The numodel-plot package*

Paul Zuurbier
mail@paulzuurbier.nl

November 12, 2026

Abstract

A PGFPlots engine that auto-sizes plots to a whole number of tick intervals, supports configurable axis-label formats (IEEE-style by default; ISO 80000-1 also supported), and automatically selects label placement for 1-, 2-, and 4-quadrant graphs. Part of the numodel package suite, but can be loaded standalone as an independent PGFPlots styling layer.

Contents

1 Introduction **1**

2 Usage **2**

3 Configuration **3**
3.1 Keys 3
3.2 PGFPlots styles 5
3.3 Tick label halos 6

4 Public API **7**

5 Requirements **7**

1 Introduction

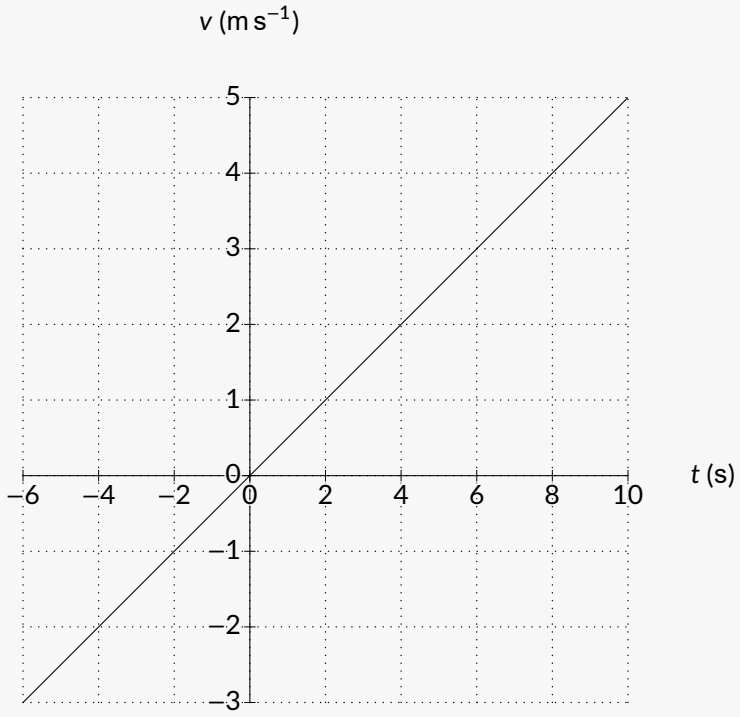
numodel-plot fills a gap between bare PGFPlots and the heavy styling required for physics-teaching material: it sizes every axis to an integer number of centimetre ticks, lays out the axis origin according to which quadrants of the coordinate plane contain data, and renders axis labels as either `quantity (unit)` (IEEE, the default) or one of four alternative conventions selectable at package level. It was extracted from a Dutch high-school physics test set where uniform plot appearance across hundreds of graphs is more valuable than per-graph tweaking, and hence adopts an opinionated default style. Users who need one-off deviations are expected to drop to plain PGFPlots with the variable macros `\xmin`, `\xmax`, … exposed by this package.

*This document corresponds to numodel-plot v0.13.1, dated November 12, 2026.

2 Usage

Minimum working example (assuming `\usepackage{numodel-plot}` in the preamble):

