

The **bodeplot** package

version 3.1

Rushikesh Kamalapurkar
rlkamalapurkar@gmail.com

May 19, 2026

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 2 |
| 1.1 | External Dependencies | 2 |
| 1.2 | Directory Structure | 2 |
| 1.3 | Limitations | 2 |
| 2 | TL;DR | 3 |
| 3 | Usage | 10 |
| 3.1 | Bode plots | 10 |
| 3.1.1 | Basic components up to first order | 17 |
| 3.1.2 | Basic components of the second order | 18 |
| 3.2 | Nyquist plots | 19 |
| 3.3 | Nichols charts | 22 |
| 3.4 | Pole-zero maps | 23 |
| 4 | Implementation | 25 |
| 4.1 | Initialization | 25 |
| 4.2 | PGF keys interface | 27 |
| 4.3 | Parametric function generators for poles, zeros, gains, and delays. | 38 |
| 4.4 | Second order systems. | 39 |
| 4.5 | Commands for Bode plots | 40 |
| 4.5.1 | User macros | 40 |
| 4.5.2 | Internal macros | 49 |
| 4.6 | Nyquist plots | 60 |
| 4.6.1 | User macros | 60 |
| 4.6.2 | Internal macros | 64 |
| 4.7 | Nichols charts | 64 |
| 4.8 | Pole-zero maps | 68 |
| 4.8.1 | User macros | 68 |
| 4.8.2 | Internal macros | 72 |
| A | TL;DR for package version 2.1.1 and earlier | 77 |
| | Index | 89 |
| | Change History | 89 |

1 Introduction

Generate Bode, Nyquist, and Nichols plots for transfer functions in the canonical (TF) form

$$G(s) = e^{-Ts} \frac{b_m s^m + \dots + b_1 s + b_0}{a_n s^n + \dots + a_1 s + a_0} \quad (1)$$

and the zero-pole-gain (ZPK) form

$$G(s) = K e^{-Ts} \frac{(s - z_1)(s - z_2) \dots (s - z_m)}{(s - p_1)(s - p_2) \dots (s - p_n)}. \quad (2)$$

In the equations above, b_m, \dots, b_0 and a_n, \dots, a_0 are real coefficients, $T \geq 0$ is the loop delay, z_1, \dots, z_m and p_1, \dots, p_n are complex zeros and poles of the transfer function, respectively, and $K \in \mathbb{R}$ is the loop gain.

For transfer functions in the ZPK format in (2) *with zero delay*, this package also supports linear and asymptotic approximation of Bode plots.

All phase plots use degrees as units. Use the `rad` package option or the optional argument `phase unit=rad` to generate plots in radians. The `phase unit` key accepts either `rad` or `deg` as inputs.

Frequency inputs and outputs are in radians per second. Use the `Hz` package option or the optional argument `frequency unit=Hz` to generate plots in Hertz. The `frequency unit` key accepts either `rad` or `Hz` as inputs.

1.1 External Dependencies

By default, the package uses `gnuplot` to do all the computations. If `gnuplot` is not available, the `pgf` package option can be used to do the calculations using the native `pgf` math engine. Compilation using the `pgf` math engine is typically slower, but the end result should be the identical (other than phase wrapping in the TF form, see limitations below).

1.2 Directory Structure

Since version 1.0.8, the `bodeplot` package places all `gnuplot` temporary files in the working directory. The package option `declutter` restores the original behavior where the temporary files are placed in a folder called `gnuplot`.

1.3 Limitations

- In `pgf` mode, Bode phase plots and Nichols charts in TF form wrap angles so that they are always between -180 and 180° or $-\pi$ and π radian. As such, these plots will show phase wrapping discontinuities. Since v1.1.1, in `gnuplot` mode, the package uses the `smooth unwrap` filter to correct wrapping discontinuities. As of now, I have not found a way to do this in `pgf` mode, any merge requests or ideas you may have are welcome! Since v1.1.4, you can redefine the `n@mod` macro using the commands `\makeatletter\renewcommand{\n@mod}{\n@mod@p}\makeatother` to wrap the phase between 0 and 360° or 0 and 2π radian. The commands `\makeatletter\renewcommand{\n@mod}{\n@mod@n}\makeatother` will wrap the phase between -360 and 0° or -2π and 0 radian.
- Use of the `declutter` option with other directory management tools such as a `tikzexternalize` prefix is not recommended.

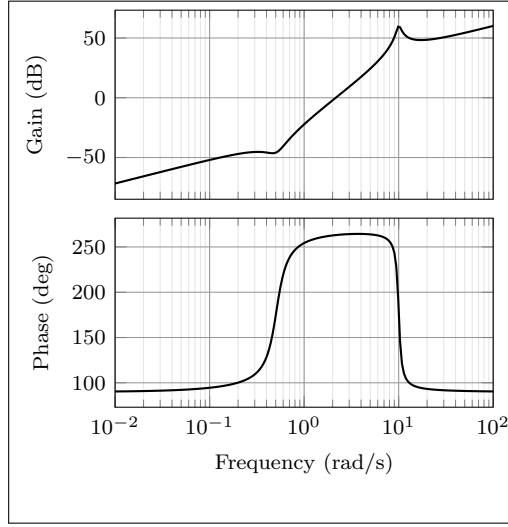
2 TL;DR

All Bode plots in this section are for the transfer function

$$G(s) = 10 \frac{s(s + 0.1 + 0.5i)(s + 0.1 - 0.5i)}{(s + 0.5 + 10i)(s + 0.5 - 10i)} = \frac{s(10s^2 + 2s + 2.6)}{(s^2 + s + 100.25)}. \quad (3)$$

Some of the examples include a transport delay. The examples use PGF-style key-value options and natural syntax for complex numbers in zeros and poles introduced in v3.0. For examples that show the legacy syntax (version 2.1.1 and earlier), please refer to Section A.

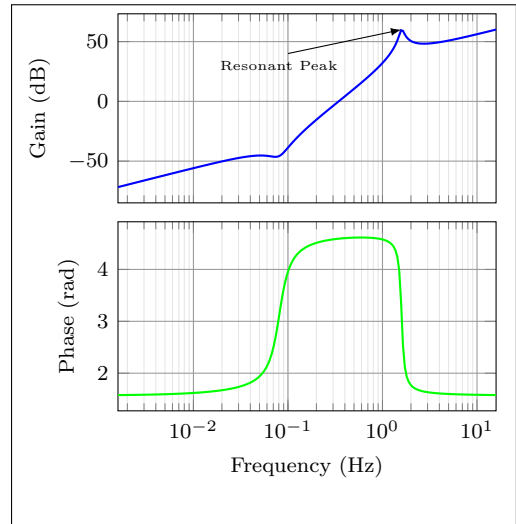
Bode plot in ZPK format



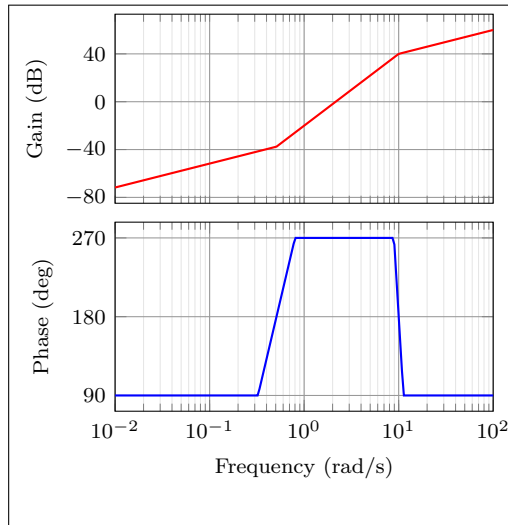
```
\BodeZPK[domain=0.01:100]{%
zeros={0,-0.1-0.5i,-0.1+0.5i},
poles={-0.5-10i,-0.5+10i},
gain=10%
}
```

Same Bode plot over the same frequency range but supplied in Hz, in TF format with arrow decoration, transport delay, unit, and color customization (the phase plot may show wrapping if the **pgf** package option is used)

```
\BodeTF[%
domain=0.01/(2*pi):100/(2*pi),
samples=1000,
mag plot={blue,thick},
ph plot={green,thick},
tikz={%
>=latex,
phase unit=rad,
frequency unit=Hz%
},
mag commands={
\draw[>](axis cs:0.1,40) -- (axis cs:{10/(2*pi)},60);
\node at (axis cs: 0.08,30) {\tiny Resonant Peak};
}%
}
{%
numerator={10,2,2.6,0},
denominator={1,1,100.25}%
}
```



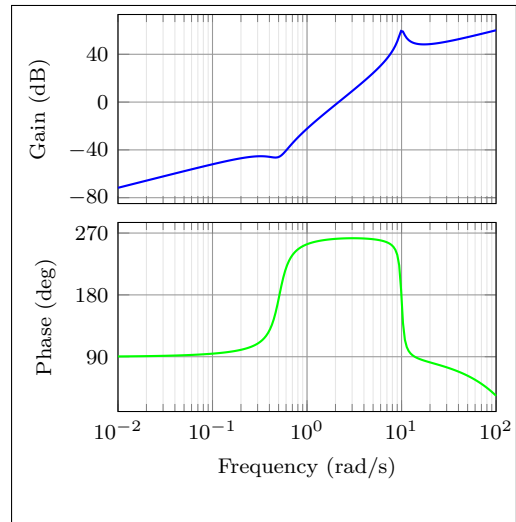
Linear approximation with customization



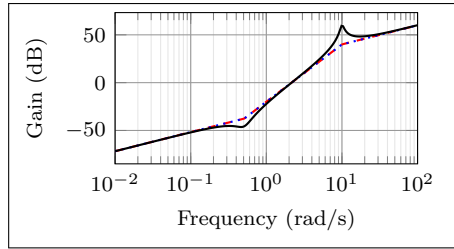
```
\BodeZPK[%
domain=0.01:100,
mag plot={red,thick},
ph plot={blue,thick},
mag axes={ytick distance=40},
ph axes={ytick distance=90},
approx=linear%
]{%
zeros={0,-0.1-0.5i,-0.1+0.5i},
poles={-0.5-10i,-0.5+10i},
gain=10%
}
```

Plot with delay and customization

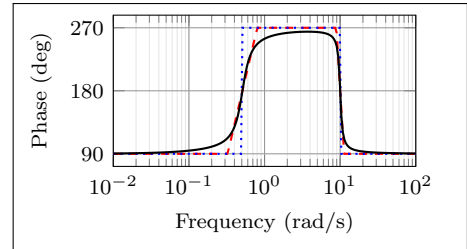
```
\BodeZPK[%
domain=0.01:100,
mag plot={blue,thick},
ph plot={green,thick},
mag axes={ytick distance=40},
ph axes={ytick distance=90}%
]{%
zeros={0,-0.1-0.5i,-0.1+0.5i},
poles={-0.5-10i,-0.5+10i},
gain=10,
delay=0.01%
}
```



Individual gain and phase plots with more customization



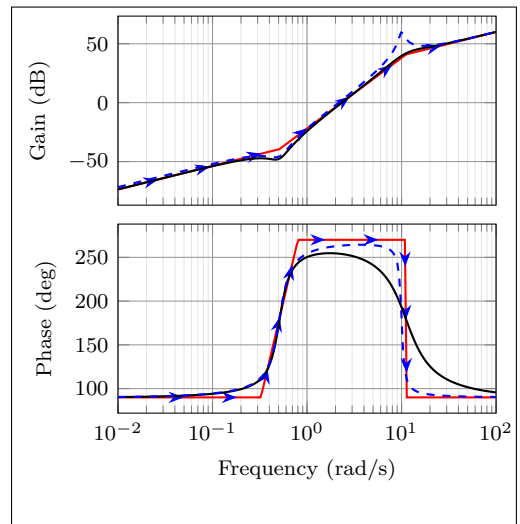
```
\begin{BodeMagPlot}{%
  axes={height=2cm,
    width=4cm},
  domain=0.01:100%
}
\addBodeZPKPlots{%
  true={black,thick},
  linear={red,dashed,thick},
  asymptotic={blue,dotted,thick}%
}
{magnitude}
{%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10%
}
}\end{BodeMagPlot}
```



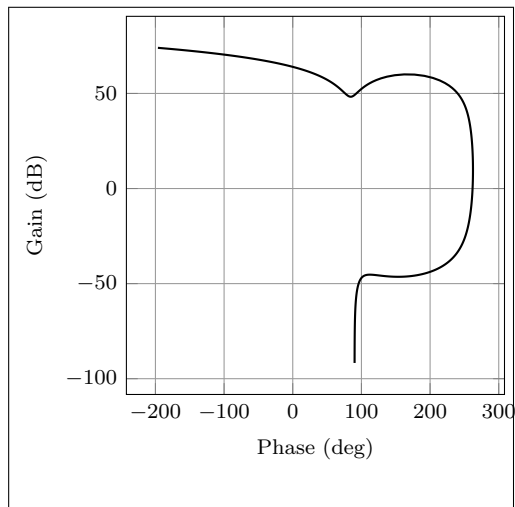
```
\begin{BodePhPlot}{%
  height=2cm,
  width=4cm,
  ytick distance=90,
  domain=0.01:100%
}
\addBodeZPKPlots{%
  true={black,thick},
  linear={red,dashed,thick},
  asymptotic={blue,dotted,thick}%
}
{phase}
{%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10%
}
}\end{BodePhPlot}
```

Multiple transfer functions in a single Bode plot using the **BodePlot** environment and the **\addBodePlot** macro introduced in v2.1.

```
\begin{BodePlot}[domain=0.01:100]
\addBodePlot[red,postaction=decorate,
  decoration={
    markings,
    mark=between positions 0.1 and 0.9 step 2em with {%
      \arrow{Stealth}[length=2mm,blue]}
    },linear]{%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-5-10i,-5+10i},
  gain=10%
}
\addBodePlot[black,thick]{%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-5-10i,-5+10i},
  gain=10%
}%
\addBodePlot[blue,dashed]{%
  numerator={10,2,2,6,0},
  denominator={1,1,100,25}%
}
}\end{BodePlot}
```



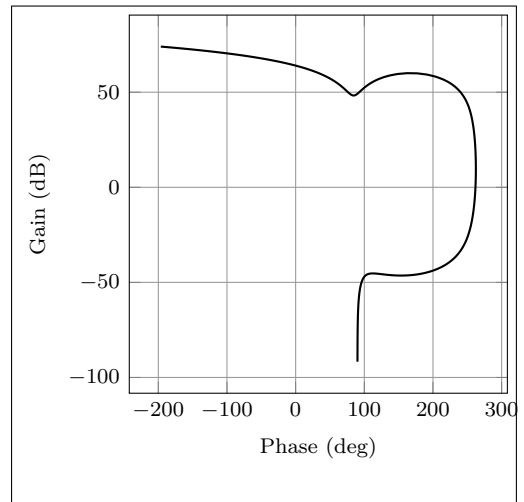
Nichols chart



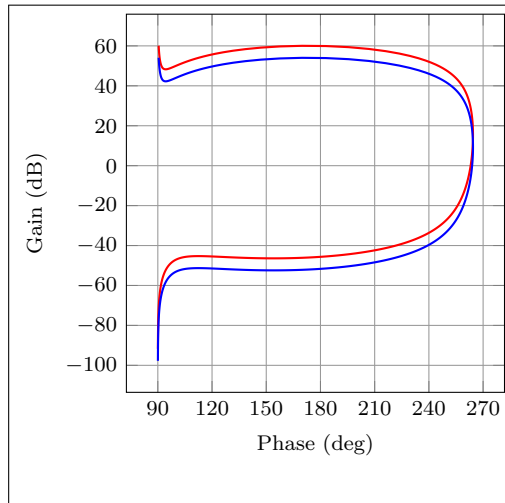
```
\NicholsZPK[domain=0.001:500,samples=1000]
{%
zeros={0,-0.1-0.5i,-0.1+0.5i},
poles={-0.5-10i,-0.5+10i},
gain=10,
delay=0.01%
}
```

Same Nichols chart in TF format (may show wrapping in **pgf** mode)

```
\NicholsTF[domain=0.001:500,samples=1000]
{%
numerator={10,2,2.6,0},
denominator={1,1,100.25},
delay=0.01%
}
```



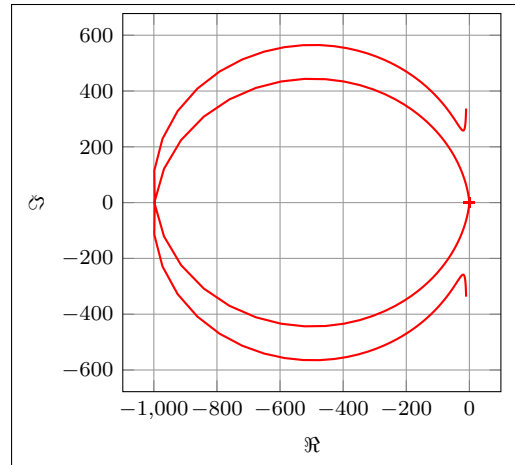
Multiple Nichols charts with customization



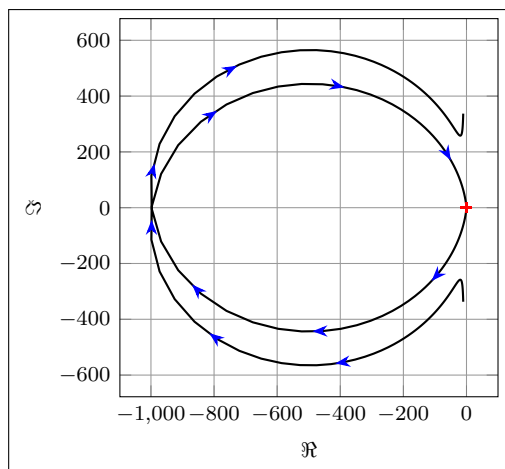
```
\begin{NicholsChart}{%
  ytick distance=20,
  xtick distance=30,
  domain=0.001:100%
}
\addNicholsZPKChart [red,samples=1000] {%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10%
}
\addNicholsZPKChart [blue,samples=1000] {%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=5%
}
\end{NicholsChart}
```

Nyquist plot

```
\NyquistZPK[domain=-30:30,plot={red,thick,samples=1000}]
{%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10%
}
```



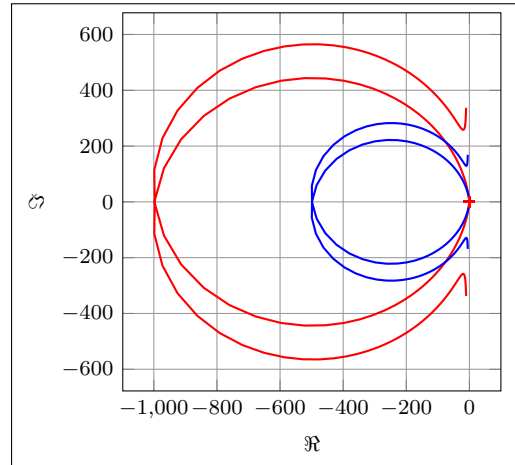
Nyquist plot in TF format with arrows



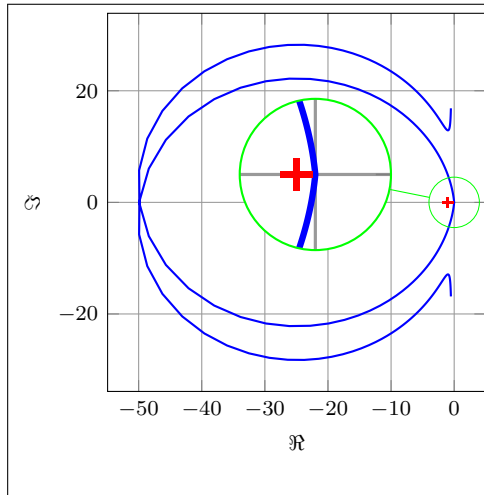
```
\NyquistTF[
  domain=-30:30,
  plot={%
    samples=1000,
    postaction=decorate,
    decoration={
      markings,
      mark=between positions 0.1 and 0.9 step 5em with {%
        \arrow{Stealth [length=2mm, blue]}
      }
    }
  ]%
  numerator={10,2,2.6,0},
  denominator={1,1,100.25}%
}
```

Multiple Nyquist plots with customization

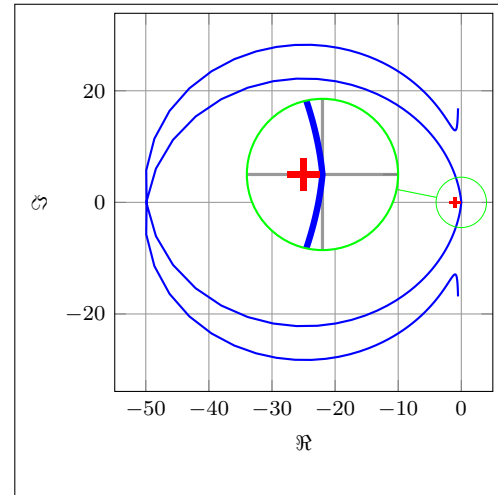
```
\begin{NyquistPlot}[domain=-30:30]
\addNyquistZPKPlot [red,samples=1000] {%
zeros={0,-0.1-0.5i,-0.1+0.5i},
poles={-0.5-10i,-0.5+10i},
gain=10%
}
\addNyquistZPKPlot [blue,samples=1000] {%
zeros={0,-0.1-0.5i,-0.1+0.5i},
poles={-0.5-10i,-0.5+10i},
gain=5%
}
\end{NyquistPlot}
```



Nyquist plots with additional commands, using two different macros

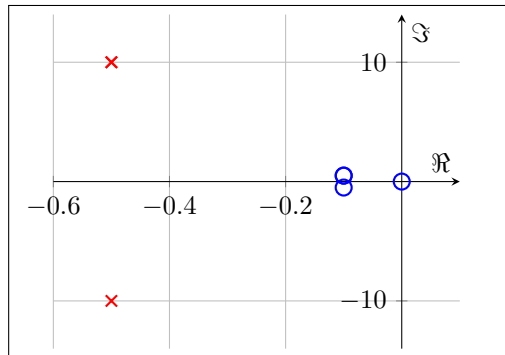


```
\begin{NyquistPlot}[%
tikz={
spy using outlines={%
circle,
magnification=3,
connect spies,
size=2cm
}
},
domain=-30:30%
]
\addNyquistZPKPlot [blue,samples=1000] {%
zeros={0,-0.1-0.5i,-0.1+0.5i},
poles={-0.5-10i,-0.5+10i},
gain=0.5%
}
\coordinate (spyon) at (axis cs:0,0);
\coordinate (spyat) at (axis cs:-22,5);
\spy [green] on (spyon) in
node [fill=white] at (spyat);
\end{NyquistPlot}
```



```
\NyquistZPK[%
plot={blue,samples=1000},
tikz={
spy using outlines={%
circle,
magnification=3,
connect spies,
size=2cm
}
},
commands={
\coordinate (spyon) at (axis cs:0,0);
\coordinate (spyat) at (axis cs:-22,5);
\spy [green] on (spyon) in
node [fill=white] at (spyat);
},
domain=-30:30%
]
{%
zeros={0,-0.1-0.5i,-0.1+0.5i},
poles={-0.5-10i,-0.5+10i},
gain=0.5%
}
```

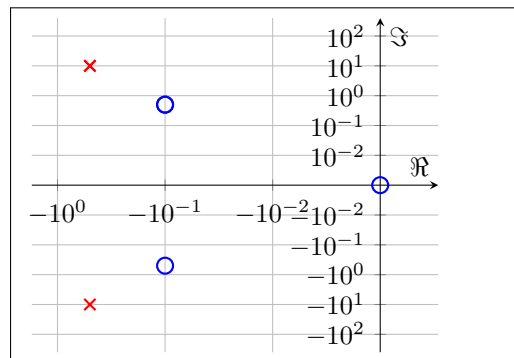

Pole-zero map



```
\PoleZeroMapZPK
{%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10%
}
```

Pole-zero map (symmetric log scale)

```
\PoleZeroMapZPK[scale={log}]
{%
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10%
}
```



3 Usage

In all the macros described here, the frequency limits supplied by the user are assumed to be in **rad/s** unless either the **Hz** package option is used or the optional argument **frequency unit=Hz** is supplied by the user. All phase plots are generated in degrees unless either the **rad** package option is used or the optional argument **phase unit=rad** is supplied by the user.

3.1 Bode plots

\BodeZPK **\BodeZPK**[*options*]{*zpk-spec*}

Plots the Bode plot of a transfer function given in ZPK format using the **groupplot** environment.

zpk-spec: A pgfkeys-style specification using keys **zeros={...}**, **poles={...}**, **gain=...**, and **delay=...** within the **/bodeplot/zpk/** family. Zeros and poles are comma-separated lists with complex poles and zeros entered using the format **a+bi**, where **a** and **b** can be arbitrary pgfmath/gnuplot expressions. The parser starts by detecting the trailing **i** and then finds the first top-level **+** or **-**. Everything between the **i** and the **+** or **-** is treated as the imaginary part, and everything before that is treated as the real part. If no **+** or **-** is found, the entire string except the trailing **i** is treated as the imaginary part and the real part is assumed to be zero. If the **gain** key is not specified, it is assumed to be 1. If the **delay** key is not specified, it is assumed to be zero.

options: The optional argument supports a comma-separated list of options in the form **key=value**. The following keys are supported:

- **domain=min:max:** Sets the frequency range, assumed to be **0.01:100** if not specified.
- **plot={opt}:** Options passed to **\addplot** for both magnitude and phase plots.
- **mag plot={opt}:** Options passed only to the magnitude plot.
- **ph plot={opt}:** Options passed only to the phase plot.
- **axes={opt}:** Options passed to **\nextgroupplot** for both plots.
- **mag axes={opt}:** Options passed only to magnitude axis.
- **ph axes={opt}:** Options passed only to phase axis.
- **group={opt}:** Options passed to the **groupplot** environment.
- **tikz={opt}:** Options passed to the **tikzpicture** environment.
- **commands={opt}:** TikZ commands added after both plots.
- **mag commands={opt}:** TikZ commands added after magnitude plot.
- **ph commands={opt}:** TikZ commands added after phase plot.
- **approx=<true/linear/asymptotic>:** Approximation type for the plot.
- **prefix={name}:** Prefix for gnuplot temporary files.
- **phase unit=<deg/rad>:** Phase axis units.
- **frequency unit=<rad/Hz>:** Frequency axis units.
- Any other key-value pair is passed as plot options to both plots.

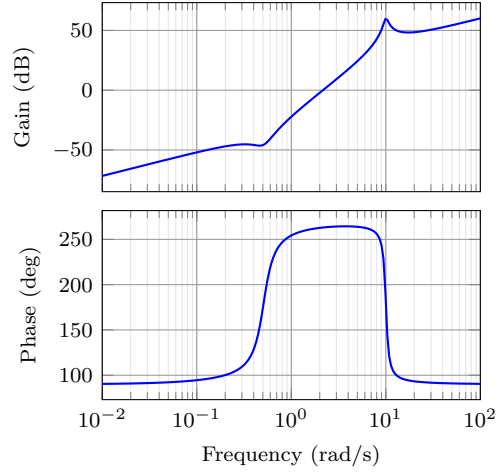


Figure 1: Output of the `\BodeZPK` macro.

For example, given a transfer function

$$G(s) = 10 \frac{s(s + 0.1 + 0.5i)(s + 0.1 - 0.5i)}{(s + 0.5 + 10i)(s + 0.5 - 10i)}, \quad (4)$$

its Bode plot over the frequency range $[0.01, 100]$ can be generated using

```
\BodeZPK [domain=0.01:100, blue, thick]
  {zeros={0, -0.1-0.5i, -0.1+0.5i},
   poles={-0.5-10i, -0.5+10i},
   gain=10}
```

which generates the plot in Figure 1. In this example, a delay is not specified, so it is assumed to be zero. A gain is not specified, so it is assumed to be 1. A single comma-separated list of options `[blue,thick]` is passed, so it is passed on to the `\addplot` commands in both the magnitude and the phase plots. The default plots are thick black lines and each of the axes, excluding ticks and labels, are 5cm wide and 2.5cm high.

The width and the height, along with other properties of the plots, the axes, and the group can be customized using native `pgf` keys. For example, a linear approximation of the Bode plot with customization of the plots, the axes, and the group can be generated using

```
\BodeZPK[%
  mag plot={red,thick},
  ph plot={blue,thick},
  mag axes={ytick distance=40,xmajorticks=true,xlabel={Frequency (rad/s)}},
  ph axes={ytick distance=90},
  group={group style={group size=2 by 1,horizontal sep=2cm,width=4cm,height=2cm}},
  approx=linear]
  {zeros={0, -0.1-0.5i, -0.1+0.5i},poles={-0.5-10i, -0.5+10i},gain=10}
```

which generates the plot in Figure 2.

Legacy interface (v2.1.1 and earlier):

```
\BodeZPK [options]{zpk-spec}{min-freq}{max-freq}
```

If both frequency arguments are provided, the macro falls back to the legacy format. This format is supported for backward compatibility with versions 2.1.1 and earlier:

zpk-spec (v2.1.1 and earlier): `z/{zeros},p/{poles},k/{gain},d/{delay}`

where zeros and poles are comma-separated lists of complex numbers of the form `{{real part 1,imaginary part 1},{real part 2,imaginary part 2},...}`. If the imaginary part is not provided, it is assumed to be zero. This format requires the frequency arguments `min-freq` and `max-freq` to be provided.

options (v2.1.1 and earlier): A comma-separated list of options that modify the plots. The following options are supported:

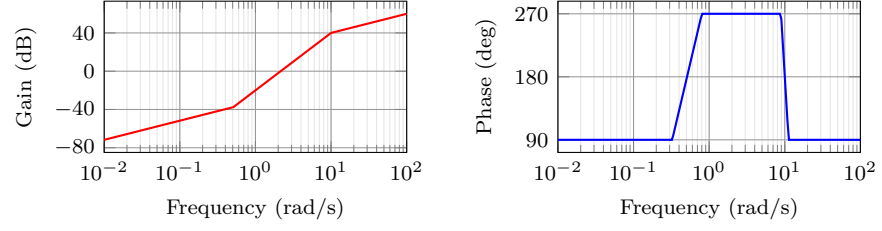


Figure 2: Customization of the default `\BodeZPK` macro.

- `plot/typ/{opt}`: modify plot properties by adding options `{opt}` to the `\addplot` macro for the magnitude plot if `typ` is `mag` and the phase plot if `typ` is `ph`.
- `axes/typ/{opt}`: modify axis properties by adding options `{opt}` to the `\nextgroupplot` macro for the magnitude plot if `typ` is `mag` and the phase plot if `typ` is `ph`.
- `commands/typ/{opt}`: add any valid TikZ commands to the magnitude plot if `typ` is `mag` and the phase plot if `typ` is `ph`.
- `plot/{opt}`: adds options `{opt}` to `\addplot` macros for both the magnitude and the phase plots.
- `axes/{opt}`: adds options `{opt}` to `\nextgroupplot` macros for both the magnitude and the phase plots.
- `group/{opt}`: adds options `{opt}` to the `groupplot` environment.
- `tikz/{opt}`: adds options `{opt}` to the `tikzpicture` environment.
- `approx/linear`: plots linear approximation.
- `approx/asymptotic`: plots asymptotic approximation.
- `prefix/{opt}`: adds the string `opt` as a prefix to all temporary files generated by `gnuplot` for this plot.
- `{opt}` (bare options): add all of the supplied options to `\addplot` macros for both the magnitude and the phase plots.

Example of the legacy interface:

```
\BodeZPK [blue,thick]
  {z/{0},{-0.1,-0.5},{-0.1,0.5}},p/{{-0.5,-10},{-0.5,10}},k/10}
  {0.01}{100}
```

`\BodeTF \BodeTF[<options>]{<tf-spec>}`

Plots the Bode plot of a transfer function given in TF format.

tf-spec: The mandatory argument contains the pgf keys `numerator={coeffs}`, `denominator={coeffs}`, and optionally `delay=value`. The coefficients are entered as a comma-separated list, in order from the highest degree of s to the lowest, with zeros for missing degrees.

options: The frequency range is specified in the first optional argument using `domain=min:max`. The optional argument can also contain any of the keys supported by the `\BodeZPK` macro except for the `approx` key since linear/asymptotic approximation is not supported for TF format.

For example, given the same transfer function as (4) in TF form and with a small transport delay,

$$G(s) = e^{-0.01s} \frac{s(10s^2 + 2s + 2.6)}{(s^2 + s + 100.25)}, \quad (5)$$

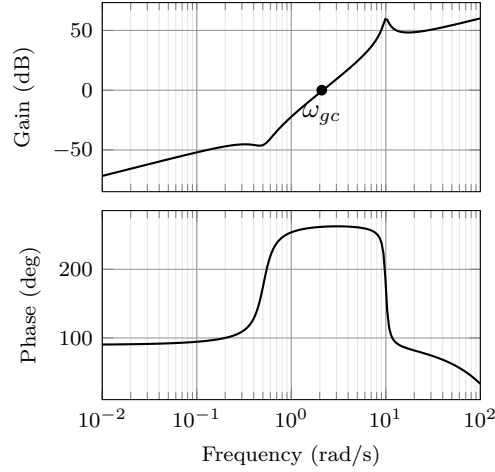


Figure 3: Output of the `\BodeTF` macro with an optional TikZ command used to mark the gain crossover frequency.

its Bode plot over the frequency range $[0.01, 100]$ can be generated using

```
\BodeTF[%
  domain=0.01:100,
  mag commands={\node at (axis cs: 2.1,0) [circle,fill,inner sep=0.05cm,
    label=below:{$\omega_{gc}$}]}{};
  {numerator={10,2,2.6,0}, denominator={1,1,100.25}, delay=0.01}
```

which generates the plot in Figure 3. Note the 0 added to the numerator coefficients to account for the fact that the numerator does not have a constant term in it. Note the semicolon after the TikZ command passed to the `\commands` option.

Legacy interface (v2.1.1 and earlier):

```
\BodeTF [options]{tf-spec}{min-freq}{max-freq}
```

If both frequency arguments are provided, the macro falls back to the legacy format. The three mandatory arguments include: (1) a list of tuples comprised of the coefficients in the numerator and the denominator of the transfer function and the transport delay (format: `num/{coeffs},den/{coeffs},d/{delay}`), (2) the lower end of the frequency range, and (3) the upper end of the frequency range. The coefficients are entered as a comma-separated list, in order from the highest degree of s to the lowest, with zeros for missing degrees. The optional arguments are the same as `\bodeZPK` except that linear/asymptotic approximation is not supported, so `approx/...` is ignored. Example of the legacy interface:

```
\BodeTF[%
  commands/mag/{\node at (axis cs: 2.1,0) [circle,fill,inner sep=0.05cm,
    label=below:{$\omega_{gc}$}]}{};
  {num/{10,2,2.6,0},den/{1,1,100.25},d/0.01}
  {0.01}{100}
```

```
BodeMagPlot (env.) \begin{BodeMagPlot}[options]
```

```
\addBode...
```

```
\end{BodeMagPlot}
```

The `BodeMagPlot` environment works in conjunction with the parametric function generator macros `\addBodeZPKPlots`, `\addBodeTFPlot`, and `\addBodeComponentPlot`, intended to be used for magnitude plots. The first optional argument can contain keys from the `/bodeplot/env/` family, including `domain=min:max` to set the frequency range, `axes={...}` for axis options, `tikz={...}` for tikzpicture options, and `prefix={...}` for gnuplot file prefixes. The `prefix` option adds the provided string as a prefix to all temporary files generated by `gnuplot` for all plots in this environment. This option is useful if data generated by `gnuplot` for a specific plot needs to be uniquely

identified. All other key-value pairs are passed as options to the `semilogaxis` environment. For examples, see the description of `\addBodeZPKPlots`, `\addBodeTFPlot`, and `\addBodeComponentPlot`.

Legacy interface (v2.1.1 and earlier):

```
\begin{BodeMagPlot}[\langle options \rangle]{\langle min-freq \rangle}{\langle max-freq \rangle}
  \addBode...
\end{BodeMagPlot}
```

If both frequency arguments are provided, the environment uses the legacy format. The first optional argument is comprised of a comma separated list of tuples, either `obj/{opt}` or just `{opt}`. Each tuple passes options to different `pgfplots` macros that generate the axes and the plots according to:

- Tuples of the form `obj/{opt}`:
 - `tikz/{opt}`: modify picture properties by adding options `{opt}` to the `tikzpicture` environment.
 - `axes/{opt}`: modify axis properties by adding options `{opt}` to the `semilogaxis` environment.
 - `prefix/{opt}`: adds the string `opt` as a prefix to all temporary files generated by `gnuplot` for all plots in this environment. This option is useful if data generated by `gnuplot` for a specific plot needs to be uniquely identified.
- Tuples of the form `{opt}` are passed directly to the `semilogaxis` environment.

