightarrow$

$$\frac{\mathrm{d}\,\ln x}{\mathrm{d}x}$$

- If you want math-italic 'd's as default, see §3.4 on changing default settings. As noted earlier, upright 'd's conform to the standard ISO 80000-2 and are used in this document.

For inclusion in a line of text you might prefer to use a slash-fraction form of derivative. That is achieved with the `\difs` command: `$ \difs yx $` produces $\mathrm{d}y/\mathrm{d}x$. If you want still more compactness, you can use the `\difc` ('c' for *compact*) command: `$ \difc yx $` produces the form $\mathrm{d}_x y$.

Partial derivatives follow the same pattern as ordinary derivatives. The commands this time are `\diffp`, `\difsp` and `\difcp` for (upright) fraction, slash fraction and compact forms of partial derivative. Thus `\diffp{F}{x}`, or `\diffp Fx` with a saving on keystrokes, produce $\frac{\partial F}{\partial x}$ in text style and

$$\frac{\partial F}{\partial x}$$

in display style. (As for `\diff`, the omission of braces is possible when dealing with a single-token differentiand or differentiation variable.) For inline use, `\difsp Fx`, displays as $\partial F/\partial x$ and `\difcp` displays as $\partial_x F$. Given that `\partial` takes 8 keystrokes to type, all forms economise on keystrokes.

2.2.1 Spacing before the differentiand

There are (at least) two different ways in which we think of derivatives. We are all familiar with the argument presented in elementary calculus books where a curve is shown, and also a point on the curve through which a chord has been drawn. The chord is the hypotenuse of a small right-angled triangle, the other sides having lengths δx and δy and being parallel to the coordinate axes. The slope of the chord is $\frac{\delta y}{\delta x}$. By drawing smaller and smaller chords through the point, the ratio $\frac{\delta y}{\delta x}$ approaches the slope of the tangent to the curve at the point. We write

$$\frac{\mathrm{d}y}{\mathrm{d}x}$$

for the limit of $\frac{\delta y}{\delta x}$. It is natural following this line of argument to think of $\mathrm{d}y$ and $\mathrm{d}x$ as tiny lengths, like δy and δx, in which case it would be quite wrong to insert space between the d and the y (let alone the d and the x). $\mathrm{d}y$ is a single object, called a differential, and we write expressions like

$$\mathrm{d}y = \frac{\mathrm{d}y}{\mathrm{d}x}\,\mathrm{d}x$$

and justly call the fraction in this expression a *differential coefficient*.

But there is another way of viewing differentiation: as a process producing (or *deriving*) one function, $y'(x)$, from another, $y(x)$. Here the sense is of applying $\frac{\mathrm{d}}{\mathrm{d}x}$ to a quite separate object, the function $y(x)$. Although we include $y(x)$ in the numerator it is distinct from the d and should be separated from it by a small space:[2]

$$y'(x) = \frac{\mathrm{d}\,y(x)}{\mathrm{d}x}.$$

Here the fraction on the right is another name for the derived function y' and is justly called the *derivative* of y. As you can see a small space has been inserted between the d and the y in the numerator. By default the space is `3 mu` but with the ability to stretch by `1 mu` or shrink by `2 mu` – `3 mu plus 1 mu minus 2 mu` in TEX-speak[3] – as TEX adjusts lines to fit on the page. (A 'mu' is a 'math unit' and is one eighteenth of a quad.) The size of the space inserted by default can be easily changed; see §3.3 and §3.4.

- You may want all or most of your derivatives to have this space before the derivand. The `spaced=1` package option switches this behaviour on. However, I have used the `spaced=-1` option for the present document which inserts space only if the derivand contains *more than one token*. Thus `y(x)` will have space inserted before it, but `y` alone will not. This (I think) maintains the distinction between a differential coefficient, thought of as a ratio of tiny lengths, and a derivative, thought of as an operator applied to a function. `spaced=0` inserts no space before the derivand.

- In version 4 of `diffcoeff` an argument was added to the `\diff` command to manually introduce a space (the `!` switch) before the differentiand. This was a mistake and has been removed. If you wish to adjust the spacing, there are plenty of (short) commands in LATEX and `diffcoeff` to do the job; see immediately below, §2.2.1.1.

Slash-form derivatives also allow space before the derivand. By default this is `2 mu plus 1 mu minus 2 mu`, slightly reduced from the fraction-form value to avoid visually detaching the initial 'd' operator from the derivative as a whole. The value can be changed; see §3.3 and §3.4. For the present document, with `spaced=-1`, multi-token derivands have the space inserted, single-token derivands do not:

`$ \difs{\ln\sin x}x, \quad \difs st $` $\Longrightarrow \mathrm{d}\ln\sin x/\mathrm{d}x, \quad \mathrm{d}s/\mathrm{d}t.$

For *compact-form* derivatives the space before the derivand is *always* inserted, irrespective of the setting of the `spaced` package option, since the subscript precludes the entire symbol ever being viewed as a differential – it is always an operator operating on a function. The inserted space, `1 mu plus 1 mu minus 1 mu` by default, can be changed should you wish; see §3.3 and §3.4:

[2] I thank Hans Schülein for first raising this issue with me and for subsequent thoughtful comments.

[3] Or even `3muplus1muminus2mu`.

`$ \difc{\ln\sin x}x, \quad \difc st $` $\Longrightarrow \mathrm{d}_x \ln\sin x, \quad \mathrm{d}_t s.$

(The space is less for compact forms since the subscript already provides some visual separation.) The `spaced` package option has the same effects on partial derivatives. Thus with `spaced=1` or `spaced=-1`, `3 mu plus 1 mu minus 2 mu` of space is inserted before the differentiand `F(x,y)` in the first member of the following example, space of `2 mu plus 1 mu minus 2 mu` in the second, and space of `1 mu plus 1 mu minus 1 mu` in the third:

```
\[ \diffp{F(x,y)}x,\;\difsp{F(x,y)}x,\;\difcp{F(x,y)x, \]
```

$\Longrightarrow$

$$\frac{\partial F(x,y)}{\partial x},\ \partial F(x,y)/\partial x,\ \partial_x F(x,y).$$

But for single-token differentiands in this document the space is not inserted for upright and slash-form derivatives:

`\[ \diffp Fx,\quad \difsp Fx. \]` $\Longrightarrow$

$$\frac{\partial F}{\partial x}, \quad \partial F/\partial x.$$

If you always want the space present, use `spaced=1`; if you never want the space for upright- or slash-form derivatives, or wish to insert such space always 'by hand', use `spaced=0`.

2.2.1.1 Spacing commands

LaTeX has its own explicit spacing commands. In particular `\,` which is `3 mu` (a thin space) and `\!` which is `-3 mu` (a negative thin space) are convenient in math mode. The `diffcoeff` package adds four simple spacing commands to 'fill in (most of) the gap' between these two. These are

`\negmu` insert spacing of `-1 mu`;

`\nilmu` insert spacing of `0 mu` (cf. use of an empty brace pair `{}`);

`\onemu` insert spacing of `1 mu`;

`\twomu` insert spacing of `2 mu`.

It is also worth recalling here the reduced spacing around scalable delimiters that results from using `\mleft`, `\mright` in place of `\left`, `\right`; see §1.1 for the `mleftright` package option, and the example at §2.2.4.3.

2.2.2 Higher order derivatives

An optional argument allows the order of differentiation to be specified. The order need not be a number; an algebraic order of differentiation is perfectly acceptable as is a mix of the two:

```
\[ \diff[2]yx, \quad \diff[n+1]yx. \] ⟹
```

$$\frac{\mathrm{d}^2 y}{\mathrm{d}x^2}, \quad \frac{\mathrm{d}^{n+1} y}{\mathrm{d}x^{n+1}}.$$

As mentioned, the braces can be and have been omitted around the x and y since they are single tokens. The square brackets around the optional order-of-differentiation argument are essential. In slash form,

```
$ \difs[2]yx, \quad \difs[n+1]yx $ ⟹ d²y/dx², d^{n+1}y/dx^{n+1},
```

the latter of which is a bit of an eyesore. In compact form,

```
$ \difc[2]yx,\quad\difc[n+1]yx $ ⟹ d²_x y, d^{n+1}_x y.
```

Note that entering `1` as the optional argument has no effect:

```
$ \diff[1]yx,\;\difs[1]yx,\;\difc[1]yx $ ⟹ dy/dx, dy/dx, d_x y.
```

For partial derivatives when differentiating in only one variable the pattern is the same:

```
\[ \diffp[2]yx, \quad \diffp[n+1]yx. \] ⟹
```

$$\frac{\partial^2 y}{\partial x^2}, \quad \frac{\partial^{n+1} y}{\partial x^{n+1}}.$$

For the slash forms,

```
$ \difsp[2]yx,\quad \difsp[n+1]yx $ ⟹ ∂²y/∂x², ∂^{n+1}y/∂x^{n+1}.
```

and in compact form,

```
$ \difcp[2]yx,\quad \difcp[n+1]yx $ ⟹ ∂²_x y, ∂^{n+1}_x y.
```

For partial differentiation in more than one variable – so-called *mixed* partial derivatives – see §2.2.5.

2.2.2.1 Alternative method

From version 5.3 of `diffcoeff` it is also possible to specify the order by the method shown in the example

```
\[ \diffp y{x:2}, \quad \diffp y{x:n+1}. \] ⟹
```

$$\frac{\partial^2 y}{\partial x^2}, \quad \frac{\partial^{n+1} y}{\partial x^{n+1}}.$$

A colon separates the variable from its order of differentiation. This is really intended not for one variable, as here, but for mixed partial derivatives when there are a number of variables subject to different orders of differentiation.

2.2.3 Appending the differentiand

Some differentiands are too big or awkward to be placed neatly in the numerator of a derivative and it is natural to *append* them to a preceding differential operator. One could leave the numerator argument empty in the `\diff` or `\diffp` command and follow the command with the differentiand, but `diffcoeff` offers a better way: star the `\diff` or `\diffp` command. This tells `diffcoeff` to append the differentiand. Thus suppose the differentiand is a polynomial, say $ax^2 + bx + c$. Add a star (an asterisk) to the `\diff` command:

```
\[ \diff*{(ax^2+bx+c)}x \] ⟹
```

$$\frac{\mathrm{d}}{\mathrm{d}x}\,(ax^2 + bx + c).$$

Or, for a partial derivative, one might want to indicate in the differentiand all the variables on which it depends:

```
\[ \diffp*[2]{\Phi(x,y,z)}x \] ⟹
```

$$\frac{\partial^2}{\partial x^2}\,\Phi(x, y, z).$$

A virtue of using an asterisk to append the differentiand is that if one isn't sure whether to append or not, it is an easy matter to simply insert or delete the asterisk to compare the results.

For instance, a second derivative is an iterated derivative – one in which a derivative forms the differentiand of another . Thus

```
\[ \diff[2]yx = \diff*{\diff yx}x \] ⟹
```

$$\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} = \frac{\mathrm{d}}{\mathrm{d}x}\,\frac{\mathrm{d}\,y}{\mathrm{d}x}.$$

This result is more elegant to my eye than what results when removing the asterisk,

```
\[ \diff[2]yx = \diff{\diff yx}x \] ⟹
```

$$\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} = \frac{\mathrm{d}\,\frac{\mathrm{d}\,y}{\mathrm{d}x}}{\mathrm{d}x},$$

although whether the *meaning* is clearer is moot.

Since the differentiand is appended *by default* in compact-form derivatives, starring such a derivative has no effect other than when a second asterisk is used to transpose the order of arguments.

2.2.3.1 Transposing the argument order

If a *second* asterisk follows the first, the order of the arguments specifying the differentiand on the one hand and variable or variables of differentiation on the other are reversed. Thus it is clearer to the eye to write

```
\[ \diff**x{(ax^2+bxy+cy^2)} \] ⟹
```

$$\frac{\mathrm{d}}{\mathrm{d}x}\,(ax^2 + bxy + cy^2)$$

than `\[ \diff*{(ax^2+bxy+cy^2)}x \]`, where the eye has to search for the variable of differentiation. This is especially the case if the differentiand contains more than one variable and includes commands like `\frac` or `\sqrt` requiring braced arguments:

```
\[ \diffp**x{\frac1{\sqrt{x^2-y^2}}} \] ⟹
```

$$\frac{\partial}{\partial x}\,\frac{1}{\sqrt{x^2 - y^2}}$$

For compact-form derivatives the initial, appending asterisk is always implicitly present. However, it must be *explicitly* present for the second asterisk to take effect:

```
\[ \difcp yx,\quad \difcp*yx,\quad \difcp**yx \] ⟹
```

$$\partial_x y, \quad \partial_x y, \quad \partial_y x$$

2.2.3.2 Operator parenthesizing

In slash style with the star option, the polynomial example becomes

```
$ \difs*{(ax^2+bx+c)}x $ ⟹ (d/dx)(ax^2+bx+c)
```

where parentheses have been automatically inserted around the differential operator. Similarly, for slash-style partial derivatives,

```
\( \difsp*[n]{f(x)}x \) ⟹ (∂^n/∂x^n) f(x)
```

parentheses are again inserted automatically around the differential operator. Like other elements of automatic formatting, this behaviour is user-adjustable; see §§3.3, 3.4.

2.2.4 Point of evaluation/variables held constant

If you want to specify a point at which a derivative is evaluated, append a final optional argument:

```
\[ \diff[2]yx[0] \] ⟹
```

$$\left(\frac{\mathrm{d}^2 y}{\mathrm{d}x^2}\right)_0$$

Note that there must be *no space* before the left square bracket of the trailing argument, otherwise it will be treated as part of the wider mathematical expression of which the derivative is part and typeset as such. (This should not cause a LaTeX error.)

