

psDSP

PsTricks Macros for Digital Signal Processing

Paolo Prandoni

The package `psDSP` is a set of PsTricks macros for plotting signal-processing-related graphs and figures. It defines environments for quick discrete- and continuous-time data plots, frequency plots, pole-zero plots and block diagrams.

Please note that these macros have been written quickly and fundamentally for personal use only so that a) their syntax is often idiosyncratic, b) their flexibility is quite limited in places and c) there *are* bugs. Use at your own risk and caveat emptor.

1 General

There are three predefined plot sizes for rectangular plots, `large`, `medium` and `small`. They are defined as a function of `textwidth` as follows, and they can be changed by using appropriate `\def`'s.

```
1 \def\psDSPlargeWidth{0.9\textwidth}
2 \def\psDSPlargeHeight{\psDSPlargeWidth * \real{0.42}}
3 \def\psDSPmediumWidth{\psDSPlargeWidth * \real{0.7}}
4 \def\psDSPmediumHeight{\psDSPmediumWidth * \real{0.5}}
5 \def\psDSPsmallWidth{\psDSPlargeWidth * \real{0.43}}
6 \def\psDSPsmallHeight{\psDSPsmallWidth * \real{0.7}}
```

For pole-zero plots, which are square, the side is the selected size's height. The default size for all plots is `large`.

2 Plotting Data

The following options are common to all data plots

size=(`small` | `medium` | `large` | `dim`) : plot size; if an explicit dimension `dim` is specified, it is taken as the width of the plot while height is 0.4 times the width

width= `dim1`, **height**= `dim2` : set plot's custom size

dx=`n`, **dy**=`m` : tick increments on X and Y axis

labelx=[`true` | `false`]: show labels on X axis (default true);

labely=[`true` | `false`]: show labels on Y axis (default true);

labelnudge=`dim` : move labels by `dim` with respect to the ticks (useful if stems cross the labels)

2.1 Discrete-Time Plots

To set up a discrete-time plot environment use

```
\begin{psDTplot}[options] {Nmin}{Nmax}{Ymin}{Ymax}
...
\end{psDTplot}
```

This defines a discrete-time plot with the time axis extending from N_{\min} to N_{\max} and with the y -axis spanning the $[Y_{\min}, Y_{\max}]$ interval. The time axis is extended by 2 units both to the left and to the right. To change this use the option **sidegap**= M .

Once the plot is set up you can use the following drawing commands for stem (or “lollipop”) plots; available options in the commands are all standards PsTricks options plus other specialized options when applicable.

To plot a single point at time n with value Y use

```
\psDTplotPoint[options] {N}{Y}
```

To plot a list of space separated time-value pairs use

```
\psDTplotData[options] {data}
```

To plot a discrete-time signal defined in terms of PostScript primitives use

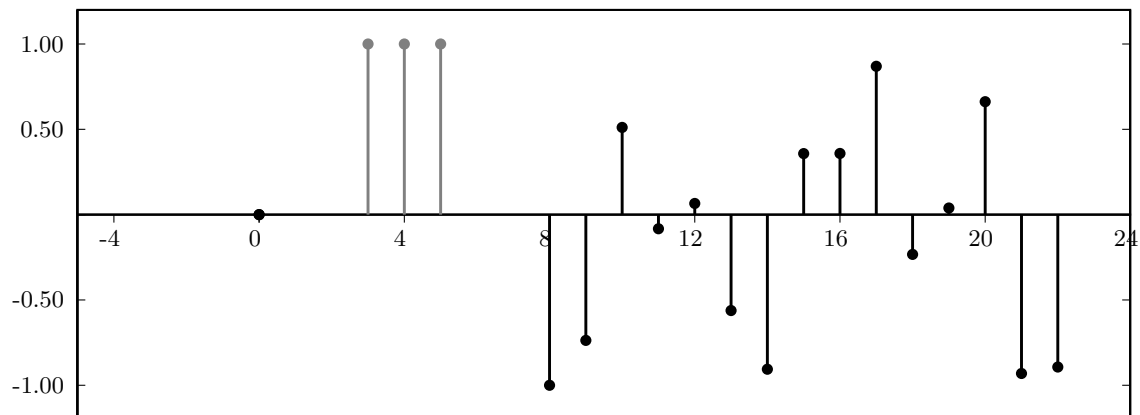
```
\psDTplotSignal[options] {PS code}
```

The PostScript code must use x as the independent variable; the psDTplot environment defines a range for x from N_{\min} to N_{\max} ; this can be changed in with the options **nmin**= n and/or **nmax**= m .