The frequency limits are translated to the x-axis limits and the domain of the `semilogaxis` environment.

```
BodePhPlot (env.) \begin{BodePhPlot}[\langle options \rangle]
  \addBode...
\end{BodePhPlot}
```

Same as `BodeMagPlot`, but intended to be used for phase plots.

```
\addBodeZPKPlots \addBodeZPKPlots[\langle options \rangle]{\langle plot-type \rangle}{\langle zpk-spec \rangle}
```

Generates the appropriate parametric functions and supplies them to multiple `\addplot` macros, one for each approximation type specified in the optional argument. If no optional argument is supplied, then a single `\addplot` command corresponding to a thick true Bode plot is generated. The `options` argument accepts keys `true={\langle plot-opts \rangle}`, `linear={\langle plot-opts \rangle}`, and `asymptotic={\langle plot-opts \rangle}`, where `\langle plot-opts \rangle` are TikZ/pgfplots styling options. Multiple approximations can be specified as `true={black,thick},linear={red,dashed},etc.` This macro can be used inside any `semilogaxis` environment as long as a domain for the x-axis is supplied in the environment or parent axis options. Use with the `BodeMagPlot` or `BodePhPlot` environments supplied with this package is recommended. The second mandatory argument, `plot-type` is either `magnitude` or `phase`. If it is not equal to `phase`, it is assumed to be `magnitude`. The third mandatory argument, `zpk-spec` is the same as that of the `\BodeZPK` macro described earlier.

For example, given the transfer function in (4), its linear, asymptotic, and true Bode plots can be superimposed using:

```
\begin{BodeMagPlot}[height=2cm,width=4cm,domain=0.01:100,xlabel={}]
  \addBodeZPKPlots[true={black,thick},
    linear={red,dashed,thick},
    asymptotic={blue,dotted,thick}]
  {magnitude}
  {zeros={0,-0.1-0.5i,-0.1+0.5i},poles={-0.5-10i,-0.5+10i},gain=10}
\end{BodeMagPlot}
\begin{BodePhPlot}[height=2cm,width=4cm,ytick distance=90,domain=0.01:100]
  \addBodeZPKPlots[true={black,thick},
    linear={red,dashed,thick},
    asymptotic={blue,dotted,thick}]
  {phase}
  {zeros={0,-0.1-0.5i,-0.1+0.5i},poles={-0.5-10i,-0.5+10i},gain=10}
\end{BodePhPlot}
```

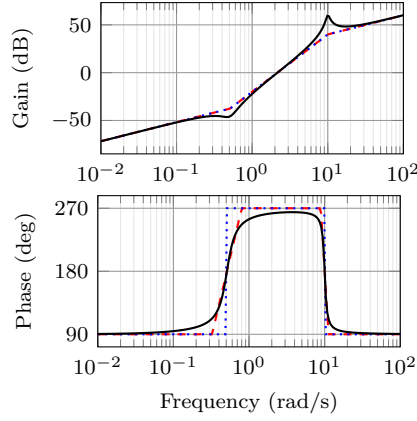


Figure 4: Superimposed approximate and true Bode plots using the **BodeMagPlot** and **BodePhPlot** environments and the `\addBodeZPKPlots` macro.

which generates the plot in Figure 4.

For backward compatibility, the legacy format with explicit frequency arguments is also supported.

`\addBodeZPKPlots[<options>]{<plot-type>}{<zpk-spec>}`

If an equals sign is not detected in the *zpk-spec*, then the macro falls back to the legacy format. In the legacy format, the macro generates the appropriate parametric functions and supplies them to multiple `\addplot` macros, one for each **approx/{opt}** pair in the optional argument. If no optional argument is supplied, then a single `\addplot` command corresponding to a thick true Bode plot is generated. If an optional argument is supplied, it needs to be one of **true/{opt}**, **linear/{opt}**, or **asymptotic/{opt}**. For example, the plots in Figure 4 can also be generated using:

```
\begin{BodeMagPlot}[height=2cm,width=4cm] {0.01} {100}
  \addBodeZPKPlots[%
    true/{black,thick},
    linear/{red,dashed,thick},
    asymptotic/{blue,dotted,thick}]
    {magnitude}
    {z/{0,{-0.1,-0.5},{-0.1,0.5}},p/{{-0.5,-10},{-0.5,10}},k/10}
\end{BodeMagPlot}

\begin{BodePhPlot}[height=2cm, width=4cm, ytick distance=90] {0.01} {100}
  \addBodeZPKPlots[%
    true/{black,thick},
    linear/{red,dashed,thick},
    asymptotic/{blue,dotted,thick}]
    {phase}
    {z/{0,{-0.1,-0.5},{-0.1,0.5}},p/{{-0.5,-10},{-0.5,10}},k/10}
\end{BodePhPlot}
```

`\addBodeTFPlot \addBodeTFPlot[<options>]{<plot-type>}{<tf-spec>}`

Generates a single parametric function for either Bode magnitude or phase plot of a transfer function in TF form. The generated parametric function is passed to the `\addplot` macro. This macro can be used inside any **semilogaxis** environment as long as a domain for the x-axis is supplied through the **options** argument or in the optional argument of the container **semilogaxis** environment. Use with the **BodeMagPlot** and **BodePhPlot** environments supplied with this package is recommended. The **options** can include any options that are accepted by the `\addplot` macro. The mandatory argument **plot-type** is either **magnitude** or **phase**. If it is not equal to **phase**, it is assumed to be **magnitude**. The last mandatory argument supports both legacy format `num/{coeffs},den/{coeffs},d/{delay}` and new format (v3.0) using pgfkeys like

numerator={...},denominator={...},delay=.... The frequency range is specified using domain=min:max in the options argument.

`\addBodeComponentPlot` `\addBodeComponentPlot[\langle plot-options \rangle]{ \langle plot-command \rangle }`

Generates a single parametric function corresponding to the mandatory argument `plot-command` and passes it to the `\addplot` macro. The plot command can be any parametric function that uses `t` as the independent variable. The parametric function must be `gnuplot` compatible (or `pgfplots` compatible if the package is loaded using the `pgf` option, **with angles passed to trigonometric functions in radian**). The intended use of this macro is to plot the parametric functions generated using the basic component macros described in Section 3.1.1 below.

`\addBodePlot` `\addBodePlot[\langle plot-options \rangle]{ \langle system-data \rangle }`

Unified macro to add Bode plots for both ZPK and TF system representations. Unlike the `\addBodeZPKPlots` and `\addBodeTFPlots` macros, this macro can only be used inside a `BodePlot` environment. **It will not function if used in a generic semilogxaxis environment or in the BodeMagPlot and BodePhPlot environments.** For ZPK systems, the `system-data` has the same format `zeros={...},poles={...},gain=..., delay=...` as `\addBodeZPKPlots`. For TF systems, it has the same format `numerator={...},denominator={...},gain=...,delay=...` as `\addBodeTFPlot`. The optional arguments are the same as the `\addBodeTfPlot` macro, with two additions. Passing `linear` or `asymptotic` options will generate linear or asymptotic approximations if the system representation is ZPK. These two options are ignored for TF systems. The macro automatically detects the system representation based on the presence of `zeros` or `poles` keys (ZPK) versus `numerator` or `denominator` keys (TF) in the `system-data` argument.

Legacy format (v2.1.1 and earlier):

`\addBodePlot[\langle plot-options \rangle]{ \langle system-type \rangle }{ \langle system-data \rangle }`

If invoked using two mandatory arguments, the command falls back to the legacy format. In the legacy format, the command does not automatically detect the system representation. The second mandatory argument `system-type` should be either `zpk` or `tf` to specify the system representation.

`BodePlot (env.)` `\begin{BodePlot}[\langle options \rangle]`

`\addBodePlot...`

`\end{BodePlot}`

Starting from version 2.1, the deprecated `BodePlot` environment is revived exclusively to host multiple instances of the unified `\addBodePlot` macro. The `\addBodeZPKPlots` and `\addBodeTFPlot` macros **will not function inside this environment**. The purpose of this environment is to plot a Bode plot that contains both phase and magnitude plots for several different transfer functions.

The environment uses the `pgfkeys` interface from the `/bodeplot/combinedenv/` family, supporting keys like `domain=min:max`, `group={...}`, `approx=...`, `mag axes={...}`, `ph axes={...}`, `mag commands={...}`, `ph commands={...}`, `tikz={...}`, and `prefix={...}` similar to the `\BodeZPK` macro.

For example, the Bode plot of two different transfer functions, one in ZPK form and the other in TF form, can be generated using:

```
\begin{BodePlot}[domain=0.01:100,group={group style={horizontal sep=2cm}}]
  \addBodePlot[true={black,thick},linear={red,dashed,thick},asymptotic={blue,dotted,thick}]
    {zeros={0,-0.1-0.5i,-0.1+0.5i},poles={-0.5-10i,-0.5+10i},gain=10}
  \addBodePlot[blue,thick]
    {numerator={10,2,2.6,0}, denominator={1,1,100.25}, delay=0.01}
\end{BodePlot}
```

which generates the plot in Figure 5.

Legacy interface (v2.1.1 and earlier):

`\begin{BodePlot}[\langle options \rangle]{ \langle min-freq \rangle }{ \langle max-freq \rangle }`

`\addBodePlot...`

`\end{BodePlot}`

If both frequency arguments are provided, the environment falls back to the legacy interface, where the optional arguments are the same as the legacy interface for the

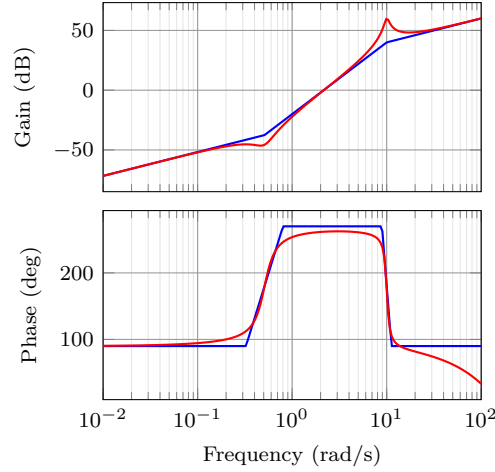


Figure 5: Bode plot combining ZPK and TF systems using the `BodePlot` environment and the unified `\addBodePlot` macro.

`\BodeZPK` macro, supporting `plot/`, `axes/`, `group/`, `tikz/` options with `mag` and `ph` sub-types.

3.1.1 Basic components up to first order

`\TypeFeatureApprox \TypeFeatureApprox{<real-part>}{<imaginary-part>}`

This entry describes 20 different macros of the form `\TypeFeatureApprox` that take the real part and the imaginary part of a complex number as arguments. The **Type** in the macro name should be replaced by either **Mag** or **Ph** to generate a parametric function corresponding to the magnitude or the phase plot, respectively. The **Feature** in the macro name should be replaced by one of **K**, **Pole**, **Zero**, or **Del**, to generate the Bode plot of a gain, a complex pole, a complex zero, or a transport delay, respectively. If the **Feature** is set to either **K** or **Del**, the **imaginary-part** mandatory argument is ignored. The **Approx** in the macro name should either be removed, or it should be replaced by **Lin** or **Asymp** to generate the true Bode plot, the linear approximation, or the asymptotic approximation, respectively. If the **Feature** is set to **Del**, then **Approx** has to be removed. For example,

- `\MagK{k}{0}` or `\MagK{k}{400}` generates a parametric function for the true Bode magnitude of $G(s) = k$
- `\PhPoleLin{a}{b}` generates a parametric function for the linear approximation of the Bode phase of $G(s) = \frac{1}{s-a-ib}$.
- `\PhDel{T}{200}` or `\PhDel{T}{0}` generates a parametric function for the Bode phase of $G(s) = e^{-Ts}$.

All 20 of the macros defined by combinations of **Type**, **Feature**, and **Approx**, and any `gnuplot` (or `pgfplot` if the `pgf` class option is loaded) compatible function of the 20 macros can be used as `plot-command` in the `addBodeComponentPlot` macro. This is sufficient to generate the Bode plot of any rational transfer function with delay. For example, the Bode phase plot in Figure 4 can also be generated using:

```
\begin{BodePhPlot}[ytick distance=90,domain=0.01:100]
  \addBodeComponentPlot[black,thick]{%
    \PhZero{0}{0} + \PhZero{-0.1}{-0.5} + \PhZero{-0.1}{0.5} +
    \PhPole{-0.5}{-10} + \PhPole{-0.5}{10} + \PhK{10}{0}}
  \addBodeComponentPlot[red,dashed,thick]{%
    \PhZeroLin{0}{0} + \PhZeroLin{-0.1}{-0.5} + \PhZeroLin{-0.1}{0.5} +
    \PhPoleLin{-0.5}{-10} + \PhPoleLin{-0.5}{10} + \PhKLin{10}{20}}
  \addBodeComponentPlot[blue,dotted,thick]{%

```

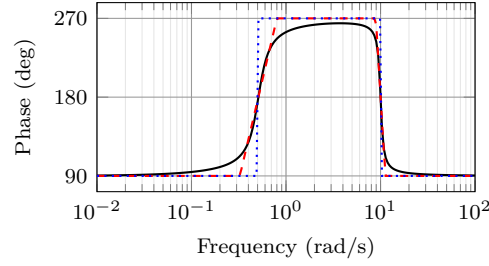


Figure 6: Superimposed approximate and true Bode Phase plot using the `BodePhPlot` environment, the `\addBodeComponentPlot` macro, and several macros of the `\TypeFeatureApprox` form.

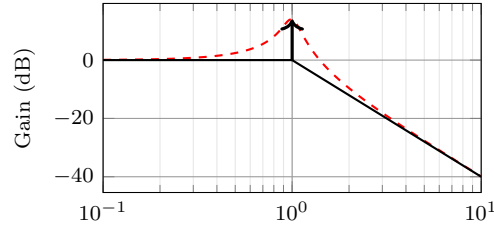


Figure 7: Resonant peak in asymptotic Bode plot using `\MagSOPolesPeak`.

```
\PhZeroAsymp{0}{0} + \PhZeroAsymp{-0.1}{-0.5} + \PhZeroAsymp{-0.1}{0.5} +
\PhPoleAsymp{-0.5}{-10} + \PhPoleAsymp{-0.5}{10} + \PhKAsymp{10}{40}
\end{BodePhPlot}
```

which gives us the plot in Figure 6.

3.1.2 Basic components of the second order

`\TypeS0FeatureApprox` `\TypeS0FeatureApprox{<a1>}{<a0>}`

This entry describes 12 different macros of the form `\TypeS0FeatureApprox` that take the coefficients a_1 and a_0 of a general second order system as inputs. The **Feature** in the macro name should be replaced by either **Poles** or **Zeros** to generate the Bode plot of $G(s) = \frac{1}{s^2 + a_1s + a_0}$ or $G(s) = s^2 + a_1s + a_0$, respectively. The **Type** in the macro name should be replaced by either **Mag** or **Ph** to generate a parametric function corresponding to the magnitude or the phase plot, respectively. The **Approx** in the macro name should either be removed, or it should be replaced by **Lin** or **Asymp** to generate the true Bode plot, the linear approximation, or the asymptotic approximation, respectively.

`\MagS0FeaturePeak` `\MagS0FeaturePeak[<draw-options>]{<a1>}{<a0>}`

This entry describes 2 different macros of the form `\MagS0FeaturePeak` that take the coefficients a_1 and a_0 of a general second order system as inputs, and draw a resonant peak using the `\draw` TikZ macro. The **Feature** in the macro name should be replaced by either **Poles** or **Zeros** to generate a peak for poles and a valley for zeros, respectively. For example, the command

```
\begin{BodeMagPlot}[xlabel={},domain=0.1:10]
\addBodeComponentPlot[red,dashed,thick]{\MagSOPoles{0.2}{1}}
\addBodeComponentPlot[black,thick]{\MagSOPolesLin{0.2}{1}}
\MagSOPolesPeak[thick]{0.2}{1}
\end{BodeMagPlot}
```

generates the plot in Figure 7.

`\TypeCSFeatureApprox` `\TypeCSFeatureApprox{<zeta>}{<omega-n>}`

This entry describes 12 different macros of the form `\TypeCSFeatureApprox` that take the damping ratio, ζ , and the natural frequency, ω_n of a canonical second order system as inputs. The **Type** in the macro name should be replaced by either **Mag** or **Ph** to generate a

parametric function corresponding to the magnitude or the phase plot, respectively. The **Feature** in the macro name should be replaced by either **Poles** or **Zeros** to generate the Bode plot of $G(s) = \frac{1}{s^2 + 2\zeta\omega_n s + \omega_n^2}$ or $G(s) = s^2 + 2\zeta\omega_n s + \omega_n^2$, respectively. The **Approx** in the macro name should either be removed, or it should be replaced by **Lin** or **Asymp** to generate the true Bode plot, the linear approximation, or the asymptotic approximation, respectively.

`\MagCSFeaturePeak` `\MagCSFeaturePeak[$\langle draw-options \rangle$]{ $\langle zeta \rangle$ }{ $\langle omega-n \rangle$ }`

This entry describes 2 different macros of the form `\MagCSFeaturePeak` that take the damping ratio, ζ , and the natural frequency, ω_n of a canonical second order system as inputs, and draw a resonant peak using the `\draw TikZ` macro. The **Feature** in the macro name should be replaced by either **Poles** or **Zeros** to generate a peak for poles and a valley for zeros, respectively.

`\MagCCFeaturePeak` `\MagCCFeaturePeak[$\langle draw-options \rangle$]{ $\langle real-part \rangle$ }{ $\langle imaginary-part \rangle$ }`

This entry describes 2 different macros of the form `\MagCCFeaturePeak` that take the real and imaginary parts of a pair of complex conjugate poles or zeros as inputs, and draw a resonant peak using the `\draw TikZ` macro. The **Feature** in the macro name should be replaced by either **Poles** or **Zeros** to generate a peak for poles and a valley for zeros, respectively.

3.2 Nyquist plots

`\NyquistZPK` `\NyquistZPK [$\langle options \rangle$]{ $\langle zpk-spec \rangle$ }`

Generates the Nyquist diagram of a transfer function given in ZPK form and marks the critical point $(-1, 0)$ with a thick red $+$.

zpk-spec: Same as the `\BodeZPK` macro. Provide zeros, poles, gain, and optional delay with the pgfkeys-style format `zeros={...},poles={...},gain=...,delay=...`. Complex numbers are written as `a+bi`, where `a` and `b` can be arbitrary pgfmath/gnuplot expressions. The macro assumes unit gain and zero delay when those keys are omitted.

options:

- **domain=min:max:** Sets the frequency sweep for the contour. The default sweep is `-30:30` when the key is not provided.
- **plot={opt}:** Appends `opt` to the `\addplot` options that draw the Nyquist curve, e.g., `samples=2000`.
- **axes={opt}:** Adds `opt` to the surrounding `axis` environment.
- **commands={opt}:** Inserts `opt` immediately after the contour inside the `axis`, which is useful for annotations or additional plotting commands.
- **tikz={opt}:** Adds `opt` to the enclosing `tikzpicture`.
- **prefix={name}:** Overrides the prefix used for auxiliary gnuplot tables.

Any additional **key=value** entries are forwarded to `\addplot`, so native `pgfplots` options may be supplied without wrapping them in `plot={...}`. Linear or asymptotic approximations are not available for Nyquist plots. To indicate the direction of increasing frequency, load `decorations.markings` and `arrows.meta` and add `plot={postaction=decorate,decoration={...}}`. Very large sample counts in combination with decorations can trigger the warning **! Dimension too big**.

For example, the command

```
\NyquistZPK[
  domain=-30:30,
  plot={red,thick,samples=2000},
  axes={blue,thick}]
{zeros={0,-0.1-0.5i,-0.1+0.5i},
 poles={-0.5-10i,-0.5+10i},
 gain=10}
```

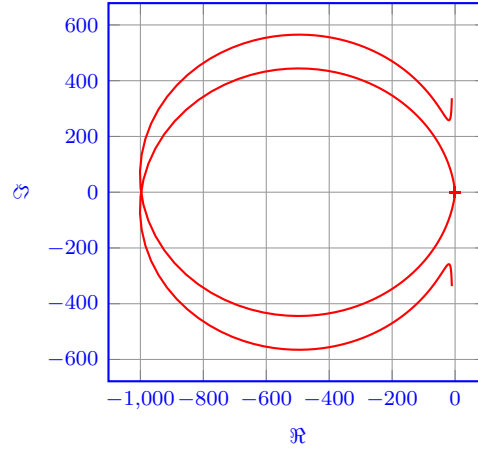


Figure 8: Output of the `\NyquistZPK` macro.

generates the Nyquist plot in Figure 8.

Legacy interface (v2.1.1 and earlier):

`\NyquistZPK[<options>]{<zpk-spec>}{<min-freq>}{<max-freq>}`

The optional argument accepts comma-separated tuples `plot/{opt}`, `axes/{opt}`, `commands/{opt}`, `tikz/{opt}`, and `prefix/{opt}`, while a bare `{opt}` is treated as `plot/{opt}`. The `zpk-spec` uses the legacy format `z/{...},p/{...},k/...,d/...`, where complex numbers are expressed as `{a,b}`. Supplying the frequency arguments auto-selects the legacy behaviour and is intended only for maintaining older documents.

Example of legacy usage:

```
\NyquistZPK[plot/{red,thick,samples=2000},axes/{blue,thick}]
  {z/{0,{-0.1,-0.5},{-0.1,0.5}},p/{{-0.5,-10},{-0.5,10}},k/10}
  {-30}{30}
```

`\NyquistTF \NyquistTF[<options>]{<tf-spec>}`

Generates the Nyquist diagram of a transfer function expressed in polynomial form. The `tf-spec` is the same as that of the `\BodeTF` macro described earlier and the `options` argument uses the same pgfkeys interface as `\NyquistZPK`. A legacy interface similar to the one in `\NyquistZPK` is also supported when both frequency arguments are provided. For example,

```
\NyquistTF[
  domain=-30:30,
  plot={green,thick,samples=500,postaction=decorate,
    decoration={markings,
      mark=between positions 0.1 and 0.9 step 5em
      with{\arrow{Stealth[length=2mm, blue]}}}}
  {numerator={10,2,2.6,0}, denominator={1,1,100.25}}
```

yields the Nyquist plot in Figure 9. As with the ZPK variant, combining dense sampling with `decorations.markings` may trigger the warning `! Dimension too big`.

Legacy syntax remains available:

```
\NyquistTF[plot/{green,thick,samples=500,postaction=decorate,
  decoration={markings,
    mark=between positions 0.1 and 0.9 step 5em
    with{\arrow{Stealth[length=2mm, blue]}}}}
  {num/{10,2,2.6,0},den/{1,1,100.25}}
  {-30}{30}
```

```
NyquistPlot (env.) \begin{NyquistPlot}[<options>]
  \addNyquist...
\end{NyquistPlot}
```

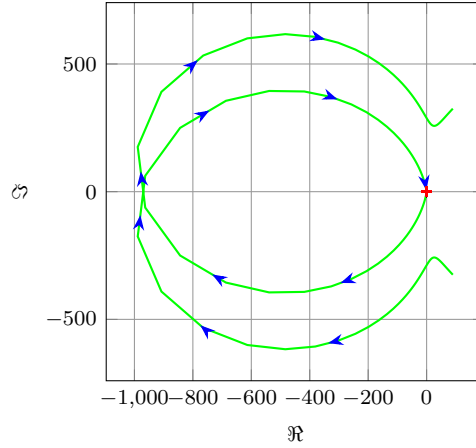


Figure 9: Output of the `\NyquistTF` macro with direction arrows. Increasing the number of samples can cause `decorations.markings` to throw errors.

Wraps a `tikzpicture/axis` pair tailored for Nyquist diagrams and hosts one or more `\addNyquist...` commands.

options:

- **domain=min:max:** Sets the frequency sweep that is forwarded to the enclosed `\addplot` commands. The default sweep is `0.01:100`; choose a symmetric interval such as `-30:30` for traditional Nyquist diagrams.
- **axes={opt}:** Appends `opt` to the `axis` options (for example, `axis equal` or `grid=both`).
- **tikz={opt}:** Adds `opt` to the surrounding `tikzpicture`.
- **prefix={name}:** Overrides the prefix used for auxiliary gnuplot tables created inside the environment.
- **phase unit=deg/rad** and **frequency unit=rad/Hz:** Synchronises the environment with the package-wide unit switches.
- Additional **key=value** pairs are passed straight to the `axis` environment, so native `pgfplots` options can be provided without extra wrapping.

The environment marks the critical point $(-1,0)$ with a red `+` by default. Pair it with `\addNyquistZPKPlot` or `\addNyquistTFPlot` to draw the actual contour.

Legacy interface (v2.1.1 and earlier):

```
\begin{NyquistPlot}[\langle options \rangle]{\langle min-freq \rangle}{\langle max-freq \rangle}
  \addNyquist...
\end{NyquistPlot}
```

The legacy optional argument accepts comma-separated tuples `obj/{opt}` or bare `{opt}`. Valid `obj` tokens are `tikz`, `axes`, and `prefix`; unqualified entries are appended to the `axis` options. The mandatory frequency limits fix the axis and plot domain.

```
\addNyquistZPKPlot \addNyquistZPKPlot[\langle plot-options \rangle]{\langle zpk-spec \rangle}
```

Adds a Nyquist contour for a transfer function supplied in ZPK form.

zpk-spec: Same as the `\NyquistZPK` macro.

plot-options: Appended verbatim to `\addplot`, enabling control over sampling density, decorations, and so on. Ensure that the surrounding `axis` (typically provided by `\begin{NyquistPlot}`) defines a suitable `domain` so that the generated parametric function matches the frequency sweep.

Legacy interface (v2.1.1 and earlier):

```
\addNyquistZPKPlot[\langle plot-options \rangle]{\langle z/{...},p/{...},k/...,d/... \rangle}
```

The legacy format is preserved for backward compatibility; the optional argument continues to be forwarded directly to `\addplot`.

`\addNyquistTFPlot` `\addNyquistTFPlot[$\langle plot-options \rangle$]{ $\langle tf-spec \rangle$ }`
 Identical to `\addNyquistZPKPlot` but expects **tf-spec** in the same polynomial format accepted by `\BodeTF`.

3.3 Nichols charts

`\NicholsZPK` `\NicholsZPK[$\langle options \rangle$]{ $\langle zpk-spec \rangle$ }`
 Generates a Nichols chart (phase versus gain) for a transfer function in ZPK form.
 The **zpk-spec** and **options** arguments are the same as those of the `\BodeZPK` macro described earlier except that the **domain=min:max** key now sets the frequency sweep for the Nichols contour (default **0.01:100**).

Supplying both frequency arguments activates the legacy behaviour for backward compatibility.

`\NicholsZPK[$\langle options \rangle$]{ $\langle zpk-spec \rangle$ }{ $\langle min-freq \rangle$ }{ $\langle max-freq \rangle$ }`

Example:

```
\NicholsZPK[
  domain=0.001:100,
  plot={red,thick,samples=2000},
  axes={ytick distance=20}]
{zeros={0,-0.1-0.5i,-0.1+0.5i},
 poles={-0.5-10i,-0.5+10i},
 gain=10}
```

Legacy usage remains available:

```
\NicholsZPK[plot/{red,thick,samples=2000}]
  {z/{0,-0.1,-0.5},{-0.1,0.5}},p/{-0.5,-10},{-0.5,10}},k/10}
{0.001}{100}
```

`\NicholsTF` `\NicholsTF[$\langle options \rangle$]{ $\langle tf-spec \rangle$ }`
 Produces a Nichols chart for transfer functions expressed in polynomial form. Same as `\NicholsZPK` except that **tf-spec** is akin to `\BodeTF`. Supplying both frequency arguments reverts the macro to the legacy interface.

Example:

```
\NicholsTF[
  domain=0.001:100,
  plot={green,thick,samples=2000}]
{numerator={10,2,2.6,0},
 denominator={1,1,100.25},
 delay=0.01}
```

Legacy usage remains available:

```
\NicholsTF[plot/{green,thick,samples=2000}]
  {num/{10,2,2.6,0},den/{1,1,100.25},d/0.01}
{0.001}{100}
```

Both examples produce the chart in Figure 10.

`NicholsChart (env.)` `\begin{NicholsChart}[$\langle options \rangle$]`
`\addNichols...`
`\end{NicholsChart}`

Provides a ready-made `tikzpicture/axis` scaffold for Nichols charts and collects `\addNichols...` commands. Current and legacy interfaces are similar to the `NyquistPlot` environment, except that the domain defaults to **0.01:100** if not supplied.

`\addNicholsZPKChart` `\addNicholsZPKChart[$\langle plot-options \rangle$]{ $\langle zpk-spec \rangle$ }`

Adds a Nichols contour for a transfer function expressed in ZPK form. Current and legacy interfaces are similar to those of `\addNyquistZPKPlot`, with the **domain** key now setting the frequency sweep for the Nichols contour, with a default of **0.01:100**.

`\addNicholsTFChart` `\addNicholsTFChart[$\langle plot-options \rangle$]{ $\langle tf-spec \rangle$ }`

Equivalent to `\addNicholsZPKChart` for transfer functions supplied in polynomial form similar to `\BodeTF`. Phase unwrapping in `gnuplot` mode mirrors the behaviour of `\BodeTF`.

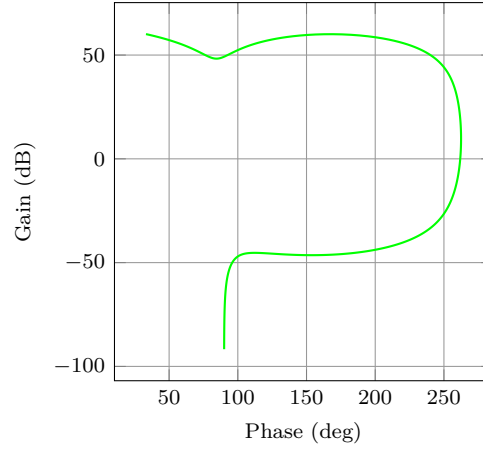


Figure 10: Output of the `\NicholsTF` macro.

3.4 Pole-zero maps

`\PoleZeroMapZPK` `\PoleZeroMapZPK` [*options*] [*zpk-spec*]

Plots the pole-zero map of a transfer function given in ZPK format, similar to MATLAB's `pzmap` function. The poles are marked with red 'x' symbols and the zeros are marked with blue 'o' symbols. The mandatory argument supports both the current `zeros={...},poles={...},gain=...` format with complex numbers expressed as `a+bi` (where `a` and `b` can be arbitrary pgfmath/gnuplot expressions) and the legacy format `z/{zeros},p/{poles},k/{gain}` with complex numbers expressed as `{a,b}`. Note that the delay parameter `d` is ignored since delays do not affect pole-zero locations. The optional argument supports the same tuples as `\NyquistZPK`.

The `scale={log}` option enables symmetric logarithmic (symlog) scaling for both axes. This is particularly useful for systems with poles and zeros spanning multiple decades. The symlog scaling preserves the sign of coordinates while applying logarithmic scaling to their magnitude, allowing visualization of both positive and negative values on the same plot.

The following example generates a standard linear pole-zero map with the zeros at the origin and at $-0.1 \pm 0.5i$, and poles at $-0.5 \pm 10i$.

```
\PoleZeroMapZPK[axes/{grid=major}]
{zeros={0, -0.1-0.5i, -0.1+0.5i}, poles={-0.5-10i, -0.5+10i}, gain=10}
```

For logarithmic scaling:

```
\PoleZeroMapZPK[scale=log]
{zeros={0, -0.1-0.5i, -0.1+0.5i}, poles={-0.5-10i, -0.5+10i}, gain=10}
```

The legacy format remains available:

```
\PoleZeroMapZPK[axes/{grid=major}]
{z/{0,{-0.1,-0.5},{-0.1,0.5}},p/{{-0.5,-10},{-0.5,10}},k/10}
\PoleZeroMapZPK[scale/log]
{zeros={0, -0.1-0.5i, -0.1+0.5i}, poles={-0.5-10i, -0.5+10i}, gain=10}
```

Both examples produce the pole-zero maps in Figure 11.

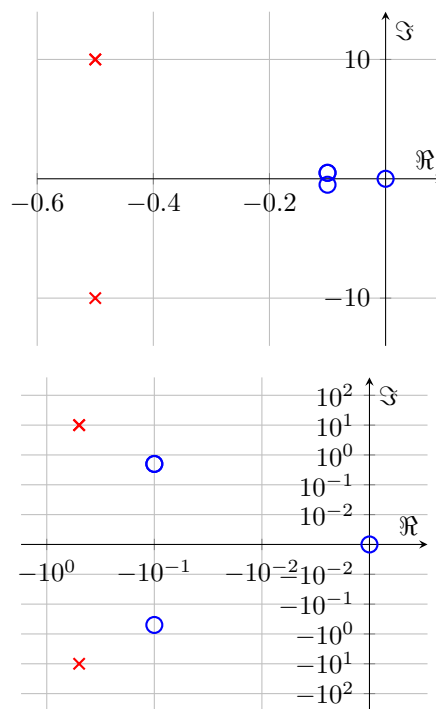


Figure 11: Pole-zero maps generated using the `\PoleZeroMapZPK` macro with linear and logarithmic scaling.