- If you prefer to use subscripted *square* brackets

$$\left[\frac{\partial F(x,y)}{\partial x}\right]_{(0,0)}$$

  or a subscripted vertical rule after the derivative

$$\left.\frac{\partial F(x,y}{\partial x}\right|_{(0,0)}$$

  to indicate a point of evaluation, then this can easily be done; see specifically §3.3.2.3 (or §3.4 on changing default settings). Parentheses are the ISO recommendation; see ISO 80000-2.

Because the slash form spreads the derivative out horizontally, parentheses are the natural way in this case to indicate a point of evaluation:

`$ \difs{\ln sin x}{sin x}[x=\pi/3] $` ⟹ $(\mathrm{d}\ln\sin x/\mathrm{d}\sin x)_{x=\pi/3}$.

A vertical rule can easily become too remote from the opening d of the differential coefficient: $\mathrm{d}\ln\sin x/\mathrm{d}\sin x|_{x=\pi/3}$. Parentheses tie the whole cluster of symbols together.

One reason to query the ISO preference for subscripted parentheses to indicate a point of evaluation is that subscripted parentheses are used with partial derivatives to indicate variables held constant. This occurs frequently in thermodynamics for example. In the following well-known relation in thermodynamics, the differentiands are appended and the trailing argument is used to indicate the variables held constant:

```
\[ \diffp*{\frac PT}U[V] = \diffp*{\frac 1T}V[U] \] ⟹
```

$$\left(\frac{\partial}{\partial U}\,\frac{P}{T}\right)_V = \left(\frac{\partial}{\partial V}\,\frac{1}{T}\right)_U.$$

This is much easier to write than building the expressions ‘by hand’, starting with `\left(` and finishing with `_U`.

2.2.4.1 Superscripts

It is easy to add a superscript to a derivative to indicate evaluation at two points and the difference between the values:

```
\[ \diff {\sin x}x[0]^{\!\!\pi/2} \] ⟹
```

$$\left(\frac{\mathrm{d}\,\sin x}{\mathrm{d}x}\right)_0^{\pi/2}$$

but to my eye either square brackets or a vertical rule are clearer for this purpose (and do not involve nudging the subscript or superscript closer to the right delimiter); see §3.3.

2.2.4.2 Empty trailing argument

If the trailing argument is included but left empty it will, with the default set-up, wrap the derivative in parentheses but with no subscript. This fact can be exploited. Thus, for a particle of mass m moving along a line, distance x at time t, the kinetic energy is:

```
$ \tfrac 12 m \difs xt[]^2 $ ⟹
```

$\frac{1}{2}m(\mathrm{d}x/\mathrm{d}t)^2$.

Or, again exploiting the parentheses resulting from an empty trailing argument, Lagrange's equations of motion in analytic mechanics can be written,

```
\[ \diffp L{q_k}-\diff**t{ \diffp L{\dot{q}_k}[] } = 0 \] ⟹
```

$$\frac{\partial L}{\partial q_k} - \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial\, L}{\partial \dot{q}_k}\right) = 0.$$

(See §2.2.3.1 for the double asterisk.) Like the author, you may feel that there is too much whitespace between $\frac{\mathrm{d}}{\mathrm{d}t}$ and the left parenthesis in this expression. One obvious remedy is to insert a negative thin space `\!` before the second `\diffp` command. Another is to use the package `mleftright`.

2.2.4.3 Use of the package `mleftright`

The `mleftright` package 'tightens' the spacing around `\left`, `\right` delimiters. The user either by explicitly replaces `\left`, `\right` by `\mleft` and `\mright` or uses the command `\mleftright` which effectively turns subsequent occurrences of `\left`, `\right` into `\mleft`, `\mright`; `\mleftrightrestore` returns `\left`, `\right` to their original selves. This process can be 'short-circuited' by using the `diffcoeff` package option `mleftright` that inserts `\mleftright` in the preamble; see §1.1.

However, that package option is not used in this document. Nonetheless `mleftright` is a required package of version 5 of `diffcoeff` and its commands are available for use. Hence to reduce the whitespace I can write

```
\mleftright
\[ \diffp L{q_k}-\diff**t{\diffp L{\dot{q}_k}[]} = 0. \]
\mleftrightrestore
```

$\Longrightarrow$

$$\frac{\partial L}{\partial q_k} - \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}_k}\right) = 0.$$

which is better, but better still to my eye is to also add a negative thin space `\!` before the second `\diffp` command:

```
\mleftright
\[ \diffp L{q_k}-\diff**t{ \!\diffp L{\dot{q}_k}[] } = 0 \]
```

$\Longrightarrow$

$$\frac{\partial L}{\partial q_k} - \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}_k}\right) = 0.$$

The problem is that in addition to the space around `\left`, `\right` pairs (which is reduced by issuing the command `\mleftright`), there is also the space inserted by `diffcoeff` before a multi-token differentiand. The user should be aware of this, and may wish to define a 'variant form' (see §3.3) that introduces no space before an appended differentiand.

2.2.5 Mixed partial derivatives

The new thing with partial derivatives, not present with ordinary derivatives, is so-called *mixed* partial derivatives, where differentiation occurs in more than one variable. If each variable is differentiated only to the first order, then it is easy to specify the derivative. Suppose F is a function of three variables, x, y and z. Then

```
\[ \diffp F{x,y,z},\quad\diffp{F(x,y,z)}{x,y,z}. \]
```

$\Longrightarrow$

$$\frac{\partial^3 F}{\partial x\,\partial y\,\partial z}, \quad \frac{\partial^3 F(x,y,z)}{\partial x\,\partial y\,\partial z}.$$

(The `spaced=-1` package option inserts space before the multi-token differentiand in the second of these.)

The variables of differentiation are listed in order in a comma list – `{x,y,z}` – forming the second mandatory argument. The total order of differentiation (3 in this example) was inserted automatically. It did not need to be specified or calculated 'by hand' – `diffcoeff` did the calculation.

The slash form is

```
$ \difsp F{x,y,z} $
```

$\Longrightarrow \partial^3 F/\partial x\,\partial y\,\partial z,$

as expected, and the compact form is

```
$ \difcp F{x,y,z} $
```

$\Longrightarrow \partial_x\partial_y\partial_z F.$

One might wonder about even more compact notations like $\partial^3_{xyz}F$ for this example but it becomes messy if different orders of differentiation are involved for different variables.

To differentiate variables to higher order, their orders need to be specified explicitly. To do so use a comma list for the optional argument (or, since version 5.3, use the alternative method of §2.2.5.2):

`$ \difcp[2,3]F{x,y,z} $` $\Longrightarrow \partial_x^2\partial_y^3\partial_z F$,

`\[ \diffp[2,3]F{x,y,z} \]` $\Longrightarrow$

$$\frac{\partial^6 F}{\partial x^2\partial y^3\partial z}.$$

Notice that the overall order of the derivative – 6 – in the second of these is again automatically calculated and inserted as a superscript on the ∂ symbol in the numerator.

In the example, the comma list of orders has only *two* members, although there are *three* variables. It is assumed that the orders given in the comma list apply in sequence to the variables, the first order to the first variable, the second to the second variable, and so on, and that any subsequent orders not listed in the optional argument are, by default, 1. Thus we need to specify only 2 and 3 in the example; the order of differentiation of z is 1 by default. But you *cannot* use an order specification like `[,,2]`; instead write `[1,1,2]` (which is the natural thing to do in any case). It is only the *tail* of an order specification which can be omitted.

In the other direction, if there are more orders of differentiation specified than there are variables, the list of orders is truncated to match the number of variables.

2.2.5.1 Algebraic orders of differentiation

Orders of differentiation do not need to be numerical. They can also be algebraic:

`\[ \diffp[2m-1,m+1,2]F{x,y,z} \]` $\Longrightarrow$

$$\frac{\partial^{3m+2} F}{\partial x^{2m-1}\partial y^{m+1}\partial z^2}$$

The total order of differentiation is still calculated by `diffcoeff`. Or again,

`\[ \diffp[1,km+1,m+k-1]{F(x,y,z)}{x,y,z} \]` $\Longrightarrow$

$$\frac{\partial^{m+k+km+1} F(x,y,z)}{\partial x\,\partial y^{km+1}\partial z^{m+k-1}}.$$

2.2.5.2 Alternative method

When there are two or more variables of differentiation, particularly when subject to different orders of differentiation, it may be easier to see which order is associated with which variable if they are paired together in the variable argument. To do so, separate the order from the variable by a colon,[4] like this:

```
\[ \diffp{F(x,y,z)}{x,y:km+1,z:m+k-1} \] ⟹
```

$$\frac{\partial^{m+k+km+1} F(x,y,z)}{\partial x\, \partial y^{km+1} \partial z^{m+k-1}}.$$

In the example, note that it suffices to write `x` rather than `x:1`.

If, in a fit of absent-mindedness, one specifies the orders of differentiation by both methods, it is the orders in the variable argument that prevail:

```
\[ \diffp[1,2,3]{F(x,y,z)}{x:4,y:5,z:6} \] ⟹
```

$$\frac{\partial^{15} F(x,y,z)}{\partial x^4 \partial y^5 \partial z^6}$$

2.2.5.3 Order-override option and command

With version 5.3 of `diffcoeff` the order-override option has been reinstated, having been replaced in versions 5.0 to 5.2 by a command `\difoverride` (see below). In version 4 and earlier this optional argument was square-bracket delimited. It is now *angle-bracket* delimited (using the 'less than' and 'greater than' symbols, `< >`).[5] The reason for angle brackets is both for visual distinction and because the alternative method of specifying the order of differentiation (by means of colons in the variable specification; see immediately above §2.2.5.2) requires the override option to be distinguishable from the order specification.

In the penultimate example above, the total order of differentiation $m+k+km+1$ factorizes to $(k+1)(m+1)$. `diffcoeff` is not a computer algebra system and does not do such factorizations but you can still express the total order in this form by using the override option, entering the factorized form between angle brackets before the differentiand:

```
\[
  \diffp<(k+1)(m+1)>{F(x,y,z)}{x,y:km+1,z:m+k-1}
\]
```

⟹

$$\frac{\partial^{(k+1)(m+1)} F(x,y,z)}{\partial x\, \partial y^{km+1} \partial z^{m+k-1}}.$$

When the override option is used, the algorithm that calculates the total order is sidestepped. It does not get called at all. In this way not only can the

[4] I thank Christophe Bal for this suggestion.

[5] I thank Christophe Bal for urging the availability of this argument and the use of angle brackets.

total order be presented in whatever manner one wishes but essentially arbitrary material can be attached as a superscript to the ∂ symbol in the numerator. (For compact-form derivatives, which do not use a total order of differentiation, the override option is irrelevant.)

Order-override command: Alternatively, you can use the `\difoverride` command in place of the override option. You might prefer to do this to avoid cluttered expressions. The command takes one (mandatory) argument, the total order of differentiation, which it stores:

```
\[
  \difoverride{(k+1)(m+1)}
  \diffp[1,km+1,m+k-1]{F(x,y,z)}{x,y,z}
  \difoverride{},\quad
  \diffp[1,km+1,m+k-1]{F(x,y,z)}{x,y,z}
\]
```

$\Longrightarrow$

$$\frac{\partial^{(k+1)(m+1)} F(x,y,z)}{\partial x\, \partial y^{km+1} \partial z^{m+k-1}}, \quad \frac{\partial^{m+k+km+1} F(x,y,z)}{\partial x\, \partial y^{km+1} \partial z^{m+k-1}}$$

Note that in the example `\difoverride` has been used *within* the math environment. This is good practice. It prevents the contents of the command erroneously overriding the orders of later derivatives in other math environments; but it does mean cancelling the override (with the statement `\diffoverride{}`) in *this* environment if a second derivative is present, to prevent the second derivative also displaying the factorized form.

2.2.5.4 Parentheses

Auto-calculation of the total order accommodates the simple use of parentheses:

`\[ \diffp[2m-(k+1),2(k+1)-m]{F(x,y,z)}{x,y,z} \]` $\Longrightarrow$

$$\frac{\partial^{m+k+2} F(x,y,z)}{\partial x^{2m-(k+1)} \partial y^{2(k+1)-m} \partial z}$$

This is an example of the use of *dynamic* parentheses: the left parenthesis in each case is preceded by a number or a sign. In evaluating the total order `diffcoeff` multiplies out the expression (or that is the effect).

On the other hand, an order specification like `[f(n+1),f(n-1)]` is an example of the use of *static* parentheses where they are part of the familiar 'function of' notation – in this case a function f of some variable, say x, evaluated at $x = n \pm 1$. `diffcoeff` *always* interprets a left parenthesis preceded by something that is neither number nor sign in this way. It does not try to multiply out such expressions when calculating the total order.

The following example combines both uses – and includes a nested pair of (dynamic) parentheses:

```
\[\diffp[2(f(n)-(m-1)),5-(f(n)+m)]F{x,y}\] ⟹
```

$$\frac{\partial^{7-3m+f(n)}F}{\partial x^{2(f(n)-(m-1))}\partial y^{5-(f(n)+m)}}$$

Where confusion arises is with specifications like `[m(k-1)+1,m(k+1)-1]` where `m` could be interpreted as either a function or a variable. As stated, `diffcoeff` *always* interprets a left parenthesis preceded by something that is not a number or a sign as signalling 'function of'. Hence:

```
\[\diffp[m(k-1)+1,m(k+1)-1]F{x,y}\] ⟹
```

$$\frac{\partial^{m(k-1)+m(k+1)}F}{\partial x^{m(k-1)+1}\partial y^{m(k+1)-1}}$$

If, in fact, `m` is intended as a *variable* then the order-override option or command is there to rescue the situation:

```
\[ \diffp<2mk>F{x:m(k-1)+1,y:m(k+1)-1} \]
```

⟹

$$\frac{\partial^{2mk}F}{\partial x^{m(k-1)+1}\partial y^{m(k+1)-1}}$$

2.2.5.5 Error messages

The order-override command is also needed when calculation of the total order is beyond the abilities of `diffcoeff`. The package is *not* a computer algebra system. It can cope with order specifications where variables are followed by diverse arithmetic operators: `n^2`, `m\times n`, `m/2` and the like cause no problems. But a *number* can be followed *only* by a sign or a variable or a left parenthesis. Anything beyond this will raise an error. For instance

```
\[ \diffp[2^k]F{x,y} \]
```

produces a message beginning '`! Package diffcoeff Error:`' and continuing,

```
number followed by ^ in the order spec. [2^k,1] on
line xx. Calculation of the total order of
differentiation fails in this case. Use the
override option (or \difoverride command) to
enter the total order. See the diffcoeff
documentation for further information.
```

(The `xx` will be replaced by a specific line number in each case. Line breaking may also differ from case to case.) To avoid such errors and enable compilation to proceed, do as the message suggests – use the override option (or `\difoverride` command). For (a slightly more complicated) example,

```
\[ \diffp[2^n+1,2^n-1]<2^{n+1}>F{x,y} \] ⟹
```

$$\frac{\partial^{2^{n+1}} F}{\partial x^{2^n+1} \partial y^{2^n-1}}$$

There are limitations on what order specifications the `diffcoeff` package can 'digest', but in real life that is unlikely to be significant. Mixed partial derivatives are used far less often than the pure derivatives, and when they *are* used it is nearly always to low numerical orders like 1 or 2. For those rare other cases, `\difoverride` is always available.

