

The **bodeplot** package

version 3.1.1

Rushikesh Kamalapurkar
rlkamalapurkar@gmail.com

October 5, 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	65
4.8	Pole-zero maps	69
4.8.1	User macros	69
4.8.2	Internal macros	72
A	TL;DR for package version 2.1.1 and earlier	78
	Index	90
	Change History	90

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.

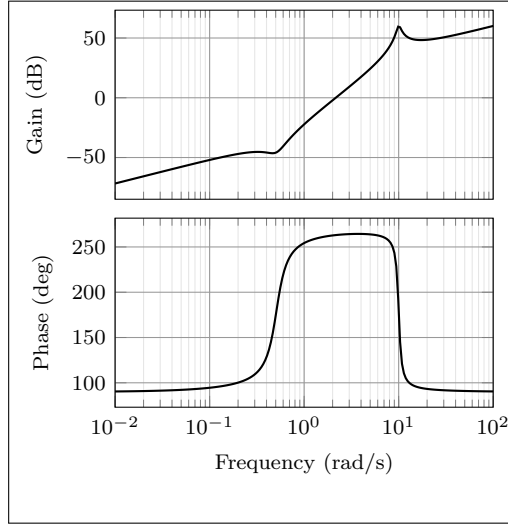
2 TL;DR

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.

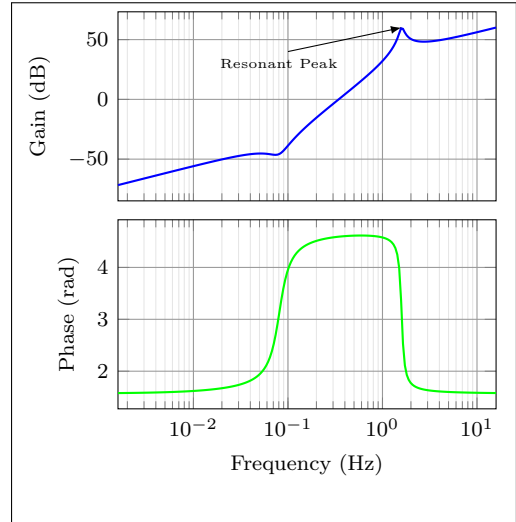
Bode plot in ZPK format



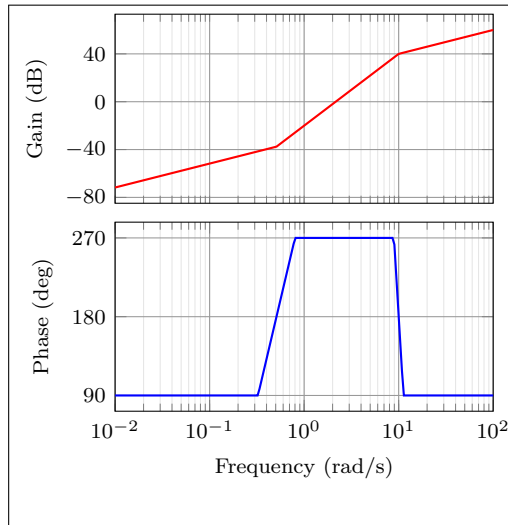
```
\BodeZPK[domain=0.01:100]{
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10
}
```

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}
}
```



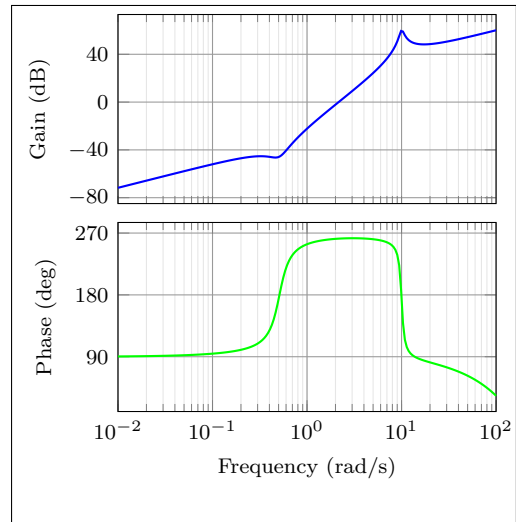
Linear approximation with customization



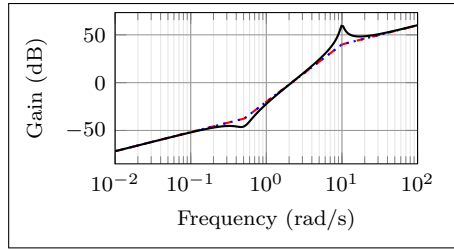
```
\BodeZPK[
  domain=0.01:100,
  mag plot={red,thick},
  ph plot={blue,thick},
  mag axes={ytick distance=40},
  ph axes={ytick distance=90},
  approx=linear
]{
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10
}
```

Plot with delay and customization

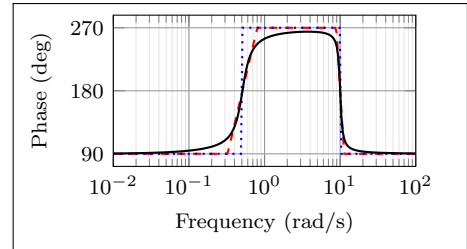
```
\BodeZPK[
  domain=0.01:100,
  mag plot={blue,thick},
  ph plot={green,thick},
  mag axes={ytick distance=40},
  ph axes={ytick distance=90}
]{
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10,
  delay=0.01
}
```



Individual gain and phase plots with more customization



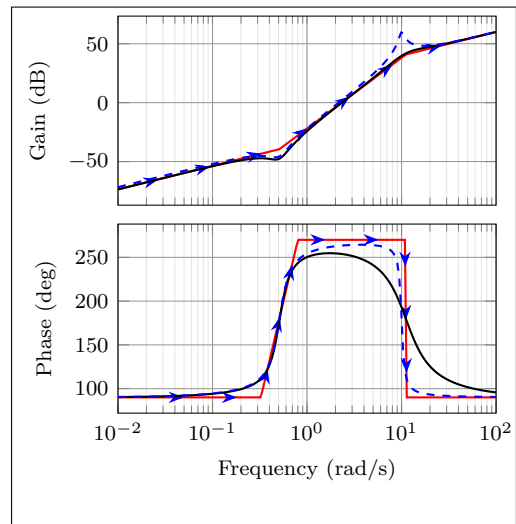
```
\begin{BodeMagPlot}[
  axes={height=2cm,
    width=4cm},
  domain=0.01:100
]
\addBodeZPKPlots[
  true={black,thick},
  linear={red,dashed,thick},
  asymptotic={blue,dotted,thick}
]
{magnitude}
{
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10
}
\end{BodeMagPlot}
```



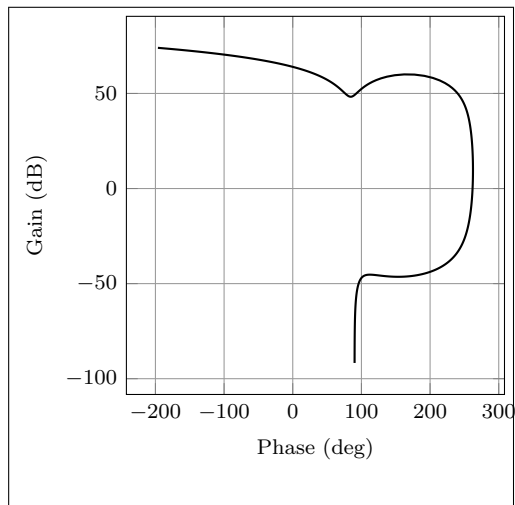
```
\begin{BodePhPlot}[
  height=2cm,
  width=4cm,
  ytick distance=90,
  domain=0.01:100
]
\addBodeZPKPlots[
  true={black,thick},
  linear={red,dashed,thick},
  asymptotic={blue,dotted,thick}
]
{phase}
{
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10
}
\end{BodePhPlot}
```

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}
```



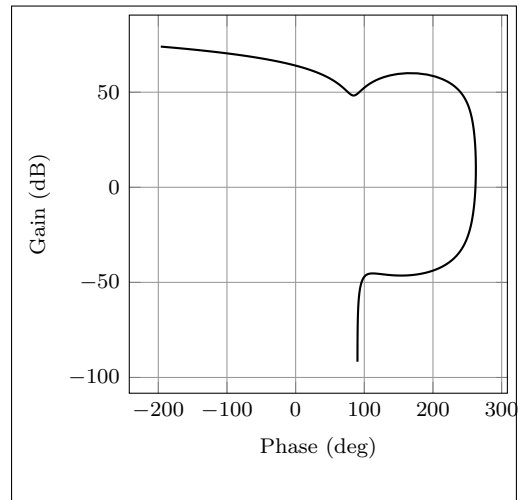
Nichols chart



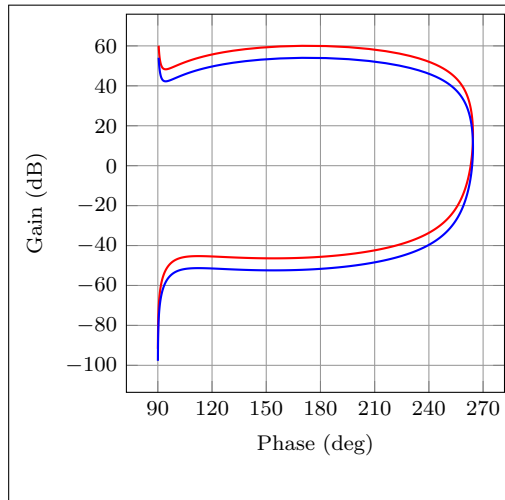
```
\NicholsZPK[domain=0.001:500,samples=1000]
{
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10,
  delay=0.01
}
```

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
}
```



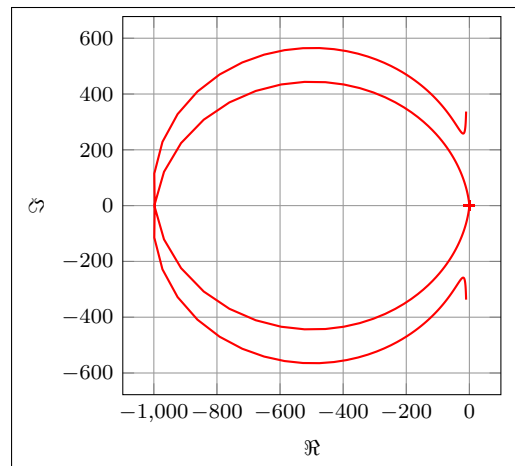
Multiple Nichols charts with customization



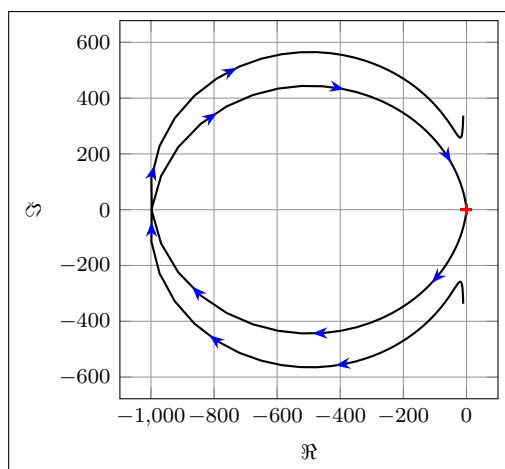
```
\begin{NicholsChart}[
  ytick distance=20,
  xtick distance=30,
  domain=0.001:100
]
\addNicholsZPKChart [red,samples=1000] {
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},
  gain=10
}
\addNicholsZPKChart [blue,samples=1000] {
  zeros={0,-0.1-0.5i,-0.1+0.5i},
  poles={-0.5-10i,-0.5+10i},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



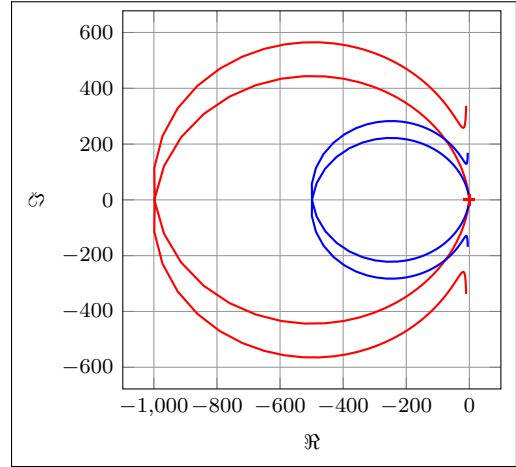
Nyquist plot in TF format with arrows



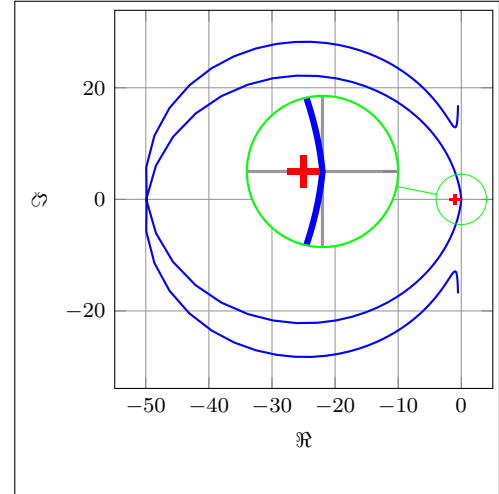
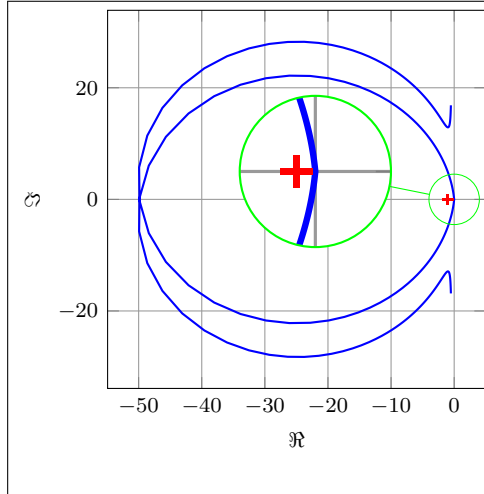
```
\NyquistTF[
  domain=-30:30,
  plot={
    samples=1000,
    postaction=decorate,
    decoration={
      markings,
      mark=between positions 0.1 and 0.9 step 5em with {
        \arrow{Stealth [length=2mm, blue]}
      }
    }
  }
]
{
  numerator={10,2,2.6,0},
  denominator={1,1,100.25}
}
```

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}
```



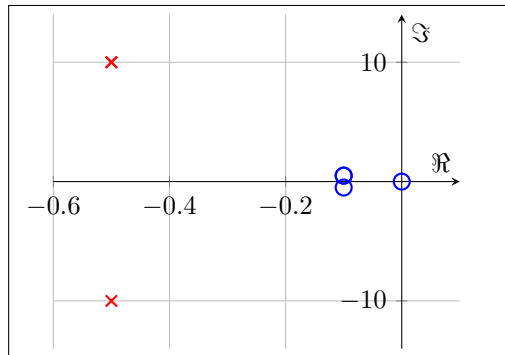
Nyquist plots with additional commands, using two different macros



```
\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
}
```

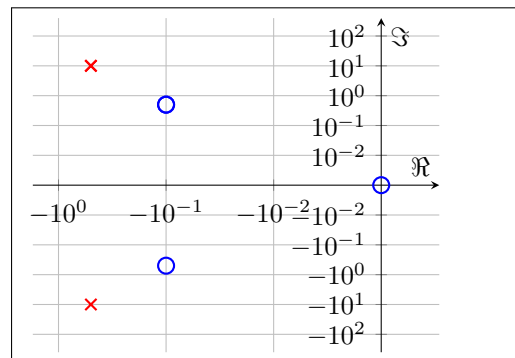
Pole-zero map



```
\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.

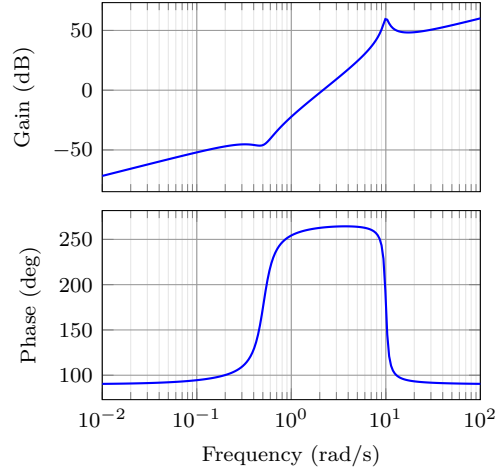


Figure 1: Output of the `\BodeZPK` macro.