4 Implementation

4.1 Initialization

```
\n@mod We start by processing the class options.
\n@mod@p 1 \newif\if@pgfarg\@pgfargfalse
\n@mod@n 2 \DeclareOption{pgf}{
\n@pow 3 \@pgfargtrue
4 }
bp@gnuplot@id 5 \newif\if@declutterarg\@declutterargfalse
bp@gnu@prefix 6 \DeclareOption{declutter}{
7 \@declutterargtrue
8 }
9 \newif\if@radarg\@radargfalse
10 \DeclareOption{rad}{
11 \@radargtrue
12 }
13 \newif\if@hzarg\@hzargfalse
14 \DeclareOption{Hz}{
15 \@hzargtrue
16 }
17 \ProcessOptions\relax
```

New macros to unify **pgfplots** and **gnuplot**. New macros are defined for the **pow** and **mod** functions to address differences between the two math engines.

```
18 \newcommand{\n@mod}[2]{(#1)-((round((#1)/(#2)))*(#2))}
19 \newcommand{\n@mod@p}[2]{(#1)-((floor((#1)/(#2)))*(#2))}
20 \newcommand{\n@mod@n}[2]{(#1)-((floor((#1)/(#2))+1)*(#2))}
21 \if@pgfarg
22 \newcommand{\n@pow}[2]{(#1)^(#2)}
23 \else
24 \newcommand{\n@pow}[2]{(#1)**(#2)}
```

A counter so that a new data table is generated and for each new plot. If the plot macros have not changed, the tables, once generated, can be reused by **gnuplot**, which reduces compilation time. The **declutter** option is used to enable the **gnuplot** directory to declutter the working directory. The **gnuplot** prefix is set to be the name of the tex file unless the user supplies a prefix through the **prefix** option.

```
25 \newcounter{bp@gnuplot@id}
26 \setcounter{bp@gnuplot@id}{0}
27 \newcommand{\bp@prefix}{%
28 \ifx\bp@user@prefix\@empty
29 \if@declutterarg
30 gnuplot/\jobname
31 \else
32 \jobname
33 \fi
34 \else
35 \if@declutterarg
36 gnuplot/\bp@user@prefix
37 \else
38 \bp@user@prefix
39 \fi
40 \fi
41 }
42 \tikzset{
43 bp@gnu@prefix/.style={
44 id=\arabic{bp@gnuplot@id},
45 prefix=\bp@prefix
46 }
47 }
```

If the operating system is not Windows, and if the **declutter** option is passed, we create the **gnuplot** folder if it does not already exist.

```

48 \ifwindows\else
49   \if@declutterarg
50     \immediate\write18{mkdir -p gnuplot}
51   \fi
52 \fi
53 \fi

```

`\if@babel` Check if the `babel` package is loaded and generate a list of shorthands if it is. The code `\bp@short@list` is based on [this stackexchange answer](#).

```

54 \newif\if@babel\@babelfalse
55 \AtBeginDocument{%
56   \ifpackageloaded{babel}{%
57     \@babeltrue
58     \let\bp@short@list\@empty
59     \def\do#1{%
60       \begingroup
61         \lccode`\~=#1\relax
62         \lowercase{\ifbabelshorthand~{\g@addto@macro\bp@short@list{~}}{}}
63       \endgroup
64     }
65     \dospecials
66   }{}
67 }

```

`bp@style` Default axis properties for all plot macros are collected in this `pgf` style.

```

68 \pgfplotsset{
69   bp@style/.style = {
70     label style={font=\footnotesize},
71     tick label style={font=\footnotesize},
72     grid=both,
73     major grid style={color=gray!80},
74     minor grid style={color=gray!20},
75     x label style={at={(ticklabel cs:0.5)},anchor=near ticklabel},
76     y label style={at={(ticklabel cs:0.5)},anchor=near ticklabel},
77     scale only axis,
78     samples=200,
79     width=5cm,
80     log basis x=10
81   }
82 }

```

`bp@freq@filter` These macros handle the `Hz` and `rad` class options and two new `pgf` keys named `bp@freq@label` frequency unit and `bp@freq@scale` respectively.

```

\bp@ph@scale 83 \pgfplotsset{bp@freq@filter/.style = {}}
bp@ph@x@label 84 \def\bp@freq@scale{1}
bp@ph@y@label 85 \pgfplotsset{bp@freq@label/.style = {xlabel = {Frequency (rad/s)}}}
86 \pgfplotsset{bp@ph@x@label/.style = {xlabel={Phase (deg)}}}
87 \pgfplotsset{bp@ph@y@label/.style = {ylabel={Phase (deg)}}}
88 \def\bp@ph@scale{180/pi}
89 \if@radarg
90   \pgfplotsset{bp@ph@y@label/.style = {ylabel={Phase (rad)}}}
91   \pgfplotsset{bp@ph@x@label/.style = {xlabel={Phase (rad)}}}
92   \def\bp@ph@scale{1}
93   \tikzset{
94     phase unit/.initial={rad},
95     phase unit/.default={rad},
96   }
97 \else
98   \tikzset{
99     phase unit/.initial={deg},
100    phase unit/.default={deg},
101  }

```

```

102 \fi
103 \if@hzarg
104   \def\bp@freq@scale{2*pi}
105   \pgfplotsset{bp@freq@label/.style = {xlabel = {Frequency (Hz)}}}
106   \if@pgfarg
107     \pgfplotsset{bp@freq@filter/.style = {x filter/.expression={x-
log10(2*pi)}}}
108   \fi
109   \tikzset{
110     frequency unit/.initial={Hz},
111     frequency unit/.default={Hz},
112   }
113 \else
114   \tikzset{
115     frequency unit/.initial={rad},
116     frequency unit/.default={rad},
117   }
118 \fi
119 \tikzset{
120   phase unit/.is choice,
121   phase unit/deg/.code={
122     \pgfkeys{/bodeplot/phase unit=deg}
123   },
124   phase unit/rad/.code={
125     \pgfkeys{/bodeplot/phase unit=rad}
126   },
127   frequency unit/.is choice,
128   frequency unit/Hz/.code={
129     \pgfkeys{/bodeplot/frequency unit=Hz}
130   },
131   frequency unit/rad/.code={
132     \pgfkeys{/bodeplot/frequency unit=rad}
133   }
134 }

```

4.2 PGF keys interface

/bodeplot/env PGF keys for environment options.

```

135 \pgfkeys{
136   /bodeplot/env/.is family,
137   /bodeplot/env/.cd,
138   reset/.code={%
139     \pgfkeyssetvalue{/bodeplot/env/@tikz}{}
140     \pgfkeyssetvalue{/bodeplot/env/@prefix}{}
141     \pgfkeyssetvalue{/bodeplot/env/@domain}{}
142     \if@hzarg
143       \pgfkeys{/bodeplot/env/frequency unit=Hz}
144     \else
145       \pgfkeys{/bodeplot/env/frequency unit=rad}
146     \fi
147     \if@radarg
148       \pgfkeys{/bodeplot/env/phase unit=rad}
149     \else
150       \pgfkeys{/bodeplot/env/phase unit=deg}
151     \fi
152     \gdef\bp@domain@start{0.01}
153     \gdef\bp@domain@end{100}
154     \gdef\bp@axes{}
155   },
156   axes/.code={
157     \ifx\bp@axes\@empty
158       \xdef\bp@axes{\unexpanded\expandafter{#1}}
159     \else

```

```

160     \xdef\bp@axes{\unexpanded\expandafter{\bp@axes},
161     \unexpanded\expandafter{#1}}
162     \fi
163 },
164 axes/.value required,
165 tikz/.code={\pgfkeyssetvalue{/bodeplot/env/@tikz}{#1}},
166 tikz/.value required,
167 prefix/.code={\pgfkeyssetvalue{/bodeplot/env/@prefix}{#1}},
168 prefix/.value required,
169 domain/.code args={#1:#2}{%
170     \gdef\bp@domain@start{#1}%
171     \gdef\bp@domain@end{#2}%
172 },
173 domain/.value required,
174 phase unit/.initial={deg},
175 phase unit/.default={deg},
176 phase unit/.is choice,
177 phase unit/deg/.code={
178     \renewcommand{\bp@ph@scale}{180/pi}
179     \pgfplotsset{bp@ph@x@label/.style = {xlabel={Phase (deg)}}}
180     \pgfplotsset{bp@ph@y@label/.style = {ylabel={Phase (deg)}}}
181 },
182 phase unit/rad/.code={
183     \renewcommand{\bp@ph@scale}{1}
184     \pgfplotsset{bp@ph@y@label/.style = {ylabel={Phase (rad)}}}
185     \pgfplotsset{bp@ph@x@label/.style = {xlabel={Phase (rad)}}}
186 },
187 frequency unit/.initial={rad},
188 frequency unit/.default={rad},
189 frequency unit/.is choice,
190 frequency unit/Hz/.code={
191     \renewcommand{\bp@freq@scale}{2*pi}
192     \pgfplotsset{bp@freq@label/.style = {xlabel = {Frequency (Hz)}}}
193     \ifpgfarg
194     \pgfplotsset{bp@freq@filter/.style = {x filter/.expression={x-
log10(2*pi)}}}
195     \fi
196 },
197 frequency unit/rad/.code={
198     \renewcommand{\bp@freq@scale}{1}
199     \pgfplotsset{bp@freq@label/.style = {xlabel = {Frequency (rad/s)}}}
200     \ifpgfarg
201     \pgfplotsset{bp@freq@filter/.style = {x filter/.expression={x}}
202     \fi
203 },
204 .unknown/.code={%
205     \edef\bp@full{\pgfkeyscurrentkey}%
206     \def\stripbodeprefix##1/bodeplot/env/##2\relax{##2}%
207     \edef\bp@short{\expandafter\stripbodeprefix\bp@full\relax}%
208     \edef\bp@checkfull{\bp@full}%
209     \edef\bp@checkshort{\bp@short}%
210     \ifx\bp@checkfull\bp@checkshort
211     \def\removeslash##1/##2\relax{##2}%
212     \edef\bp@short{\expandafter\removeslash\bp@full\relax}%
213     \fi
214     \ifx\pgfkeyscurrentvalue\pgfkeysnovalue
215     \edef\bp@new{\bp@short}%
216     \else
217     \edef\bp@new{\bp@short=
218     {\unexpanded\expandafter{\pgfkeyscurrentvalue}}}%
219     \fi
220     \ifx\bp@new\@empty\else
221     \ifx\bp@axes\@empty

```

```

222         \xdef\bp@axes{\unexpanded\expandafter{\bp@new}}
223     \else
224         \xdef\bp@axes{\unexpanded\expandafter{\bp@axes},%
225         \unexpanded\expandafter{\bp@new}}
226     \fi
227 \fi
228 }
229 }

```

/bodeplot PGF keys for bode command options.

```

230 \pgfkeys{
231   /bodeplot/.is family,
232   /bodeplot/.cd,
233   reset/.code={%
234     \pgfkeyssetvalue{/bodeplot/@axes/mag}{}
235     \pgfkeyssetvalue{/bodeplot/@axes/ph}{}
236     \pgfkeyssetvalue{/bodeplot/@group}{}
237     \pgfkeyssetvalue{/bodeplot/@approx}{true}
238     \pgfkeyssetvalue{/bodeplot/@commands/mag}{}
239     \pgfkeyssetvalue{/bodeplot/@commands/ph}{}
240     \pgfkeyssetvalue{/bodeplot/@tikz}{}
241     \pgfkeyssetvalue{/bodeplot/@prefix}{}
242     \gdef\bp@domain@start{0.01}
243     \gdef\bp@domain@end{100}
244     \if@hzarg
245       \pgfkeys{/bodeplot/frequency unit=Hz}
246     \else
247       \pgfkeys{/bodeplot/frequency unit=rad}
248     \fi
249     \if@radarg
250       \pgfkeys{/bodeplot/phase unit=rad}
251     \else
252       \pgfkeys{/bodeplot/phase unit=deg}
253     \fi
254     \gdef\bp@mag@plot{}
255     \gdef\bp@ph@plot{}
256   },
257   plot/.code={%
258     \ifx\bp@mag@plot\@empty
259       \xdef\bp@mag@plot{\unexpanded\expandafter{#1}}
260     \else
261       \xdef\bp@mag@plot{\unexpanded\expandafter{\bp@mag@plot,#1}}
262     \fi
263     \ifx\bp@ph@plot\@empty
264       \xdef\bp@ph@plot{\unexpanded\expandafter{#1}}
265     \else
266       \xdef\bp@ph@plot{\unexpanded\expandafter{\bp@ph@plot,#1}}
267     \fi
268   },
269   plot/.value required,
270   mag plot/.code={%
271     \ifx\bp@mag@plot\@empty
272       \xdef\bp@mag@plot{\unexpanded\expandafter{#1}}
273     \else
274       \xdef\bp@mag@plot{\unexpanded\expandafter{\bp@mag@plot},
275       \unexpanded\expandafter{#1}}
276     \fi
277   },
278   mag plot/.value required,
279   ph plot/.code={%
280     \ifx\bp@ph@plot\@empty
281       \xdef\bp@ph@plot{\unexpanded\expandafter{#1}}
282     \else

```

```

283     \xdef\bp@ph@plot{\unexpanded\expandafter{\bp@ph@plot},
284     \unexpanded\expandafter{#1}}
285     \fi
286 },
287 ph plot/.value required,
288 axes/.code={%
289     \pgfkeysalso{/bodeplot/mag axes={#1}}
290     \pgfkeysalso{/bodeplot/ph axes={#1}}%
291 },
292 axes/.value required,
293 mag axes/.code={\pgfkeyssetvalue{/bodeplot/@axes/mag}{#1}},
294 mag axes/.value required,
295 ph axes/.code={\pgfkeyssetvalue{/bodeplot/@axes/ph}{#1}},
296 ph axes/.value required,
297 group/.code={\pgfkeyssetvalue{/bodeplot/@group}{#1}},
298 group/.value required,
299 approx/.code={\pgfkeyssetvalue{/bodeplot/@approx}{#1}},
300 approx/.value required,
301 commands/.code={%
302     \pgfkeysalso{/bodeplot/mag commands={#1}}
303     \pgfkeysalso{/bodeplot/ph commands={#1}}%
304 },
305 commands/.value required,
306 mag commands/.code={\pgfkeyssetvalue{/bodeplot/@commands/mag}{#1}},
307 mag commands/.value required,
308 ph commands/.code={\pgfkeyssetvalue{/bodeplot/@commands/ph}{#1}},
309 ph commands/.value required,
310 tikz/.code={\pgfkeyssetvalue{/bodeplot/@tikz}{#1}},
311 tikz/.value required,
312 prefix/.code={\pgfkeyssetvalue{/bodeplot/@prefix}{#1}},
313 prefix/.value required,
314 domain/.code args={#1:#2}{%
315     \gdef\bp@domain@start{#1}%
316     \gdef\bp@domain@end{#2}%
317 },
318 domain/.value required,
319 phase unit/.initial={deg},
320 phase unit/.default={deg},
321 phase unit/.is choice,
322 phase unit/deg/.code={
323     \renewcommand{\bp@ph@scale}{180/pi}
324     \pgfplotsset{bp@ph@x@label/.style = {xlabel={Phase (deg)}}}
325     \pgfplotsset{bp@ph@y@label/.style = {ylabel={Phase (deg)}}}
326 },
327 phase unit/rad/.code={
328     \renewcommand{\bp@ph@scale}{1}
329     \pgfplotsset{bp@ph@y@label/.style = {ylabel={Phase (rad)}}}
330     \pgfplotsset{bp@ph@x@label/.style = {xlabel={Phase (rad)}}}
331 },
332 frequency unit/.initial={rad},
333 frequency unit/.default={rad},
334 frequency unit/.is choice,
335 frequency unit/Hz/.code={
336     \renewcommand{\bp@freq@scale}{2*pi}
337     \pgfplotsset{bp@freq@label/.style = {xlabel = {Frequency (Hz)}}}
338     \ifpgfarg
339     \pgfplotsset{bp@freq@filter/.style = {x filter/.expression={x-
log10(2*pi)}}}
340     \fi
341 },
342 frequency unit/rad/.code={
343     \renewcommand{\bp@freq@scale}{1}
344     \pgfplotsset{bp@freq@label/.style = {xlabel = {Frequency (rad/s)}}}

```

```

345 \if@pgfarg
346 \pgfplotsset{bp@freq@filter/.style = {x filter/.expression={x}}}}
347 \fi
348 },
349 .unknown/.code={%
350 \edef\bp@full{\pgfkeyscurrentkey}%
351 \def\stripbodeprefix##1/bodeplot/##2\relax{##2}%
352 \edef\bp@short{\expandafter\stripbodeprefix\bp@full\relax}%
353 \edef\bp@checkfull{\bp@full}%
354 \edef\bp@checkshort{\bp@short}%
355 \ifx\bp@checkfull\bp@checkshort
356 \def\removeslash##1/##2\relax{##2}%
357 \edef\bp@short{\expandafter\removeslash\bp@full\relax}%
358 \fi
359 \ifx\pgfkeyscurrentvalue\pgfkeysnovalue
360 \edef\bp@new{\bp@short}%
361 \else
362 \edef\bp@new{\bp@short=
363 {\unexpanded\expandafter{\pgfkeyscurrentvalue}}}%
364 \fi
365 \ifx\bp@new\@empty\else
366 \ifx\bp@mag@plot\@empty
367 \xdef\bp@mag@plot{\unexpanded\expandafter{\bp@new}}
368 \else
369 \xdef\bp@mag@plot{\unexpanded\expandafter{\bp@mag@plot},%
370 \unexpanded\expandafter{\bp@new}}
371 \fi
372 \fi
373 \ifx\bp@new\@empty\else
374 \ifx\bp@ph@plot\@empty
375 \xdef\bp@ph@plot{\unexpanded\expandafter{\bp@new}}
376 \else
377 \xdef\bp@ph@plot{\unexpanded\expandafter{\bp@ph@plot},%
378 \unexpanded\expandafter{\bp@new}}
379 \fi
380 \fi
381 }
382 }

```

/bodeplot/combinedenv PGF keys for combined Bode environment options.

```

383 \pgfkeys{
384 /bodeplot/combinedenv/.is family,
385 /bodeplot/combinedenv/.cd,
386 reset/.code={%
387 \pgfkeyssetvalue{/bodeplot/combinedenv/@group}{}
388 \pgfkeyssetvalue{/bodeplot/combinedenv/@approx}{true}
389 \pgfkeyssetvalue{/bodeplot/combinedenv/@commands/mag}{}
390 \pgfkeyssetvalue{/bodeplot/combinedenv/@commands/ph}{}
391 \pgfkeyssetvalue{/bodeplot/combinedenv/@tikz}{}
392 \pgfkeyssetvalue{/bodeplot/combinedenv/@prefix}{}
393 \gdef\bp@domain@start{0.01}
394 \gdef\bp@domain@end{100}
395 \if@hzarg
396 \pgfkeys{/bodeplot/frequency unit=Hz}
397 \else
398 \pgfkeys{/bodeplot/frequency unit=rad}
399 \fi
400 \if@radarg
401 \pgfkeys{/bodeplot/phase unit=rad}
402 \else
403 \pgfkeys{/bodeplot/phase unit=deg}
404 \fi
405 \gdef\bp@mag@axes{}

```

```

406 \gdef\bp@ph@axes{}
407 },
408 axes/.code={%
409 \ifx\bp@mag@axes\empty
410 \xdef\bp@mag@axes{\unexpanded\expandafter{#1}}
411 \else
412 \xdef\bp@mag@axes{\unexpanded\expandafter{\bp@mag@axes,#1}}
413 \fi
414 \ifx\bp@ph@axes\empty
415 \xdef\bp@ph@axes{\unexpanded\expandafter{#1}}
416 \else
417 \xdef\bp@ph@axes{\unexpanded\expandafter{\bp@ph@axes,#1}}
418 \fi
419 },
420 axes/.value required,
421 mag axes/.code={%
422 \ifx\bp@mag@axes\empty
423 \xdef\bp@mag@axes{\unexpanded\expandafter{#1}}
424 \else
425 \xdef\bp@mag@axes{\unexpanded\expandafter{\bp@mag@axes,#1}}
426 \fi
427 },
428 mag axes/.value required,
429 ph axes/.code={%
430 \ifx\bp@ph@axes\empty
431 \xdef\bp@ph@axes{\unexpanded\expandafter{#1}}
432 \else
433 \xdef\bp@ph@axes{\unexpanded\expandafter{\bp@ph@axes,#1}}
434 \fi
435 },
436 ph axes/.value required,
437 group/.code={\pgfkeyssetvalue{/bodeplot/combinedenv/@group}{#1}},
438 group/.value required,
439 approx/.code={\pgfkeyssetvalue{/bodeplot/combinedenv/@approx}{#1}},
440 approx/.value required,
441 commands/.code={%
442 \pgfkeysalso{/bodeplot/combinedenv/mag commands={#1}}
443 \pgfkeysalso{/bodeplot/combinedenv/ph commands={#1}}%
444 },
445 commands/.value required,
446 mag commands/.code={\pgfkeyssetvalue{/bodeplot/combinedenv/@commands/mag}{#1}},
447 mag commands/.value required,
448 ph commands/.code={\pgfkeyssetvalue{/bodeplot/combinedenv/@commands/ph}{#1}},
449 ph commands/.value required,
450 tikz/.code={\pgfkeyssetvalue{/bodeplot/combinedenv/@tikz}{#1}},
451 tikz/.value required,
452 prefix/.code={\pgfkeyssetvalue{/bodeplot/combinedenv/@prefix}{#1}},
453 prefix/.value required,
454 domain/.code args={#1:#2}{%
455 \gdef\bp@domain@start{#1}%
456 \gdef\bp@domain@end{#2}%
457 },
458 domain/.value required,
459 phase unit/.initial={deg},
460 phase unit/.default={deg},
461 phase unit/.is choice,
462 phase unit/deg/.code={
463 \renewcommand{\bp@ph@scale}{180/pi}
464 \pgfplotsset{bp@ph@x@label/.style = {xlabel={Phase (deg)}}}
465 \pgfplotsset{bp@ph@y@label/.style = {ylabel={Phase (deg)}}}
466 },
467 phase unit/rad/.code={
468 \renewcommand{\bp@ph@scale}{1}

```



```

469 \pgfplotsset{bp@ph@y@label/.style = {ylabel={Phase (rad)}}}
470 \pgfplotsset{bp@ph@x@label/.style = {xlabel={Phase (rad)}}}
471 },
472 frequency unit/.initial={rad},
473 frequency unit/.default={rad},
474 frequency unit/.is choice,
475 frequency unit/Hz/.code={
476 \renewcommand{\bp@freq@scale}{2*pi}
477 \pgfplotsset{bp@freq@label/.style = {xlabel = {Frequency (Hz)}}}
478 \if@pgfarg
479 \pgfplotsset{bp@freq@filter/.style = {x filter/.expression={x-
log10(2*pi)}}}
480 \fi
481 },
482 frequency unit/rad/.code={
483 \renewcommand{\bp@freq@scale}{1}
484 \pgfplotsset{bp@freq@label/.style = {xlabel = {Frequency (rad/s)}}}
485 \if@pgfarg
486 \pgfplotsset{bp@freq@filter/.style = {x filter/.expression={x}}}
487 \fi
488 },
489 .unknown/.code={%
490 \edef\bp@full{\pgfkeyscurrentkey}%
491 \def\stripbodeprefix##1/bodeplot/combinedenv/##2\relax{##2}%
492 \edef\bp@short{\expandafter\stripbodeprefix\bp@full\relax}%
493 \edef\bp@checkfull{\bp@full}%
494 \edef\bp@checkshort{\bp@short}%
495 \ifx\bp@checkfull\bp@checkshort
496 \def\removeslash##1/##2\relax{##2}%
497 \edef\bp@short{\expandafter\removeslash\bp@full\relax}%
498 \fi
499 \ifx\pgfkeyscurrentvalue\pgfkeysnovalue
500 \edef\bp@new{\bp@short}%
501 \else
502 \edef\bp@new{\bp@short=
503 {\unexpanded\expandafter{\pgfkeyscurrentvalue}}}%
504 \fi
505 \ifx\bp@new\@empty\else
506 \ifx\bp@mag@axes\@empty
507 \xdef\bp@mag@axes{\unexpanded\expandafter{\bp@new}}
508 \else
509 \xdef\bp@mag@axes{\unexpanded\expandafter{\bp@mag@axes},%
510 \unexpanded\expandafter{\bp@new}}
511 \fi
512 \fi
513 \ifx\bp@new\@empty\else
514 \ifx\bp@ph@axes\@empty
515 \xdef\bp@ph@axes{\unexpanded\expandafter{\bp@new}}
516 \else
517 \xdef\bp@ph@axes{\unexpanded\expandafter{\bp@ph@axes},%
518 \unexpanded\expandafter{\bp@new}}
519 \fi
520 \fi
521 }
522 }

```

/bodeplot/zpk PGF keys for supplying zero-pole-gain-delay (ZPK) representations.

```

523 \pgfkeys{
524 /bodeplot/zpk/.is family,
525 /bodeplot/zpk/.cd,
526 reset/.code={%
527 \pgfkeyssetvalue{/bodeplot/zpk/@zeros}{}
528 \pgfkeyssetvalue{/bodeplot/zpk/@poles}{}

```

```

529 \pgfkeyssetvalue{/bodeplot/zpk/@gain}{}
530 \pgfkeyssetvalue{/bodeplot/zpk/@delay}{}
531 },
532 zeros/.code={\pgfkeyssetvalue{/bodeplot/zpk/@zeros}{#1}},
533 zeros/.value required,
534 poles/.code={\pgfkeyssetvalue{/bodeplot/zpk/@poles}{#1}},
535 poles/.value required,
536 gain/.code={\pgfkeyssetvalue{/bodeplot/zpk/@gain}{#1}},
537 gain/.value required,
538 delay/.code={\pgfkeyssetvalue{/bodeplot/zpk/@delay}{#1}},
539 delay/.value required,
540 }

```

/bodeplot/tf PGF keys for supplying transfer function (TF) representations.

```

541 \pgfkeys{
542 /bodeplot/tf/.is family,
543 /bodeplot/tf/.cd,
544 reset/.code={%
545 \pgfkeyssetvalue{/bodeplot/tf/@numerator}{}
546 \pgfkeyssetvalue{/bodeplot/tf/@denominator}{}
547 \pgfkeyssetvalue{/bodeplot/tf/@delay}{}
548 },
549 numerator/.code={\pgfkeyssetvalue{/bodeplot/tf/@numerator}{#1}},
550 numerator/.value required,
551 denominator/.code={\pgfkeyssetvalue{/bodeplot/tf/@denominator}{#1}},
552 denominator/.value required,
553 delay/.code={\pgfkeyssetvalue{/bodeplot/tf/@delay}{#1}},
554 delay/.value required
555 }

```

/bodeplot/add PGF keys for adding asymptotic and linear plots.

```

556 \pgfkeys{
557 /bodeplot/add/.is family,
558 /bodeplot/add/.cd,
559 reset/.code={%
560 \gdef\bp@add@0{}%
561 },
562 true/.default={},
563 true/.code={%
564 \pgfutil@ifempty{#1}{
565 \ifx\bp@add@0\empty
566 \g@addto@macro\bp@add@0{true}%
567 \else
568 \g@addto@macro\bp@add@0{,true}%
569 \fi
570 }{%
571 \ifx\bp@add@0\empty
572 \g@addto@macro\bp@add@0{true/{#1}}%
573 \else
574 \g@addto@macro\bp@add@0{,true/{#1}}%
575 \fi
576 }
577 },
578 linear/.default={},
579 linear/.code={%
580 \pgfutil@ifempty{#1}{
581 \ifx\bp@add@0\empty
582 \g@addto@macro\bp@add@0{linear}%
583 \else
584 \g@addto@macro\bp@add@0{,linear}%
585 \fi
586 }{%
587 \ifx\bp@add@0\empty

```

```

588     \g@addto@macro\bp@add@0{linear/{#1}}%
589     \else
590     \g@addto@macro\bp@add@0{,linear/{#1}}%
591     \fi
592 }
593 },
594 asymptotic/.default={},
595 asymptotic/.code={%
596     \pgfutil@ifempty{#1}{
597         \ifx\bp@add@0\empty
598             \g@addto@macro\bp@add@0{asymptotic}%
599         \else
600             \g@addto@macro\bp@add@0{,asymptotic}%
601         \fi
602     }{%
603         \ifx\bp@add@0\empty
604             \g@addto@macro\bp@add@0{asymptotic/{#1}}%
605         \else
606             \g@addto@macro\bp@add@0{,asymptotic/{#1}}%
607         \fi
608     }%
609 },
610 .unknown/.code={%
611     \edef\bp@full{\pgfkeyscurrentkey}%
612     \def\stripbodeprefix##1/bodeplot/add/##2\relax{##2}%
613     \edef\bp@short{\expandafter\stripbodeprefix\bp@full\relax}%
614     \edef\bp@checkfull{\bp@full}%
615     \edef\bp@checkshort{\bp@short}%
616     \ifx\bp@checkfull\bp@checkshort
617         \def\removeslash##1/##2\relax{##2}%
618     \edef\bp@short{\expandafter\removeslash\bp@full\relax}%
619     \fi
620     \ifx\pgfkeyscurrentvalue\pgfkeysnovalue
621         \edef\bp@new{\bp@short}%
622     \else
623         \edef\bp@new{\bp@short=
624             {\unexpanded\expandafter{\pgfkeyscurrentvalue}}}%
625     \fi
626     \ifx\bp@new\empty\else
627         \ifx\bp@add@0\empty
628             \xdef\bp@add@0{\unexpanded\expandafter{\bp@new}}
629         \else
630             \xdef\bp@add@0{\unexpanded\expandafter{\bp@add@0},%
631                 \unexpanded\expandafter{\bp@new}}
632         \fi
633     \fi
634 }
635 }

```

/bodeplot/nyquist PGF keys for Nyquist plot options.

```

636 \pgfkeys{
637     /bodeplot/nyquist/.is family,
638     /bodeplot/nyquist/.cd,
639     reset/.code={
640         \pgfkeyssetvalue{/bodeplot/nyquist/@axes}{}
641         \pgfkeyssetvalue{/bodeplot/nyquist/@plot}{}
642         \pgfkeyssetvalue{/bodeplot/nyquist/@commands}{}
643         \pgfkeyssetvalue{/bodeplot/nyquist/@tikz}{}
644         \pgfkeyssetvalue{/bodeplot/nyquist/@prefix}{}
645         \gdef\bp@domain@start{-30}
646         \gdef\bp@domain@end{30}
647         \gdef\bp@plot{}
648     },

```

```

649 axes/.code={\pgfkeyssetvalue{/bodeplot/nyquist/@axes}{#1}},
650 axes/.value required,
651 plot/.code={%
652   \xdef\bp@plot{\unexpanded\expandafter{#1}}
653 },
654 plot/.value required,
655 commands/.code={\pgfkeyssetvalue{/bodeplot/nyquist/@commands}{#1}},
656 commands/.value required,
657 tikz/.code={\pgfkeyssetvalue{/bodeplot/nyquist/@tikz}{#1}},
658 tikz/.value required,
659 prefix/.code={\pgfkeyssetvalue{/bodeplot/nyquist/@prefix}{#1}},
660 prefix/.value required,
661 domain/.code args={#1:#2}{\gdef\bp@domain@start{#1}\gdef\bp@domain@end{#2}},
662 domain/.value required,
663 .unknown/.code={%
664   \edef\bp@full{\pgfkeyscurrentkey}%
665   \def\stripbodeprefix##1/bodeplot/nyquist/##2\relax{##2}%
666   \edef\bp@short{\expandafter\stripbodeprefix\bp@full\relax}%
667   \edef\bp@checkfull{\bp@full}%
668   \edef\bp@checkshort{\bp@short}%
669   \ifx\bp@checkfull\bp@checkshort
670     \def\removeslash##1/##2\relax{##2}%
671     \edef\bp@short{\expandafter\removeslash\bp@full\relax}%
672   \fi
673   \ifx\pgfkeyscurrentvalue\pgfkeysnovalue
674     \edef\bp@new{\bp@short}%
675   \else
676     \edef\bp@new{\bp@short=
677       {\unexpanded\expandafter{\pgfkeyscurrentvalue}}}%
678   \fi
679   \ifx\bp@new\@empty\else
680     \ifx\bp@plot\@empty
681       \xdef\bp@plot{\unexpanded\expandafter{\bp@new}}
682     \else
683       \xdef\bp@plot{\unexpanded\expandafter{\bp@plot},%
684         \unexpanded\expandafter{\bp@new}}
685     \fi
686   \fi
687 }
688 }

```

/bodeplot/nichols PGF keys for Nichols plot options.

```

689 \pgfkeys{
690   /bodeplot/nichols/.is family,
691   /bodeplot/nichols/.cd,
692   reset/.code={
693     \pgfkeyssetvalue{/bodeplot/nichols/@axes}{}
694     \pgfkeyssetvalue{/bodeplot/nichols/@plot}{}
695     \pgfkeyssetvalue{/bodeplot/nichols/@commands}{}
696     \pgfkeyssetvalue{/bodeplot/nichols/@tikz}{}
697     \pgfkeyssetvalue{/bodeplot/nichols/@prefix}{}
698     \gdef\bp@domain@start{0.01}
699     \gdef\bp@domain@end{100}
700     \gdef\bp@plot{}
701   },
702   axes/.code={\pgfkeyssetvalue{/bodeplot/nichols/@axes}{#1}},
703   axes/.value required,
704   plot/.code={%
705     \xdef\bp@plot{\unexpanded\expandafter{#1}}
706   },
707   plot/.value required,
708   commands/.code={\pgfkeyssetvalue{/bodeplot/nichols/@commands}{#1}},
709   commands/.value required,

```

```

710 tikz/.code={\pgfkeyssetvalue{/bodeplot/nichols/@tikz}{#1}},
711 tikz/.value required,
712 prefix/.code={\pgfkeyssetvalue{/bodeplot/nichols/@prefix}{#1}},
713 prefix/.value required,
714 domain/.code args={#1:#2}{\gdef\bp@domain@start{#1}\gdef\bp@domain@end{#2}},
715 domain/.value required,
716 .unknown/.code={%
717   \edef\bp@full{\pgfkeyscurrentkey}%
718   \def\stripbodeprefix##1/bodeplot/nichols/##2\relax{##2}%
719   \edef\bp@short{\expandafter\stripbodeprefix\bp@full\relax}%
720   \edef\bp@checkfull{\bp@full}%
721   \edef\bp@checkshort{\bp@short}%
722   \ifx\bp@checkfull\bp@checkshort
723     \def\removeslash##1/##2\relax{##2}%
724     \edef\bp@short{\expandafter\removeslash\bp@full\relax}%
725   \fi
726   \ifx\pgfkeyscurrentvalue\pgfkeysnovalue
727     \edef\bp@new{\bp@short}%
728   \else
729     \edef\bp@new{\bp@short=
730       {\unexpanded\expandafter{\pgfkeyscurrentvalue}}}%
731   \fi
732   \ifx\bp@new\@empty\else
733     \ifx\bp@plot\@empty
734       \xdef\bp@plot{\unexpanded\expandafter{\bp@new}}
735     \else
736       \xdef\bp@plot{\unexpanded\expandafter{\bp@plot},%
737         \unexpanded\expandafter{\bp@new}}
738     \fi
739   \fi
740 }
741 }

```

/bodeplot/pzmap PGF keys for pole-zero map options.

```

742 \pgfkeys{
743   /bodeplot/pzmap/.is family,
744   /bodeplot/pzmap/.cd,
745   reset/.code={
746     \pgfkeyssetvalue{/bodeplot/pzmap/@axes}{}
747     \pgfkeyssetvalue{/bodeplot/pzmap/@plot}{}
748     \pgfkeyssetvalue{/bodeplot/pzmap/@commands}{}
749     \pgfkeyssetvalue{/bodeplot/pzmap/@tikz}{}
750     \pgfkeyssetvalue{/bodeplot/pzmap/@prefix}{}
751     \pgfkeyssetvalue{/bodeplot/pzmap/@scale}{linear}
752   },
753   axes/.code={\pgfkeyssetvalue{/bodeplot/pzmap/@axes}{#1}},
754   axes/.value required,
755   plot/.code={\pgfkeyssetvalue{/bodeplot/pzmap/@plot}{#1}},
756   plot/.value required,
757   commands/.code={\pgfkeyssetvalue{/bodeplot/pzmap/@commands}{#1}},
758   commands/.value required,
759   tikz/.code={\pgfkeyssetvalue{/bodeplot/pzmap/@tikz}{#1}},
760   tikz/.value required,
761   prefix/.code={\pgfkeyssetvalue{/bodeplot/pzmap/@prefix}{#1}},
762   prefix/.value required,
763   scale/.initial=linear,
764   scale/.default=linear,
765   scale/.is choice,
766   scale/linear/.code={\pgfkeyssetvalue{/bodeplot/pzmap/@scale}{linear}},
767   scale/log/.code={\pgfkeyssetvalue{/bodeplot/pzmap/@scale}{log}},
768 }

```

4.3 Parametric function generators for poles, zeros, gains, and delays.

All calculations are carried out assuming that frequency inputs are in **rad/s**. Magnitude outputs are in **dB** and phase outputs are in degrees or radians, depending on the value of `\bp@ph@scale`.

`\MagK` True, linear, and asymptotic magnitude and phase parametric functions for a pure gain
`\MagKAsymp` $G(s) = k + 0i$. The macros take two arguments corresponding to real and imaginary
`\MagKLin` part of the gain to facilitate code reuse between delays, gains, poles, and zeros, but only
`\PhK` real gains are supported. The second argument, if supplied, is ignored.

```

\PhKAsymp 769 \newcommand*\MagK[2]{(20*log10(abs(#1)))}
\PhKLin    770 \newcommand*\MagKAsymp{\MagK}
           771 \newcommand*\MagKLin{\MagK}
           772 \newcommand*\PhK[2]{((#1<0?-pi:0)*\bp@ph@scale)}
           773 \newcommand*\PhKAsymp{\PhK}
           774 \newcommand*\PhKLin{\PhK}

```

`\PhKAsymp` True magnitude and phase parametric functions for a pure delay $G(s) = e^{-Ts}$. The
`\PhKLin` macros take two arguments corresponding to real and imaginary part of the gain to
facilitate code reuse between delays, gains, poles, and zeros, but only real gains are
supported. The second argument, if supplied, is ignored.

```

           775 \newcommand*\MagDel[2]{0}
           776 \newcommand*\PhDel[2]{(-#1*t*\bp@ph@scale)}

```

`\MagPole` These macros are the building blocks for most of the plotting functions provided by this
`\MagPoleAsymp` package. We start with Parametric function for the true magnitude of a complex pole.
`\MagPoleLin` 777 \newcommand*\MagPole[2]

```

\PhPole    778 {( -20*log10(sqrt(\n@pow{#1}{2} + \n@pow{t - (#2)}{2})))}
\PhPoleAsymp Parametric function for linear approximation of the magnitude of a complex pole.
\PhPoleLin 779 \newcommand*\MagPoleLin[2]{(t < sqrt(\n@pow{#1}{2} + \n@pow{#2}{2}) ?
           780 -20*log10(sqrt(\n@pow{#1}{2} + \n@pow{#2}{2})) :
           781 -20*log10(t)
           782 )}
```

Parametric function for asymptotic approximation of the magnitude of a complex pole,
same as linear approximation.

```

           783 \newcommand*\MagPoleAsymp{\MagPoleLin}

```

Parametric function for the true phase of a complex pole.

```

           784 \newcommand*\PhPole[2]{((#1 > 0 ? (#2 > 0 ?
           785 (\n@mod@p{-atan2((t - (#2)),-(#1))}{2*pi}) :
           786 (-atan2((t - (#2)),-(#1)))) :
           787 (-atan2((t - (#2)),-(#1))))*\bp@ph@scale)}

```

Parametric function for linear approximation of the phase of a complex pole.

```

           788 \newcommand*\PhPoleLin[2]{
           789 ((abs(#1)+abs(#2) == 0 ? -pi/2 :
           790 (t < (sqrt(\n@pow{#1}{2} + \n@pow{#2}{2}) /
           791 (\n@pow{10}{sqrt(\n@pow{#1}{2}/(\n@pow{#1}{2} + \n@pow{#2}{2})))) ?
           792 (-atan2(-(#2),-(#1))) :
           793 (t >= (sqrt(\n@pow{#1}{2} + \n@pow{#2}{2}) *
           794 (\n@pow{10}{sqrt(\n@pow{#1}{2}/(\n@pow{#1}{2} + \n@pow{#2}{2})))) ?
           795 (#2>0?(#1>0?3*pi/2:-pi/2):-pi/2) :
           796 (-atan2(-(#2),-(#1)) + (log10(t/(sqrt(\n@pow{#1}{2} + \n@pow{#2}{2}) /
           797 (\n@pow{10}{sqrt(\n@pow{#1}{2}/(\n@pow{#1}{2} +
           798 \n@pow{#2}{2})))))))*((#2>0?(#1>0?3*pi/2:-pi/2):-pi/2) + atan2(-(#2),-
           799 (#1)))/
           800 (\n@pow{#1}{2} + \n@pow{#2}{2})))))))*\bp@ph@scale)}

```

Parametric function for asymptotic approximation of the phase of a complex pole.

```

           801 \newcommand*\PhPoleAsymp[2]{((t < (sqrt(\n@pow{#1}{2} + \n@pow{#2}{2})) ?