2.2.5.6 Comma list of variables of differentiation

In tensor calculus differentiations are almost always in terms of super- or subscripted coordinates. In many other contexts this is the case too – the reciprocal of the temperature in thermodynamics or generalized coordinates in analytical mechanics. This is why a comma list is used in `diffcoeff` for specifying variables of differentiation for mixed partial derivatives. Although it would be nice to write the minimal `{xy}` rather than `{x,y}` when two variables x and y are involved, the extra writing is trivial and the comma list allows a simpler handling of multi-character variables. For instance in tensor calculus we get expressions like

```
\[ \diffp{A_i}{x^j,x^k} \] ⟹
```

$$\frac{\partial^2 A_i}{\partial x^j \partial x^k}.$$

It is easier to write `{x^j,x^k}` here than, say, `{{x^j}{x^k}}` to distinguish the variables. It's also easier to read, particularly if the indices themselves get ornamented and need surrounding braces:

```
\[ \diffp{A_i}{x^{j'},x^{k'}} \] ⟹
```

$$\frac{\partial^2 A_i}{\partial x^{j'} \partial x^{k'}}.$$

Compare that variable specification with `{{x^{j'}}{x^{k'}}}`. Admittedly some extra whitespace would help here, but the point stands: the comma list requires fewer nested braces – unless a variable of differentiation includes a comma, for then the comma needs to be enclosed in braces. There are plenty of instances of this out in the world (see, e.g., the last equation of (1.1)) but it is overall a rare occurrence.

2.2.5.7 Spacing in the denominator

In Chapter 18 of the *The TEXbook*, Knuth suggests inserting a thin space, `\,` (or `3 mu`), between differentials in appropriate contexts, giving as an example

$dx\,dy = r\,dr\,d\theta$. In the denominator of a derivative, however, that degree of extra spacing – to my eye – seems too great, interfering with seeing the derivative 'as a whole',

$$\frac{\partial^3 F}{\partial x\,\partial y\,\partial z},$$

especially for the slash-form of derivative: $\partial^3 F/\partial x\,\partial y\,\partial z$. Some spacing is desirable, but less. By default `diffcoeff` inserts `2 mu` (with stretch and shrink) between the differentials: $\partial^3 F/\partial x\,\partial y\,\partial z$.

Should a differentiation occur to higher order and so a variable acquire a superscript, an adjustment is made to the extra spacing. By default `1 mu` is subtracted from the default spacing. Thus in

$$\frac{\partial^4 F}{\partial x^2\,\partial y\,\partial z},$$

`2 mu` of spacing is inserted between the ∂y and ∂z, but because the superscript already provides some separation between them, only `1 mu` is inserted between ∂x^2 and ∂y. The values used for the spacing and its adjustment in the presence of a superscript can be changed by the user; see Chapter 3.

When the variables themselves are super- or subscripted, as happens in tensor calculus, no automatic adjustment is made. Any fine-tuning must be done by the user using explicit spacing commands – like `\negmu` (a space of `-1 mu`); see §2.2.1.1:

```
\[ \diffp{A_i}{ x^j\negmu,x^k } \]
```

$$\Longrightarrow \quad \frac{\partial^2 A_i}{\partial x^j\,\partial x^k}.$$

The `\negmu` decreases the spacing between the terms from the default `2 mu` (with stretch and shrink) to `1 mu`.

2.2.6 Multi-token variables: parenthesizing

Differentiating a function of a function may involve a multi-character differentiation variable. For instance, to differentiate $\ln\sin x$ in x means forming the product

```
\[ \diff{\ln\sin x}{\sin x}\diff{\sin x}x \]
```

$$\Longrightarrow \quad \frac{\mathrm{d}\,\ln\sin x}{\mathrm{d}\sin x}\frac{\mathrm{d}\,\sin x}{\mathrm{d}x}.$$

Forming the *second* derivative of $\ln\sin x$ will now involve forming, among other quantities,

```
\[\diff[2]{\ln\sin x}{\sin x}\]
```

$$\Longrightarrow \quad \frac{\mathrm{d}^2\,\ln\sin x}{\mathrm{d}(\sin x)^2}$$

Parentheses have been inserted automatically by `diffcoeff` around $\sin x$ in the denominator to avoid any visual hint that we are differentiating in the sine of x^2.

That is the problem: with a long (multi-character) variable, the superscript in a higher order derivative may look as if it applies to only part – the last character – of a multi-character variable. To solve that problem, `diffcoeff` inserts parentheses around the variable – for *higher-order* derivatives, but not for first-order derivatives where the problem does not arise. You may prefer `diffcoeff` not to parenthesize by default. Changing the default setting is easily accomplished; see §3.4.

And if you do want parentheses, are they in the right place? Logically, no. They should include the d: $(d\sin x)^2$ – it is the differential $d\sin x$ that is of the second order. But as the examples in the Rogues' Gallery show – see particularly (1.2) – the inclination seems to be to do otherwise. This may be because one wants in any case to parenthesise the variable to show that the 'd' symbol attaches to the whole variable and not just its *first* character. A second, outer pair of parentheses then seems too fussy and detracts from comprehending the symbol 'at a glance':

$$\frac{\mathrm{d}^2 f(x)}{(\mathrm{d}(x/k))^2}.$$

Customary but illogical notations are familiar in mathematics – think of the position of the superscripts in an identity like $\sin^2\theta + \cos^2\theta = 1$. In any case, the manner of this wrapping in parentheses – if any – of long variables for *higher order* derivatives is customisable (§3.4).

For first order derivatives parenthesising does not occur automatically. If you want the variable of differentiation to be parenthesised, you need to do it yourself:

```
\[\diff {f(x)}{x/k}, \quad \diff {f(x)}{(x/k)}.\] ⟹
```

$$\frac{\mathrm{d}\, f(x)}{\mathrm{d}x/k}, \quad \frac{\mathrm{d}\, f(x)}{\mathrm{d}(x/k)}.$$

To my eye, in this particular case, the parenthesized version seems necessary. The discussion applies equally to ordinary and partial derivatives. In thermodynamics and statistical mechanics one may want to differentiate in the reciprocal of the temperature, $1/\Theta$ say:

```
\[ \diffp[2]q{\frac 1\Theta} \] ⟹
```

$$\frac{\partial^2 q}{\partial\left(\frac{1}{\Theta}\right)^2}.$$

As noted, when differentiating to first order, parenthesising is up to the user:

```
\[ \diffp q{(\frac 1\Theta),V} \] ⟹
```

$$\frac{\partial^2 q}{\partial(\frac{1}{\Theta})\,\partial V}.$$

Chapter 3

Templates, defaults & variants

`diffcoeff` is built on the facilities offered by the `xtemplate` package (included in the LaTeX3 bundle `l3packages`). The stuff of `xtemplate` is *templates*, their definition, their manipulation. For `diffcoeff` a template is a list of parameter values determining how a derivative looks in the pdf. The parameters may be broad-brush settings like whether the derivative is built from `\frac` or the slash `/` or in compact form, or whether the operator symbol is `d` or `\partial` (or `\nabla` or `\delta` or ...), or the parameters may be finer-grained, determining minutiae of spacing, easily missed at a casual glance but giving some cumulative overall effect. Access to the parameters is gained through the command `\difdef`[1], one argument of which is a *key=value* list of parameter values. Each such list is given a name (the second argument of `\difdef`) and is 'turned into a derivative' by placing the name between dots as the first argument of the appropriate `\difx`, `\difxp` commands[2]. All this is discussed in §3.3 below.

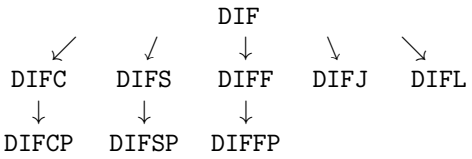
3.1 Template structure

To write a derivative one doesn't want to have to type a long list of *key=value* statements each time. The *default* values given to keys is crucial. Only some of the defaults appropriate for, say, an upright fraction ordinary derivative are going to be relevant for a slash-fraction partial derivative let alone a compact form partial derivative. This suggests creating a primary template as a 'super-repository' of default values and from this creating secondary or child templates in which (only) *some* of the defaults are changed – and, if necessary, creating from these child templates children of their own (grandchild templates) in which again some further defaults are adjusted.

[1] In version 5; it has *three* arguments and replaces the two-argument command `\diffdef` in version 4 of `diffcoeff`.

[2] And – see Chapter 4 – of the differential and jacobian commands, `\dl` and `\jacob`.

Table 3.1: Template inheritance



In `diffcoeff`, the template that is the 'primogenitor' of the lines of default inheritance is named `DIF`. It is the repository of all possible keys used in all possible forms of derivative (at least in `diffcoeff`!) and so has keys appropriate to upright-fraction, slash-fraction and compact forms of derivative; it has keys appropriate to multi-variable partial derivatives and single-variable ordinary derivatives, but it is not actually used to form derivatives. That is the role of its child templates `DIFF`, `DIFS` and `DIFC` corresponding to derivatives of upright-fraction, slash-fraction and compact forms. These child templates inherit the defaults of `DIF` save for some settings explicitly changed in the child template relevant to the specific forms of fraction specified by each.

Apart from the operator symbol, most of the settings in the child templates `DIFF`, `DIFS` and `DIFC` are also appropriate for partial derivatives. From a code design point of view, there is a certain neatness at not multiplying the number of templates in play, but actual use – for instance, forming the 'Rogues' gallery' of §1.2 – suggests the further step of creating additional templates specifically for *partial* derivatives in the three fraction forms. Apart from the operator symbol, the templates `DIFFP`, `DIFSP` and `DIFCP` inherit nearly all the defaults of their parents `DIFF`, `DIFS` and `DIFC` respectively.

Again, actual use suggests two further templates, both direct children of `DIF`, for the creation of jacobians, template `DIFJ`, and differentials, template `DIFL`, with default values appropriate to each. In all `diffcoeff` uses nine templates, the arrows in Table 3.2 indicating lines of inheritance of default values. Only the child and grandchild templates of `DIF` are used for actual construction of derivatives (and jacobians and differentials). `DIF` itself sits 'above the fray'.

3.2 Default values for template `DIF`

Table 3.2 lists the keys available for forming derivatives and the default values assigned in the 'grandparent' template, the primogenitor, `DIF`. Different forms of derivative demand different defaults for some keys. Where a key is relevant for more than one style of derivative the default value is chosen according to the following precedence scheme:

1. *ordinary upright*-fraction derivatives in *display*-style environments
2. *ordinary slash*-fraction derivatives in *text*-style environments
3. *ordinary compact*-form derivatives in *text*-style environments

Users of version 4 of `diffcoeff` will notice similarities with and differences from the earlier version. Some key names remain (`op-symbol`), some names have changed (`multi-term-sep` for `denom-term-sep`), keys beginning with an asterisk, `*`, lack a following hyphen (`*derivand-sep` rather than `*-derivand-sep`), some keys have vanished (the `/` keys), and there are some new keys (`lvwrap-Ldelim`, `lvwrap-Rdelim`). The redesign of the user interface – `\difs`, `\difsp` for the `/` switch, the new compact form commands `\difc`, `\difcp`, and bringing the jacobian and differential within the `DIF` template structure – meant revisiting and rethinking the list of keys. In the end it seemed simpler (less confusing) to treat this as a completely new list rather than an amendment of the earlier one.

The first column in table 3.2 lists key names, the second column default values, and the third column to which form or forms of derivative the key is *relevant* – meaning that assigning a different value to the key can change the appearance of the corresponding derivative in some way. The identifiers have these meanings:

f, fp upright fraction ordinary derivative, partial derivative;

s, sp slash fraction ordinary derivative, partial derivative;

c, cp compact ordinary derivative, partial derivative;

j jacobian;

l differential.

In Table 3.2 and following tables, all values specifying a space require the unit (`mu`) to be included; a number alone does not suffice. (A 'mu' is a 'math unit', 1/18 of a quad. A thin space `\,` is 3 mu.) 'Elastic' spaces with stretch and shrink can be compacted, like `3muplus1muminus2mu` for `3 mu plus 1 mu minus 2 mu`.

