The pst-sigsys Package

(version 1.4)

Farshid Delgosha
fdelgosha@gmail.com

March 7$^{\text{th}}$, 2011

Abstract

This package is a collection of useful macros for disciplines related to signal processing. It defines macros for plotting a sequence of numbers, drawing the pole-zero diagram of a system, shading the region of convergence, creating an adder or a multiplier node, placing a framed node at a given coordinate, creating an up-sampler or a down-sampler node, drawing the block diagram of a system, drawing adaptive systems, sequentially connecting a list of nodes, and connecting a list of nodes to one node using any node-connecting macro. The author welcomes all comments for further improvements of this package and suggestions for adding new macros or features.

Contents

1 Introduction

To use the `pst-sigsys` package, add the command

```
\usepackage{pst-sigsys}
```

to the preamble of the document. This package loads `pstricks` [?], `pst-node` [?], and `pst-xkey` [?] packages. Moreover, it activates polar coordinates through the `\SpecialCoor` macro defined by the `pstricks` package. Hence, all macros support polar coordinates. Simultaneously loading the `pst-sigsys` package along with some other packages in regular TEX might be impossible due to memory restrictions. If TEX runs out of memory, load the `etex` package.

Section 2 keeps a change log from previous versions of the package. All macros defined by the `pst-sigsys` package are introduced in Section 3. The extra functionalities of the package are introduced in Section 4. Many practical examples are provided in Section 5 that illustrate the applications of the introduced macros.

2 Change Log

- **Version 1.4 (03/07/2011):** The code for the `\psaxeslabels` macro is updated to accommodate for cases when one of the two axes lines has length zero. The code for `\pspole` is updated due to the new changes in the `pst-node` package. The new key `afac` is added to the `\psadaptive` macro.

- **Version 1.3 (06/18/2010):** In the `\pstick` and `\psTick` macros, the tick angle is either directly specified by the user or set by the `angle` key when unspecified. The `ticklength` key refers to the entire length of a tick, not half of it. The new key `killzero` is added to the `\psstem` macro. In the `\psldots` and `\ldotsnode` macros, the angle of dots is either directly specified by the user or set by the `angle` key when unspecified. Two new macros `\psadaptive` and `\psknob` are added. The new keys `framewidth`, `frameheight`, and `FillColor` are introduced.

- **Version 1.2 (01/15/2010):** Five new macros `\pstick`, `\psTick`, `\pssignal`, `\ldotsnode`, and `\ncstar` are added. The macros `\pshtick`, `\psvtick`, `\pshTick`, and `\psvTick` are not available any longer since their functionalities are carried out by the newly defined macros

`\pstick` and `\psTick`. Codes for the macros `\pscircleop`, `\psframeop`, `\psldots`, and `\nclist` are updated. Four new keys `gratioWh`, `gratioWv`, `gratioHh`, and `gratioHv` are added that allow frames with edges proportional by the golden ratio. The global round-cornering settings are removed because of their undesired effects in other packages. Hence, the option `notelegant` is not available any longer. Instead, the new style `RoundCorners` is introduced. The styles `BraceUp`, `BraceDown`, `BraceLeft`, and `BraceRight` are not available any longer. Instead, the macros `\psBraceUp`, `\psBraceDown`, `\psBraceLeft`, and `\psBraceRight` are defined. The option `pstadd` is not available any longer. If the package `pstricks-add` is loaded, the relevant styles are automatically defined. The macros `\RE`, `\IM`, `\sRE`, and `\sIM` are not available any longer because of their irrelevance to the objectives of the package.

- **Version 1.1 (04/01/2009):** Four new macros `\pshtick`, `\psvtick`, `\pshTick`, and `\psvTick` are added. The codes of macros `\psusampler` and `\psdsampler` are updated. However, there is no change in their user interface.
- **Version 1.0 (01/15/2009):** The fist version of the package.

3 Macros

In this section, we introduce all the macros defined by the `pst-sigsys` package. Every macro has some optional keys that can be assigned either directly inside brackets right after the macro name or through the `\psset` macro provided by the `pstricks` package. In the syntax of every macro, the optional portions are identified by the shaded background. Unless directly stated, all coordinates specified by (*coor*) could be either in the cartesian format (x, y) or the polar format $(\rho; \theta)$[1]. After the introduction of every macro, some examples are provided to illustrate the usage of that macro.

3.1 `\psaxeslabels`

`\psaxeslabels` [*keys*] {*arrows*} $(x_0, y_0)(x_1, y_1)(x_2, y_2)$ {*x-label*}{*y-label*}

This macro is a simplified version of the `\psaxes` macro defined by the `pst-plot` package [?]. As depicted in Figure 1, the `\psaxeslabels` draws two straight lines, one vertical and one horizontal, that intersect at the point (x_0, y_0). These lines are enclosed by a virtual rectangular box with the lower left corner at (x_1, y_1) and the upper right corner at (x_2, y_2). The two lines are labeled *x-label* and *y-label*, respectively. Similar to the `\psaxes` macro, the use of *arrows* is optional. The keys specific to the `\psaxeslabels` are summarized in Table 1.

Table 1. `\psaxeslabels` keys

Key	*Value*	*Default*	*Description*
`xlpos`	`t | b`	`b`	Position of the x-label along the horizontal axis
`ylpos`	`l | r`	`r`	Position of the y-label along the vertical axis

[1] Recall that `pst-sigsys` activates the polar coordinates on loading. Hence, there is no need to use the `\SpecialCoor` macro.

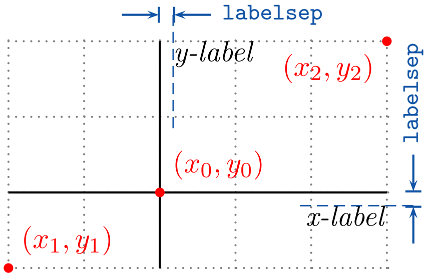
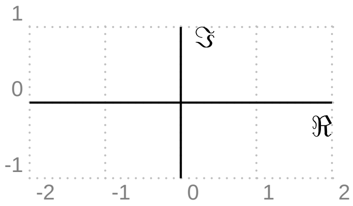
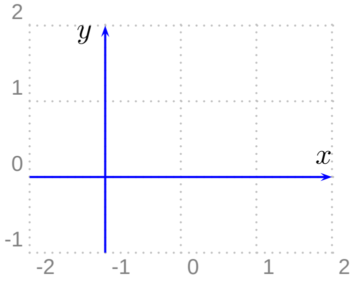


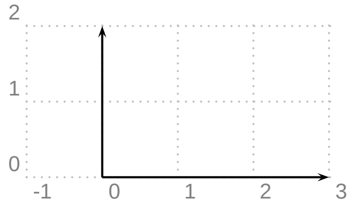
Figure 1. `\psaxeslabels` macro



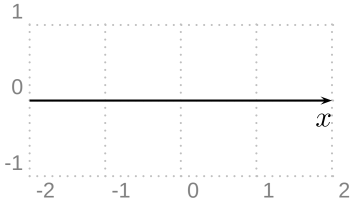
```
\begin{pspicture}[showgrid](-2,-1)(2,1)
  \psaxeslabels(0,0)(-2,-1)(2,1){$\Re$}{$\Im$}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](-2,-1)(2,2)
  \psset{linecolor=blue,xlpos=t,ylpos=l}
  \psaxeslabels{->}(-1,0)(-2,-1)(2,2){$x$}{$y$}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](-1,0)(3,2)
   \psaxeslabels{->}(0,0)(0,0)(3,2){}{}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](-2,-1)(2,1)
   \psaxeslabels{->}(0,0)(-2,0)(2,0){$x$}{}
\end{pspicture}
```

3.2 \pstick

`\pstick` [*keys*] {*angle*} (*coor*){*ticklength*}

As depicted in Figure 2, the `\pstick` macro draws a straight line with length *ticklength* centered at (*coor*) and angled *angle* with respect to the horizontal axis. If the optional parameter *angle* is absent, then the angle is determined using the `angle` key. This macro could be used for adding tick lines to coordinate axes in addition to many other usages. The keys specific to the `\pstick` are summarized in Table 2.

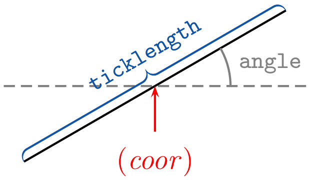
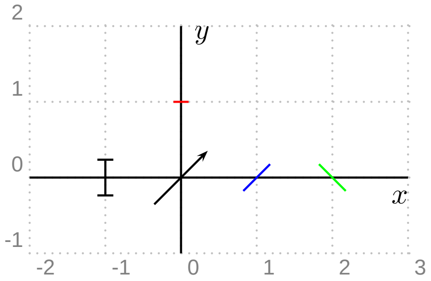


Figure 2. `\pstick` macro

Table 2. `\pstick` keys

Key	*Value*	*Default*	*Description*
`angle`	*num*	0	Tick angle



```
\begin{pspicture}[showgrid](-2,-1)(3,2)
   \psaxeslabels(0,0)(-2,-1)(3,2){$x$}{$y$}
   \pstick[linecolor=red](0,1){.2}
   \pstick[arrows=|-|]{90}(-1,0){.5}
   \psset{angle=45}
   \pstick[arrows=->](0,0){1}
   \pstick[linecolor=blue](1,0){.5}
   \pstick[linecolor=green]{135}(2,0){.5}
\end{pspicture}
```

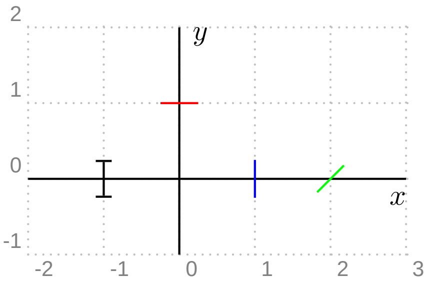
3.3 \psTick

`\psTick` [*keys*] {*angle*} (*coor*)

Similar to `\pstick`, the `\psTick` macro draws a straight line centered at (*coor*) and angled *angle* with respect to the horizontal axis. The only difference is that the tick length is specified by the `ticklength` key (Table 3). This macro is useful when multiple ticks are to be drawn all with the same length.

Table 3. `\psTick` keys

Key	*Value*	*Default*	*Description*
`ticklength`	*num[dimen]*	0.15	Tick length
`angle`	*num*	0	Tick angle



```
\begin{pspicture}[showgrid](-2,-1)(3,2)
   \psaxeslabels(0,0)(-2,-1)(3,2){$x$}{$y$}
   \psset{ticklength=.5}
   \psTick[linecolor=red](0,1)
   \psTick[arrows=|-|]{90}(-1,0)
   \psTick[linecolor=blue]{90}(1,0)
   \psset{angle=45}
   \psTick[linecolor=green](2,0)
\end{pspicture}
```

3.4 `\pssignal`

`\pssignal` [*keys*] (*coor*){*node*}{*stuff*}

This macro places *stuff* inside an invisible frame centered at (*coor*) and makes that a node labeled *node* (Figure 3). The separation of the frame and the *stuff* is determined by the key `signalsep` (Table 4).