```

```

802 (-atan2(-(#2),-(#1))) :
803 (#2>0?(#1>0?3*pi/2:-pi/2):-pi/2))*\bp@ph@scale)}

```

\MagZero Plots of zeros are defined to be negative of plots of poles. The 0- is necessary due to a bug in gnuplot (fixed in version 5.4, patchlevel 3).

```

\MagZeroAsymp
\MagZeroLin 804 \newcommand*\MagZero{0-\MagPole}
\PhZero      805 \newcommand*\MagZeroLin{0-\MagPoleLin}
\PhZeroAsymp 806 \newcommand*\MagZeroAsymp{0-\MagPoleAsymp}
\PhZeroLin    807 \newcommand*\PhZero{0-\PhPole}
              808 \newcommand*\PhZeroLin{0-\PhPoleLin}
              809 \newcommand*\PhZeroAsymp{0-\PhPoleAsymp}

```

4.4 Second order systems.

Although second order systems can be dealt with using the macros defined so far, the following dedicated macros for second order systems involve less computation.

\MagCSPoles Consider the canonical second order transfer function $G(s) = \frac{1}{s^2 + 2\zeta\omega_n s + \omega_n^2}$. We start with true, linear, and asymptotic magnitude plots for this transfer function.

```

\MagCSPolesLin 810 \newcommand*\MagCSPoles[2]{(-20*log10(sqrt(\n@pow{\n@pow{#2}{2}
\PhCSPoles      - \n@pow{t}{2}}{2} + \n@pow{2*#1*#2*t}{2}})))}
\PhCSPolesAsymp 811 \newcommand*\MagCSPolesLin[2]{(t < #2 ? -40*log10(#2) : - 40*log10(t))}
\PhCSPolesLin    812 \newcommand*\MagCSPolesAsymp{\MagCSPolesLin}

```

\MagCSZeros True, linear, and asymptotic phase plots for the canonical second order transfer function.

```

\MagCSZerosAsymp 814 \newcommand*\PhCSPoles[2]{((-atan2((2*(#1)*(#2)*t),(\n@pow{#2}{2}
\MagCSZerosLin    - \n@pow{t}{2}}))) * \bp@ph@scale)}
\PhCSZeros        815 \newcommand*\PhCSPolesLin[2]{((t < (#2 / (\n@pow{10}{abs(#1)})) ?
\PhCSZerosAsymp    816 0 :
\PhCSZerosLin      817 (t >= (#2 * (\n@pow{10}{abs(#1)})) ?
                    818 (#1>0 ? -pi : pi) :
                    819 (#1>0 ? (-pi*(log10(t*(\n@pow{10}{#1})/#2))/(2*#1)) :
                    820 (pi*(log10(t*(\n@pow{10}{abs(#1)})/#2))/(2*abs(#1)))) * \bp@ph@scale)}
                    821 \newcommand*\PhCSPolesAsymp[2]{((#1>0?(t<#2?0:-
                    822 pi):(t<#2?0:pi))*\bp@ph@scale)}

```

Plots of the inverse function $G(s) = s^2 + 2\zeta\omega_n s + \omega_n^2$ are defined to be negative of plots of poles. The 0- is necessary due to a bug in gnuplot (fixed in version 5.4, patchlevel 3).

```

823 \newcommand*\MagCSZeros{0-\MagCSPoles}
824 \newcommand*\MagCSZerosLin{0-\MagCSPolesLin}
825 \newcommand*\MagCSZerosAsymp{0-\MagCSPolesAsymp}
826 \newcommand*\PhCSZeros{0-\PhCSPoles}
827 \newcommand*\PhCSZerosLin{0-\PhCSPolesLin}
828 \newcommand*\PhCSZerosAsymp{0-\PhCSPolesAsymp}

```

\MagCSPolesPeak These macros are used to add a resonant peak to linear and asymptotic plots of canonical second order poles and zeros. Since the plots are parametric, a separate \draw command is needed to add a vertical arrow.

```

829 \newcommand*\MagCSPolesPeak[3][]{
830 \draw[#1,->] (axis cs:{#3},{-40*log10(#3)}) --
831 (axis cs:{#3},{-40*log10(#3)-20*log10(2*abs(#2))})
832 }
833 \newcommand*\MagCSZerosPeak[3][]{
834 \draw[#1,->] (axis cs:{#3},{40*log10(#3)}) --
835 (axis cs:{#3},{40*log10(#3)+20*log10(2*abs(#2))})
836 }

```

\MagSOPoles Consider a general second order transfer function $G(s) = \frac{1}{s^2 + as + b}$. We start with true, linear, and asymptotic magnitude plots for this transfer function.

```

\MagSOPolesLin 837 \newcommand*\MagSOPoles[2]{
\PhSOPoles      (-20*log10(sqrt(\n@pow{#2} - \n@pow{t}{2}}{2} + \n@pow{#1*t}{2}})))}
\PhSOPolesAsymp 838 \newcommand*\MagSOPolesLin[2]{
\PhSOPolesLin
\MagSOPoles
\MagSOPolesAsymp
\MagSOPolesLin
\PhSOPoles
\PhSOPolesAsymp

```

```

840 (t < sqrt(abs(#2)) ? -20*log10(abs(#2)) : - 40*log10(t))}
841 \newcommand*\MagSOPolesAsymp{\MagSOPolesLin}

```

True, linear, and asymptotic phase plots for the general second order transfer function.

```

842 \newcommand*\PhSOPoles[2]{((-atan2((#1)*t,((#2) - \n@pow{t}{2}))) * \bp@ph@scale)}
843 \newcommand*\PhSOPolesLin[2]{((#2>0 ?
844 \PhCSPolesLin{(#1/(2*sqrt(#2)))}{(sqrt(#2))} :
845 (#1>0 ? -pi : pi))}
846 \newcommand*\PhSOPolesAsymp[2]{((#2>0 ?
847 \PhCSPolesAsymp{(#1/(2*sqrt(#2)))}{(sqrt(#2))} :
848 (#1>0 ? -pi : pi))}

```

Plots of the inverse function $G(s) = s^2 + as + b$ are defined to be negative of plots of poles. The 0- is necessary due to a bug in **gnuplot** (fixed in version 5.4, patchlevel 3).

```

849 \newcommand*\MagSOPoles{0-\MagSOPoles}
850 \newcommand*\MagSOPolesLin{0-\MagSOPolesLin}
851 \newcommand*\MagSOPolesAsymp{0-\MagSOPolesAsymp}
852 \newcommand*\PhSOPoles{0-\PhSOPoles}
853 \newcommand*\PhSOPolesLin{0-\PhSOPolesLin}
854 \newcommand*\PhSOPolesAsymp{0-\PhSOPolesAsymp}

```

\MagSOPolesPeak These macros are used to add a resonant peak to linear and asymptotic plots of general second order poles and zeros. Since the plots are parametric, a separate **\draw** command is needed to add a vertical arrow.

```

855 \newcommand*\MagSOPolesPeak[3]{\draw[>] (axis cs:{sqrt(abs(#3))},{-20*log10(abs(#3))} --
856 (axis cs:{sqrt(abs(#3))},{-20*log10(abs(#3)) -
857 20*log10(abs(#2/sqrt(abs(#3)))});
858 }
859 }
860 \newcommand*\MagSOPolesPeak[3]{\draw[>] (axis cs:{sqrt(abs(#3))},{20*log10(abs(#3))} --
861 (axis cs:{sqrt(abs(#3))},{20*log10(abs(#3)) +
862 20*log10(abs(#2/sqrt(abs(#3)))});
863 }
864 }

```

4.5 Commands for Bode plots

4.5.1 User macros

\BodeZPK This macro takes lists of complex poles and zeros of the form **{re,im}**, and values of gain and delay as inputs and constructs parametric functions for the Bode magnitude and phase plots. This is done by adding together the parametric functions generated by the macros for individual zeros, poles, gain, and delay, described above. The parametric functions are then plotted in a **tikzpicture** environment using the **\addplot** macro. Unless the package is loaded with the option **pgf**, the parametric functions are evaluated using **gnuplot**.

```

865 \NewDocumentCommand{\BodeZPK}{0}{m G{} G{}}{%

```

The macro now accepts an optional argument followed by a mandatory ZPK specification and two optional frequency arguments. If the frequency arguments are not provided, the **pgfkeys** interface introduced in v3.0 is used. Otherwise, the legacy interface is used for backward compatibility.

```

866 \pgfutil@ifempty{#3}{%
867 \pgfkeys{/bodeplot/.cd, reset}
868 \pgfkeys{/bodeplot/.cd, #1}
869 \pgfkeysgetvalue{/bodeplot/@axes/mag}{\bp@mag@axes}
870 \pgfkeysgetvalue{/bodeplot/@axes/ph}{\bp@ph@axes}
871 \pgfkeysgetvalue{/bodeplot/@group}{\bp@group}
872 \pgfkeysgetvalue{/bodeplot/@approx}{\bp@approx}
873 \pgfkeysgetvalue{/bodeplot/@commands/mag}{\bp@mag@commands}
874 \pgfkeysgetvalue{/bodeplot/@commands/ph}{\bp@ph@commands}
875 \pgfkeysgetvalue{/bodeplot/@tikz}{\bp@tikz}
876 \pgfkeysgetvalue{/bodeplot/@prefix}{\bp@user@prefix}

```



```

877 \bp@zpk@new@to@legacy{#2}
878 }{%
879 \if@radarg
880 \pgfkeys{/bodeplot/phase unit=rad}
881 \else
882 \pgfkeys{/bodeplot/phase unit=deg}
883 \fi
884 \if@hzarg
885 \pgfkeys{/bodeplot/frequency unit=Hz}
886 \else
887 \pgfkeys{/bodeplot/frequency unit=rad}
888 \fi
889 \bp@parse@opt{#1}
890 \edef\bp@legacy{#2}
891 \edef\bp@domain@start{#3}
892 \edef\bp@domain@end{#4}
893 }%
894
895 \edef\bp@tmp{\noexpand\begin{tikzpicture}
896 [\unexpanded\expandafter{\bp@tikz}]}
897 \bp@tmp
898 \gdef\bp@mag{}
899 \gdef\bp@ph{}
900 \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@approx}{\bp@legacy}
901 \edef\bp@tmp{\noexpand\begin{groupplot}[
902 bp@style,
903 xmin=\bp@domain@start,
904 xmax=\bp@domain@end,
905 domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
906 height=2.5cm,
907 xmode=log,
908 group style = {group size = 1 by 2,vertical sep=0.25cm},
909 \unexpanded\expandafter{\bp@group}
910 ]}
911 \bp@tmp
912 \edef\bp@tmp@mag{\noexpand\nextgroupplot
913 [ylabel={Gain (dB)}, xmajor ticks=false, \bp@mag@axes]
914 \noexpand\addplot [bp@freq@filter, variable=t, thick,
915 \unexpanded\expandafter{\bp@mag@plot}]}
916 \edef\bp@tmp@ph{\noexpand\nextgroupplot
917 [bp@ph@y@label, bp@freq@label, \bp@ph@axes]
918 \noexpand\addplot [bp@freq@filter, variable=t, thick, trig format plots=rad,
919 \unexpanded\expandafter{\bp@ph@plot}]}
920 \if@pgfarg
921 \bp@tmp@mag {\bp@mag};
922 \bp@mag@commands
923 \bp@tmp@ph {\bp@ph};
924 \bp@ph@commands
925 \else
926 \stepcounter{bp@gnuplot@id}
927 \edef\gnu@id{\arabic{bp@gnuplot@id}}
928 \bp@gnu@plot{\bp@mag}{\gnu@id}
929 \expandafter\bp@tmp@mag\bp@gnu@cmd
930 \bp@mag@commands
931 \stepcounter{bp@gnuplot@id}
932 \edef\gnu@id{\arabic{bp@gnuplot@id}}
933 \bp@gnu@plot{\bp@ph}{\gnu@id}
934 \expandafter\bp@tmp@ph\bp@gnu@cmd
935 \bp@ph@commands
936 \fi
937 \end{groupplot}
938 \end{tikzpicture}
939 }

```

The following code handles active characters to avoid conflicts with ‘babel.’

```

940 \AtBeginDocument{%
941   \if@babel
942   \let\Orig@BodeZPK\BodeZPK
943   \renewcommand{\BodeZPK}{%
944     \expandafter\shorthandoff\expandafter{\bp@short@list}
945     \BodeZPK@Shorthandoff
946   }
947   \NewDocumentCommand{\BodeZPK@Shorthandoff}{ 0{ } m G{ } G{ } }{%
948     \Orig@BodeZPK[#1]{#2}{#3}{#4}
949     \expandafter\shorthandon\expandafter{\bp@short@list}
950   }
951   \fi
952 }

```

\BodeTF Implementation of this macro is very similar to the **\BodeZPK** macro above. The only difference is the lack of linear and asymptotic plots and slightly different parsing of the mandatory arguments.

```

953 \NewDocumentCommand{\BodeTF}{ 0{ } m G{ } G{ } }{%
954   \if@pgfarg
955     \bp@tf@to@zpk{#2}{\bp@tmp@zpk}{\bp@tmp@status}
956     \ifnum\pdf@strcmp{\bp@tmp@status}{true}=0
957       \pgfutil@ifempty{#3}{%
958         \BodeZPK[#1]{\bp@tmp@zpk}%
959       }{%
960         \BodeZPK[domain=#3:#4,#1]{\bp@tmp@zpk}%
961       }%
962       \def\bp@fallback{false}
963     \else
964       \def\bp@fallback{true}
965     \fi
966   \else
967     \def\bp@fallback{true}
968   \fi
969   \ifnum\pdf@strcmp{\bp@fallback}{true}=0
970     \pgfutil@ifempty{#3}{%
971       \pgfkeys{/bodeplot/.cd, reset}
972       \pgfkeys{/bodeplot/.cd, #1}
973       \pgfkeysgetvalue{/bodeplot/@axes/mag}{\bp@mag@axes}
974       \pgfkeysgetvalue{/bodeplot/@axes/ph}{\bp@ph@axes}
975       \pgfkeysgetvalue{/bodeplot/@group}{\bp@group}
976       \pgfkeysgetvalue{/bodeplot/@approx}{\bp@approx}
977       \pgfkeysgetvalue{/bodeplot/@commands/mag}{\bp@mag@commands}
978       \pgfkeysgetvalue{/bodeplot/@commands/ph}{\bp@ph@commands}
979       \pgfkeysgetvalue{/bodeplot/@tikz}{\bp@tikz}
980       \pgfkeysgetvalue{/bodeplot/@prefix}{\bp@user@prefix}
981       \bp@tf@new@to@legacy{#2}
982     }{%
983       \if@radarg
984         \pgfkeys{/bodeplot/phase unit=rad}
985       \else
986         \pgfkeys{/bodeplot/phase unit=deg}
987       \fi
988       \if@hzarg
989         \pgfkeys{/bodeplot/frequency unit=Hz}
990       \else
991         \pgfkeys{/bodeplot/frequency unit=rad}
992       \fi
993       \bp@parse@opt{#1}
994       \edef\bp@legacy{#2}
995       \edef\bp@domain@start{#3}
996       \edef\bp@domain@end{#4}
997     }%

```

```

998 \edef\bp@cmd{\noexpand\begin{tikzpicture}
999 [\unexpanded\expandafter{\bp@tikz}]}
1000 \bp@cmd
1001 \gdef\bp@mag{}
1002 \gdef\bp@ph{}
1003 \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}
1004 \edef\bp@cmd{\noexpand\begin{groupplot}[
1005 bp@style,
1006 xmin=\bp@domain@start,
1007 xmax=\bp@domain@end,
1008 domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
1009 height=2.5cm,
1010 xmode=log,
1011 group style = {group size = 1 by 2,vertical sep=0.25cm},
1012 \unexpanded\expandafter{\bp@group}
1013 ]}
1014 \bp@cmd
1015 \edef\bp@mag@cmd{\noexpand\nextgroupplot
1016 [ylabel={Gain (dB)}, xmajor ticks=false, \bp@mag@axes]
1017 \noexpand\addplot [bp@freq@filter, variable=t, thick,
1018 \unexpanded\expandafter{\bp@mag@plot}]}
1019 \edef\bp@ph@cmd{\noexpand\nextgroupplot
1020 [bp@ph@y@label, bp@freq@label, \bp@ph@axes]
1021 \noexpand\addplot [bp@freq@filter, variable=t, thick, trig format plots=rad,
1022 \unexpanded\expandafter{\bp@ph@plot}]}
1023 \ifpgfarg
1024 \bp@mag@cmd {\bp@mag};
1025 \bp@mag@commands
1026 \bp@ph@cmd {\n@mod{\bp@ph}{2*pi*\bp@ph@scale}};
1027 \bp@ph@commands
1028 \else
1029 \stepcounter{bp@gnuplot@id}
1030 \edef\gnu@id{\arabic{bp@gnuplot@id}}
1031 \bp@gnu@plot{\bp@mag}{\gnu@id}
1032 \expandafter\bp@mag@cmd\bp@gnu@cmd
1033 \bp@mag@commands
1034 \stepcounter{bp@gnuplot@id}
1035 \edef\gnu@id{\arabic{bp@gnuplot@id}}
1036 \bp@gnu@unwrap@plot{\bp@ph}{\gnu@id}
1037 \expandafter\bp@ph@cmd\bp@gnu@cmd
1038 \bp@ph@commands
1039 \fi
1040 \end{groupplot}
1041 \end{tikzpicture}
1042 \fi
1043 }

```

The following code handles active characters to avoid conflicts with ‘babel.’

```

1044 \AtBeginDocument{
1045 \if@babel
1046 \let\Orig@BodeTF\BodeTF
1047 \renewcommand{\BodeTF}{%
1048 \expandafter\shorthandoff\expandafter{\bp@short@list}
1049 \BodeTF@Shorthandoff
1050 }
1051 \NewDocumentCommand{\BodeTF@Shorthandoff}{ 0 } m G{} G{} }{%
1052 \Orig@BodeTF[#1]{#2}{#3}{#4}
1053 \expandafter\shandon\expandafter{\bp@short@list}
1054 }
1055 \fi
1056 }

```

\addBodeZPKPlots This macro is designed to issue multiple **\addplot** macros for the same set of poles, zeros, gain, and delay. All of the work is done by the **\bp@ZPK@plot** macro.

```

1057 \NewDocumentCommand{\addBodeZPKPlots}{ 0{} m m }{%
1058   \edef\bp@spec@arg{#3}%
1059   \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{
1060     \pgfkeys{/bodeplot/add/.cd, reset}
1061     \pgfkeys{/bodeplot/add/.cd, #1}
1062     \bp@zpk@new@to@legacy{\bp@spec@arg}
1063     \expandafter\bp@fix@add@opt\expandafter{\bp@add@0}
1064   }{
1065     \bp@fix@add@opt{#1}
1066     \edef\bp@legacy{\bp@spec@arg}
1067   }
1068   \foreach \approx/\opt in \bp@add@0 {
1069     \ifx\approx\@empty\else
1070       \gdef\bp@plot{}
1071       \gdef\bp@tmp{}
1072       \ifnum\pdf@strcmp{#2}{phase}=0
1073         \bp@ZPK@plot{\bp@tmp}{\bp@plot}{\approx}{\bp@legacy}
1074       \else
1075         \bp@ZPK@plot{\bp@plot}{\bp@tmp}{\approx}{\bp@legacy}
1076       \fi
1077       \if@pgfarg
1078         \edef\bp@cmd{\noexpand\addplot [bp@freq@filter,
1079           domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
1080           variable=t, thick, trig format plots=rad, \unexpanded\expandafter{\opt}]}
1081         \bp@cmd {\bp@plot};
1082       \else
1083         \stepcounter{bp@gnuplot@id}
1084         \edef\gnu@id{\arabic{bp@gnuplot@id}}
1085         \bp@gnu@plot[\bp@freq@scale]{\bp@plot}{\gnu@id}
1086         \edef\bp@cmd{\noexpand\addplot [variable=t, thick,
1087           \unexpanded\expandafter{\opt}]}
1088         \expandafter\bp@cmd\bp@gnu@cmd
1089       \fi
1090     \fi
1091   }
1092 }

```

\addBodeTFPlot This macro is designed to issues a single **\addplot** macros for the set of coefficients and delay. All of the work is done by the **\bp@TF@plot** macro.

```

1093 \NewDocumentCommand{\addBodeTFPlot}{ 0{} m m }{%
1094   \edef\bp@spec@arg{#3}%
1095   \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{
1096     \bp@tf@new@to@legacy{\bp@spec@arg}
1097   }{
1098     \edef\bp@legacy{\bp@spec@arg}
1099   }
1100   \gdef\bp@plot{}
1101   \gdef\bp@tmp{}
1102   \ifnum\pdf@strcmp{#2}{phase}=0
1103     \bp@TF@plot{\bp@tmp}{\bp@plot}{\bp@legacy}
1104   \else
1105     \bp@TF@plot{\bp@plot}{\bp@tmp}{\bp@legacy}
1106   \fi
1107   \if@pgfarg
1108     \ifnum\pdf@strcmp{#2}{phase}=0
1109       \edef\bp@cmd{\noexpand\addplot [bp@freq@filter,
1110         domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
1111         variable=t, thick, trig format plots=rad,
1112         \unexpanded\expandafter{#1}]}
1113       \bp@cmd {\n@mod{\bp@plot}{2*pi}};
1114     \else
1115       \edef\bp@cmd{\noexpand\addplot [bp@freq@filter,
1116         domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,

```

```

1117     variable=t, thick, \unexpanded\expandafter{#1}}
1118     \bp@cmd {\bp@plot};
1119   \fi
1120 \else
1121   \stepcounter{bp@gnuplot@id}
1122   \edef\gnu@id{\arabic{bp@gnuplot@id}}
1123   \edef\bp@cmd{\noexpand\addplot [variable=t, thick,
1124   \unexpanded\expandafter{#1}}
1125   \ifnum\pdf@strcmp{#2}{phase}=0
1126     \bp@gnu@unwrap@plot[\bp@freq@scale]{\bp@plot}{\gnu@id}
1127     \expandafter\bp@cmd\bp@gnu@cmd
1128   \else
1129     \bp@gnu@plot[\bp@freq@scale]{\bp@plot}{\gnu@id}
1130     \expandafter\bp@cmd\bp@gnu@cmd
1131   \fi
1132 \fi
1133 }

```

\addBodeTFPlots Similar to **\addBodeZPKPlots** but for systems specified in the TF format. This macro converts the TF specification to ZPK format internally using Python via `--shell-escape` and then calls **\addBodeZPKPlots** to generate the plots. If Python with the **scipy** and **numpy** libraries is not available on the shell, the package returns an error message.

```

1134 \NewDocumentCommand{\addBodeTFPlots}{0}{m m}{%
1135   \bp@tf@to@zpk{#3}{\bp@tmp@zpk}{\bp@tmp@status}
1136   \ifnum\pdf@strcmp{\bp@tmp@status}{true}=0
1137     \addBodeZPKPlots[1]{#2}{\bp@tmp@zpk}
1138   \else
1139     \PackageError{bodeplot}{The |\addBodeTFPlots| macro enables linear and asymptotic app
1140     shell-escape. If Python is not available, use |\addBodeTFPlot| to generate plots without
1141   \fi
1142 }

```

\addBodeComponentPlot This macro is designed to create a single **\addplot** macro capable of plotting linear combinations of the basic components described in Section 3.1.1. The only work to do here is to handle the **pgf** package option.

```

1142 \newcommand{\addBodeComponentPlot}[2]{}{
1143   \if@pgfarg
1144     \edef\bp@cmd{\noexpand\addplot [bp@freq@filter,
1145     domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
1146     variable=t, thick, trig format plots=rad, #1]}
1147     \bp@cmd {#2};
1148   \else
1149     \stepcounter{bp@gnuplot@id}
1150     \edef\gnu@id{\arabic{bp@gnuplot@id}}
1151     \edef\bp@cmd{\noexpand\addplot [variable=t, thick, #1]}
1152     \bp@gnu@plot[\bp@freq@scale]{#2}{\gnu@id}
1153     \expandafter\bp@cmd\bp@gnu@cmd
1154   \fi
1155 }

```

BodePhPlot (*env.*) An environment to host phase plot macros that pass parametric functions to **\addplot** macros. Uses the defaults specified in **bp@style** to create a shortcut that includes the **tikzpicture** and **semilogaxis** environments. The body of the environment is grabbed as a macro to maintain compatibility with externalization in **tikz**.

```

1156 \AtBeginDocument{%
1157   \if@babel
1158     \AddToHook{env/BodePhPlot/begin}{%
1159       \expandafter\shorthandoff\expandafter{\bp@short@list}
1160     }
1161     \AddToHook{env/BodePhPlot/end}{%
1162       \expandafter\shorthandon\expandafter{\bp@short@list}

```

```

1163     }
1164 \fi
1165 }
1166 \NewDocumentEnvironment{BodePhPlot}{ 0{} G{} G{} +b }{
1167   \pgfutil@ifempty{#2}{%
1168     \pgfkeys{/bodeplot/env/.cd, reset}
1169     \pgfkeys{/bodeplot/env/.cd, #1}
1170     \pgfkeysgetvalue{/bodeplot/env/@tikz}{\bp@tikz}
1171     \pgfkeysgetvalue{/bodeplot/env/@prefix}{\bp@user@prefix}
1172   }{%
1173     \bp@parse@env@opt{#1}
1174     \edef\bp@domain@start{#2}
1175     \edef\bp@domain@end{#3}
1176   }
1177   \edef\bp@cmd{\noexpand\begin{tikzpicture}
1178   [\unexpanded\expandafter{\bp@tikz}]}
1179   \bp@cmd
1180   \edef\bp@cmd{\noexpand\begin{semilogxaxis}[
1181     bp@ph@y@label,
1182     bp@freq@label,
1183     bp@style,
1184     xmin={\bp@domain@start},
1185     xmax={\bp@domain@end},
1186     domain=\bp@domain@start:\bp@domain@end,
1187     height=2.5cm,
1188     \unexpanded\expandafter{\bp@axes}
1189   ]}
1190   \bp@cmd
1191   #4
1192   \end{semilogxaxis}
1193   \end{tikzpicture}
1194 }{}

```

BodeMagPlot (*env.*) An environment to host magnitude plot macros that pass parametric functions to `\addplot` macros. Uses the defaults specified in `bp@style` to create a shortcut that includes the `tikzpicture` and `semilogaxis` environments.

```

1195 \AtBeginDocument{%
1196   \if@babel
1197     \AddToHook{env/BodeMagPlot/begin}{%
1198       \expandafter\shorthandoff\expandafter{\bp@short@list}
1199     }
1200     \AddToHook{env/BodeMagPlot/end}{%
1201       \expandafter\shorthandon\expandafter{\bp@short@list}
1202     }
1203   \fi
1204 }
1205 \NewDocumentEnvironment{BodeMagPlot}{ 0{} G{} G{} +b }{
1206   \pgfutil@ifempty{#2}{%
1207     \pgfkeys{/bodeplot/env/.cd, reset}
1208     \pgfkeys{/bodeplot/env/.cd, #1}
1209     \pgfkeysgetvalue{/bodeplot/env/@tikz}{\bp@tikz}
1210     \pgfkeysgetvalue{/bodeplot/env/@prefix}{\bp@user@prefix}
1211   }{%
1212     \bp@parse@env@opt{#1}
1213     \edef\bp@domain@start{#2}
1214     \edef\bp@domain@end{#3}
1215   }
1216   \edef\bp@cmd{\noexpand\begin{tikzpicture}
1217   [\unexpanded\expandafter{\bp@tikz}]}
1218   \bp@cmd
1219   \edef\bp@cmd{\noexpand\begin{semilogxaxis}[
1220     bp@style,
1221     bp@freq@label,

```

```

1222     xmin={\bp@domain@start},
1223     xmax={\bp@domain@end},
1224     domain=\bp@domain@start:\bp@domain@end,
1225     height=2.5cm,
1226     ylabel={Gain (dB)},
1227     \unexpanded\expandafter{\bp@axes}
1228   ]}
1229   \bp@cmd
1230   #4
1231   \end{semilogxaxis}
1232 \end{tikzpicture}
1233 }{}

```

\addBodePlot Unified macro to add Bode plots for both ZPK and TF system representations inside a BodePlot environment. Supports both pgfkeys interface introduced in v3.0 and the legacy interface.

```

1234 \NewDocumentCommand{\addBodePlot}{ 0{ } m G{ } }{%
1235   \gdef\bp@mag{}
1236   \gdef\bp@ph{}
1237   \pgfutil@ifempty{#3}{%
1238     \pgfkeys{/bodeplot/add/.cd, reset}
1239     \pgfkeys{/bodeplot/add/.cd, #1}
1240     \expandafter\bp@parse@add@Bode@opt\expandafter{\bp@add@0}
1241     \bp@contains@num{#2}{%
1242       \bp@tf@new@to@legacy{#2}
1243       \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}
1244       \edef\bp@mode{tf}
1245     }{
1246       \bp@zpk@new@to@legacy{#2}
1247       \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@approx}{\bp@legacy}
1248       \edef\bp@mode{zpk}
1249     }
1250   }{
1251     \bp@parse@add@Bode@opt{#1}
1252     \ifnum\pdf@strcmp{#2}{zpk}=0
1253       \edef\bp@mode{zpk}
1254       \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@approx}{#3}
1255     \else
1256       \ifnum\pdf@strcmp{#2}{tf}=0
1257         \edef\bp@mode{tf}
1258         \bp@TF@plot{\bp@mag}{\bp@ph}{#3}
1259       \else
1260         \PackageError {bodeplot} {Unknown system representation `#2'.}
1261         {Supported representations are `zpk' and `tf' .}
1262       \fi
1263     \fi
1264   }
1265   \if@pgfarg
1266     \xdef\bp@mag@cmd{\unexpanded\expandafter{\bp@mag@cmd}\noexpand\addplot
1267       [bp@freq@filter, domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain},
1268       variable=t, thick, trig format plots=rad,
1269       \unexpanded\expandafter{\bp@plot}] {\bp@mag};}
1270     \xdef\bp@ph@cmd{\unexpanded\expandafter{\bp@ph@cmd}\noexpand\addplot
1271       [bp@freq@filter, domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain},
1272       variable=t, thick, trig format plots=rad,
1273       \unexpanded\expandafter{\bp@plot}] {\bp@ph};}
1274   \else
1275     \stepcounter{bp@gnuplot@id}
1276     \edef\gnu@id{\arabic{bp@gnuplot@id}}
1277     \bp@gnu@plot{\bp@mag}{\gnu@id}
1278     \xdef\bp@mag@cmd{\unexpanded\expandafter{\bp@mag@cmd}\noexpand\addplot
1279       [variable=t, thick, \unexpanded\expandafter{\bp@plot}]\bp@gnu@cmd}
1280     \stepcounter{bp@gnuplot@id}

```

```

1281 \edef\gnu@id{\arabic{bp@gnuplot@id}}
1282 \ifnum\pdf@strcmp{\bp@mode}{zpk}=0
1283 \bp@gnu@plot{\bp@ph}{\gnu@id}
1284 \else
1285 \ifnum\pdf@strcmp{\bp@mode}{tf}=0
1286 \bp@gnu@unwrap@plot{\bp@ph}{\gnu@id}
1287 \fi
1288 \fi
1289 \xdef\bp@ph@cmd{\unexpanded\expandafter{\bp@ph@cmd}\noexpand\addplot
1290 [variable=t, thick, \unexpanded\expandafter{\bp@plot}]\bp@gnu@cmd}
1291 \fi
1292 }

```

BodePlot (*env.*) An environment that works with the unified `\addBodePlot` macro. Creates a grouped plot with magnitude on top and phase on bottom, automatically collecting and inserting plot commands generated by `\addBodePlot` calls within the environment body.

```

1293 \NewDocumentEnvironment{BodePlot}{0}{G}{G}{+b}{%
1294 \pgfutil@ifempty{#2}{%
1295 \pgfkeys{/bodeplot/combinedenv/.cd, reset}
1296 \pgfkeys{/bodeplot/combinedenv/.cd, #1}
1297 \pgfkeysgetvalue{/bodeplot/combinedenv/@group}{\bp@group}
1298 \pgfkeysgetvalue{/bodeplot/combinedenv/@approx}{\bp@approx}
1299 \pgfkeysgetvalue{/bodeplot/combinedenv/@commands/mag}{\bp@mag@commands}
1300 \pgfkeysgetvalue{/bodeplot/combinedenv/@commands/ph}{\bp@ph@commands}
1301 \pgfkeysgetvalue{/bodeplot/combinedenv/@tikz}{\bp@tikz}
1302 \pgfkeysgetvalue{/bodeplot/combinedenv/@prefix}{\bp@user@prefix}
1303 }{%
1304 \bp@parse@opt{#1}
1305 \edef\bp@domain@start{#2}
1306 \edef\bp@domain@end{#3}
1307 }
1308 \gdef\bp@mag@cmd{}
1309 \gdef\bp@ph@cmd{}
1310 \edef\bp@cmd{\noexpand\begin{tikzpicture}
1311 [\unexpanded\expandafter{\bp@tikz}]}
1312 \bp@cmd
1313 \edef\bp@cmd{\noexpand\begin{groupplot}[
1314 bp@style,
1315 xmin=\bp@domain@start,
1316 xmax=\bp@domain@end,
1317 domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
1318 height=2.5cm,
1319 xmode=log,
1320 group style = {group size = 1 by 2,vertical sep=0.25cm},
1321 \unexpanded\expandafter{\bp@group}
1322 ]}
1323 \bp@cmd
1324 #4
1325 \edef\temp@mag@cmd{\noexpand\nextgroupplot
1326 [ylabel={Gain (dB)}, xmajorticks=false,
1327 \unexpanded\expandafter{\bp@mag@axes}]}
1328 \edef\temp@ph@cmd{\noexpand\nextgroupplot
1329 [bp@ph@y@label, bp@freq@label,
1330 \unexpanded\expandafter{\bp@ph@axes}]}
1331 \temp@mag@cmd
1332 \bp@mag@cmd
1333 \temp@ph@cmd
1334 \bp@ph@cmd
1335 \end{groupplot}
1336 \end{tikzpicture}
1337 }{}