Available keys and their defaults are the following (if you are dissatisfied with some choices, they can be changed; see the discussion at §3.4):

style the fraction form of derivative;

- for upright-fraction derivatives, `\diff`, `\diffp`, a choice of `frac`, `tfrac` or `dfrac`:
  - `frac` results in a fraction formed from `\frac`, scalable
  - `tfrac` results in a fraction formed from `\tfrac`, not scalable
  - `dfrac` results in a fraction formed from `\dfrac`, not scalable
  - default in templates `DIFF`, `DIFFP` = `frac`
- for slash-fraction derivatives,`\difs`, `\difsp`, a choice of `/`, `auto`, `big`, `Big`, `bigg` or `Bigg`
  - `/` forms the slash fraction with `/`, not scalable

Table 3.2: `DIF` defaults

key	default	relevance
style	`frac`	f, fp, s, sp, c, cp, j
slash-tok	`/`	s, sp, j
slash-sep	`0 mu`	s, sp, j
derivand-sep	`3 mu plus 1 mu minus 2 mu`	f, fp, s, sp, c, cp
op-symbol	`\mathrm{d}`	f, fp, s, sp, c, cp, j, l
op-symbol-alt	`op-symbol`	f, fp, s, sp, j
op-order-nudge	`0 mu`	f, fp, s, sp, c, cp
var-sup-nudge	`1 mu`	f, fp, s, sp, l
multi-term-sep	`2 mu plus 1 mu minus 1 mu`	f, fp, s, sp, c, cp, l
term-sep-adjust	`-1 mu`	f, fp, s, sp, c, cp, l
long-var-wrap	`d(v)`	f, fp, s, sp, l
lvwrap-Ldelim	`\mleft (`	f, fp, s, sp, j, l
lvwrap-Rdelim	`\mright )`	f, fp, s, sp, j, l
lvwrap-sup-nudge	`-2 mu`	f, fp, s, sp, l
outer-Ldelim	`\left (`	f, fp, s, sp, c, cp, j, l
outer-Rdelim	`\right )`	f, fp, s, sp, c, cp, j, l
elbowroom	`0 mu`	f, fp, s, sp, c, cp, j, l
sub-nudge	`-5 mu`	f, fp, s, sp, c, cp
op-sub-nudge	`0 mu`	c, cp
*derivand-sep	`derivand-sep`	f, fp, s, sp, c, cp
*op-set-left	`false`	f, fp, j
*italic-nudge	`0 mu`	f, fp, j
*inner-wrap	`false`	s, sp
*inner-Ldelim	`(`	s, sp
*inner-Rdelim	`)`	s, sp
*outer-Ldelim	`\big [`	s, sp
*outer-Rdelim	`\big ]`	s, sp
*sub-nudge	`0 mu`	s, sp

  - `auto` forms the slash fraction with `\left. \middle/ \right.`, scalable
  - `big`, `Big`, `bigg` and `Bigg` form the slash fraction with `\big/`, `\Big/`, `\bigg/` and `\Bigg/` respectively, not scalable
  - default in templates `DIFS`, `DIFSP` = `/`
- for compact-form derivatives, `\difc`, `\difcp`, a choice of `_` or `dl`
  - `_` forms derivatives of compact form like $\mathrm{d}_x y$, $\partial_x \partial_y^2 z$
  - `dl` forms differentials like $\mathrm{d}x$ and $\partial x^2 \partial y \partial z$
  - default in templates `DIFC`, `DIFCP` = `_`
- overall default in template `DIF` = `frac`

slash-tok token or tokens used for the slash fraction; (see §3.3.3 for a different assignment) default `/`

slash-sep space inserted on either side of the **slash-tok**; default `0 mu`

derivand-sep horizontal space added before the differentiand if the `spaced` package option is set to `1`, or before a multi-tokened differentiand if the `spaced` package option is set to `-1`; note that compact-form derivatives *always* have this space inserted; default (appropriate for an upright-fraction derivative) = `3mu plus 1mu minus 2mu`

op-symbol the operator symbol; for ordinary derivatives generally one of `d` or `\mathrm{d}`, for partial derivatives `\partial`; default = `\mathrm{d}`

op-symbol-alt if different from **op-symbol** then used in the denominator of a fraction-form derivative while `op-symbol` is used in the numerator; e.g. for the acceleration $\frac{\nabla v^i}{\mathrm{d}t}$, `op-symbol = \nabla` and `op-symbol-alt = d`; defaults to **op-symbol** default

op-order-nudge extra horizontal space added between the op-symbol and the superscripted order of differentiation in higher order derivatives; for math-italic forms compare d^2 with d^2, ∂^2 with ∂^2 where the first symbol in each case has no extra space and the second has an extra `1 mu`; since **op-symbol** defaults to an upright 'd', default = `0 mu`

var-sup-nudge extra horizontal space added between a variable in the denominator of a derivative and the superscripted order of differentiation in higher order derivatives (cf **op-order-nudge**); default = `1 mu`

multi-term-sep horizontal spacing inserted between the differentials in, for example, the denominator of a mixed partial derivative to avoid a solid cluster like $\partial x \partial y \partial z$; with the default `2 mu` this is spread a little, $\partial x \partial y \partial z$; default = `2 mu plus 1 mu minus 1 mu`

term-sep-adjust adjustment (usually a reduction) to **multi-term-sep** when differentiation in a variable occurs to an order other than 1; if, e.g.,

$\partial x^2 \partial y \partial z$ is the denominator of a mixed partial derivative in three variables, because of the superscript the spacing between ∂x^2 and ∂y is reduced by **term-sep-adjust** from the spacing between ∂y and ∂z; default = `-1 mu`

long-var-wrap to avoid ambiguity in higher order derivatives it may aid clarity to wrap multi-token variables of differentiation in parentheses; the choices are

- `dv` no wrapping, e.g. $\mathrm{d}x_i^2$ or $\partial \frac{1}{\Theta}^2$,
- `d(v)` wrap the variable only, e.g. $\mathrm{d}(x_i)^2$ or $\partial(\frac{1}{\Theta})^2$,
- `(dv)` wrap both op-symbol and variable, e.g. $(\mathrm{d}x_i)^2$ or $(\partial\frac{1}{\Theta})^2$;

default = `d(v)`

lvwrap-Ldelim left delimiter when wrapping a long variable in a higher order derivative; also applies to the left delimiter used in a jacobian; default = `\mleft (`

lvwrap-Rdelim right delimiter when wrapping a long variable in a higher order derivative; also applies to the right delimiter used in a jacobian; default = `\mright )`

lvwrap-sup-nudge horizontal adjustment to the superscript position when a multi-token variable is wrapped in (e.g.) parentheses and its order of differentiation is not `1`; default = `-2 mu`

outer-Ldelim the left member of a delimiter pair wrapping the derivative, the right member of which is subscripted to indicate a point of evaluation or variables held constant; ISO recommends parentheses for this purpose, hence default = `\left (`

outer-Rdelim the right member of a delimiter pair wrapping the derivative and subscripted to indicate a point of evaluation or variables held constant; ISO recommends parentheses for this purpose, hence default = `\right )`

elbowroom adjustment to the whitespace between **outer-Ldelim**, **outer-Rdelim** and the enclosed derivative; negative values reduce the space; default = `0 mu`

sub-nudge horizontal adjustment of the subscript's placing relative to the **outer-Rdelim**iter for a point of evaluation or variable held constant; a negative value compensates for the curving inwards of a large right parenthesis; default = `-5 mu`

op-sub-nudge horizontal adjustment of the position of the subscript in derivatives of compact form relative to the operator; since `\mathrm{d}` is the default operator, default = `0 mu`

***derivand-sep** when the derivand is appended, horizontal space added before the differentiand (derivand) depending on the setting of the `spaced` package option; default = **derivand-sep** default

***op-set-left** a choice of `true` or `false` indicating whether the op-symbol is left-aligned or not when the differentiand is appended; generally it is centred; applies only to upright-fraction forms of the derivative; default = `false`

***italic-nudge** if ***op-set-left** is `true`, makes an italic adjustment in the numerator, so that the op-symbols in numerator and denominator align in the same slanting column; for `d` or `\partial` an appropriate value might be `3 mu`; because of the default `\mathrm{d}`, default = `0 mu`

***inner-wrap** when the differentiand is appended, a choice of `true` or `false` dictating whether the differential operator is wrapped in parentheses, as here $(\partial/\partial x)\,F(x,y)$, or not; for a slash-fraction derivative `true` is an appropriate default, but the overall default, appropriate for an upright-fraction derivative, = `false`

***inner-Ldelim** if ***inner-wrap** is `true`, the left member of a delimiter pair around the differential operator; default = `(`

***inner-Rdelim** if ***inner-wrap** is `true`, the right member of a delimiter pair around the differential operator ; default = `)`

***outer-Ldelim** if ***inner-wrap** is `true`, the left member of a delimiter pair around both the differential operator and appended differentiand, the right member of which may be subscripted to indicate a point of evaluation or variables held constant; to avoid too many parentheses, given the default values of ***inner-Ldelim**, ***inner-Rdelim**, default = `\bigl [`

***outer-Rdelim** if ***inner-wrap** is `true`, the right member of a delimiter pair around the differential operator and appended differentiand; may be subscripted to indicate a point of evaluation or variables held constant; to avoid too many parentheses, given the default values of ***inner-Ldelim**, ***inner-Rdelim**, default = `\bigr ]`

***sub-nudge** if ***inner-wrap** is `true`, horizontal adjustment of the subscript's placing relative to the ***outer-Rdelim**iter for a point of evaluation or variable held constant; a negative value compensates for the curving inwards of a large right parenthesis; since the default ***outer-Rdelim** is a square bracket, default = `0 mu`

3.2.1 Ordinary upright-fraction derivatives; template `DIFF`

The defaults assigned in template `DIF` are inherited by template `DIFF` without change. Template `DIFF` is therefore strictly unnecessary but, with templates `DIFS` and `DIFC` in mind, was created for the sake of a consistent naming scheme.

Table 3.3: Defaults differing from the parent template

(a) DIFS

key	default
style	/
derivand-sep	2muplus1muminus2mu
outer-Ldelim	(
outer-Rdelim	)
sub-nudge	0 mu
*inner-wrap	true

(b) DIFC

key	default
style	_
derivand-sep	1muplus1muminus1mu
multi-term-sep	1 mu
term-sep-adjust	0 mu
outer-Ldelim	\bigl (
outer-Rdelim	\bigr)
sub-nudge	-2 mu

(c) DIFFP

key	default
op-symbol	\partial
op-order-nudge	1 mu
*italic-nudge	3 mu

(d) DIFSP, DIFCP

key	default
op-symbol	\partial
op-order-nudge	1 mu

The \diff command uses the values in the DIFF template to form an upright-fraction derivative. Only keys with an 'f' in the third column of Table 3.2 are used in this process. Keys without an 'f' play no part in the process and their default values are ignored. See §5.2.2 for the complete list of *relevant* DIFF defaults.

3.2.2 Ordinary slash-fraction derivatives; template DIFS

When you use the command \difs to form a slash-fraction derivative it is the keys in template DIF with an 's' in the third column of Table 3.2 which are used. Table 3.3a records those keys used for this purpose which are assigned default values *different* from those in DIF. See §5.2.3 for the complete list of *relevant* DIFS defaults.

3.2.3 Ordinary compact-form derivatives; template DIFC

When you use the command \difc to form a compact derivative it is the keys in template DIF with a 'c' in the third column of Table 3.2 which are used. Table 3.3b records those keys used for this purpose which are assigned default values *different* from those in DIF. See §5.2.4 for the complete list of *relevant* DIFC defaults.

3.2.4 Partial derivatives; templates DIFFP, DIFSP, DIFCP

The default values given in the tables so far apply to ordinary derivatives. For *partial* derivatives, only a few defaults change. These are listed in Tables 3.3c, 3.3d. All other keys take the default values of the respective parent templates, DIFF, DIFS and DIFC.

3.3 Variant forms: the \difdef command

You may be dissatisfied with the scheme of default values listed in the preceding tables and wish to 'Re-mould it nearer to the Heart's Desire'. How to do so is discussed in §3.4 below. In *this* section it is assumed that the user is largely satisfied with the assigned defaults but has need to write an occasional derivative that deviates from the default form. For instance, to write the range of different examples displayed in the Rogues' Gallery (§1.2) I had to make extensive use of such *variant forms* of derivative. I needed forms that displayed different ways of indicating a point of evaluation, a form that showed a math-italic 'd' rather than the default upright 'd', forms that displayed different parenthesizing styles for higher-order derivatives with multi-token variable names, and so on.

The process of defining and using such variants is a two-step process. The 'using' part is easy: you simply put the name of the variant form between dots and append to the relevant \difx or \difxp command.

The 'defining' part makes use of a command \difdef,

```
\difdef{id-list}{variant-name}{key-value list}
```

which has three *mandatory* arguments:

1. id-list A comma-list of identifiers, one or more of f, s, c, fp, sp, cp, j, l distinguishing the respective templates DIFF, DIFS, DIFC, DIFFP, DIFSP, DIFCP, DIFJ and DIFL (for the last two see Chapter 4).

2. variant-name A (preferably brief) name for the variant form; it may include characters other than letters, like numbers, punctuation marks (excluding full stops), mathematical symbols like + and =, but not control sequences or active characters, nor % , # or braces.

3. key-value list A *key=value* list where the settings differ from the default settings for the relevant template or templates (as determined by the id-list).

In the preamble to the present document I have included the following definition:

```
\difdef { f } { p }
  {
    op-symbol      = \partial,
    op-order-nudge = 1 mu
  }
```

This defines a variant, with name `p`, of an ordinary upright-fraction derivative (the `f` in the first argument) that displays as a partial derivative. To use the variant simply append the name, as a dot-delimited argument, to the `\diff` command. For instance, repeating an earlier example from thermodynamics,

```
\[ \diff.p.*{\frac PT}U[V] = \diff.p.*{\frac 1T}V[U] \] ⟹
```

$$\left(\frac{\partial}{\partial U}\,\frac{P}{T}\right)_V = \left(\frac{\partial}{\partial V}\,\frac{1}{T}\right)_U$$

The effect is exactly the same as previously and it would have been possible to define `\diffp` as this variant by following the definition of the variant with the statement,

```
\NewDocumentCommand \diffp { } { \diff.p. }
```

`diffcoeff` has not followed this path, instead choosing to put the status and configurability of partial derivatives on the same footing as ordinary derivatives.