For example:

```
1 \begin{psDTplot}{-3}{22}{-1.2}{1.2}
2   \psDTplotSignal[nmin=8]{rand 2147483647 div 0.5 sub 2 mul}
3   \psDTplotData[linecolor=gray]{3 1 4 1 5 1}
4   \psDTplotPoint{0}{0}
5 \end{psDTplot}}
```

produces the following plot:



You can plot a continuous-time curve over a discrete-time plot by using the command

`\psDTplotFunction[options]{PS code}`

again, the PostScript code must use x as the independent variable while the range is the same as for `\psDTplotSignal`. For a continuous-time-like smooth interpolation of pre-computed data points, use

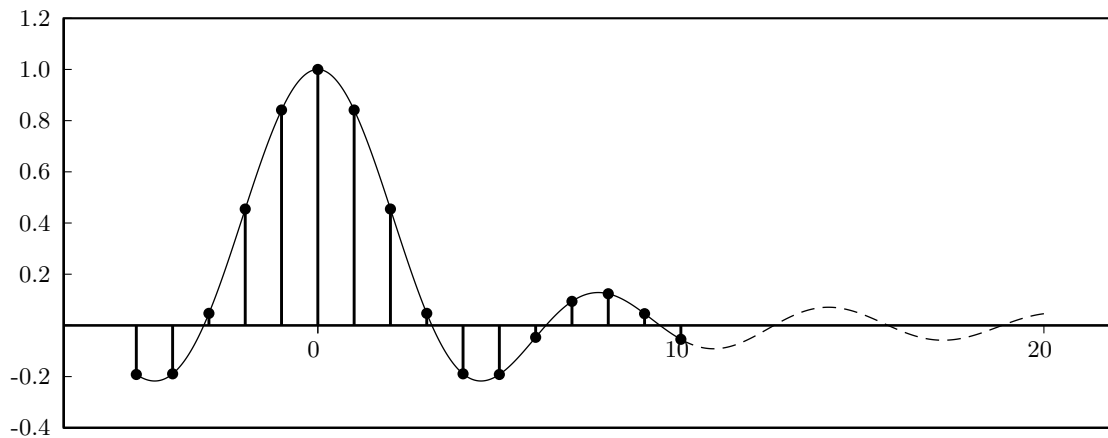
`\psDTplotFile[options]{filename}`

where `filename` indicates a text file containing the data points as a space-separated list of abscissae and ordinates.

```

1 \begin{psDTplot}[dx=10,dy=0.2]{-5}{20}{-0.4}{1.2}
2   \def\sincx{x 0 eq {1} {x RadtoDeg sin x div} ifelse}
3   \psDTplotSignal[nmax=10]{\sincx}
4   \psDTplotFunction[linewidth=0.5pt,nmax=10]{\sincx}
5   \psDTplotFunction[linestyle=dashed,linewidth=0.5pt,nmin=10]{\sincx}
6 \end{psDTplot}

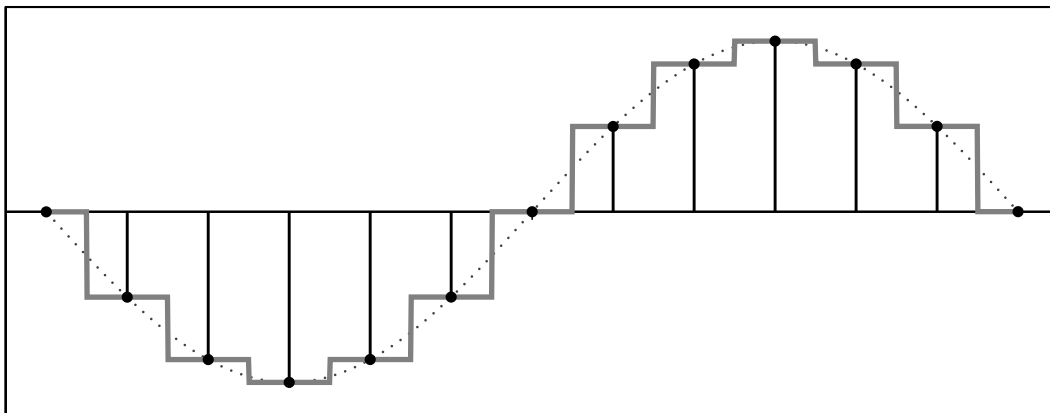
```



```

1 \begin{psDTplot}[dx=1,dy=0.5,sidegap=0.5,labelx=false,labely=false]%
2   {-6}{6}{-1.2}{1.2}
3   \psDTplotFunction[linestyle=dotted,linecolor=darkgray,linewidth=1pt]%
4     {x 0.5235 mul RadtoDeg sin}
5   \psDTplotFunction[linecolor=gray,linewidth=2pt]%
6     {x x 0 gt {-0.5} {0.5} ifelse sub truncate 0.5235 mul RadtoDeg sin}
7   \psDTplotSignal{x 0.5235 mul RadtoDeg sin}
8 \end{psDTplot}