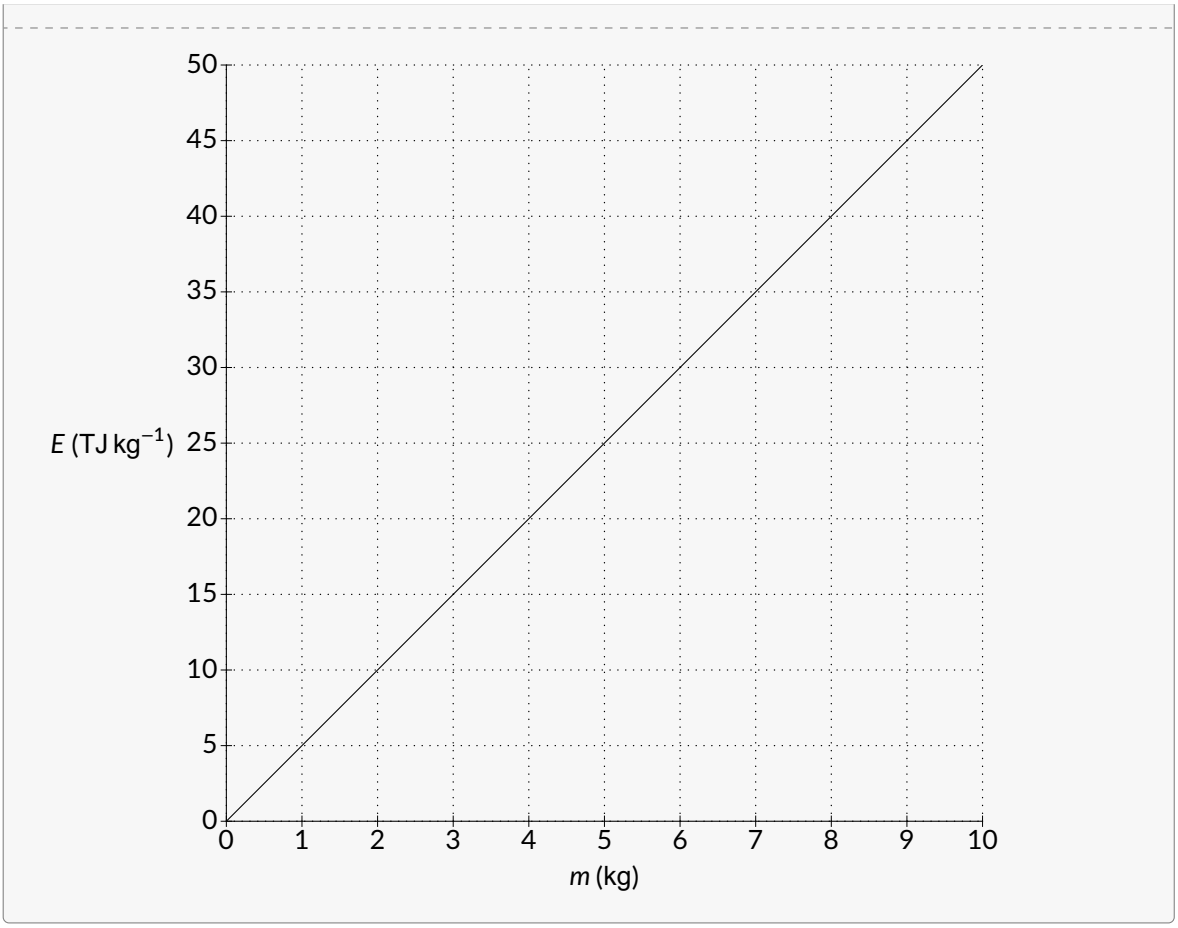
```
\def\xmin{-5}   \def\xmax{10}
\def\ymin{-3}   \def\ymax{5}
\def\xlabelqty{t}  \def\xlabelunit{\s}
\def\ylabelqty{v}  \def\ylabelunit{\m\per\s}
\drawplot{\addplot[domain=\xmin:\xmax]{0.5*x};}
```



The user sets the data range (`\xmin`…`\ymax`) and optionally a quantity symbol plus siunitx unit for each axis. `\drawplot` internally calls `\calcplotdims` to round the range to a clean tick lattice and compute the axis size in centimetres, then renders a full `tikzpicture+axis` environment whose body is the argument (one or more `\addplot` lines).

Labels are built automatically from `\xlabelqty+\xlabelunit` (and likewise for the *y*-axis). If the data magnitude exceeds 10^4 or is below 10^{-2}, an engineering factor 10^n (with n a multiple of 3) is split off and PGFPlots' own `scaled ticks` are configured so that tick numbers remain small. By default the factor is folded into an SI prefix on the unit — 5000 m becomes `s (km)` — and only when no engineering prefix fits (say 10^3 m^2, which would need $10^{1.5}$ per metre) does the label fall back to showing the power of ten itself. The `scale-format` key (section 3.1) selects between three presentations of the same factor. In the next example the data magnitude is 5×10^7 and the unit `\mega\joule\per\kilo\gram` already carries a prefix on the leading unit; with the default the injected 10^6 combines with mega into tera, giving `E (TJ/kg)` — the alternatives would read `E (10^12 J/kg)` and `E (10^6 MJ/kg)`:

```
\def\xmin{0}   \def\xmax{10}
\def\ymin{0}   \def\ymax{5e7}
\def\xlabelqty{m}  \def\xlabelunit{\kilo\gram}
\def\ylabelqty{E}  \def\ylabelunit{\mega\joule\per\kilo\gram}
\drawplot{\addplot[domain=\xmin:\xmax]{5e6*x};}
```



Users preferring full control can omit `\xlabelqty`/`\xlabelunit` and set `\xlabel`/`\ylabel` directly; the package will use them verbatim.