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:

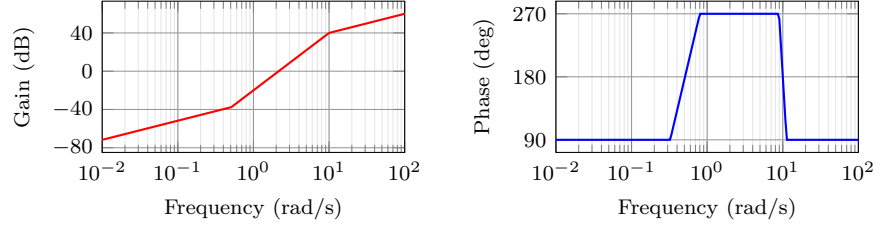


Figure 2: Customization of the default `\BodeZPK` macro.

- `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,

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

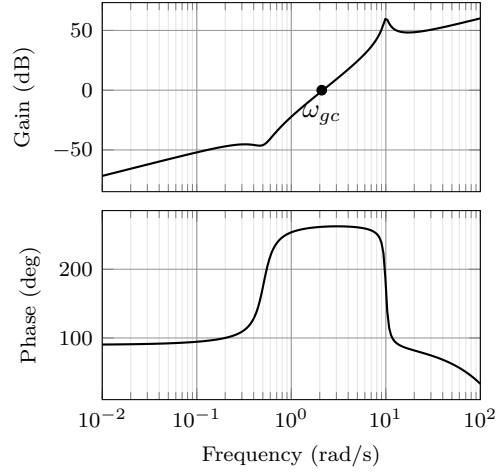


Figure 3: Output of the `\BodeTF` macro with an optional TikZ command used to mark the gain crossover frequency.

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}
```

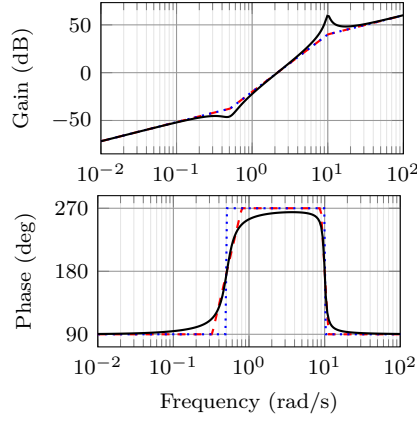


Figure 4: Superimposed approximate and true Bode plots using the **BodeMagPlot** and **BodePhPlot** environments and the `\addBodeZPKPlots` macro.

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

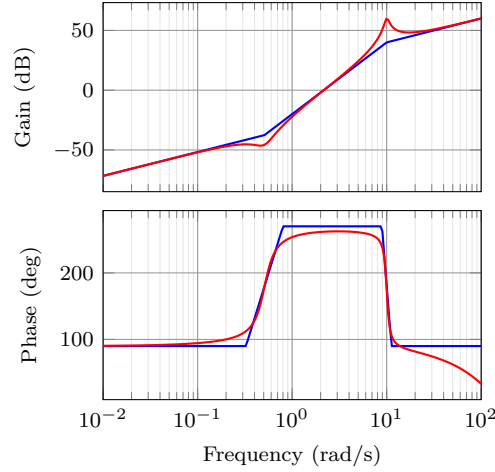


Figure 5: Bode plot combining ZPK and TF systems using the `BodePlot` environment and the unified `\addBodePlot` macro.

`\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]{

```

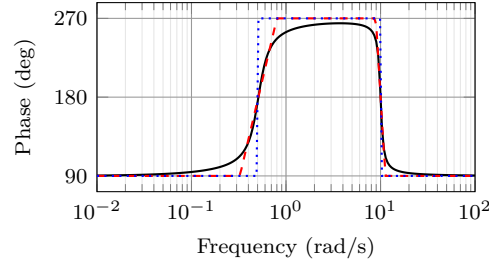


Figure 6: Superimposed approximate and true Bode Phase plot using the `BodePhPlot` environment, the `\addBodeComponentPlot` macro, and several macros of the `\TypeFeatureApprox` form.

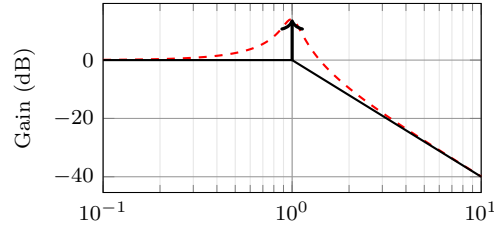


Figure 7: Resonant peak in asymptotic Bode plot using `\MagSOPolesPeak`.

```
\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}
```

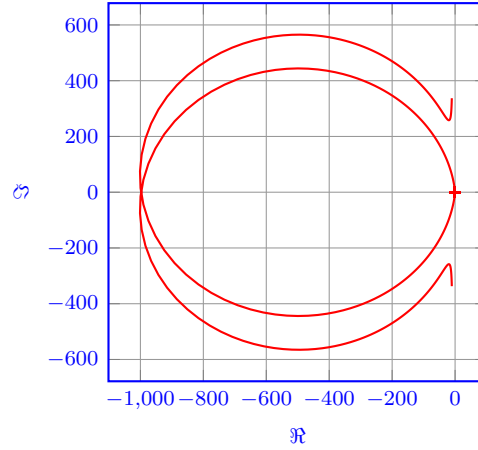


Figure 8: Output of the `\NyquistZPK` macro.

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}
```

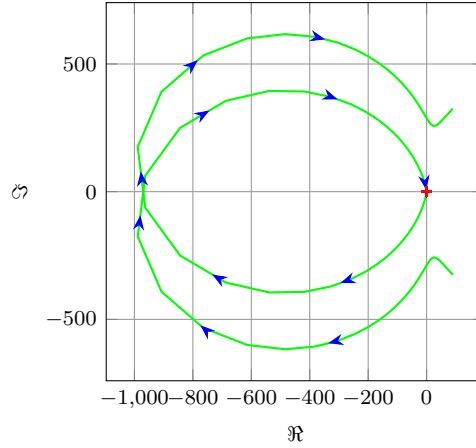


Figure 9: Output of the `\NyquistTF` macro with direction arrows. Increasing the number of samples can cause `decorations.markings` to throw errors.

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`.

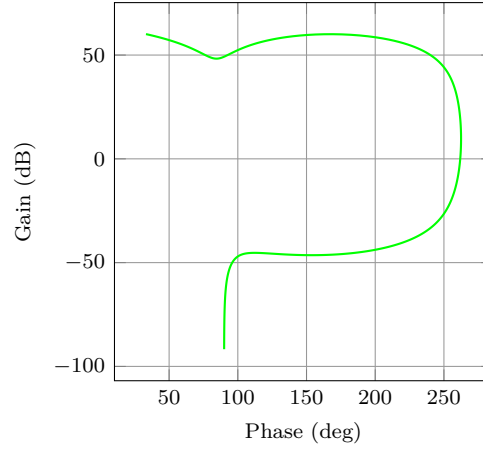


Figure 10: Output of the `\NicholsTF` macro.

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.

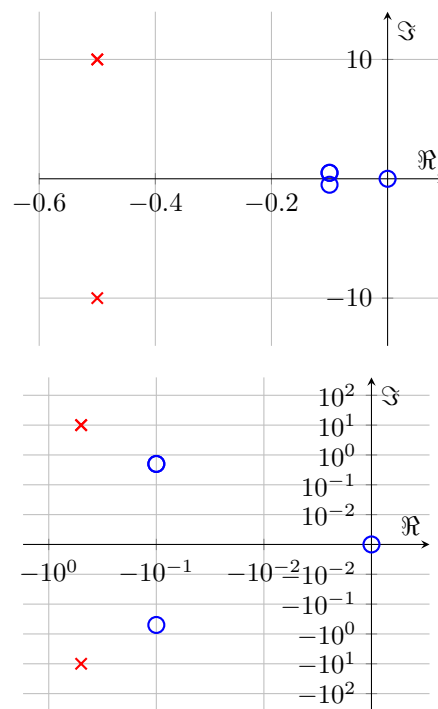


Figure 11: Pole-zero maps generated using the `\PoleZeroMapZPK` macro with linear and logarithmic scaling.

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 \ifpgfarg
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 \ifpgfarg
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 (\log10(\n@pow{10}{sqrt((4*\n@pow{#1}{2})/
           (\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 $G(s) = \frac{1}{s^2 + 2\zeta\omega_n s + \omega_n^2}$ 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*\MagSOPolesPeak{0-\MagSOPoles}
850 \newcommand*\MagSOPolesLinPeak{0-\MagSOPolesLin}
851 \newcommand*\MagSOPolesAsympPeak{0-\MagSOPolesAsymp}
852 \newcommand*\PhSOPolesPeak{0-\PhSOPoles}
853 \newcommand*\PhSOPolesLinPeak{0-\PhSOPolesLin}
854 \newcommand*\PhSOPolesAsympPeak{0-\PhSOPolesAsymp}

```

\MagSOPolesPeak These macros are used to add a resonant peak to linear and asymptotic plots of general
\MagSOPolesLinPeak second order poles and zeros. Since the plots are parametric, a separate **\draw** command
\MagSOPolesAsympPeak 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 \newcommand*\MagSOPolesLinPeak[3]{\draw[>] (axis cs:{sqrt(abs(#3))},{20*log10(abs(#3))} --
860 (axis cs:{sqrt(abs(#3))},{20*log10(abs(#3)) +
861 20*log10(abs(#2/sqrt(abs(#3)))});
862 }
863 \newcommand*\PhSOPolesPeak[3]{\draw[>] (axis cs:{sqrt(abs(#3))},{-20*log10(abs(#3))} --
864 (axis cs:{sqrt(abs(#3))},{-20*log10(abs(#3)) -
865 20*log10(abs(#2/sqrt(abs(#3)))});
866 }

```

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}{>\TrimSpaces} 0{} >\TrimSpaces} m >\TrimSpaces} G{} >\TrimSpaces}

```

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}{ >{\TrimSpaces} 0{ } >{\TrimSpaces} m >{\TrimSpaces} G{ } >{\TrimSpaces} }{
954   \ifpgfarg
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}{ >{\TrimSpaces}0{} >{\TrimSpaces} m >{\TrimSpaces} 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}{ >{\TrimSpaces} 0{} >{\TrimSpaces} m >{\TrimSpaces} 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}{>{\TrimSpaces} 0}{>{\TrimSpaces} m >{\TrimSpaces} }
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}{ >{\TrimSpaces} 0{} >{\TrimSpaces} G{} >{\TrimSpaces}
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}{ >{\TrimSpaces} 0{} >{\TrimSpaces} G{} >{\TrimSpaces}
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}{>{\TrimSpaces} 0{} >{\TrimSpaces} m >{\TrimSpaces} 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}{>\TrimSpaces}0{}>\TrimSpaces}G{}>\TrimSpaces}G{
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:nNt { \l__bp_pos_int } > {0}
1458 {
1459     \int_compare:nNt { \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:nNtF { \l__bp_pos_int } < { \l__bp_seppos_int }
1489     { \tl_put_right:Nn \l__bp_before_tl {#1} }
1490     {
1491         \int_compare:nNtF { \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 }

```

\bp@trim Removes leading and trailing spaces from a macro's contents, used to clean up option values extracted by the option parsers.

```

1536 \cs_new_eq:NN \bp@trim \tl_trim_spaces:N
1537 \ExplSyntaxOff

```

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

```

1538 \NewDocumentCommand{\bp@fix@add@opt}{ m }{%
1539     \gdef\bp@add@0{}
1540     \gdef\bp@add@tmp{}
1541     \foreach \approx/\opt in {#1} {
1542         \ifx\approx\@empty\else
1543             \ifnum\pdf@strcmp{\approx}{linear}=0
1544                 \xdef\bp@add@0{\unexpanded\expandafter{\bp@add@0}%
1545                     linear/{\unexpanded\expandafter{\opt}},}
1546             \else
1547                 \ifnum\pdf@strcmp{\approx}{asymptotic}=0
1548                     \xdef\bp@add@0{\unexpanded\expandafter{\bp@add@0}%
1549                         asymptotic/{\unexpanded\expandafter{\opt}},}
1550                 \else
1551                     \ifnum\pdf@strcmp{\approx}{true}=0
1552                         \xdef\bp@add@tmp{\unexpanded\expandafter{\bp@add@tmp}%
1553                             \unexpanded\expandafter{\opt}},}
1554                     \else
1555                         \xdef\bp@add@tmp{\unexpanded\expandafter{\bp@add@tmp}%
1556                             \unexpanded\expandafter{\approx}},}
1557                     \fi
1558                 \fi
1559             \fi
1560         \fi
1561     }
1562     \ifx\bp@add@tmp\@empty\else
1563         \xdef\bp@add@0{\unexpanded\expandafter{\bp@add@0}%
1564             true/{\unexpanded\expandafter{\bp@add@tmp}}}
1565     \fi
1566 }

```

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

```

1567 \newcommand*{\bp@add}[3]{
1568   \ifcat$\detokenize\expandafter{#1}$
1569     \xdef#1{\unexpanded\expandafter{#1 0+#2}}
1570   \else
1571     \xdef#1{\unexpanded\expandafter{#1+#2}}
1572   \fi
1573   \foreach \y [count=\n] in #3 {
1574     \xdef#1{\unexpanded\expandafter{#1}{\y}}
1575     \xdef\Last@LoopValue{\n}
1576   }
1577   \ifnum\Last@LoopValue=1
1578     \xdef#1{\unexpanded\expandafter{#1}{0}}
1579   \fi
1580 }

```

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

```

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

```

```

1627         \bp@add{#2}{\PhKAsymp}{\values}
1628         \bp@add{#1}{\MagKAsymp}{\values}
1629     \else
1630         \bp@add{#2}{\PhK}{\values}
1631         \bp@add{#1}{\MagK}{\values}
1632     \fi
1633 \fi
1634 \fi
1635 \ifnum\pdf@strcmp{\feature}{d}=0
1636     \ifnum\pdf@strcmp{#3}{linear}=0
1637         \PackageError {bodeplot}
1638         {Linear approximation for pure delays is not supported.}
1639         {Plot the true Bode plot using `true' instead of `linear'.}
1640     \else
1641         \ifnum\pdf@strcmp{#3}{asymptotic}=0
1642             \PackageError {bodeplot}
1643             {Asymptotic approximation for pure delays is not supported.}
1644             {Plot the true Bode plot using `true' instead of `asymptotic'}.
1645         \else
1646             \ifdim\values pt < 0pt
1647                 \PackageError {bodeplot} {Delay needs to be a positive number.}
1648             \fi
1649             \bp@add{#2}{\PhDel}{\values}
1650             \bp@add{#1}{\MagDel}{\values}
1651         \fi
1652     \fi
1653 \fi
1654 \fi
1655 }
1656 }

```

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

```

1657 \newcommand{\bp@gnu@plot}[3][1]{
1658     \xdef\bp@gnu@cmd{ gnuplot [raw gnuplot, id=#3, prefix=\bp@prefix]
1659     { set table $meta;
1660       set dummy t;
1661       set logscale x 10;
1662       set xrange [#1*\pgfkeysvalueof{/pgfplots/domain}*#1];
1663       set samples \pgfkeysvalueof{/pgfplots/samples};
1664       plot #2;
1665       set table "\bp@prefix#3.table";
1666       plot "$meta" using ($1/(\bp@freq@scale)):$2);
1667     };}
1668 }

```

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

```

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

```

```

1685 \gdef\bp@p@list{}
1686 \ifx\bp@p@empty\else
1687   \foreach \p in \bp@p {
1688     \bp@parse@complex{\p}
1689     \xdef\bp@p@list{\unexpanded\expandafter{\bp@p@list} \bp@complex,}
1690   }
1691 \fi
1692 \pgfkeysgetvalue{/bodeplot/zpk/@gain}{\bp@k}
1693 \pgfkeysgetvalue{/bodeplot/zpk/@delay}{\bp@d}
1694 \xdef\bp@legacy{z/{\bp@z@list},p/{\bp@p@list},k/\bp@k,d/\bp@d}
1695 }

```

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