```



2.2 Continuous-Time Plots

To set up a continuous-time plot environment use

```
\begin{psCTplot}[options] {Tmin}{Tmax}{Ymin}{Ymax}
...
\end{psCTplot}
```

This defines a continuous-time plot with the time axis extending from T_{\min} to T_{\max} and with the y -axis spanning the $[Y_{\min}, Y_{\max}]$ interval. The time axis can be extended by a small amount both to the left and to the right using the option **sidegap**= M .

To plot a Dirac delta (symbolized by a vertical arrow) at time T with "amplitude" Y use

```
\psCTplotDelta[options] {T}{Y}
```

To plot a list of space separated time-value pairs (smoothly connected) use

```
\psCTplotData[options] {data}
```

or, if the data is in an external text file,

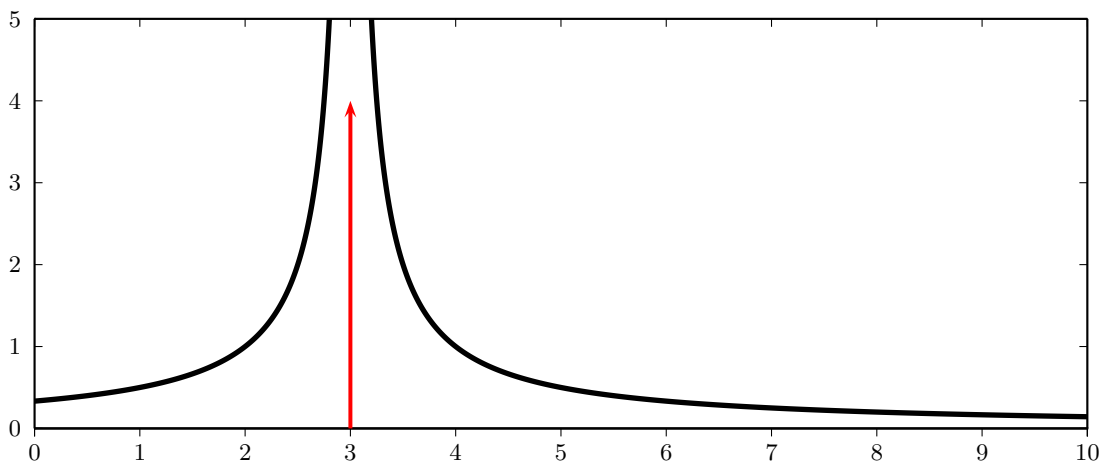
```
\psCTplotFile[options] {filename}
```

To plot a discrete-time signal defined in terms of PostScript primitives use

```
\psCTplotFunction[options] {PS code}
```

The PostScript code must use x as the independent variable; the psDTplot environment defines a range for x from T_{\min} to T_{\max} ; this can be changed in with the options **tmin**= n and/or **tmax**= m .

```
1 \begin{psCTplot}[dy=1]{0}{10}{0}{5}
2   \psclip{\psframe[linewidth=0](0,0)(10,5)}
3   \psCTplotFunction{1 3 x sub div abs}
4   \endpsclip
5   \psCTplotDelta[linecolor=red]{3}{4}
6 \end{psCTplot}
```



2.3 Digital Frequency Plots

To set up a an environment to plot digital spectra use

```
\begin{psDFplot}[options] {Ymin}{Ymax}
...
\end{psDTplot}
```

This defines a plot with the frequency axis extending from $-\pi$ to π and with the y -axis spanning the $[Y_{\min}, Y_{\max}]$ interval. To increase the span of the frequency axis, you can define the number of repetitions of the $[-\pi, \pi]$ interval using **reps**= N (note that N must be odd). You can concurrently truncate the span using **fmin**= f and/or **fmax**= g ; by default **fmin**= $-\text{reps} \times \pi$ and **fmax**= $\text{reps} \times \pi$. Finally, the ticks on the frequency axis are placed at each fraction π/N as specified by the parameter **pifrac**= N .

To plot a Dirac delta (symbolized by a vertical arrow) at frequenc F with "amplitude" Y use

```
\psDFplotDelta[options] {F}{Y}
```

To plot a list of space separated frequency-value pairs (smoothly connected) use

```
\psDFplotData[options] {data}
```

or, if the data is in an external text file,

```
\psDFplotFile[options] {filename}
```

In both cases the frequency values are assumed to be π -normalized, i.e. the maximum digital frequency $\omega = \pi$ corresponds to the value 1.

