10. Connections

At times, we may want to connect two objects (text or graphics) in a document using lines or curves, such as for example
Consider the following row-transformation:
\[
\begin{bmatrix}
a_1 & b_1 & c_1 \\
a_2 & b_2 & c_2 \\
a_3 & b_3 & c_3 \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
a_3 & b_3 & c_3 \\
a_2 & b_2 & c_2 \\
a_1 & b_1 & c_1 \\
\end{bmatrix}
\]

or again like this:

\[
x^2 + y^2 = 5 \quad \text{Thus we find that } x + y = 3 \text{ and using this together with } x^2 + y^2 = 3 \text{ found earlier, we see that } x = 2 \text{ and } y = 1
\]

The package `pst-node` is the one for such jobs. Note that any such connection has three components:

1. the objects to be connected, called `nodes`
2. the type of connections (such as lines or curves), called `node connectors`
3. the labels for the node connectors (called `labels`)

Let’s look at each of these in turn. In all the examples below, we have used the `pst-node` package, by declaring `\usepackage{pst-node}` in the preamble.
10.1. Nodes

We first look at the different kinds of nodes. Since the basic purpose of defining nodes is to connect them in various ways, we also use the simplest of node connections here, called `ncline`, which connects nodes with a single segment of a straight line. The command `rnode` treats the node as a rectangular box. The example below shows the basic usage of this command:

\begin{center}
\color{Blue}
\rnode{1}{\LARGE Node A}
\hspace{2cm}
\rnode{2}{\LARGE Node B}
\ncline[linecolor=Red]{1}{2}
\end{center}

Here the numbers 1 and 2 are the *names* of the nodes, used for referring to them in node connections. We can use any string of letters and numbers as names for nodes.

We can place the nodes wherever we wish using `rput` as shown below:

\begin{center}
\begin{pspicture}(-1,-1)(4,2)
\colgrid
\color{Blue}
\rput(0,0){\color{Blue}
\rnode{1}{\LARGE Node A}}
\rput(3,1){\color{Blue}
\rnode{2}{\LARGE Node B}}
\ncline[linecolor=Red]{->}{1}{2}
\end{pspicture}
\end{center}
How’s the direction of the \texttt{nccline} determined? Well, though the visible part of the connector starts and ends at the boundaries of the nodes, it is actually a part of the line segment joining the centers of the boxes, which are the default reference points. The magnified picture of the last example gives below illustrates this:

The reference point can be changed from the default position, as in the \texttt{rput} command. Look at this example:

\begin{verbatim}
\begin{center}
\begin{pspicture}(-1,-1)(4,2)
colgrid
rput(0,0){\rnode[br]{1}{\color{Blue} \LARGE Node A}}
rput(3,1){\rnode[tl]{2}{\color{Blue} \LARGE Node B}}
\nccline[linecolor=Red]{->}{1}{2}
\end{pspicture}
\end{center}
\end{verbatim}

One trouble with \texttt{rnodes} is that when they are aligned by their baselines, the difference in the heights and depths of the boxes makes the \texttt{nccline} connecting them not quite horizontal, as in this example:
\begin{center}
\color{Blue}
\rnode{1}{\Huge cap}
\hspace{2cm}
\rnode{2}{\Huge hat}
\ncline[linecolor=Red]{->}{1}{2}
\end{center}

The magnified picture below shows why the connector is slanted:

\begin{center}
\color{Blue}
\Rnode{1}{\Huge cap}
\hspace{2cm}
\Rnode{2}{\Huge hat}
\ncline[linecolor=Red]{->}{1}{2}
\end{center}

In such cases, we can use the \Rnode which makes the \ncline parallel to the baseline.

\begin{center}
\color{Blue}
\Rnode{1}{\Huge cap}
\hspace{2cm}
\Rnode{2}{\Huge hat}
\ncline[linecolor=Red]{->}{1}{2}
\end{center}

In \Rnode also, the node is rectangular, but the reference point is with reference to the baseline. By default it is horizontally at the middle of the box and vertically 0.7 \text{ ex} above the base line. These specifications can be altered using the href and vref parameters. The length vref is the height of the reference
point from the baseline; \texttt{href} is not a length but a number, the fraction of the horizontal distance of the reference point from the center of the box by half the length of the box, positive for the right half of the box and negative for the left half. The example below will make this clear.

\begin{center}
\color{Blue}
\Rnode[\texttt{href}=0.6,\%
  \texttt{vref}=5pt]{}{1}{\Huge cap}
\hspace{2cm}
\Rnode[\texttt{href}=-0.3,\%
  \texttt{vref}=10pt]{}{2}{\Huge hat}
\ncline[\texttt{linecolor}=Red]
\{\texttt{->}\}{1}{2}
\end{center}

The larger picture below shows how the parameters are used:

Now we can typeset something like

We can easily change a cat to a dog by cat \longrightarrow cot \longrightarrow dot \longrightarrow dog, changing one letter at a time

with the code
We can easily change a cat to a dog by
\color{Blue}
\psset{vref=0.5tex}
\Rnode{1}{cat}
\quad
\Rnode{2}{cot}
\quad
\Rnode{3}{dot}
\quad
\Rnode{4}{dog}
\psset{linecolor=Red,arrows=->}
ncline{1}{2}
ncline{2}{3}
ncline{3}{4}
, changing one letter at a time

(What happens if we use \rnode instead of \Rnode in the above example?) By putting a \psframebox within an \rnode, we can draw a frame around the node, as shown below:
In the last chapter, we saw the commands for putting text in boxes of various shapes. We have analogous commands for setting up nodes.