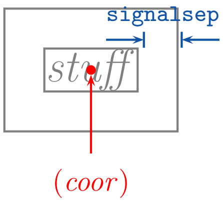
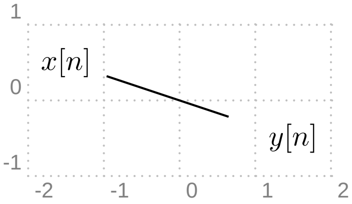


Figure 3. `\pssignal` macro

Table 4. `\pssignal` keys

Key	*Value*	*Default*	*Description*
`signalsep`	*num[dimen]*	5pt	Frame separation



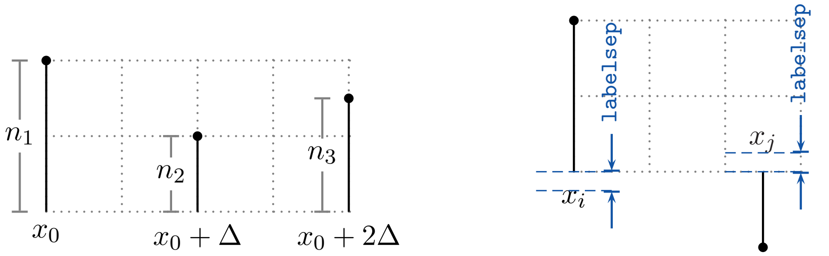
```
\begin{pspicture}[showgrid](-2,-1)(2,1)
   \pssignal(-1.5,.5){x}{$x[n]$}
   \pssignal[signalsep=.5](1.5,-.5){y}{$y[n]$}
   \ncline{x}{y}
\end{pspicture}
```

3.5 `\psstem`

`\psstem` [*keys*] (x_0, Δ)\{*list*\}

`\psstem` [*keys*] \{*list*\}

The `\psstem` macro plots the sequence defined by *list*, which is a comma-separated list of numbers. As shown in Figure 4a, if *list* $= n_1, n_2, n_3, \ldots$, then `\psstem` draws vertical lines (stems) at $x_0, x_0 + \Delta, x_0 + 2\Delta, \ldots$ on the horizontal axis with heights $n_1, n_2, n_3, \ldots$, respectively. *It is important to remember that both x_0 and Δ must be integers.*[2] In case their values are not explicitly given, they are assumed $x_0 = 0$ and $\Delta = 1$. The stem ends are determined by the `stemhead` key. The `\psstem` macro is also capable of numerically tagging the stems. As depicted in Figure 4b, the tag of every stem is placed either below or above it depending on whether the corresponding number in the sequence is nonnegative (positive or zero) or negative, respectively. The distance of tags to stems is determined by the `labelsep` key. In some cases (e.g., when the stemhead is `>`), it is desirable to remove zero-height stems. The key `killzero`, when used, removes such stems. The keys specific to the `\psstem` macro are summarized in Table 5.



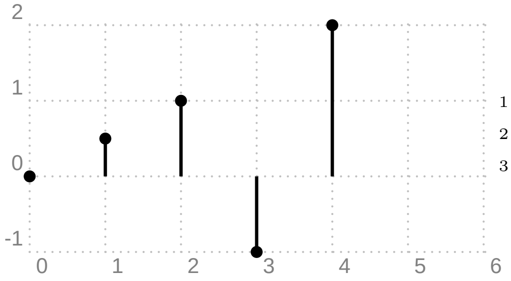
(a) Sample sequence

(b) Tagging

Figure 4. `\psstem` macro

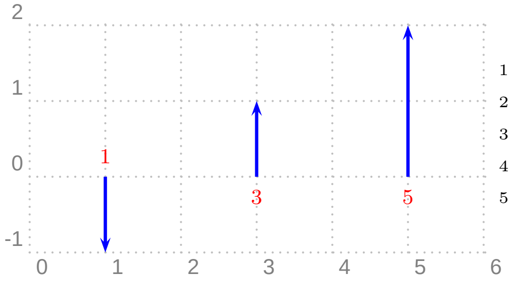
Table 5. `\psstem` keys

Key	*Value*	*Default*	*Description*
`stemhead`	*style*	`*`	Stem head. Possible choices are `*`, `o`, `>`, `<`, `>>`, `<<`, `|`, `)`, `(`, `>|`, and `<|`.
`stemtag`	*Boolean*	`false`	Tagging the stems
`stemtagformat`	*format*	`\scriptstyle`	Tag format
`killzero`	*Boolean*	`false`	Removing zero-height stems

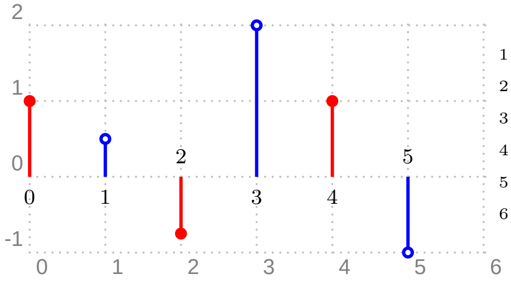


```
\begin{pspicture}[showgrid](0,-1)(6,2)
  \psstem[style=Stem]{0,.5,1,-1,2}
\end{pspicture}
```

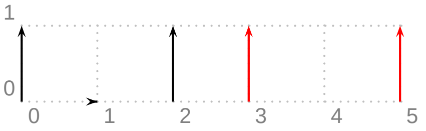
[2] If you must use non-integer values, utilize the `xunit` key to arbitrarily choose any real value.



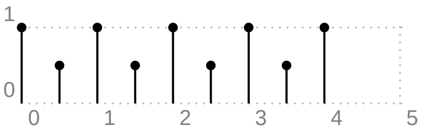
```
\begin{pspicture}[showgrid](0,-1)(6,2)
  \psset{style=Stem,linecolor=blue,%
  stemtagformat=\color{red}\scriptstyle}
  \psstem[stemhead=>,stemtag](1,2){-1,1,2}
\end{pspicture}
```



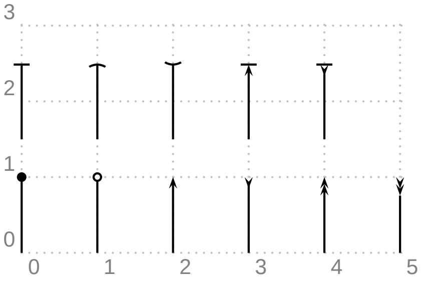
```
\begin{pspicture}[showgrid](0,-1)(6,2)
   \psset{style=Stem,stemtag}
   \psstem[linecolor=red](0,2){1,-.75,1}
   \psset{stemhead=o}
   \psstem[linecolor=blue](1,2){.5,2,-1}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](5,1)
  \psset{stemhead=>}
  \psstem{1,0,1}
  \psset{linecolor=red,killzero}
  \psstem(3,1){1,0,1}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](5,1)
   \psstem[xunit=.5]{1,.5,1,.5,1,.5,1,.5,1}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](5,3)
  \psstem[stemhead=*](0,1){1}
  \psstem[stemhead=o](1,1){1}
  \psstem[stemhead=>](2,1){1}
  \psstem[stemhead=<](3,1){1}
  \psstem[stemhead=>>](4,1){1}
  \psstem[stemhead=<<](5,1){1}

  \rput(0,1.5){%
    \psstem[stemhead=|](0,1){1}
    \psstem[stemhead=)](1,1){1}
    \psstem[stemhead=(](2,1){1}
    \psstem[stemhead=>|](3,1){1}
    \psstem[stemhead=<|](4,1){1}
  }
\end{pspicture}
```

3.6 `\pszero`

`\pszero` [*keys*] (*coor*){*node*}

This macro is used to generate a circle node centered at (*coor*) and labeled *node* that represents a zero of a system. It could also be used to generate several circles, all centered at (*coor*), representing high order zeros as shown in Figure 5. The radius of innermost circle is **`zeroradius`**, and it is incremented by **`zeroradiusinc`** for high order zeros. The line-width of all circles is determined by the **`zerowidth`** key. The key **`order`** determines the order of the zero, i.e., the number of circles. The key **`scale`** can be used to scale up or down the radius of the innermost circle, the radius increment, and the line-width of all circles. Table 6 summarizes keys corresponding to `\pszero` and their default values.

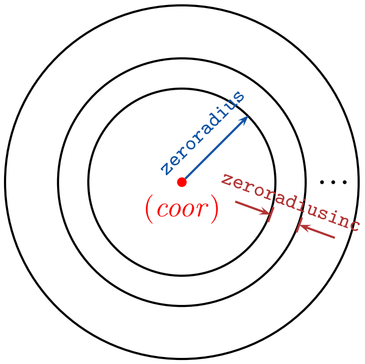
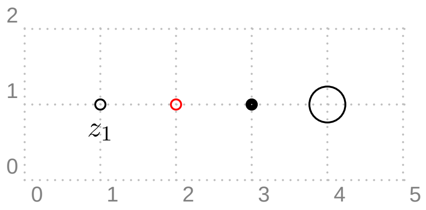


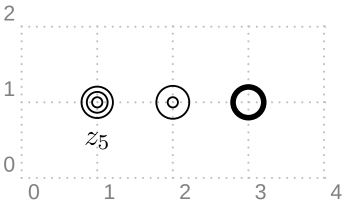
Figure 5. `\pszero` macro

Table 6. `\pszero` keys

Key	*Value*	*Default*	*Description*
`zerowidth`	*num[dimen]*	0.7pt	Line-width of all circles
`zeroradius`	*num[dimen]*	0.08	Radius of the innermost circle
`zeroradiusinc`	*num[dimen]*	0.07	Radius increment
`order`	*int*	1	Order of the zero
`scale`	*num*	1	Scale factor



```
\begin{pspicture}[showgrid](5,2)
  \pszero(1,1){z1}
    \nput{-90}{z1}{$z_1$}
  \pszero[linecolor=red](2,1){z2}
  \pszero[zerowidth=2pt](3,1){z3}
  \pszero[zeroradius=.25](4,1){z4}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](4,2)
  \pszero[order=3](1,1){z5}
    \nput{-90}{z5}{$z_5$}
  \pszero[zeroradiusinc=.15,%
          order=2](2,1){z6}
  \pszero[scale=3](3,1){z7}
\end{pspicture}
```

3.7 `\pspole`

`\pspole` [*keys*] (*coor*){*node*}

This macro is used to generate a cross node, as shown in Figure 6, centered at (*coor*) and labeled *node* that represents the pole of a system. The length and the line width of the cross are controlled by the `polelength` and `polewidth` keys, respectively. The key `scale` can be used to scale up or down the pole line-width and the pole length. The keys corresponding to the `\pspole` macro are summarized in Table 7.

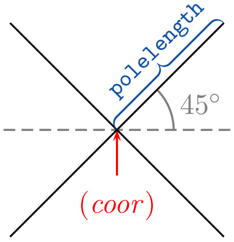
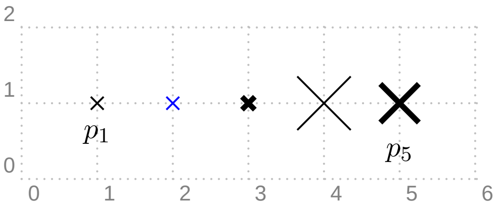


Figure 6. `\pspole` macro

Table 7. `\pspole` keys

Key	*Value*	*Default*	*Description*
`polelength`	*num[dimen]*	0.12	Cross length
`polewidth`	*num[dimen]*	0.7pt	Cross line-width
`scale`	*num*	1	Scale factor



```
\begin{pspicture}[showgrid](6,2)
  \pspole(1,1){p1}
    \nput{-90}{p1}{$p_1$}
  \pspole[linecolor=blue](2,1){p2}
  \pspole[polewidth=2pt](3,1){p3}
  \pspole[polelength=.5](4,1){p4}
  \pspole[scale=3](5,1){p5}
    \nput{-90}{p5}{$p_5$}
\end{pspicture}
```

3.8 \pscircleop

\pscircleop [*keys*] (*coor*){*node*}

This macro draws a cross inside a circle. Both the circle and the cross are centered at (*coor*). Then, it turns the circle into a node labeled *node* as shown in Figure 7. The length of the cross and its line-width are controlled by the `oplength` and `opwidth` keys, respectively. The line-width of the enclosing circle is separately controlled by the `linewidth` key. The distance between the circle and the cross is determined by the `opsep` key. The type of operation (whether plus or times) is controlled by the `operation` key. Another way of determining the operation inside the circle is through the key `angle` that determines the angle of the cross. The key `scale` can be used to scale up or down the cross line-width, the cross length, the separation between the cross and the circle, and the circle line-width. The keys corresponding to the `\pscircleop` macro are summarized in Table 8.