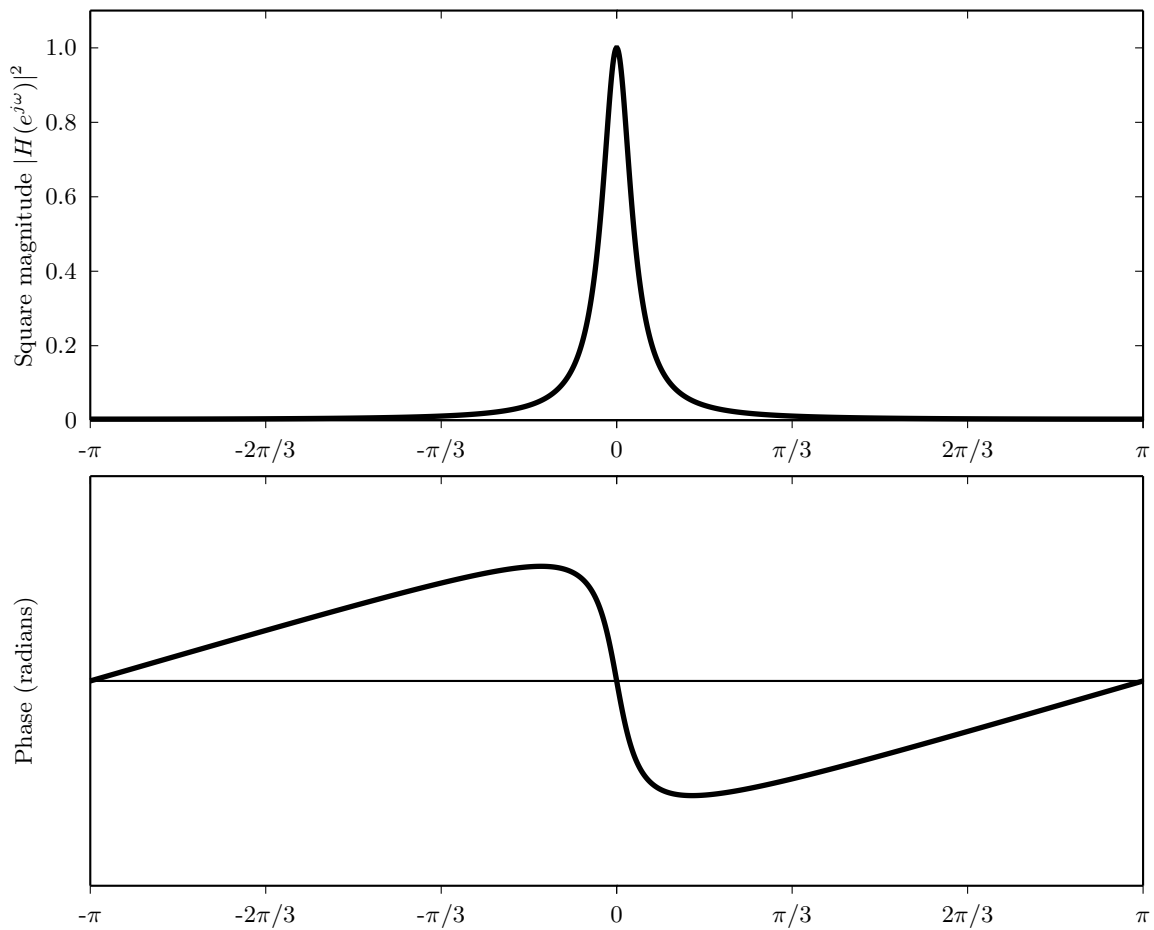
To plot a spectrum defined in terms of PostScript primitives use

```
\psDFplotFunction[options] {PS code}
```

The PostScript code must use x as the independent variable; the psDTplot environment defines a range for x from $-\pi$ to π ; this can be changed in with the options **fmin**= n and/or **fmax**= m .