\begin{center}
\psset{arrows=->,
  linecolor=Red,%
  fillstyle=solid}
\setlength{\tabcolsep}{0.3cm}
\renewcommand{\arraystretch}{8}
\large\bfseries
\begin{tabular}{cc}
\trinode[fillcolor=Cyan,%
  linecolor=Blue]{}{t}{\color{Red}trinode}
& \circlenode[fillcolor=Yellow,%
  linecolor=Orange]%
  {c}{\color{Blue}circlenode} \\
\dianode[fillcolor=Magenta,%
  linecolor=Red]%
  {d}{\color{Green}dianode}
& \ovalnode[fillcolor=Black,%
  linecolor=Black]%
  {o}{\color{White}ovalnode}
\ncline{t}{d}
\ncline{c}{o}
\ncline{t}{c}
\ncline{d}{o}
\end{tabular}
\end{center}

Again, any of these can be used within an \rput to place them wherever we wish. In the case of circular nodes, there is a single command \cnodeput which combines the actions of \rput and \cirlenode, as shown below:
There are also some node-making commands which just draw rectangles or circles as nodes, without enclosing anything. For example, the \fnode command by default draws a square node 10 point wide with its center at (0,0). The dimensions of the rectangle can be specified through the \framesize parameter and the center can be specified through coordinates. Look at the example below:
\begin{center}
\begin{pspicture}(-1,-1)(6,4)
    \colgrid
    \psset{linecolor=Red}
    \fnode{A}
    \fnode[framesize=2cm 1cm]{(4,1)}{B}
    \fnode[framesize=1cm,fillstyle=solid,fillcolor=Yellow]{(3,3)}{C}
    \psset{linecolor=Blue,arrows=->}
    \ncline{A}{B}
    \ncline{B}{C}
    \ncline{C}{A}
\end{pspicture}
\end{center}

Note that though the \texttt{framesize} parameter accepts two numbers (for the width and the height of the box), to get a square we need specify the width only once, as in the second \texttt{fnode} of the example.

The \texttt{cnode} command draws a circular node of specified center and radius.

\begin{center}
\begin{pspicture}(-1,-1)(5,3)
    \colgrid
    \psset{linecolor=OliveGreen}
    \cnode{0.5}{A}
    \cnode[fillstyle=solid,fillcolor=GreenYellow]{(4,2)}{B}
    \ncline[\textcolor{Green}]{->}{A}{B}
\end{pspicture}
\end{center}
Note that the default center is (0,0), but the radius must be specified. There is also a \Cnode command which by default draws a circular node of radius 2 point, centered at (0,0). The radius can be changed by the radius parameter.

\begin{center}
\begin{pspicture}(-1,-1)(5,3)
\colgrid
\psset{linecolor=OliveGreen}
\Cnode{A}
\Cnode[radius=0.5cm, fillstyle=solid, fillcolor=GreenYellow](4,2){B}
\ncline[linecolor=Green]{->}{A}{B}
\end{pspicture}
\end{center}

Two other commands of his type are the \dotnode command which puts a \psdot at a specified position as a node, and the \pnode command, which does not draw anything but allows treating a specified point as a node:
The \texttt{\pnode} is also useful in typesetting things like this:

\begin{center}
\begin{pspicture}(0,0)(4,4)
\psset{linecolor=Green}
\pnode(1,1){a}
\dotnode[dotstyle=asterisk, dotsize=10pt 10, dotangle=-15, linecolor=Magenta](3,3){b}
\pnode(1.8,1.8){c}
\dotnode[dotstyle=diamond*, dotsize=5pt 5, dotangle=100, linecolor=SpringGreen](3,2){d}
\ncline{a}{b}\ncline{c}{d}
\end{pspicture}
\end{center}

A palindrome is a word or phrase reading the same in reverse, such as civic and nurses run
10.2. Node connectors

We have so far seen only one way of connecting nodes, namely \cline{2}. Now we will see the other node connectors. But before that let’s say something about the general parameters that control all the connectors.

The parameter nodesep is the gap the ends of the connectors leave from the boundary of the node. Its default value is 0 pt, so that the ends of the connectors touch the boundary of the nodes.

\begin{center}
\color{Blue}\rnode{1}{\LARGE Node A} \hspace{2cm} \rnode{2}{\LARGE Node B} \\
\ncline[nodesep=10pt, linecolor=Red]{->}{1}{2}
\end{center}

We can control the gaps with the two nodes separately using the parameters, nodesepA (for the starting node) and nodesepB for the ending node.

\begin{center}
\color{Blue}\rnode{1}{\LARGE Node A} \hspace{2cm} \rnode{2}{\LARGE Node B} \\
\ncline[nodesepA=5pt, nodesepB=20pt, linecolor=Red]{->}{1}{2}
\end{center}

The parameter offset (default value 0 pt) shifts the connection points: for a horizontal connector from left to right, it is upward for positive values and downward for negative values.
\begin{center}
\color{Blue}
\rnode{1}{\LARGE Node A}
\hspace{2cm}
\rnode{2}{\LARGE Node B}
\ncline[linecolor=Red]{->}{1}{2}
\ncline[linecolor=Green,\%
          offset=6pt]{->}{1}{2}
\ncline[linecolor=Cyan,\%
          offset=-6pt]{->}{1}{2}
\end{center}