The command `\difdef` in version 5 of `diffcoeff` takes *three* mandatory arguments for defining variant forms of derivative. Do not confuse with the command `\diffdef` of earlier versions of `diffcoeff` which took *two* mandatory arguments for this purpose. The third argument is required to identify which one or more of the fraction forms `f`, `s`, `c`, `fp`, `sp`, `cp` of the commands `\difx`, `\difxp`, the variant applies to. In earlier versions this was not necessary since there was only the one primary derivative command `\diff`.

The present document comes with a number of variant definitions. These are divided into two groups. One, in the preamble, contains definitions, like the example just given, designed to illustrate various effects in this document – as in the Rogues' Gallery. These preamble definitions are listed in §5.4. The other, in the associated file `diffcoeff5.def`, contains definitions that may be of more general usefulness; these are listed in §5.3.

3.3.1 The `.def` file

A `.def` file (in `diffcoeff`) is a text file containing a list of definitions of variant derivatives after the fashion of the example above. The reason for placing such variant definitions in a file is that they can be easily transferred from document to document by means of the `def-file` package option. If the name of your `.def` file is `myfile`, then invoking `diffcoeff` with the call

```
\usepackage[def-file=myfile]{diffcoeff}
```

makes the definitions in `myfile.def` available to your current document – provided `diffcoeff` can find the file.

The question is: where to put the `.def` file? The directory of the current document is an obvious candidate and for the current document serves well,

but it does mean copying the `.def` file from directory to directory to work on *different* documents. To make a definition file available for *all* documents, place it in the `texmf` tree, preferably not the one created by your TEX distribution, but your own *personal* `texmf` tree. Provided your TEX distribution knows about your personal `texmf` tree and the files it contains, then a `.def` file placed within it will be accessible to all documents.

Personal `texmf` tree?

This is a directory created by you for 'waifs and strays' of the TEX system that are not included in standard distributions like MiKTEX or TEXLive. For instance, it is the place for personal packages designed for your own particular circumstances or preferences, and is structured like the standard `texmf` hierarchy but placed in another location so that there is no chance of its being overwritten when your TEX distribution is updated. But that distribution needs to be alerted to the existence of your personal `texmf` tree and any new files added to it. For MiKTEX, open the MiKTEX console, click on Settings (in the column on the left) and then the Directories tab. Click the + button and navigate to your personal texmf tree to add it to the MiKTEX search path, using the arrow keys to place it as high in the search path as possible. Having added it, you will then need to refresh the filename database by clicking on the Tasks menu and selecting the obvious entry. I am not familiar with TEXLive but presume an analogous process will apply there.

`diffcoeff.def`

In earlier versions of `diffcoeff`, if there was no explicit `def-file=<filename>` package option statement, then a file `diffcoeff.def` was searched for and if found loaded. This is no longer the case. Version 5 of `diffcoeff` searches for a `.def` *only if it is explicitly named* in a package option statement. (This decision was made at least in part to avoid conflict with a `diffcoeff.def` file from an earlier version of `diffcoeff` tucked away in some non-obvious place and producing obscure errors in the current version 5.)

3.3.1.1 Log file message

If the `.def` file named in the package option statement cannot be located by TEX, a message to that effect is sent to the terminal and log file, but `diffcoeff` continues loading.

3.3.2 Examples of variants

The dot-delimited name argument must always be the *first* argument of the `\difx` or `\difxp` command, even preceding an asterisk (star) signalling 'append the differentiand'. Now for some examples.

Acceleration In tensor calculus acceleration is sometimes written $\nabla v^i/\mathrm{d}t$, where different operator symbols occur in numerator and denominator. In the preamble to this document I have included the definition

```
\difdef { f, s } { n }
  {
    op-symbol     = \nabla,
    op-symbol-alt = \mathrm{d}
  }
```

to give both upright- (the `f`) and slash-fraction (the `s`) forms of the acceleration. Appending tne dot-delimited name `n` to `\difs`, `$ \difs.n.{v^i}t $` $\Longrightarrow$ $\nabla v^i/\mathrm{d}t$, and appending the dot-delimited name `n` to `\diff`,

`\[ \diff.n.{v^i}t. \]` $\Longrightarrow$

$$\frac{\nabla v^i}{\mathrm{d}t}.$$

Detached subscripts To show the effect of the key `sub-nudge`, the preamble contains the definition

```
\difdef { fp } { wsp }
  { sub-nudge = 0 mu }
```

The name `wsp` is a contraction of 'whitespace'. The defintion applies only to an upright-fraction form of partial derivative (the `fp` in the optional argument). By giving `sub-nudge` a zero value in the `wsp` variant, the subscript is cast adrift (perhaps to float away?) on a sea of whitespace. The default setting, `sub-nudge=-5 mu`, maintains visual connection between subscript and right parenthesis:

`\[ \diffp.wsp.Fx[0],\quad \diffp Fx[0] \]` $\Longrightarrow$

$$\left(\frac{\partial F}{\partial x}\right)_0, \quad \left(\frac{\partial F}{\partial x}\right)_0$$

Lagrange's equations An earlier example used Lagrange's equations of motion, which showed a problem with the amount of whitespace introduced before a differentiand bounded by a `\left`, `\right` pair. The file `diffcoeff5.def` contains the definition

```
\difdef { f, fp } { *0 }
  {
    *derivand-sep = 0 mu ,
    outer-Ldelim  = \mleft ( ,
    outer-Rdelim  = \mright )
  }
```

The first thing to notice is that the name of the variant, `*0`, is not formed from letters (there are other examples below). Now Lagrange's equations are rendered (just right to my eye!)

```
\[ \diffp L{q_{k}}-\diff.*0.**t{\diffp L{\dot{q}_{k}}[]}=0 \]
```

$$\frac{\partial L}{\partial q_k} - \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}_k}\right) = 0.$$

3.3.2.1 Editing variant forms

Bug in `xtemplate` If you wish to successfully *edit* a variant form that has (already) been defined – by you or in `diffcoeff` or in the `.def` file – then you will need a version of `xtemplate` from 2022-12-17 or later. Earlier versions contained a bug that didn't otherwise affect the workings of `diffcoeff` but did prevent changes being made to already defined variants.

You may wish to edit an already defined variant form – perhaps to give a *negative* value to `*derivand-sep` in the last example. You don't need to repeat the full definition. It suffices to change the setting only of the relevant key or keys:

```
\difdef { f, fp } { *0 }
  { *derivand-sep = -3 mu }
```

which gives for Lagrange's equations (too tight to my eye!)

$$\frac{\partial L}{\partial q_k} - \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}_k}\right) = 0.$$

3.3.2.2 Parenthesizing multi-token variables

To illustrate the different modes of parenthesizing 'long' variables in higher order derivatives, I have put these two definitions in `diffcoeff5.def`:

```
\difdef { f, fp } { (dv) }
  { long-var-wrap = (dv) }

\difdef { f, fp }{ dv }
  { long-var-wrap = dv }
```

The three possibilities for wrapping multitoken variables can now be illustrated:

```
\[ \diffp[2]f{x^{i}},\quad
   \diffp.dv.[2]f{x^{i}},\quad
   \diffp.(dv).[2]f{x^{i}} \]
```

$\Longrightarrow$

$$\frac{\partial^2 f}{\partial (x^i)^2}, \quad \frac{\partial^2 f}{\partial {x^i}^2}, \quad \frac{\partial^2 f}{(\partial x^i)^2}$$

To my eye parenthesizing seems unnecessary in this case, but in the following desirable (as in the first, default, setting):

```
\[ \diffp[2]F{1/T},\quad
   \diffp.dv.[2]F{1/T},\quad
   \diffp.(dv).[2]F{1/T} \]
```

$\Longrightarrow$

$$\frac{\partial^2 F}{\partial (1/T)^2}, \quad \frac{\partial^2 F}{\partial {1/T}^2}, \quad \frac{\partial^2 F}{(\partial 1/T)^2}$$

3.3.2.3 Point of evaluation

Although ISO recommends subscripting parentheses to indicate a point of evaluation, some (like the author) prefer to subscript a vertical rule and save subscripted parentheses for the case of variables held constant in partial derivatives. The file `diffcoeff5.def` contains the definition

```
\difdef { f, fp, s, sp } { | }
  {
    outer-Ldelim  = \left . ,
    outer-Rdelim = \right |,
    sub-nudge   = 0 mu
  }
```

where the 'pipe' character is used for the name of the variant.

```
\[ \diffp.|.{F(x,y)}x[x=1] \] $\Longrightarrow$
```

$$\left.\frac{\partial F(x,y)}{\partial x}\right|_{x=1}$$

For slash fractions, I think parentheses give a better result than a vertical rule:

```
$ \difs yx[0],\quad \difs.|.yx[0] $
```
$\Longrightarrow (\mathrm{d}y/\mathrm{d}x)_0, \quad \mathrm{d}y/\mathrm{d}x|_0$

They tie the whole expression together. However, it is easy to create expressions that suffer from 'parenthesis overload':

```
$ \difs{F(x)}{(2x)}[x=0],\quad\difs.|.{F(x)}{(2x)}[x=0] $
```
$\Longrightarrow (\mathrm{d}\,F(x)/\mathrm{d}(2x))_{x=0}, \quad \mathrm{d}\,F(x)/\mathrm{d}(2x)|_{x=0}$

The vertical rule is better in this case, but best of all in this case (to my eye) is the use of *square* brackets. The file `diffcoeff5.def` contains the definition

```
\difdef { f, fp, s, sp } { ] }
  {
    outer-Ldelim = \left [ ,
    outer-Rdelim = \right ],
    elbowroom    = 1 mu,
    sub-nudge    = 0 mu
  }
```

giving the result

$$\texttt{\$ \difs.].\{F(x)\}\{(2x)\}[x=0] \$} \Longrightarrow [\mathrm{d}\,F(x)/\mathrm{d}(2x)]_{x=0},$$

which both avoids 'parenthesis overload' and is 'tied together' by the square brackets (and at least gives a nod in the direction of the ISO standard).

3.3.2.4 Upright text-style derivatives

`diffcoeff` assumes that derivatives of upright-fraction form will be used mainly in display-style expressions and that the slash form will be used mainly for inline use. But if one does want to use the fraction form in an inline expression, then `$ \diffp ST $` displaying as $\frac{\partial S}{\partial T}$ is fine, but adding a trailing optional argument, `$ \diffp ST[V] $`, to indicate (in the present example) a variable held constant is not: $\left(\frac{\partial S}{\partial T}\right)_V$. Clearly the subscript is too close to the right parenthesis and (to my eye) there is too much 'elbowroom' between the derivative and the enclosing parentheses. Hence the file `diffcoeff5.def` contains the following definition for text-style upright fraction derivatives:

```
\difdef { f, fp } { t }
  {
    style           = tfrac  ,
    derivand-sep    = 1 mu plus 1 mu minus 1 mu,
    multi-term-sep  = 0 mu   ,
    term-sep-adjust = 0 mu   ,
    wrap-sup-nudge  = 0 mu   ,
    outer-Ldelim    = \bigl (,
    outer-Rdelim    = \bigr ),
    elbowroom       = -2 mu  ,
    sub-nudge       = -3 mu
  }
```

With this definition, the variant form `$ \diffp.t.ST[V] $` displays as $\left(\frac{\partial S}{\partial T}\right)_V$. The subscript now is better positioned and there is a better fit between parentheses and derivative. Note that the `style=tfrac` entry in the definition means `\diffp.t.` will not scale in a display-style environment and may give a ridiculous result if used inappropriately:

```
\[ \frac AB\diffp.t.yx \] ⟹
```

$$\frac{A}{B}\frac{\partial y}{\partial x}$$

For a non-scaling display-style derivative using `\dfrac`, given the defaults in templates `DIFF` and `DIFFP`, the definition would be much simpler,

```
\difdef { f, fp } { d } { style = dfrac }
```

but there seems little point in so doing.

3.3.2.5 Slash-fraction styles

The default slash-fraction form `$ \difs yx $` displaying as $\mathrm{d}y/\mathrm{d}x$ does not scale. It is intended for inline use, but sometimes you may want a slash fraction of a different size – perhaps a fraction is present in the differentiand or in the variable of differentiation. The file `diffcoeff5.def` contains a definition of a scaling slash fraction (name `0`) and a slightly larger-than-default slash fraction (name `1`):

```
\difdef { s, sp } { 0 }
  {
    style         = auto     ,
    outer-Ldelim  = \left [  ,
    outer-Rdelim  = \right ] ,
    sub-nudge     = 0 mu     ,
    *inner-Ldelim = \mleft ( ,
    *inner-Rdelim = \mright ),
    *outer-Ldelim = \left [  ,
    *outer-Rdelim = \right ]
  }
\difdef { s, sp } { 1 }
  {
    style          =  big   ,
    outer-Ldelim   = \bigl (,
    outer-Rdelim   = \bigr ),
    sub-nudge      = -2   mu,
    *inner-Ldelim  = \bigl (,
    *inner-Rdelim  = \bigr ),
    *outer-Ldelim  = \bigl [,
    *outer-Rdelim  = \bigr ]
  }
```

The names arise from the sequence `\big/`, `\Big/`, `\bigg/`, `\Bigg/`, hence `1`, `2`, `3`, `4`, which leaves `0` for the scaling form (which is built around `\left.`, `\middle/`, `\right.`). `diffcoeff5.def` does not contain definitions for the `2`, `3`, `4` variants, only the two shown, because the larger sizes give ridiculous results. For the scaling variant, it is also easy to produce eyesores:

```
\[ \difsp.0.{\frac1Y}{\frac1X} \] ⟹
```

$$\partial \frac{1}{Y} \Big/ \partial \frac{1}{X}$$

But for small size increases, the results can be pleasing. To the author's eye, both `0` and `1` variants give better results than the default:

`$ \difsp.1.{F(x,y)}{\tfrac1x}[0] $` $\Longrightarrow \left(\partial F(x,y)/\partial \tfrac{1}{x}\right)_0$

`$ \difsp.0.{F(x,y)}{\tfrac1x}[0] $` $\Longrightarrow \left[\partial F(x,y)/\partial \tfrac{1}{x}\right]_0$

`$ \difsp{F(x,y)}{\tfrac1x}[0] $` $\Longrightarrow (\partial F(x,y)/\partial \tfrac{1}{x})_0$

Subscripted *square* brackets are chosen for the scaling variant so that the setting `sub-nudge=0 mu` is appropriate at all scales. They provide good visual contrast with the parentheses of $F(x,y)$.