```

1696 \NewDocumentCommand{\bp@tf@new@to@legacy}{ m }{%
1697   % Pre-expand #1 so pgfkeys sees character tokens (not a macro token).
1698   \edef\bp@tf@n@spec{/bodeplot/tf/.cd, reset, #1}%
1699   \expandafter\pgfkeys\expandafter{\bp@tf@n@spec}
1700   \pgfkeysgetvalue{/bodeplot/tf/@numerator}{\bp@num}
1701   \pgfkeysgetvalue{/bodeplot/tf/@denominator}{\bp@den}
1702   \pgfkeysgetvalue{/bodeplot/tf/@delay}{\bp@d}
1703   \gdef\bp@num@list{}
1704   \ifx\bp@num@empty\else
1705     \foreach \n in \bp@num {
1706       \xdef\bp@num@list{\unexpanded\expandafter{\bp@num@list} \n,}
1707     }
1708   \fi
1709   \gdef\bp@den@list{}
1710   \ifx\bp@den@empty\else
1711     \foreach \d in \bp@den {
1712       \xdef\bp@den@list{\unexpanded\expandafter{\bp@den@list} \d,}
1713     }
1714   \fi
1715   \xdef\bp@legacy{num/{\bp@num@list},den/{\bp@den@list},d/\bp@d}
1716 }

```

\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[\<opts>]\expandafter{\<output-
cs>}

```

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{<tf-spec>}{<zpk-cs>}{<status>}`

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`).

```

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

```

```

1767 \def#3{true}%
1768 }{%
1769 \def#3{false}%
1770 }%
1771 }

```

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

```

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

```

```

1824         \fi
1825     }
1826     \fi
1827     \ifnum\pdf@strcmp{\feature}{d}=0
1828         \xdef\bp@loop@delay{\values}
1829     \fi
1830     \fi
1831 }
1832 \xdef#2{((atan2((\bp@num@im),(\bp@num@re))-atan2((\bp@den@im),%
1833 (\bp@den@re))- \bp@loop@delay*t)*(\bp@ph@scale))}
1834 \xdef#1{(20*log10(sqrt((\n@pow{\bp@num@re}{2})+(\n@pow{\bp@num@im}{2}))) -
%
1835 20*log10(sqrt((\n@pow{\bp@den@re}{2})+(\n@pow{\bp@den@im}{2}))))}
1836 }

```

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

```

1837 \newcommand{\bp@gnu@unwrap@plot}[3][1]{
1838 \xdef\bp@gnu@cmd{ gnuplot [raw gnuplot, id=#3, prefix=\bp@prefix]
1839 { set table $meta;
1840   set dummy t;
1841   set logscale x 10;
1842   set trange [#1*\pgfkeysvalueof{/pgfplots/domain}*\#1];
1843   set samples \pgfkeysvalueof{/pgfplots/samples};
1844   plot '+' using (t) : ((#2)/(\bp@ph@scale)) smooth unwrap;
1845   set table "\bp@prefix#3.table";
1846   plot "$meta" using ($1/(\bp@freq@scale)) : ($2*\bp@ph@scale);
1847   };}
1848 }

```

\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.

```

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

```

```

1875 \ifnum\pdf@strcmp{\unexpanded\expandafter{\typ}}{mag}=0
1876 \xdef\bp@mag@axes{\unexpanded\expandafter{\opt}}
1877 \else
1878 \ifnum\pdf@strcmp{\unexpanded\expandafter{\typ}}{ph}=0
1879 \xdef\bp@ph@axes{\unexpanded\expandafter{\opt}}
1880 \else
1881 \xdef\bp@mag@axes{\unexpanded\expandafter{\opt}}
1882 \xdef\bp@ph@axes{\unexpanded\expandafter{\opt}}
1883 \fi
1884 \fi
1885 \else
1886 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{group}=0
1887 \xdef\bp@group{\unexpanded\expandafter{\opt}}
1888 \else
1889 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{approx}=0
1890 \xdef\bp@approx{\unexpanded\expandafter{\opt}}
1891 \else
1892 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{commands}=0
1893 \ifnum\pdf@strcmp{\unexpanded\expandafter{\typ}}{ph}=0
1894 \xdef\bp@ph@commands{\unexpanded\expandafter{\opt}}
1895 \else
1896 \xdef\bp@mag@commands{\unexpanded\expandafter{\opt}}
1897 \fi
1898 \else
1899 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{tikz}=0
1900 \xdef\bp@tikz{\unexpanded\expandafter{\opt}}
1901 \else
1902 \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{prefix}=0
1903 \xdef\bp@user@prefix{\unexpanded\expandafter{\opt}}
1904 \else
1905 \xdef\bp@user@prefix{}
1906 \xdef\bp@mag@plot{\unexpanded\expandafter{\bp@mag@plot},
1907 \unexpanded\expandafter{\obj}}
1908 \xdef\bp@ph@plot{\unexpanded\expandafter{\bp@ph@plot},
1909 \unexpanded\expandafter{\obj}}
1910 \fi
1911 \fi
1912 \fi
1913 \fi
1914 \fi
1915 \fi
1916 \fi
1917 }
1918 }

```

`\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.

```

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

```

```

1932         \xdef\bp@user@prefix{\unexpanded\expandafter{\opt}}
1933     \else
1934         \xdef\bp@user@prefix{}
1935         \xdef\bp@axes{\unexpanded\expandafter{\bp@axes},
1936             \unexpanded\expandafter{\obj}}
1937     \fi
1938 \fi
1939 \fi
1940 }
1941 }

```

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

```

1942 \newcommand{\bp@parse@add@Bode@opt}[1]{
1943   \gdef\bp@plot{}
1944   \gdef\bp@approx{}
1945   \foreach \opt in {#1} {
1946     \bp@trim\opt
1947     \ifx\opt\@empty\else
1948       \ifnum\pdf@strcmp{\unexpanded\expandafter{\opt}}{linear}=0
1949         \xdef\bp@approx{\unexpanded\expandafter{\opt}}
1950       \else
1951         \ifnum\pdf@strcmp{\unexpanded\expandafter{\opt}}{asymptotic}=0
1952           \xdef\bp@approx{\unexpanded\expandafter{\opt}}
1953         \else
1954           \xdef\bp@plot{\unexpanded\expandafter{\bp@plot}%
1955             \unexpanded\expandafter{\opt},}
1956         \fi
1957       \fi
1958     \fi
1959   }
1960 }

```

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!

```

1961 \NewDocumentCommand{\NyquistZPK}{ >\TrimSpaces} 0{} >\TrimSpaces} m >\TrimSpaces} G{} :
1962   \pgfutil@ifempty{#3}{%
1963     \pgfkeys{/bodeplot/nyquist/.cd, reset}
1964     \pgfkeys{/bodeplot/nyquist/.cd, #1}
1965     \pgfkeysgetvalue{/bodeplot/nyquist/@axes}{\bp@axes}
1966     \pgfkeysgetvalue{/bodeplot/nyquist/@commands}{\bp@commands}
1967     \pgfkeysgetvalue{/bodeplot/nyquist/@tikz}{\bp@tikz}
1968     \pgfkeysgetvalue{/bodeplot/nyquist/@prefix}{\bp@user@prefix}
1969     \bp@zpk@new@to@legacy{#2}
1970   }{%
1971     \bp@parse@N@opt{#1}
1972     \edef\bp@legacy{#2}
1973     \edef\bp@domain@start{#3}
1974     \edef\bp@domain@end{#4}
1975   }%
1976   \gdef\bp@mag{}\gdef\bp@ph{}%

```

```

1977 \edef\bp@cmd{\noexpand\begin{tikzpicture}
1978 [\unexpanded\expandafter{\bp@tikz}}]\bp@cmd
1979 \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
1980 \edef\bp@cmd{\noexpand\begin{axis}[
1981     bp@style,
1982     domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
1983     height=5cm,
1984     xlabel={\Re$},
1985     ylabel={\Im$},
1986     samples=500,
1987     \unexpanded\expandafter{\bp@axes}
1988 ]}
1989 \bp@cmd
1990 \addplot [only marks,mark=+,thick,red] (-1 , 0);
1991 \edef\bp@cmd{\noexpand\addplot
1992 [variable=t, thick, trig format plots=rad,
1993 \unexpanded\expandafter{\bp@plot}]}
1994 \ifpgfarg
1995     \bp@cmd
1996     ( {\n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale))},
1997       {\n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))} );
1998     \bp@commands
1999 \else
2000     \stepcounter{bp@gnuplot@id}
2001     \bp@cmd gnuplot [parametric, bp@gnu@prefix] {
2002         \n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale)),
2003         \n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))
2004     };
2005     \bp@commands
2006 \fi
2007 \end{axis}
2008 \end{tikzpicture}
2009 }

```

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

```

2010 \AtBeginDocument{%
2011 \if@babel
2012 \let\Orig@NyquistZPK\NyquistZPK
2013 \renewcommand{\NyquistZPK}{%
2014 \expandafter\shorthandoff\expandafter{\bp@short@list}
2015 \NyquistZPK@Shorthandoff
2016 }
2017 \NewDocumentCommand{\NyquistZPK@Shorthandoff}{ 0{} m G{} G{} }{%
2018 \Orig@NyquistZPK[#1]{#2}{#3}{#4}
2019 \expandafter\shorthandon\expandafter{\bp@short@list}
2020 }
2021 \fi
2022 }

```

\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**.

```

2023 \NewDocumentCommand{\NyquistTF}{ >\TrimSpaces 0{} >\TrimSpaces m >\TrimSpaces G{} >
2024 \pgfutil@ifempty{#3}{%
2025 \pgfkeys{/bodeplot/nyquist/.cd, reset}
2026 \pgfkeys{/bodeplot/nyquist/.cd, #1}
2027 \pgfkeysgetvalue{/bodeplot/nyquist/@axes}{\bp@axes}
2028 \pgfkeysgetvalue{/bodeplot/nyquist/@commands}{\bp@commands}
2029 \pgfkeysgetvalue{/bodeplot/nyquist/@tikz}{\bp@tikz}
2030 \pgfkeysgetvalue{/bodeplot/nyquist/@prefix}{\bp@user@prefix}
2031 \bp@tf@new@to@legacy{#2}
2032 }{%
2033 \bp@parse@N@opt{#1}
2034 \edef\bp@legacy{#2}
2035 \edef\bp@domain@start{#3}

```

```

2036 \edef\bp@domain@end{#4}
2037 }%
2038 \gdef\bp@mag{}\gdef\bp@ph{}%
2039 \edef\bp@cmd{\noexpand\begin{tikzpicture}
2040 [\unexpanded\expandafter{\bp@tikz}]\bp@cmd
2041 \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2042 \edef\bp@cmd{\noexpand\begin{axis}[
2043 bp@style,
2044 domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
2045 height=5cm,
2046 xlabel={\Re$},
2047 ylabel={\Im$},
2048 samples=500,
2049 \unexpanded\expandafter{\bp@axes}
2050 ]}
2051 \bp@cmd
2052 \addplot [only marks, mark=+, thick, red] (-1 , 0);
2053 \edef\bp@cmd{\noexpand\addplot
2054 [variable=t, thick, trig format plots=rad,
2055 \unexpanded\expandafter{\bp@plot}]}
2056 \if@pgfarg
2057 \bp@cmd
2058 ( {\n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale))},
2059 {\n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))} );
2060 \bp@commands
2061 \else
2062 \stepcounter{bp@gnuplot@id}
2063 \bp@cmd gnuplot [parametric, bp@gnu@prefix] {
2064 \n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale)),
2065 \n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))
2066 };
2067 \bp@commands
2068 \fi
2069 \end{axis}
2070 \end{tikzpicture}
2071 }

```

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

```

2072 \AtBeginDocument{%
2073 \if@babel
2074 \let\Orig@NyquistTF\NyquistTF
2075 \renewcommand{\NyquistTF}{%
2076 \expandafter\shorthandoff\expandafter{\bp@short@list}
2077 \NyquistTF@Shorthandoff
2078 }
2079 \NewDocumentCommand{\NyquistTF@Shorthandoff}{ 0{} m G{} G{} }{%
2080 \Orig@NyquistTF[#1]{#2}{#3}{#4}
2081 \expandafter\shorthandon\expandafter{\bp@short@list}
2082 }
2083 \fi
2084 }

```

\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.

```

2085 \NewDocumentCommand{\addNyquistZPKPlot}{ >\TrimSpaces 0{} >\TrimSpaces m }{%
2086 \edef\bp@spec@arg{#2}%
2087 \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2088 \bp@zpk@new@to@legacy{\bp@spec@arg}}{\edef\bp@legacy{\bp@spec@arg}}%
2089 \gdef\bp@mag{}\gdef\bp@ph{}%
2090 \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2091 \if@pgfarg
2092 \edef\bp@cmd{\noexpand\addplot
2093 [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
2094 variable=t, thick, trig format plots=rad, #1]}%

```

```

2095 \bp@cmd ( {\n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale))},
2096           {\n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))} );
2097 \else
2098 \stepcounter{bp@gnuplot@id}%
2099 \edef\bp@cmd{\noexpand\addplot
2100 [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale, thick, #1]
2101 \bp@cmd gnuplot [parametric, bp@gnu@prefix] {%
2102 \n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale)),
2103 \n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))
2104 };
2105 \fi
2106 }

```

\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.

```

2107 \NewDocumentCommand{\addNyquistTFPlot}{>\TrimSpaces} 0{} >\TrimSpaces} m }{%
2108 \edef\bp@spec@arg{#2}%
2109 \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2110 \bp@tf@new@to@legacy{\bp@spec@arg}}{\edef\bp@legacy{\bp@spec@arg}}%
2111 \gdef\bp@mag{}\gdef\bp@ph{}%
2112 \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2113 \if@pgfarg
2114 \edef\bp@cmd{\noexpand\addplot
2115 [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
2116 variable=t, thick, trig format plots=rad, #1]}%
2117 \bp@cmd ( {\n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale))},
2118           {\n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))} );
2119 \else
2120 \stepcounter{bp@gnuplot@id}%
2121 \edef\bp@cmd{\noexpand\addplot
2122 [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale, thick, #1]
2123 \bp@cmd gnuplot [parametric, bp@gnu@prefix] {%
2124 \n@pow{10}{((\bp@mag)/20)}*cos((\bp@ph)/(\bp@ph@scale)),
2125 \n@pow{10}{((\bp@mag)/20)}*sin((\bp@ph)/(\bp@ph@scale))
2126 };
2127 \fi
2128 }

```

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.

```

2129 \AtBeginDocument{%
2130 \if@babel
2131 \AddToHook{env/NyquistPlot/begin}{%
2132 \expandafter\shorthandoff\expandafter{\bp@short@list}
2133 }
2134 \AddToHook{env/NyquistPlot/end}{%
2135 \expandafter\shorthandon\expandafter{\bp@short@list}
2136 }
2137 \fi
2138 }
2139 \NewDocumentEnvironment{NyquistPlot}{>\TrimSpaces} 0{} >\TrimSpaces} G{} >\TrimSpaces}
2140 \pgfutil@ifempty{#2}{%
2141 \pgfkeys{/bodeplot/env/.cd, reset}
2142 \pgfkeys{/bodeplot/env/.cd, #1}
2143 \pgfkeysgetvalue{/bodeplot/env/@tikz}{\bp@tikz}
2144 \pgfkeysgetvalue{/bodeplot/env/@prefix}{\bp@user@prefix}
2145 }{%
2146 \bp@parse@env@opt{#1}
2147 \edef\bp@domain@start{#2}%
2148 \edef\bp@domain@end{#3}%
2149 }
2150 \edef\bp@cmd{\noexpand\begin{tikzpicture}

```

```

2151 [\unexpanded\expandafter{\bp@tikz}]]\bp@cmd
2152 \edef\bp@cmd{\noexpand\begin{axis}[
2153   bp@style,
2154   height=5cm,
2155   domain=\bp@domain@start:\bp@domain@end,
2156   xlabel={\Re$},
2157   ylabel={\Im$},
2158   \unexpanded\expandafter{\bp@axes}%
2159 ]}\bp@cmd
2160 \addplot [only marks,mark=+,thick,red] (-1 , 0);
2161 #4
2162 \end{axis}
2163 \end{tikzpicture}
2164 }{}