Node A\nodarrow{}\nodarrow{}Node B

For horizontal connectors from right to left, the shifts go the other way:

\begin{center}
\color{Blue}
\rnode{1}{\LARGE Node B}
\hspace{2cm}
\rnode{2}{\LARGE Node A}
\ncline[linecolor=Red]{->}{2}{1}
\ncline[linecolor=Green,\%
          offset=6pt]{->}{2}{1}
\ncline[linecolor=Cyan,\%
          offset=-6pt]{->}{2}{1}
\end{center}

Node B\nodarrow{}\nodarrow{}Node A

The shifts for slanted connectors is relative to a frame of reference in which
the connector is from left to right:
Again, we can have different offsets for the two nodes by using `offsetA` and `offsetB`:

```latex
\begin{center}
\color{Blue}
\rnode[r]{1}{\LARGE Node A}
\hspace{2cm}
\raisebox{1cm}{{\rnode[l]{2}{\LARGE Node B}}}
\ncline[linecolor=Red]{}{}{1}{2}
\ncline[linecolor=Green, offsetA=1pt, offsetB=6pt]{}{}{1}{2}
\ncline[linecolor=Cyan, offsetA=-1pt, offsetB=-6pt]{}{}{1}{2}
\end{center}
```

We now look at the various node-connectors. We first consider those connectors consisting of two or more line segments. The simplest of these is the \texttt{ncdiag} which draws an \textit{arm} of default length 10 points from each node and then draws a line segment joining these arms:
The lengths of the arms can be specified using the `armA` and `armB` parameters and the angles of the arms with the horizontal can be specified using the `angleA` and `angleB` parameters. This is illustrated in the example below:

```latex
\begin{center}
\psset{nodesep=3pt,%
linecolor=Red}
\color{Blue}
\rnode{1}{\LARGE Node A} %
\hspace{1cm} %
\raisebox{2cm}{% %
\rnode{2}{\LARGE Node B}} %
\ncdiag{1}{2} %
\end{center}
```

If both arms are to be the same length, we can specify the length using the `arm` parameter. Similarly, the `angle` parameter can be used, if the arms are to make the same angle with the horizontal. By carefully calculating the lengths of the arms, we can get nice connections as in the next example. (Note that here we use the `calc` package to compute the length of the arm).
\begin{center}
\newlength{\boxht}
\settowidth{\boxht}{\LARGE Node B}
\newlength{\armlen}
\setlength{\armlen}{(2cm-\boxht-6pt)/2}
\psset{nodesep=3pt,linestyle=red}
\color{blue}
\rnode{1}{\LARGE Node A}
\hspace{1cm}
\raisebox{2cm}{\rnode{2}{\LARGE Node B}}
\ncdiag[angleA=90,angleB=270,arm=\armlen]{1}{2}
\end{center}

The corners can be rounded using the \texttt{linearc} parameter.
A similar connector is `ncdiagg` which draws only an arm from the initial node and then joins it directly to the final node.
In \texttt{ncdiagg}, we can control only the length and slant of the initial arm and we specify them using \texttt{arm} and \texttt{angle} (or \texttt{armA} and \texttt{angleA}). The example below shows the difference between \texttt{ncdiagg} and \texttt{ncdiag} with \texttt{armB=0}.

\begin{center}
\begin{pspicture}(-1,-0.5)(4,2.5)
\Cnode(4,0){A1}
\Cnode(4,2){A2}
\Cnode(0,1){B}
\psset{fillstyle=none,linecolor=OliveGreen}
\ncdiagg[arm=2cm,angle=180]{A1}{B}
\ncdiagg[arm=2cm,angle=180]{A2}{B}
\end{pspicture}
\end{center}

The next in the list is the \texttt{ncbar} connector which draws parallel arms of lengths \texttt{armA} and \texttt{armB} (of default length 10 points) and both inclined at \texttt{angle} (or \texttt{angleA}) with the horizontal (default 0) and then extends one of the arms till it meets the perpendicular from the end of the other arm.
\begin{center}
\psset{linecolor=Red}
\color{Blue}
\rnode{1}{\LARGE Node A}\
\hspace{1cm}\
\raisebox{2cm}{}\
\rnode{2}{\LARGE Node B}\
\ncbar[angle=20,\]
armA=3cm,\%
armB=2cm\}{1}{2} \\
\end{center}

The picture below will make the scheme of drawing clear:

\begin{center}
\psset{linecolor=Red}
\color{Blue}
\rnode{1}{\LARGE Node A}\
\hspace{1cm}\
\raisebox{2cm}{}\
\rnode{2}{\LARGE Node B}\
\ncbar[angle=20,\]
armA=3cm,\%
armB=2cm\}{1}{2} \\
\end{center}

Node A
Node B

In this example, the arm of the final node was 2 centimetres long as specified, but the arm of the initial node was stretched to meet the perpendicular from the end of the final arm. In the next example, it’s the other way round:
The picture below shows \ncbar with the default settings:

\begin{center}
\psset{linecolor=Red}
\color{Blue}
\rnode{1}{\LARGE Node A}
\hspace{2cm}
\rnode{2}{\LARGE Node B}
\ncbar{1}{2}
\end{center}

The \ncbar comes in handy in situation such as this:

The Gaussian method of finding the sum of numbers from 1 to 100 is given below:

\[1 + 2 + 3 + \cdots + 98 + 99 + 100 = 50 \times 101 = 5050\]

This is produced as follows:
The Gaussian method of finding the sum of numbers from 1 to 100 is given below:
\begin{center}
\textcolor{Blue}{\psset{linecolor=Red}}
\begin{equation*}
\rnode[t]{1}{1} + \rnode[t]{2}{2} + \rnode[t]{3}{3} + \ldots + \rnode[t]{98}{98} + \rnode[t]{99}{99} + \rnode[t]{100}{100}
\ncbar[angle=90,arm=20pt]{->}{1}{100}
\ncbar[angle=90,arm=15pt]{->}{2}{99}
\ncbar[angle=90,arm=10pt]{->}{3}{98}
= 50 \times 101 = 5050
\end{equation*}
\end{center}

The connector $\textsc{ncangle}$ is like $\textsc{ncbar}$ in that it also draws arms from the initial and final nodes; it differs from $\textsc{ncbar}$ on two counts:

- The angle of the initial and final arms (with the horizontal) may be different. (In $\textsc{ncbar}$ they are equal.)