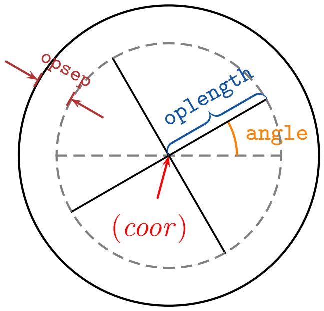
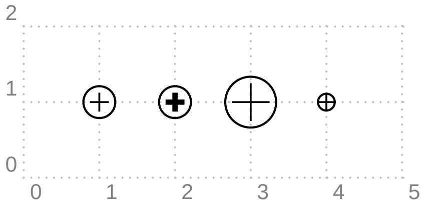


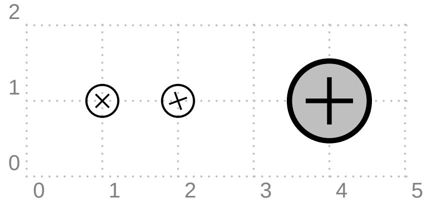
Figure 7. \pscircleop macro

Table 8. \pscircleop and \psframeop keys

Key	*Value*	*Default*	*Description*
`oplength`	*num[dimen]*	0.125	Cross length
`opwidth`	*num[dimen]*	0.7pt	Cross line-width
`opsep`	*num[dimen]*	0.1	Separation between the cross and the frame
`operation`	`plus|times`	`plus`	Operation
`angle`	*angle*	0	Cross angle
`scale`	*num*	1	Scale factor



```
\begin{pspicture}[showgrid](5,2)
  \pscircleop(1,1){op1}
  \pscircleop[opwidth=2pt](2,1){op2}
  \pscircleop[oplength=.25](3,1){op3}
  \pscircleop[opsep=0](4,1){op4}
\end{pspicture}
```

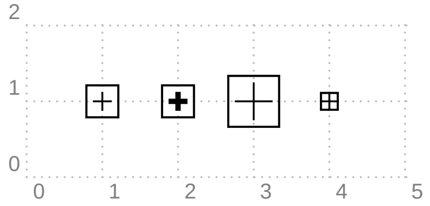


```
\begin{pspicture}[showgrid](5,2)
  \pscircleop[operation=times](1,1){op5}
  \pscircleop[angle=20](2,1){op6}
  \psset{fillstyle=solid,fillcolor=gray!50}
  \pscircleop[scale=2.5](4,1){op7}
\end{pspicture}
```

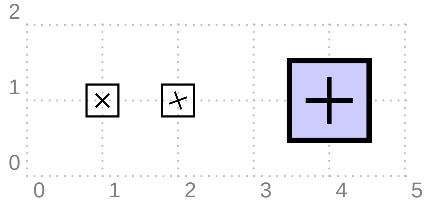
3.9 \psframeop

\psframeop [*keys*] (*coor*){*node*}

This macro is very similar to the \pscircleop macro with the same keys as in Table 8. The only difference is that the operation is enclosed in a square frame rather than a circular one.



```
\begin{pspicture}[showgrid](5,2)
  \psframeop(1,1){op1}
  \psframeop[opwidth=2pt](2,1){op2}
  \psframeop[oplength=.25](3,1){op3}
  \psframeop[opsep=0](4,1){op4}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](5,2)
  \psframeop[operation=times](1,1){op5}
  \psframeop[angle=20](2,1){op6}
  \psset{fillstyle=solid,fillcolor=blue!20}
  \psframeop[scale=2.5](4,1){op7}
\end{pspicture}
```

3.10 \psdisk

\psdisk [*keys*] (*coor*){*radius*}

This macro is used to shade the region of convergence of a system in the z plane. It draws a solid disk centered at (*coor*) with radius *radius* as depicted in Figure 8. The fill color is specified by the fillcolor key.

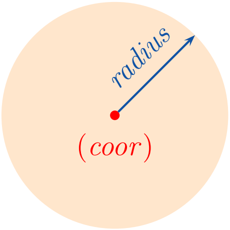
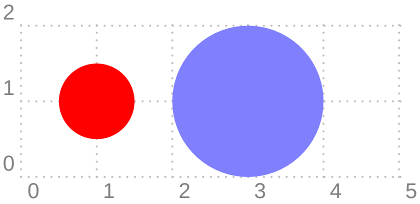


Figure 8. `\psdisk` macro



```latex
\begin{pspicture}[showgrid](5,2)
  \psdisk[fillcolor=red](1,1){.5}
  \psdisk[fillcolor=blue!50](3,1){1}
\end{pspicture}
```

3.11 `\psring`

`\psring` [*keys*] (*coor*){*inner-radius*}{*outer-radius*}

This macro is used to shade the region of convergence of a system in the z plane. It draws a solid ring centered at (*coor*) with inner radius *inner-radius* and outer radius *outer-radius* as shown in Figure 9. The fill color is specified by the `fillcolor` key.

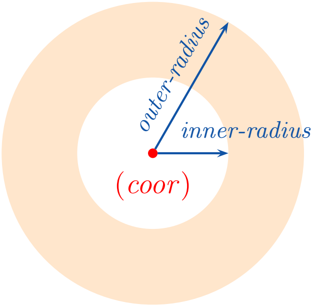
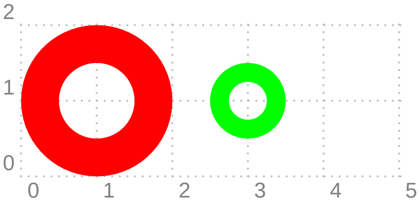


Figure 9. `\psring` macro



```latex
\begin{pspicture}[showgrid](5,2)
  \psring[fillcolor=red](1,1){.5}{1}
  \psring[fillcolor=green](3,1){.25}{.5}
\end{pspicture}
```

3.12 `\psdiskc`

`\psdiskc` [*keys*] (*coor*)(x_0, y_0){*radius*}

This macro is used to shade the region of convergence of a system in the z plane. As shown in Figure 10, it shades the area confined between a circle centered at (*coor*) with radius *radius* and a rectangle centered at (*coor*) with width $2x_0$ and height $2y_0$. The fill color is specified by the `fillcolor` key.

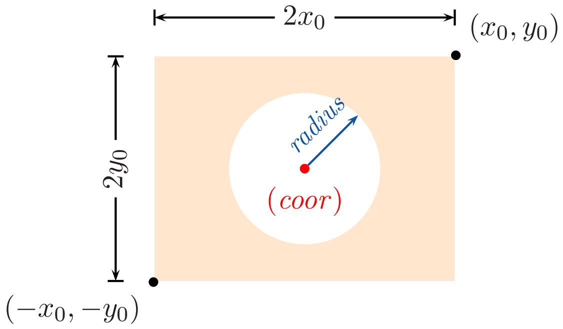
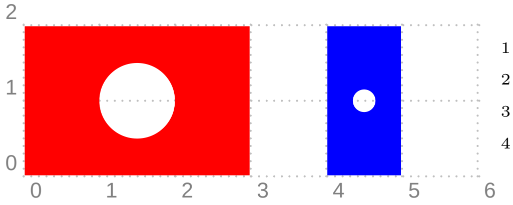


Figure 10. `\psdiskc` macro



```
\begin{pspicture}[showgrid](6,2)
  \psdiskc[fillcolor=red](1.5,1)(1.5,1){.5}
  \psdiskc[fillcolor=blue](4.5,1)(.5,1){.15}
\end{pspicture}
```

3.13 `\psldots`

`\psldots` [*keys*] {*angle*} (*coor*)

As depicted in Figure 11, this macro draws three dots each with diameter `ldotssize` on the same straight line, where the middle one is centered at (*coor*). Every two consecutive dots are separated by `ldotssep`. The angle of the line on which the dots lie with the horizontal axis is controlled by the optional parameter *angle*. In case it is absent, the angle is determined by the key `angle`. The key `scale` can be used to scale up or down the dot diameter and the dot separation. The keys corresponding to `\psldots` are summarized in Table 9.

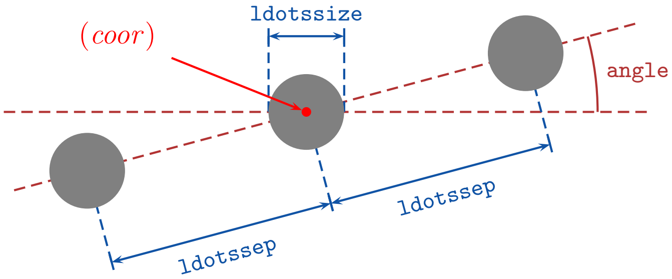
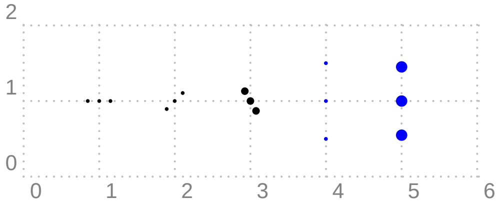


Figure 11. `\psldots` macro

Table 9. `\psldots` keys

Key	*Value*	*Default*	*Description*
`ldotssize`	*num[dimen]*	0.05	Dot diameter
`ldotssep`	*num[dimen]*	0.15	Distance between consecutive dots
`angle`	*angle*	0	Dots angle
`scale`	*num*	1	Scale factor



```
\begin{pspicture}[showgrid](6,2)
  \psldots(1,1)
  \psldots{45}(2,1)
  \psldots[ldotssize=.1]{120}(3,1)
  \psset{linecolor=blue,angle=90}
  \psldots[ldotssep=.5](4,1)
  \psldots[scale=3](5,1)
\end{pspicture}
```

3.14 `\ldotsnode`

`\ldotsnode` [*keys*] {*angle*} (*coor*){*node*}

This macro is very similar to the `\psldots` macro. The only difference is that the `\ldotsnode` places the dots inside an invisible frame and turns that frame into a node labeled *node* as shown in Figure 12. The frame is separated from the dots by half `signalsep`.

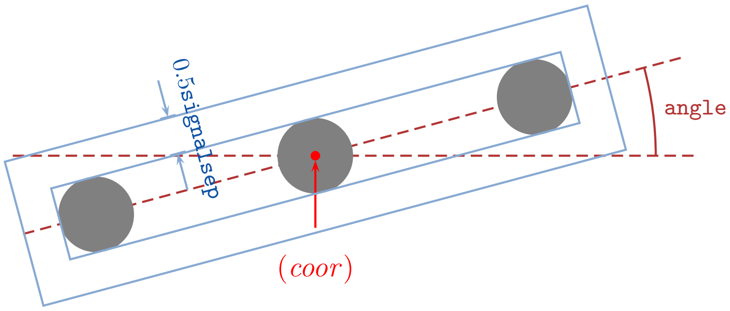
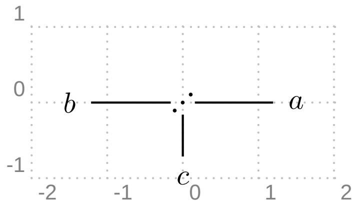


Figure 12. `\ldotsnode` macro

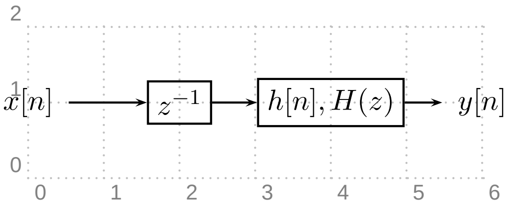


```
\begin{pspicture}[showgrid](-2,-1)(2,1)
   \pssignal(1.5;0){a}{$a$}
   \pssignal(1.5;180){b}{$b$}
   \pssignal(1;270){c}{$c$}
   \ldotsnode{45}(0,0){dots}
   \ncline{a}{dots}  \ncline{b}{dots}
   \ncline{c}{dots}
\end{pspicture}
```

3.15 `\psblock`

`\psblock` [*keys*] (*coor*){*node*}{*stuff*}

This macro places *stuff* at coordinate (*coor*), encloses it in a rectangular frame, and turns that frame into a node labeled *node*. The separation between the *stuff* and the frame is controlled by the `framesep` key.



```
\begin{pspicture}[showgrid](6,2)
   \pssignal(0,1){x}{$x[n]$}
   \psblock(2,1){a}{$z^{-1}$}
   \psblock(4,1){b}{$h[n], H(z)$}
   \pssignal(6,1){y}{$y[n]$}
   %-----------------
   \psset{arrows=->}
   \ncline{x}{a}  \ncline{a}{b}
\ncline{b}{y}
\end{pspicture}
```