```

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.

```

2165 \newcommand{\bp@parse@N@opt}[1]{
2166   \gdef\bp@axes{}
2167   \gdef\bp@plot{}
2168   \gdef\bp@commands{}
2169   \gdef\bp@tikz{}
2170   \gdef\bp@user@prefix{}
2171   \gdef\bp@scale{linear}
2172   \foreach \obj/\opt in {#1} {
2173     \bp@trim\obj\bp@trim\opt
2174     \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{axes}=0
2175       \xdef\bp@axes{\unexpanded\expandafter{\opt}}
2176     \else
2177       \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{plot}=0
2178         \xdef\bp@plot{\unexpanded\expandafter{\opt}}
2179       \else
2180         \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{commands}=0
2181           \xdef\bp@commands{\unexpanded\expandafter{\opt}}
2182         \else
2183           \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{tikz}=0
2184             \xdef\bp@tikz{\unexpanded\expandafter{\opt}}
2185           \else
2186             \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{scale}=0
2187               \xdef\bp@scale{\unexpanded\expandafter{\opt}}
2188             \else
2189               \ifnum\pdf@strcmp{\unexpanded\expandafter{\obj}}{prefix}=0
2190                 \xdef\bp@user@prefix{\unexpanded\expandafter{\opt}}
2191               \else
2192                 \xdef\bp@user@prefix{}
2193                 \xdef\bp@plot{\unexpanded\expandafter{\bp@plot},
2194                   \unexpanded\expandafter{\obj}}
2195               \fi
2196             \fi
2197           \fi
2198         \fi
2199       \fi
2200     \fi
2201   }
2202 }

```

4.7 Nichols charts

\NicholsZPK

```

2203 \NewDocumentCommand{\NicholsZPK}{>{\TrimSpaces}0{}>{\TrimSpaces}m>{\TrimSpaces}G{}>
2204 \pgfutil@ifempty{#3}{%
2205 \pgfkeys{/bodeplot/nichols/.cd,reset}
2206 \pgfkeys{/bodeplot/nichols/.cd,#1}
2207 \pgfkeysgetvalue{/bodeplot/nichols/@axes}{\bp@axes}
2208 \pgfkeysgetvalue{/bodeplot/nichols/@commands}{\bp@commands}
2209 \pgfkeysgetvalue{/bodeplot/nichols/@tikz}{\bp@tikz}
2210 \pgfkeysgetvalue{/bodeplot/nichols/@prefix}{\bp@user@prefix}
2211 \bp@zpk@new@to@legacy{#2}
2212 }{%
2213 \bp@parse@N@opt{#1}
2214 \edef\bp@legacy{#2}
2215 \edef\bp@domain@start{#3}
2216 \edef\bp@domain@end{#4}
2217 }%
2218 \gdef\bp@mag{}\gdef\bp@ph{}%
2219 \edef\bp@cmd{\noexpand\begin{tikzpicture}
2220 [\unexpanded\expandafter{\bp@tikz}]\bp@cmd
2221 \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2222 \edef\bp@cmd{\noexpand\begin{axis}[
2223 bp@ph@x@label,
2224 bp@style,
2225 domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
2226 height=5cm,
2227 ylabel={Gain (dB)},
2228 samples=500,
2229 \unexpanded\expandafter{\bp@axes}
2230 ]}
2231 \bp@cmd
2232 \edef\bp@cmd{\noexpand\addplot
2233 [variable=t,thick,trig format plots=rad,
2234 \unexpanded\expandafter{\bp@plot}]}
2235 \ifpgfarg
2236 \bp@cmd({\bp@ph},{\bp@mag});
2237 \bp@commands
2238 \else
2239 \stepcounter{bp@gnuplot@id}
2240 \bp@cmd gnuplot [raw gnuplot, bp@gnu@prefix]
2241 { set table $meta;
2242 set logscale x 10;
2243 set dummy t;
2244 set samples \pgfkeysvalueof{/pgfplots/samples};
2245 set trange [\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale];
2246 plot '+' using (\bp@mag):((\bp@ph)/(\bp@ph@scale));
2247 unset logscale x;
2248 set table "\bp@prefix\arabic{bp@gnuplot@id}.table";
2249 plot "$meta" using ($2*\bp@ph@scale):($1);
2250 };
2251 \bp@commands
2252 \fi
2253 \end{axis}
2254 \end{tikzpicture}
2255 }
2256 \AtBeginDocument{%
2257 \if@babel
2258 \let\Orig@NicholsZPK\NicholsZPK
2259 \renewcommand{\NicholsZPK}{%
2260 \expandafter\shorthandoff\expandafter{\bp@short@list}
2261 \NicholsZPK@Shorthandoff
2262 }

```

```

2263 \NewDocumentCommand{\NicholsZPK@Shorthandoff}{ 0{} m G{} G{} }{%
2264   \Orig@NicholsZPK[#1]{#2}{#3}{#4}
2265   \expandafter\shorthandon\expandafter{\bp@short@list}
2266 }
2267 \fi
2268 }

```

\NicholsTF

```

2269 \NewDocumentCommand{\NicholsTF}{ >{\TrimSpaces}0{} >{\TrimSpaces} m >{\TrimSpaces} G{} >{\TrimSpaces}0{} }{%
2270   \ifpgfarg
2271     \bp@tf@to@zpk{#2}{\bp@tmp@zpk}{\bp@tmp@status}
2272     \ifnum\pdf@strcmp{\bp@tmp@status}{true}=0
2273       \pgfutil@ifempty{#3}{%
2274         \NicholsZPK[#1]{\bp@tmp@zpk}%
2275       }{%
2276         \NicholsZPK[domain=#3:#4,#1]{\bp@tmp@zpk}%
2277       }%
2278       \def\bp@fallback{false}
2279     \else
2280       \def\bp@fallback{true}
2281     \fi
2282   \else
2283     \def\bp@fallback{true}
2284   \fi
2285   \ifnum\pdf@strcmp{\bp@fallback}{true}=0
2286     \pgfutil@ifempty{#3}{%
2287       \pgfkeys{/bodeplot/nichols/.cd, reset}
2288       \pgfkeys{/bodeplot/nichols/.cd, #1}
2289       \pgfkeysgetvalue{/bodeplot/nichols/@axes}{\bp@axes}
2290       \pgfkeysgetvalue{/bodeplot/nichols/@commands}{\bp@commands}
2291       \pgfkeysgetvalue{/bodeplot/nichols/@tikz}{\bp@tikz}
2292       \pgfkeysgetvalue{/bodeplot/nichols/@prefix}{\bp@user@prefix}
2293       \bp@tf@new@to@legacy{#2}
2294     }{%
2295       \bp@parse@N@opt{#1}
2296       \edef\bp@legacy{#2}
2297       \edef\bp@domain@start{#3}
2298       \edef\bp@domain@end{#4}
2299     }%
2300     \gdef\bp@mag{}\gdef\bp@ph{}%
2301     \edef\bp@cmd{\noexpand\begin{tikzpicture}
2302       [\unexpanded\expandafter{\bp@tikz}]]\bp@cmd
2303     \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2304     \edef\bp@cmd{\noexpand\begin{axis}[
2305       bp@ph@x@label,
2306       bp@style,
2307       domain=\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale,
2308       height=5cm,
2309       ylabel={Gain (dB)},
2310       samples=500,
2311       \unexpanded\expandafter{\bp@axes}
2312     ]}
2313     \bp@cmd
2314     \edef\bp@cmd{\noexpand\addplot
2315       [variable=t, thick, trig format plots=rad,
2316       \unexpanded\expandafter{\bp@plot}]]%
2317     \ifpgfarg
2318       \bp@cmd ( {\n@mod{\bp@ph}{2*pi*\bp@ph@scale}} , {\bp@mag} );
2319     \bp@commands
2320   \else
2321     \stepcounter{bp@gnuplot@id}
2322     \bp@cmd gnuplot [raw gnuplot, bp@gnu@prefix]
2323     { set table $metal;

```

```

2324         set logscale x 10;
2325         set dummy t;
2326         set samples \pgfkeysvalueof{/pgfplots/samples};
2327         set trange [\bp@domain@start*\bp@freq@scale:\bp@domain@end*\bp@freq@scale];
2328         plot '+' using (\bp@mag) : ((\bp@ph)/(\bp@ph@scale));
2329         unset logscale x;
2330         set table $meta2;
2331         plot "$meta1" using ($1):($2) smooth unwrap;
2332         set table "\bp@prefix\arabic{\bp@gnuplot@id}.table";
2333         plot "$meta2" using ($2*\bp@ph@scale):($1);
2334     };
2335     \bp@commands
2336 \fi
2337 \end{axis}
2338 \end{tikzpicture}
2339 \fi
2340 }

```

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

```

2341 \AtBeginDocument{
2342   \if@babel
2343     \let\Orig@NicholsTF\NicholsTF
2344     \renewcommand{\NicholsTF}{%
2345       \expandafter\shorthandoff\expandafter{\bp@short@list}
2346       \NicholsTF@Shorthandoff
2347     }
2348     \NewDocumentCommand{\NicholsTF@Shorthandoff}{ 0{} m G{} G{} }{%
2349       \Orig@NicholsTF[#1]{#2}{#3}{#4}
2350       \expandafter\shorthandon\expandafter{\bp@short@list}
2351     }
2352     \AddToHook{env/NicholsChart/begin}{%
2353       \expandafter\shorthandoff\expandafter{\bp@short@list}
2354     }
2355     \AddToHook{env/NicholsChart/end}{%
2356       \expandafter\shorthandon\expandafter{\bp@short@list}
2357     }
2358   \fi
2359 }

```

NicholsChart

```

2360 \NewDocumentEnvironment{NicholsChart}{ >\TrimSpaces}0{} >\TrimSpaces} G{} >\TrimSpaces}
2361   \pgfutil@ifempty{#2}{%
2362     \pgfkeys{/bodeplot/env/.cd, reset}
2363     \pgfkeys{/bodeplot/env/.cd, #1}
2364     \pgfkeysgetvalue{/bodeplot/env/@tikz}{\bp@tikz}
2365     \pgfkeysgetvalue{/bodeplot/env/@prefix}{\bp@user@prefix}
2366   }{%
2367     \bp@parse@env@opt{#1}
2368     \edef\bp@domain@start{#2}
2369     \edef\bp@domain@end{#3}
2370   }
2371   \edef\bp@cmd{\noexpand\begin{tikzpicture}
2372     [\unexpanded\expandafter{\bp@tikz}]\bp@cmd
2373   \edef\bp@cmd{\noexpand\begin{axis}[
2374     bp@ph@x@label,
2375     bp@style,
2376     domain=\bp@domain@start:\bp@domain@end,
2377     height=5cm,
2378     ylabel={Gain (dB)},
2379     \unexpanded\expandafter{\bp@axes}%
2380   ]}\bp@cmd
2381   #4
2382   \end{axis}
2383   \end{tikzpicture}

```

2384 }{}

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

```

2385 \NewDocumentCommand{\addNicholsZPKChart}{>{\TrimSpaces}0}{>{\TrimSpaces} m }{%
2386   \edef\bp@spec@arg{#2}%
2387   \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2388     \bp@zpk@new@to@legacy{\bp@spec@arg}}{\edef\bp@legacy{\bp@spec@arg}}%
2389   \gdef\bp@mag{}\gdef\bp@ph{}%
2390   \bp@ZPK@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2391   \if@pgfarg
2392     \edef\bp@cmd{\noexpand\addplot
2393       [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
2394         variable=t, thick, trig format plots=rad, #1]}%
2395     \bp@cmd ( {\bp@ph} , {\bp@mag} );
2396   \else
2397     \stepcounter{bp@gnuplot@id}%
2398     \addplot [thick, #1] gnuplot [raw gnuplot, bp@gnu@prefix] {%
2399       set table $meta;
2400       set logscale x 10;
2401       set dummy t;
2402       set samples \pgfkeysvalueof{/pgfplots/samples};
2403       set trange [\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale];
2404       plot '+' using (\bp@mag) : ((\bp@ph)/(\bp@ph@scale));
2405       unset logscale x;
2406       set table "\bp@prefix\arabic{bp@gnuplot@id}.table";
2407       plot "$meta" using ($2*\bp@ph@scale):($1);
2408     };
2409   \fi
2410 }

```

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

```

2411 \NewDocumentCommand{\addNicholsTFChart}{>{\TrimSpaces}0}{>{\TrimSpaces} m }{%
2412   \edef\bp@spec@arg{#2}%
2413   \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2414     \bp@tf@new@to@legacy{\bp@spec@arg}}{\edef\bp@legacy{\bp@spec@arg}}%
2415   \gdef\bp@mag{}\gdef\bp@ph{}%
2416   \bp@TF@plot{\bp@mag}{\bp@ph}{\bp@legacy}%
2417   \if@pgfarg
2418     \edef\bp@cmd{\noexpand\addplot
2419       [domain=\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale,
2420         variable=t, thick, trig format plots=rad, #1]}%
2421     \bp@cmd ( {\n@mod{\bp@ph}{2*pi*\bp@ph@scale}} , {\bp@mag} );
2422   \else
2423     \stepcounter{bp@gnuplot@id}%
2424     \addplot [thick, #1] gnuplot [raw gnuplot, bp@gnu@prefix] {%
2425       set table $meta1;
2426       set logscale x 10;
2427       set dummy t;
2428       set samples \pgfkeysvalueof{/pgfplots/samples};
2429       set trange [\bp@freq@scale*\pgfkeysvalueof{/pgfplots/domain}*\bp@freq@scale];
2430       plot '+' using (\bp@mag) : ((\bp@ph)/(\bp@ph@scale));
2431       unset logscale x;
2432       set table $meta2;
2433       plot "$meta1" using ($1):($2) smooth unwrap;
2434       set table "\bp@prefix\arabic{bp@gnuplot@id}.table";
2435       plot "$meta2" using ($2*\bp@ph@scale):($1);
2436     };
2437   \fi
2438 }