- Once the arms are drawn, the length of the initial arm is adjusted so as to meet the perpendicular from the end of the final arm. (In $\textsc{ncbar}$ either arm may be extended depending on the context.)

Look at the example below:
\begin{center}
\psset{nodesep=3pt, linecolor=Red}
\color{Blue}
\rnode{1}{\LARGE Node A}%
\hspace{1cm}%
\raisebox{2cm}{% 
\rnode{2}{\LARGE Node B}}
\ncangle[angleA=20, %
angleB=10, %
arm=2cm]{1}{2}
\end{center}

By suitable choice of angles, we can use \texttt{\textbackslash ncangle} to produce a right angle (with just two line segments) connecting two nodes.
The next one in this class of connectors is the `\ncangles`, which connects two nodes like this: first, the initial and final arms are drawn in specified lengths; then a line is drawn from the end of the initial arm making an angle $90^\circ + 2 \times \text{angle A}$ with it and extended till it meets the perpendicular from the end of the other arm. Thus the connector, in general, consists of four line segments—the two arms from the nodes which are connected by a right angle. (The PSTricks User’s Guide says the right angle joining the arms meets the initial arm at a right angle, but it is not generally true),
The pictures below show the scheme of drawing `\ncangles`
By choosing angles appropriately, we can get nice connections as follows:

\begin{align*}
\text{angleA} &= 20^\circ \\
\text{angleB} &= 10^\circ \\
\text{armA} &= 2 \text{ cm} \\
\text{armB} &= 3 \text{ cm} \\
\end{align*}

\[90 + (2 \times 20) = 130^\circ\]
The connector \texttt{ncloop} brings in another twist to \texttt{ncangles}. As with \texttt{ncangles}, this also starts with the initial and final arms of specified lengths and then a line making angle \(90^\circ + 2 \times \text{angle A}\) with the initial arm. The length of this line is specified by the parameter \texttt{ncloop} (default value 1 cm). Then a line perpendicular to this is drawn and extended till it meets the perpendicular from the end of the other arm.
The drawing scheme for `\ncloop` is shown below:

This connector is useful in producing pictures like this:
Another use of `\ncloop` is in connecting a node with itself. (And that’s why the “loop” in the name.)

\begin{center}
\color{Blue}
\psset{nodesep=3pt, linecolor=Red}
\rnode{a}{\LARGE Node A}
\ncloop[angleA=180, angleB=180, linearc=0.5]{a}{a}
\end{center}

So far we’ve been dealing with connectors made up of line segments. Now let’s look at the curvy connectors. First, the `\ncurve` connector. This draws a Bézier curve joining the two nodes. As explained in Section 4 of Chapter 4, we need four points to draw a Bézier curve. In `\ncurve`, the first and the last points are the points where the curve connect to the nodes, determined by such parameters as `nodesep`, `offset`, `angleA`, `angleB`. The positions of the two
intermediate points are determined by the values of the parameters \texttt{angleA}, \texttt{angleB} and two other parameters \texttt{ncurvA} and \texttt{ncurvB}. The first intermediate point of \texttt{nccurve} is at a distance equal to half the product of \texttt{ncurvA} and the distance between the end-points of the curve from the initial point, in the direction of \texttt{angleA}; the second intermediate point is at a distance equal to half the product of \texttt{ncurvB} and the distance between the nodes from the terminal point, in the direction of \texttt{angleB}. Look at this example:

\begin{verbatim}
\begin{center}
\color{Blue}
\begin{pspicture}(0,0)(6,4)
\colgrid
\rput[t](1,1){%
 \rnode{1}{\LARGE Node A}}
\rput[l](4,3){%
 \rnode{2}{\LARGE Node B}}
\nccurve[angleA=90,\
 ncurvA=0.4,\n \angleB=180,\n ncurvB=1,\n linecolor=Red]{1}{2}
\end{pspicture}\[2cm]
\begin{pspicture}(0,0)(6,4)
\colgrid
\rput[t](1,1){\LARGE Node A}
\rput[l](4,3){\LARGE Node B}
\psdots(1,1)(4,3)
\psbezier[linecolor=Red,5%
 showpoints=true]
 (1,1)(1,1.7211)(2.2972,3)(4,3)
\end{pspicture}
\end{center}
\end{verbatim}

In this example, the distance between the nodes is $\sqrt{(4-1)^2 + (3-1)^2} = \sqrt{13}$. Since \texttt{ncurvA} is set to be 0.4, the first intermediate point is $\frac{1}{2} \times 0.4 \times \sqrt{13} \approx$
0.7211 cm away from the connection point of the first node and since \texttt{angleA} is set to be 90, the connection point is (1, 1); so the intermediate point is 0.7211 cm upward of this point, which means it is (1.7211, 1). Similar computations show that the second intermediate point is $\frac{1}{2} \times 1 \times \sqrt{13} \approx 1.8028$ cm to the left of the connection point (4, 3) of the second node and so is at (2.2972, 3).