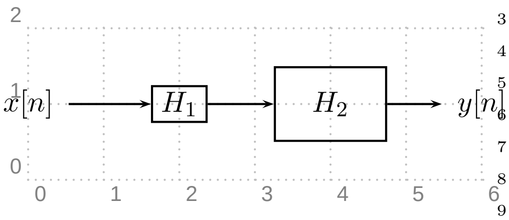
3.16 `\psfblock`

`\psfblock` [*keys*] (*coor*){*node*}{*stuff*}

This macro is very similar to the `\psblock` macro except that the size of the frame is controlled by the key `framesize`. The frame size is specified as

`framesize=`*num1[dimen] num2[dimen]*

in which *num1* and *num2* are separated by a space, not by a comma. If *num2* is absent, then a square frame is created.



```
\begin{pspicture}[showgrid](6,2)
  \pssignal(0,1){x}{$x[n]$}
  \psfblock[framesize=.75 .5](2,1){a}{$H_1$}
  \psfblock[framesize=1.5 1](4,1){b}{$H_2$}
  \pssignal(6,1){y}{$y[n]$}
  %-----------------
  \psset{arrows=->}
  \ncline{x}{a}  \ncline{a}{b}
  \ncline{b}{y}
\end{pspicture}
```

3.17 `\psadaptive`

`\psadaptive` [*keys*] {*arrows*} {*node A*}(*coor*){*node B*}

This macro is useful in drawing adaptive systems. It creates *node B* at coordinate (*coor*) with respect to the center of *node A*. Then, it connects *node B* to *node A* and continues to *node C* on the same line (Figure 13). The proportion of the distances of *node B* and *node C* from *node A* is determined by the key `afac`. A horizontal offset to the location of *node B* is achieved through the `aoffset` key (Table 10).

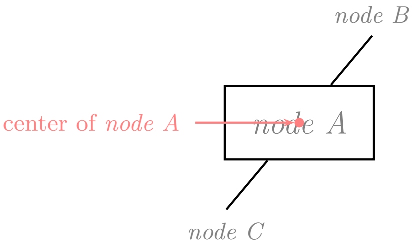
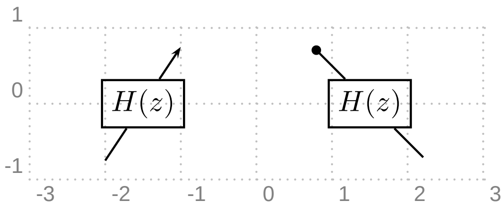


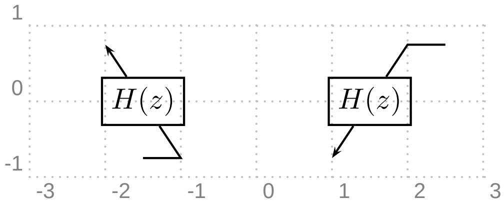
Figure 13. `\psadaptive` macro

Table 10. `\psadaptive` keys

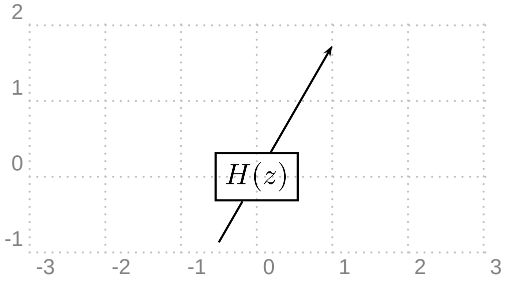
Key	*Value*	*Default*	*Description*
`aoffset`	*num*	0	Horizontal offset
`afac`	*num*	1	Length factor



```
\begin{pspicture}[showgrid](-3,-1)(3,1)
  \psblock(-1.5,0){H}{$H(z)$}
  \psadaptive{->}{H}(-.5,-.75){Ha}
  %-----------------
  \psblock(1.5,0){H}{$H(z)$}
  \psadaptive{-*}{H}(1;-45){Ha}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](-3,-1)(3,1)
  \psblock(-1.5,0){H}{$H(z)$}
  \psadaptive[aoffset=-.5]{->}{H}
      (.5,-.75){Ha}
  %-----------------
  \psblock(1.5,0){H}{$H(z)$}
  \psadaptive[aoffset=.5]{->}{H}
      (.5,.75){Ha}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](-3,-1)(3,2)
  \psblock(0,0){H}{$H(z)$}
  \psadaptive[afac=2]{->}{H}
       (1;-120){Ha}
\end{pspicture}
```

3.18 \psknob

`\psknob` [*keys*] (*coor*){*node*}

This macro is useful in drawing adjustable weights in adaptive systems. It creates a circle node centered at (*coor*) and labeled *node*. The radius of this circle is determined by the `radius` key. Then, it draws a straight arrow centered at (*coor*) (Figure 14). The length and the angle of this line are controlled by the `knoblength` and the `knobangle` keys. The line width of both the circle and the line are controlled by the `knobwidth` key. The key `scale` can be used to control the length of the line and the line width of both the circle and the line. The keys specific to `\psknob` are summarized in Table 11.

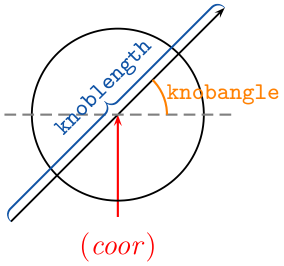
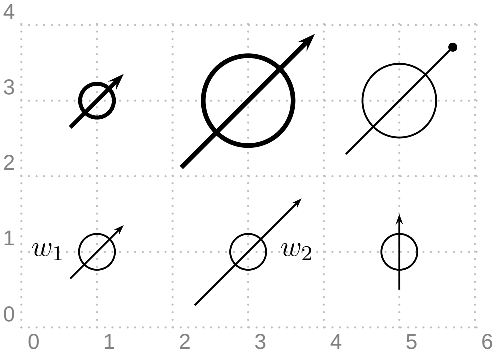


Figure 14. `\psknob` macro

Table 11. `\psknob` keys

Key	*Value*	*Default*	*Description*
`knobwidth`	*num[dimen]*	0.7pt	Line width
`knoblength`	*num[dimen]*	1	Line length
`knobangle`	*num*	45	Line angle
`radius`	*num[dimen]*	0.25cm	Circle radius
`scale`	*num*	1	Scale factor

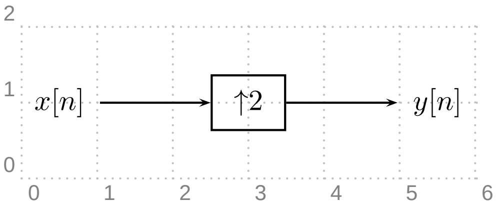


```
\begin{pspicture}[showgrid](6,4)
  \psknob(1,1){w1}
    \nput{180}{w1}{$w_1$}
  \psknob[knoblength=2](3,1){w2}
    \nput{0}{w2}{$w_2$}
  \psknob[knobangle=90](5,1){w3}
  \psknob[knobwidth=1.5pt](1,3){w4}
  \psknob[scale=2.5](3,3){w5}
  \psset{radius=.5,knoblength=2}
  \psknob[arrows=-*](5,3){w6}
\end{pspicture}
```

3.19 \psusampler

\psusampler [*keys*] (*coor*){*node*}{*stuff*}

This macro is similar to the \psfblock except that *stuff* is placed next to an up-arrow in the math mode representing an up-sampler. *It is important to remember that stuff must be in the text mode, not in the math mode, i.e., do not put $ around stuff.*

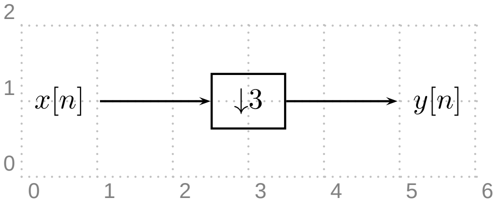


```
\begin{pspicture}[showgrid](6,2)
  \pssignal(.5,1){x}{$x[n]$}
  \psusampler[framesize=1 .75](3,1){a}{2}
  \pssignal(5.5,1){y}{$y[n]$}
  %-----------------
  \psset{arrows=->}
  \ncline{x}{a}
  \ncline{a}{y}
\end{pspicture}
```

3.20 \psdsampler

\psdsampler [*keys*] (*coor*){*node*}{*stuff*}

This macro is similar to the \psfblock except that *stuff* is placed next to a down-arrow in the math mode representing a down-sampler. *It is important to remember that stuff must be in the text mode, not in the math mode, i.e., do not put $ around stuff.*



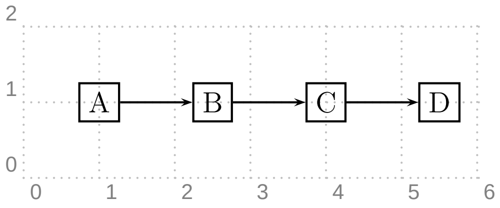
```
\begin{pspicture}[showgrid](6,2)
  \pssignal(.5,1){x}{$x[n]$}
  \psdsampler[framesize=1 .75](3,1){a}{3}
  \pssignal(5.5,1){y}{$y[n]$}
  %-----------------
  \psset{arrows=->}
  \ncline{x}{a}
  \ncline{a}{y}
\end{pspicture}
```

3.21 \nclist

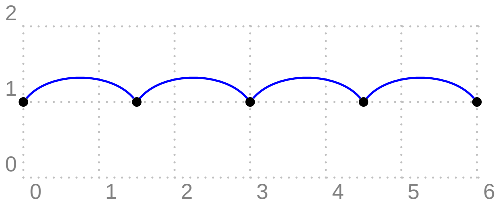
`\nclist` [*keys*] {*arrows*} {*nc-macro*}{*list*}
`\nclist` [*keys*] {*arrows*} {*nc-macro*}[*nc-label*]{*list*}

This macro is very useful when sequentially connecting several nodes using a single node-connecting macro. In addition, it is capable of labeling the node connections. The *list* must be a comma-separated list of items. Possible uses of the `\nclist` are summarized below.

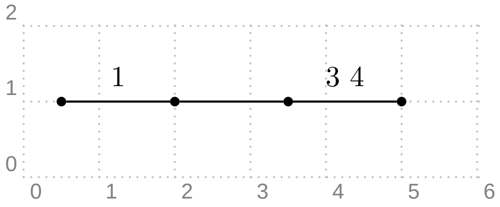
- `\nclist` [*keys*] {*arrows*} {*nc-macro*}$\{n_1, n_2, n_3, \ldots\}$ connects the node n_{i-1} to the node n_i, for all $i = 2, 3, \ldots$, using the macro *nc-macro*.
- `\nclist` [*keys*] {*arrows*} {*nc-macro*}[*nc-label*]$\{n_1, n_2\ l_2, n_3\ l_3, \ldots\}$ connects the node n_{i-1} to the node n_i, for all $i = 2, 3, \ldots$, using the macro *nc-macro*. Moreover, it puts the label l_i on the connection n_{i-1}–n_i, for all $i = 2, 3, \ldots$, using the macro *nc-label*. It is important to remember the following:
  1. In the list, the node n_i and the label l_i are separated by a space. If the label contains spaces, then it must be enclosed in double curly braces, i.e., $n_i\ \{\{l_i\}\}$.
  2. The first element of the list must be a single node (n_1); it should not have any labels.
- `\nclist` [*keys*] {*arrows*} {*nc-macro*}[*nc-label*]$\{n_1, n_2\ \texttt{ncl}_2\ l_2, n_3\ \texttt{ncl}_3\ l_3, \ldots\}$ connects the node n_{i-1} to the node n_i, for all $i = 2, 3, \ldots$, using the macro *nc-macro*. Moreover, it puts the label l_i on the connection n_{i-1}–n_i using the macro $\texttt{ncl}_i$ for all $i = 2, 3, \ldots$. If for some i, $\texttt{ncl}_i$ is empty, then the macro *nc-label* is used. In other words, the *nc-label* is the default macro for labeling connections when such macro is not explicitly present in the list. It is important to remember the following:
  1. In the list, the node n_i, the connection-labeling macro $\texttt{ncl}_i$, and the label l_i are separated by spaces. If the label contains spaces, then it must be enclosed in double curly braces, i.e., $n_i\ \texttt{ncl}_i\ \{\{l_i\}\}$.
  2. The first element of the list must be a single node (n_1); it should not have any labels.