For example:

```
1 \def\lambda{0.9 }
2 \def\magn{\lambda 1 sub dup mul 1 \lambda \lambda mul add x RadtoDeg cos %
3 2 mul \lambda mul sub div}
4 \def\phase{\lambda x RadtoDeg sin mul -1 mul 1 \lambda x RadtoDeg cos mul %
5 sub atan DegtoRad}
6 \begin{tabular}{c}
7 \begin{psDFplot}[dy=0.2,pifrac=3]{0}{1.1}
8 \psDFplotFunction{\magn }
9 \DFylabel{Square magnitude  $|H(e^{j\omega})|^2$ }
10 \end{psDFplot}
11 \\
12 \begin{psDFplot}[dy=1,pifrac=3,labely=false]{-2}{2}
13 \psDFplotFunction[fmax=0]{\phase }
14 \psDFplotFunction[fmin=0]{-\phase -1 mul}
15 \DFylabel{Phase (radians)}
16 \end{psDFplot}
17 \end{tabular}
```

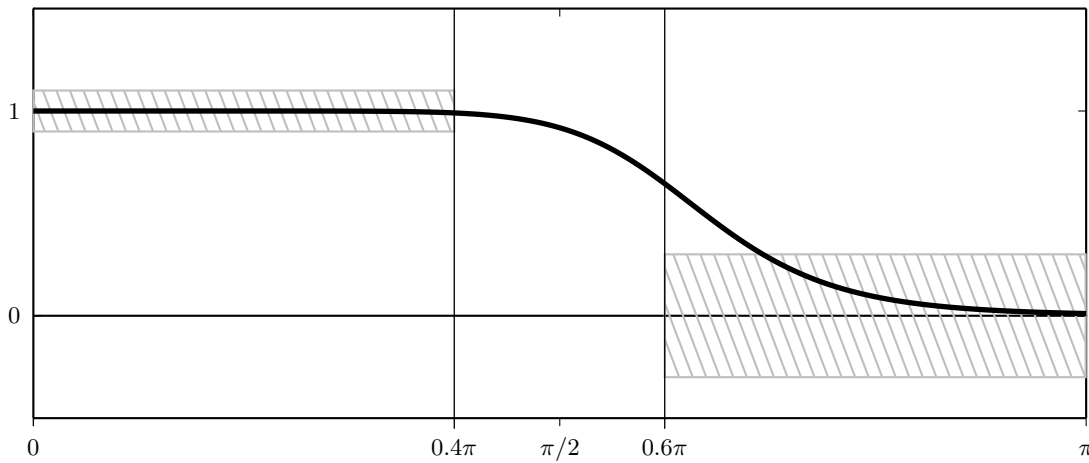


If you need to use standard psTricks primitives in a frequency plot, it is probably more convenient to set the internal x -unit to π so that all frequency values are normalized to one. This can be achieved with the psTricks command `\psset{xunit=3.14}`; to revert the unit to the plot-specific psDSP internal unit, use the command `\psset{xunit=\psDSPunitX}`.

```

1 \begin{psDFplot}[dy=1,pifrac=2,fmin=0]{-0.5}{1.5}
2   \psset{xunit=3.14}
3   \psframe[fillstyle=vlines,%
4     hatchcolor=lightgray,hatchangle=20,%
5     linecolor=lightgray]%
6     (0,1.1)(0.4,0.9)
7   \psframe[fillstyle=vlines,%
8     hatchcolor=lightgray,hatchangle=20,%
9     linecolor=lightgray]%
10    (0.6,0.3)(1,-0.3)
11  \psline[linewidth=0.5pt](0.4,-.5)(0.4,1.5)
12  \psline[linewidth=0.5pt](0.6,-.5)(0.6,1.5)
13  \psticklab[-0.5]{0.4}{\$0.4\pi\$}
14  \psticklab[-0.5]{0.6}{\$0.6\pi\$}
15  \psset{xunit=\psDSPunitX}
16  \psDFplotFunction{x 0.5 mul 10 exp 1 add 1 exch div}
17 \end{psDFplot}