3 Configuration

`\numodelplotsetup` Configuration is set through a single key–value interface:

```
\numodelplotsetup{axis-label-format=ieee, grid=mm-dots}
```

3.1 Keys

`axis-label-format` Default `ieee`. Determines the notation emitted for axis labels built from `\xlabelqty` and `\xlabelunit`:

`ieee`	`v (m/s)`	IEEE (default)
`iso`	`v / (m/s)`	ISO 80000-1
`brackets`	`v [m/s]`	older physics convention
`qty-only`	`v`	quantity symbol only
`unit-only`	`m/s`	unit only

When scaling is applied (data exceeds 10^4 or below 10^{-2}), the factor is integrated into the label according to `scale-format`, e.g. `v (km/s)` for IEEE. Under `qty-only` the exponent remains in PGFPlots' scaled-tick label instead (otherwise the scale information would be lost).

`scale-format` Default `prefix`. How a scaled axis presents its power of ten in the label. Taking a y-range up to 5×10^7 in `\mega\joule\per\kilo\gram`, so that the axis is scaled by 10^6:

`prefix`	`E (TJ/kg)`	fold into an SI prefix (default)
`exponent`	`E (10^12 J/kg)`	every prefix out, one power of ten
`input`	`E (10^6 MJ/kg)`	unit as written, power of ten in front

`prefix` folds the factor into an SI prefix on (the first unit atom of) the unit: `s (km)`, `E (TJ/kg)`, `m (Mg)`. The fold respects unit powers (10^6 m^2 becomes km^2) and existing prefixes, and silently falls back to the `exponent` form whenever no engineering prefix fits (10^3 m^2, or 10^3 m^3 which would need the non-engineering deca). `exponent` pulls every prefix out of the unit — also those the user wrote — and shows a single power of ten on the bare unit: `s (10^3 m)`, `E (10^12 J/kg)`. `input` leaves the unit exactly as written and puts the axis factor in front of it: `s (10^3 m)`, `E (10^6 MJ/kg)`; use it when the unit's own prefix carries meaning that should stay visible.

`grid` Default `mm-dots` (black millimetre dots, matching engineering millimetre paper). Accepts `none`, or any PGFPlots style list which will be passed verbatim to the `numodel/grid` style.

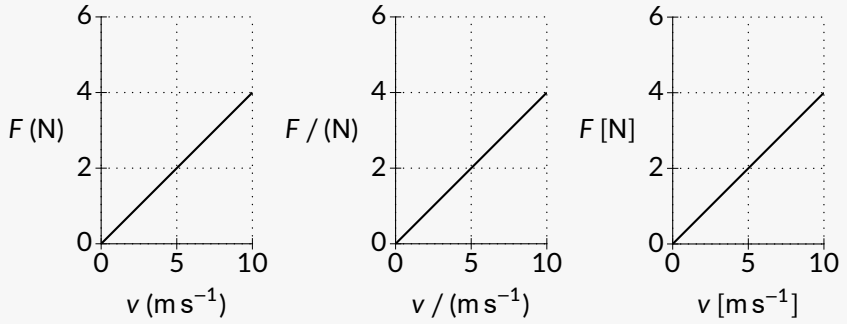
`xcmmax`, `ycmmax` Maximum axis width and height in centimetres (defaults 12 and 10).

`equal` `true` (or just `equal`) gives both axes the same amount per centimetre, for a path such as y against x: `\calcplotdims` uses the larger of the two tick distances for both axes, so the plot stays within `xcmmax` and `ycmmax`. Default `false`. Use it instead of pgfplots' `axis equal`, which stretches an axis after the ticks have been chosen.