It may be noted that the default value of \texttt{ncurvA} as well as \texttt{ncurvB} is 0.67. Also, equal values to both these parameters can be set by simply specifying \texttt{ncurv}.

Now we can describe how the second example at the beginning of this chapter is produced: the code is as below:

\begin{verbatim}
\noindent\makebox[0cm][r]{\rnode[t]{corr}{\color{Blue}$x^2+y^2=5$}}

Thus we find that $x+y=3$ and using this together with
\ovalnode[linecolor=Red,boxsep=false]{err}{$x^2+y^2=3$}
found earlier, we see that $x=2$ and $y=1$
\nccurve[linecolor=Red,angleA=175,angleB=15]{->}{err}{corr}
\end{verbatim}

Another curvy connector is \texttt{ncarc} which again draws a Bézier curve, but the intermediate points are computed in a different way. The \textit{distances} of the intermediate points from the end-points are as in \texttt{ncurve}, but the \textit{directions} are measured from the \textit{line joining the end points} and are specified by the parameters \texttt{arcangleA} and \texttt{acangleB}(default value 8). Look at the example below:
\begin{center}
\color{Blue}
\begin{pspicture}(0,0)(6,4)
\colgrid
\rput[t]{1,1}{%\Large Node A}
\rput[b]{4,3}{%\Large Node B}
\ncarc[arcangleA=90, %
arcangleB=270, %
ncurvA=0.4, %
ncurvB=1, %
linecolor=Red]{1}{2}
\psline[linecolor=Cyan](1,1)(4,3)
\end{pspicture}
\begin{pspicture}(0,0)(6,4)
\colgrid
\rput[t]{1,1}{\Large Node A}
\rput[b]{4,3}{\Large Node B}
\psbezier[linecolor=Red, %
showpoints=true](1,1)(0.6,1.6)(5,1.5)(4,3)
\psline[linecolor=Cyan](1,1)(4,3)
\end{pspicture}
\end{center}

For the default values of arcangle and ncurv the \texttt{ncarc} connector approximates an arc of a circle:
Note that as before, equal values for \texttt{arcangleA} and \texttt{arcangleB} can be set by the single parameter \texttt{arcangle}.

The last of this set of connectors is \texttt{ncircle} which draws an arc of a circle from a node to itself, with a specified radius and which, if completed would pass through the reference point of the node at an angle specified by \texttt{angle} (or \texttt{angleA}).

\begin{center}
\begin{pspicture}(0,0)(6,4)
\psset{nodesep=3pt,offset=2pt}
\colgrid
\rput[t]{1,1}{% \LARGE Node A}
\rput[b]{4,3}{% \LARGE Node B}
\ncarc[\textcolor{Red}]{->}{1}{2}
\ncarc[\textcolor{Red}]{->}{2}{1}
\end{pspicture}
\end{center}

\begin{center}
\color{Blue}
\rnode{A}{\LARGE Node}
\ncircle[\textcolor{Red},\texttt{angle=90},\texttt{angle=270}]{A}{1cm}
\end{center}
Finally we have two connectors which draw boxes around the nodes. The first is `\ncbox` which draws a rectangular box around the nodes, whose size is determined by `nodesep` and two parameters `boxheight` and `boxdepth`. Look at this example

\begin{center}
\color{Blue}
\psset{linecolor=Red}
\rnode{1}{\LARGE Node A}
\hspace{2cm}
\raisebox{2cm}{\rnode{2}{\LARGE Node B}}
\ncbox[nodesep=0.2cm, %
  boxheight=1cm, %
  boxdepth=1.5cm]{1}{2}
\end{center}

The box is drawn by first drawing two `\ncline`'s with offsets `boxheight` and `−boxdepth`, and then completing the box (taking `nodesep` into account). This is illustrated in the picture below.
\begin{center}
\color{Blue}
\psset{linecolor=Red}
\rnode{1}{\LARGE Node A}
\hspace{2cm}
\raisebox{2cm}{\rnode{2}{\LARGE Node B}}
\ncbox[nodesep=0.2cm, boxsize=1cm]{1}{2}
\end{center}

For setting equal values to boxheight and boxdepth, the parameter boxsize can be used. In this case, the box is symmetrical about the line joining the nodes.

\begin{center}
\color{Blue}
\psset{linecolor=Red}
\rnode{1}{\LARGE Node A}
\hspace{2cm}
\raisebox{2cm}{\rnode{2}{\LARGE Node B}}
\ncbox[nodesep=0.2cm, boxsize=1cm]{1}{2}
\end{center}

The connector \texttt{ncarcbox} is similar, using \texttt{narc}c’s instead of \texttt{ncline}’s to make a box:
The various node connectors can be used with \pnode to draw pictures also. For this purpose, all node connectors, except \nccircle have variants named with pc (for point-connection) in the place of nc. Thus

\pcline(1,2)(3,4)

is equivalent to

\pnode(1,2){a}\pnode(3,4){b}\ncline{a}{b}

An example using \pcarc is given below

\begin{pspicture}(-2,-2)(2,2)
\psset{linecolor=Magenta,fillstyle=solid,fillcolor=Lavender}
\petalput{0}\petalput{60}\petalput{120}\petalput{180}\petalput{240}\petalput{300}
\end{pspicture}