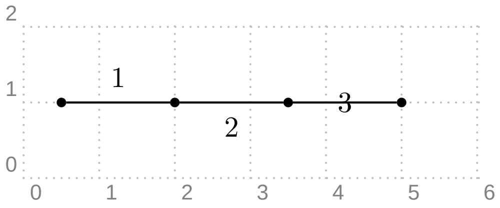
```
\begin{pspicture}[showgrid](6,2)
  \psblock(1,1){a}{A}
  \psblock(2.5,1){b}{B}
  \psblock(4,1){c}{C}
  \psblock(5.5,1){d}{D}
  \nclist{->}{ncline}{a,b,c,d}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](6,2)
  \dotnode(0,1){a}
  \dotnode(1.5,1){b}
  \dotnode(3,1){c}
  \dotnode(4.5,1){d}
  \dotnode(6,1){e}
  \psset{arcangle=50,linecolor=blue}
  \nclist{ncarc}{a,b,c,d,e}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](6,2)
    \dotnode(.5,1){a}
    \dotnode(2,1){b}
    \dotnode(3.5,1){c}
    \dotnode(5,1){d}
    \nclist{ncline}[naput]%
       {a,b $1$,c,d {{$3$ $4$}}}
\end{pspicture}
```



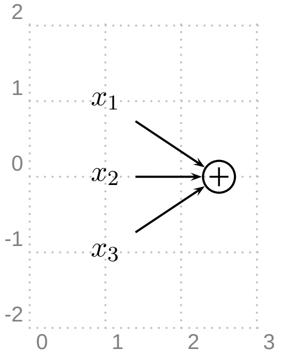
```
\begin{pspicture}[showgrid](6,2)
    \dotnode(.5,1){a}
    \dotnode(2,1){b}
    \dotnode(3.5,1){c}
    \dotnode(5,1){d}
    \nclist{ncline}[naput]%
       {a,b $1$,c nbput $2$,d ncput $3$}
\end{pspicture}
```

3.22 \ncstar

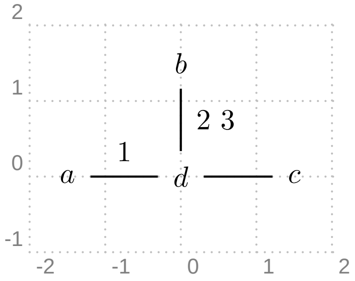
\ncstar [*keys*] {*arrows*} {*nc-macro*}{*list*}{*Node*}
\ncstar [*keys*] {*arrows*} {*nc-macro*}[*nc-label*]{*list*}{*Node*}

This macro is used to connect several nodes to a single node. It is also capable of labeling the node connections. The *list* must be a comma-separated list of items. Possible uses of the `\ncstar` are summarized below.

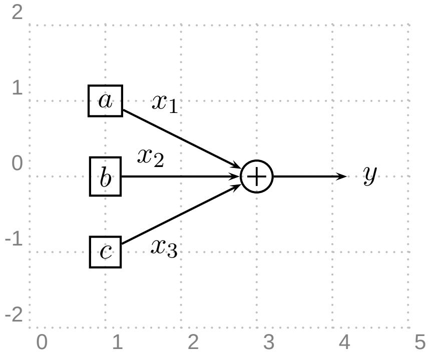
- `\ncstar` [*keys*] {*arrows*} {*nc-macro*}{$n_1, n_2, \ldots$}{N} connects the node n_i to the node N, for all $i = 1, 2, \ldots$, using the macro *nc-macro*.
- `\ncstar` [*keys*] {*arrows*} {*nc-macro*}[*nc-label*]{$n_1\ l_1, n_2\ l_2, \ldots$}{$N$} connects the node n_i to node N, for all $i = 1, 2, \ldots$, using the macro *nc-macro*. Moreover, it puts the label l_i on the connection n_i–N, for all $i = 1, 2, \ldots$, using the macro *nc-label*. It is important to remember that the node n_i and the label l_i are separated by a space in the list. If the label contains spaces, then it must be enclosed in double curly braces, i.e., n_i $\{\{l_i\}\}$.
- `\ncstar` [*keys*] {*arrows*} {*nc-macro*}[*nc-label*]{n_1 $\texttt{ncl}_1$ l_1, n_2 $\texttt{ncl}_2$ $l_2, \ldots$}{N} connects node n_i to node N, for all $i = 1, 2, \ldots$, using the macro *nc-macro*. Moreover, it puts the label l_i on the connection n_i–N using the macro $\texttt{ncl}_i$ for all $i = 1, 2, \ldots$. If for some i, $\texttt{ncl}_i$ is empty, then the macro *nc-label* is used. In other words, the *nc-label* is the default macro for labeling connections when such macro is not explicitly present in the list. It is important to remember that the node n_i and the label l_i are separated by a space in the list. If the label contains spaces, then it must be enclosed in double curly braces, i.e., n_i $\texttt{ncl}_i$ $\{\{l_i\}\}$.



```
\begin{pspicture}[showgrid](0,-2)(3,2)
   \pssignal(1,1){x1}{$x_1$}
   \pssignal(1,0){x2}{$x_2$}
   \pssignal(1,-1){x3}{$x_3$}
   \pscircleop(2.5,0){oplus}
   \ncstar{->}{ncline}{x1,x2,x3}{oplus}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](-2,-1)(2,2)
   \pssignal(-1.5,0){a}{$a$}
   \pssignal(0,1.5){b}{$b$}
   \pssignal(1.5,0){c}{$c$}
   \pssignal(0,0){d}{$d$}
   \ncstar{ncline}[naput]%
      {a $1$,b {{$2$ $3$}},c}{d}
\end{pspicture}
```



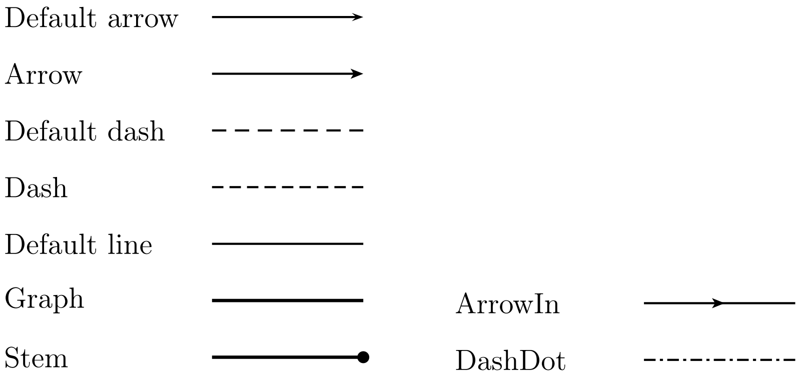
```
\begin{pspicture}[showgrid](0,-2)(5,2)
   \psblock(1,1){a}{$a$}
   \psblock(1,0){b}{$b$}
   \psblock(1,-1){c}{$c$}
   \pscircleop(3,0){oplus}
   \pssignal(4.5,0){y}{$y$}
   \psset{labelsep=.1,npos=.25}
   \ncstar{->}{ncline}[naput]%
      {a $x_1$,b $x_2$,c nbput $x_3$}{oplus}
   \ncline{->}{oplus}{y}
\end{pspicture}
```

4 Extras

In addition to the macros introduced in Section 3, the `pst-sigsys` package defines some extra styles, brace macros, and new keys that are introduced in this section. Their usages are shown in Section 5 with some examples.

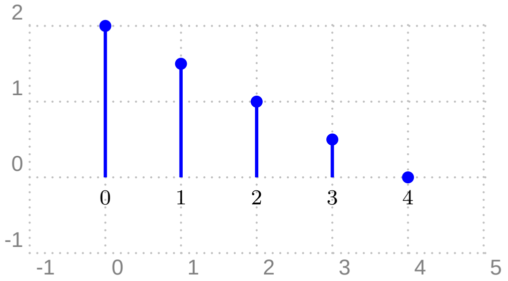
4.1 New Styles

The `pst-sigsys` package defines a few useful PSTricks styles for drawling arrows and dashed lines as shown in Figure 15. Some of these styles, which are shown in Figure 15b, can be used only with the `pstricks-add` package.



(a) `pstricks` styles (b) `pstricks-add` styles

Figure 15. New styles

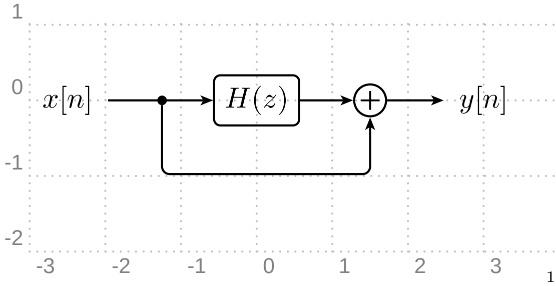


```
\begin{pspicture}[showgrid](-1,-1)(5,2)
  \psset{style=Stem,linecolor=blue}
  \psstem[stemtag]{2,1.5,1,.5,0}
\end{pspicture}
```

In addition, the pst-sigsys package defines the style RoundCorners that makes the following settings:

```
framesep=0.125
framearc=0.25
linearc=0.1
```

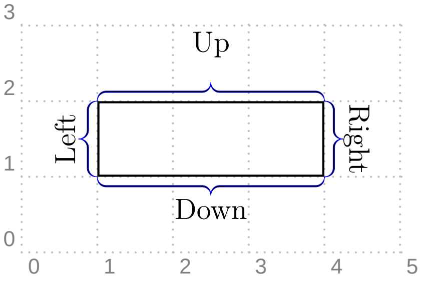
The author believes that when drawing block diagrams, it is more elegant to have round corners.



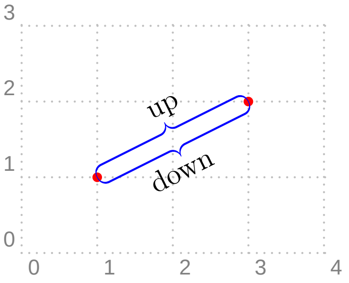
```
\begin{pspicture}[showgrid](-3,-2)(4,1)
  \psset{style=RoundCorners,style=Arrow}
  \pssignal(-2.5,0){x}{$x[n]$}
  \dotnode(-1.25,0){dot}
  \psblock(0,0){H}{$H(z)$}
  \pscircleop(1.5,0){oplus}
  \pssignal(3,0){y}{$y[n]$}
  \nclist{ncline}{x,H,oplus,y}
  \ncbar[angle=-90,arm=.75]{dot}{oplus}
\end{pspicture}
```

4.2 Brace Macros

The pst-sigsys package defines four new macros \psBraceUp, \psBraceDown, \psBraceRight, and \psBraceLeft that are derived from the \psbrace macro (using the \newpsobject macro) defined by the pstricks-add package. They all have the same syntax that is the same as that of the \psbrace macro. The usage of these macros is shown by the following examples:



```
\begin{pspicture}[showgrid](5,3)
  \psframe(1,1)(4,2)
  \psset{linecolor=blue}
  \psBraceUp[nodesepB=-.5](4,2)(1,2){Up}
  \psBraceDown(1,1)(4,1){Down}
  \psBraceRight(4,1)(4,2){Right}
  \psBraceLeft(1,2)(1,1){Left}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](4,3)
  \psset{linecolor=red}
  \dotnode(1,1){a}
  \dotnode(3,2){b}
  %--------------------
  \psset{linecolor=blue}
  \psBraceUp*(b)(a){up}
  \psBraceDown*(a)(b){down}
\end{pspicture}
```

4.3 Golden Ratio

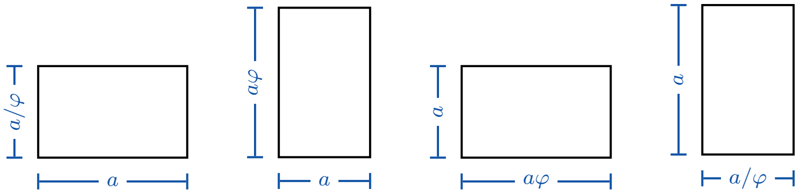
The pst-sigsys package defines four keys gratioWh, gratioWv, gratioHh, and gratioHv for determining the frame size by the golden ratio φ defined as

$$\varphi = \frac{1+\sqrt{5}}{2} \approx 1.61803398875 \ .$$

The ancient Greeks thought a rectangle is the most pleasing to the eye if its edges a and b were in the proportion $a\colon b = \varphi$ [?]. In the gratio keys, the capital letters W and H stand for the width and the height of the frame, respectively. The ending letters h and v imply whether the frame is horizontal or vertical, respectively. In a horizontal frame, the longest edge is horizontal while in a vertical one, the longest edge is vertical.