An axis without unit (an empty `\xlabelunit`) gets no parentheses in its label, unless the axis is scaled: "N" or "N (10^3)".

The first three axis-label formats render as follows. Each plot uses `\numodelplotsetup{xcmmax=3, ycmmax=3}` so the axis itself is trimmed to a 2 cm by 3 cm tick lattice (the package's invariant 1 cm major-grid spacing is preserved):

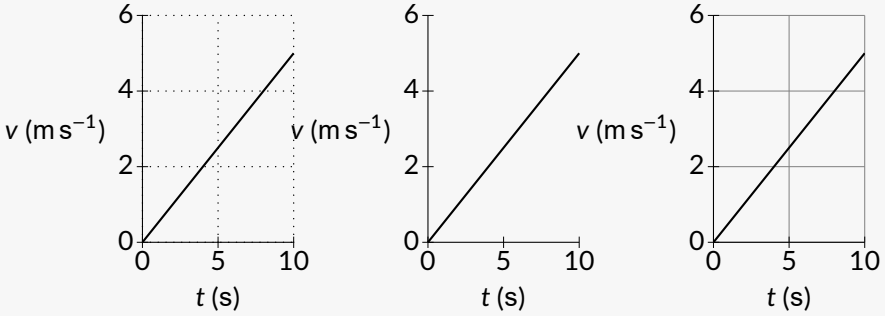
```
\captionsetup{type=figure}%
\begin{subfigure}{0.33\textwidth}\centering
\numodelplotsetup{xcmmax=3, ycmmax=3, axis-label-format=ieee}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{v}\def\xlabelunit{\m\per\s}%
\def\ylabelqty{F}\def\ylabelunit{\N}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.4*x};}
\caption*{\texttt{ieee}}
\end{subfigure}\hspace{-1cm}%
\begin{subfigure}{0.33\textwidth}\centering
\numodelplotsetup{xcmmax=3, ycmmax=3, axis-label-format=iso}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{v}\def\xlabelunit{\m\per\s}%
\def\ylabelqty{F}\def\ylabelunit{\N}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.4*x};}
\caption*{\texttt{iso}}
\end{subfigure}\hspace{-1cm}%
\begin{subfigure}{0.33\textwidth}\centering
\numodelplotsetup{xcmmax=3, ycmmax=3, axis-label-format=brackets}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{v}\def\xlabelunit{\m\per\s}%
\def\ylabelqty{F}\def\ylabelunit{\N}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.4*x};}
\caption*{\texttt{brackets}}
\end{subfigure}
```



ieee iso brackets

Three grid variants, sized the same way:

```
\captionsetup{type=figure}%
\begin{subfigure}{0.33\textwidth}\centering
\numodelplotsetup{xcmmax=3, ycmmax=3, grid=mm-dots}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{t}\def\xlabelunit{\s}%
\def\ylabelqty{v}\def\ylabelunit{\m\per\s}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.5*x};}
\caption*{\texttt{mm-dots}}
\end{subfigure}\hspace{-1cm}%
\begin{subfigure}{0.33\textwidth}\centering
\numodelplotsetup{xcmmax=3, ycmmax=3, grid=none}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{t}\def\xlabelunit{\s}%
\def\ylabelqty{v}\def\ylabelunit{\m\per\s}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.5*x};}
\caption*{\texttt{none}}
\end{subfigure}\hspace{-1cm}%
\begin{subfigure}{0.33\textwidth}\centering
\numodelplotsetup{xcmmax=3, ycmmax=3,
  grid={grid=major, grid style={gray, very thin}}}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{t}\def\xlabelunit{\s}%
\def\ylabelqty{v}\def\ylabelunit{\m\per\s}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.5*x};}
\caption*{\texttt{\{major,gray\}}}
\end{subfigure}
```