3.3.2.6 Compact–form derivatives

Two styles are available for compact-form derivatives, derivative style, `style=_`, and differential style, `style=dl`. The first is the default style; the orders of differentiation are applied to the operator symbol, in this example `\partial`:

`$ \difcp[3,2]f{x,y,z} $` $\Longrightarrow \partial_x^3\partial_y^2\partial_z f$

What happens if no differentiation variable is specified, only an empty brace pair?

`$ \difc[3]f{} $` $\Longrightarrow \mathrm{d}^3 f$

Note that this is the behaviour from version 5.2 of `diffcoeff`. In version 5.1, a brace pair alone would halt compilation; a *nested* brace pair, `{{}}`, was required.

Suppose now we define a variant form (as is done in `diffcoeff5.def`),

```
\difdef { cp } { dl } { style = dl }
```

and use it to form a similar expression but without the differentiand, an empty argument in its place:

`$ \difcp.dl.[3,2]{}{x,y,z} $` $\Longrightarrow \partial x^3\partial y^2\partial z$

In this `dl` style, the orders of differentiation are applied to the *variables*. This allows discussion of, for example, the denominator of a mixed partial derivative – perhaps a remark about minutiae of spacing. (See §4.1 on differentials which perhaps more conveniently similarly allow the writing of, for example, $\mathrm{d}x^3$.)

3.3.2.7 D, \delta, \Delta derivatives

In introductory calculus texts a derivative-like symbol is created with the lowercase Greek delta, δ . An uppercase Greek delta, Δ, is often used in a derivative-like symbol for an average. In fluid dynamics the *material* (also *substantive* or *total*) derivative uses an uppercase D in place of d. Texts on differential equations often use a D operator. The file `diffcoeff5.def` contains the definitions

```
\difdef { f, s } { gd }
  { op-symbol = \delta }
\difdef { f, s } { gD }
  { op-symbol = \Delta }
\difdef { f, s } { D }
  { op-symbol = \mathrm{D} }
\difdef { c } { bD }
  {
    op-symbol    = \mathbf{D},
    op-sub-nudge = -2mu
  }
```

(where the 'g' in the first two suggests 'greek'), meaning one can write expressions like `$ \difs.gd.yx $` $\Longrightarrow \delta y/\delta x$, or `$ \difs.gD.yx $` $\Longrightarrow \Delta s/\Delta t$ (for the average speed), or

`\[ \diff.D.{\rho}t=\diffp\rho t + \mathbf{u\cdot}\nabla\rho \]` $\Longrightarrow$

$$\frac{\mathrm{D}\rho}{\mathrm{D}t} = \frac{\partial\rho}{\partial t} + \mathbf{u}\cdot\nabla\rho$$

for the total derivative of ρ (perhaps in fluid dynamics), or

`$ \difc.bD.[2]y{x\,}+2\difc.bD.y{x\,}-4=0 $` $\Longrightarrow \mathbf{D}_x^2\, y + 2\mathbf{D}_x\, y - 4 = 0$

for an example in the study of differential equations.

3.3.3 Other notations

`diffcoeff` and this document are about defining *derivatives* but it is worth pointing out that other notations can be built from the `diffcoeff` constituents, in particular from the slash fraction forms. For example, some other token than `/`, or indeed series of tokens, can be used to link numerator and denominator. It could be `\vert` or `\Vert`, displaying as | and ‖ respectively, or `\otimes` (requiring for example `\usepackage{stmaryrd}` in the preamble), displaying as $\otimes$, or the sequence of tokens `\otimes\ldots\otimes` displaying as $\otimes \ldots \otimes$. The critical key is `slash-tok`, with possible extra spacing on either side through the key `slash-sep`. Or, one may want to void the `op-symbol` key by giving it an empty value or do something like `op-symbol=\mathbf`, or give `outer-Ldelim`, `outer-Rdelim` special values, e.g.,`\langle`, `\rvert`.

In the preamble I have included the following definition, in order to mimic the `\Braket` command of the `braket` package,

```
\difdef{ s }{ bk }
  {
    slash-tok = ,
    op-symbol = ,
    multi-term-sep = 3mu\middle|\mskip3mu ,
```

```
    outer-Ldelim = \left\langle ,
    outer-Rdelim = \right\rangle
  }
```

and supplemented it with the definition:

```
\NewDocumentCommand \Braket { m }
  {
    \difoverride {\negmu}
    \difs.bk.{}{#1}[]
  }
```

Testing the new command, `\Braket`, gives this display:

```
\[ \Braket{\phi,\diffp[2]{}t,\psi} \] ⟹
```

$$\left\langle \phi \middle| \frac{\partial}{\partial t^2} \middle| \psi \right\rangle$$

Comparison with the `\Braket` command of the `braket` package, which uses `|` as the separator in the argument rather than commas, shows the displayed results to be the same (as far as I can judge).

3.4 Defaults: setting your own

The use of variant forms of derivative assumes the user is reasonably satisfied with the default values of the various templates. The user may not be. You may want different defaults. That is again accomplished by means of the `\difdef` command. The procedure is identical with that for defining a variant except that *no* `variant-name` is supplied; an empty argument is used instead.

For example, suppose you wish to indicate a point of evaluation for ordinary upright-fraction derivatives by means of a subscripted vertical rule rather than parentheses. In §3.3.2.3 we have seen how to create a variant form with this property but now we want to make it the default in the template `DIFF`. That is easy – simply omit any content from the second argument:

```
\difdef { f } {}
  {
    outer-Ldelim = \left . ,
    outer-Rdelim = \right |,
    sub-nudge    = 0 mu
  }
```

If this is placed in the preamble of your document or in your `.def` file (see §3.3.1) then the command `\diff yx[0]` will produce

$$\left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_0$$

by default. By leaving the second argument empty the `\difdef` command has changed the default value of those templates indicated by the list of identifiers in the first argument – in the present case only the template `DIFF`. In particular, note that the new default is *not* inherited by `DIFFP`. Inheritance occurs *only* at load time. How to change defaults that will be inherited is discussed below in §3.4.1.

If you wish to change other defaults of other templates follow the same procedure. In the first argument of the `\difdef` command insert a comma-list of the derivative identifiers (`f`, `s`, `c`, `fp`, `sp`, `cp`, `j` or `l`) that you want the new defaults to apply to, *leave the second argument empty*, and in the third argument provide the *key=value* list of new defaults.

For instance, you might prefer math-italic 'd's rather than the upright ISO recommendation, and you want this to apply across *all* ordinary-derivative templates. The most straightforward way of achieving that would be through the definition

```
\difdef { f, s, c, l } {}
  {
    op-symbol      = d ,
    op-order-nudge = 1 mu
  }
```

which includes the list of identifiers `f, s, c, l` in the first argument (the `l` referring to the template `DIFL` of the differential – see §4.1) and leaves the second argument empty. Thereafter, all ordinary derivatives will be graced with math-italic rather than upright 'd's.

3.4.1 Changing defaults in `DIF`

You might wonder if this last effect could not have been obtained more simply by changing the default in the 'primogenitor' template `DIF` – perhaps leave both *first* and second arguments empty in the `\difdef` command. That, however, has no effect. The `\difdef` command does nothing if the first argument is empty. Inheritance occurs only once, at time of birth – *load time* – and not thereafter. If you want to make a change affecting a number of templates by changing a default in `DIF`, then it has to be done at the time when `diffcoeff` is loaded.

There are two ways to do this. The first is to create a text file with the specific name `diffcoeff.DIF` with the desired settings. For example, if we want math-italic 'd's and a subscripted vertical rule for points of evaluation, then the file might look like

```
op-symbol      = d,
op-order-nudge = 1 mu,
outer-Ldelim   = \left . ,
outer-Rdelim   = \right |,
sub-nudge      = 0 mu
```

By locating the file in a place where your TEX distribution can find it – either in the directory of the current document or in your personal `texmf` tree (see the earlier discussion at §3.3.1, and in particular the need to alert your TEX distro to the presence of the file) – `diffcoeff.DIF` will be read at load time and the new defaults not only incorporated into template `DIF` but inherited by all child and grandchild templates unless explicitly countermanded (for example by `op-symbol = \partial` and similar statements in the definitions of those templates).

The second method is to use the package option `DIF`. For instance loading `diffcoeff` with the call

```
\usepackage
  [ DIF =
      {
        op-symbol = d,
        op-order-nudge  = 1 mu,
        outer-Ldelim    = \left . ,
        outer-Rdelim    = \right |,
        sub-nudge       = 0 mu
      }
  ]{diffcoeff}
```

will overwrite the built-in defaults with these new values, which will be inherited by child (and grandchild) templates unless explicitly countermanded. Notice that since `DIF` is a comma list it requires braces around the list of *key=value* statements.

If both methods of changing the template `DIF` are employed, the order of use is, first, read and act on the file `diffcoeff.DIF`, then read and act on the package option `DIF`. (In other words, to avoid complicating the preamble, preferably use the file `diffcoeff.DIF`; use the package option `DIF` only for fine-tuning – perhaps a setting specific to that particular document.)

Chapter 4

Differentials and jacobians

In addition to the six derivative commands, `\difx` and `\difxp`, the `diffcoeff` package has two further commands, `\dl` and `\jacob`, for writing differentials and jabobian determinants respectively. These commands use the settings of the templates `DIFL` and `DIFJ`, and both are correspondingly configurable by means of the `\difdef` command.

4.1 Differentials

Forms like $\mathrm{d}x$ occur not only as components of derivatives but also in other contexts like the expression for a total differential,

$$\mathrm{d}P = \frac{\partial P}{\partial x}\,\mathrm{d}x + \frac{\partial P}{\partial y}\,\mathrm{d}y + \frac{\partial P}{\partial z}\,\mathrm{d}z,$$

or in integrals, like $\int \sin x\,\mathrm{d}x$, or multi-variable integrals like

$$\iiint_{-\infty}^{\infty} V(x,y,z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z.$$

They also occur in differential geometry and elsewhere in the form of line elements like

$$\mathrm{d}x^2 + \mathrm{d}y^2 + \mathrm{d}z^2 \quad \text{and} \quad c^2\mathrm{d}t^2 - \mathrm{d}x^2 - \mathrm{d}y^2 - \mathrm{d}z^2.$$

Surely we want the 'd's in these expressions to correspond to their form (upright or math italic) in derivatives? To this end, `diffcoeff` provides a command `\dl` to write the 'd' in a differential in a manner consistent with the default form used in derivatives. In the present document, the default form is upright and so

`$ \dl x $` $\Longrightarrow \mathrm{d}x$.

To use the command before a multi-token variable of differentiation, put the variable in braces:

```
$ \dl{\vec{x}},\quad \dl{\mathbf{x}} $
```
$\Longrightarrow \mathrm{d}\vec{x}, \quad \mathrm{d}\mathbf{x}.$

For the triple integral above, writing the differentials required not three but just the *one* command:

```
$ \dl{x,y,z} $
```
$\Longrightarrow \mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z.$

To write the line elements I made use of a dot-delimited argument producing a variant form of the differential (see below §4.1.3.1):

```
$ \dl.+.{x,y,z}^2 $
```
$\Longrightarrow \mathrm{d}x^2 + \mathrm{d}y^2 + \mathrm{d}z^2,$

```
$ c^2\dl.-.{t,x,y,z}^2 $
```
$\Longrightarrow c^2\mathrm{d}t^2 - \mathrm{d}x^2 - \mathrm{d}y^2 - \mathrm{d}z^2.$

(If what you want is not $\mathrm{d}x^2$ but $\mathrm{d}^2 f$, with the superscript attached to the `d`, see §3.3.2.6.)

4.1.1 Template `DIFL`

The differential command `\dl` gives access to a template `DIFL` which inherits the default values of the fundamental template `DIF` with the (few) changes shown in Table 4.1. Note that the `style` key is fixed at the value `dl`; it cannot be changed. The `outer-Ldelim` key inserts a small space before the differential; the `outer-Rdelim` key does nothing. For the differential, both `outer-Ldelim` and `outer-Rdelim` are *always inserted*. This differs from the derivative for which `outer-Ldelim` and `outer-Rdelim` are inserted only if there is a trailing optional argument. It is as if the differential command `\dl` had a built-in empty trailing optional argument.

Table 4.1: `DIFL` defaults

key	default	comment
style	`dl`	locked
outer-Ldelim	`\,`	
outer-Rdelim		

That so few of the `DIF` defaults are changed in `DIFL` indicates that much of the machinery of derivative formation is irrelevant for forming a differential. A list of *relevant* keys for the creation of differentials – those that have some effect on the appearance of the thing – can be found at §5.2.6.

4.1.2 Syntax and options

If all options are present the differential command has the syntax

```
\dl.name.[order-spec]{variable(s)}^{exponent}
```

where the arguments have the following significance:

1. `name` (optional) A dot-delimited name to distinguish a variant form (non-default form) of differential; see §4.1.3 below.

2. `order-spec` (optional) The power or comma-list of powers to which the differential or differentials will be raised. If all powers are `1` then no specification is needed; indeed, if fewer powers are specified than there are variables, all 'missing' powers are assumed to be `1`; see the discussion for mixed partial derivatives, §2.2.5.

