My funny little ODE/PDE package

Start by first having `odesandpdes.sty` downloaded in an accessible directory, or in the same directory as your overleaf main.tex, using it by inserting;

```
\usepackage{odesandpdes}
```

into the preamble, Ideally after amsmath/mathtools. There are a couple notation options, which can be set document-wide with;

```
\usepackage[notation=<option>]{odesandpdes}|
```

The options included are based off of the three most common notations (according to Wikipedia), Lagrange, Leibniz, and Newton. However, if you wish to change it on a section to section basis, the command `\setDE{notation=⟨option⟩}` will change the form of the subsequent uses.

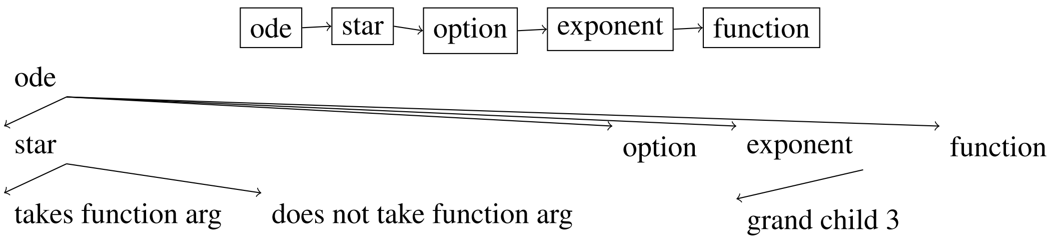
\ode
\ode* The command(s) are approached with the philosophy of of an intuitive and modular usage. The general form can be understood as;

```
\ode[<variable>]<exponent>{<function>}
\ode*[<variable>]<exponent>
```

\pde
\pde*

```
\pde[<variable>]<exponent>{<function>}
\pde*[<variable>]<exponent>
```

And the starred varients which allow one to omit the function



While this should be sufficient for most use, there are a couple of tricks incorporated into the mechanism that makes this package better than the generic `\newcommand*\ode[2][t]{⟨...⟩}` Including automatic assignment of degree and starred variants. Example;

```
\begin{align*}
\ode N(t) = N(t) * t && \ode N(t,x) = N(t,x) + \ode[x]^2 N(t,x) && \pde^2 f^2
\end{align*}
```

$$\frac{\mathrm{d}N(t)}{\mathrm{d}t} = N(t) * t \qquad \frac{\mathrm{d}N(t,x)}{\mathrm{d}t} = N(t,x) + \frac{\mathrm{d}^2N(t,x)}{\mathrm{d}x^2} \qquad \frac{\partial^2 f^2}{\partial t^2}$$

The `\ode` command will scan for an exponent between `[`⟨*variable*⟩`]` and `{`⟨*function*⟩`}`. Should there is indeed an exponent, that exponent gets ‘yoinked’ away and processed in accordance to the style of notation. In the case of Lagrange or Netwon notation, there is the `maxprimes=<integer>` option for either the package or the `\setDE{⟨option⟩}` command;

`\usepackage[maxprimes=<integer>]{odesandpdes}` *and/or*
`\setDE{maxprimes=<integer>}`

3 being the default.

There was rational in choosing to check for the exponent immediately after the macro command opposed to checking for the exponent at the end after the function. As, often you would want add a higher degree very quickly as opposed to after defining the function. `\ode^2{f(x)}` as opposed to `\ode{f(x)}^2`. As well, with the “proper” spacing, there is little need for the use of the braces, so as to help promote an easier workflow without always needed to worry about the damn brace. Not that one can not use the brace for personal taste. For the sake of parity, the `\pde` command will also take its variable in brackets.