```

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.

```
2439 \NewDocumentCommand{\PoleZeroMapZPK}{>\TrimSpaces}0{>\TrimSpaces} m }{%
2440 \edef\bp@spec@arg{#2}%
2441 \expandafter\bp@contains@equal\expandafter{\bp@spec@arg}{%
2442   \pgfkeys{/bodeplot/pzmap/.cd, reset}
2443   \pgfkeys{/bodeplot/pzmap/.cd, #1}
2444   \pgfkeysgetvalue{/bodeplot/pzmap/@axes}{\bp@axes}
2445   \pgfkeysgetvalue{/bodeplot/pzmap/@plot}{\bp@plot}
2446   \pgfkeysgetvalue{/bodeplot/pzmap/@commands}{\bp@commands}
2447   \pgfkeysgetvalue{/bodeplot/pzmap/@tikz}{\bp@tikz}
2448   \pgfkeysgetvalue{/bodeplot/pzmap/@prefix}{\bp@user@prefix}
2449   \pgfkeysgetvalue{/bodeplot/pzmap/@scale}{\bp@scale}
2450   \bp@zpk@new@to@legacy{\bp@spec@arg}
2451 }{%
2452   \bp@parse@N@opt{#1}
2453   \edef\bp@legacy{\bp@spec@arg}
2454 }%
2455 \ifnum\pdf@strcmp{\bp@scale}{log}=0
2456   \bp@min@real@ZPK{\bp@legacy}
2457   \bp@min@im@ZPK{\bp@legacy}
2458   \pgfkeys{/pgf/fpu=true}
2459   \ifx\bp@has@positive@values\bp@min@false
2460     \xdef\bp@PZZPK@ticksXPos{0}
2461   \else
2462     \pgfmathparse{max(\bp@max@re@pos@pow@10 - \bp@min@re@pow@10 + 1, 1)}
2463     \pgfmathfloattoint{\pgfmathresult}
2464     \xdef\bp@PZZPK@ticksXPos{\pgfmathresult}
2465   \fi
2466   \ifx\bp@has@negative@values\bp@min@false
2467     \xdef\bp@PZZPK@ticksXNeg{0}
2468   \else
2469     \pgfmathparse{max(\bp@max@re@neg@pow@10 - \bp@min@re@pow@10 + 1, 1)}
2470     \pgfmathfloattoint{\pgfmathresult}
2471     \xdef\bp@PZZPK@ticksXNeg{\pgfmathresult}
2472   \fi
2473   \pgfkeys{/pgf/fpu=false}
2474   \def\PoleZeroMapZPK@formatXTick##1{%
2475     \pgfmathtruncatemacro{\PoleZeroMapZPK@tick}{##1}%
2476     \ifnum\PoleZeroMapZPK@tick=0
2477       $0$
2478     \else
2479       \pgfmathtruncatemacro{\PoleZeroMapZPK@exp}
2480       {\bp@min@re@pow@10 + abs(\PoleZeroMapZPK@tick) - 1}%
2481       \ifnum\PoleZeroMapZPK@tick>0
2482         $10^{\PoleZeroMapZPK@exp}$%
2483       \else
2484         $-10^{\PoleZeroMapZPK@exp}$%
2485       \fi
2486     \fi
2487   }
2488   \def\PoleZeroMapZPK@formatYTick##1{%
2489     \pgfmathtruncatemacro{\PoleZeroMapZPK@tick}{##1}%
2490     \ifnum\PoleZeroMapZPK@tick=0
2491       $0$
2492     \else
2493       \pgfmathtruncatemacro{\PoleZeroMapZPK@exp}
```

```

2494      {\bp@min@im@pow@10 + abs(\PoleZeroMapZPK@tick) - 1}%
2495      \ifnum\PoleZeroMapZPK@tick>0
2496      $10^{\PoleZeroMapZPK@exp}$%
2497      \else
2498      $-10^{\PoleZeroMapZPK@exp}$%
2499      \fi
2500    \fi
2501  }
2502  \def\PoleZeroMapZPK@xticklabel{\PoleZeroMapZPK@formatXTick{\tick}}
2503  \def\PoleZeroMapZPK@yticklabel{\PoleZeroMapZPK@formatYTick{\tick}}
2504  \fi
2505  \edef\bp@cmd{\noexpand\begin{tikzpicture}
2506    [\unexpanded\expandafter{\bp@tikz}]}
2507  \bp@cmd
2508    \ifnum\pdf@strcmp{\bp@scale}{log}=0
2509    \edef\bp@cmd{\noexpand\begin{axis}[
2510      xlabel={\Re$},
2511      ylabel={\Im$},
2512      axis lines=center,
2513      grid=major,
2514      height=6cm,
2515      enlarge x limits=0.2,
2516      enlarge y limits=0.2,
2517      xtick distance=1,
2518      ytick distance=1,
2519      xticklabel=\noexpand\PoleZeroMapZPK@xticklabel,
2520      yticklabel=\noexpand\PoleZeroMapZPK@yticklabel,
2521      x filter/.expression={abs(x) < \bp@min@re@threshold@result ?
2522      0 : (x >= 0 ? (log10(max(min(x, 1e100), 1e-
2523      100)) - \bp@min@re@pow@10 + 1) :
2524      (-log10(max(min(-x, 1e100), 1e-100)) + \bp@min@re@pow@10 - 1))},
2525      y filter/.expression={abs(y) < \bp@min@im@threshold@result ?
2526      0 : (y >= 0 ? (log10(max(min(y, 1e100), 1e-
2527      100)) - \bp@min@im@pow@10 + 1) :
2528      (-log10(max(min(-y, 1e100), 1e-100)) + \bp@min@im@pow@10 - 1))},
2529      \unexpanded\expandafter{\bp@axes}
2530    ]}
2531  \else
2532  \edef\bp@cmd{\noexpand\begin{axis}[
2533    xlabel={\Re$},
2534    ylabel={\Im$},
2535    axis lines=center,
2536    grid=major,
2537    height=6cm,
2538    enlarge x limits=0.2,
2539    enlarge y limits=0.2,
2540    \unexpanded\expandafter{\bp@axes}
2541  ]}
2542  \fi
2543  \bp@cmd
2544    \foreach \feature/\values in \bp@legacy {
2545      \ifnum\pdf@strcmp{\feature}{z}=0
2546      \foreach \z in \values {
2547        \foreach \y [count=\zcnt] in \z {
2548          \ifnum\zcnt=1
2549            \xdef\bp@zre{\y}
2550          \fi
2551          \ifnum\zcnt=2
2552            \xdef\bp@zim{\y}
2553          \fi
2554          \xdef\bp@Last@Loop@z{\zcnt}
2555        }
2556      }
2557      \ifnum\bp@Last@Loop@z=1

```

```

2555         \xdef\bp@zim{0}
2556     \fi
2557     \edef\bp@cmd{\noexpand\addplot
2558     [only marks, mark=o, mark size=3pt, thick, blue,
2559     \unexpanded\expandafter{\bp@plot}]}
2560     \bp@cmd coordinates {(\bp@zre,\bp@zim)};
2561 }
2562 \fi
2563 \ifnum\pdf@strcmp{\feature}{p}=0
2564     \foreach \p in \values {
2565         \foreach \y [count=\pcnt] in \p {
2566             \ifnum\pcnt=1
2567                 \xdef\bp@pre{\y}
2568             \fi
2569             \ifnum\pcnt=2
2570                 \xdef\bp@pim{\y}
2571             \fi
2572             \xdef\bp@Last@Loop@p{\pcnt}
2573         }
2574         \ifnum\bp@Last@Loop@p=1
2575             \xdef\bp@pim{0}
2576         \fi
2577         \edef\bp@cmd{\noexpand\addplot
2578         [only marks, mark=x, mark size=3pt, thick, red,
2579         \unexpanded\expandafter{\bp@plot}]}
2580         \bp@cmd coordinates {(\bp@pre,\bp@pim)};
2581     }
2582 \fi
2583 }
2584 \bp@commands
2585 \end{axis}
2586 \end{tikzpicture}
2587 }

```

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

```

2588 \AtBeginDocument{%
2589     \if@babel
2590         \let\Orig@PoleZeroMapZPK\PoleZeroMapZPK
2591         \renewcommand{\PoleZeroMapZPK}{%
2592             \expandafter\shorthandoff\expandafter{\bp@short@list}
2593             \PoleZeroMapZPK@Shorthandoff
2594         }
2595         \newcommand{\PoleZeroMapZPK@Shorthandoff}[2][]{%
2596             \Orig@PoleZeroMapZPK[#1]{#2}
2597             \expandafter\shandon\expandafter{\bp@short@list}
2598         }
2599     \fi
2600 }

```

\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.

```

2601 \NewDocumentCommand{\PoleZeroMapTF}{ >{\TrimSpaces}0{ } >{\TrimSpaces} m }{%
2602     \bp@tf@to@zpk{#2}{\bp@tmp@zpk}{\bp@tmp@status}
2603     \ifnum\pdf@strcmp{\bp@tmp@status}{true}=0
2604         \PoleZeroMapZPK[#1]{\bp@tmp@zpk}
2605     \else
2606         \PackageError{bodeplot}{Python with scipy and numpy is required to convert TF to ZPK
2607         zero map. Please ensure Python is installed and accessible via --shell-
2608         escape.}{}
2609     \fi
2610 }

```

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

```

2609 \AtBeginDocument{%
2610   \if@babel
2611   \let\Orig@PoleZeroMapTF\PoleZeroMapTF
2612   \renewcommand{\PoleZeroMapTF}{%
2613     \expandafter\shorthandoff\expandafter{\bp@short@list}
2614     \PoleZeroMapTF@Shorthandoff
2615   }
2616   \newcommand{\PoleZeroMapTF@Shorthandoff}[2][\]{%
2617     \Orig@PoleZeroMapTF[#1]{#2}
2618     \expandafter\shorthandon\expandafter{\bp@short@list}
2619   }
2620   \fi
2621 }

```

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`.

```

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

```

```

2664         \else
2665             \gdef\bp@has@negative@values{true}
2666             \pgfkeys{/pgf/fpu=true}
2667             \pgfmathparse{\abs@valuefloat > \bp@max@re@neg@float ? 1 : 0}
2668             \pgfmathfloattoint{\pgfmathresult}
2669             \pgfkeys{/pgf/fpu=false}
2670             \ifnum\pgfmathresult=1
2671                 \global\let\bp@max@re@neg@float=\abs@valuefloat
2672             \fi
2673         \fi
2674         \pgfkeys{/pgf/fpu=true}
2675         \pgfmathparse{\abs@valuefloat > \bp@max@re@float ? 1 : 0}
2676         \pgfmathfloattoint{\pgfmathresult}
2677         \pgfkeys{/pgf/fpu=false}
2678         \ifnum\pgfmathresult=1
2679             \global\let\bp@max@re@float=\abs@valuefloat
2680             \xdef\bp@max@re@value{\abs@value}
2681         \fi
2682         \ifx\bp@min@threshold@found\@bp@min@false
2683             \xdef\bp@min@re@threshold@result{\abs@value}
2684             \global\let\bp@min@thresh@float=\abs@valuefloat
2685             \gdef\bp@min@threshold@found{true}
2686         \else
2687             \pgfkeys{/pgf/fpu=true}
2688             \pgfmathparse{\abs@valuefloat < \bp@min@thresh@float ? 1 : 0}
2689             \pgfmathfloattoint{\pgfmathresult}
2690             \pgfkeys{/pgf/fpu=false}
2691             \ifnum\pgfmathresult=1
2692                 \xdef\bp@min@re@threshold@result{\abs@value}
2693                 \global\let\bp@min@thresh@float=\abs@valuefloat
2694             \fi
2695         \fi
2696     \fi
2697 \fi
2698 \fi
2699 }
2700 }
2701 \fi
2702 \ifnum\pdf@strcmp{\feature}{p}=0
2703     \foreach \p in \values {
2704         \foreach \y [count=\pcnt] in \p {
2705             \ifnum\pcnt=1
2706                 \pgfkeys{/pgf/fpu=true}
2707                 \pgfmathparse{abs(\y)}
2708                 \let\abs@valuefloat=\pgfmathresult
2709                 \pgfmathfloattofixed{\abs@valuefloat}
2710                 \edef\abs@value{\pgfmathresult}
2711                 \pgfkeys{/pgf/fpu=false}
2712                 \ifnum\pdf@strcmp{\abs@value}{0}=0\else
2713                     \ifnum\pdf@strcmp{\abs@value}{0.0}=0\else
2714                         \pgfkeys{/pgf/fpu=true}
2715                         \pgfmathparse{\y >= 0 ? 1 : 0}
2716                         \pgfmathfloattoint{\pgfmathresult}
2717                         \pgfkeys{/pgf/fpu=false}
2718                         \ifnum\pgfmathresult=1
2719                             \gdef\bp@has@positive@values{true}
2720                             \pgfkeys{/pgf/fpu=true}
2721                             \pgfmathparse{\abs@valuefloat > \bp@max@re@pos@float ? 1 : 0}
2722                             \pgfmathfloattoint{\pgfmathresult}
2723                             \pgfkeys{/pgf/fpu=false}
2724                             \ifnum\pgfmathresult=1
2725                                 \global\let\bp@max@re@pos@float=\abs@valuefloat
2726                             \fi

```

```

2727         \else
2728             \gdef\bp@has@negative@values{true}
2729             \pgfkeys{/pgf/fpu=true}
2730             \pgfmathparse{\abs@valuefloat > \bp@max@re@neg@float ? 1 : 0}
2731             \pgfmathfloattoint{\pgfmathresult}
2732             \pgfkeys{/pgf/fpu=false}
2733             \ifnum\pgfmathresult=1
2734                 \global\let\bp@max@re@neg@float=\abs@valuefloat
2735             \fi
2736         \fi
2737         \ifx\bp@min@threshold@found\@bp@min@false
2738             \xdef\bp@min@re@threshold@result{\abs@value}
2739             \global\let\bp@min@thresh@float=\abs@valuefloat
2740             \gdef\bp@min@threshold@found{true}
2741         \else
2742             \pgfkeys{/pgf/fpu=true}
2743             \pgfmathparse{\abs@valuefloat < \bp@min@thresh@float ? 1 : 0}
2744             \pgfmathfloattoint{\pgfmathresult}
2745             \pgfkeys{/pgf/fpu=false}
2746             \ifnum\pgfmathresult=1
2747                 \xdef\bp@min@re@threshold@result{\abs@value}
2748                 \global\let\bp@min@thresh@float=\abs@valuefloat
2749             \fi
2750         \fi
2751     \fi
2752 \fi
2753 \fi
2754 }
2755 }
2756 \fi
2757 }
2758 \ifx\bp@min@threshold@found\@bp@min@false
2759     \gdef\bp@min@re@threshold@result{0.01}
2760 \fi
2761 \xdef\bp@min@threshold@result{\bp@min@re@threshold@result}
2762 \pgfkeys{/pgf/fpu=true}
2763 \pgfmathparse{log10(\bp@min@re@threshold@result)}
2764 \let\log@result=\pgfmathresult
2765 \pgfmathparse{\log@result + 1e-5}
2766 \let\log@adjusted=\pgfmathresult
2767 \pgfmathparse{floor(\log@adjusted)}
2768 \pgfmathfloattofixed{\pgfmathresult}
2769 \xdef\bp@min@re@pow@10{\pgfmathresult}
2770 \xdef\bp@min@pow@10{\bp@min@re@pow@10}
2771 \ifx\bp@has@positive@values\@bp@min@false
2772     \xdef\bp@max@re@pos@pow@10{\bp@min@re@pow@10}
2773 \else
2774     \pgfmathparse{log10(max(\bp@max@re@pos@float,1e-100))}
2775     \let\log@bp@max@re@pos=\pgfmathresult
2776     \pgfmathparse{\log@bp@max@re@pos + 1e-5}
2777     \let\log@bp@max@re@pos@adjusted=\pgfmathresult
2778     \pgfmathparse{ceil(\log@bp@max@re@pos@adjusted)}
2779     \pgfmathfloattoint{\pgfmathresult}
2780     \xdef\bp@max@re@pos@pow@10{\pgfmathresult}
2781 \fi
2782 \ifx\bp@has@negative@values\@bp@min@false
2783     \xdef\bp@max@re@neg@pow@10{\bp@min@re@pow@10}
2784 \else
2785     \pgfmathparse{log10(max(\bp@max@re@neg@float,1e-100))}
2786     \let\log@bp@max@re@neg=\pgfmathresult
2787     \pgfmathparse{\log@bp@max@re@neg + 1e-5}
2788     \let\log@bp@max@re@neg@adjusted=\pgfmathresult
2789     \pgfmathparse{ceil(\log@bp@max@re@neg@adjusted)}

```

```

2790 \pgfmathfloattoint{\pgfmathresult}
2791 \xdef\bp@max@re@neg@pow@10{\pgfmathresult}
2792 \fi
2793 \pgfmathparse{max(\bp@max@re@pos@float > 0 ?
2794 \bp@max@re@pos@float : 0, \bp@max@re@neg@float > 0 ? \bp@max@re@neg@float : 0)}
2795 \let\bp@max@re@valuefloat=\pgfmathresult
2796 \pgfmathparse{\bp@max@re@valuefloat > 0 ?
2797 \bp@max@re@valuefloat : \bp@min@re@threshold@result}
2798 \let\bp@max@re@valuefloat=\pgfmathresult
2799 \pgfmathparse{log10(max(\bp@max@re@valuefloat,1e-100))}
2800 \let\log@bp@max@re=\pgfmathresult
2801 \pgfmathparse{\log@bp@max@re + 1e-5}
2802 \let\log@bp@max@re@adjusted=\pgfmathresult
2803 \pgfmathparse{ceil(\log@bp@max@re@adjusted)}
2804 \pgfmathfloattoint{\pgfmathresult}
2805 \xdef\bp@max@re@pow@10{\pgfmathresult}
2806 \pgfkeys{/pgf/fpu=false}
2807 }

```

`\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`.

```

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