```


4.5.2 Internal macros

`\bp@parse@complex` Parses complex numbers in the format `a+bi` where `a` and `b` can be arbitrary pgf-math/gnuplot expressions. The scans to the left starting from the trailing `i` and finds the first top-level `+` or `-`, i.e., the first such operator not inside braces or parentheses. Everything between the `i` and the `+` or `-` is treated as the imaginary part, and everything else is treated as the real part. Sets `\bp@realpart`, `\bp@imagpart`, and `\bp@complex`.

```

1338 \ExplSyntaxOn
1339 \tl_new:N \l_real_tl
1340 \tl_new:N \l_imag_tl
1341 \tl_new:N \bp@complex
1342 \tl_new:N \bp@realpart
1343 \tl_new:N \bp@imagpart
1344
1345 % Variables for the complex number parser
1346 \tl_new:N \l__bp_input_tl
1347 \tl_new:N \l__bp_before_tl
1348 \tl_new:N \l__bp_after_tl
1349 \tl_new:N \l__bp_sep_tl
1350 \int_new:N \l__bp_pos_int
1351 \int_new:N \l__bp_seppos_int
1352 \int_new:N \l__bp_paren_int
1353 \bool_new:N \l__bp_found_bool
1354
1355 % Generate V variant for tl_if_single_token
1356 \prg_generate_conditional_variant:Nnn \tl_if_single_token:n { V } { TF, T, F }
1357
1358 \NewDocumentCommand{\bp@parse@complex}{m}{
1359   \tl_set:Nx \l__bp_input_tl {#1}
1360   \tl_remove_all:Nn \l__bp_input_tl { ~ }
1361
1362   % Check if it ends with 'i'
1363   \regex_match:nVTF { i$ } \l__bp_input_tl
1364   { \__bp_parse_complex_with_i: }
1365   { \__bp_parse_real_only: }
1366
1367   \tl_gset_eq:NN \bp@realpart \l_real_tl
1368   \tl_gset_eq:NN \bp@imagpart \l_imag_tl
1369   \tl_gset:Nx \bp@complex { { { \l_real_tl } , { \l_imag_tl } } }
1370 }
1371
1372 % Parse a pure real number (no trailing i)
1373 \cs_new_protected:Npn \__bp_parse_real_only:
1374 {
1375   \__bp_unwrap_braces:N \l__bp_input_tl
1376   \tl_set_eq:NN \l_real_tl \l__bp_input_tl
1377   \tl_set:Nn \l_imag_tl {0}
1378 }
1379
1380 % Parse a complex number (has trailing i)
1381 \cs_new_protected:Npn \__bp_parse_complex_with_i:
1382 {
1383   % Remove trailing 'i'
1384   \regex_replace_once:nnN { i$ } {} \l__bp_input_tl
1385
1386   % Find the last + or - at depth 0 (but not at position 0)
1387   \__bp_find_separator:
1388
1389   \bool_if:NTF \l__bp_found_bool
1390   { \__bp_split_at_separator: }
1391   { \__bp_parse_pure_imaginary: }
1392 }

```

```

1393
1394 % Parse pure imaginary (no separator found)
1395 \cs_new_protected:Npn \__bp_parse_pure_imaginary:
1396 {
1397     \tl_set:Nn \l_real_tl {0}
1398     \__bp_set_signed_value:NN \l_imag_tl \l__bp_input_tl
1399 }
1400
1401 % Set target to value with sign handling
1402 \cs_new_protected:Npn \__bp_set_signed_value:NN #1#2
1403 {
1404     \tl_if_empty:NTF #2
1405     { \tl_set:Nn #1 {1} }
1406     {
1407         \tl_set:Nx \l_tmpa_tl { \tl_head:N #2 }
1408         \str_if_eq:VnTF \l_tmpa_tl {-}
1409         {
1410             \tl_set:Nx \l_tmpb_tl { \tl_tail:N #2 }
1411             \tl_if_empty:NTF \l_tmpb_tl
1412             { \tl_set:Nn #1 {-1} }
1413             {
1414                 \__bp_unwrap_braces:N \l_tmpb_tl
1415                 \tl_set:Nx #1 { -( \l_tmpb_tl ) }
1416             }
1417         }
1418         {
1419             \str_if_eq:VnTF \l_tmpa_tl {+}
1420             {
1421                 \tl_set:Nx \l_tmpb_tl { \tl_tail:N #2 }
1422                 \tl_if_empty:NTF \l_tmpb_tl
1423                 { \tl_set:Nn #1 {1} }
1424                 {
1425                     \__bp_unwrap_braces:N \l_tmpb_tl
1426                     \tl_set_eq:NN #1 \l_tmpb_tl
1427                 }
1428             }
1429             {
1430                 \__bp_unwrap_braces:N #2
1431                 \tl_set_eq:NN #1 #2
1432             }
1433         }
1434     }
1435 }
1436
1437 % Helper to unwrap a single braced group
1438 \cs_new_protected:Npn \__bp_unwrap_braces:N #1
1439 {
1440     \tl_if_single_token:VF #1
1441     {
1442         \tl_set:Nx \l_tmpc_tl { \tl_count:N #1 }
1443         \int_compare:nNnT { \l_tmpc_tl } = {1}
1444         { \tl_set:Nx #1 { \exp_after:wN \use:n #1 } }
1445     }
1446 }
1447
1448 % Process one token/group to find separator
1449 \cs_new_protected:Npn \__bp_process_item:n #1
1450 {
1451     \tl_if_single_token:nTF {#1}
1452     {
1453         % Track parenthesis depth
1454         \token_if_eq_charcode:NNT #1 ( { \int_incr:N \l__bp_paren_int }
1455         \token_if_eq_charcode:NNT #1 ) { \int_decr:N \l__bp_paren_int }

```

```

1456      % Only consider separators at position > 0 and paren depth 0
1457      \int_compare:nNnT { \l__bp_pos_int } > {0}
1458      {
1459          \int_compare:nNnT { \l__bp_paren_int } = {0}
1460          {
1461              \bool_lazy_or:nnT
1462              { \token_if_eq_charcode_p:NN #1 + }
1463              { \token_if_eq_charcode_p:NN #1 - }
1464              {
1465                  \bool_set_true:N \l__bp_found_bool
1466                  \int_set_eq:NN \l__bp_seppos_int \l__bp_pos_int
1467                  \tl_set:Nn \l__bp_sep_tl {#1}
1468              }
1469          }
1470      }
1471  }
1472  { }
1473  \int_incr:N \l__bp_pos_int
1474 }
1475
1476 % Find the last + or - at depth 0
1477 \cs_new_protected:Npn \__bp_find_separator:
1478 {
1479     \bool_set_false:N \l__bp_found_bool
1480     \int_zero:N \l__bp_pos_int
1481     \int_zero:N \l__bp_paren_int
1482     \tl_map_function:NN \l__bp_input_tl \__bp_process_item:n
1483 }
1484
1485 % Collect items for splitting
1486 \cs_new_protected:Npn \__bp_collect_item:n #1
1487 {
1488     \int_compare:nNnTF { \l__bp_pos_int } < { \l__bp_seppos_int }
1489     { \tl_put_right:Nn \l__bp_before_tl {#1} }
1490     {
1491         \int_compare:nNnF { \l__bp_pos_int } = { \l__bp_seppos_int }
1492         { \tl_put_right:Nn \l__bp_after_tl {#1} }
1493     }
1494     \int_incr:N \l__bp_pos_int
1495 }
1496
1497 % Split the input at the separator position
1498 \cs_new_protected:Npn \__bp_split_at_separator:
1499 {
1500     \tl_clear:N \l__bp_before_tl
1501     \tl_clear:N \l__bp_after_tl
1502     \int_zero:N \l__bp_pos_int
1503
1504     \tl_map_function:NN \l__bp_input_tl \__bp_collect_item:n
1505
1506     \__bp_unwrap_braces:N \l__bp_before_tl
1507     \__bp_unwrap_braces:N \l__bp_after_tl
1508
1509     \tl_if_empty:NTF \l__bp_before_tl
1510     { \tl_set:Nn \l_real_tl {0} }
1511     { \tl_set_eq:NN \l_real_tl \l__bp_before_tl }
1512
1513     \tl_if_empty:NTF \l__bp_after_tl
1514     {
1515         \str_if_eq:VnTF \l__bp_sep_tl {-}
1516         { \tl_set:Nn \l_imag_tl {-1} }
1517         { \tl_set:Nn \l_imag_tl {1} }
1518     }

```

```

1519     {
1520         \str_if_eq:VnTF \l__bp_sep_tl {-}
1521         { \tl_set:Nx \l_imag_tl { -( \l__bp_after_tl ) } }
1522         { \tl_set_eq:NN \l_imag_tl \l__bp_after_tl }
1523     }
1524 }

```

`\bp_if_contains:nnTF` Helper function to check if a token list contains a substring.

```

1525 \tl_new:N \l__bode_check_tl
1526 \cs_new_protected:Npn \bp_if_contains:nnTF #1#2#3#4 {
1527     \tl_set:Nx \l__bode_check_tl {#1}
1528     \tl_if_in:VnTF \l__bode_check_tl {#2} {#3} {#4}
1529 }

```

`\bp@contains@equal` Checks if argument contains '=' to distinguish new from legacy interface.

```

1530 \NewDocumentCommand{\bp@contains@equal}{m m m}{%
1531     \bp_if_contains:nnTF {#1} {=} {#2} {#3}%
1532 }

```

`\bp@contains@num` Checks if argument contains 'numerator' keyword.

```

1533 \NewDocumentCommand{\bp@contains@num}{m m m}{%
1534     \bp_if_contains:nnTF {#1} {numerator} {#2} {#3}%
1535 }
1536 \ExplSyntaxOff

```

`\bp@fix@add@opt` Processes options for addplot commands, organizing by approximation type.

```

1537 \NewDocumentCommand{\bp@fix@add@opt}{ m }{%
1538     \gdef\bp@add@0{}
1539     \gdef\bp@add@tmp{}
1540     \foreach \approx/\opt in {#1} {
1541         \ifx\approx\@empty\else
1542             \ifnum\pdf@strcmp{\approx}{linear}=0
1543                 \xdef\bp@add@0{\unexpanded\expandafter{\bp@add@0}%
1544                     linear/{\unexpanded\expandafter{\opt}}},}
1545             \else
1546                 \ifnum\pdf@strcmp{\approx}{asymptotic}=0
1547                     \xdef\bp@add@0{\unexpanded\expandafter{\bp@add@0}%
1548                         asymptotic/{\unexpanded\expandafter{\opt}}},}
1549                 \else
1550                     \ifnum\pdf@strcmp{\approx}{true}=0
1551                         \xdef\bp@add@tmp{\unexpanded\expandafter{\bp@add@tmp}%
1552                             \unexpanded\expandafter{\opt}},}
1553                     \else
1554                         \xdef\bp@add@tmp{\unexpanded\expandafter{\bp@add@tmp}%
1555                             \unexpanded\expandafter{\approx}},}
1556                     \fi
1557                 \fi
1558             \fi
1559         \fi
1560     }
1561     \ifx\bp@add@tmp\@empty\else
1562         \xdef\bp@add@0{\unexpanded\expandafter{\bp@add@0}%
1563             true/{\unexpanded\expandafter{\bp@add@tmp}}}
1564     \fi
1565 }

```

`\bp@add` Helper macro to build up magnitude and phase expressions by adding contributions from zeros, poles, gain, and delay.

```

1566 \newcommand*{\bp@add}[3]{
1567     \ifcat$\detokenize\expandafter{#1}$
1568         \xdef#1{\unexpanded\expandafter{#1 0+#2}}
1569     \else
1570         \xdef#1{\unexpanded\expandafter{#1+#2}}

```

```

1571 \fi
1572 \foreach \y [count=\n] in #3 {
1573   \xdef#1{\unexpanded\expandafter{#1}{\y}}
1574   \xdef\Last@LoopValue{\n}
1575 }
1576 \ifnum\Last@LoopValue=1
1577   \xdef#1{\unexpanded\expandafter{#1}{0}}
1578 \fi
1579 }

```

`\bp@ZPK@plot` Builds magnitude and phase plot expressions from zero-pole-gain representation. Handles linear, asymptotic, and true Bode plot approximations.

```

1580 \newcommand{\bp@ZPK@plot}[4]{
1581   \edef\bp@list{#4}
1582   \foreach \feature/\values in \bp@list {
1583     \ifx\values\empty\else
1584       \ifnum\pdf@strcmp{\feature}{z}=0
1585         \foreach \z in \values {
1586           \ifx\z\empty\else
1587             \ifnum\pdf@strcmp{#3}{linear}=0
1588               \bp@add{#2}{\PhZeroLin}{\z}
1589               \bp@add{#1}{\MagZeroLin}{\z}
1590             \else
1591               \ifnum\pdf@strcmp{#3}{asymptotic}=0
1592                 \bp@add{#2}{\PhZeroAsymp}{\z}
1593                 \bp@add{#1}{\MagZeroAsymp}{\z}
1594               \else
1595                 \bp@add{#2}{\PhZero}{\z}
1596                 \bp@add{#1}{\MagZero}{\z}
1597               \fi
1598             \fi
1599           \fi
1600         }
1601       \fi
1602       \ifnum\pdf@strcmp{\feature}{p}=0
1603         \foreach \p in \values {
1604           \ifx\p\empty\else
1605             \ifnum\pdf@strcmp{#3}{linear}=0
1606               \bp@add{#2}{\PhPoleLin}{\p}
1607               \bp@add{#1}{\MagPoleLin}{\p}
1608             \else
1609               \ifnum\pdf@strcmp{#3}{asymptotic}=0
1610                 \bp@add{#2}{\PhPoleAsymp}{\p}
1611                 \bp@add{#1}{\MagPoleAsymp}{\p}
1612               \else
1613                 \bp@add{#2}{\PhPole}{\p}
1614                 \bp@add{#1}{\MagPole}{\p}
1615               \fi
1616             \fi
1617           \fi
1618         }
1619       \fi
1620       \ifnum\pdf@strcmp{\feature}{k}=0
1621         \ifnum\pdf@strcmp{#3}{linear}=0
1622           \bp@add{#2}{\PhKLin}{\values}
1623           \bp@add{#1}{\MagKLin}{\values}
1624         \else
1625           \ifnum\pdf@strcmp{#3}{asymptotic}=0
1626             \bp@add{#2}{\PhKAsymp}{\values}
1627             \bp@add{#1}{\MagKAsymp}{\values}
1628           \else
1629             \bp@add{#2}{\PhK}{\values}
1630             \bp@add{#1}{\MagK}{\values}

```

```

1631         \fi
1632     \fi
1633 \fi
1634 \ifnum\pdf@strcmp{\feature}{d}=0
1635     \ifnum\pdf@strcmp{#3}{linear}=0
1636         \PackageError {bodeplot}
1637         {Linear approximation for pure delays is not supported.}
1638         {Plot the true Bode plot using `true' instead of `linear'.}
1639     \else
1640         \ifnum\pdf@strcmp{#3}{asymptotic}=0
1641             \PackageError {bodeplot}
1642             {Asymptotic approximation for pure delays is not supported.}
1643             {Plot the true Bode plot using `true' instead of `asymptotic'.}
1644         \else
1645             \ifdim\values pt < 0pt
1646                 \PackageError {bodeplot} {Delay needs to be a positive number.}
1647             \fi
1648             \bp@add{#2}{\PhDel}{\values}
1649             \bp@add{#1}{\MagDel}{\values}
1650         \fi
1651     \fi
1652 \fi
1653 \fi
1654 }
1655 }

```

\bp@gnu@plot Generates gnuplot commands for computing Bode plot data.

```

1656 \newcommand{\bp@gnu@plot}[3][1]{
1657   \xdef\bp@gnu@cmd{ gnuplot [raw gnuplot, id=#3, prefix=\bp@prefix]
1658     { set table $meta;
1659       set dummy t;
1660       set logscale x 10;
1661       set xrange [#1*\pgfkeysvalueof{/pgfplots/domain}*#1];
1662       set samples \pgfkeysvalueof{/pgfplots/samples};
1663       plot #2;
1664       set table "\bp@prefix#3.table";
1665       plot "$meta" using ($1/(\bp@freq@scale)):(#2);
1666     };}
1667 }

```

\bp@zpk@new@to@legacy Converts the pgfkeys interface format to the legacy ZPK format. Parses complex zeros and poles using **\bp@parse@complex**.

```

1668 \NewDocumentCommand{\bp@zpk@new@to@legacy}{ m }{%
1669   % Pre-expand #1 so pgfkeys sees character tokens (not a macro token).
1670   % This allows passing a macro whose expansion is a key=value list.
1671   \edef\bp@zpknl@spec{/bodeplot/zpk/.cd, reset, #1}%
1672   \expandafter\pgfkeys\expandafter{\bp@zpknl@spec}
1673   \pgfkeysgetvalue{/bodeplot/zpk/@zeros}{\bp@z}
1674   \edef\bp@z{\bp@z}%
1675   \gdef\bp@z@list{}
1676   \ifx\bp@z\empty\else
1677     \foreach \z in \bp@z {
1678       \bp@parse@complex{\z}
1679       \xdef\bp@z@list{\unexpanded\expandafter{\bp@z@list} \bp@complex,}
1680     }
1681   \fi
1682   \pgfkeysgetvalue{/bodeplot/zpk/@poles}{\bp@p}
1683   \edef\bp@p{\bp@p}%
1684   \gdef\bp@p@list{}
1685   \ifx\bp@p\empty\else
1686     \foreach \p in \bp@p {
1687       \bp@parse@complex{\p}
1688       \xdef\bp@p@list{\unexpanded\expandafter{\bp@p@list} \bp@complex,}

```

```

1689   }
1690   \fi
1691   \pgfkeysgetvalue{/bodeplot/zpk/@gain}{\bp@k}
1692   \pgfkeysgetvalue{/bodeplot/zpk/@delay}{\bp@d}
1693   \xdef\bp@legacy{z/{\bp@z@list},p/{\bp@p@list},k/{\bp@k},d/{\bp@d}
1694 }

```

`\bp@tf@new@to@legacy` Converts the pgfkeys interface format to the legacy TF format.

```

1695 \NewDocumentCommand{\bp@tf@new@to@legacy}{ m }{%
1696   % Pre-expand #1 so pgfkeys sees character tokens (not a macro token).
1697   \edef\bp@tfnl@spec{/bodeplot/tf/.cd, reset, #1}%
1698   \expandafter\pgfkeys\expandafter{\bp@tfnl@spec}
1699   \pgfkeysgetvalue{/bodeplot/tf/@numerator}{\bp@num}
1700   \pgfkeysgetvalue{/bodeplot/tf/@denominator}{\bp@den}
1701   \pgfkeysgetvalue{/bodeplot/tf/@delay}{\bp@d}
1702   \gdef\bp@num@list{}
1703   \ifx\bp@num\empty\else
1704     \foreach \n in \bp@num {
1705       \xdef\bp@num@list{\unexpanded\expandafter{\bp@num@list} \n,}
1706     }
1707   \fi
1708   \gdef\bp@den@list{}
1709   \ifx\bp@den\empty\else
1710     \foreach \d in \bp@den {
1711       \xdef\bp@den@list{\unexpanded\expandafter{\bp@den@list} \d,}
1712     }
1713   \fi
1714   \xdef\bp@legacy{num/{\bp@num@list},den/{\bp@den@list},d/{\bp@d}
1715 }

```

`\bp@tztzk@outfile` Converts a transfer-function specification to zero-pole-gain (ZPK) form by invoking a `\bp@tf@to@zpk` python3 helper script via `--shell-escape`.

At the start of the document the package registers an `\AtBeginDocument` hook that writes a six-line Python driver to `\jobname-tf2zpk.py`. The script is intentionally written as a set of zero-indented one-liners so that TeX's space-collapsing rules inside `\write` arguments do not corrupt Python indentation. Characters that would require a backslash in TeX (`\`, `{`, `}`) and the newline character are obtained at runtime using `chr(92)`, `chr(123)`, `chr(125)`, and `chr(10)`, so that no `\detokenize` wrapper is needed for any line.

`\bp@tf@to@zpk` accepts a TF specification in either the new pgfkeys format (e.g. `numerator={2,10},denominator={1,3,2},delay=0`) or the legacy slash format (e.g. `num/{2,10},den/{1,3,2},d/0`), mirroring the two interfaces accepted by `\BodeTF`. The numerator and denominator coefficient lists are extracted and passed to the Python helper as comma-separated strings.

On success the macro `\edefs` into the caller-supplied output control sequence a ZPK specification string in the new pgfkeys format, e.g. `zeros={-5.0},poles={-2.0,-1.0},gain=2.000000,delay=`. The delay value is taken from the TF spec and passed through unchanged; if no delay was specified it is the empty string, which the ZPK pgfkeys family treats as zero. The result can be passed to `\BodeZPK` via `\expandafter`:

```

\expandafter\BodeZPK\expandafter[\langle opts \rangle]\expandafter{\langle output-
cs \rangle}

```

and directly to `\bp@zpk@new@to@legacy` or `\bp@ZPK@plot` via the same `\expandafter` idiom for internal use.

Usage:

```

\bp@tf@to@zpk{\langle tf-spec \rangle}{\langle zpk-cs \rangle}{\langle status \rangle}

```

tf-spec Transfer-function specification in either the pgfkeys format accepted by `\BodeTF` (new interface) or the legacy slash format (legacy interface).

\zpk-cs Control sequence **\edef**-bound on success to a ZPK spec string in the format accepted by **\BodeZPK**: **zeros={...},poles={...},gain=...,delay=...**

\status Control sequence **\def**-bound to **true** on success, or **false** if the Python script did not produce an output file (python3 unavailable, missing dependencies, or no **--shell-escape**).

```

1716 \newcommand*{\bp@tztpk@outfile}{\jobname-tf2zpk-out.tex}
1717 \AtBeginDocument{%
1718   \newwrite\bp@tztpk@file
1719   \immediate\openout\bp@tztpk@file=\jobname-tf2zpk.py
1720   \immediate\write\bp@tztpk@file{import sys,numpy as np}
1721   \immediate\write\bp@tztpk@file{from scipy.signal import tf2zpk}
1722   \immediate\write\bp@tztpk@file{bs,lb,rb=chr(92),chr(123),chr(125)}
1723   \immediate\write\bp@tztpk@file{fmt=lambda arr:",".join([str(round(c.real,6)) if abs(round(
5   else str(round(c.real,6))+("+"+str(round(c.imag,6)) if round(c.imag,6)>=0 else str(round(
1724   \immediate\write\bp@tztpk@file{try: num=[float(x) for x in sys.argv[1].split(",")]; den=
1725   \immediate\write\bp@tztpk@file{except Exception as e: sys.stderr.write(str(e))}
1726   \immediate\closeout\bp@tztpk@file
1727 }
1728 \NewDocumentCommand{\bp@tf@to@zpk}{ m m m }{%
1729   % Step 1: Extract numerator, denominator, and delay from the TF spec.
1730   % Supports both the new pgfkeys format (numerator={...},denominator={...},
1731   % delay=...) and the legacy slash format (num/{...},den/{...},d/...).
1732   \bp@contains@equal{#1}{%
1733     % New pgfkeys format
1734     \pgfkeys{/bodeplot/tf/.cd, reset}%
1735     \pgfkeys{/bodeplot/tf/.cd, #1}%
1736     \pgfkeysgetvalue{/bodeplot/tf/@numerator}{\bp@tztpk@num}%
1737     \pgfkeysgetvalue{/bodeplot/tf/@denominator}{\bp@tztpk@den}%
1738     \pgfkeysgetvalue{/bodeplot/tf/@delay}{\bp@tztpk@d}%
1739   }{%
1740     % Legacy slash format
1741     \gdef\bp@tztpk@num{}\gdef\bp@tztpk@den{}\gdef\bp@tztpk@d{}%
1742     \edef\bp@tztpk@list{#1}%
1743     \foreach \bp@tztpk@feat/\bp@tztpk@val in \bp@tztpk@list {%
1744       \ifnum\pdf@strcmp{\bp@tztpk@feat}{num}=0
1745         \xdef\bp@tztpk@num{\bp@tztpk@val}%
1746       \fi
1747       \ifnum\pdf@strcmp{\bp@tztpk@feat}{den}=0
1748         \xdef\bp@tztpk@den{\bp@tztpk@val}%
1749       \fi
1750       \ifnum\pdf@strcmp{\bp@tztpk@feat}{d}=0
1751         \xdef\bp@tztpk@d{\bp@tztpk@val}%
1752       \fi
1753     }%
1754   }%
1755   % Step 2: Invoke the Python helper.
1756   \ifwindows
1757     \immediate\write18{if exist "\bp@tztpk@outfile" del /f "\bp@tztpk@outfile"}%
1758   \else
1759     \immediate\write18{rm -f "\bp@tztpk@outfile"}%
1760   \fi
1761   \immediate\write18{python3 "\jobname-tf2zpk.py" "\bp@tztpk@num" "\bp@tztpk@den" "\bp@tz
1762   % Step 3: Build the ZPK spec and set status.
1763   \IfFileExists{\bp@tztpk@outfile}{%
1764     \input{\bp@tztpk@outfile}%
1765     \edef#2{zeros={\zpkZeros},poles={\zpkPoles},gain=\zpkGain,delay=\bp@tztpk@d}%
1766     \def#3{true}%
1767   }{%
1768     \def#3{false}%
1769   }%
1770 }

```


\bp@TF@plot Builds magnitude and phase plot expressions from transfer function representation.

```

1771 \newcommand{\bp@TF@plot}[3]{
1772   \gdef\bp@num@re{0}
1773   \gdef\bp@num@im{0}
1774   \gdef\bp@den@re{0}
1775   \gdef\bp@den@im{0}
1776   \gdef\bp@loop@delay{0}
1777   \edef\bp@list{#3}
1778   \foreach \feature/\values in \bp@list {
1779     \ifx\values\empty\else
1780       \ifnum\pdf@strcmp{\feature}{num}=0
1781         \foreach \numcoeff [count=\numpow] in \values {
1782           \ifx\numcoeff\empty\else
1783             \xdef\bp@num@deg{\numpow}
1784             \fi
1785           }
1786         \foreach \numcoeff [count=\numpow] in \values {
1787           \ifx\numcoeff\empty\else
1788             \pgfmathtruncatemacro{\currentdegree}{\bp@num@deg-\numpow}
1789             \ifnum\currentdegree = 0
1790               \xdef\bp@num@re{\bp@num@re+\numcoeff}
1791             \else
1792               \ifodd\currentdegree
1793                 \xdef\bp@num@im{\bp@num@im+(\numcoeff*(\n@pow{-
1794 1}}{(\currentdegree-1)/2}))*%
1795                 (\n@pow{t}{\currentdegree})))}
1796               \else
1797                 \xdef\bp@num@re{\bp@num@re+(\numcoeff*(\n@pow{-
1798 1}}{(\currentdegree)/2}))*%
1799                 (\n@pow{t}{\currentdegree})))}
1800               \fi
1801             \fi
1802           }
1803       \ifnum\pdf@strcmp{\feature}{den}=0
1804         \foreach \dencoeff [count=\denpow] in \values {
1805           \ifx\dencoeff\empty\else
1806             \xdef\bp@den@deg{\denpow}
1807             \fi
1808           }
1809         \foreach \dencoeff [count=\denpow] in \values {
1810           \ifx\dencoeff\empty\else
1811             \pgfmathtruncatemacro{\currentdegree}{\bp@den@deg-\denpow}
1812             \ifnum\currentdegree = 0
1813               \xdef\bp@den@re{\bp@den@re+\dencoeff}
1814             \else
1815               \ifodd\currentdegree
1816                 \xdef\bp@den@im{\bp@den@im+(\dencoeff*(\n@pow{-
1817 1}}{(\currentdegree-1)/2}))*%
1818                 (\n@pow{t}{\currentdegree})))}
1819               \else
1820                 \xdef\bp@den@re{\bp@den@re+(\dencoeff*(\n@pow{-
1821 1}}{(\currentdegree)/2}))*%
1822                 (\n@pow{t}{\currentdegree})))}
1823               \fi
1824             \fi
1825           }
1826       \ifnum\pdf@strcmp{\feature}{d}=0
1827         \xdef\bp@loop@delay{\values}
1828       \fi

```

```

1829 \fi
1830 }
1831 \xdef#2{((atan2((\bp@num@im),(\bp@num@re))-atan2((\bp@den@im),%
1832 (\bp@den@re))-\bp@loop@delay*t)*(\bp@ph@scale))}
1833 \xdef#1{(20*log10(sqrt((\n@pow{\bp@num@re}{2})+(\n@pow{\bp@num@im}{2}))) -
%
1834 20*log10(sqrt((\n@pow{\bp@den@re}{2})+(\n@pow{\bp@den@im}{2}))))}
1835 }

```

`\bp@gnu@unwrap@plot` Generates gnuplot commands with phase unwrapping for TF plots.

```

1836 \newcommand{\bp@gnu@unwrap@plot}[3][1]{
1837 \xdef\bp@gnu@cmd{ gnuplot [raw gnuplot, id=#3, prefix=\bp@prefix]
1838 { set table $meta;
1839 set dummy t;
1840 set logscale x 10;
1841 set trange [#1*\pgfkeysvalueof{/pgfplots/domain}*\#1];
1842 set samples \pgfkeysvalueof{/pgfplots/samples};
1843 plot '+' using (t) : ((#2)/(\bp@ph@scale)) smooth unwrap;
1844 set table "\bp@prefix#3.table";
1845 plot "$meta" using ($1/(\bp@freq@scale)):(($2*\bp@ph@scale));
1846 };}
1847 }

```

`\bp@parse@opt` Parses options supplied to the main Bode macros. A `for` loop over tuples of the form `\obj/\typ/\opt` with a long list of nested if-else statements does the job. If the input `\obj` is `plot`, `axes`, `group`, `approx`, or `tikz` the corresponding `\opt` are passed, unexpanded, to the `\addplot` macro, the `\nextgroupplot` macro, the `groupplot` environment, the `\build@ZPK@plot` macro, and the `tikzpicture` environment, respectively. If `\obj` is `commands`, the corresponding `\opt` are stored, unexpanded, in the macros `\bp@ph@commands` and `\bp@mag@commands`, to be executed in appropriate axis environments.

```

1848 \newcommand{\bp@parse@opt}[1]{
1849 \gdef\bp@mag@axes{}
1850 \gdef\bp@ph@axes{}
1851 \gdef\bp@ph@plot{}
1852 \gdef\bp@mag@plot{}
1853 \gdef\bp@group{}
1854 \gdef\bp@approx{}
1855 \gdef\bp@ph@commands{}
1856 \gdef\bp@mag@commands{}
1857 \gdef\bp@tikz{}
1858 \gdef\bp@user@prefix{}
1859 \foreach \obj/\typ/\opt in {#1} {
1860 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{plot}=0
1861 \ifnum\pdf@strcmp{\unexpanded\expandafter{\typ}}{mag}=0
1862 \xdef\bp@mag@plot{\unexpanded\expandafter{\opt}}
1863 \else
1864 \ifnum\pdf@strcmp{\unexpanded\expandafter{\typ}}{ph}=0
1865 \xdef\bp@ph@plot{\unexpanded\expandafter{\opt}}
1866 \else
1867 \xdef\bp@mag@plot{\unexpanded\expandafter{\opt}}
1868 \xdef\bp@ph@plot{\unexpanded\expandafter{\opt}}
1869 \fi
1870 \fi
1871 \else
1872 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{axes}=0
1873 \ifnum\pdf@strcmp{\unexpanded\expandafter{\typ}}{mag}=0
1874 \xdef\bp@mag@axes{\unexpanded\expandafter{\opt}}
1875 \else
1876 \ifnum\pdf@strcmp{\unexpanded\expandafter{\typ}}{ph}=0
1877 \xdef\bp@ph@axes{\unexpanded\expandafter{\opt}}
1878 \else
1879 \xdef\bp@mag@axes{\unexpanded\expandafter{\opt}}

```

```

1880         \xdef\bp@ph@axes{\unexpanded\expandafter{\opt}}
1881     \fi
1882 \fi
1883 \else
1884     \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{group}=0
1885         \xdef\bp@group{\unexpanded\expandafter{\opt}}
1886     \else
1887         \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{approx}=0
1888             \xdef\bp@approx{\unexpanded\expandafter{\opt}}
1889         \else
1890             \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{commands}=0
1891                 \ifnum\pdf@strcmp{\unexpanded\expandafter{\typ}}{ph}=0
1892                     \xdef\bp@ph@commands{\unexpanded\expandafter{\opt}}
1893                 \else
1894                     \xdef\bp@mag@commands{\unexpanded\expandafter{\opt}}
1895                 \fi
1896             \else
1897                 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{tikz}=0
1898                     \xdef\bp@tikz{\unexpanded\expandafter{\opt}}
1899                 \else
1900                     \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{prefix}=0
1901                         \xdef\bp@user@prefix{\unexpanded\expandafter{\opt}}
1902                     \else
1903                         \xdef\bp@user@prefix{}
1904                         \xdef\bp@mag@plot{\unexpanded\expandafter{\bp@mag@plot},
1905                             \unexpanded\expandafter{\obj}}
1906                         \xdef\bp@ph@plot{\unexpanded\expandafter{\bp@ph@plot},
1907                             \unexpanded\expandafter{\obj}}
1908                     \fi
1909                 \fi
1910             \fi
1911         \fi
1912     \fi
1913 \fi
1914 \fi
1915 }
1916 }

```

\bp@parse@env@opt Parses options supplied to the Bode, Nyquist, and Nichols environments. A **for** loop over tuples of the form **\obj/\opt**, processed using nested if-else statements does the job. The input **\obj** should either be **axes** or **tikz**, and the corresponding **\opt** are passed, unexpanded, to the **axis** environment and the **tikzpicture** environment, respectively.

```

1917 \newcommand{\bp@parse@env@opt}[1]{
1918     \gdef\bp@axes{}
1919     \gdef\bp@tikz{}
1920     \gdef\bp@user@prefix{}
1921     \foreach \obj/\opt in {#1} {
1922         \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{axes}=0
1923             \xdef\bp@axes{\unexpanded\expandafter{\opt}}
1924         \else
1925             \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{tikz}=0
1926                 \xdef\bp@tikz{\unexpanded\expandafter{\opt}}
1927             \else
1928                 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{prefix}=0
1929                     \xdef\bp@user@prefix{\unexpanded\expandafter{\opt}}
1930                 \else
1931                     \xdef\bp@user@prefix{}
1932                     \xdef\bp@axes{\unexpanded\expandafter{\bp@axes},
1933                         \unexpanded\expandafter{\obj}}
1934                 \fi
1935             \fi
1936         \fi

```

```

1937 }
1938 }

```

`\bp@parse@add@Bode@opt` Parses options for the unified `\addBodePlot` macro. Handles `linear` and `asymptotic` approximation options plus general plot styling options.

```

1939 \newcommand{\bp@parse@add@Bode@opt}[1]{
1940   \gdef\bp@plot{}
1941   \gdef\bp@approx{}
1942   \foreach \opt in {#1} {
1943     \ifx\opt\@empty\else
1944       \ifnum\pdf@strcmp{\unexpanded\expandafter{\opt}}{linear}=0
1945         \xdef\bp@approx{\unexpanded\expandafter{\opt}}
1946       \else
1947         \ifnum\pdf@strcmp{\unexpanded\expandafter{\opt}}{asymptotic}=0
1948           \xdef\bp@approx{\unexpanded\expandafter{\opt}}
1949         \else
1950           \xdef\bp@plot{\unexpanded\expandafter{\bp@plot}%
1951             \unexpanded\expandafter{\opt}},
1952         \fi
1953       \fi
1954     \fi
1955   }
1956 }

```

4.6 Nyquist plots

4.6.1 User macros

`\NyquistZPK` Converts magnitude and phase parametric functions built using `\bp@ZPK@plot` into real part and imaginary part parametric functions. A plot of these is the Nyquist plot. The parametric functions are then plotted in a `tikzpicture` environment using the `\addplot` macro. Unless the package is loaded with the option `pgf`, the parametric functions are evaluated using `gnuplot`. A large number of samples is typically needed to get a smooth plot because frequencies near 0 result in plot points that are very close to each other. Linear frequency sampling is unnecessarily fine near zero and very coarse for large ω . Logarithmic sampling makes it worse, perhaps inverse logarithmic sampling will help, pull requests to fix that are welcome!

```

1957 \NewDocumentCommand{\NyquistZPK}{ 0{} m G{} G{} }{%
1958   \pgfutil@ifempty{#3}{%
1959     \pgfkeys{/bodeplot/nyquist/.cd, reset}
1960     \pgfkeys{/bodeplot/nyquist/.cd, #1}
1961     \pgfkeysgetvalue{/bodeplot/nyquist/@axes}{\bp@axes}
1962     \pgfkeysgetvalue{/bodeplot/nyquist/@commands}{\bp@commands}
1963     \pgfkeysgetvalue{/bodeplot/nyquist/@tikz}{\bp@tikz}
1964     \pgfkeysgetvalue{/bodeplot/nyquist/@prefix}{\bp@user@prefix}
1965     \bp@zpk@new@to@legacy{#2}
1966   }{%
1967     \bp@parse@N@opt{#1}
1968     \edef\bp@legacy{#2}
1969     \edef\bp@domain@start{#3}
1970     \edef\bp@domain@end{#4}
1971   }%
1972   \gdef\bp@mag{}\gdef\bp@ph{}%
1973   \edef\bp@cmd{\noexpand\begin{tikzpicture}
1974     [\unexpanded\expandafter{\bp@tikz}]\bp@cmd
1975     \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
1976   \edef\bp@cmd{\noexpand\begin{axis}[
1977     bp@style,
1978     domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
1979     height=5cm,
1980     xlabel={$\text{Re}$},
1981     ylabel={$\text{Im}$},