The following examples all take the identical form, shown in the following verbatim enviroment.

```
\begin{align*}
\ode f(x)      && \ode[x]{f(x)} && \ode^1  f(x)     && \ode[x]^5 {f(x)} \\
\ode*          && \ode*[x]      && \ode*^2          && \ode*[x]^6       \\
\pde[t] {f(x)} && \pde[x] f(x)  && \pde[t]^3{f(x)}  && \pde[x]^7 f(x)   \\
\pde*[t]       && \pde*[x]      && \pde*[t]^4       && \pde*[x]^8
\end{align*}
```

`\setDE{notation=default}` *and/or* `\usepackage[notation=default]{odesandpdes}` `\setDE{notation=`

and/or `\usepackage[notation=Lagrange]{odesandpdes}`

$$\begin{array}{cccc}
f'(x) & f(x)' & f'(x) & f(x \\
f'(t) & f'(x) & f''(t) & f^{(6} \\
f(x)'_t & f'_x(x) & f(x)'''_t & f_x^{(7} \\
f'_t(t) & f'_x(x) & f_t^{(4)}(t) & f_x^{(8}
\end{array}$$

`\setDE{notation=Leibniz}` *and/or* `\usepackage[notation=Leibniz]{odesandpdes}`

$$\begin{array}{c}
\frac{\mathrm{d}f(x)}{\mathrm{d}t} \\
\frac{\mathrm{d}}{\mathrm{d}t} \\
\frac{\partial f(x)}{\partial t} \\
\frac{\partial}{\partial t}
\end{array}$$

`\setDE{notation=Newton}` *and/or* `\usepackage[notation=Newton]{odesandpdes}`

$$\begin{array}{c}
\dot{f}(x) \\
\dot{t} \\
\dot{f(x)} \\
\dot{t}
\end{array}$$

`\setDE{maxprimes=7}` *and/or* `\usepackage[maxprimes=7]{odesandpdes}`

$$\begin{array}{ccccc}
f' & f'' & f''' & f'''' & f''''' \\
\dot{f} & \ddot{f} & \dddot{f} & \ddddot{f} &
\end{array}$$

Now, because I am not an insane person, and have mostly learned how TEX deconstructs text into constitute registries and boxes, the way any sane person might. When using a non-starred version of a command, after the function is defined, you can place an '`at` ⟨*value*⟩`;`',[1] and the representation will shown according to notational convention.

```
\begin{align*}
    \ode[x]   c at 23\pi;    &= i \\
    \ode[x]^3   c at 69;     &= i \\
    \ode[x]^{69} c at L;+t &= i \\
    \ode[x]^9  c af 420;     &= i \\
    \ode[x]^6   c 13         &= i
\end{align*}
```

`\setDE{notation=Leibniz}`

$$\left.\frac{\mathrm{d}c}{\mathrm{d}x}\right|_{x=23\pi} = i$$
$$\left.\frac{\mathrm{d}^3c}{\mathrm{d}x^3}\right|_{x=69} = i$$
$$\left.\frac{\mathrm{d}^{69}c}{\mathrm{d}x^{69}}\right|_{x=L} + t = i$$
$$\frac{\mathrm{d}^9c}{\mathrm{d}x^9}af420; = i$$
$$\frac{\mathrm{d}^6c}{\mathrm{d}x^6}13 = i$$

`\setDE{notation=Newton}`

$$\dot{c}(23\pi) = i$$
$$\dddot{c}(69) = i$$
$$\overset{69}{\dot{c}}(L) + t = i$$
$$\overset{9}{\dot{c}}af420; = i$$
$$\overset{6}{\dot{c}}13 = i$$

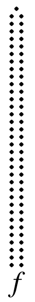
`\setDE{notation=Lagrange}`

$$c'(23\pi) = i$$
$$c'''(69) = i$$
$$c^{(69)}(L) + t = i$$
$$c^{(9)}af420; = i$$
$$c^{(6)}13 = i$$

Also the Newton and Lagrange notation is procedural;

```
\setDE{maxprimes=69}
```

$$f'''$$



Truly beautiful.

[1] heavily inspired by Tikz

In the next semester I expect to try seeing if its possible to, given that you put multiple variable in the options, to procedurally generate partials that address separate variables sequatentially.

$$\frac{\partial^2}{\partial x \partial y}$$