```

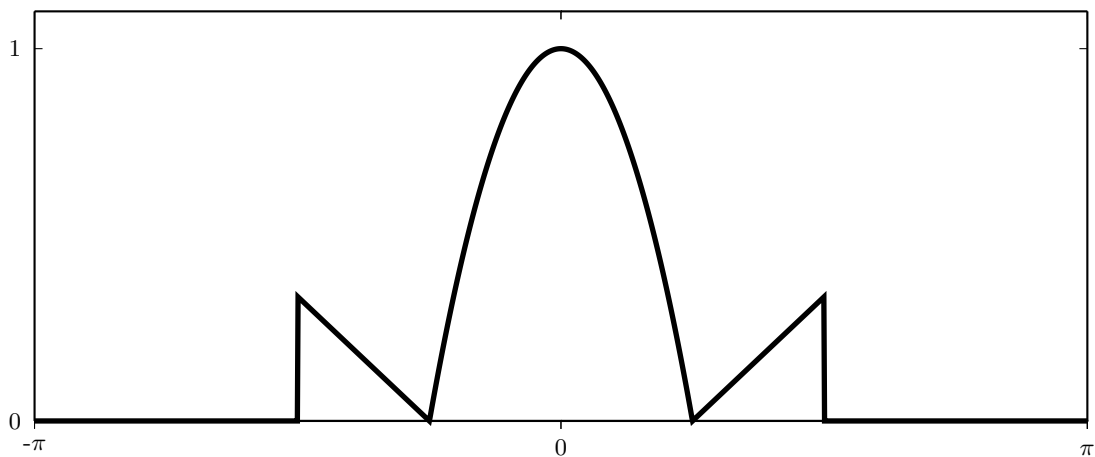


The following example shows how to repeat an arbitrary spectral shape over more than one period. First let's define and plot a spectral shape (we work here in terms of normalized frequency since it is easier in the case of piecewise-regular shapes):

```

1 % triangular shape:
2 \def\triFun{abs 0.25 sub 1 0.25 sub div }
3 % parabolic shape:
4 \def\parFun{abs 0.25 div dup mul 1 exch sub }
5 % composite shape (cutoff at 0.5pi)
6 \def\comFun{
7   dup dup dup dup %
8   -0.5 lt {pop pop pop pop 0} { % zero for x < -0.5
9     0.5 gt {pop pop pop 0 } { % zero for x > 0.5
10    -0.25 lt {pop \triFun } { % triangle between
11     0.25 gt {\triFun } % -.25 and -.5
12     {\parFun} % else parabola
13     ifelse }%
14     ifelse }%
15     ifelse }%
16   ifelse }
17 %
18 \begin{psDFplot}[dy=1]{0}{1.1}
19   \psset{xunit=3.14}
20   \psDFplotFunction[fmin=-1,fmax=1]{x \comFun }
21 \end{psDFplot}

```

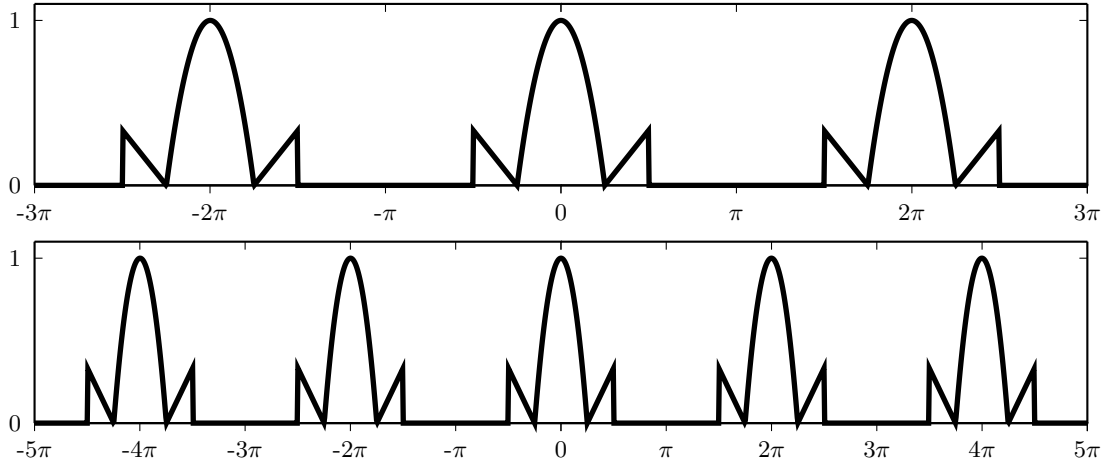


Now we can periodize the function using only standard PostScript commands, as shown in the `\parFun` macro below; plotting multiple periods becomes as simple as changing the **reps** value:

```

1 % 2-pi periodization macro:
2 \def\periodSpec{3.14 sub 6.28 div dup floor sub 2 mul 1 sub }
3 %
4 \begin{tabular}{c}
5   \begin{psDFplot}[dy=1,reps=3,height=3cm]{0}{1.1}
6     \psDFplotFunction{x \periodSpec \comFun }
7   \end{psDFplot}
8   \\
9   \begin{psDFplot}[dy=1,reps=5,height=3cm]{0}{1.1}
10    \psDFplotFunction{x \periodSpec \comFun }
11  \end{psDFplot}
12 \end{tabular}

```



2.4 Analog Frequency Plots

To plot analog spectra, you can either use the environment for continuous-time plots or the environment for digital spectra; in both cases, you should set up the environment with the option `labelx=false` and place your own frequency labels using the macro