```

```

1982     samples=500,
1983     \unexpanded\expandafter{\bp@axes}
1984   }}
1985   \bp@cmd
1986     \addplot [only marks,mark=+,thick,red] (-1 , 0);
1987     \edef\bp@cmd{\noexpand\addplot
1988     [variable=t, thick, trig format plots=rad,
1989     \unexpanded\expandafter{\bp@plot}}}
1990     \ifpgfarg
1991       \bp@cmd
1992       ( {\n@pow{10}{((\bp@mag)/20)}}*cos((\bp@ph)/(\bp@ph@scale))),
1993       {\n@pow{10}{((\bp@mag)/20)}}*sin((\bp@ph)/(\bp@ph@scale))) );
1994     \bp@commands
1995   \else
1996     \stepcounter{bp@gnuplot@id}
1997     \bp@cmd gnuplot [parametric, bp@gnu@prefix] {
1998       \n@pow{10}{((\bp@mag)/20)}}*cos((\bp@ph)/(\bp@ph@scale)),
1999       \n@pow{10}{((\bp@mag)/20)}}*sin((\bp@ph)/(\bp@ph@scale))
2000     };
2001     \bp@commands
2002   \fi
2003   \end{axis}
2004 \end{tikzpicture}
2005 }

```

The following code handles active characters to avoid conflicts with ‘babel.’

```

2006 \AtBeginDocument{%
2007   \if@babel
2008     \let\Orig@NyquistZPK\NyquistZPK
2009     \renewcommand{\NyquistZPK}{%
2010       \expandafter\shorthandoff\expandafter{\bp@short@list}
2011       \NyquistZPK@Shorthandoff
2012     }
2013     \NewDocumentCommand{\NyquistZPK@Shorthandoff}{ 0{} m G{} G{} }{%
2014       \Orig@NyquistZPK[#1]{#2}{#3}{#4}
2015       \expandafter\shorthandon\expandafter{\bp@short@list}
2016     }
2017   \fi
2018 }

```

\NyquistTF Implementation of this macro is very similar to the **\NyquistZPK** macro above. The only difference is a slightly different parsing of the mandatory arguments via **\bp@TF@plot**.

```

2019 \NewDocumentCommand{\NyquistTF}{ 0{} m G{} G{} }{%
2020   \pgfutil@ifempty{#3}{%
2021     \pgfkeys{/bodeplot/nyquist/.cd, reset}
2022     \pgfkeys{/bodeplot/nyquist/.cd, #1}
2023     \pgfkeysgetvalue{/bodeplot/nyquist/@axes}{\bp@axes}
2024     \pgfkeysgetvalue{/bodeplot/nyquist/@commands}{\bp@commands}
2025     \pgfkeysgetvalue{/bodeplot/nyquist/@tikz}{\bp@tikz}
2026     \pgfkeysgetvalue{/bodeplot/nyquist/@prefix}{\bp@user@prefix}
2027     \bp@tf@new@to@legacy{#2}
2028   }{%
2029     \bp@parse@N@opt{#1}
2030     \edef\bp@legacy{#2}
2031     \edef\bp@domain@start{#3}
2032     \edef\bp@domain@end{#4}
2033   }%
2034   \gdef\bp@mag{}\gdef\bp@ph{}%
2035   \edef\bp@cmd{\noexpand\begin{tikzpicture}
2036   [\unexpanded\expandafter{\bp@tikz}]\bp@cmd
2037   \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2038   \edef\bp@cmd{\noexpand\begin{axis}[
2039     bp@style,
2040     domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,

```

```

2041     height=5cm,
2042     xlabel={\Re$},
2043     ylabel={\Im$},
2044     samples=500,
2045     \unexpanded\expandafter{\bp@axes}
2046   }}
2047   \bp@cmd
2048     \addplot [only marks, mark=+, thick, red] (-1 , 0);
2049     \edef\bp@cmd{\noexpand\addplot
2050     [variable=t, thick, trig format plots=rad,
2051     \unexpanded\expandafter{\bp@plot}]}
2052     \if@pgfarg
2053       \bp@cmd
2054         ( {\n@pow{10}{((\bp@mag)/20)}}*cos((\bp@ph)/(\bp@ph@scale))),
2055         {\n@pow{10}{((\bp@mag)/20)}}*sin((\bp@ph)/(\bp@ph@scale)) );
2056       \bp@commands
2057     \else
2058       \stepcounter{bp@gnuplot@id}
2059       \bp@cmd gnuplot [parametric, bp@gnu@prefix] {
2060         \n@pow{10}{((\bp@mag)/20)}}*cos((\bp@ph)/(\bp@ph@scale)),
2061         \n@pow{10}{((\bp@mag)/20)}}*sin((\bp@ph)/(\bp@ph@scale))
2062       };
2063       \bp@commands
2064     \fi
2065   \end{axis}
2066 \end{tikzpicture}
2067 }

```

The following code handles active characters to avoid conflicts with ‘babel.’

```

2068 \AtBeginDocument{%
2069   \if@babel
2070     \let\Orig@NyquistTF\NyquistTF
2071     \renewcommand{\NyquistTF}{%
2072       \expandafter\shorthandoff\expandafter{\bp@short@list}
2073       \NyquistTF@Shorthandoff
2074     }
2075     \NewDocumentCommand{\NyquistTF@Shorthandoff}{ 0{} m G{} G{} }{%
2076       \Orig@NyquistTF[#1]{#2}{#3}{#4}
2077       \expandafter\shorthandon\expandafter{\bp@short@list}
2078     }
2079   \fi
2080 }

```

\addNyquistZPKPlot Adds Nyquist plot of a transfer function in ZPK form with dual interface support. Converts magnitude and phase to real and imaginary parts for parametric plotting.

```

2081 \NewDocumentCommand{\addNyquistZPKPlot}{ 0{} m }{%
2082   \edef\bp@spec@arg{#2}%
2083   \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2084     \bp@zpk@new@to@legacy{\bp@spec@arg}}{\edef\bp@legacy{\bp@spec@arg}}%
2085   \gdef\bp@mag{}\gdef\bp@ph{}%
2086   \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2087   \if@pgfarg
2088     \edef\bp@cmd{\noexpand\addplot
2089     [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
2090     variable=t, thick, trig format plots=rad, #1]}%
2091     \bp@cmd ( {\n@pow{10}{((\bp@mag)/20)}}*cos((\bp@ph)/(\bp@ph@scale))),
2092             {\n@pow{10}{((\bp@mag)/20)}}*sin((\bp@ph)/(\bp@ph@scale)) );
2093   \else
2094     \stepcounter{bp@gnuplot@id}%
2095     \edef\bp@cmd{\noexpand\addplot
2096     [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale, thick, #1]
2097     \bp@cmd gnuplot [parametric, bp@gnu@prefix] {%
2098       \n@pow{10}{((\bp@mag)/20)}}*cos((\bp@ph)/(\bp@ph@scale)),
2099       \n@pow{10}{((\bp@mag)/20)}}*sin((\bp@ph)/(\bp@ph@scale))

```

```

2100 };
2101 \fi
2102 }

```

\addNyquistTFPlot Adds Nyquist plot of a transfer function in TF form with dual interface support. Converts magnitude and phase to real and imaginary parts for parametric plotting.

```

2103 \NewDocumentCommand{\addNyquistTFPlot}{ 0{ } m }{%
2104   \edef\bp@spec@arg{#2}%
2105   \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2106     \bp@tf@new@to@legacy{\bp@spec@arg}}{\edef\bp@legacy{\bp@spec@arg}}%
2107   \gdef\bp@mag{}\gdef\bp@ph{}%
2108   \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2109   \if@pgfarg
2110     \edef\bp@cmd{\noexpand\addplot
2111       [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
2112       variable=t, thick, trig format plots=rad, #1]}%
2113     \bp@cmd ( {\n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale))},
2114       {\n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))} );
2115   \else
2116     \stepcounter{bp@gnuplot@id}%
2117     \edef\bp@cmd{\noexpand\addplot
2118       [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale, thick, #1]
2119     \bp@cmd gnuplot [parametric, bp@gnu@prefix] {%
2120       \n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale)),
2121       \n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))
2122     };
2123   \fi
2124 }

```

NyquistPlot An environment to host `\addNyquist...` macros that pass parametric functions to `\addplot`. Uses the defaults specified in `bp@style` to create a shortcut that includes the `tikzpicture` and `axis` environments.

```

2125 \AtBeginDocument{%
2126   \if@babel
2127     \AddToHook{env/NyquistPlot/begin}{%
2128       \expandafter\shorthandoff\expandafter{\bp@short@list}
2129     }
2130     \AddToHook{env/NyquistPlot/end}{%
2131       \expandafter\shorthandon\expandafter{\bp@short@list}
2132     }
2133   \fi
2134 }
2135 \NewDocumentEnvironment{NyquistPlot}{ 0{ } G{ } G{ } +b }{
2136   \pgfutil@ifempty{#2}{%
2137     \pgfkeys{/bodeplot/env/.cd, reset}
2138     \pgfkeys{/bodeplot/env/.cd, #1}
2139     \pgfkeysgetvalue{/bodeplot/env/@tikz}{\bp@tikz}
2140     \pgfkeysgetvalue{/bodeplot/env/@prefix}{\bp@user@prefix}
2141   }{%
2142     \bp@parse@env@opt{#1}
2143     \edef\bp@domain@start{#2}%
2144     \edef\bp@domain@end{#3}%
2145   }
2146   \edef\bp@cmd{\noexpand\begin{tikzpicture}
2147     [\unexpanded\expandafter{\bp@tikz}]\bp@cmd
2148   \edef\bp@cmd{\noexpand\begin{axis}[
2149     bp@style,
2150     height=5cm,
2151     domain=\bp@domain@start:\bp@domain@end,
2152     xlabel={\$Re\$},
2153     ylabel={\$Im\$},
2154     \unexpanded\expandafter{\bp@axes}%
2155   ]}\bp@cmd

```

```

2156 \addplot [only marks,mark=+,thick,red] (-1 , 0);
2157 #4
2158 \end{axis}
2159 \end{tikzpicture}
2160 }{}

```

4.6.2 Internal macros

`\bp@parse@N@opt` Parses options supplied to the main Nyquist and Nichols macros. A `for` loop over tuples of the form `\obj/\opt`, processed using nested if-else statements does the job. If the input `\obj` is `plot`, `axes`, `scale`, or `tikz` then the corresponding `\opt` are passed, unexpanded, to the `\addplot` macro, the `axis` environment, the scaling option, and the `tikzpicture` environment, respectively.

```

2161 \newcommand{\bp@parse@N@opt}[1]{
2162 \gdef\bp@axes{}
2163 \gdef\bp@plot{}
2164 \gdef\bp@commands{}
2165 \gdef\bp@tikz{}
2166 \gdef\bp@user@prefix{}
2167 \gdef\bp@scale{linear}
2168 \foreach \obj/\opt in {#1} {
2169 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{axes}=0
2170 \xdef\bp@axes{\unexpanded\expandafter{\opt}}
2171 \else
2172 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{plot}=0
2173 \xdef\bp@plot{\unexpanded\expandafter{\opt}}
2174 \else
2175 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{commands}=0
2176 \xdef\bp@commands{\unexpanded\expandafter{\opt}}
2177 \else
2178 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{tikz}=0
2179 \xdef\bp@tikz{\unexpanded\expandafter{\opt}}
2180 \else
2181 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{scale}=0
2182 \xdef\bp@scale{\unexpanded\expandafter{\opt}}
2183 \else
2184 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{prefix}=0
2185 \xdef\bp@user@prefix{\unexpanded\expandafter{\opt}}
2186 \else
2187 \xdef\bp@user@prefix{}
2188 \xdef\bp@plot{\unexpanded\expandafter{\bp@plot},
2189 \unexpanded\expandafter{\obj}}
2190 \fi
2191 \fi
2192 \fi
2193 \fi
2194 \fi
2195 \fi
2196 }
2197 }

```

4.7 Nichols charts

`\NicholsZPK`

```

2198 \NewDocumentCommand{\NicholsZPK}{ 0{ } m G{ } G{ } }{%
2199 \pgfutil@ifempty{#3}{%
2200 \pgfkeys{/bodeplot/nichols/.cd, reset}
2201 \pgfkeys{/bodeplot/nichols/.cd, #1}
2202 \pgfkeysgetvalue{/bodeplot/nichols/@axes}{\bp@axes}
2203 \pgfkeysgetvalue{/bodeplot/nichols/@commands}{\bp@commands}
2204 \pgfkeysgetvalue{/bodeplot/nichols/@tikz}{\bp@tikz}
2205 \pgfkeysgetvalue{/bodeplot/nichols/@prefix}{\bp@user@prefix}

```



```

2206 \bp@zpk@new@to@legacy{#2}
2207 }{%
2208 \bp@parse@N@opt{#1}
2209 \edef\bp@legacy{#2}
2210 \edef\bp@domain@start{#3}
2211 \edef\bp@domain@end{#4}
2212 }%
2213 \gdef\bp@mag{}\gdef\bp@ph{}%
2214 \edef\bp@cmd{\noexpand\begin{tikzpicture}
2215 [\unexpanded\expandafter{\bp@tikz}]]\bp@cmd
2216 \bp@ZPK@plot{\bp@mag}{\bp@ph}{}\bp@legacy}%
2217 \edef\bp@cmd{\noexpand\begin{axis}[
2218 bp@ph@x@label,
2219 bp@style,
2220 domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
2221 height=5cm,
2222 ylabel={Gain (dB)},
2223 samples=500,
2224 \unexpanded\expandafter{\bp@axes}
2225 ]}
2226 \bp@cmd
2227 \edef\bp@cmd{\noexpand\addplot
2228 [variable=t, thick, trig format plots=rad,
2229 \unexpanded\expandafter{\bp@plot}]}
2230 \ifpgfarg
2231 \bp@cmd ( {\bp@ph} , {\bp@mag} );
2232 \bp@commands
2233 \else
2234 \stepcounter{bp@gnuplot@id}
2235 \bp@cmd gnuplot [raw gnuplot, bp@gnu@prefix]
2236 { set table $meta;
2237 set logscale x 10;
2238 set dummy t;
2239 set samples \pgfkeysvalueof{/pgfplots/samples};
2240 set trange [\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale];
2241 plot '+' using (\bp@mag) : ((\bp@ph)/(\bp@ph@scale));
2242 unset logscale x;
2243 set table "\bp@prefix\arabic{bp@gnuplot@id}.table";
2244 plot "$meta" using ($2*\bp@ph@scale):($1);
2245 };
2246 \bp@commands
2247 \fi
2248 \end{axis}
2249 \end{tikzpicture}
2250 }
2251 \AtBeginDocument{%
2252 \if@babel
2253 \let\Orig@NicholsZPK\NicholsZPK
2254 \renewcommand{\NicholsZPK}{%
2255 \expandafter\shorthandoff\expandafter{\bp@short@list}
2256 \NicholsZPK@Shorthandoff
2257 }
2258 \NewDocumentCommand{\NicholsZPK@Shorthandoff}{ 0{} m G{} G{} }{%
2259 \Orig@NicholsZPK[#1]{#2}{#3}{#4}
2260 \expandafter\shorthandon\expandafter{\bp@short@list}
2261 }
2262 \fi
2263 }

```

\NicholsTF

```

2264 \NewDocumentCommand{\NicholsTF}{ 0{} m G{} G{} }{%
2265 \ifpgfarg
2266 \bp@tf@to@zpk{#2}{\bp@tmp@zpk}{\bp@tmp@status}

```

```

2267 \ifnum\pdf@strcmp{\bp@tmp@status}{true}=0
2268 \pgfutil@ifempty{#3}{%
2269 \NicholsZPK[#1]{\bp@tmp@zpk}%
2270 }{%
2271 \NicholsZPK[domain=#3:#4,#1]{\bp@tmp@zpk}%
2272 }%
2273 \def\bp@fallback{false}
2274 \else
2275 \def\bp@fallback{true}
2276 \fi
2277 \else
2278 \def\bp@fallback{true}
2279 \fi
2280 \ifnum\pdf@strcmp{\bp@fallback}{true}=0
2281 \pgfutil@ifempty{#3}{%
2282 \pgfkeys{/bodeplot/nichols/.cd, reset}
2283 \pgfkeys{/bodeplot/nichols/.cd, #1}
2284 \pgfkeysgetvalue{/bodeplot/nichols/@axes}{\bp@axes}
2285 \pgfkeysgetvalue{/bodeplot/nichols/@commands}{\bp@commands}
2286 \pgfkeysgetvalue{/bodeplot/nichols/@tikz}{\bp@tikz}
2287 \pgfkeysgetvalue{/bodeplot/nichols/@prefix}{\bp@user@prefix}
2288 \bp@tf@new@to@legacy{#2}
2289 }{%
2290 \bp@parse@N@opt{#1}
2291 \edef\bp@legacy{#2}
2292 \edef\bp@domain@start{#3}
2293 \edef\bp@domain@end{#4}
2294 }%
2295 \gdef\bp@mag{}\gdef\bp@ph{}%
2296 \edef\bp@cmd{\noexpand\begin{tikzpicture}
2297 [\unexpanded\expandafter{\bp@tikz}]]\bp@cmd
2298 \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2299 \edef\bp@cmd{\noexpand\begin{axis}[
2300 \bp@ph@x@label,
2301 \bp@style,
2302 domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
2303 height=5cm,
2304 ylabel={Gain (dB)},
2305 samples=500,
2306 \unexpanded\expandafter{\bp@axes}
2307 ]}
2308 \bp@cmd
2309 \edef\bp@cmd{\noexpand\addplot
2310 [variable=t, thick, trig format plots=rad,
2311 \unexpanded\expandafter{\bp@plot}]]%
2312 \if@pgfarg
2313 \bp@cmd ( {\n@mod{\bp@ph}{2*pi*\bp@ph@scale}} , {\bp@mag} );
2314 \bp@commands
2315 \else
2316 \stepcounter{bp@gnuplot@id}
2317 \bp@cmd gnuplot [raw gnuplot, bp@gnu@prefix]
2318 { set table $meta1;
2319 set logscale x 10;
2320 set dummy t;
2321 set samples \pgfkeysvalueof{/pgfplots/samples};
2322 set trange [\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale];
2323 plot '+' using (\bp@mag) : ((\bp@ph)/(\bp@ph@scale));
2324 unset logscale x;
2325 set table $meta2;
2326 plot "$meta1" using ($1):($2) smooth unwrap;
2327 set table "\bp@prefix\arabic{bp@gnuplot@id}.table";
2328 plot "$meta2" using ($2*\bp@ph@scale):($1);
2329 };

```

```

2330         \bp@commands
2331     \fi
2332     \end{axis}
2333 \end{tikzpicture}
2334 \fi
2335 }

```

The following code handles active characters to avoid conflicts with ‘babel.’

```

2336 \AtBeginDocument{
2337   \if@babel
2338     \let\Orig@NicholsTF\NicholsTF
2339     \renewcommand{\NicholsTF}{%
2340       \expandafter\shorthandoff\expandafter{\bp@short@list}
2341       \NicholsTF@Shorthandoff
2342     }
2343     \NewDocumentCommand{\NicholsTF@Shorthandoff}{ O{} m G{} G{} }{%
2344       \Orig@NicholsTF[#1]{#2}{#3}{#4}
2345       \expandafter\shorthandon\expandafter{\bp@short@list}
2346     }
2347     \AddToHook{env/NicholsChart/begin}{%
2348       \expandafter\shorthandoff\expandafter{\bp@short@list}
2349     }
2350     \AddToHook{env/NicholsChart/end}{%
2351       \expandafter\shorthandon\expandafter{\bp@short@list}
2352     }
2353   \fi
2354 }

```

NicholsChart

```

2355 \NewDocumentEnvironment{NicholsChart}{ O{} G{} G{} +b }{
2356   \pgfutil@ifempty{#2}{%
2357     \pgfkeys{/bodeplot/env/.cd, reset}
2358     \pgfkeys{/bodeplot/env/.cd, #1}
2359     \pgfkeysgetvalue{/bodeplot/env/@tikz}{\bp@tikz}
2360     \pgfkeysgetvalue{/bodeplot/env/@prefix}{\bp@user@prefix}
2361   }{%
2362     \bp@parse@env@opt{#1}
2363     \edef\bp@domain@start{#2}
2364     \edef\bp@domain@end{#3}
2365   }
2366   \edef\bp@cmd{\noexpand\begin{tikzpicture}
2367     [\unexpanded\expandafter{\bp@tikz}]\bp@cmd
2368   \edef\bp@cmd{\noexpand\begin{axis}[
2369     bp@ph@x@label,
2370     bp@style,
2371     domain=\bp@domain@start:\bp@domain@end,
2372     height=5cm,
2373     ylabel={Gain (dB)},
2374     \unexpanded\expandafter{\bp@axes}%
2375   ]}\bp@cmd
2376     #4
2377   \end{axis}
2378 \end{tikzpicture}
2379 }{}

```

\addNicholsZPKChart Generates Nichols chart for ZPK system with dual interface support. Plots phase vs magnitude.

```

2380 \NewDocumentCommand{\addNicholsZPKChart}{ O{} m }{%
2381   \edef\bp@spec@arg{#2}%
2382   \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2383     \bp@zpk@new@to@legacy{\bp@spec@arg}}{\edef\bp@legacy{\bp@spec@arg}}%
2384   \gdef\bp@mag{\gdef\bp@ph{}}%
2385   \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2386   \if@pgfarg

```

```

2387 \edef\bp@cmd{\noexpand\addplot
2388 [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
2389 variable=t, thick, trig format plots=rad, #1]]%
2390 \bp@cmd ( {\bp@ph} , {\bp@mag} );
2391 \else
2392 \stepcounter{bp@gnuplot@id}%
2393 \addplot [thick, #1] gnuplot [raw gnuplot, bp@gnu@prefix] {%
2394 set table $meta;
2395 set logscale x 10;
2396 set dummy t;
2397 set samples \pgfkeysvalueof{/pgfplots/samples};
2398 set trange [\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale];
2399 plot '+' using (\bp@mag) : ((\bp@ph)/(\bp@ph@scale));
2400 unset logscale x;
2401 set table "\bp@prefix\arabic{bp@gnuplot@id}.table";
2402 plot "$meta" using ($2*\bp@ph@scale):($1);
2403 };
2404 \fi
2405 }

```

\addNicholsTFChart Generates Nichols chart for TF system with dual interface support. Plots phase vs magnitude with phase unwrapping.

```

2406 \NewDocumentCommand{\addNicholsTFChart}{0}{m}{%
2407 \edef\bp@spec@arg{#2}%
2408 \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2409 \bp@tf@new@to@legacy{\bp@spec@arg}}{\edef\bp@legacy{\bp@spec@arg}}}%
2410 \gdef\bp@mag{\gdef\bp@ph}%
2411 \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2412 \ifpgfarg
2413 \edef\bp@cmd{\noexpand\addplot
2414 [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
2415 variable=t, thick, trig format plots=rad, #1]]%
2416 \bp@cmd ( {\nmod{\bp@ph}{2*pi*\bp@ph@scale}} , {\bp@mag} );
2417 \else
2418 \stepcounter{bp@gnuplot@id}%
2419 \addplot [thick, #1] gnuplot [raw gnuplot, bp@gnu@prefix] {%
2420 set table $meta1;
2421 set logscale x 10;
2422 set dummy t;
2423 set samples \pgfkeysvalueof{/pgfplots/samples};
2424 set trange [\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale];
2425 plot '+' using (\bp@mag) : ((\bp@ph)/(\bp@ph@scale));
2426 unset logscale x;
2427 set table $meta2;
2428 plot "$meta1" using ($1):($2) smooth unwrap;
2429 set table "\bp@prefix\arabic{bp@gnuplot@id}.table";
2430 plot "$meta2" using ($2*\bp@ph@scale):($1);
2431 };
2432 \fi
2433 }

```

4.8 Pole-zero maps

4.8.1 User macros

\PoleZeroMapZPK Creates a pole-zero map similar to MATLAB's `pzmap` function. Poles are plotted as 'x' markers and zeros as 'o' markers on the complex s-plane. The gain parameter is ignored since it does not affect pole-zero locations, and delay is also ignored since it does not add poles or zeros.

```

2434 \NewDocumentCommand{\PoleZeroMapZPK}{0}{m}{%
2435 \edef\bp@spec@arg{#2}%
2436 \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2437 \pgfkeys{/bodeplot/pzmap/.cd, reset}

```

```

2438 \pgfkeys{/bodeplot/pzmap/.cd, #1}
2439 \pgfkeysgetvalue{/bodeplot/pzmap/@axes}{\bp@axes}
2440 \pgfkeysgetvalue{/bodeplot/pzmap/@plot}{\bp@plot}
2441 \pgfkeysgetvalue{/bodeplot/pzmap/@commands}{\bp@commands}
2442 \pgfkeysgetvalue{/bodeplot/pzmap/@tikz}{\bp@tikz}
2443 \pgfkeysgetvalue{/bodeplot/pzmap/@prefix}{\bp@user@prefix}
2444 \pgfkeysgetvalue{/bodeplot/pzmap/@scale}{\bp@scale}
2445 \bp@zpk@new@to@legacy{\bp@spec@arg}
2446 }{%
2447 \bp@parse@N@opt{#1}
2448 \edef\bp@legacy{\bp@spec@arg}
2449 }%
2450 \ifnum\pdf@strcmp{\bp@scale}{log}=0
2451 \bp@min@real@ZPK{\bp@legacy}
2452 \bp@min@im@ZPK{\bp@legacy}
2453 \pgfkeys{/pgf/fpu=true}
2454 \ifx\bp@has@positive@values\bp@min@false
2455 \xdef\bp@PZZPK@ticksXPos{0}
2456 \else
2457 \pgfmathparse{max(\bp@max@re@pos@pow@10 - \bp@min@re@pow@10 + 1, 1)}
2458 \pgfmathfloatoint{\pgfmathresult}
2459 \xdef\bp@PZZPK@ticksXPos{\pgfmathresult}
2460 \fi
2461 \ifx\bp@has@negative@values\bp@min@false
2462 \xdef\bp@PZZPK@ticksXNeg{0}
2463 \else
2464 \pgfmathparse{max(\bp@max@re@neg@pow@10 - \bp@min@re@pow@10 + 1, 1)}
2465 \pgfmathfloatoint{\pgfmathresult}
2466 \xdef\bp@PZZPK@ticksXNeg{\pgfmathresult}
2467 \fi
2468 \pgfkeys{/pgf/fpu=false}
2469 \def\PoleZeroMapZPK@formatXTick##1{%
2470 \pgfmathtruncatemacro{\PoleZeroMapZPK@tick}{##1}%
2471 \ifnum\PoleZeroMapZPK@tick=0
2472 $0$
2473 \else
2474 \pgfmathtruncatemacro{\PoleZeroMapZPK@exp}
2475 {\bp@min@re@pow@10 + abs(\PoleZeroMapZPK@tick) - 1}%
2476 \ifnum\PoleZeroMapZPK@tick>0
2477 $10^{\PoleZeroMapZPK@exp}$%
2478 \else
2479 $-10^{\PoleZeroMapZPK@exp}$%
2480 \fi
2481 \fi
2482 }
2483 \def\PoleZeroMapZPK@formatYTick##1{%
2484 \pgfmathtruncatemacro{\PoleZeroMapZPK@tick}{##1}%
2485 \ifnum\PoleZeroMapZPK@tick=0
2486 $0$
2487 \else
2488 \pgfmathtruncatemacro{\PoleZeroMapZPK@exp}
2489 {\bp@min@im@pow@10 + abs(\PoleZeroMapZPK@tick) - 1}%
2490 \ifnum\PoleZeroMapZPK@tick>0
2491 $10^{\PoleZeroMapZPK@exp}$%
2492 \else
2493 $-10^{\PoleZeroMapZPK@exp}$%
2494 \fi
2495 \fi
2496 }
2497 \def\PoleZeroMapZPK@xticklabel{\PoleZeroMapZPK@formatXTick{\tick}}
2498 \def\PoleZeroMapZPK@yticklabel{\PoleZeroMapZPK@formatYTick{\tick}}
2499 \fi
2500 \edef\bp@cmd{\noexpand\begin{tikzpicture}}

```

```

2501     [\unexpanded\expandafter{\bp@tikz}}]
2502 \bp@cmd
2503 \ifnum\pdf@strcmp{\bp@scale}{log}=0
2504   \edef\bp@cmd{\noexpand\begin{axis}[
2505     xlabel={\Re$},
2506     ylabel={\Im$},
2507     axis lines=center,
2508     grid=major,
2509     height=6cm,
2510     enlarge x limits=0.2,
2511     enlarge y limits=0.2,
2512     xtick distance=1,
2513     ytick distance=1,
2514     xticklabel=\noexpand\PoleZeroMapZPK@xticklabel,
2515     yticklabel=\noexpand\PoleZeroMapZPK@yticklabel,
2516     x filter/.expression={abs(x) < \bp@min@re@threshold@result ?
2517     0 : (x >= 0 ? (log10(max(min(x, 1e100), 1e-
2518 100)) - \bp@min@re@pow@10 + 1) :
2519     (-log10(max(min(-x, 1e100), 1e-100)) + \bp@min@re@pow@10 - 1))},
2520     y filter/.expression={abs(y) < \bp@min@im@threshold@result ?
2521     0 : (y >= 0 ? (log10(max(min(y, 1e100), 1e-
2522 100)) - \bp@min@im@pow@10 + 1) :
2523     (-log10(max(min(-y, 1e100), 1e-100)) + \bp@min@im@pow@10 - 1))},
2524     \unexpanded\expandafter{\bp@axes}
2525   ]}
2526 \else
2527   \edef\bp@cmd{\noexpand\begin{axis}[
2528     xlabel={\Re$},
2529     ylabel={\Im$},
2530     axis lines=center,
2531     grid=major,
2532     height=6cm,
2533     enlarge x limits=0.2,
2534     enlarge y limits=0.2,
2535     \unexpanded\expandafter{\bp@axes}
2536   ]}
2537 \fi
2538 \bp@cmd
2539 \foreach \feature/\values in \bp@legacy {
2540   \ifnum\pdf@strcmp{\feature}{z}=0
2541     \foreach \z in \values {
2542       \foreach \y [count=\zcnt] in \z {
2543         \ifnum\zcnt=1
2544           \xdef\bp@zre{\y}
2545         \fi
2546         \ifnum\zcnt=2
2547           \xdef\bp@zim{\y}
2548         \fi
2549         \xdef\bp@Last@Loop@z{\zcnt}
2550       }
2551     }
2552     \ifnum\bp@Last@Loop@z=1
2553       \xdef\bp@zim{0}
2554     \fi
2555     \edef\bp@cmd{\noexpand\addplot
2556     [only marks, mark=o, mark size=3pt, thick, blue,
2557     \unexpanded\expandafter{\bp@plot}]}
2558     \bp@cmd coordinates {(\bp@zre,\bp@zim)};
2559   }
2560 \fi
2561 \ifnum\pdf@strcmp{\feature}{p}=0
2562   \foreach \p in \values {
2563     \foreach \y [count=\pcnt] in \p {
2564       \ifnum\pcnt=1

```

```

2562         \xdef\bp@pre{\y}
2563     \fi
2564     \ifnum\pcnt=2
2565         \xdef\bp@pim{\y}
2566     \fi
2567     \xdef\bp@Last@Loop@p{\pcnt}
2568 }
2569 \ifnum\bp@Last@Loop@p=1
2570     \xdef\bp@pim{0}
2571 \fi
2572 \edef\bp@cmd{\noexpand\addplot
2573 [only marks, mark=x, mark size=3pt, thick, red,
2574 \unexpanded\expandafter{\bp@plot}]}
2575 \bp@cmd coordinates {(\bp@pre,\bp@pim)};
2576 }
2577 \fi
2578 }
2579 \bp@commands
2580 \end{axis}
2581 \end{tikzpicture}
2582 }

```

The following code handles active characters to avoid conflicts with ‘babel.’

```

2583 \AtBeginDocument{%
2584     \if@babel
2585     \let\Orig@PoleZeroMapZPK\PoleZeroMapZPK
2586     \renewcommand{\PoleZeroMapZPK}{%
2587         \expandafter\shorthandoff\expandafter{\bp@short@list}
2588         \PoleZeroMapZPK@Shorthandoff
2589     }
2590     \newcommand{\PoleZeroMapZPK@Shorthandoff}[2][]{%
2591         \Orig@PoleZeroMapZPK[#1]{#2}
2592         \expandafter\shorthandon\expandafter{\bp@short@list}
2593     }
2594     \fi
2595 }

```

\PoleZeroMapTF Similar to **\PoleZeroMapZPK** but for the system specified in the TF format. This macro converts the TF specification to ZPK format internally using Python via **--shell-escape** and then calls **\PoleZeroMapZPK** to generate the plot. If Python with the **scipy** and **numpy** libraries is not available on the shell, the package returns an error message.

```

2596 \NewDocumentCommand{\PoleZeroMapTF}{ 0{ } m }{%
2597     \bp@tf@to@zpk{#2}{\bp@tmp@zpk}{\bp@tmp@status}
2598     \ifnum\pdf@strcmp{\bp@tmp@status}{true}=0
2599         \PoleZeroMapZPK[#1]{\bp@tmp@zpk}
2600     \else
2601         \PackageError{bodeplot}{Python with scipy and numpy is required to convert TF to ZPK
2602         zero map. Please ensure Python is installed and accessible via --shell-
2603         escape.}{}
2604     \fi
2605 }

```

The following code handles active characters to avoid conflicts with ‘babel.’

```

2604 \AtBeginDocument{%
2605     \if@babel
2606     \let\Orig@PoleZeroMapTF\PoleZeroMapTF
2607     \renewcommand{\PoleZeroMapTF}{%
2608         \expandafter\shorthandoff\expandafter{\bp@short@list}
2609         \PoleZeroMapTF@Shorthandoff
2610     }
2611     \newcommand{\PoleZeroMapTF@Shorthandoff}[2][]{%
2612         \Orig@PoleZeroMapTF[#1]{#2}
2613         \expandafter\shorthandon\expandafter{\bp@short@list}

```

```

2614 }
2615 \fi
2616 }

```

4.8.2 Internal macros

`\bp@min@real@ZPK` Computes the minimum nonzero absolute value of real parts from all poles and zeros in ZPK format. This is used for automatically setting the threshold in logarithmic pole-zero maps. The result is stored in `\bp@min@re@threshold@result` and the corresponding power of 10 is stored in `\bp@min@re@pow@10`.

```

2617 \newcommand{\bp@min@real@ZPK}[1]{
2618   \gdef\bp@min@re@threshold@result{1000}
2619   \def\@bp@min@false{false}
2620   \gdef\bp@min@threshold@found{false}
2621   \global\let\bp@min@thresh@float\relax
2622   \pgfkeys{/pgf/fpu=true}
2623   \pgfmathparse{0}
2624   \global\let\bp@max@re@float=\pgfmathresult
2625   \pgfmathparse{0}
2626   \global\let\bp@max@re@pos@float=\pgfmathresult
2627   \pgfmathparse{0}
2628   \global\let\bp@max@re@neg@float=\pgfmathresult
2629   \pgfkeys{/pgf/fpu=false}
2630   \gdef\bp@max@re@value{0}
2631   \gdef\bp@has@positive@values{false}
2632   \gdef\bp@has@negative@values{false}
2633   \foreach \feature/\values in {#1} {
2634     \ifnum\pdf@strcmp{\feature}{z}=0
2635       \foreach \z in \values {
2636         \foreach \y [count=\zcnt] in \z {
2637           \ifnum\zcnt=1
2638             \pgfkeys{/pgf/fpu=true}
2639             \pgfmathparse{abs(\y)}
2640             \let\abs@valuefloat=\pgfmathresult
2641             \pgfmathfloattofixed{\abs@valuefloat}
2642             \edef\abs@value{\pgfmathresult}
2643             \pgfkeys{/pgf/fpu=false}
2644             \ifnum\pdf@strcmp{\abs@value}{0}=0\else
2645               \ifnum\pdf@strcmp{\abs@value}{0.0}=0\else
2646                 \pgfkeys{/pgf/fpu=true}
2647                 \pgfmathparse{\y >= 0 ? 1 : 0}
2648                 \pgfmathfloattoint{\pgfmathresult}
2649                 \pgfkeys{/pgf/fpu=false}
2650                 \ifnum\pgfmathresult=1
2651                   \gdef\bp@has@positive@values{true}
2652                   \pgfkeys{/pgf/fpu=true}
2653                   \pgfmathparse{\abs@valuefloat > \bp@max@re@pos@float ? 1 : 0}
2654                   \pgfmathfloattoint{\pgfmathresult}
2655                   \pgfkeys{/pgf/fpu=false}
2656                   \ifnum\pgfmathresult=1
2657                     \global\let\bp@max@re@pos@float=\abs@valuefloat
2658                   \fi
2659                 \else
2660                   \gdef\bp@has@negative@values{true}
2661                   \pgfkeys{/pgf/fpu=true}
2662                   \pgfmathparse{\abs@valuefloat > \bp@max@re@neg@float ? 1 : 0}
2663                   \pgfmathfloattoint{\pgfmathresult}
2664                   \pgfkeys{/pgf/fpu=false}
2665                   \ifnum\pgfmathresult=1
2666                     \global\let\bp@max@re@neg@float=\abs@valuefloat
2667                   \fi
2668                 \fi
2669             \pgfkeys{/pgf/fpu=true}