```

```

2847         \pgfmathfloattoint{\pgfmathresult}
2848         \pgfkeys{/pgf/fpu=false}
2849         \ifnum\pgfmathresult=1
2850             \xdef\bp@min@im@threshold@result{\abs@value}
2851             \global\let\bp@min@im@thresh@float=\abs@valuefloat
2852         \fi
2853     \fi
2854 \fi
2855 \fi
2856 \fi
2857 }
2858 }
2859 \fi
2860 \ifnum\pdf@strcmp{\feature}{p}=0
2861     \foreach \p in \values {
2862         \foreach \y [count=\pcnt] in \p {
2863             \ifnum\pcnt=2
2864                 \pgfkeys{/pgf/fpu=true}
2865                 \pgfmathparse{abs(\y)}
2866                 \let\abs@valuefloat=\pgfmathresult
2867                 \pgfmathfloattofixed{\abs@valuefloat}
2868                 \edef\abs@value{\pgfmathresult}
2869                 \pgfkeys{/pgf/fpu=false}
2870                 \ifnum\pdf@strcmp{\abs@value}{0}=0\else
2871                     \ifnum\pdf@strcmp{\abs@value}{0.0}=0\else
2872                         \ifx\bp@min@im@threshold@found\@bp@min@false
2873                             \xdef\bp@min@im@threshold@result{\abs@value}
2874                             \global\let\bp@min@im@thresh@float=\abs@valuefloat
2875                             \gdef\bp@min@im@threshold@found{true}
2876                         \else
2877                             \pgfkeys{/pgf/fpu=true}
2878                             \pgfmathparse{\abs@valuefloat < \bp@min@im@thresh@float ?
2879                                 1 : 0}
2880                             \pgfmathfloattoint{\pgfmathresult}
2881                             \pgfkeys{/pgf/fpu=false}
2882                             \ifnum\pgfmathresult=1
2883                                 \xdef\bp@min@im@threshold@result{\abs@value}
2884                                 \global\let\bp@min@im@thresh@float=\abs@valuefloat
2885                             \fi
2886                         \fi
2887                     \fi
2888                 \fi
2889             }
2890         }
2891     }
2892 \fi
2893 }
2894 \ifx\bp@min@im@threshold@found\@bp@min@false
2895     \gdef\bp@min@im@threshold@result{0.01}
2896 \fi
2897 \pgfkeys{/pgf/fpu=true}
2898 \pgfmathparse{log10(\bp@min@im@threshold@result)}
2899 \let\log@result=\pgfmathresult
2900 \pgfmathparse{\log@result + 1e-5}
2901 \let\log@adjusted=\pgfmathresult
2902 \pgfmathparse{floor(\log@adjusted)}
2903 \pgfmathfloattofixed{\pgfmathresult}
2904 \xdef\bp@min@im@pow@10{\pgfmathresult}
2905 \xdef\bp@min@im@pow@10{\bp@min@im@pow@10}
2906 \pgfmathparse{\bp@max@im@float > 0 ?
2907 \bp@max@im@float : \bp@min@im@threshold@result}
2908 \let\bp@max@im@valuefloat=\pgfmathresult
2909 \pgfmathparse{log10(max(\bp@max@im@valuefloat,1e-100))}

```

```

2910 \let\log@bp@max@im=\pgfmathresult
2911 \pgfmathparse{\log@bp@max@im + 1e-5}
2912 \let\log@bp@max@im@adjusted=\pgfmathresult
2913 \pgfmathparse{ceil(\log@bp@max@im@adjusted)}
2914 \pgfmathfloattoint{\pgfmathresult}
2915 \xdef\bp@max@im@pow@10{\pgfmathresult}
2916 \pgfkeys{/pgf/fpu=false}
2917 }

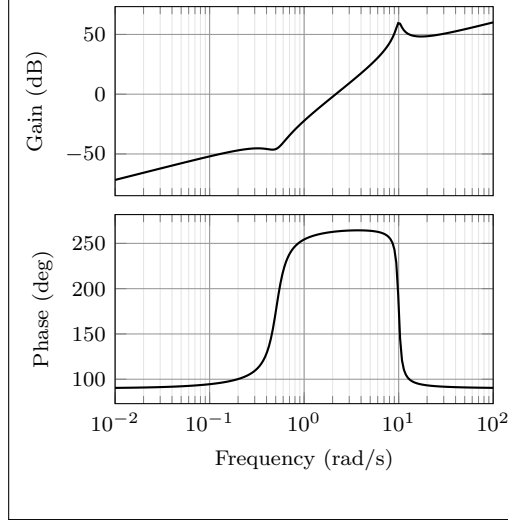
```

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)

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

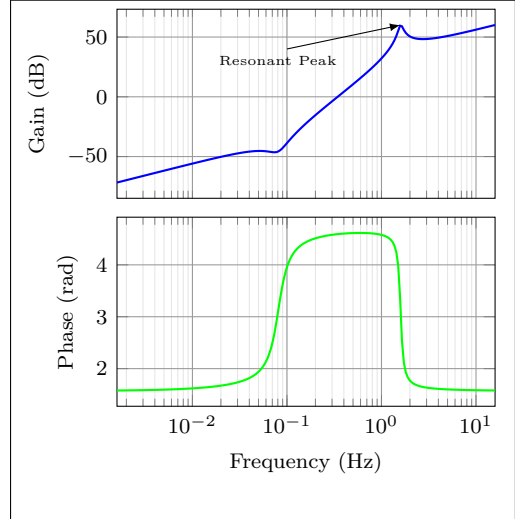
Bode plot in ZPK format



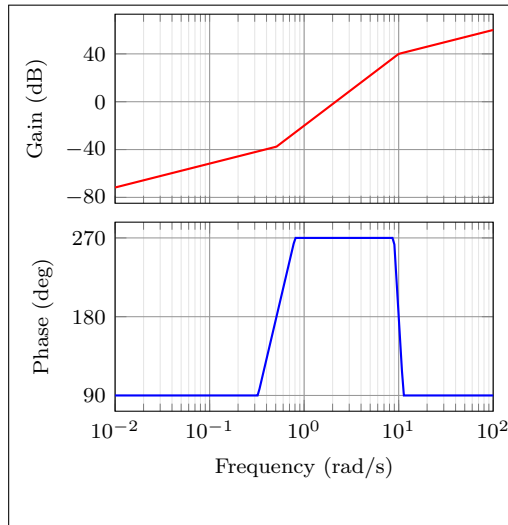
```
\BodeZPK{
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10
}
{0.01}
{100}
```

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)}
```



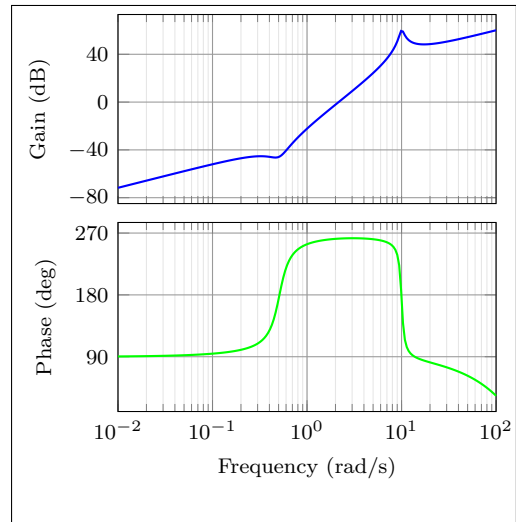
Linear approximation with customization



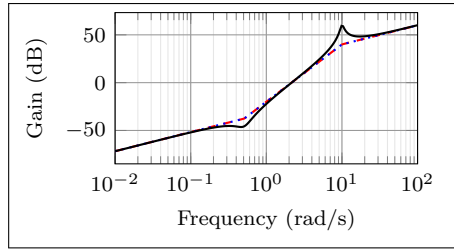
```
\BodeZPK[
  plot/mag/{red,thick},
  plot/ph/{blue,thick},
  axes/mag/{ytick distance=40},
  axes/ph/{ytick distance=90},
  approx/linear
]{
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10
}
{0.01}
{100}
```

Plot with delay and customization

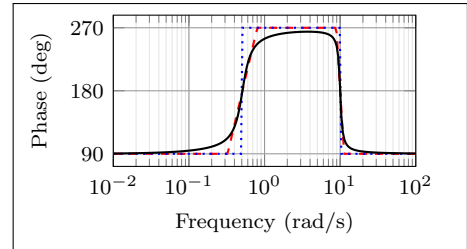
```
\BodeZPK[
  plot/mag/{blue,thick},
  plot/ph/{green,thick},
  axes/mag/{ytick distance=40},
  axes/ph/{ytick distance=90},
  d/0.01
]{
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10
}
{0.01}
{100}
```



Individual gain and phase plots with more customization



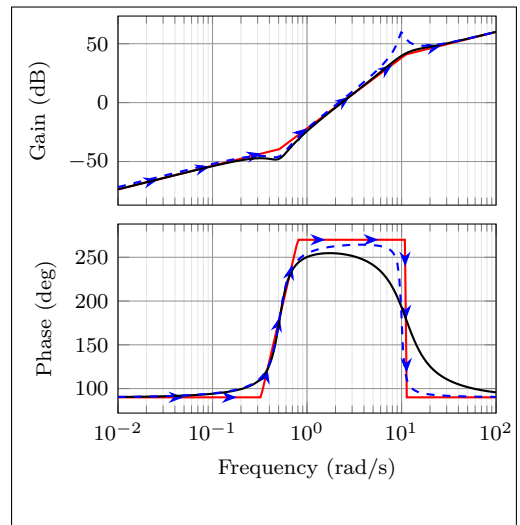
```
\begin{BodeMagPlot}[
  axes/{height=2cm,
  width=4cm}
]{0.01}
{100}
\addBodeZPKPlots[
  true/{black,thick},
  linear/{red,dashed,thick},
  asymptotic/{blue,dotted,thick}
]
{magnitude}
{
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10
}
}\end{BodeMagPlot}
```



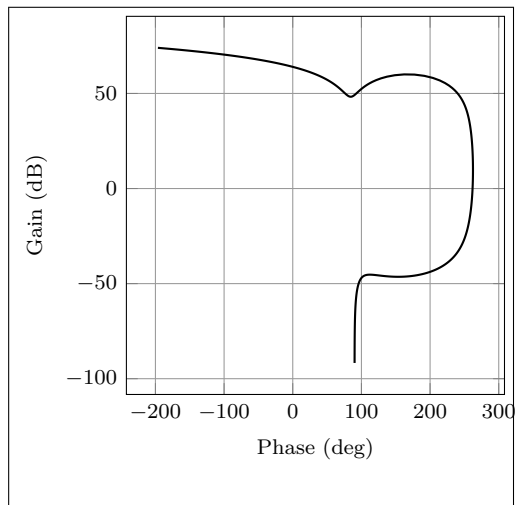
```
\begin{BodePhPlot}[
  height=2cm,
  width=4cm,
  ytick distance=90
]{0.01}
{100}
\addBodeZPKPlots[
  true/{black,thick},
  linear/{red,dashed,thick},
  asymptotic/{blue,dotted,thick}
]
{phase}
{
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10
}
}\end{BodePhPlot}
```

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/{{-0.5,-10},{-0.5,10}},
  k/10
}
\addBodePlot[black,thick]{zpk}{
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10
}
\addBodePlot[blue,dashed]{tf}{
  num/{10,2,2.6,0},
  den/{1,1,100.25}
}
}\end{BodePlot}
```



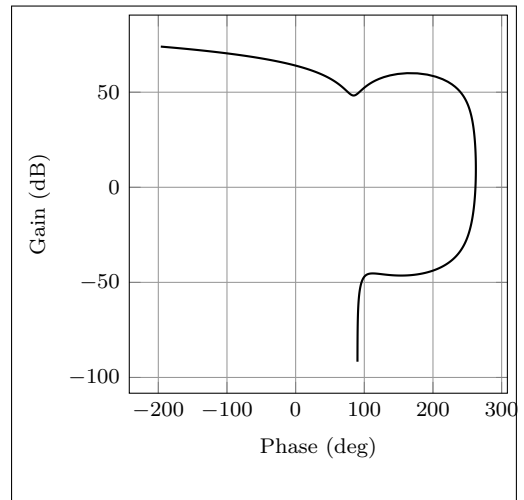
Nichols chart



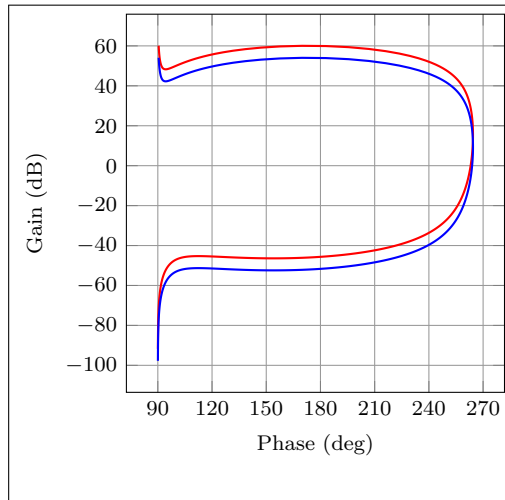
```
\NicholsZPK[samples=1000]
{
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10,
  d/0.01
}
{0.001}
{500}
```

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}
```



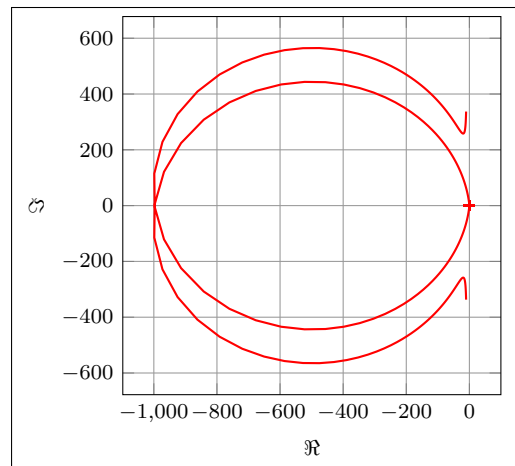
Multiple Nichols charts with customization



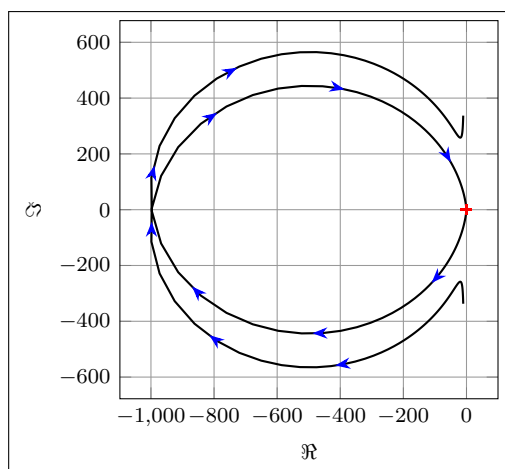
```
\begin{NicholsChart}[
  ytick distance=20,
  xtick distance=30
]
{0.001}
{100}
\addNicholsZPKChart [red,samples=1000] {
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/10
}
\addNicholsZPKChart [blue,samples=1000] {
  z/{0,{-0.1,-0.5},{-0.1,0.5}},
  p/{{-0.5,-10},{-0.5,10}},
  k/5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



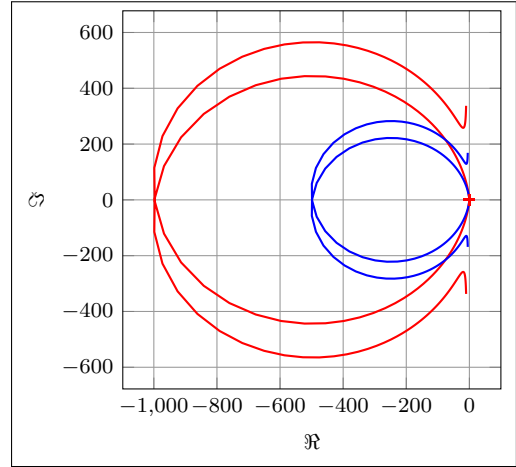
Nyquist plot in TF format with arrows



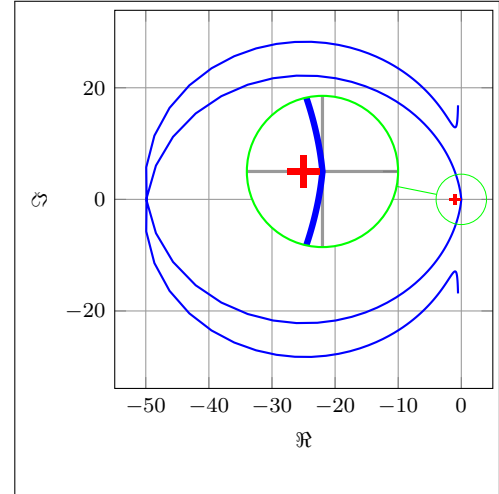
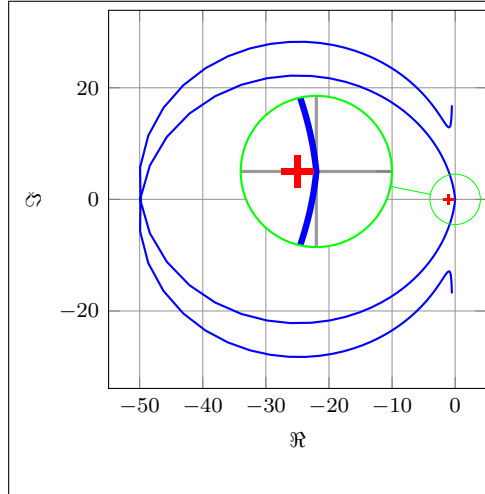
```
\NyquistTF[
  plot/{%
    samples=1000,
    postaction=decorate,
    decoration={
      markings,
      mark=between positions 0.1 and 0.9 step 5em with {
        \arrow{Stealth [length=2mm, blue]}
      }
    }
  }
]
{
  num/{10,2,2.6,0},
  den/{1,1,100.25}
}
{-30}
{30}
```