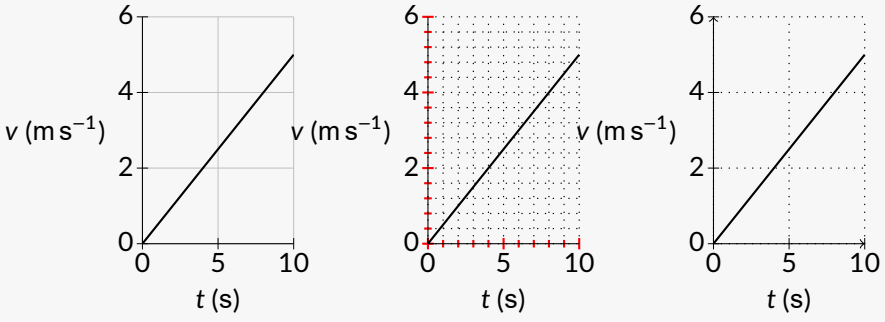
mm-dots none {major,gray}

3.2 PGFPlots styles

The package defines three PGFPlots styles applied by `\drawplot`: `numodel/grid`, `numodel/t`
`icks`, `numodel/axis`. These can be overridden wholesale through `\pgfplotsset{numodel/a`

`xis/.style={...}}` from the calling preamble, giving projects a single choke point for house-style customisation. A fourth style, `numodel/auto`, is not meant to be set by hand: `\calcplotdims` empties it and fills it at every call with what it works out for the plot at hand (an axis through the middle, the label positions, the power of ten of the tick labels), so that nothing carries over from one plot to the next. `\drawplot` applies it after `numodel/axis`; an `axis` environment of your own after a bare `\calcplotdims` should list both. One override per style, on the same plot:

```
\captionsetup{type=figure}%
\begin{subfigure}{0.33\textwidth}\centering
\pgfplotsset{numodel/grid/.style={grid=major,
  grid style={gray!50, very thin}}}%
\numodelplotsetup{xcmmax=3, ycmmax=3}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{t}\def\xlabelunit{\s}%
\def\ylabelqty{v}\def\ylabelunit{\m\per\s}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.5*x};}
\caption*{\texttt{numodel/grid}}
\end{subfigure}\hspace{-1cm}%
\begin{subfigure}{0.33\textwidth}\centering
\pgfplotsset{numodel/ticks/.style={tick style={red, thick},
  minor tick num=4}}%
\numodelplotsetup{xcmmax=3, ycmmax=3}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{t}\def\xlabelunit{\s}%
\def\ylabelqty{v}\def\ylabelunit{\m\per\s}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.5*x};}
\caption*{\texttt{numodel/ticks}}
\end{subfigure}\hspace{-1cm}%
\begin{subfigure}{0.33\textwidth}\centering
\pgfplotsset{numodel/axis/.append style={axis line style={->}}}%
\numodelplotsetup{xcmmax=3, ycmmax=3}%
\def\xmin{0}\def\xmax{10}\def\ymin{0}\def\ymax{5}%
\def\xlabelqty{t}\def\xlabelunit{\s}%
\def\ylabelqty{v}\def\ylabelunit{\m\per\s}%
\drawplot{\addplot[domain=\xmin:\xmax,thick]{0.5*x};}
\caption*{\texttt{numodel/axis}}
\end{subfigure}
```



`numodel/grid` `numodel/ticks` `numodel/axis`

3.3 Tick label halos

When `\calcplotdims` places an axis *through the middle* of the plot (the data straddles zero), grid lines and plotted curves pass behind the scale numbers. To keep those numbers readable, every tick label on such an axis is backed by a semi-transparent halo that follows the character outlines: the label is printed twice, first stroke-only — a fat pen (`1.6pt`, round joins, 80 % opacity, via the pdfrender package) traces the glyph outlines — and then the normal label on top. Between and around the digits the grid and the curves run through untouched; there is no rectangular patch. So that the halo covers the curves and not only the grid, the haloed labels are lifted onto the pgfplots layer `axis de⌋ scriptions`, above the `main` layer carrying the curves (`set layers` is activated automatically in

these cases).

An axis placed on the plot edge (the default bottom/left, or the `top`/`right` placements used for single-sign data) leaves its labels on the white margin and gets no halo. Tick labels also stay visible where a crossing axis would otherwise hide them (`hide obscured x ticks=false` and the *y*-counterpart); the halo provides the visual separation from the crossing axis line.