3. `variable(s)` (mandatory) The variable or comma-list of variables the differential operator applies to. `\dl x`, `\dl{\vec{x}}`, `\dl{x,y,z}` are all valid variable specifications, displaying as $\mathrm{d}x$, $\mathrm{d}\vec{x}$ and $\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$ respectively.

4. `exponent` (optional) An exponent to which all differentials will be raised; overrides the `order-spec` ; see §4.1.3.1 for examples of use.

Only the third argument is mandatory, although it may be empty.

4.1.3 Variant forms of differential

The first argument of the differential command `\dl` is the optional `name` which is used – like the corresponding argument in the derivative commands – to define *variant forms*.

To create such variant forms, the `\defdif` command is again used, but with `l` (lowercase `L`) used as the identifier in the first argument. For example, you may want a 'partial' differential, using `\partial` in place of `d`. It seems natural to give this the name `p`:

```
\difdef { l } { p }
  { op-symbol = \partial }
```

In fact just this definition can be found in the file `diffcoeff5.def`, so that

`$ \dl.p.x $` $\Longrightarrow \partial x$

which is seven keystrokes in all versus ten (space included) for `\partial x`. Defining `\dlp` by writing

`\NewDocumentCommand \dlp {} { \dl.p. }`.

saves another keystroke. However, I doubt the few keystrokes saved justify the trouble of defining such a variant. The real reason one might do so is to ease the writing of expressions like $\partial x^3 \partial y^2 \partial z$ – perhaps in a document like the present one to discuss the minutiae of spacing in the denominators of mixed partial derivatives.

`$\dl.p.[3,2]{x,y,z}$` $\Longrightarrow \partial x^3 \partial y^2 \partial z$

As you can see from the example, just as for mixed partial derivatives, if more than one variable is specified but the `order-spec` contains fewer than that number of entries, `diffcoeff` assumes the missing entries are `1`.

A second example of a variant form of differential is provided by the definition

```
\difdef { l } { b }
   { op-symbol = \mathrm{d}\mathbf }
```

which can be found in the file `diffcoeff5.def`. If you distinguish vectors, say, by boldface type, then you can avoid writing `\mathbf` for differentials of vectors by using the variant form `\dl.b.`:

`$ \dl.b.x,\quad \dl.b.{x,y,z} $` $\Longrightarrow \mathrm{d}\mathbf{x}, \quad \mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{y}\,\mathrm{d}\mathbf{z}.$

4.1.3.1 Line elements

Variant forms can be used to write line elements of Pythagorean or Minkowskian form. The definition

```
\difdef { l } { + }
  {
    multi-term-sep  = 0 mu +,
    term-sep-adjust = 0 mu  ,
    outer-Ldelim    =
  }
```

which can be found in the file `diffcoeff5.def`, inserts a `+` sign between terms in the variable specification. Notice that the value assigned to the key `multi-term-sep` begins with `0 mu`. A dimension here *initially* is essential. Also note that the thin space inserted by default before a differential by means of the `outer-Ldelim` setting is now removed. But the intriguing feature of the definition is what follows the `0 mu` in the `multi-term-sep` value: a `+` sign. Applying this variant to `{x,y,z}` the result is $\mathrm{d}x + \mathrm{d}y + \mathrm{d}z$, which may be mildly interesting but definitely becomes so when we add an exponent to the variable spec.:

`$ \dl.+.{x,y,z}^2 $` $\Longrightarrow \mathrm{d}x^2 + \mathrm{d}y^2 + \mathrm{d}z^2.$

The exponent acts as if an order specification `[2,2,2]` had been included. If an order specification *is* included, whatever the values listed, the trailing exponent overrides it.

Similarly, the file `diffcoeff5.def` contains an identical definition save that the plus sign is replaced by a minus. This enables the writing of a Minkowski metric:

`$ c^2\dl.-.{t,x,y,z}^2 $` $\Longrightarrow c^2\mathrm{d}t^2 - \mathrm{d}x^2 - \mathrm{d}y^2 - \mathrm{d}z^2.$

4.1.4 Changing defaults

To change the *default* values of the `DIFL` template use the `\difdef` command but leave its second argument, the `name` argument, empty. For instance if you want slightly less space by default before a differential than the thin space (`\,` or `3 mu`) specified in the `DIFL` template – say you want `2 mu` – then write

```
\difdef { l } {} { outer-Ldelim = \twomu }
```

and ensure that this is in your `.def` file or in the preamble of your document. If you want a rubber length, say `3 mu plus 1 mu minus 2 mu` (which can also be written more compactly as `3muplus1muminus2mu`), then write (notice the `\mskip`)

```
\difdef { l } {}
  { outer-Ldelim = \mskip 3muplus1muminus2mu }
```

The crucial point is to leave the second argument of `\difdef`, the `variant name`, empty. That changes the *default* values in `DIFL` of the keys listed in the third argument of `\difdef`.

4.1.5 Rationale

But why bother with the differential command at all? It only seems to complicate the simple typing of `d` followed by `x`. Admittely typing `\dl x` requires fewer keystrokes than typing `\mathrm{d}x` (or even `\mathrm dx`), but there are other, more substantive, reasons why one might prefer an explicit command.

1. *Consistency* with the derivative.
2. *Spacing* is inserted automatically before the differential, and between differentials in (e.g.) multiple integrals.
3. *Parsing integrals* for some other package or program is much easier to do when looking for a concluding differential command `\dl` than when looking for `d` or `\mathrm{d}` or `\mathnormal{d}` (or whatever) followed by an arbitrary variable name.
4. *Configurability.* There are values other than the defaults that can be given to keys to give novel effects for variant forms of differential – see the examples `\dl.b.`, `\dl.+.` and `\dl.-.` above.

4.2 Jacobians

`diffcoeff` provides a command `\jacob` for writing jacobians – not the determinant as such but the symbol conventionally used to denote the determinant. For example

```
\[ \jacob{u,v,w}{x,y,z} \] ⟹
```

$$\frac{\partial(u,v,w)}{\partial(x,y,z)}$$

The comma lists can contain any number of variables, even one or none,

$$\frac{\partial(u)}{\partial(v)}, \quad \frac{\partial()}{\partial()},$$

nor need the numbers in numerator and denominator be equal. `\jacob` does *not* check such things. It is perfectly possible to form unbalanced objects like

```
\[ \jacob{u,v,w}{x,y},\quad\jacob{u,v}{x,y,z}. \] ⟹
```

$$\frac{\partial(u,v,w)}{\partial(x,y)}, \quad \frac{\partial(u,v)}{\partial(x,y,z)}.$$

Perhaps there are contexts where these are meaningful?

4.2.1 Template `DIFJ`

Jacobians are configurable. Like other commands of `diffcoeff`, `\jacob` gives access to a template, in this case `DIFJ`, which is a child of the fundamental template `DIF` and inherits most of its default values with only a few changes as shown in Table 4.2. Note that the keys `outer-Ldelim` and `outer-Rdelim` are both empty and, as with the differential, are *always inserted* – which is why they are empty by default.

Table 4.2: `DIFJ` defaults

key	default
op-symbol	`\partial`
outer-Ldelim	
outer-Rdelim	

The lack of entries in Table 4.2 is because many keys are irrelevant for forming jacobians – it doesn't matter what their default values are. For a list of *relevant* keys – ones that have some effect on the appearance of a jacobian – see §5.2.5.

4.2.2 Syntax and variant forms

The `\jacob` command has only three arguments. The syntax is simple:

```
\jacob.name. {numer} {denom}
```

The arguments have the following significance:

1. `name` (optional) The dot-delimited name of a variant form of jacobian.
2. `numer` (mandatory) A comma list of variables forming the numerator of the jacobian.

3. `denom` (mandatory) A comma list of variables forming the denominator of the jacobian.

The default form of jacobian is an upright fraction with `\partial` operators and parentheses around the variable lists in both numerator and denominator.

If you want a jacobian in, say, slash-fraction form then once again the `\difdef` command is used. The file `diffcoeff5.def` contains the definition

```
\difdef { j } { s } { style = / }
```

To access this style, use the name – which is at your discretion but here I have chosen `s` (`/` also suggests itself) – between dots after the `\jacob` command:

`$ \jacob.s.{u,v,w}{x,y,z} $` $\Longrightarrow \partial(u,v,w)/\partial(x,y,z).$

If you want to change the operator symbol from `\partial` to `D`, as I have seen used, then the definition is:

```
\difdef { j } { D } { op-symbol = D }
```

(Again the name is at your discretion but `D` seems obvious.) I have added this to the preamble of the present document, so that

`\[ \jacob.D.{u,v,w}{x,y,z} \]` $\Longrightarrow$

$$\frac{D(u,v,w)}{D(x,y,z)}.$$

If you want square brackets rather than parentheses around the variable lists, then `lvwrap-Ldelim` and `lvwrap-Rdelim` (perhaps not intuitively) are the keys to change:

```
\difdef { j } { [ }
  {
    lvwrap-Ldelim = \onemu\mleft [,
    lvwrap-Rdelim = \mright ]
  }
```

the `\onemu` giving, to my eye, better spacing between the `\partial` symbols and the left brackets. This definition, too, has been added to the preamble so that

`\[ \jacob.[.{u,v,w}{x,y,z} \]` $\Longrightarrow$

$$\frac{\partial[u,v,w]}{\partial[x,y,z]}.$$

4.2.3 Changing defaults

To change the *default* values of the `DIFJ` template leave the second argument of the `\difdef` command – the `variant-name` – empty. For instance, if you want square brackets to be your default setting, the `\difdef` command would be

```
\difdef { j } {}
  {
    lvwrap-Ldelim = \onemu\mleft [,
    lvwrap-Rdelim = \mright ]
  }
```

The only difference from the previous definition is the absence of the name from the second argument, which is now empty. If this definition were added to the preamble or to the `.def` file of your current document then writing `\jacob{u,v,w}{x,y,z}` would give the same result as obtained above with the variant `\jacob.[.{u,v,w}{x,y,z}`.

Chapter 5

Reference

For convenience I list here the commands of `diffcoeff`, the template defaults, and the files and preamble definitions associated with this document.

5.1 Commands

`\diff`, `\diffp`, `\difs`, `\difsp`, `\difc`, `\difcp` (sometimes summarised as `\difx` and `\difxp`), ordinary and partial derivatives of upright-fraction, slash-fraction and compact forms respectively, with arguments (all optional unless otherwise indicated) and their delimiters:

1. `.name.` name of variant form of derivative;
2. `*` append-differentiand switch;
3. `*` reverse order of mandatory arguments 6 and 7 when differentiand is appended, available only if first star is also present;
4. `[order(s)]` order of differentiation, or comma list of orders of differentiation (for mixed partial derivatives);
5. `<override>` total order of differentiation override (for mixed partial derivatives);
6. `{differentiand}` (mandatory) function being differentiated;
7. `{variable(s)}` (mandatory) differentiation variable or, for mixed partial derivatives, comma list of differentiation variables;
8. `[pt of eval/const vars]` point of evaluation or, for partial derivatives, variables held constant.

`\difoverride` order-override command with one mandatory argument:

1. `{total order}` total order of differentiation; may be (and generally is) empty.

`\jacob` jacobian with arguments and delimiters:

1. `.name.` (optional) name of variant form of jacobian;
2. `{numer}` (mandatory) comma list of variables forming the numerator;
3. `{denom}` (mandatory) comma list of variables forming the denominator.

`\dl` differential with arguments and delimiters:

1. `.name.` (optional) name of variant form of differential;
2. `[order(s)]` (optional) order of differential or comma list of orders of differentials;
3. `{variable(s)}` (mandatory) variable or comma list of variables;
4. `^{exponent}` (optional) exponent that overrides the `order(s)` specification, raising each differential to this power.

`\difdef` with arguments (all mandatory) and delimiters:

1. `{id(s)}` comma list of one, some or all of the identifiers `f`, `s`, `c`, `fp`, `sp`, `cp`, `j`, `l` identifying upright fraction, slash fraction and compact ordinary derivatives; upright fraction, slash fraction and compact partial derivatives, and jacobians and differentials;
2. `{name}` name for a variant form of derivative; as well as letters may include numbers and other keyboard characters, but not braces, `%` or `#`;
3. `{settings}` comma list of changed *key=value* settings.

`\negmu` insert a −1 mu space

`\nilmu` insert a 0 mu space

`\onemu` insert a 1 mu space

`\twomu` insert a 2 mu space

5.2 Templates

The following lists record the default values of the templates used by `diffcoeff`. A marginal `>` indicates where a setting differs from that in `DIF`, `>>` where a setting differs from that in `DIFF`, `DIFS` or `DIFC` as the case may be. For the latter templates, only *relevant* keys have been listed – those which affect the appearance of the derivative (or jacobian or differential).