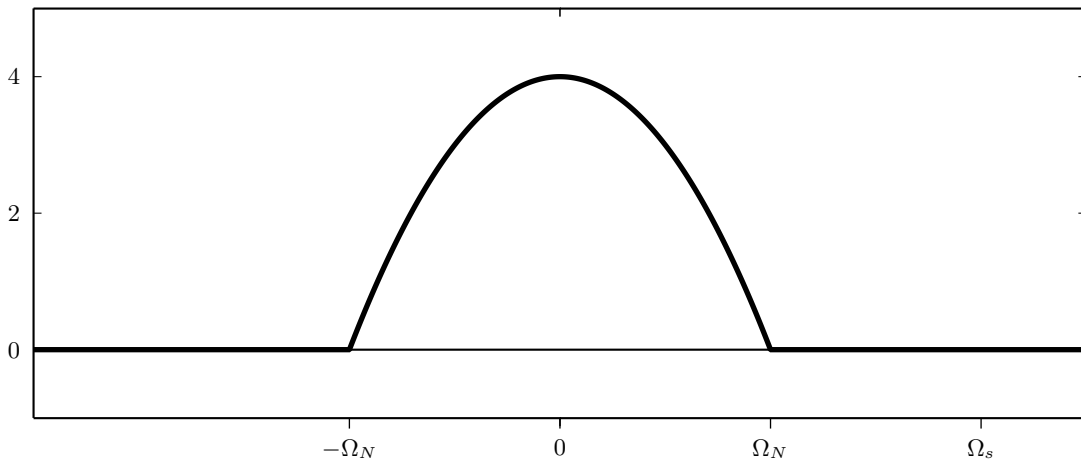
```
\psticklab[Y]{X}{label}
```

which creates an horizontal axis tick at X with the given label; explicitly specify the value for Y if the minimum ordinate is not zero.

```

1 \begin{psCTplot}[dy=2,labelx=false]{-10}{10}{-1}{5}
2   \psCTplotFunction{x abs 4 gt {0} {x abs 2 div dup mul 4 exch sub} ifelse}
3   \selectPlotFont
4   \psticklab[-1]{0}{0$}
5   \psticklab[-1]{4}{$\Omega_N$}
6   \psticklab[-1]{-4}{$-\Omega_N$}
7   \psticklab[-1]{8}{$\Omega_s$}
8 \end{psCTplot}

```

3 Pole-Zero Plots

To set up a pole-zero plot use the environment

```
\begin{psPZplot}[options] {Zmax}
...
\end{psPZplot}
```

This defines a square plot of the complex plane where both axes span the $[-Z_{\max}, Z_{\max}]$ interval. Options for the plot are:

circle=[**true** | **false**]: draws the unit circle (default true);

unit=*label* : use this label at $z = 1 + j \cdot 0$.

To plot a pole or a zero at $z = a + jb$ use

```
\psPZplotPoint[options] {a}{b}
```

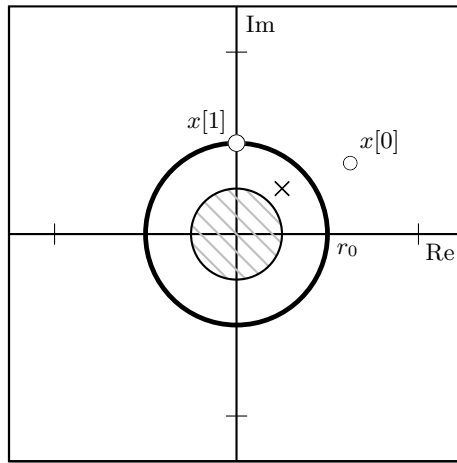
which plots a pole by default; to plot a zero use the option **type=zero**. To associate a label to the point, use the option **label=text**; if **text=false** (which is the default value) no label is printed. Finally, you can specify the position of the label using the option **labelpos=angle**; by default, the angle's value is 45 degrees.

To plot a region of convergence (ROC) with a given radius, use

```
\psPZplotROC[options] {radius}
```

the ROC type is defined by the option **roctype**=[**causal** | **anticausal**]; by default the ROC is causal.

```
1 \begin{psPZplot}[unit=$r_0$]{2.5}
2   \psPZplotROC[roctype=anticausal]{0.5}
3   \psPZplotPoint{0.5}{0.5}
4   \psPZplotPoint[type=zero,label={x[1]},labelpos=135]{0}{1}
5   \psPZplotPoint[type=zero,label={x[0]},labelpos=45]{1.25}{0.78}
6 \end{psPZplot}
```