```



```

2670 \pgfmathparse{\abs@valuefloat > \bp@max@re@float ? 1 : 0}
2671 \pgfmathfloattoint{\pgfmathresult}
2672 \pgfkeys{/pgf/fpu=false}
2673 \ifnum\pgfmathresult=1
2674 \global\let\bp@max@re@float=\abs@valuefloat
2675 \xdef\bp@max@re@value{\abs@value}
2676 \fi
2677 \ifx\bp@min@threshold@found\@bp@min@false
2678 \xdef\bp@min@re@threshold@result{\abs@value}
2679 \global\let\bp@min@thresh@float=\abs@valuefloat
2680 \gdef\bp@min@threshold@found{true}
2681 \else
2682 \pgfkeys{/pgf/fpu=true}
2683 \pgfmathparse{\abs@valuefloat < \bp@min@thresh@float ? 1 : 0}
2684 \pgfmathfloattoint{\pgfmathresult}
2685 \pgfkeys{/pgf/fpu=false}
2686 \ifnum\pgfmathresult=1
2687 \xdef\bp@min@re@threshold@result{\abs@value}
2688 \global\let\bp@min@thresh@float=\abs@valuefloat
2689 \fi
2690 \fi
2691 \fi
2692 \fi
2693 \fi
2694 }
2695 }
2696 \fi
2697 \ifnum\pdf@strcmp{\feature}{p}=0
2698 \foreach \p in \values {
2699 \foreach \y [count=\pcnt] in \p {
2700 \ifnum\pcnt=1
2701 \pgfkeys{/pgf/fpu=true}
2702 \pgfmathparse{abs(\y)}
2703 \let\abs@valuefloat=\pgfmathresult
2704 \pgfmathfloattofixed{\abs@valuefloat}
2705 \edef\abs@value{\pgfmathresult}
2706 \pgfkeys{/pgf/fpu=false}
2707 \ifnum\pdf@strcmp{\abs@value}{0}=0\else
2708 \ifnum\pdf@strcmp{\abs@value}{0.0}=0\else
2709 \pgfkeys{/pgf/fpu=true}
2710 \pgfmathparse{\y >= 0 ? 1 : 0}
2711 \pgfmathfloattoint{\pgfmathresult}
2712 \pgfkeys{/pgf/fpu=false}
2713 \ifnum\pgfmathresult=1
2714 \gdef\bp@has@positive@values{true}
2715 \pgfkeys{/pgf/fpu=true}
2716 \pgfmathparse{\abs@valuefloat > \bp@max@re@pos@float ? 1 : 0}
2717 \pgfmathfloattoint{\pgfmathresult}
2718 \pgfkeys{/pgf/fpu=false}
2719 \ifnum\pgfmathresult=1
2720 \global\let\bp@max@re@pos@float=\abs@valuefloat
2721 \fi
2722 \else
2723 \gdef\bp@has@negative@values{true}
2724 \pgfkeys{/pgf/fpu=true}
2725 \pgfmathparse{\abs@valuefloat > \bp@max@re@neg@float ? 1 : 0}
2726 \pgfmathfloattoint{\pgfmathresult}
2727 \pgfkeys{/pgf/fpu=false}
2728 \ifnum\pgfmathresult=1
2729 \global\let\bp@max@re@neg@float=\abs@valuefloat
2730 \fi
2731 \fi
2732 \ifx\bp@min@threshold@found\@bp@min@false

```

```

2733         \xdef\bp@min@re@threshold@result{\abs@value}
2734         \global\let\bp@min@thresh@float=\abs@valuefloat
2735         \gdef\bp@min@threshold@found{true}
2736     \else
2737         \pgfkeys{/pgf/fpu=true}
2738         \pgfmathparse{\abs@valuefloat < \bp@min@thresh@float ? 1 : 0}
2739         \pgfmathfloattoint{\pgfmathresult}
2740         \pgfkeys{/pgf/fpu=false}
2741         \ifnum\pgfmathresult=1
2742             \xdef\bp@min@re@threshold@result{\abs@value}
2743             \global\let\bp@min@thresh@float=\abs@valuefloat
2744         \fi
2745     \fi
2746 \fi
2747 \fi
2748 \fi
2749 }
2750 }
2751 \fi
2752 }
2753 \ifx\bp@min@threshold@found\@bp@min@false
2754     \gdef\bp@min@re@threshold@result{0.01}
2755 \fi
2756 \xdef\bp@min@threshold@result{\bp@min@re@threshold@result}
2757 \pgfkeys{/pgf/fpu=true}
2758 \pgfmathparse{log10(\bp@min@re@threshold@result)}
2759 \let\log@result=\pgfmathresult
2760 \pgfmathparse{\log@result + 1e-5}
2761 \let\log@adjusted=\pgfmathresult
2762 \pgfmathparse{floor(\log@adjusted)}
2763 \pgfmathfloattofixed{\pgfmathresult}
2764 \xdef\bp@min@re@pow@10{\pgfmathresult}
2765 \xdef\bp@min@pow@10{\bp@min@re@pow@10}
2766 \ifx\bp@has@positive@values\@bp@min@false
2767     \xdef\bp@max@re@pos@pow@10{\bp@min@re@pow@10}
2768 \else
2769     \pgfmathparse{log10(max(\bp@max@re@pos@float,1e-100))}
2770     \let\log@bp@max@re@pos=\pgfmathresult
2771     \pgfmathparse{\log@bp@max@re@pos + 1e-5}
2772     \let\log@bp@max@re@pos@adjusted=\pgfmathresult
2773     \pgfmathparse{ceil(\log@bp@max@re@pos@adjusted)}
2774     \pgfmathfloattoint{\pgfmathresult}
2775     \xdef\bp@max@re@pos@pow@10{\pgfmathresult}
2776 \fi
2777 \ifx\bp@has@negative@values\@bp@min@false
2778     \xdef\bp@max@re@neg@pow@10{\bp@min@re@pow@10}
2779 \else
2780     \pgfmathparse{log10(max(\bp@max@re@neg@float,1e-100))}
2781     \let\log@bp@max@re@neg=\pgfmathresult
2782     \pgfmathparse{\log@bp@max@re@neg + 1e-5}
2783     \let\log@bp@max@re@neg@adjusted=\pgfmathresult
2784     \pgfmathparse{ceil(\log@bp@max@re@neg@adjusted)}
2785     \pgfmathfloattoint{\pgfmathresult}
2786     \xdef\bp@max@re@neg@pow@10{\pgfmathresult}
2787 \fi
2788 \pgfmathparse{max(\bp@max@re@pos@float > 0 ?
2789 \bp@max@re@pos@float : 0, \bp@max@re@neg@float > 0 ? \bp@max@re@neg@float : 0)}
2790 \let\bp@max@re@valuefloat=\pgfmathresult
2791 \pgfmathparse{\bp@max@re@valuefloat > 0 ?
2792 \bp@max@re@valuefloat : \bp@min@re@threshold@result}
2793 \let\bp@max@re@valuefloat=\pgfmathresult
2794 \pgfmathparse{log10(max(\bp@max@re@valuefloat,1e-100))}
2795 \let\log@bp@max@re=\pgfmathresult

```

```

2796 \pgfmathparse{\log@bp@max@re + 1e-5}
2797 \let\log@bp@max@re@adjusted=\pgfmathresult
2798 \pgfmathparse{ceil(\log@bp@max@re@adjusted)}
2799 \pgfmathfloattoint{\pgfmathresult}
2800 \xdef\bp@max@re@pow@10{\pgfmathresult}
2801 \pgfkeys{/pgf/fpu=false}
2802 }

```

\bp@min@im@ZPK Computes the minimum nonzero absolute value of imaginary parts from all poles and zeros in ZPK format. This is used for automatically setting thresholds in logarithmic pole-zero maps for imaginary axis scaling. The result is stored in \bp@min@im@threshold@result and the corresponding power of 10 is stored in \bp@min@im@pow@10.

```

2803 \newcommand{\bp@min@im@ZPK}[1]{
2804   \gdef\bp@min@im@threshold@result{1000}
2805   \def\@bp@min@false{false}
2806   \gdef\bp@min@im@threshold@found{false}
2807   \global\let\bp@min@im@thresh@float\relax
2808   \pgfkeys{/pgf/fpu=true}
2809   \pgfmathparse{0}
2810   \global\let\bp@max@im@float=\pgfmathresult
2811   \pgfkeys{/pgf/fpu=false}
2812   \gdef\bp@max@im@value{0}
2813   \foreach \feature/\values in {#1} {
2814     \ifnum\pdf@strcmp{\feature}{z}=0
2815       \foreach \z in \values {
2816         \foreach \y [count=\zcnt] in \z {
2817           \ifnum\zcnt=2
2818             \pgfkeys{/pgf/fpu=true}
2819             \pgfmathparse{abs(\y)}
2820             \let\abs@valuefloat=\pgfmathresult
2821             \pgfmathfloattofixed{\abs@valuefloat}
2822             \edef\abs@value{\pgfmathresult}
2823             \pgfkeys{/pgf/fpu=false}
2824             \ifnum\pdf@strcmp{\abs@value}{0}=0\else
2825               \ifnum\pdf@strcmp{\abs@value}{0.0}=0\else
2826                 \pgfkeys{/pgf/fpu=true}
2827                 \pgfmathparse{\abs@valuefloat > \bp@max@im@float ? 1 : 0}
2828                 \pgfmathfloattoint{\pgfmathresult}
2829                 \pgfkeys{/pgf/fpu=false}
2830                 \ifnum\pgfmathresult=1
2831                   \global\let\bp@max@im@float=\abs@valuefloat
2832                   \xdef\bp@max@im@value{\abs@value}
2833                 \fi
2834                 \ifx\bp@min@im@threshold@found\@bp@min@false
2835                   \xdef\bp@min@im@threshold@result{\abs@value}
2836                   \global\let\bp@min@im@thresh@float=\abs@valuefloat
2837                   \gdef\bp@min@im@threshold@found{true}
2838                 \else
2839                   \pgfkeys{/pgf/fpu=true}
2840                   \pgfmathparse{\abs@valuefloat < \bp@min@im@thresh@float ?
2841                     1 : 0}
2842                   \pgfmathfloattoint{\pgfmathresult}
2843                   \pgfkeys{/pgf/fpu=false}
2844                   \ifnum\pgfmathresult=1
2845                     \xdef\bp@min@im@threshold@result{\abs@value}
2846                     \global\let\bp@min@im@thresh@float=\abs@valuefloat
2847                   \fi
2848                 \fi
2849               \fi
2850             \fi
2851           \fi
2852         }
2853       }
2854     }
2855   }

```

```

2853     }
2854     \fi
2855     \ifnum\pdf@strcmp{\feature}{p}=0
2856     \foreach \p in \values {
2857         \foreach \y [count=\pcnt] in \p {
2858             \ifnum\pcnt=2
2859                 \pgfkeys{/pgf/fpu=true}
2860                 \pgfmathparse{abs(\y)}
2861                 \let\abs@valuefloat=\pgfmathresult
2862                 \pgfmathfloattofixed{\abs@valuefloat}
2863                 \edef\abs@value{\pgfmathresult}
2864                 \pgfkeys{/pgf/fpu=false}
2865                 \ifnum\pdf@strcmp{\abs@value}{0}=0\else
2866                     \ifnum\pdf@strcmp{\abs@value}{0.0}=0\else
2867                         \ifx\bp@min@im@threshold@found\@bp@min@false
2868                             \xdef\bp@min@im@threshold@result{\abs@value}
2869                             \global\let\bp@min@im@thresh@float=\abs@valuefloat
2870                             \gdef\bp@min@im@threshold@found{true}
2871                         \else
2872                             \pgfkeys{/pgf/fpu=true}
2873                             \pgfmathparse{\abs@valuefloat < \bp@min@im@thresh@float ?
2874                                 1 : 0}
2875                             \pgfmathfloattoint{\pgfmathresult}
2876                             \pgfkeys{/pgf/fpu=false}
2877                             \ifnum\pgfmathresult=1
2878                                 \xdef\bp@min@im@threshold@result{\abs@value}
2879                                 \global\let\bp@min@im@thresh@float=\abs@valuefloat
2880                             \fi
2881                         \fi
2882                     \fi
2883                 \fi
2884             \fi
2885         }
2886     }
2887     \fi
2888 }
2889 \ifx\bp@min@im@threshold@found\@bp@min@false
2890     \gdef\bp@min@im@threshold@result{0.01}
2891     \fi
2892     \pgfkeys{/pgf/fpu=true}
2893     \pgfmathparse{log10(\bp@min@im@threshold@result)}
2894     \let\log@result=\pgfmathresult
2895     \pgfmathparse{\log@result + 1e-5}
2896     \let\log@adjusted=\pgfmathresult
2897     \pgfmathparse{floor(\log@adjusted)}
2898     \pgfmathfloattofixed{\pgfmathresult}
2899     \xdef\bp@min@im@pow@10{\pgfmathresult}
2900     \xdef\bp@min@im@pow@10{\bp@min@im@pow@10}
2901     \pgfmathparse{\bp@max@im@float > 0 ?
2902         \bp@max@im@float : \bp@min@im@threshold@result}
2903     \let\bp@max@im@valuefloat=\pgfmathresult
2904     \pgfmathparse{log10(max(\bp@max@im@valuefloat,1e-100))}
2905     \let\log@bp@max@im=\pgfmathresult
2906     \pgfmathparse{\log@bp@max@im + 1e-5}
2907     \let\log@bp@max@im@adjusted=\pgfmathresult
2908     \pgfmathparse{ceil(\log@bp@max@im@adjusted)}
2909     \pgfmathfloattoint{\pgfmathresult}
2910     \xdef\bp@max@im@pow@10{\pgfmathresult}
2911     \pgfkeys{/pgf/fpu=false}
2912 }

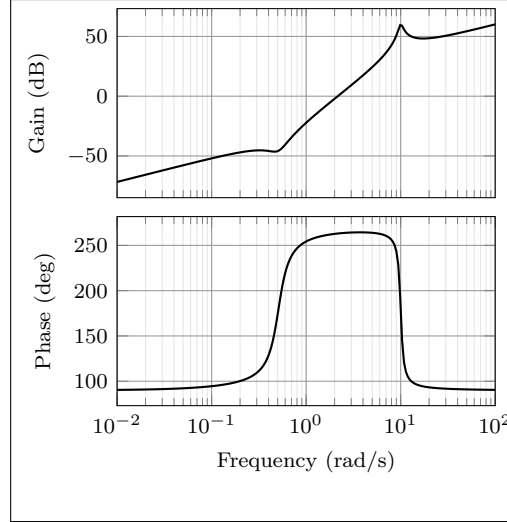
```

A TL;DR for package version 2.1.1 and earlier

All Bode plots in this section are for the transfer function (with and without a transport delay)

$$G(s) = 10 \frac{s(s + 0.1 + 0.5i)(s + 0.1 - 0.5i)}{(s + 0.5 + 10i)(s + 0.5 - 10i)} = \frac{s(10s^2 + 2s + 2.6)}{(s^2 + s + 100.25)}. \quad (6)$$

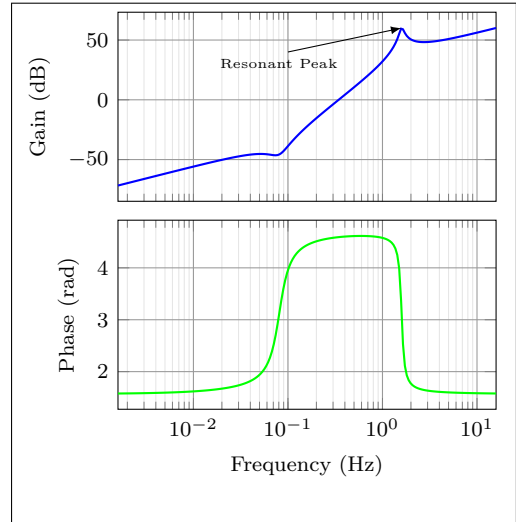
Bode plot in ZPK format



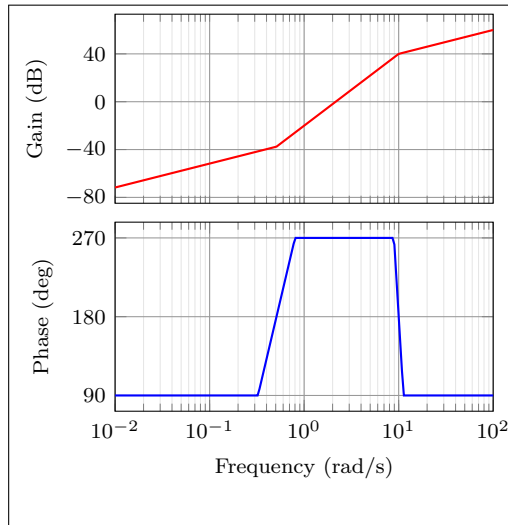
```
\BodeZPK{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10%
}
{0.01}
{100}
```

Same Bode plot over the same frequency range but supplied in Hz, in TF format with arrow decoration, transport delay, unit, and color customization (the phase plot may show wrapping if the **pgf** package option is used)

```
\BodeTF[%
  samples=1000,
  plot/mag/{blue,thick},
  plot/ph/{green,thick},
  tikz/{%
    >=latex,
    phase unit=rad,
    frequency unit=Hz%
  },
  commands/mag/{
    \draw[->](axis cs:0.1,40) -- (axis cs:{10/(2*pi)},60);
    \node at (axis cs: 0.08,30) {\tiny Resonant Peak};
  }%
]
{%
  num/{10,2,2.6,0},
  den/{1,1,100.25}%
}
{0.01/(2*pi)}
{100/(2*pi)}
```



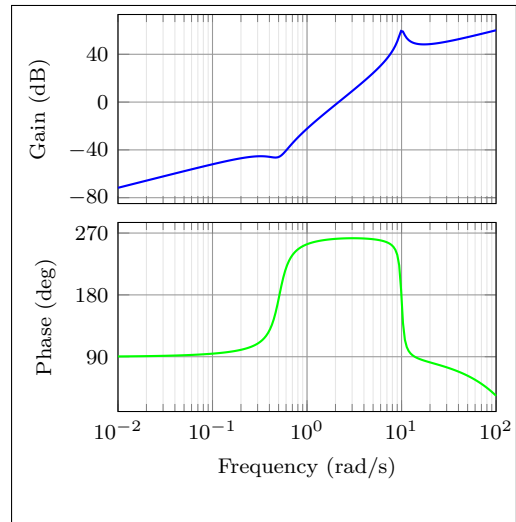
Linear approximation with customization



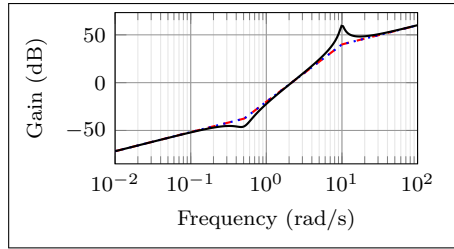
```
\BodeZPK[%
  plot/mag/{red,thick},
  plot/ph/{blue,thick},
  axes/mag/{ytick distance=40},
  axes/ph/{ytick distance=90},
  approx/linear%
]{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10%
}
{0.01}
{100}
```

Plot with delay and customization

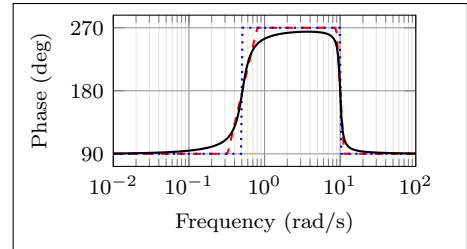
```
\BodeZPK[%
  plot/mag/{blue,thick},
  plot/ph/{green,thick},
  axes/mag/{ytick distance=40},
  axes/ph/{ytick distance=90%
]{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10,
  d/0.01%
}
{0.01}
{100}
```



Individual gain and phase plots with more customization



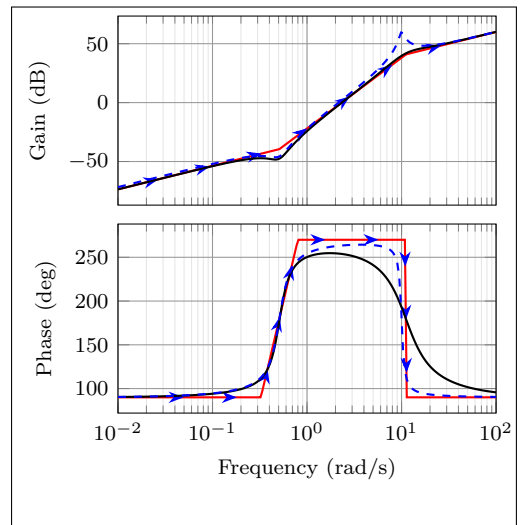
```
\begin{BodeMagPlot}{%
  axes/{height=2cm,
    width=4cm}%
}
{0.01}
{100}
\addBodeZPKPlots[%
  true/{black,thick},
  linear/{red,dashed,thick},
  asymptotic/{blue,dotted,thick}%
]
{magnitude}
{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10%
}
}\end{BodeMagPlot}
```



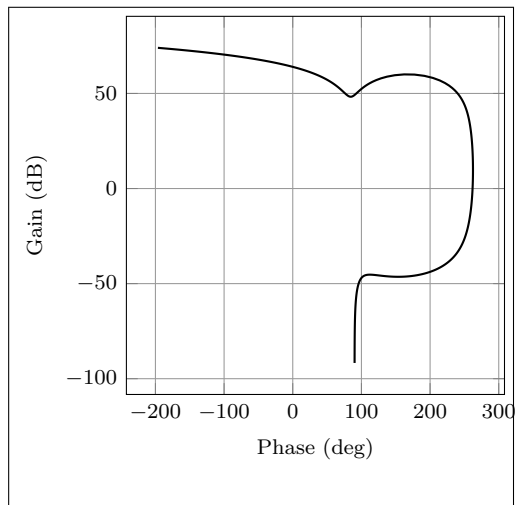
```
\begin{BodePhPlot}{%
  height=2cm,
  width=4cm,
  ytick distance=90
}
{0.01}
{100}
\addBodeZPKPlots[%
  true/{black,thick},
  linear/{red,dashed,thick},
  asymptotic/{blue,dotted,thick}%
]
{phase}
{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10%
}
}\end{BodePhPlot}
```

Multiple transfer functions in a single Bode plot using the **BodePlot** environment and the **\addBodePlot** macro introduced in v2.1.

```
\begin{BodePlot}{0.01}{100}
\addBodePlot[red,postaction=decorate,
  decoration={%
    markings,
    mark=between positions 0.1 and 0.9 step 2em with {%
      \arrow{Stealth [length=2mm, blue]}
    }
  },linear]{zpk}{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-5,-10},{-5,10}},
  k/10%
}
\addBodePlot[black,thick]{zpk}{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-5,-10},{-5,10}},
  k/10%
}
\addBodePlot[blue,dashed]{tf}{%
  num/{10,2,2.6,0},
  den/{1,1,100.25}%
}
}\end{BodePlot}
```



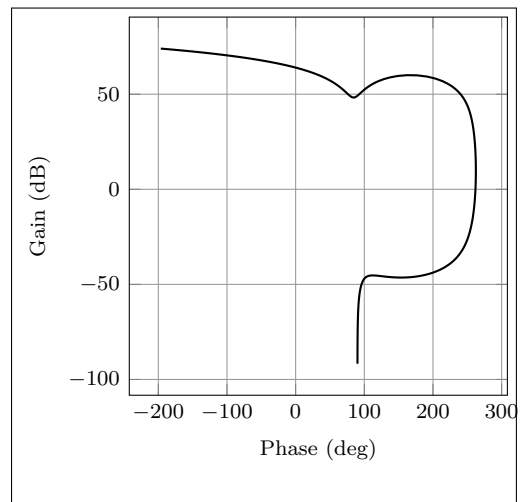
Nichols chart



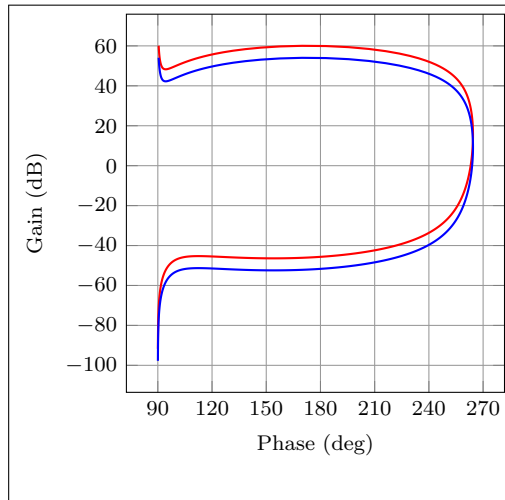
```
\NicholsZPK[samples=1000]
{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10,
  d/0.01%
}
{0.001}
{500}
```

Same Nichols chart in TF format (may show wrapping in **pgf** mode)

```
\NicholsTF[samples=1000]
{%
  num/{10,2,2.6,0},
  den/{1,1,100.25},
  d/0.01%
}
{0.001}
{500}
```



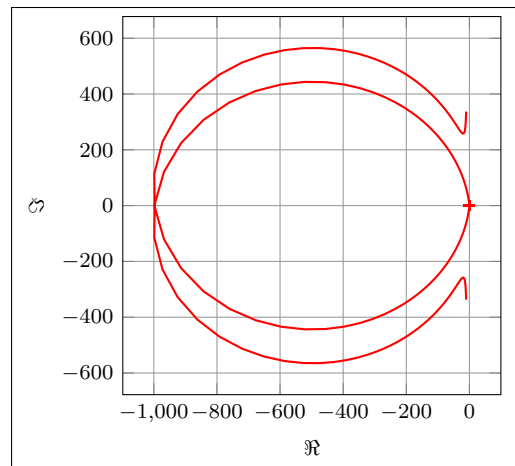
Multiple Nichols charts with customization



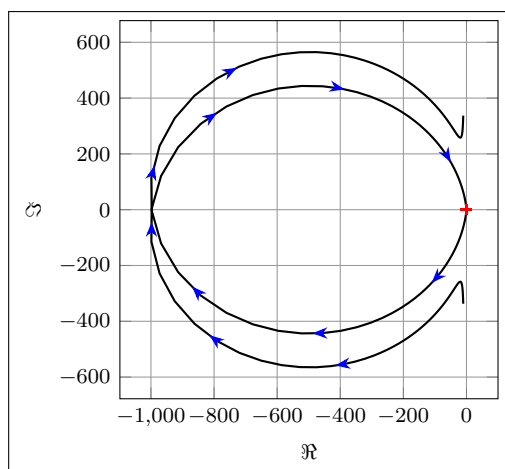
```
\begin{NicholsChart}{%
ytick distance=20,
xtick distance=30
}
{0.001}
{100}
\addNicholsZPKChart [red,samples=1000] {%
z/{0,{-0.1,-0.5},{-0.1,0.5}},
p/{{-0.5,-10},{-0.5,10}},
k/10%
}
\addNicholsZPKChart [blue,samples=1000] {%
z/{0,{-0.1,-0.5},{-0.1,0.5}},
p/{{-0.5,-10},{-0.5,10}},
k/5%
}
}\end{NicholsChart}
```

Nyquist plot

```
\NyquistZPK[plot/{red,thick,samples=1000}]
{%
z/{0,{-0.1,-0.5},{-0.1,0.5}},
p/{{-0.5,-10},{-0.5,10}},
k/10%
}
{-30}
{30}
```



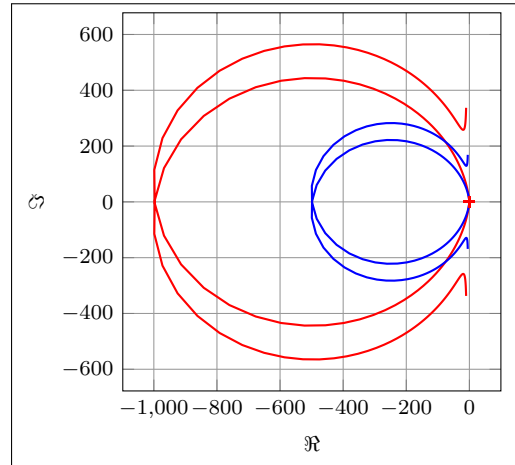
Nyquist plot in TF format with arrows



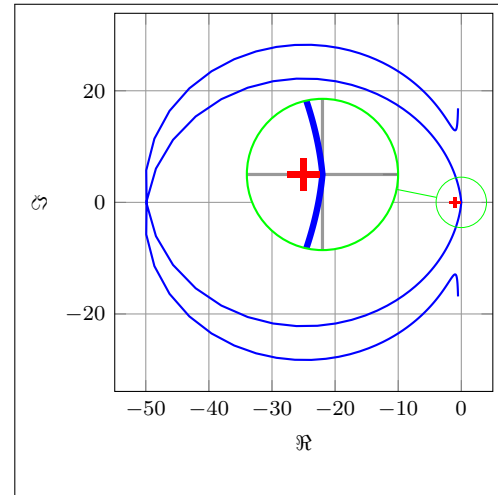
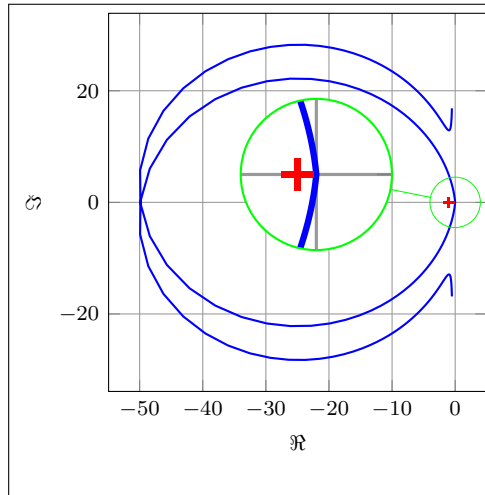
```
\NyquistTF[%
plot/{%
samples=1000,
postaction=decorate,
decoration={%
markings,
mark=between positions 0.1 and 0.9 step 5em with {%
\arrow{Stealth [length=2mm, blue]}
}
}%
}%
}%
num/{10,2,2.6,0},
den/{1,1,100.25}%
}
{-30}
{30}
```

Multiple Nyquist plots with customization

```
\begin{NyquistPlot}{-30}{30}
\addNyquistZPKPlot [red,samples=1000] {%
z/{0,{-0.1,-0.5},{-0.1,0.5}},
p/{{-0.5,-10},{-0.5,10}},
k/10%
}
\addNyquistZPKPlot [blue,samples=1000] {%
z/{0,{-0.1,-0.5},{-0.1,0.5}},
p/{{-0.5,-10},{-0.5,10}},
k/5%
}
\end{NyquistPlot}
```



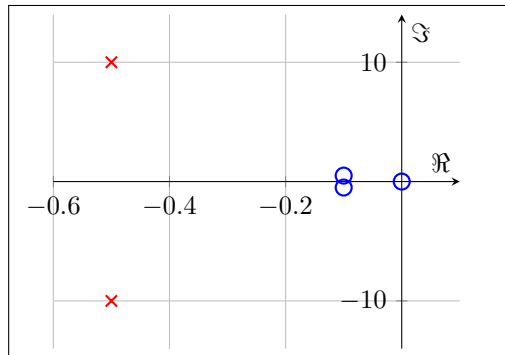
Nyquist plots with additional commands, using two different macros



```
\begin{NyquistPlot}{%
tikz/{
spy using outlines={%
circle,
magnification=3,
connect spies,
size=2cm
}
}%
}
\addNyquistZPKPlot [blue,samples=1000] {%
z/{0,{-0.1,-0.5},{-0.1,0.5}},
p/{{-0.5,-10},{-0.5,10}},
k/0.5%
}
\coordinate (spyon) at (axis cs:0,0);
\coordinate (spyat) at (axis cs:-22,5);
\spy [green] on (spyon) in
node [fill=white] at (spyat);
\end{NyquistPlot}
```

```
\NyquistZPK[%
plot/{blue,samples=1000},
tikz/{
spy using outlines={%
circle,
magnification=3,
connect spies,
size=2cm
}
},
commands/{
\coordinate (spyon) at (axis cs:0,0);
\coordinate (spyat) at (axis cs:-22,5);
\spy [green] on (spyon) in
node [fill=white] at (spyat);
}%
}
}%
{-30}
{30}
```

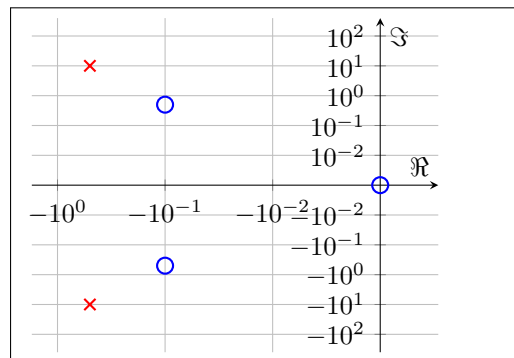
Pole-zero map



```
\PoleZeroMapZPK
{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10%
}
```

Pole-zero map (symmetric log scale)

```
\PoleZeroMapZPK[scale/{log}]
{%
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10%
}
```



Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in **roman** refer to the code lines where the entry is used.