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}
```



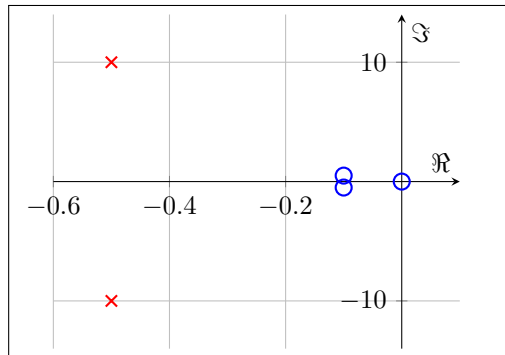
Nyquist plots with additional commands, using two different macros



```
\begin{NyquistPlot}[
tikz/{
  spy using outlines={%
    circle,
    magnification=3,
    connect spies,
    size=2cm
  }
}
]{-30}{30}
\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}
```

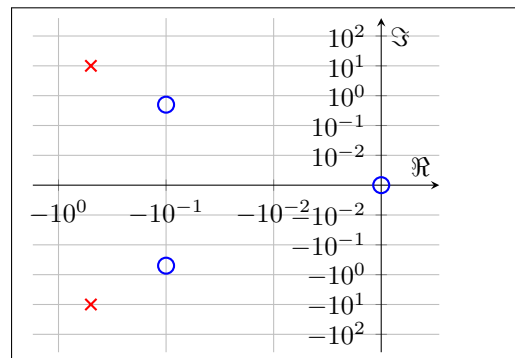
Pole-zero map



```
\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	2667, 2671, 2675,	\bp@axes 154,
\bodeplot <u>230</u>	2679, 2684, 2688,	157, 158, 160, 221,
\bodeplot/add <u>556</u>	2693, 2708, 2709,	222, 224, 1188,
\bodeplot/combinedenv	2721, 2725, 2730,	1227, 1920, 1926,
. <u>383</u>	2734, 2739, 2743,	1935, 1965, 1987,
\bodeplot/env <u>135</u>	2748, 2825, 2826,	2027, 2049, 2158,
\bodeplot/nichols . . <u>689</u>	2832, 2836, 2841,	2166, 2175, 2207,
\bodeplot/nyquist . . <u>636</u>	2845, 2851, 2866,	2229, 2289, 2311,
\bodeplot/pzmap . . . <u>742</u>	2867, 2874, 2878, 2884	2379, 2444, 2527, 2538
\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
. . . . 2459, 2466,	\addBodeZPKPlots . . .	. . . 209, 210, 354,
2624, 2682, 2737,	 <u>1057</u> , <u>1137</u>	355, 494, 495, 615,
2758, 2771, 2782,	\addNicholsTFChart . <u>2411</u>	616, 668, 669, 721, 722
2810, 2839, 2872, 2894	\addNicholsZPKChart <u>2385</u>	\bp@cmd 998, 1000, 1004,
\@declutterargfalse . . 5	\addNyquistTFPlot . . <u>2107</u>	1014, 1078, 1081,
\@declutterargtrue . . . 7	\addNyquistZPKPlot . <u>2085</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, 1977, 1978,
__bp_find_separator:	\BodeZPK . . . 865, 958, 960	1980, 1989, 1991,
. 1387, 1477	\BodeZPK@Shorthandoff	1995, 2001, 2039,
__bp_parse_complex_with_i:	 945, 947	2040, 2042, 2051,
. 1364, 1381	\bool_if:NTF 1389	2053, 2057, 2063,
__bp_parse_pure_imaginary:	\bool_lazy_or:nnT . . 1461	2092, 2095, 2099,
. 1391, 1395	\bool_new:N 1353	2101, 2114, 2117,
__bp_parse_real_only:	\bool_set_false:N . . 1479	2121, 2123, 2150,
. 1365, 1373	\bool_set_true:N . . 1465	2151, 2152, 2159,
__bp_process_item:n	\bp@add . . . <u>1567</u> , 1589,	2219, 2220, 2222,
. 1449, 1482	1590, 1593, 1594,	2231, 2232, 2236,
__bp_set_signed_value:NN	1596, 1597, 1607,	2240, 2301, 2302,
. 1398, 1402	1608, 1611, 1612,	2304, 2313, 2314,
__bp_split_at_separator:	1614, 1615, 1623,	2318, 2322, 2371,
. 1390, 1498	1624, 1627, 1628,	2372, 2373, 2380,
__bp_unwrap_braces:N	1630, 1631, 1649, 1650	2392, 2395, 2418,
1375, 1414, 1425,	\bp@add@0 . . . 560, 565,	2421, 2505, 2507,
1430, 1438, 1506, 1507	566, 568, 571, 572,	2509, 2530, 2541,
	574, 581, 582, 584,	2557, 2560, 2577, 2580
	587, 588, 590, 597,	\bp@commands
A	598, 600, 603, 604,	1966, 1998, 2005,
\abs@value	606, 627, 628, 630,	2028, 2060, 2067,
2647, 2649, 2650,	1063, 1068, 1240,	2168, 2181, 2208,
2680, 2683, 2692,	1539, 1544, 1548, 1563	2237, 2251, 2290,
2710, 2712, 2713,	\bp@add@tmp 1540,	2319, 2335, 2446, 2584
2738, 2747, 2827,	1552, 1555, 1562, 1564	\bp@complex 1341,
2829, 2830, 2837,	\bp@approx 872,	1369, 1680, 1689
2840, 2850, 2868,	900, 976, 1247,	\bp@contains@equal . .
2870, 2871, 2873, 2883	1254, 1298, 1855,	 1059, 1095,
\abs@valuefloat 2645,	1890, 1944, 1949, 1952	<u>1530</u> , 1733, 2087,
2646, 2658, 2662,		

2109, 2387, 2413, 2441	212, 350, 352, 353,	2415, 2416, 2421, 2430
\bp@contains@num ...	357, 490, 492, 493,	\bp@mag@axes
..... 1241, <u>1533</u>	497, 611, 613, 614,	... 405, 409, 410,
\bp@d 1693, 1694, 1702, 1715	618, 664, 666, 667,	412, 422, 423, 425,
\bp@den .. 1701, 1710, 1711	671, 717, 719, 720, 724	506, 507, 509, 869,
\bp@den@deg .. 1807, 1812	\bp@gnu@cmd . 929, 934,	913, 973, 1016,
\bp@den@im 1776,	1032, 1037, 1088,	1327, 1850, 1876, 1881
1817, 1832, 1835	1127, 1130, 1153,	\bp@mag@cmd
\bp@den@list	1279, 1290, 1658, 1838	1015, 1024, 1032,
... 1709, 1712, 1715	\bp@gnu@plot 928, 933,	1266, 1278, 1308, 1332
\bp@den@re 1775,	1031, 1085, 1129,	\bp@mag@commands ...
1814, 1820, 1833, 1835	1152, 1277, 1283, <u>1657</u>	 873, 922,
\bp@domain@end	\bp@gnu@prefix <u>1</u>	930, 977, 1025,
..... 153, 171,	\bp@gnu@unwrap@plot .	1033, 1299, 1857, 1896
243, 316, 394, 456,	 1036,	\bp@mag@plot ... 254,
646, 661, 699, 714,	1126, 1286, <u>1837</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,	1853, 1864, 1869, 1906
1223, 1224, 1306,	1297, 1321, 1854, 1887	\bp@max@im@float 2815,
1316, 1317, 1974,	\bp@has@negative@values	2832, 2836, 2906, 2907
1982, 2036, 2044,	 2466,	\bp@max@im@pow@ ... 2915
2148, 2155, 2216,	2637, 2665, 2728, 2782	\bp@max@im@value ...
2225, 2245, 2298,	\bp@has@positive@values	 2817, 2837
2307, 2327, 2369, 2376	 2459,	\bp@max@im@valuefloat
\bp@domain@start 152,	2636, 2656, 2719, 2771	 2908, 2909
170, 242, 315, 393,	\bp@imagpart . 1343, 1368	\bp@max@re@float ...
455, 645, 661, 698,	\bp@k 1692, 1694	... 2629, 2675, 2679
714, 891, 903, 905,	\bp@Last@Loop@p 2572, 2574	\bp@max@re@neg@float
995, 1006, 1008,	\bp@Last@Loop@z 2552, 2554	2633, 2667, 2671,
1174, 1184, 1213,	\bp@legacy	2730, 2734, 2785, 2794
1222, 1305, 1315,	... 890, 900, 994,	\bp@max@re@neg@pow@ .
1317, 1973, 1982,	1003, 1066, 1073,	... 2469, 2783, 2791
2035, 2044, 2147,	1075, 1098, 1103,	\bp@max@re@pos@float
2215, 2225, 2245,	1105, 1243, 1247,	 2631,
2297, 2307, 2327, 2368	1694, 1715, 1972,	2658, 2662, 2721,
\bp@domain@start: ...	1979, 2034, 2041,	2725, 2774, 2793, 2794
..... 1186,	2088, 2090, 2110,	\bp@max@re@pos@pow@ .
1224, 2155, 2376	2112, 2214, 2221,	... 2462, 2772, 2780
\bp@fallback	2296, 2303, 2388,	\bp@max@re@pow@ ... 2805
962, 964, 967, 969,	2390, 2414, 2416,	\bp@max@re@value ...
2278, 2280, 2283, 2285	2453, 2456, 2457, 2542	 2635, 2680
\bp@fix@add@opt	\bp@list 1582,	\bp@max@re@valuefloat
... 1063, 1065, <u>1538</u>	1583, 1778, 1779	 2795,
\bp@freq@filter <u>83</u>	\bp@loop@delay	2796, 2797, 2798, 2799
\bp@freq@label <u>83</u>	... 1777, 1828, 1833	\bp@min@Im@pow@ ... 2905
\bp@freq@scale	\bp@mag . 898, 900, 921,	\bp@min@im@pow@ 2494,
83, 191, 198, 336,	928, 1001, 1003,	2525, 2526, 2904, 2905
343, 476, 483, 905,	1024, 1031, 1235,	\bp@min@im@thresh@float
1008, 1079, 1085,	1243, 1247, 1254,	2812, 2841, 2845,
1110, 1116, 1126,	1258, 1269, 1277,	2851, 2874, 2878, 2884
1129, 1145, 1152,	1976, 1979, 1996,	\bp@min@im@threshold@found
1267, 1271, 1317,	1997, 2002, 2003,	... 2811, 2839,
1666, 1846, 1982,	2038, 2041, 2058,	2842, 2872, 2875, 2894
2044, 2093, 2100,	2059, 2064, 2065,	\bp@min@im@threshold@result
2115, 2122, 2225,	2089, 2090, 2095,	... 2524, 2809,
2245, 2307, 2327,	2096, 2102, 2103,	2840, 2850, 2873,
2393, 2403, 2419, 2429	2111, 2112, 2117,	2883, 2895, 2898, 2907
\bp@freq@scale:	2118, 2124, 2125,	\bp@min@im@ZPK <u>2457, 2808</u>
..... 905, 1008,	2218, 2221, 2236,	\bp@min@pow@ 2770
1317, 1982, 2044,	2246, 2300, 2303,	\bp@min@re@pow@
2225, 2245, 2307, 2327	2318, 2328, 2389,	... 2462, 2469,
\bp@full 205, 207, 208,	2390, 2395, 2404,	

2480, 2522, 2523,	2218, 2221, 2236,	\bp@scale 2171,
2769, 2770, 2772, 2783	2246, 2300, 2303,	2187, 2449, 2455, 2508
\bp@min@re@threshold@result	2318, 2328, 2389,	\bp@short 207,
2521, 2623, 2683,	2390, 2395, 2404,	209, 212, 215, 217,
2692, 2738, 2747,	2415, 2416, 2421, 2430	352, 354, 357, 360,
2759, 2761, 2763, 2797	\bp@ph@axes	362, 492, 494, 497,
\bp@min@real@ZPK . . .	. . . 406, 414, 415,	500, 502, 613, 615,
. 2456, 2622	417, 430, 431, 433,	618, 621, 623, 666,
\bp@min@thresh@float	514, 515, 517, 870,	668, 671, 674, 676,
2626, 2684, 2688,	917, 974, 1020,	719, 721, 724, 727, 729
2693, 2739, 2743, 2748	1330, 1851, 1879, 1882	\bp@short@list
\bp@min@threshold@found	\bp@ph@cmd	. . . 54, 944, 949,
. 2625, 2682,	1019, 1026, 1037,	1048, 1053, 1159,
2685, 2737, 2740, 2758	1270, 1289, 1309, 1334	1162, 1198, 1201,
\bp@min@threshold@result	\bp@ph@commands	2014, 2019, 2076,
. 2761	 874, 924,	2081, 2132, 2135,
\bp@mode . . 1244, 1248,	935, 978, 1027,	2260, 2265, 2345,
1253, 1257, 1282, 1285	1038, 1300, 1856, 1894	2350, 2353, 2356,
\bp@new . 215, 217, 220,	\bp@ph@plot 255,	2592, 2597, 2613, 2618
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,	1852, 1867, 1870, 1908	1095, 1096, 1098,
515, 518, 621, 623,	\bp@ph@scale	2086, 2087, 2088,
626, 628, 631, 674,	. . . 83, 178, 183,	2108, 2109, 2110,
676, 679, 681, 684,	323, 328, 463, 468,	2386, 2387, 2388,
727, 729, 732, 734, 737	772, 776, 787, 800,	2412, 2413, 2414,
\bp@num . . 1700, 1704, 1705	803, 815, 821, 822,	2440, 2441, 2450, 2453
\bp@num@deg . . 1784, 1789	842, 1026, 1833,	\bp@style 68
\bp@num@im 1774,	1844, 1846, 1996,	\bp@tf@new@to@legacy
1794, 1832, 1834	1997, 2002, 2003,	 981,
\bp@num@list	2058, 2059, 2064,	1096, 1242, 1696,
. . . 1703, 1706, 1715	2065, 2095, 2096,	2031, 2110, 2293, 2414
\bp@num@re 1773,	2102, 2103, 2117,	\bp@TF@plot
1791, 1797, 1832, 1834	2118, 2124, 2125,	1003, 1103, 1105,
\bp@p 1683, 1684, 1686, 1687	2246, 2249, 2318,	1243, 1258, 1772,
\bp@p@list 1685, 1689, 1694	2328, 2333, 2404,	2041, 2112, 2303, 2416
\bp@parse@add@Bode@opt	2407, 2421, 2430, 2435	\bp@tf@to@zpk . . 955,
. . . 1240, 1251, 1942	\bp@ph@x@label 83	1135, 1717, 2271, 2602
\bp@parse@complex . . .	\bp@ph@y@label 83	\bp@tf@fnl@spec 1698, 1699
. . . 1338, 1679, 1688	\bp@pim . . 2570, 2575, 2580	\bp@tikz 875, 896,
\bp@parse@env@opt . . .	\bp@plot 647, 652,	979, 999, 1170,
. 1173,	680, 681, 683, 700,	1178, 1209, 1217,
1212, 1919, 2146, 2367	705, 733, 734, 736,	1301, 1311, 1858,
\bp@parse@N@opt	1070, 1073, 1075,	1900, 1921, 1929,
. 1971, 2033,	1081, 1085, 1100,	1967, 1978, 2029,
2165, 2213, 2295, 2452	1103, 1105, 1113,	2040, 2143, 2151,
\bp@parse@opt	1118, 1126, 1129,	2169, 2184, 2209,
889, 993, 1304, 1849	1269, 1273, 1279,	2220, 2291, 2302,
\bp@ph	1290, 1943, 1954,	2364, 2372, 2447, 2506
899, 900, 923, 933,	1993, 2055, 2167,	\bp@tmp . 895, 897, 901,
1002, 1003, 1026,	2178, 2193, 2234,	911, 1071, 1073,
1036, 1236, 1243,	2316, 2445, 2559, 2579	1075, 1101, 1103, 1105
1247, 1254, 1258,	\bp@pre 2567, 2580	\bp@tmp@mag 912, 921, 929
1273, 1283, 1286,	\bp@prefix 27, 45, 1658,	\bp@tmp@ph . 916, 923, 934
1976, 1979, 1996,	1665, 1838, 1845,	\bp@tmp@status . 955,
1997, 2002, 2003,	2248, 2332, 2406, 2434	956, 1135, 1136,
2038, 2041, 2058,	\bp@PZZPK@ticksXNeg .	2271, 2272, 2602, 2603
2059, 2064, 2065,	 2467, 2471	\bp@tmp@zpk
2089, 2090, 2095,	\bp@PZZPK@ticksXPos .	. . . 955, 958, 960,
2096, 2102, 2103,	 2460, 2464	1135, 1137, 2271,
2111, 2112, 2117,	\bp@realpart . 1342, 1367	2274, 2276, 2602, 2604
2118, 2124, 2125,		