5.2.1 DIF (primogenitor)

```
style            = frac,
slash-tok        = /,
slash-sep        = 0 mu,
derivand-sep     = 3 mu plus 1 mu minus 2 mu,
op-symbol        = \mathrm{d},
op-symbol-alt    = \KeyValue{ op-symbol },
op-order-nudge   = 0 mu,
var-sup-nudge    = 1 mu,
multi-term-sep   = 2 mu plus 1 mu minus 1 mu,
term-sep-adjust  = -1 mu,
long-var-wrap    = d(v),
lvwrap-Ldelim    = \mleft (,
lvwrap-Rdelim    = \mright ),
lvwrap-sup-nudge = -2 mu,
outer-Ldelim     = \left (,
outer-Rdelim     = \right ),
elbowroom        = 0 mu,
sub-nudge        = -5 mu,
op-sub-nudge     = 0 mu,
*derivand-sep    = \KeyValue{ derivand-sep },
*op-set-left     = false,
*italic-nudge    = 0 mu,
*inner-wrap      = false,
*inner-Ldelim    = (,
*inner-Rdelim    = ),
*outer-Ldelim    = \big [,
*outer-Rdelim    = \big ],
*sub-nudge       = 0 mu
```

5.2.2 DIFF (upright-fraction derivative)

Relevant keys and default values for template DIFF.

```
style            = frac,
derivand-sep     = 3 mu plus 1 mu minus 2 mu,
op-symbol        = \mathrm{d},
op-symbol-alt    = \KeyValue { op-symbol },
op-order-nudge   = 0 mu,
var-sup-nudge    = 1 mu,
multi-term-sep   = 2 mu plus 1 mu minus 1 mu,
term-sep-adjust  = -1 mu,
long-var-wrap    = d(v),
lvwrap-Ldelim    = \mleft (,
lvwrap-Rdelim    = \mright ),
lvwrap-sup-nudge = -2 mu,
```

```
    outer-Ldelim     = \left (,
    outer-Rdelim     = \right ),
    elbowroom        = 0 mu,
    sub-nudge        = -5 mu,
    *derivand-sep    = \KeyValue { derivand-sep },
    *op-set-left     = false,
    *italic-nudge    = 0 mu
```

5.2.2.1 DIFFP

DIFF defaults as above with these changes:

```
>>  op-symbol        = \partial,
>>  op-order-nudge   = 1 mu,
>>  *italic-nudge    = 3 mu
```

5.2.3 DIFS (slash-fraction derivative)

Relevant keys and default values for template DIFS.

```
>   style            = /,
    slash-tok        = /,
    slash-sep        = 0 mu,
>   derivand-sep     = 2 mu plus 1 mu minus 2 mu,
    op-symbol        = \mathrm{d},
    op-symbol-alt    = \KeyValue { op-symbol },
    op-order-nudge   = 0 mu,
    var-sup-nudge    = 1 mu,
    multi-term-sep   = 2 mu plus 1 mu minus 1 mu,
    term-sep-adjust  = -1 mu,
    long-var-wrap    = d(v),
    lvwrap-Ldelim    = \mleft (,
    lvwrap-Rdelim    = \mright ),
    lvwrap-sup-nudge = -2 mu,
>   outer-Ldelim     = (,
>   outer-Rdelim     = ),
    elbowroom        = 0 mu,
>   sub-nudge        = 0 mu,
    *derivand-sep    = \KeyValue { derivand-sep },
>   *inner-wrap      = true,
    *inner-Ldelim    = (,
    *inner-Rdelim    = ),
    *outer-Ldelim    = \big [,
    *outer-Rdelim    = \big ],
    *sub-nudge       = 0 mu
```

5.2.3.1 DIFSP

DIFS defaults as above with these changes:

```
>>  op-symbol        = \partial,
>>  op-order-nudge   = 1 mu
```

5.2.4 DIFC (compact derivative)

Relevant keys and default values for template DIFC.

```
>   style            = _ ,
>   derivand-sep     = 1 mu plus 1 mu minus 2 mu,
    op-symbol        = \mathrm{d},
    op-order-nudge   = 0 mu,
>   multi-term-sep   = 1 mu,
>   term-sep-adjust  = 0 mu,
>   outer-Ldelim     = \bigl (,
>   outer-Rdelim     = \bigr ),
    elbowroom        = 0 mu,
>   sub-nudge        = -2 mu,
    op-sub-nudge     = 0 mu,
    *derivand-sep    = \KeyValue { derivand-sep }
```

5.2.4.1 DIFCP

DIFC defaults as above with these changes:

```
>>  op-symbol        = \partial,
>>  op-order-nudge   = 1 mu
```

5.2.5 DIFJ (jacobian)

Relevant keys and default values for template DIFJ.

```
    style            = frac,
    slash-tok        = /,
    slash-sep        = 0 mu,
>   op-symbol        = \partial,
    op-symbol-alt    = \KeyValue{ op-symbol },
    lvwrap-Ldelim    = \mleft (,
    lvwrap-Rdelim    = \mright ),
>   outer-Ldelim     = ,
>   outer-Rdelim     = ,
    elbowroom        = 0 mu ,
    *op-set-left     = false,
    *italic-nudge    = 0 mu
```

5.2.6 DIFL (differential)

Relevant keys and default values for template DIFL.

```
    op-symbol        = \mathrm{d},
    var-sup-nudge    = 1 mu,
    multi-term-sep   = 2 mu plus 1 mu minus 1 mu,
    term-sep-adjust  = -1 mu,
>   long-var-wrap    = dv,
    lvwrap-Ldelim    = \mleft (,
    lvwrap-Rdelim    = \mright ),
    lvwrap-sup-nudge = -2 mu,
>   outer-Ldelim     = \, ,
>   outer-Rdelim     = ,
    elbowroom        = 0 mu
```

5.3 The file diffcoeff5.def

```
% file 'diffcoeff5.def'
% definitions for variant forms
% 2023/04/10
% Andrew Parsloe ajparsloe@gmail.com
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% material derivative
\difdef { f, s } { D }
  { op-symbol = \mathrm{D} }
% math italic
\difdef { f, s, c } { d' }
  {
    op-symbol      = d,
    op-order-nudge = 1 mu
  }
\difdef { f, s, c } { D' }
  {
    op-symbol  = D,
    op-order-nudge = 1 mu
  }
% Greek
\difdef { f, s } { gd }
  { op-symbol = \delta }
\difdef { f, s } { gD }
  { op-symbol = \Delta }
% spaceless appending
\difdef { f, fp } { *0 }
  {
```

```
    *derivand-sep = 0 mu     ,
    outer-Ldelim  = \mleft ( ,
    outer-Rdelim  = \mright )
  }
% tfrac, nonscalable
\difdef { f, fp } { t }
  {
    style             = tfrac  ,
    derivand-sep      = 1 mu plus 1 mu minus 1 mu,
    multi-term-sep    = 0 mu   ,
    term-sep-adjust   = 0 mu   ,
    lvwrap-sup-nudge  = 0 mu   ,
    outer-Ldelim      = \bigl (,
    outer-Rdelim      = \bigr ),
    elbowroom         = -2 mu  ,
    sub-nudge         = -3 mu
  }
% slash fractions: 0=scalable,
% 1=big, 2=Big, 3=bigg, 4=Bigg
% but > 1 gives eyesores
\difdef { s, sp } { 0 }
  {
    style         = auto     ,
    outer-Ldelim  = \left [  ,
    outer-Rdelim  = \right ] ,
    sub-nudge     = 0 mu    ,
    *inner-Ldelim = \mleft ( ,
    *inner-Rdelim = \mright ),
    *outer-Ldelim = \left [  ,
    *outer-Rdelim = \right ]
  }
\difdef { s, sp } { 1 }
  {
    style          =  big   ,
    outer-Ldelim   = \bigl (,
    outer-Rdelim   = \bigr ),
    sub-nudge      = -2 mu  ,
    *inner-Ldelim  = \bigl (,
    *inner-Rdelim  = \bigr ),
    *outer-Ldelim  = \bigl [,
    *outer-Rdelim  = \bigr ]
  }
% vrule point of evaluation
\difdef { f, fp, s, sp } { | }
  {
    outer-Ldelim = \left . ,
```

```
    outer-Rdelim = \right |,
    sub-nudge    = 0 mu
  }
% sq. bracket pt of eval.
\difdef { f, fp, s, sp } { ] }
  {
    outer-Ldelim = \left [ ,
    outer-Rdelim = \right ],
    elbowroom    = 1 mu,
    sub-nudge    = 0 mu
  }
% long var wrap
\difdef { f, fp } { (dv) }
  { long-var-wrap = (dv) }
\difdef { f, fp } { dv }
  { long-var-wrap = dv }
% compact, D operator
\difdef { c } { D }
  {
    op-symbol    = \mathrm{D},
    op-sub-nudge = -2mu
  }
\difdef { c } { D' }
  {
    op-symbol    = D,
    op-sub-nudge = -2mu
  }
% bold
\difdef { c } { bD }
  {
    op-symbol    = \mathbf{D},
    op-sub-nudge = -2mu
  }
% differential style
\difdef { c, cp } { dl }
  { style = dl }
%%%%%%%%%% differential %%%%%%%%%
% partial
\difdef { l } { p }
  { op-symbol = \partial }
% bold
\difdef { l } { b }
  { op-symbol = \mathrm{d}\mathbf }

% line elements: Pythagoras (+)
\difdef { l } { + }
```

```
  {
    multi-term-sep  = 0 mu +,
    term-sep-adjust = 0 mu  ,
    outer-Ldelim    =
  }
% Minkowski (-)
\difdef { l } { - }
  {
    multi-term-sep  = 0 mu -,
    term-sep-adjust = 0 mu  ,
    outer-Ldelim    =
  }
%%%%%%%%% jacobian %%%%%%%%%
% slash fraction
\difdef { j } { s }
  { style = / }
```

5.4 Preamble definitions

The preamble to the present document contains the command

```
\usepackage[def-file=diffcoeff5,spaced=-1]{diffcoeff}
```

and definitions:

```
% nabla in numer, d in denom
\difdef { f, s } { n }
  {
    op-symbol     = \nabla,
    op-symbol-alt = \mathrm{d}
  }
% no sub nudge (a sea of white space)
\difdef { fp } { wsp }
  { sub-nudge = 0 mu }
% align op left; no italic nudge
\difdef { f } { left0 }
  {
    *op-set-left  = true,
    *italic-nudge = 0 mu
  }
% align op left; italic nudge
\difdef { fp } { left }
  {
    op-symbol      = \partial,
    op-order-nudge = 1 mu,
    *op-set-left  = true,
```

```
    *italic-nudge = 3 mu
  }
% partial variant of \diff
\difdef { f } { p }
  {
    op-symbol      = \partial,
    op-order-nudge = 1 mu
  }
% partial, 3mu sep of terms
\difdef { fp, sp } { 3mu }
  {  multi-term-sep  = 3 mu }
% D jacobian
\difdef { j } { D }
  { op-symbol = D }
% square bracket jabobian
\difdef { j } { [ }
  {
    lvwrap-Ldelim = \onemu\mleft [,
    lvwrap-Rdelim = \mright ]
  }
% mimicking the \Braket command
% of the braket package
\difdef{ s }{ bk }
  {
      slash-tok = ,
      op-symbol =  ,
      multi-term-sep = 3mu\middle|\mskip3mu ,
      outer-Ldelim=\left\langle ,
      outer-Rdelim=\right\rangle
  }
\NewDocumentCommand \Braket { m }
  {
    \difoverride {\negmu}
    \difs.bk.{}{#1}[]
  }
```

5.5 \DeclareChildTemplate

`xtemplate` provides only a single function, `\DeclareRestrictedTemplate`, for creating a child template from a parent. *All* the keys of the child template are present in the parent. The child inherits not only the keys of the parent but the default settings of those keys. Some of those settings are 'marked' so that they cannot be changed by any *instance* of the child (the *restricted* keys) .

Unfortunately there is no similar function available in `xtemplate` at present by which one can create a child with *new* as well as restricted default values. I

found I could achieve this functionality with the following code cobbled together from publicly declared functions in `xtemplate`.

```
% Child template with both new and restricted defaults
% #1 object; #2 parent template; #3 child template;
% #4 restricted, #5 new defaults (both key=value)
\NewDocumentCommand \DeclareChildTemplate { m m m m m }
  {
    \DeclareRestrictedTemplate {#1} {#2} {#3} {}
    \EditTemplateDefaults {#1} {#3} {#5}
    \DeclareRestrictedTemplate {#1} {#3} {#3} {#4}
  }
```

The first `\DeclareRestrictedTemplate` call creates the child template `#3` from the parent template `#2`, inheriting all its keys and default values. *No* restrictions are imposed at this stage because the following `\EditTemplateDefaults` would immediately cancel them. That statement specifies the *new* defaults `#5` of the child – those that differ from the parent. The default settings `#4` of the parent that are restricted to particular values in the child are imposed by the second `\DeclareRestrictedTemplate` call through the aritifice of treating the child template `#3` as a child of itself. In that way its new defaults are not lost.

5.6 Version history

Version 5 was conceived as a new package (under the name `diffcoefx`) and only at the end, after discussion with CTAN maintainers, changed to version 5.0 of `diffcoeff`.

1. Version 5.0 (2023-01-02) of `diffcoeff`
   (a) splits the `\diff` command of version 4 into three pairs of commands: `\diff` and `\diffp` for upright-fraction derivatives; `\difs` and `difsp` for slash-fraction derivatives, and `\difc` and `\difcp` for 'compact form' derivatives;
   (b) replaces the order-override option by a new command `\difoverride` (to avoid cluttering formulas with a second square-bracket delimited optional argument before the differentiand);
   (c) adds a second star option to reverse the order of differentiand and variable(s) of differentiation when the differentiand is appended;
   (d) replaces the two-argument `\diffdef` command of earlier versions with the three-argument command `\difdef` command, the additional argument determining which one or more of the `f`, `s`, `c`, `fp`, `sp` or `cp` forms the defined variant applies to;
   (e) rewrites the differential command `\dl` which is now template-configurable (e.g. allowing easy writing of line elements like $\mathrm{d}x^2 + \mathrm{d}y^2 + \mathrm{d}z^2$);

(f) rewrites the jacobian command `\jacob` which is now template-configurable;

(g) uses ISO defaults;

(h) includes version conflict messages.

2. Version 5.1 (2023-01-16)

   (a) adds a now-redundant `ISO` package option and related version conflict message;

   (b) makes some corresponding tweaks to documentation (including this version 5 history).

3. Version 5.2 (2023-01-24)

   (a) Simplifies the treatment of the empty argument of an absent differentiation variable;

   (b) initializes (clears) two sequence variables that otherwise caused error when `scrbook` class was used;

   (c) amends documentation.

4. Version 5.3 (2023-04-10)

   (a) Fixes a bug when `\dl` was used in a particular way in `beamer` (e.g. `\[\alert{\dl x}\]`).

   (b) Provides an alternative method of specifying orders of differentiation by means of colon separators in the variable argument.

   (c) Reinstates (from v.4) the order-override option as an alternative to `\difoverride` but now angle-bracket delimited.