\end{center}
(This could be done more efficiently using \texttt{multido} package, but that’s another story—well, another chapter)
10.3. Labels

Now let’s see how we attach labels to nodes and connectors. First we look at labels for connectors. There are two sets of commands for this, which differ in the way in which they compute the positions of label placement. The first set consists of three commands \naput, \nbput and \ncput. For connectors with a single segment (\ncline) or a single piece of curve (\nccurv and \ncarc) from left to right, these place the labels above, below and on the connector, with the the center of the label-text at the middle of the connector.

\begin{pspicture}(0,-1)(3,4)
\psset{linecolor=Red,%
labelsep=10pt}
\pcline{->}(0,3)(3,3)
\naput{\color{Blue} naput}
\nbput{\color{Blue} nbput}
\ncput*{\color{Blue} ncput}
\pcline{->}(3,0)(0,0)
\naput{\color{Blue} naput}
\nbput{\color{Blue} nbput}
\ncput*{\color{Blue} ncput}
\end{pspicture}

Note that in this example, the positions “above” and “below” are reversed for a connector from right to left. (Compare the effect of positive and negative values for offset.) Note also the use of the starred form \ncput* of the \ncput command in this example. As in the case of \uput, we can use the labelsep parameter to control the distance between the label and the connector.

There are various parameters which affect the positioning of the labels. The parameter nrot can be used to rotate the label.
In this example, we rotated the “hypotenuse” label through 60°, since we know that the line itself is inclined at this angle. What if we don’t know this angle? Instead of specifying the angle of rotation in absolute terms (as in \texttt{nrot=60°} in the above example), we can also specify it relative to a frame of reference for which the connector is from left to right, using the \texttt{nrot=} syntax. Thus, \texttt{nrot=} \texttt{0} makes the label parallel to the connector, as in the next example. This saves us the trouble of computing the slope of the line.
As another application of this, see how we can quickly draw the perpendicular bisector of a line.

The parameter \texttt{npos} controls the position of the label with respect to the \textit{length} of the connector. Its value is the distance from the beginning of the connector to the point on on it corresponding to the position of the label, given as a fraction of the total length of the connector. Thus its value must be between 0 and 1 and the default value is 0.5, which corresponds to the midpoint of the connector. By tweaking this parameter, we can divide the connector in any ratio we want, as shown below.
Another useful device in drawing pictures is that we can specify the coordinates of the reference point of a node by simply specifying its name within parenthesis, under `\SpecialCoor`. Look at the code for a portion of a picture (slightly changed), given earlier in the chapter.

\begin{center}
\psset{linecolor=Red}
\begin{pspicture}(1,0.5)(3,4.5)
\pcline{*-*}(1,1)(3,4)
\ncput[npos=0.2]{
\psdots[linecolor=Blue](0,0)}
\nbput[npos=0.1]{
\color{Blue} $\frac{1}{5}$}
\nbput[npos=0.6]{
\color{Blue} $\frac{4}{5}$}
\end{pspicture}
\end{center}
Note how in the last line of the code, we draw the reference point of the Rnode \(a\) (set in line 8) by simply specifying \(\text{psdots}(a)\). Again, in the last but one line, we draw the baseline of the enclosed text by \(\text{psline}(ab)(ae)\), where \(\text{pnode \ ab}\) and \(\text{pnode \ ae}\) are set at the beginning and end of this text in lines 7 and 9. We use this device in line 16 also, to draw the perpendicular from the reference point to the baseline. The specification \((a|0,0)\) is for the point with \(x\)-co-ordinate that of node \(a\) and \(y\) co-ordinate that of the points with coordinates \((0,0)\) (see Chapter 5, More on Coordinates).

Let’s now look at connectors with multiple segments. The only essential difference is in the \(\text{npos}\) parameter; the value of this parameter can be given as a number with an integer part specifying the segment to which the label is to be attached (starting with 0) and a decimal part specifying the position of the label with respect to the length of \(this\) segment.
The default value of npos is half the number of the (maximum) number of segments the connector has. Thus for ncloop which has (a maximum number of) 5 segments, the default value of npos is 2.5 and it corresponds to the mid-point of the third segment of the ncloop.

There are another set of six commands for placing labels, \taput, \tbput, \t1put, \trput, \thput, \tvput. The difference with the n*put commands is that these compute the position of the label with respect to the distance between the reference points of the nodes, instead of the actual length of the node connector. Thus they are useful in aligning the labels horizontally or vertically in mathematical diagrams or trees. (The t in these commands suggest tree. Trees will be discussed in another chapter). The commands
\texttt{\texttt{taput}} and \texttt{\texttt{tbput}} place labels above and below the connector, while \texttt{lput} and \texttt{rput} place them on the left and right (all with reference to a frame in which the segment is from left to right, default at the midpoint of the line joining the reference points of the nodes). Compare the two pictures below:
\newcolumntype{b}{>{\color{Blue}}c}
\setlength{\arraycolsep}{0.9cm}
\renewcommand{\arraystretch}{6}
\psset{linecolor=Red,arrows=->,nodesep=5pt}
\begin{array}{bb}
\Rnode{1}{\dfrac{X}{Z}\left/\dfrac{Y}{Z}\right.} & \Rnode{2}{U} \\
\Rnode{3}{V} & \Rnode{4}{\dfrac{X}{Z}\times\dfrac{Y}{Z}}
\end{array}
\begin{array}{bb}
\Rnode{1}{\dfrac{X}{Z}\left/\dfrac{Y}{Z}\right.} & \Rnode{2}{U} \\
\Rnode{3}{V} & \Rnode{4}{\dfrac{X}{Z}\times\dfrac{Y}{Z}}
\end{array}
\n\begin{array}{bb}
\ncline{1}{2}\naput{f} \\
\ncline{1}{3}\naput{k} \\
\ncline{2}{4}\naput{h} \\
\ncline{3}{4}\naput{g}
\end{array}
\nNote that in the top picture (with the \texttt{\n*put} commands), the labels $f$ and $g$ are not horizontally aligned, nor are the labels $h$ and $k$ vertically aligned.
and these are so in the bottom picture (with the \t*put commands). The \newcolumntype command comes from the array, so that we will have to \usepackage{array} for this example to work.