The halo colour is controlled by the setup key `halo-color`. Its default, `halo-color=auto`, matches the halo to the surrounding background at every `\drawplot`: inside a tcolorbox it takes the box's `colback` (this manual's `plotexample` boxes rely on exactly that), on a beamer slide it takes the colour theme's `bg`, on a page coloured through the pagecolor package it takes `\thepagecolor`, and otherwise it falls back to white. (Plain `\pagecolor` without the pagecolor package leaves no readable record, so it cannot be detected.) A colour value pins the halo instead:

```
\numodelplotsetup{halo-color=<your-color>}  % pin
\numodelplotsetup{halo-color=auto}          % re-enable matching
```

The resolved colour is exposed as the named colour `numodelhalo`. The mechanism itself lives in the styles `numodel/xticklabel halo` and `numodel/yticklabel halo`; redefining them to be empty (`\pgfplotsset{numodel/xticklabel halo/.style={}}` and the *y*-counterpart) removes the halos entirely.

4 Public API

`\drawplot` Renders a `tikzpicture` containing an `axis` whose body is the single mandatory argument. Typically a block of `\addplot` and `\addlegendentry` lines. Calls `\calcplotdims` internally, so the user does not need to invoke it separately.

`\calcplotdims` Reads `\xmin`, `\xmax`, `\ymin`, `\ymax`, and (if set) `\xlabelqty`/`\xlabelunit`/`\ylabelqty`/`\ylabelunit`. Writes `\xcm`, `\ycm`, `\xtickdistance`, `\ytickdistance`, `\xlabel`, `\ylabel`, and may rewrite `\xmin`…`\ymax` to align with the tick lattice (floor/ceil to the nearest tick). It also appends axis-positioning styles to `numodel/axis` based on which quadrants the data occupies. `\drawplot` invokes it automatically; expose for advanced cases where dimensions are needed before rendering (overlay TikZ, custom `axis` environment).

`\xlabelqty` `\xlabelunit` `\ylabelqty` `\ylabelunit` Input hooks for automatic label construction. `\xlabelqty` is the mathematical quantity symbol (e.g. `v`, `F`, `E`); the corresponding `\xlabelunit` is a bare siunitx unit macro sequence (e.g. `\m\per\s`, `\J`, `\N\m`) *without* a surrounding `\si{}` or `\qty{}` wrapper.

`\xcmmax` `\ycmmax` Maximum axis dimensions in centimetres. Can be set directly through `\def` for backwards compatibility, or via `\numodelplotsetup`.

`\qtyPlain` Like siunitx's `\qty` but prints no numeric mantissa when the number has come out as exactly 1 with no exponent left over: the unit then stands alone, without the `quantity-product` symbol that would otherwise separate the two. Used internally to inject scale factors into axis labels; exposed because the same need recurs in other scaled-axis contexts.

All three siunitx `prefix-mode` values are supported, and everything `\qtyPlain` does not special-case — a number carrying an uncertainty, a blank number, `parse-numbers=false` — is passed to `\qty` unchanged. A mantissa of 1 that is preceded by a sign or a comparator (`-1`, `<1`) keeps its digit.

`pzuIfUnitNonEngTF` Boolean conditional testing whether a unit macro sequence carries a power of ten that is not a multiple of three — `\centi`, `\deci`, `\deca`, `\hecto` and their siunitx abbreviations `\cm`, `\dm`, `\hL`, … Used internally to suppress scaling on units where the user has already encoded the order of magnitude; exposed for completeness.

The verdict is the net power of ten siunitx itself extracts from the whole unit expression, so it accounts for unit powers and for prefixes that cancel: `\centi\metre\squared` is 10^{-4} and counts as non-engineering, while `\centi\metre\cubed` (10^{-6}) and `\centi\metre\per\centi\second` (10^{0}) do not.

5 Requirements

numodel-plot requires expl3, xparse, l3keys2e, siunitx (mandatory, for quantities in labels), and pgfplots (with the `fillbetween` library). LuaLaTeX is not required at the plot layer (it is required by the sibling numodel package). TeX Live 2024 or later (or an up-to-date MiKTeX) has all of them in recent enough versions.

siunitx is requested with a date of 2023-11-06 (v3.3.8). Everything the axis labels call has been public since siunitx v3.0.0, but two fixes in v3.3.7 and v3.3.8 — the empty-exponent case of `prefix-mode=combine-exponent` and the printing of a bare 1 under `print-unity-mantissa=false` — are what keeps a scaled label correct. An older siunitx produces a LaTeX warning rather than an error, and the labels may come out subtly wrong.