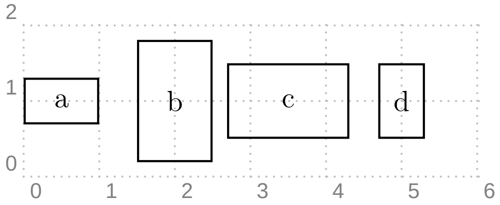
The four aforementioned keys set one of the edges of a frame as specified by the user and determine the other one by the golden ratio φ as follows:

- The key assignment gratioWh $= a$ sets the width of the frame to a and the height to a/φ as in Figure 16a.
- The key assignment gratioWv $= a$ sets the width of the frame to a and the height to $a\varphi$ as in Figure 16b.
- The key assignment gratioHh $= a$ sets the height of the frame to a and the width to $a\varphi$ as in Figure 16c.
- The key assignment gratioHv $= a$ sets the height of the frame to a and the width to a/φ as in Figure 16d.



(a) gratioWh $= a$ (b) gratioWv $= a$ (c) gratioHh $= a$ (d) gratioHv $= a$

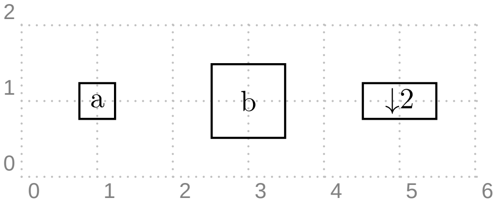
Figure 16. Setting the size of a frame by the golden ratio φ



```
\begin{pspicture}[showgrid](6,2)
  \psfblock[gratioWh=1](.5,1){a}{a}
  \psfblock[gratioWv=1](2,1){b}{b}
  \psfblock[gratioHh=1](3.5,1){c}{c}
  \psfblock[gratioHv=1](5,1){d}{d}
\end{pspicture}
```

4.4 Frame Width and Height

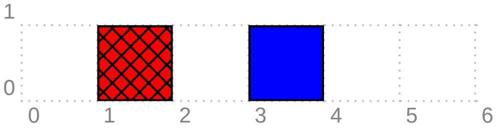
When drawing block diagrams, it is sometimes useful to change only the width or the height of a frame. This goal is achieved through the keys `framewidth` and `frameheight`.



```
\begin{pspicture}[showgrid](6,2)
  \psset{framesize=1 .5}
  \psfblock[framewidth=.5](1,1){a}{a}
  \psfblock[frameheight=1](3,1){b}{b}
  \psdsampler(5,1){c}{2}
\end{pspicture}
```

4.5 Fill Color

To emphasize the functions of some blocks in a diagram, it is useful to color them. For this purpose, both the `fillstyle` and the `fillcolor` keys must be set. This could be cumbersome when many blocks are to be colored. Since almost always the fill style is solid, it makes sense to define a single key that automatically sets the fill style to solid. The key `FillColor` plays this role.



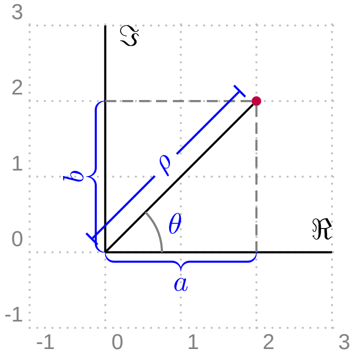
```
\begin{pspicture}[showgrid](6,1)
  \psset{fillstyle=crosshatch*}
  \psframe[fillcolor=red](1,0)(2,1)
  \psframe[FillColor=blue](3,0)(4,1)
\end{pspicture}
```

5 Examples

In this section, we provide some examples to illustrate the benefits and usages of the macros, styles, and keys defined in Sections 3 and 4. Some of these examples require the use of additional packages. In that case, additional packages are mentioned next to the example number.

5.1 Complex Number

Example 1. (use `pstricks-add`) Show the complex number $c = a + jb = \rho e^{j\theta}$ as a point in the complex plane.

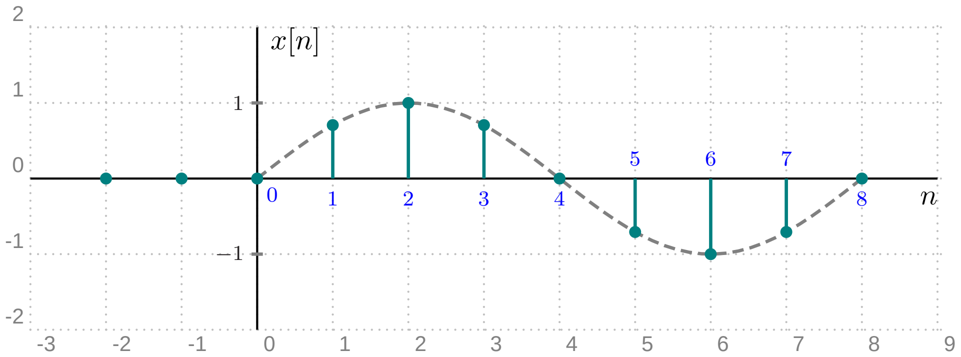


```
\begin{pspicture}[showgrid](-1,-1)(3,3)
  %--- Drawing axes ---
  \psaxeslabels[xlpos=t](0,0)(0,0)(3,3)
       {$\Re$}{$\Im$}

  %--- Defining some useful nodes ---
  \dotnode[linecolor=purple](2,2){c}
  \pnode(0,0){org}
  \pnode(2,0){a}
  \pnode(0,2){b}

  %--- Connecting nodes ---
  \ncline{org}{c}
  \ncstar[style=Dash,linecolor=gray]
      {ncline}{a,b}{c}

  %--- Labeling ---
  \color{blue}
  \psset{linecolor=blue,nrot=:U}
  \psBraceDown*(org)(a){$a$}
  \psBraceLeft*(b)(org){$b$}
  \ncline[offset=.25]{|*-|*}{org}{c}
  \ncput*{$\rho$}
  \psarc[linecolor=gray](org){.75}{0}{45}
  \rput(1;22.5){$\theta$}
\end{pspicture}
```

5.2 Plotting

Example 2. (use `pst-plot`) Draw the sampled sequence $x[n] = x_c(\pi n/4)$, where

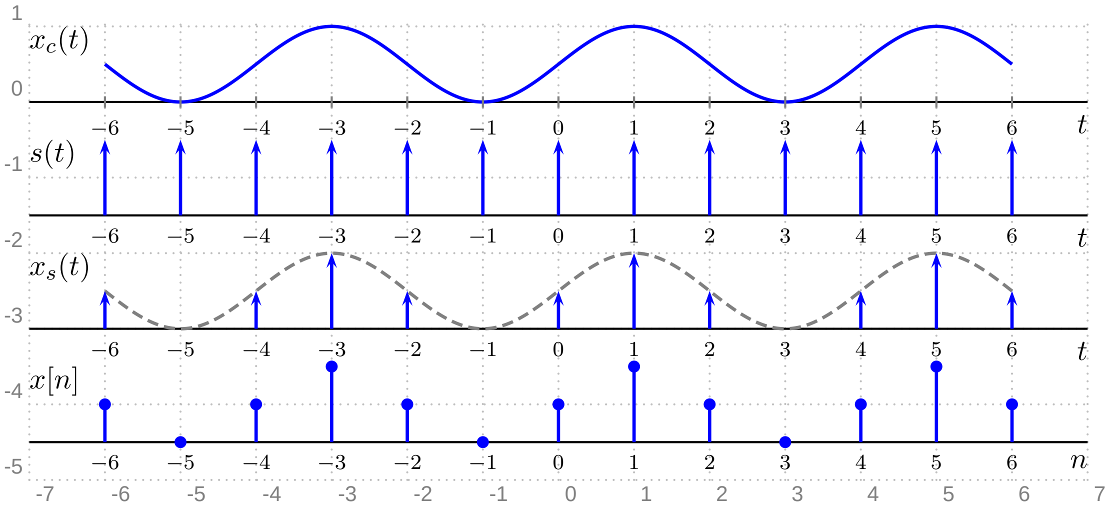
$$x_c(t) = \begin{cases} \sin(t) \ , & t \geq 0 \\ 0 \ , & t < 0 \ . \end{cases}$$



```
\begin{pspicture}[showgrid](-3,-2)(9,2)
  %--- Drawing axes ---
  \psaxeslabels(0,0)(-3,-2)(9,2){$n$}{$x[n]$}

  %--- x_c(t) ---
  \psplot[style=Graph,style=Dash,linecolor=gray]{0}{8}{x 45 mul sin}

  %--- x[n] ---
  \psset{style=Stem,linecolor=teal,
        stemtagformat={\color{blue}\scriptstyle}}
  \psstem(0,-1){0,0,0}
  \psstem[stemtag](1,1){.707107,1,.707107,0,-.707107,-1,-.707107,0}

  %--- Labeling the origin ---
  \uput[-45](0,0){$\color{blue}\scriptstyle 0$}

  %--- Horizontal ticks ----
  \psset{linecolor=gray}
  \psTick(0,1)
  \psTick(0,-1)
  \uput[180](0,1){$\scriptstyle 1$}
  \uput[180](0,-1){$\scriptstyle -1$}
\end{pspicture}
```

5.3 Sampling

Example 3. (use `pst-plot` and `multido`) Consider the process of sampling a continuous-time signal $x_c(t)$ with period T: (1) multiply $x_c(t)$ by the impulse train $s(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT)$ to get $x_s(t) = x_c(t)s(t)$, and (2) convert every delta in $x_s(t)$ into a sample to get the sequence $x[n]$. Demonstrate this process for the continuous-time signal $x_c(t) = 0.5 \sin(\pi t/2) + 0.5$ and $T = 1$.