For slanted lines, the commands \tput and \tput compute the position of the labels with respect to the horizontal distance between the reference points of the nodes, while \taput and \trput use the vertical distance (default at the middle). So, the labels for such connectors can be aligned with those for other horizontal or vertical connectors in the picture. Look at the pictures below:

&indent=0.5\textwidth
\begin{verbatim}
\newcolumntype{b}{>{\color{Blue}}c}
\setlength{\arraycolsep}{1cm}
\renewcommand{\arraystretch}{5}
\psset{linecolor=Red,arrows=->,nodesep=5pt}
\begin{array}{bb}
\Rnode{1}{X} & \Rnode{2}{Y} \\
\Rnode{3}{\dfrac{X}{Z}} & \color{Mahogany}
\ncline{1}{2}\taput{f} \\
\ncline{1}{3}\tlput{p} \\
\ncline{3}{2}\tput{h}
\end{array}\end{verbatim}
\end{itemize}

The commands \thput and \tvput place the labels on the connectors, using the horizontal and vertical distance between the reference points of the nodes.
In all the six `\put` commands, the position of the label can be controlled using the `tpos` parameter just like the `npos` parameter for the `\nput` commands. However, there is no parameter such as `nrot` to rotate the labels for the `\put` commands.

For labeling nodes, we use the `\put` command, which is somewhat like the `\put` command with general syntax `\put [parameters] {dirangle} {name} {stuff}` where `dirangle` is the direction of the label with respect to the reference point, specified by an angle. We can specify the distance of the label from the node using `labelsep` and `offset` and rotate it using `rot`. See this example:

\begin{verbatim}
\scalebox{8}{%
  \psset{linewidth=0.3pt}
  \pnode(0,0){ab}%
  \Rnode{a}{%
    \psframebox[linecolor=Cyan,%
      framesep=0pt,%
      boxsep=false]{{%
        \color{Blue}
        ' \LARGE cap}}%
  \pnode(0,0){ae}%
  \SpecialCoor
  \pcline[linecolor=Melon](ab)(ae)
  \color{Mahogany}
  \put[labelsep=1pt]{l}{ab}{%\scalebox{0.12}{% baseline}}
  \put[rot=-90,offset=-1pt,%
    labelsep=2pt]{u}{ae}{{%
    \scalebox{0.12}{height}}
  \put[rot=-90,offset=1pt,%
    labelsep=0.3pt]{d}{ae}{{%
    \scalebox{0.12}{depth}}
  \put[labelsep=1pt]{u}{a}{%\scalebox{0.12}{width}}
}
\end{verbatim}
10.4. Nodes in a matrix

We have seen a few examples where the nodes are placed on a grid or more technically a matrix. These can be more conveniently done using the psmatrix environment defined in pst-node. Within this environment we position the nodes in rows and columns using \& and \\, as in the array environment. We can refer to the nodes using the syntax \{rownumber, columnnumber\}, so that we are saved the trouble of naming them. What is more, we can use the shorter commands \^ for \taput and _ for \tbput and the commands < for \tlput and > for \trput. Look at an earlier example, redone using this environment:

\begin{psmatrix}
\dfrac{X}{Z}\left/\dfrac{Y}{Z}\right. & U \\
V & \dfrac{X}{Z}\times\dfrac{Y}{Z}
\end{psmatrix}

Incidentally, the shorter forms of the \t*put commands can be used outside the psmatrix environment also, by setting the parameter shortput equal to tablr. Again, the short-cuts \^ and _ can be used for \naput and \nbp put commands also, by setting shortput=nab. The default value of this parameter is none. (It can also be set to tab, but this is only for “trees” and will be explained in a later chapter).
The distance between the rows and columns are controlled by the \rowsep and \colsep parameters (of default value 1.5 centimetres). The shape of the nodes can be changed using the \mnode parameter.

\begin{psmatrix}[mnode=circle,colsep=3cm]
\cline{1,1}{2,2}^a
\cline{2,1}{2,2}^b
\cline{3,1}{2,2}_c
\cline{1,1}{2,1}^d
\cline{2,1}{3,1}^e
\psset{fillstyle=none, arcangle=-60}
\ncarc{1,1}{2,1}^f
\ncarc{2,1}{3,1}^g
\end{psmatrix}

As in this example, such parameter changes can be given in square brackets within the \psmatrix environment. They can also be set using the \psset command. The possible values of the \mnode parameter and the corresponding shapes of the nodes are given in the table below:
The default value of \texttt{mnode} is \texttt{R}, so that nodes in the \texttt{psmatrix} environment are by default \texttt{\textbackslash Rnode}'s.