4 Block Diagrams

Block diagrams rely heavily on psTrick's `psmatrix` environment, for which ample documentation is available. To set up a block diagram use the environment

```
\begin{psdrawBDmatrix}{x_spacing}{y_spacing}
...
\end{psdrawBDmatrix}
```

where `x_spacing` and `y_spacing` define the horizontal and vertical spacing of the blocks in the diagram. In this environment you can use the macros in Table 1, all of which create a node. Node connections can be drawn using the standard primitive `\ncline`.

```
1 \begin{psdrawBDmatrix}{.3}{1}
2 $x[n]$ & & & \BDsplit & \BDdelay & \BDsplit & %
3 \BDdelay & \BDsplit & \BDdelay & \BDsplit & \hspace{3em} & & %
4 \BDdelay
5 %
6 \ \
7 %
8 & & & & & \BDadd & %
9 & \BDadd & & \BDadd & \hspace{3em} & & %
10 & \BDadd & & $y[n]$
11 %
12 \psset{linewidth=1.5pt}
13 \ncline{1,1}{1,3}\ncline{1,3}{1,5}
14 \ncline{1,5}{1,7}\ncline{1,7}{1,9}
15 \ncline{1,9}{1,10}
16 \ncline[linestyle=dotted]{1,10}{1,12}\ncline{1,12}{1,13}
17 \ncline{1,13}{1,14}
18 \ncline{1,4}{2,4}\tlput{$b_0$}
19 \psset{arrows=->}
20 \ncline{2,4}{2,6}\ncline{1,6}{2,6}\tlput{$b_1$}
21 \ncline{2,6}{2,8}\ncline{1,8}{2,8}\tlput{$b_2$}
22 \ncline{2,8}{2,10}\ncline{1,10}{2,10}\tlput{$b_3$}
23 \ncline[-]{2,10}{2,11}
24 \ncline[linestyle=dotted]{-}{2,10}{2,13}
25 \ncline{2,13}{2,14}\ncline{1,14}{2,14}\tlput{$b_{M-1}$}
26 \ncline{2,14}{2,16}
27 \end{psdrawBDmatrix}
```



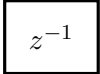
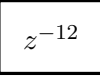
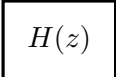



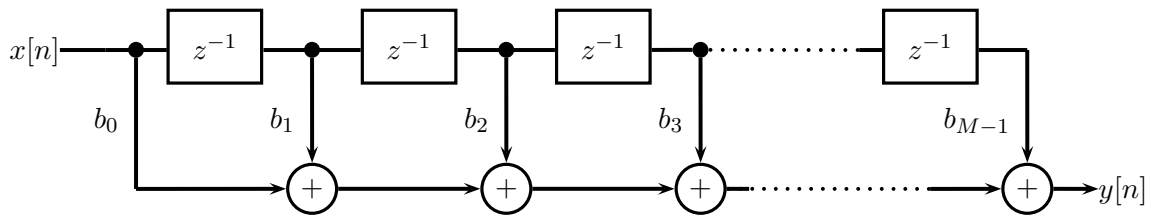
<code>\BDadd</code>	an adder	
<code>\BDmul</code>	a multiplier	
<code>\BDdelay</code>	a delay	
<code>\BDdelayN{N}</code>	a delay by N	
<code>\BDfiltertext</code>	a filter described by <i>text</i>	
<code>\BDsplit</code>	a split (dot)	
<code>\BDupsmp{N}</code>	an upsampler	
<code>\BDdwsmp{N}</code>	an downsampler	

Table 1: Block diagram macros



More complicated building blocks can be drawn using standard PsTricks primitives:

```

1 \begin{psdrawBDmatrix}{1.3}{0.4}
2   $x(t)$
3   &
4   \raisebox{-1.6em}{\psframebox[linewidth=1.5pt]{%
5     \psset{xunit=1em,yunit=1em}%
6     \pspicture(-3,-2)(3,2)%
7       \psline{->}(-2.8,-1)(2.8,-1)%
8       \psline{->}(0,-1.8)(0,1.8)%
9       \psline[linewidth=1.8pt](-1,-1)(-1,0.8)(1,0.8)(1,-1)%
10      \endpspicture}}
11  &
12  &
13  \raisebox{-1.4em}{\psframebox[linewidth=1.5pt]{%
14    \psset{xunit=1em,yunit=1em,linewidth=1.8pt}%
15    \pspicture(-3,-1.8)(2,1.8)%
16      \psline(-2.8,0)(-1.6,0)(1.2,1.4)
17      \psline(1.1,0)(1.8,0)
18      \psarc[linewidth=1pt]{<-}(-1.6,0){2em}{-10}{55}
19    \endpspicture}}
20  &
21  $x[n]$
22  %
23  \psset{linewidth=1.5pt}
24  \ncline{->}{1,1}{1,2}
25  \ncline{1,2}{1,4}{1,4}~{$x_{LP}(t)$}
26  \ncline{->}{1,4}{1,5}
27 \end{psdrawBDmatrix}

```