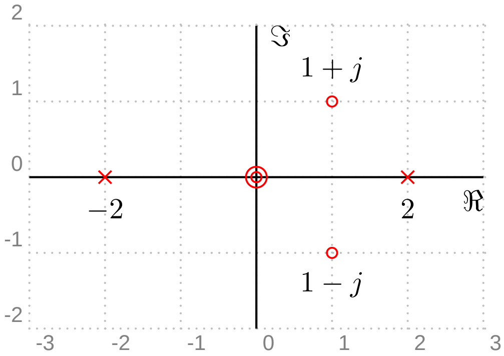


```
\begin{pspicture}[showgrid](-7,-5)(7,1)
  \psset{plotpoints=500,stemtag}
  %--- x_c(t) ---
  \psaxeslabels(0,0)(-7,0)(7,0){$t$}{}
  \rput[tl](-7,1){$x_c(t)$}
  \psplot[style=Graph,linecolor=blue]{-6}{6}{x 90 mul sin .5 mul .5 add}
  \multirput(-6,0)(1,0){13}{\psTick[linecolor=gray]{90}(0,0)}
  \multido{\nn=-6+1}{13}{\rput[t](\nn,-.25){$\scriptstyle\nn$}}

  %--- s(t) ----
  \rput(0,-1.5){\psaxeslabels(0,0)(-7,0)(7,0){$t$}{}
     \rput[tl](-7,1){$s(t)$}
     \psstem[style=Stem,stemhead=>,linecolor=blue](-6,1)
            {1,1,1,1,1,1,1,1,1,1,1,1,1}}

  %--- x_s(t) ---
  \rput(0,-3){\psaxeslabels(0,0)(-7,0)(7,0){$t$}{}
     \rput[tl](-7,1){$x_s(t)$}
     \psplot[style=Graph,style=Dash,linecolor=gray]{-6}{6}
            {x 90 mul sin .5 mul .5 add}
     \psset{style=Stem,stemhead=>,linecolor=blue}
     \psstem[killzero](-6,1){.5,0,.5,1,.5,0,.5,1,.5,0,.5,1,.5}}

  %--- x[n] ----
  \rput(0,-4.5){\psaxeslabels(0,0)(-7,0)(7,0){$n$}{}
     \rput[tl](-7,1){$x[n]$}
     \psstem[style=Stem,linecolor=blue](-6,1)
            {.5,0,.5,1,.5,0,.5,1,.5,0,.5,1,.5}}
\end{pspicture}
```

5.4 Pole-Zero Diagram

Example 4. Draw the pole-zero diagram of a system with the following system function:

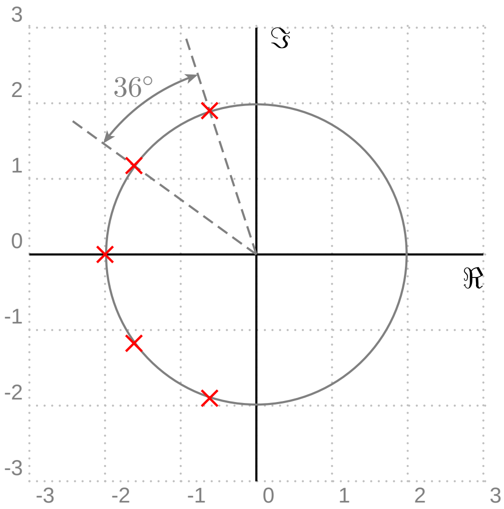
$$H(z) = \frac{z^4 - 2z^3 + 2z^2}{z^2 - 4} \ .$$



```
\begin{pspicture}[showgrid](-3,-2)(3,2)
  \psaxeslabels(0,0)(-3,-2)(3,2){$\Re$}{$\Im$}
  \psset{linecolor=red}

  %--- Placing zeros ---
  \pszero[order=2](0,0){z1}
  \pszero(1,1){z2}   \nput{90}{z2}{$1 + j$}
  \pszero(1,-1){z3}  \nput{-90}{z3}{$1 - j$}

  %--- Placing poles ---
  \pspole(2,0){p1}   \nput{-90}{p1}{$2$}
  \pspole(-2,0){p2}  \nput{-90}{p2}{$-2$}
\end{pspicture}
```

5.5 Butterworth Filter

Example 5. (use `multido`) Draw the pole-zero diagram of a fifth-order Butterworth filter.



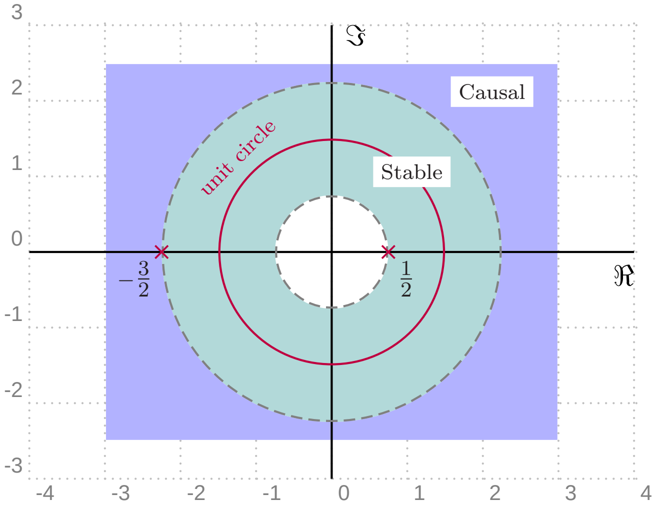
```latex
\begin{pspicture}[showgrid](-3,-3)(3,3)
  %--- Drawing axes ---
  \psaxeslabels(0,0)(-3,-3)(3,3){$\Re$}{$\Im$}
  \pscircle[linecolor=gray](0,0){2}

  %--- Angle between poles ---
  \psset{linecolor=gray}
  \psline[style=Dash](3;108)
  \psline[style=Dash](3;144)
  \psarc[style=Arrow]{<->}(0,0){2.5}{108}{144}
  \rput(2.75;126){\textcolor{gray}{$36^\circ$}}

  %--- Placing poles ---
  \psset{linecolor=red,scale=1.25}
  \multido{\np=108+36}{5}{\pspole(2;\np){p}}
\end{pspicture}
```

5.6 Region of Convergence

Example 6. Shade the region of convergence (ROC) of a system with the following system function assuming it is: (1) causal, and (2) stable.

$$H(z) = \frac{1}{z^2 + z - \frac{3}{4}}$$

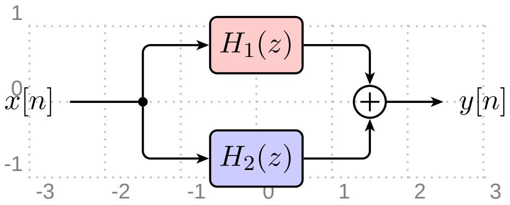
Since the poles of the system are at $z = \frac{1}{2}$ and $z = -\frac{3}{2}$, the ROC of the system with the given assumptions is as follows:

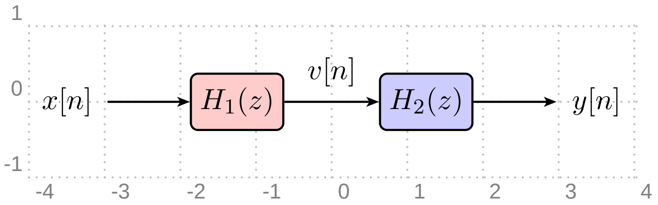


```
\begin{pspicture}[showgrid](-4,-3)(4,3)
  %--- Shading ROCs ---
  \psring[fillcolor=teal!30](0,0){.75}{2.25}
  \psdiskc[fillcolor=blue!30](0,0)(3,2.5){2.25}

  %--- Drawing axes ---
  \psaxeslabels(0,0)(-4,-3)(4,3){$\Re$}{$\Im$}

  %--- Placing poles ---
  \psset{linecolor=purple,labelsep=.05}
  \pscircle(0,0){1.5}
  \rput[b]{45}(1.68;135){{\scriptsize\color{purple}unit circle}}
  \pscircle[style=Dash,linecolor=gray](0,0){.75}
  \pspole(.75,0){p1}    \nput{-45}{p1}{$\tfrac{1}{2}$}
  \pscircle[style=Dash,linecolor=gray](0,0){2.25}
  \pspole(-2.25,0){p2}  \nput{225}{p2}{$\scriptstyle-\tfrac{3}{2}$}

  %--- Labeling the stable and causal ROCs ---
  \rput*(1.5;45){\scriptsize Stable}
  \rput*(3;45){\scriptsize Causal}
\end{pspicture}
```

5.7 Block Diagrams

Example 7. Draw the block diagrams of two systems $H_1(z)$ and $H_2(z)$ in both parallel and series combinations.

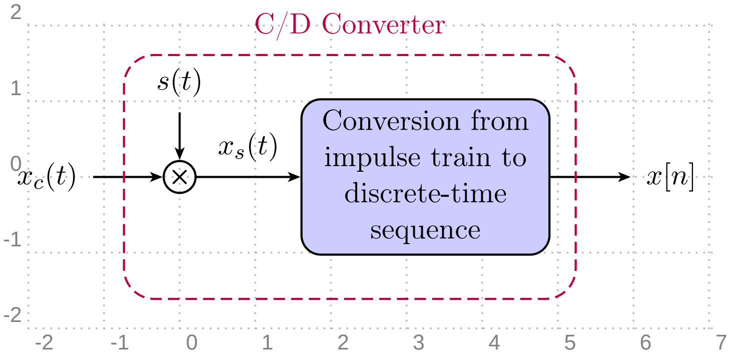




```
%=== Parallel Combination ===
\begin{pspicture}[showgrid](-3,-1)(3,1)
  \psset{style=RoundCorners,gratioWh=1.25}

  %--- Defining blocks ---
  \pssignal(-3,0){x}{$x[n]$}
  \dotnode(-1.5,0){dot}
  \psfblock[FillColor=red!20](0,.75){H1}{$H_1(z)$}
  \psfblock[FillColor=blue!20](0,-.75){H2}{$H_2(z)$}
  \pscircleop(1.5,0){oplus}
  \pssignal(3,0){y}{$y[n]$}

  %--- Connecting blocks ---
  \psset{style=Arrow}
  \ncline{-}{x}{dot}
  \ncangle[angleA=90,angleB=180]{dot}{H1}
  \ncangle[angleA=-90,angleB=180]{dot}{H2}
  \ncangle[angleB=90]{H1}{oplus}
  \ncangle[angleB=-90]{H2}{oplus}
  \ncline{oplus}{y}
\end{pspicture}
%
\hspace{1cm}
%
%=== Series Combination ===
\begin{pspicture}[showgrid](-4,-1)(4,1)
  \psset{style=RoundCorners,gratioWh=1.25}

  %--- Defining blocks ---
  \pssignal(-3.5,0){x}{$x[n]$}
  \psfblock[FillColor=red!20](-1.25,0){H1}{$H_1(z)$}
  \psfblock[FillColor=blue!20](1.25,0){H2}{$H_2(z)$}
  \pssignal(3.5,0){y}{$y[n]$}

  %--- Connecting blocks ---
  \nclist[style=Arrow]{ncline}[naput]{x,H1,H2 $v[n]$,y}
\end{pspicture}
```

5.8 C/D Converter

Example 8. Draw the block diagram of a continuous-to-discrete-time (C/D) converter.