<code>\bp@trim</code> 1536,	<code>G</code>	<code>\l_bp_pos_int</code>
1861, 1924, 1946, 2173	<code>\g@addto@macro</code>	 1350, 1457,
<code>\bp@tztpk@d</code> 1739,	62, 566, 568, 572,	1466, 1473, 1480,
1742, 1752, 1766	574, 582, 584, 588,	1488, 1491, 1494, 1502
<code>\bp@tztpk@den</code> .. 1738,	590, 598, 600, 604, 606	<code>\l_bp_sep_tl</code> .. 1349,
1742, 1749, 1762	<code>\gnu@id</code>	1467, 1515, 1520
<code>\bp@tztpk@feat</code> . 1744,	927, 928, 932, 933,	<code>\l_bp_sep_pos_int</code> ...
1745, 1748, 1751	1030, 1031, 1035,	 1351,
<code>\bp@tztpk@file</code>	1036, 1084, 1085,	1466, 1488, 1491
.... 1719, 1720,	1122, 1126, 1129,	<code>\l_imag_tl</code> 1340, 1368,
1721, 1722, 1723,	1150, 1152, 1276,	1369, 1377, 1398,
1724, 1725, 1726, 1727	1277, 1281, 1283, 1286	1516, 1517, 1521, 1522
<code>\bp@tztpk@list</code> 1743, 1744	<code>I</code>	<code>\l_real_tl</code>
<code>\bp@tztpk@num</code> .. 1737,	<code>\if@babel</code> 54, 941,	1339, 1367, 1369,
1742, 1746, 1762	1045, 1157, 1196,	1376, 1397, 1510, 1511
<code>\bp@tztpk@outfile</code> .. 1717	2011, 2073, 2130,	<code>\l_tmpa_tl</code> 1407, 1408, 1419
<code>\bp@tztpk@val</code> .. 1744,	2257, 2342, 2589, 2610	<code>\l_tmpb_tl</code>
1746, 1749, 1752	<code>\if@declutterarg</code> ...	1411, 1414, 1415,
<code>\bp@user@prefix</code> . 28,	 5, 29, 35, 49	1421, 1422, 1425, 1426
36, 38, 876, 980,	<code>\if@hzarg</code> 13, 103,	<code>\l_tmpc_tl</code> ... 1442, 1443
1171, 1210, 1302,	142, 244, 395, 884, 988	<code>\Last@LoopValue</code> 1575, 1577
1859, 1903, 1905,	<code>\if@pgfarg</code> 1,	<code>\log@adjusted</code> .. 2766,
1922, 1932, 1934,	21, 106, 193, 200,	2767, 2901, 2902
1968, 2030, 2144,	338, 345, 478, 485,	<code>\log@bp@max@im</code> 2910, 2911
2170, 2190, 2192,	920, 954, 1023,	<code>\log@bp@max@im@adjusted</code>
2210, 2292, 2365, 2448	1077, 1107, 1143,	 2912, 2913
<code>\bp@z</code> 1674, 1675, 1677, 1678	1265, 1994, 2056,	<code>\log@bp@max@re</code> 2800, 2801
<code>\bp@z@list</code> 1676, 1680, 1694	2091, 2113, 2235,	<code>\log@bp@max@re@adjusted</code>
<code>\bp@zim</code> .. 2550, 2555, 2560	2270, 2317, 2391, 2417	 2802, 2803
<code>\bp@zpk@new@to@legacy</code>	<code>\if@radarg</code> 9, 89,	<code>\log@bp@max@re@neg</code> ..
..... 877, 1062,	147, 249, 400, 879, 983	 2786, 2787
1246, 1669, 1969,	<code>\IfFileExists</code> 1764	<code>\log@bp@max@re@neg@adjusted</code>
2088, 2211, 2388, 2450	<code>\input</code> 1765	 2788, 2789
<code>\bp@ZPK@plot</code>	<code>\int_compare:nNf</code> .. 1491	<code>\log@bp@max@re@pos</code> ..
. 900, 1073, 1075,	<code>\int_compare:nNt</code> ...	 2775, 2776
1247, 1254, 1581,	... 1443, 1457, 1459	<code>\log@bp@max@re@pos@adjusted</code>
1979, 2090, 2221, 2390	<code>\int_compare:nNtTF</code> . 1488	 2777, 2778
<code>\bp@zpknl@spec</code> 1672, 1673	<code>\int_decr:N</code> 1455	<code>\log@result</code> 2764,
<code>\bp@zre</code> 2547, 2560	<code>\int_incr:N</code>	2765, 2899, 2900
<code>\bp_if_contains:nNTF</code>	... 1454, 1473, 1494	<code>\lowercase</code> 62
... 1525, 1531, 1534	<code>\int_new:N</code> 1350, 1351, 1352	<code>M</code>
<code>C</code>	<code>\int_set_eq:NN</code> 1466	<code>\MagCSPoles</code> 810
<code>\closeout</code> 1727	<code>\int_zero:N</code>	<code>\MagCSPolesAsymp</code> ... 810
<code>\cs_new_eq:NN</code> 1536	... 1480, 1481, 1502	<code>\MagCSPolesLin</code> 810
<code>\cs_new_protected:Npn</code>	<code>L</code>	<code>\MagCSPolesPeak</code> 829
1373, 1381, 1395,	<code>\l__bode_check_tl</code> ...	<code>\MagCSZeros</code> 810
1402, 1438, 1449,	... 1525, 1527, 1528	<code>\MagCSZerosAsymp</code> ... 810
1477, 1486, 1498, 1526	<code>\l__bp_after_tl</code>	<code>\MagCSZerosLin</code> 810
<code>\currentdegree</code> . 1789,	1348, 1492, 1501,	<code>\MagCSZerosPeak</code> 829
1790, 1793, 1794,	1507, 1513, 1521, 1522	<code>\MagDel</code> 775, 1650
1795, 1797, 1798,	<code>\l__bp_before_tl</code> ...	<code>\MagK</code> 769, 1631
1812, 1813, 1816,	 1347, 1489,	<code>\MagKAsymp</code> 769, 1628
1817, 1818, 1820, 1821	1500, 1506, 1509, 1511	<code>\MagKLin</code> 769, 1624
<code>D</code>	<code>\l__bp_found_bool</code> ...	<code>\MagPole</code> ... 777, 804, 1615
<code>\dospecials</code> 65	 1353,	<code>\MagPoleAsymp</code>
<code>E</code>	1389, 1465, 1479	 777, 806, 1612
environments:		

\MagS0ZerosAsymp ... 837	\PhCSpolesLin .. 810 , 844	\stripbodeprefix 206 , 207 , 351 , 352 , 491 , 492 , 612 , 613 , 665 , 666 , 718 , 719
\MagS0ZerosLin 837	\PhCSZeros 810	
\MagS0ZerosPeak 855	\PhCSZerosAsymp 810	
\MagZero 804 , 1597	\PhCSZerosLin 810	
\MagZeroAsymp . 804 , 1594	\PhDel 776 , 1649	
\MagZeroLin ... 804 , 1590	\PhK 769 , 1630	T
N	\PhKAsymp .. 769 , 775 , 1627	\temp@mag@cmd 1325 , 1331
\n@mod 1 , 1026 , 1113 , 2318 , 2421	\PhKLin 769 , 775 , 1623	\temp@ph@cmd . 1328 , 1333
\n@mod@n 1	\PhPole 777 , 807 , 1614	\tl_clear:N .. 1500 , 1501
\n@mod@p 1 , 785	\PhPoleAsymp 777 , 809 , 1611	\tl_count:N 1442
\n@pow 1 , 778 , 779 , 780 , 790 , 791 , 793 , 794 , 796 , 797 , 798 , 799 , 800 , 801 , 810 , 811 , 814 , 815 , 816 , 818 , 820 , 821 , 838 , 842 , 1794 , 1795 , 1797 , 1798 , 1817 , 1818 , 1820 , 1821 , 1834 , 1835 , 1996 , 1997 , 2002 , 2003 , 2058 , 2059 , 2064 , 2065 , 2095 , 2096 , 2102 , 2103 , 2117 , 2118 , 2124 , 2125	\PhS0Poles 837	\tl_gset:Nx 1369
	\PhS0PolesAsymp 837	\tl_gset_eq:NN 1367 , 1368
	\PhS0PolesLin 837	\tl_head:N 1407
	\PhS0Zeros 837	\tl_if_empty:NTF 1404 , 1411 , 1422 , 1509 , 1513
	\PhS0ZerosAsymp 837	\tl_if_in:VnTF 1528
	\PhS0ZerosLin 837	\tl_if_single_token:n 1356
	\PhZero 804 , 1596	\tl_if_single_token:nTF 1451
	\PhZeroAsymp .. 804 , 1593	\tl_if_single_token:VF 1440
	\PhZeroLin 804 , 1589	\tl_map_function:NN 1482 , 1504
	\PoleZeroMapTF 2601	\tl_new:N 1339 , 1340 , 1341 , 1342 , 1343 , 1346 , 1347 , 1348 , 1349 , 1525
	\PoleZeroMapTF@Shorthandoff 2614 , 2616	\tl_put_right:Nn 1489 , 1492
	\PoleZeroMapZPK 2439 , 2604	\tl_remove_all:Nn .. 1360
	\PoleZeroMapZPK@exp 2479 , 2482 , 2484 , 2493 , 2496 , 2498	\tl_set:Nn 1377 , 1397 , 1405 , 1412 , 1423 , 1467 , 1510 , 1516 , 1517
	\PoleZeroMapZPK@formatXTick 2474 , 2502	\tl_set:Nx 1359 , 1407 , 1410 , 1415 , 1421 , 1442 , 1444 , 1521 , 1527
	\PoleZeroMapZPK@formatYTick 2488 , 2503	\tl_set_eq:NN .. 1376 , 1426 , 1431 , 1511 , 1522
	\PoleZeroMapZPK@Shorthandoff 2593 , 2595	\tl_tail:N ... 1410 , 1421
	\PoleZeroMapZPK@tick 2475 , 2476 , 2480 , 2481 , 2489 , 2490 , 2494 , 2495	\variant:nspaces:N .. 1536
	\PoleZeroMapZPK@xticklabel 2502 , 2519	\token_if_eq_charcode:NNT 1454 , 1455
	\PoleZeroMapZPK@yticklabel 2503 , 2520	\token_if_eq_charcode_p:NN 1462 , 1463
\prg_generate_conditional_variant:nn 1356		\TrimSpaces . 865 , 953 , 1057 , 1093 , 1134 , 1166 , 1205 , 1234 , 1293 , 1961 , 2023 , 2085 , 2107 , 2139 , 2203 , 2269 , 2360 , 2385 , 2411 , 2439 , 2601
O	R	U
\openout 1720	\regex_match:nVTF .. 1363	\use:n 1444
\Orig@BodeTF . 1046 , 1052	\regex_replace_once:nnN 1384	
\Orig@BodeZPK .. 942 , 948	\removeslash 211 , 212 , 356 , 357 , 496 , 497 , 617 , 618 , 670 , 671 , 723 , 724	
\Orig@NicholsTF 2343 , 2349	\renewcommand .. 178 , 183 , 191 , 198 , 323 , 328 , 336 , 343 , 463 , 468 , 476 , 483 , 943 , 1047 , 2013 , 2075 , 2259 , 2344 , 2591 , 2612	
\Orig@NicholsZPK 2258 , 2264		
\Orig@NyquistTF 2074 , 2080		
\Orig@NyquistZPK 2012 , 2018		
\Orig@PoleZeroMapTF 2611 , 2617		
\Orig@PoleZeroMapZPK 2590 , 2596		
P	S	Z
\PhCSpoles 810	\str_if_eq:VnTF 1408 , 1419 , 1515 , 1520	\zpkGain 1766
\PhCSpolesAsymp 810 , 847		\zpkPoles 1766
		\zpkZeros 1766

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		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
\bp@fix@add@opt:	Added a conditional to avoid always plotting the true plot, fixes issue #18		52
v3.0.3		\NicholsTF: In pgf mode, legacy 4-argument calls now forward frequency limits via domain options when dispatching to \NicholsZPK	65
\bp@parse@complex:	Rewritten to support arbitrary expressions with nested braces and parentheses		49
v3.0.4		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
\BodeTF:	Fixed babel wrapper command argument specification for grabbing arguments		43
\BodeZPK:	Fixed babel wrapper command argument specification for grabbing arguments		41
\NicholsTF:	Fixed babel wrapper command argument specification for grabbing arguments		67
\NyquistTF:	Fixed babel wrapper command argument specification for grabbing arguments		62
\NyquistZPK:	Fixed babel wrapper command argument specification for grabbing arguments		61
v3.1		v3.1.1	
\addBodeTFPlot:	Pre-expand $\{ \langle tf-spec \rangle \}$ and expand it before \bp@contains@equal so macro-valued specs route correctly		44
\addBodeTFPlots:	Added support for linear and asymptotic approximations when using the TF format		45
\addBodeZPKPlots:	Pre-expand $\{ \langle zpk-spec \rangle \}$ and expand it before \bp@contains@equal so macro-valued specs route correctly		43
\addNicholsTFChart:	Use \expandafter when calling \bp@contains@equal so the pre-expanded spec is tested as character tokens		68
\addNicholsZPKChart:	Use \expandafter when calling \bp@contains@equal so the pre-expanded spec is tested as character tokens		67
\addNyquistTFPlot:	Use \expandafter when calling \bp@contains@equal so the pre-expanded spec is tested as character tokens		63
\addNyquistZPKPlot:	Use \expandafter when calling \bp@contains@equal so the pre-expanded spec is tested as character tokens		62
\BodeTF:	In pgf mode, legacy 4-argument calls now forward frequency limits via domain options when dispatching to \BodeZPK		42
\NicholsTF:	In pgf mode, legacy 4-argument calls now forward frequency limits via domain options when dispatching to \NicholsZPK		65
\PoleZeroMapTF:	Added support for pole-zero maps with TF specifications using Python		71
\PoleZeroMapZPK:	Use \expandafter when calling \bp@contains@equal so the pre-expanded spec is tested as character tokens		68
General:	Fixed issue with leading and trailing spaces in arguments		1
\addBodePlot:	Fixed issue with leading and trailing spaces in arguments		47
\addBodeTFPlot:	Fixed issue with leading and trailing spaces in arguments		44
\addBodeTFPlots:	Fixed issue with leading and trailing spaces in arguments		45
\addBodeZPKPlots:	Fixed issue with leading and trailing spaces in arguments		43
\addNicholsTFChart:	Fixed issue with leading and trailing spaces in arguments		68
\addNicholsZPKChart:	Fixed issue with leading and trailing spaces in arguments		67
\addNyquistTFPlot:	Fixed issue with leading and trailing spaces in arguments		63
\addNyquistZPKPlot:	Fixed issue with leading and trailing spaces in arguments		62
\BodeTF:	Fixed issue with leading and trailing spaces in arguments		42
\BodeZPK:	Fixed issue with leading and trailing spaces in arguments		40
\bp@parse@add@Bode@opt:	Trim spaces inside braced option values		60
\bp@parse@env@opt:	Trim spaces inside braced option values		59
\bp@parse@N@opt:	Trim spaces inside braced option values		64
\bp@parse@opt:	Trim spaces inside braced option values		58

\bp@trim: Added a helper to trim spaces inside braced option values	52	\NyquistZPK: Fixed issue with leading and trailing spaces in arguments	60
\NicholsTF: Fixed issue with leading and trailing spaces in arguments	65	\PoleZeroMapTF: Fixed issue with leading and trailing spaces in arguments	71
\NicholsZPK: Fixed issue with leading and trailing spaces in arguments	64	\PoleZeroMapZPK: Fixed issue with leading and trailing spaces in arguments	68
\NyquistTF: Fixed issue with leading and trailing spaces in arguments	61		