Note also that the node-shape commands do not affect the “empty” nodes. For that, we can use the \texttt{emnode} parameter.

\begin{verbatim}
\color{Blue} \psset{linecolor=Red} \begin{psmatrix} \[rowsep=0.5cm, colsep=0.5cm, emnode=C\]
  2 & \circ & 6 & 8 & 10 \\
  \& & 6 & 9 & \& & 15 \\
\end{psmatrix}
\end{verbatim}

Besides defining the shapes of all nodes globally in a \texttt{psmatrix}, we can also control each node individually, by setting the \texttt{mnode} value within square brackets before an entry, as in the next example:
Note in particular, the use of \space at the end of line 5 of the above code. This is to insert a space after the \\ so that \LaTeX{} is prevented from treating the [mnode=\text{dia}] in the next line as the optional (space increasing) argument of \\ (and thus signaling a “missing number” error).

We can also give names the nodes in our own fashion using the name parameter and use these in connections instead of the row-column number.

\begin{verbatim}
\color{Blue}
\psset{linecolor=Red}
\begin{psmatrix} [rowsep=1cm,colsep=0.5cm]
[\text{mnode=\text{dia}}] 2 & 4 & [\text{mnode=\text{circle}}] 6 & 8 & 10 & [\text{mnode=\text{circle}}] 12 \\space
[\text{[mnode=\text{dia}}] 3 & [\text{mnode=\text{circle}}] 6 & 9 & [\text{mnode=\text{circle}}] 12 & 15 & 18
\end{psmatrix}
\end{verbatim}

This is useful, for example, when we want to connect the nodes of different \texttt{psmatrix} environments.
The entries in each column in a \texttt{psmatrix} are centrally aligned by default, as can be seen from this example:
\color{Blue} 
\psset{linecolor=Red} 
\begin{psmatrix} 
\scshape Planet & \scshape Diameter(km) \\
Earth & 12756 \\
Mars & 6794 \\
Jupiter & 142984 \\
Saturn & 120536 \\
\end{psmatrix}

The alignment of the entries can be changed by setting the \texttt{mcol} parameter, either globally or locally:

\color{Blue} 
\psset{linecolor=Red} 
\begin{psmatrix} 
\scshape Planet & \scshape Diameter(km) \\
Earth & \texttt{[mcol=r]} 12756 \\
Mars & \texttt{[mcol=r]} 6794 \\
Jupiter & \texttt{[mcol=r]} 142984 \\
Saturn & \texttt{[mcol=r]} 120536 \\
\end{psmatrix}

In this example, instead of typing \texttt{[mcol=r]} before each entry in the second column, we can set this globally for the entire second column, by defining \texttt{pscolhookii} as in the next example.
Here the line \texttt{\def\pscolhookii{\psset{mcol=r}}} executes the command \texttt{\psset{mcol=r}} at the beginning of every entry of the second column. (The \texttt{ii} at the end of \texttt{\pscolhookii} stands for the second column.) Similarly, the line \texttt{\def\psrowhooki{\scshape}} executes the command \texttt{\scshape} at the beginning of each entry in the first row.

An entry in a \texttt{psmatrix} can be made to span several columns, using the \texttt{\psspan} command:

\pdfescapequote{%}
\begin{psmatrix}
\[\rowsep=0.3cm,mcol=l\]
\begin{tabular}{l}
Planet & Diameter(km)\\
Earth & 12756\\
Mars & 6794\\
Jupiter & 142984\\
Saturn & 120536\\
\end{tabular}
\ncbox[linelc=0.3,nodesep=5pt]{4,1}{4,2}
\end{psmatrix}

Here the line \texttt{\def\pscolhookii{\psset{mcol=r}}} executes the command \texttt{\psset{mcol=r}} at the beginning of every entry of the second column. (The \texttt{ii} at the end of \texttt{\pscolhookii} stands for the second column.) Similarly, the line \texttt{\def\psrowhooki{\scshape}} executes the command \texttt{\scshape} at the beginning of each entry in the first row.

An entry in a \texttt{psmatrix} can be made to span several columns, using the \texttt{\psspan} command:

\begin{psmatrix}
\[colsep=3.5cm\]
\begin{tabular}{l}
N & \psspan{2}\\
W & E\\
S & \psspan{2}\\
\end{tabular}
\psspan{5pt,arrows=<->}
\ncline{1,1}{3,1}
\ncline{2,1}{2,2}
\end{psmatrix}

Note that the \texttt{\psspan} command is given at the end of the entry which is to span multiple columns, Also, the argument 2 of \texttt{\psspan{2}} in the above
example gives the number of columns to be spanned.

With this device, the table in the last but one example can be formatted a bit more nicely:

```
\color{Blue}
\psset{linecolor=Red}
\def\pscolhookv{\psset{mcol=r}}
\def\psrowhooki{\psset{mcol=c}\scshape}
\begin{psmatrix} [rowsep=0.3cm,colsep=0.75cm,%
    mcol=1]
    \text{Planet} \psspan{3} & \text{Diameter(km)} \psspan{3} \\
    & \text{Earth} & 12756 \\
    & \text{Mars} & 6794 \\
    & \text{Jupiter} & 142984 \\
    & \text{Saturn} & 120536 \\
    \end{psmatrix}
```

<table>
<thead>
<tr>
<th>P</th>
<th>D</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Earth</td>
<td>12756</td>
<td></td>
</tr>
<tr>
<td>Mars</td>
<td>6794</td>
<td></td>
</tr>
<tr>
<td>Jupiter</td>
<td>142984</td>
<td></td>
</tr>
<tr>
<td>Saturn</td>
<td>120536</td>
<td></td>
</tr>
</tbody>
</table>