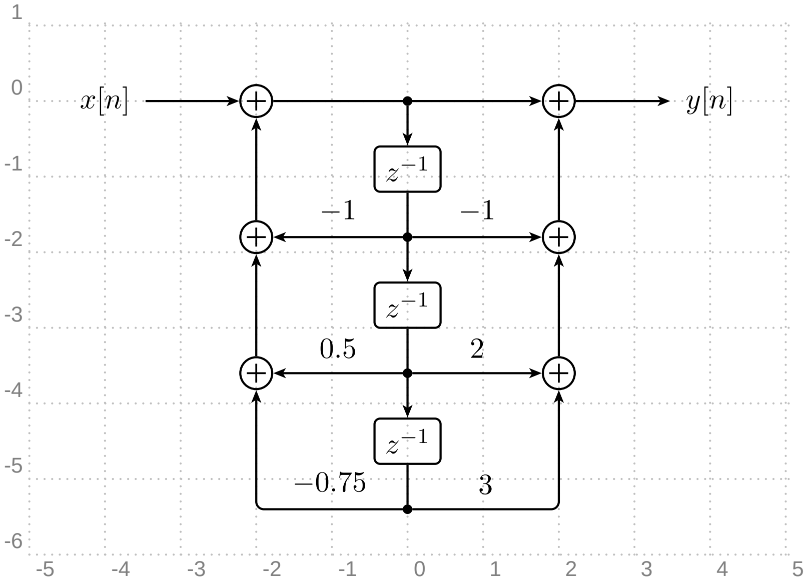


```
\begin{pspicture}[showgrid](-2,-2)(7,2)
  \psset{style=RoundCorners}

  %--- Defining blocks ---
  \pssignal(-1.75,0){xc}{$x_c(t)$}
  \pscircleop[operation=times](0,0){otimes}
  \pssignal(0,1.25){s}{$s(t)$}
  \psblock[FillColor=blue!20](3.25,0){conv}{\parbox[c]{3\psunit}%
  {\centering Conversion from impulse train to discrete-time sequence}}
  \pssignal(6.5,0){x}{$x[n]$}

  %--- Connecting blocks ---
  \psset{style=Arrow}
  \nclist{ncline}[naput]{xc,otimes,conv $x_s(t)$,x}
  \ncline{s}{otimes}

  %--- Drawing the dashed frame ---
  \fnode[style=Dash,linecolor=purple,framesize=6 3.25](2.25,0){box}
  \nput{90}{box}{\textcolor{purple}{C/D Converter}}
\end{pspicture}
```

5.9 Direct Form II

Example 9. (use `multido`) Draw the direct-form II block diagram of a discrete-time LTI system with the following system function:

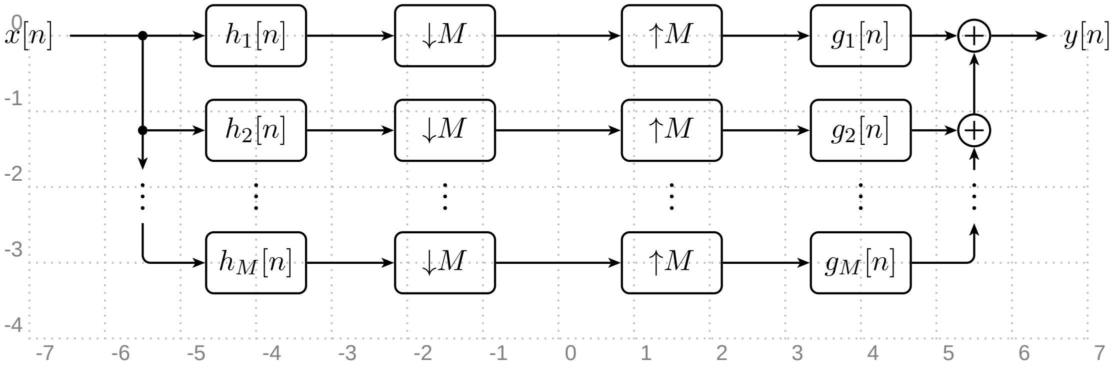
$$H(z) = \frac{1 - z^{-1} + 2z^{-2} + 3z^{-3}}{1 + z^{-1} - 0.5z^{-2} + 0.75z^{-3}} \ .$$



```
\begin{pspicture}[showgrid](-5,-6)(5,1)
  \psset{style=RoundCorners,style=Arrow}

  %--- Defining blocks ---
  \dotnode(0,0){dot1}
  \multido{\nA=1+1,\nB=2+1,\ryA=-.9+-1.8,\ryB=-1.8+-1.8}{3}{%
    \psblock(0,\ryA){D\nA}{$z^{-1}$}   \dotnode(0,\ryB){dot\nB}}
  \multido{\nn=1+1,\ry=0+-1.8}{3}{\pscircleop(-2,\ry){oplusL\nn}}
  \multido{\nn=1+1,\ry=0+-1.8}{3}{\pscircleop(2,\ry){oplusR\nn}}
  \pssignal(-4,0){x}{$x[n]$}
  \pssignal(4,0){y}{$y[n]$}

  %--- Connecting blocks ---
  \psset{style=Arrow}
  \nclist{ncline}{x,oplusL1,oplusR1,y}
  \nclist{ncline}{dot1,D1,D2,D3}
  \ncline{-}{D3}{dot4}
  \nclist{ncline}{oplusL3,oplusL2,oplusL1}
  \nclist{ncline}{oplusR3,oplusR2,oplusR1}
  \ncstar{<-}{ncline}[naput]{oplusL2 $-1$,oplusR2 nbput $-1$}{dot2}
  \ncstar{<-}{ncline}[naput]{oplusL3 $0.5$,oplusR3 nbput $2$}{dot3}
  \ncangle[angleA=180,angleB=-90]{dot4}{oplusL3}
  \nbput[npos=.5]{$-0.75$}
  \ncangle[angleB=-90]{dot4}{oplusR3}
  \naput[npos=.5]{$3$}
\end{pspicture}
```

5.10 Filter Bank

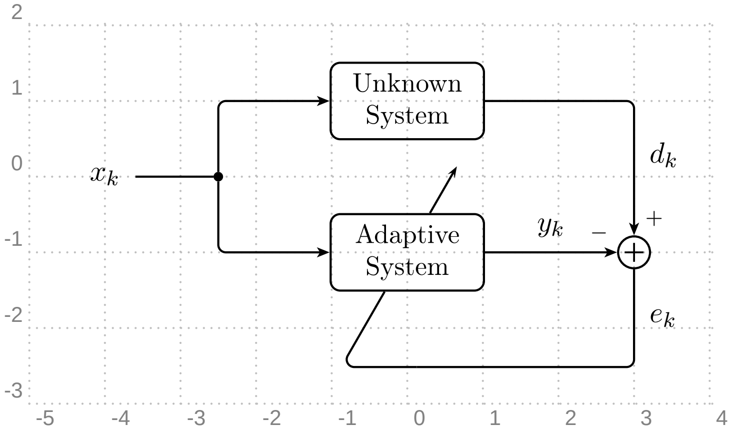
Example 10. Draw the block diagram of an M-channel maximally-decimated filter bank.



```
\begin{pspicture}[showgrid](-7,-3.5)(7,.5)
  \psset{style=RoundCorners,style=Arrow,gratioWh=1.35}
  \pssignal(-7,0){x}{$x[n]$}       \pssignal(7,0){y}{$y[n]$}
  \dotnode(-5.5,0){dot1}           \dotnode(-5.5,-1.25){dot2}
  \newcount\cnt

  %--- First and second channels ---
  \cnt=0
  \psforeach{\ry}{0,-1.25}{\advance\cnt by 1\relax
     \psfblock(-4,\ry){h\the\cnt}{$h_{\the\cnt}[n]$}
     \psdsampler(-1.5,\ry){ds\the\cnt}{M}
     \psusampler(1.5,\ry){us\the\cnt}{M}
     \psfblock(4,\ry){g\the\cnt}{$g_{\the\cnt}[n]$}
     \pscircleop(5.5,\ry){oplus\the\cnt}}

  %--- Placing dots ---
  \cnt=0
  \psforeach{\rx}{-5.5,-4,-1.5,1.5,4,5.5}{\advance\cnt by 1\relax
     \ldotsnode[angle=90](\rx,-2.125){dots\the\cnt}}

  %--- M-th channel ---
  \psfblock(-4,-3){hM}{$h_M[n]$}
  \psdsampler(-1.5,-3){dsM}{M}
  \psusampler(1.5,-3){usM}{M}
  \psfblock(4,-3){gM}{$g_M[n]$}

  %--- Connecting blocks ---
  \nclist{ncline}{x,h1,ds1,us1,g1,oplus1,y}
  \nclist{ncline}{dot2,h2,ds2,us2,g2,oplus2}
  \ncline{dot1}{dots1}
  \ncangle[angleA=-90,angleB=180]{dots1}{hM}
  \nclist{ncline}{hM,dsM,usM,gM}
  \ncangle[angleB=-90]{gM}{dots6}
  \nclist{ncline}{dots6,oplus2,oplus1}
\end{pspicture}
```

5.11 System Identification

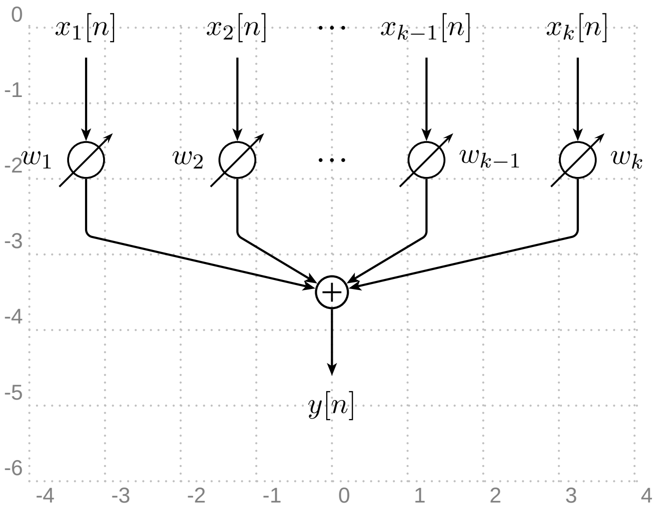
Example 11. Draw the block diagram of an adaptive system used for system identification.



```
\begin{pspicture}[showgrid](-5,-3)(4,2)
  \psset{style=RoundCorners}

  %--- Placing the input signal and drawing blocks ---
  \pssignal(-4,0){x}{$x_k$}
  \dotnode(-2.5,0){dot}
  \psblock(0,-1){AdapSys}
       {\parbox[c]{1.75\psunit}{\centering\small Adaptive System}}
  \psadaptive[aoffset=1,afac=.75]{->}{AdapSys}(-1.75;60){AdapSysA}
  \pscircleop(3,-1){oplus}
  \nput{150}{oplus}{$\scriptstyle -$}
  \nput{60}{oplus}{$\scriptstyle +$}
  \psblock(0,1){UnSys}
       {\parbox[c]{1.75\psunit}{\centering\small Unknown System}}

  %--- Connecting blocks ---
  \psset{style=Arrow}
  \ncline{-}{x}{dot}
  \ncangle[angleA=-90,angleB=180]{dot}{AdapSys}
  \ncangle[angleA=90,angleB=180]{dot}{UnSys}
  \ncangle[angleB=90]{UnSys}{oplus}   \naput[npos=1.5]{$d_k$}
  \ncline{AdapSys}{oplus}
  \naput{$y_k$}
  \ncangle[angleA=-90]{-}{oplus}{AdapSysA}
  \naput[npos=.5]{$e_k$}
\end{pspicture}
```

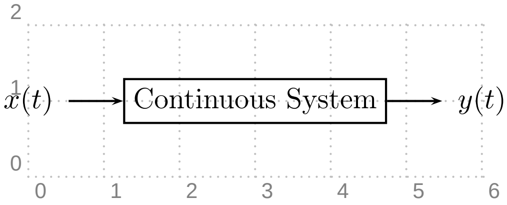
5.12 Adaptive Linear Combiner

Example 12. (use `multido`) Draw the block diagram of an adaptive linear combiner.



```
\begin{pspicture}[showgrid](-4,-6)(4,.5)
  \psset{style=RoundCorners,gratioWh=1,radius=.25}

  %--- Signals ---
  \pssignal(-3.25,0){x1}{$x_1[n]$}
  \pssignal(-1.25,0){x2}{$x_2[n]$}
  \pssignal(1.25,0){x3}{$x_{k-1}[n]$}
  \pssignal(3.25,0){x4}{$x_k[n]$}
  \pssignal(0,-5){y}{$y[n]$}

  %--- Gains, dots, and the adder ---
  \psknob(-3.25,-1.75){w1}   \nput{180}{w1}{$w_1$}
  \psknob(-1.25,-1.75){w2}   \nput{180}{w2}{$w_2$}
  \psknob(1.25,-1.75){w3}    \nput{0}{w3}{$w_{k-1}$}
  \psknob(3.25,-1.75){w4}    \nput{0}{w4}{$w_k$}
  \psldots(0,0)   \psldots(0,-1.75)
  \pscircleop(0,-3.5){oplus}

  %--- Connections ---
  \psset{style=Arrow}
  \multido{\nn=1+1}{4}{\ncline{x\nn}{w\nn}}
  \ncstar[armA=.75,angleA=-90]{ncdiagg}{w1,w2,w3,w4}{oplus}
  \ncline{oplus}{y}
\end{pspicture}
```

6 Pruebas Pablo


2
1
0
0
1
2
3
4
5
6
$x(t)$
Continuous System
$y(t)$