| | | |
|---|---|----------------------------------|
| Symbols | 2662, 2666, 2670, | \bp@axes 154, |
| \bodeplot <u>230</u> | 2674, 2679, 2683, | 157, 158, 160, 221, |
| \bodeplot/add <u>556</u> | 2688, 2703, 2704, | 222, 224, 1188, |
| \bodeplot/combinedenv | 2716, 2720, 2725, | 1227, 1918, 1923, |
| <u>383</u> | 2729, 2734, 2738, | 1932, 1961, 1983, |
| \bodeplot/env <u>135</u> | 2743, 2820, 2821, | 2023, 2045, 2154, |
| \bodeplot/nichols . . <u>689</u> | 2827, 2831, 2836, | 2162, 2170, 2202, |
| \bodeplot/nyquist . . <u>636</u> | 2840, 2846, 2861, | 2224, 2284, 2306, |
| \bodeplot/pzmap . . . <u>742</u> | 2862, 2869, 2873, 2879 | 2374, 2439, 2522, 2533 |
| \bodeplot/tf <u>541</u> | \addBodeComponentPlot | \bp@checkfull |
| \bodeplot/zpk <u>523</u> | <u>1142</u> | . . . 208, 210, 353, |
| \@babelfalse 54 | \addBodePlot <u>1234</u> | 355, 493, 495, 614, |
| \@babeltrue 57 | \addBodeTFPlot <u>1093</u> , <u>1139</u> | 616, 667, 669, 720, 722 |
| \@bp@min@false | \addBodeTFPlots . . . <u>1134</u> | \bp@checkshort |
| 2454, 2461, | \addBodeZPKPlots . . . | . . . 209, 210, 354, |
| 2619, 2677, 2732, | <u>1057</u> , <u>1137</u> | 355, 494, 495, 615, |
| 2753, 2766, 2777, | \addNicholsTFChart . <u>2406</u> | 616, 668, 669, 721, 722 |
| 2805, 2834, 2867, 2889 | \addNicholsZPKChart <u>2380</u> | \bp@cmd 998, 1000, 1004, |
| \@declutterargfalse . . 5 | \addNyquistTFPlot . . <u>2103</u> | 1014, 1078, 1081, |
| \@declutterargtrue . . . 7 | \addNyquistZPKPlot . <u>2081</u> | 1086, 1088, 1109, |
| \@hzargfalse 13 | | 1113, 1115, 1118, |
| \@hzargtrue 15 | B | 1123, 1127, 1130, |
| \@pgfargfalse 1 | BodeMagPlot (env.) . . <u>1195</u> | 1144, 1147, 1151, |
| \@pgfargtrue 3 | BodePhPlot (env.) . . . <u>1156</u> | 1153, 1177, 1179, |
| \@radargfalse 9 | BodePlot (env.) <u>1293</u> | 1180, 1190, 1216, |
| \@radargtrue 11 | \BodeTF <u>953</u> | 1218, 1219, 1229, |
| __bp_collect_item:n | \BodeTF@Shorthandoff | 1310, 1312, 1313, |
| 1486, 1504 | 1049, 1051 | 1323, 1973, 1974, |
| __bp_find_separator: | \BodeZPK . . . 865, 958, 960 | 1976, 1985, 1987, |
| 1387, 1477 | \BodeZPK@Shorthandoff | 1991, 1997, 2035, |
| __bp_parse_complex_with_i: | 945, 947 | 2036, 2038, 2047, |
| 1364, 1381 | \bool_if:NTF 1389 | 2049, 2053, 2059, |
| __bp_parse_pure_imaginary: | \bool_lazy_or:nnT . . 1461 | 2088, 2091, 2095, |
| 1391, 1395 | \bool_new:N 1353 | 2097, 2110, 2113, |
| __bp_parse_real_only: | \bool_set_false:N . . 1479 | 2117, 2119, 2146, |
| 1365, 1373 | \bool_set_true:N . . 1465 | 2147, 2148, 2155, |
| __bp_process_item:n | \bp@add . . . <u>1566</u> , 1588, | 2214, 2215, 2217, |
| 1449, 1482 | 1589, 1592, 1593, | 2226, 2227, 2231, |
| __bp_set_signed_value:NN | 1595, 1596, 1606, | 2235, 2296, 2297, |
| 1398, 1402 | 1607, 1610, 1611, | 2299, 2308, 2309, |
| __bp_split_at_separator: | 1613, 1614, 1622, | 2313, 2317, 2366, |
| 1390, 1498 | 1623, 1626, 1627, | 2367, 2368, 2375, |
| __bp_unwrap_braces:N | 1629, 1630, 1648, 1649 | 2387, 2390, 2413, |
| 1375, 1414, 1425, | \bp@add@0 . . . 560, 565, | 2416, 2500, 2502, |
| 1430, 1438, 1506, 1507 | 566, 568, 571, 572, | 2504, 2525, 2536, |
| | 574, 581, 582, 584, | 2552, 2555, 2572, 2575 |
| | 587, 588, 590, 597, | \bp@commands |
| A | 598, 600, 603, 604, | 1962, 1994, 2001, |
| \abs@value | 606, 627, 628, 630, | 2024, 2056, 2063, |
| 2642, 2644, 2645, | 1063, 1068, 1240, | 2164, 2176, 2203, |
| 2675, 2678, 2687, | 1538, 1543, 1547, 1562 | 2232, 2246, 2285, |
| 2705, 2707, 2708, | \bp@add@tmp 1539, | 2314, 2330, 2441, 2579 |
| 2733, 2742, 2822, | 1551, 1554, 1561, 1563 | \bp@complex 1341, |
| 2824, 2825, 2832, | \bp@approx 872, | 1369, 1679, 1688 |
| 2835, 2845, 2863, | 900, 976, 1247, | \bp@contains@equal . . |
| 2865, 2866, 2868, 2878 | 1254, 1298, 1854, | 1059, 1095, |
| \abs@valuefloat 2640, | 1888, 1941, 1945, 1948 | <u>1530</u> , 1732, 2083, |
| 2641, 2653, 2657, | | |

| | | |
|--------------------------------|---------------------------------------|----------------------------------|
| 2105, 2382, 2408, 2436 | 212, 350, 352, 353, | 2410, 2411, 2416, 2425 |
| \bp@contains@num ... | 357, 490, 492, 493, | \bp@mag@axes |
| 1241, <u>1533</u> | 497, 611, 613, 614, | ... 405, 409, 410, |
| \bp@d 1692, 1693, 1701, 1714 | 618, 664, 666, 667, | 412, 422, 423, 425, |
| \bp@den .. 1700, 1709, 1710 | 671, 717, 719, 720, 724 | 506, 507, 509, 869, |
| \bp@den@deg .. 1806, 1811 | \bp@gnu@cmd . 929, 934, | 913, 973, 1016, |
| \bp@den@im 1775, | 1032, 1037, 1088, | 1327, 1849, 1874, 1879 |
| 1816, 1831, 1834 | 1127, 1130, 1153, | \bp@mag@cmd |
| \bp@den@list | 1279, 1290, 1657, 1837 | 1015, 1024, 1032, |
| ... 1708, 1711, 1714 | \bp@gnu@plot 928, 933, | 1266, 1278, 1308, 1332 |
| \bp@den@re 1774, | 1031, 1085, 1129, | \bp@mag@commands ... |
| 1813, 1819, 1832, 1834 | 1152, 1277, 1283, <u>1656</u> | 873, 922, |
| \bp@domain@end | \bp@gnu@prefix <u>1</u> | 930, 977, 1025, |
| 153, 171, | \bp@gnu@unwrap@plot . | 1033, 1299, 1856, 1894 |
| 243, 316, 394, 456, | 1036, | \bp@mag@plot ... 254, |
| 646, 661, 699, 714, | 1126, 1286, <u>1836</u> | 258, 259, 261, 271, |
| 892, 904, 905, 996, | \bp@gnuplot@id <u>1</u> | 272, 274, 366, 367, |
| 1007, 1008, 1175, | \bp@group 871, | 369, 915, 1018, |
| 1185, 1186, 1214, | 909, 975, 1012, | 1852, 1862, 1867, 1904 |
| 1223, 1224, 1306, | 1297, 1321, 1853, 1885 | \bp@max@im@float 2810, |
| 1316, 1317, 1970, | \bp@has@negative@values | 2827, 2831, 2901, 2902 |
| 1978, 2032, 2040, | 2461, | \bp@max@im@pow@ ... 2910 |
| 2144, 2151, 2211, | 2632, 2660, 2723, 2777 | \bp@max@im@value ... |
| 2220, 2240, 2293, | \bp@has@positive@values | 2812, 2832 |
| 2302, 2322, 2364, 2371 | 2454, | \bp@max@im@valuefloat |
| \bp@domain@start 152, | 2631, 2651, 2714, 2766 | 2903, 2904 |
| 170, 242, 315, 393, | \bp@imagpart . 1343, 1368 | \bp@max@re@float ... |
| 455, 645, 661, 698, | \bp@k 1691, 1693 | ... 2624, 2670, 2674 |
| 714, 891, 903, 905, | \bp@Last@Loop@p 2567, 2569 | \bp@max@re@neg@float |
| 995, 1006, 1008, | \bp@Last@Loop@z 2547, 2549 | 2628, 2662, 2666, |
| 1174, 1184, 1213, | \bp@legacy | 2725, 2729, 2780, 2789 |
| 1222, 1305, 1315, | ... 890, 900, 994, | \bp@max@re@neg@pow@ . |
| 1317, 1969, 1978, | 1003, 1066, 1073, | ... 2464, 2778, 2786 |
| 2031, 2040, 2143, | 1075, 1098, 1103, | \bp@max@re@pos@float |
| 2210, 2220, 2240, | 1105, 1243, 1247, | 2626, |
| 2292, 2302, 2322, 2363 | 1693, 1714, 1968, | 2653, 2657, 2716, |
| \bp@domain@start: ... | 1975, 2030, 2037, | 2720, 2769, 2788, 2789 |
| 1186, | 2084, 2086, 2106, | \bp@max@re@pos@pow@ . |
| 1224, 2151, 2371 | 2108, 2209, 2216, | ... 2457, 2767, 2775 |
| \bp@fallback | 2291, 2298, 2383, | \bp@max@re@pow@ ... 2800 |
| 962, 964, 967, 969, | 2385, 2409, 2411, | \bp@max@re@value ... |
| 2273, 2275, 2278, 2280 | 2448, 2451, 2452, 2537 | 2630, 2675 |
| \bp@fix@add@opt | \bp@list 1581, | \bp@max@re@valuefloat |
| ... 1063, 1065, <u>1537</u> | 1582, 1777, 1778 | 2790, |
| \bp@freq@filter <u>83</u> | \bp@loop@delay 1776, 1827, 1832 | 2791, 2792, 2793, 2794 |
| \bp@freq@label <u>83</u> | \bp@mag . 898, 900, 921, | \bp@min@Im@pow@ ... 2900 |
| \bp@freq@scale | 928, 1001, 1003, | \bp@min@im@pow@ 2489, |
| 83, 191, 198, 336, | 1024, 1031, 1235, | 2520, 2521, 2899, 2900 |
| 343, 476, 483, 905, | 1243, 1247, 1254, | \bp@min@im@thresh@float |
| 1008, 1079, 1085, | 1258, 1269, 1277, | 2807, 2836, 2840, |
| 1110, 1116, 1126, | 1972, 1975, 1992, | 2846, 2869, 2873, 2879 |
| 1129, 1145, 1152, | 1993, 1998, 1999, | \bp@min@im@threshold@found |
| 1267, 1271, 1317, | 2034, 2037, 2054, | ... 2806, 2834, |
| 1665, 1845, 1978, | 2055, 2060, 2061, | 2837, 2867, 2870, 2889 |
| 2040, 2089, 2096, | 2085, 2086, 2091, | \bp@min@im@threshold@result |
| 2111, 2118, 2220, | 2092, 2098, 2099, | ... 2519, 2804, |
| 2240, 2302, 2322, | 2107, 2108, 2113, | 2835, 2845, 2868, |
| 2388, 2398, 2414, 2424 | 2114, 2120, 2121, | 2878, 2890, 2893, 2902 |
| \bp@freq@scale: | 2213, 2216, 2231, | \bp@min@im@ZPK <u>2452, 2803</u> |
| 905, 1008, | 2241, 2295, 2298, | \bp@min@pow@ 2765 |
| 1317, 1978, 2040, | 2313, 2323, 2384, | \bp@min@re@pow@ |
| 2220, 2240, 2302, 2322 | 2385, 2390, 2399, | 2457, 2464, |
| \bp@full 205, 207, 208, | | |

| | | |
|------------------------------|------------------------------|----------------------------|
| 2475, 2517, 2518, | 2213, 2216, 2231, | \bp@scale 2167, |
| 2764, 2765, 2767, 2778 | 2241, 2295, 2298, | 2182, 2444, 2450, 2503 |
| \bp@min@re@threshold@result | 2313, 2323, 2384, | \bp@short 207, |
| 2516, 2618, 2678, | 2385, 2390, 2399, | 209, 212, 215, 217, |
| 2687, 2733, 2742, | 2410, 2411, 2416, 2425 | 352, 354, 357, 360, |
| 2754, 2756, 2758, 2792 | \bp@ph@axes | 362, 492, 494, 497, |
| \bp@min@real@ZPK . . . | . . . 406, 414, 415, | 500, 502, 613, 615, |
| 2451, 2617 | 417, 430, 431, 433, | 618, 621, 623, 666, |
| \bp@min@thresh@float | 514, 515, 517, 870, | 668, 671, 674, 676, |
| 2621, 2679, 2683, | 917, 974, 1020, | 719, 721, 724, 727, 729 |
| 2688, 2734, 2738, 2743 | 1330, 1850, 1877, 1880 | \bp@short@list |
| \bp@min@threshold@found | \bp@ph@cmd | . . . 54, 944, 949, |
| 2620, 2677, | 1019, 1026, 1037, | 1048, 1053, 1159, |
| 2680, 2732, 2735, 2753 | 1270, 1289, 1309, 1334 | 1162, 1198, 1201, |
| \bp@min@threshold@result | \bp@ph@commands | 2010, 2015, 2072, |
| 2756 | 874, 924, | 2077, 2128, 2131, |
| \bp@mode . . 1244, 1248, | 935, 978, 1027, | 2255, 2260, 2340, |
| 1253, 1257, 1282, 1285 | 1038, 1300, 1855, 1892 | 2345, 2348, 2351, |
| \bp@new . 215, 217, 220, | \bp@ph@plot 255, | 2587, 2592, 2608, 2613 |
| 222, 225, 360, 362, | 263, 264, 266, 280, | \bp@spec@arg |
| 365, 367, 370, 373, | 281, 283, 374, 375, | 1058, 1059, |
| 375, 378, 500, 502, | 377, 919, 1022, | 1062, 1066, 1094, |
| 505, 507, 510, 513, | 1851, 1865, 1868, 1906 | 1095, 1096, 1098, |
| 515, 518, 621, 623, | \bp@ph@scale | 2082, 2083, 2084, |
| 626, 628, 631, 674, | . . . 83, 178, 183, | 2104, 2105, 2106, |
| 676, 679, 681, 684, | 323, 328, 463, 468, | 2381, 2382, 2383, |
| 727, 729, 732, 734, 737 | 772, 776, 787, 800, | 2407, 2408, 2409, |
| \bp@num . . 1699, 1703, 1704 | 803, 815, 821, 822, | 2435, 2436, 2445, 2448 |
| \bp@num@deg . . 1783, 1788 | 842, 1026, 1832, | \bp@style 68 |
| \bp@num@im 1773, | 1843, 1845, 1992, | \bp@tf@new@to@legacy |
| 1793, 1831, 1833 | 1993, 1998, 1999, | 981, |
| \bp@num@list | 2054, 2055, 2060, | 1096, 1242, 1695, |
| . . . 1702, 1705, 1714 | 2061, 2091, 2092, | 2027, 2106, 2288, 2409 |
| \bp@num@re 1772, | 2098, 2099, 2113, | \bp@TF@plot |
| 1790, 1796, 1831, 1833 | 2114, 2120, 2121, | 1003, 1103, 1105, |
| \bp@p 1682, 1683, 1685, 1686 | 2241, 2244, 2313, | 1243, 1258, 1771, |
| \bp@p@list 1684, 1688, 1693 | 2323, 2328, 2399, | 2037, 2108, 2298, 2411 |
| \bp@parse@add@Bode@opt | 2402, 2416, 2425, 2430 | \bp@tf@to@zpk . . 955, |
| . . . 1240, 1251, 1939 | \bp@ph@x@label 83 | 1135, 1716, 2266, 2597 |
| \bp@parse@complex . . . | \bp@ph@y@label 83 | \bp@tf@fnl@spec 1697, 1698 |
| . . . 1338, 1678, 1687 | \bp@pim . . 2565, 2570, 2575 | \bp@tikz 875, 896, |
| \bp@parse@env@opt . . . | \bp@plot 647, 652, | 979, 999, 1170, |
| 1173, | 680, 681, 683, 700, | 1178, 1209, 1217, |
| 1212, 1917, 2142, 2362 | 705, 733, 734, 736, | 1301, 1311, 1857, |
| \bp@parse@N@opt | 1070, 1073, 1075, | 1898, 1919, 1926, |
| 1967, 2029, | 1081, 1085, 1100, | 1963, 1974, 2025, |
| 2161, 2208, 2290, 2447 | 1103, 1105, 1113, | 2036, 2139, 2147, |
| \bp@parse@opt | 1118, 1126, 1129, | 2165, 2179, 2204, |
| 889, 993, 1304, 1848 | 1269, 1273, 1279, | 2215, 2286, 2297, |
| \bp@ph | 1290, 1940, 1950, | 2359, 2367, 2442, 2501 |
| 899, 900, 923, 933, | 1989, 2051, 2163, | \bp@tmp . 895, 897, 901, |
| 1002, 1003, 1026, | 2173, 2188, 2229, | 911, 1071, 1073, |
| 1036, 1236, 1243, | 2311, 2440, 2554, 2574 | 1075, 1101, 1103, 1105 |
| 1247, 1254, 1258, | \bp@pre 2562, 2575 | \bp@tmp@mag 912, 921, 929 |
| 1273, 1283, 1286, | \bp@prefix 27, 45, 1657, | \bp@tmp@ph . 916, 923, 934 |
| 1972, 1975, 1992, | 1664, 1837, 1844, | \bp@tmp@status . 955, |
| 1993, 1998, 1999, | 2243, 2327, 2401, 2429 | 956, 1135, 1136, |
| 2034, 2037, 2054, | \bp@PZZPK@ticksXNeg . | 2266, 2267, 2597, 2598 |
| 2055, 2060, 2061, | 2462, 2466 | \bp@tmp@zpk |
| 2085, 2086, 2091, | \bp@PZZPK@ticksXPos . | . . . 955, 958, 960, |
| 2092, 2098, 2099, | 2455, 2459 | 1135, 1137, 2266, |
| 2107, 2108, 2113, | \bp@realpart . 1342, 1367 | 2269, 2271, 2597, 2599 |
| 2114, 2120, 2121, | | |

| | | |
|------------------------------|-----------------------------|-----------------------------|
| \bp@tztpk@ed 1738, | 574, 582, 584, 588, | 1466, 1473, 1480, |
| 1741, 1751, 1765 | 590, 598, 600, 604, 606 | 1488, 1491, 1494, 1502 |
| \bp@tztpk@den .. 1737, | \gnu@id | \l__bp_sep_tl .. 1349, |
| 1741, 1748, 1761 | 927, 928, 932, 933, | 1467, 1515, 1520 |
| \bp@tztpk@feat . 1743, | 1030, 1031, 1035, | \l__bp_sep_pos_int ... |
| 1744, 1747, 1750 | 1036, 1084, 1085, | 1351, |
| \bp@tztpk@file | 1122, 1126, 1129, | 1466, 1488, 1491 |
| 1718, 1719, | 1150, 1152, 1276, | \l_imag_tl 1340, 1368, |
| 1720, 1721, 1722, | 1277, 1281, 1283, 1286 | 1369, 1377, 1398, |
| 1723, 1724, 1725, 1726 | | 1516, 1517, 1521, 1522 |
| \bp@tztpk@list 1742, 1743 | I | \l_real_tl |
| \bp@tztpk@num .. 1736, | \if@babel 54, 941, | 1339, 1367, 1369, |
| 1741, 1745, 1761 | 1045, 1157, 1196, | 1376, 1397, 1510, 1511 |
| \bp@tztpk@outfile .. 1716 | 2007, 2069, 2126, | \l_tmpa_tl 1407, 1408, 1419 |
| \bp@tztpk@val .. 1743, | 2252, 2337, 2584, 2605 | \l_tmpb_tl 1410, |
| 1745, 1748, 1751 | \if@declutterarg ... | 1411, 1414, 1415, |
| \bp@user@prefix . 28, | 5, 29, 35, 49 | 1421, 1422, 1425, 1426 |
| 36, 38, 876, 980, | \if@hzarg 13, 103, | \l_tmpc_tl ... 1442, 1443 |
| 1171, 1210, 1302, | 142, 244, 395, 884, 988 | \Last@LoopValue 1574, 1576 |
| 1858, 1901, 1903, | \if@pgfarg 1, | \log@adjusted .. 2761, |
| 1920, 1929, 1931, | 21, 106, 193, 200, | 2762, 2896, 2897 |
| 1964, 2026, 2140, | 338, 345, 478, 485, | \log@bp@max@im 2905, 2906 |
| 2166, 2185, 2187, | 920, 954, 1023, | \log@bp@max@im@adjusted |
| 2205, 2287, 2360, 2443 | 1077, 1107, 1143, | 2907, 2908 |
| \bp@z 1673, 1674, 1676, 1677 | 1265, 1990, 2052, | \log@bp@max@re 2795, 2796 |
| \bp@z@list 1675, 1679, 1693 | 2087, 2109, 2230, | \log@bp@max@re@adjusted |
| \bp@zim .. 2545, 2550, 2555 | 2265, 2312, 2386, 2412 | 2797, 2798 |
| \bp@zpk@new@to@legacy | \if@radarg 9, 89, | \log@bp@max@re@neg .. |
| 877, 1062, | 147, 249, 400, 879, 983 | 2781, 2782 |
| 1246, 1668, 1965, | \IfFileExists 1763 | \log@bp@max@re@neg@adjusted |
| 2084, 2206, 2383, 2445 | \input 1764 | 2783, 2784 |
| \bp@ZPK@plot | \int_compare:nNf .. 1491 | \log@bp@max@re@pos .. |
| . 900, 1073, 1075, | \int_compare:nNT ... | 2770, 2771 |
| 1247, 1254, 1580, | ... 1443, 1457, 1459 | \log@bp@max@re@pos@adjusted |
| 1975, 2086, 2216, 2385 | \int_compare:nNTF . 1488 | 2772, 2773 |
| \bp@zpknl@spec 1671, 1672 | \int_decr:N 1455 | \log@result 2759, |
| \bp@zre 2542, 2555 | \int_incr:N | 2760, 2894, 2895 |
| \bp_if_contains:nnTF | ... 1454, 1473, 1494 | \lowercase 62 |
| ... 1525, 1531, 1534 | \int_new:N 1350, 1351, 1352 | |
| C | \int_set_eq:NN 1466 | M |
| \closeout 1726 | \int_zero:N | \MagCSPoles 810 |
| \cs_new_protected:Npn | ... 1480, 1481, 1502 | \MagCSPolesAsymp ... 810 |
| 1373, 1381, 1395, | L | \MagCSPolesLin 810 |
| 1402, 1438, 1449, | \l__bode_check_tl ... | \MagCSPolesPeak 829 |
| 1477, 1486, 1498, 1526 | ... 1525, 1527, 1528 | \MagCSZeros 810 |
| \currentdegree . 1788, | \l__bp_after_tl | \MagCSZerosAsymp ... 810 |
| 1789, 1792, 1793, | 1348, 1492, 1501, | \MagCSZerosLin 810 |
| 1794, 1796, 1797, | 1507, 1513, 1521, 1522 | \MagCSZerosPeak 829 |
| 1811, 1812, 1815, | \l__bp_before_tl ... | \MagDel 775, 1649 |
| 1816, 1817, 1819, 1820 | 1347, 1489, | \MagK 769, 1630 |
| D | 1500, 1506, 1509, 1511 | \MagKAsymp 769, 1627 |
| \dospecials 65 | \l__bp_found_bool ... | \MagKLin 769, 1623 |
| E | 1353, | \MagPole ... 777, 804, 1614 |
| environments: | 1389, 1465, 1479 | \MagPoleAsymp |
| BodeMagPlot 1195 | \l__bp_input_tl | 777, 806, 1611 |
| BodePhPlot 1156 | 1346, 1359, 1360, | \MagPoleLin 777, 805, 1607 |
| BodePlot 1293 | 1363, 1375, 1376, | \MagSOPoles 837 |
| \exp_after:wN 1444 | 1384, 1398, 1482, 1504 | \MagSOPolesAsymp ... 837 |
| G | \l__bp_paren_int 1352, | \MagSOPolesLin 837 |
| \g@addto@macro | 1454, 1455, 1459, 1481 | \MagSOPolesPeak 855 |
| 62, 566, 568, 572, | \l__bp_pos_int | \MagSOPolesPeak 837 |
| | 1350, 1457, | \MagSOPolesLin 837 |

Change History

| | | | |
|--|----|---|----|
| v1.0 | | \BodeTF: Fixed phase wrapping in gnuplot mode | 42 |
| General: Initial release | 1 | | |
| v1.0.1 | | v1.1.1 | |
| \addBodeZPKPlots: Improved optional argument handling. | 43 | General: Enable Hz and rad units . . . | 1 |
| \BodeZPK: Pass arbitrary TikZ commands as options. | 40 | \addBodeComponentPlot: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 45 |
| v1.0.2 | | \addBodeTFPlot: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 44 |
| bp@gnu@prefix: Fixed issue #1 | 25 | \addBodeZPKPlots: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 43 |
| v1.0.3 | | \addNicholsTFChart: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 68 |
| BodePlot: Added tikz option to environments | 48 | \addNyquistTFPlot: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 63 |
| \BodeTF: Added Tikz option | 42 | \addNyquistZPKPlot: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 62 |
| \bp@parse@env@opt: Added tikz option to environments | 59 | BodeMagPlot: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 46 |
| \bp@parse@N@opt: Added commands and tikz options | 64 | BodePhPlot: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 45 |
| \bp@parse@opt: Added Tikz option . . | 58 | BodePlot: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 48 |
| NicholsChart: Added tikz option to environments | 67 | \BodeTF: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 42 |
| \NicholsTF: Added commands and tikz options | 65 | \NyquistTF: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 61 |
| \NicholsZPK: Added commands and tikz options | 64 | \NyquistZPK: Enabled 'Hz' and 'rad' units for frequency and phase, respectively | 60 |
| \NyquistTF: Added commands and tikz options | 61 | bp@ph@y@label: New macros to enable 'Hz' and 'rad' units for frequency and phase, respectively . . | 26 |
| \NyquistZPK: Added commands and tikz options | 60 | | |
| bp@gnu@prefix: Added jobname to gnuplot prefix | 25 | | |
| NyquistPlot: Added tikz option to environments | 63 | | |
| v1.0.4 | | | |
| General: Fixed unintended optional argument macro expansion | 1 | | |
| v1.0.5 | | | |
| \bp@parse@opt: Fixed a bug | 58 | | |
| v1.0.6 | | | |
| General: Fixed issue #3 | 1 | | |
| v1.0.7 | | | |
| General: Removed unnecessary semicolons | 1 | | |
| Updated documentation | 1 | | |
| v1.0.8 | | | |
| General: Added a new class option 'declutter' | 1 | | |
| bp@gnu@prefix: Fixed issue #6 | 25 | | |
| v1.1.0 | | | |
| General: Fixed phase wrapping in gnuplot mode | 1 | | |
| \addBodeTFPlot: Fixed phase wrapping in gnuplot mode | 44 | | |
| BodeMagPlot: Added separate environments for phase and magnitude plots | 46 | | |
| BodePhPlot: Added separate environments for phase and magnitude plots | 45 | | |
| BodePlot: Deprecated BodePlot environment | 48 | | |

| | | | |
|---|----|--|--------|
| <code>\PhS0ZerosLin</code> : Fix scaling bug introduced in v1.1.1 | 39 | <code>\NicholsTF</code> : Added code to handle active characters | 65, 67 |
| <code>NyquistPlot</code> : Defined using the ‘NewEniron’ command from the ‘environ’ package to fix conflicts with externalization | 63 | <code>\NicholsZPK</code> : Added code to handle active characters | 64 |
| v1.1.3 | | <code>\NyquistTF</code> : Added code to handle active characters | 62 |
| <code>\addBodeComponentPlot</code> : Changed implementation to respect user-supplied domain | 45 | <code>\NyquistZPK</code> : Added code to handle active characters | 61 |
| <code>\addBodeTFPlot</code> : Changed implementation to respect user-supplied domain | 44 | <code>NyquistPlot</code> : Defined using the ‘NewDocumentEnvironment’ command from the ‘xparse’ package and added a hook to handle active characters | 63 |
| <code>\addBodeZPKPlots</code> : Changed implementation to respect user-supplied domain | 43 | v1.1.6 | |
| <code>\addNicholsTFChart</code> : Changed implementation to respect user-supplied domain | 68 | <code>\bp@short@list</code> : Detect ‘babel-french’ using ‘frenchbsetup’ | 26 |
| <code>\addNicholsZPKChart</code> : Changed implementation to respect user-supplied domain | 67 | v1.1.7 | |
| <code>\addNyquistTFPlot</code> : Changed implementation to respect user-supplied domain | 63 | General: Detect and turn off shorthands to improve ‘babel’ compatibility | 1 |
| <code>\addNyquistZPKPlot</code> : Changed implementation to respect user-supplied domain | 62 | <code>BodeMagPlot</code> : Use auto-generated list of active characters instead of manually entering them. | 46 |
| v1.1.4 | | <code>BodePhPlot</code> : Use auto-generated list of active characters instead of manually entering them. | 45 |
| <code>\addBodeTFPlot</code> : Changed phase wrapping in pgf mode | 44 | <code>BodePlot</code> : Use auto-generated list of active characters instead of manually entering them | 48 |
| <code>\addNicholsTFChart</code> : Changed phase wrapping in pgf mode | 68 | <code>\BodeTF</code> : Use auto-generated list of shorthands instead of manually specifying them | 43 |
| <code>\BodeTF</code> : Changed phase wrapping in pgf mode | 42 | <code>\BodeZPK</code> : Use auto-generated list of shorthands instead of manually specifying them | 41 |
| <code>bp@gnu@prefix</code> : Changed phase wrapping in pgf mode | 25 | <code>\bp@short@list</code> : Directly detect shorthands instead of detecting the language. | 26 |
| v1.1.5 | | <code>NicholsChart</code> : Use auto-generated list of active characters instead of manually entering them. | 67 |
| <code>BodeMagPlot</code> : Defined using the ‘NewDocumentEnvironment’ command from the ‘xparse’ package and added a hook to handle active characters | 46 | <code>\NicholsTF</code> : Use auto-generated list of shorthands instead of manually specifying them | 65, 67 |
| <code>BodePhPlot</code> : Defined using the ‘NewDocumentEnvironment’ command from the ‘xparse’ package and added a hook to handle active characters | 45 | <code>\NicholsZPK</code> : Use auto-generated list of shorthands instead of manually specifying them | 64 |
| <code>BodePlot</code> : Defined using the ‘NewDocumentEnvironment’ command from the ‘xparse’ package and added a hook to handle active characters | 48 | <code>\NyquistTF</code> : Use auto-generated list of shorthands instead of manually specifying them | 62 |
| <code>\BodeTF</code> : Added code to handle active characters | 43 | <code>\NyquistZPK</code> : Use auto-generated list of shorthands instead of manually specifying them | 61 |
| <code>\BodeZPK</code> : Added code to handle active characters | 41 | <code>NyquistPlot</code> : Use auto-generated list of active characters instead of manually entering them. | 63 |
| <code>NicholsChart</code> : Defined using the ‘NewDocumentEnvironment’ command from the ‘xparse’ package and added a hook to handle active characters | 67 | v1.2 | |
| | | General: Removed global option to process pgf commands in radians | 1 |
| | | <code>bp@gnu@prefix</code> : Removed global option to process pgf commands in radians | 25 |

| | | | |
|--|----|---|----|
| v2.0 | | /bodeplot: Added a pgfkeys-based simplified user interface | 29 |
| General: Added pole zero maps | 1 | /bodeplot/add: Added a pgfkeys-based simplified user interface | 34 |
| Added pole-zero map functionality | 23 | /bodeplot/combinedenv: Added a pgfkeys-based simplified user interface | 31 |
| \PoleZeroMapZPK: Added pole-zero map functionality | 68 | /bodeplot/env: Added a pgfkeys-based simplified user interface | 27 |
| v2.1 | | /bodeplot/nichols: Added a pgfkeys-based simplified user interface | 36 |
| General: Added an environment and macros to add multiple transfer functions to a single Bode plot | 1 | /bodeplot/nyquist: Added a pgfkeys-based simplified user interface | 35 |
| Added unified Bode plot macro supporting both ZPK and TF systems | 16 | /bodeplot/pzmap: Added a pgfkeys-based simplified user interface | 37 |
| Enhanced BodePlot environment with unified plot command collection | 17 | /bodeplot/tf: Added a pgfkeys-based simplified user interface | 34 |
| \addBodePlot: Added unified Bode plot macro supporting both ZPK and TF systems | 47 | /bodeplot/zpk: Added a pgfkeys-based simplified user interface | 33 |
| BodePlot: Enhanced BodePlot environment with unified plot command collection | 48 | \BodeTF: Added a pgfkeys-based simplified user interface | 42 |
| \bp@parse@add@Bode@opt: Added option parser for unified Bode plot macro | 60 | \BodeZPK: Added a pgfkeys-based simplified user interface | 40 |
| v2.1.1 | | Updated to use a pgfkeys-based interface | 40 |
| bp@gnu@prefix: Added a new helper command to allow the user to set per-plot gnuplot file names to address feature request #14 | 25 | \bp@contains@equal: Added interface detector | 52 |
| v3.0 | | \bp@contains@num: Added keyword detector | 52 |
| General: Improved the interface to enable PGF-style options and more natural syntax for complex numbers in zeros and poles | 1 | \bp@fix@add@opt: Added option processor for add commands | 52 |
| \addBodeComponentPlot: Updated to use \bp@prefix for consistency | 45 | \bp@parse@complex: Added complex number parser for new interface | 49 |
| \addBodePlot: Added a pgfkeys-based simplified user interface | 47 | \bp@tf@new@to@legacy: Added converter from new to legacy TF interface | 55 |
| \addBodeTFPlot: Added a pgfkeys-based simplified user interface | 44 | \bp@zpk@new@to@legacy: Added converter from new to legacy ZPK interface | 54 |
| \addBodeZPKPlots: Added a pgfkeys-based simplified user interface | 43 | \bp_if_contains:nnTF: Added substring checker | 52 |
| \addNicholsTFChart: Added a pgfkeys-based simplified user interface | 68 | \NicholsTF: Added a pgfkeys-based simplified user interface | 65 |
| \addNicholsZPKChart: Added a pgfkeys-based simplified user interface | 67 | \NicholsZPK: Added a pgfkeys-based simplified user interface | 64 |
| \addNyquistTFPlot: Added a pgfkeys-based simplified user interface | 63 | \NyquistTF: Added a pgfkeys-based simplified user interface | 61 |
| \addNyquistZPKPlot: Added a pgfkeys-based simplified user interface | 62 | \NyquistZPK: Added a pgfkeys-based simplified user interface | 60 |
| BodeMagPlot: Added a pgfkeys-based simplified user interface | 46 | \PoleZeroMapZPK: Added a pgfkeys-based simplified user interface | 68 |
| BodePhPlot: Added a pgfkeys-based simplified user interface | 45 | v3.0.1 | |
| BodePlot: Added a pgfkeys-based simplified user interface | 48 | BodePhPlot: Fixed a typo to resolve issue #17 | 45 |

| | | | |
|--------|---|---|----|
| v3.0.2 | | character tokens. | 68 |
| | <code>\bp@fix@add@opt</code> : Added a conditional to avoid always plotting the true plot, fixes issue #18 | | 52 |
| v3.0.3 | | | |
| | <code>\bp@parse@complex</code> : Rewritten to support arbitrary expressions with nested braces and parentheses . . | | 49 |
| v3.0.4 | | | |
| | <code>\BodeTF</code> : Fixed babel wrapper command argument specification for grabbing arguments | | 43 |
| | <code>\BodeZPK</code> : Fixed babel wrapper command argument specification for grabbing arguments | | 41 |
| | <code>\NicholsTF</code> : Fixed babel wrapper command argument specification for grabbing arguments | | 67 |
| | <code>\NyquistTF</code> : Fixed babel wrapper command argument specification for grabbing arguments | | 62 |
| | <code>\NyquistZPK</code> : Fixed babel wrapper command argument specification for grabbing arguments | | 61 |
| v3.1 | | | |
| | <code>\addBodeTFPlot</code> : Pre-expand <code>{\tf-spec}</code> and expand it before <code>\bp@contains@equal</code> so macro-valued specs route correctly. | | 44 |
| | <code>\addBodeTFPlots</code> : Added support for linear and asymptotic approximations when using the TF format. | | 45 |
| | <code>\addBodeZPKPlots</code> : Pre-expand <code>{\zpk-spec}</code> and expand it before <code>\bp@contains@equal</code> so macro-valued specs route correctly. | | 43 |
| | <code>\addNicholsTFChart</code> : Use <code>\expandafter</code> when calling <code>\bp@contains@equal</code> so the pre-expanded spec is tested as | | |
| | | <code>\addNicholsZPKChart</code> : Use <code>\expandafter</code> when calling <code>\bp@contains@equal</code> so the pre-expanded spec is tested as | |
| | | character tokens. | 67 |
| | <code>\addNyquistTFPlot</code> : Use <code>\expandafter</code> when calling <code>\bp@contains@equal</code> so the pre-expanded spec is tested as | | |
| | | character tokens. | 63 |
| | <code>\addNyquistZPKPlot</code> : Use <code>\expandafter</code> when calling <code>\bp@contains@equal</code> so the pre-expanded spec is tested as | | |
| | | character tokens. | 62 |
| | <code>\BodeTF</code> : In pgf mode, legacy 4-argument calls now forward frequency limits via domain options when dispatching to <code>\BodeZPK</code> | | 42 |
| | To avoid wrapping in pgf mode, use python to convert tf to zpk if available. If python is not available, fall back to the legacy implementation. | | 42 |
| | <code>\NicholsTF</code> : In pgf mode, legacy 4-argument calls now forward frequency limits via domain options when dispatching to <code>\NicholsZPK</code> | | 65 |
| | To avoid wrapping in pgf mode, use python to convert tf to zpk if available. If python is not available, fall back to the legacy implementation. | | 65 |
| | <code>\PoleZeroMapTF</code> : Added support for pole-zero maps with TF specifications using Python. . . . | | 71 |
| | <code>\PoleZeroMapZPK</code> : Use <code>\expandafter</code> when calling <code>\bp@contains@equal</code> so the pre-expanded spec is tested as | | |
| | | character tokens. | 68 |