6. GRAPHICS

MAPLE can plot functions of one variable, planar curves, functions of two variables, and surfaces in three dimensions. It can also handle parametric plots and animations. The two main plotting functions are `plot` and `plot3d`.

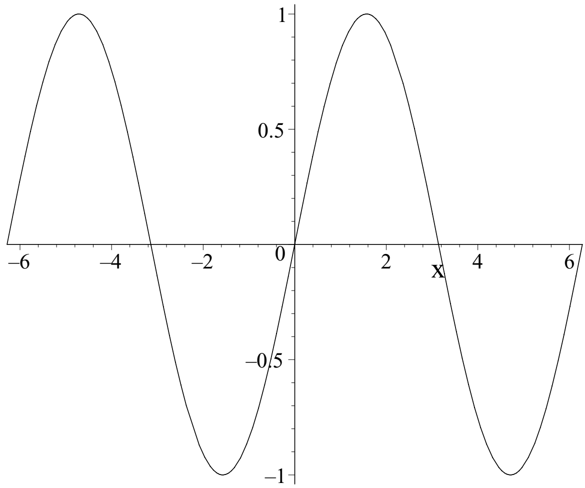


Figure 6.1 MAPLE plot of $y = \sin x$.

6.1 Two-dimensional plotting

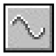
The syntax for plotting an expression (or function) in x is `plot(f(x), x=a..b)`. For example, to plot $\sin(x)$ for $-2\pi \le x \le 2\pi$, we type

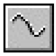
```
>  plot(sin(x),x=-2*Pi..2*Pi);
```

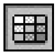
The resulting plot appears in Figure 6.1.

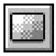
Observe that in MAPLE the plot actually appears in the current document. Click on the MAPLE plot with the left mouse button. A rectangle should now border the plot. You will notice eight dots: one in each corner and one at the midpoint of each side. The dots mark positions for resizing the plot. Move the mouse on the dot in the bottom right corner. A little ⤡ appears. Try stretching the plot display into a different shape. Notice also that the menu bar and the context bar have changed. The menu bar consists of the File, Edit, View, Format, Style, Legend, Axes, Projection, Animation, Export, Window, and Help menus. The context bar has changed completely. There should be a small window containing a pair of coordinates and nine new buttons. Try clicking on each button to see its effect.

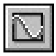
0.53, 0.50	Displays the coordinates of the point under the tracker (i.e., the point clicked).

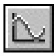
 Render the plot using the usual line style.

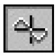
 Render the plot using the usual point style.

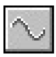
 Render the plot using the polygon patch with gridlines style.

 Render the plot using the polygon patch style.

 Draw the plot axes as an enclosed box.

 Draw the plot axes as an exterior frame.

 Draw the plot axes in traditional form.

 Suppress the drawing of plot axes.

 Use the same scale on both axes.

Now click on the plot with the right mouse button. A context menu should appear:

Copy
Style ▶
Legend ▶
Axes ▶
Projection ▶
Animation ▶
Export As ▶

Click in Style. A submenu should appear:

Line
Point
✓ Patch
Patch w/o grid
Default
Symbol As ▶
Symbol Size...
Line Style ▶
Line Width ▶

Select Point. The resulting plot is just a set of points interpolating the curve.

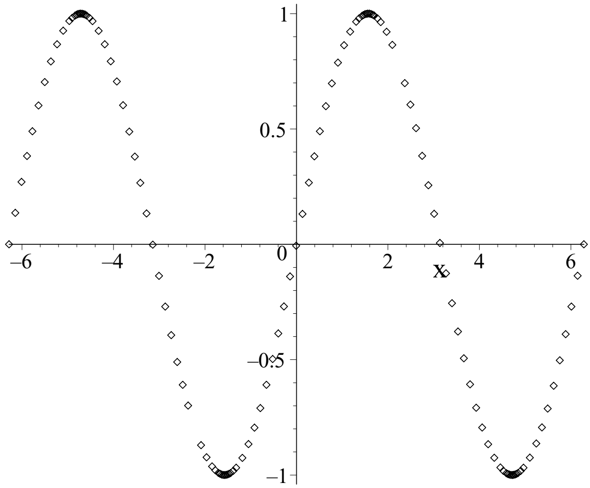


Figure 6.2 MAPLE point-style plot of $y = \sin x$.

Try some of the other selections in the context menu.

6.1.1 Restricting domain and range

Try the plot command `plot(sec(x),x=-Pi.. 2*Pi)`. Notice the "spikes" at $x = -\pi/2$, $\pi/2$, and $3\pi/2$ in your MAPLE plot. These correspond to singularities of $\sec(x)$. We restrict the range to get a more reasonable plot.

```
> plot(sec(x),x=-Pi..2*Pi,y=-5..5);
```

The resulting plot appears in Figure 6.3. Observe the vertical lines in the plot. MAPLE has tried to plot a continuous curve even though the function $\sec x$ has discontinuities at $x = -\pi/2$, $\pi/2$, and $3\pi/2$ in the interval $[-\pi, 2\pi]$. To allow for these discontinuities we can use the `discont` option. Try

```
> plot(sec(x),x=-Pi..2*Pi,y=-5..5,discont=true);
```

So, to plot $y = f(x)$, where $a \le x \le b$, and $c \le y \le d$, in MAPLE we use the command `plot(f(x),x=a..b,y=c..d)`.

6.1.2 Parametric plots

To plot the curve parameterized by

$$x = f(t), \qquad y = g(t), \qquad \text{for } a \le t \le b,$$

we use the command `plot([f(t),g(t),t=a..b])`. The ellipse

$$x^2 + 4y^2 = 1,$$

can be parameterized as

$$x = \cos(t), \quad y = \frac{1}{2}\sin(t), \quad \text{where } 0 \le t \le 2\pi.$$

Try

```
> plot([cos(t),1/2*sin(t),t=0..2*Pi]);
```

This should give you the desired plot.

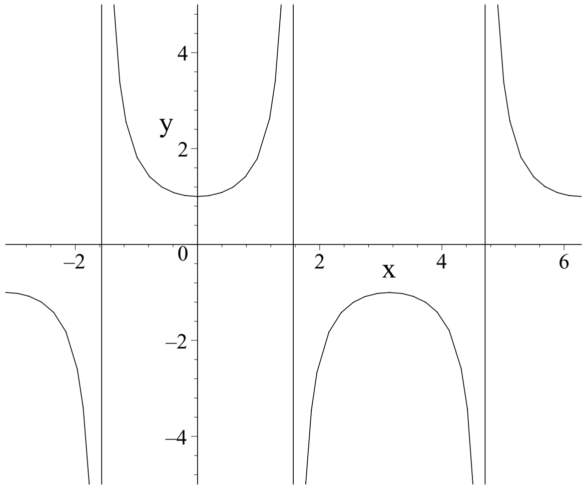


Figure 6.3 MAPLE plot of $y = \sec x$.

6.1.3 Multiple plots

To plot the two functions

$$y = \sqrt{x}, \qquad y = 3\log(x),$$

try

```
> plot([sqrt(x),3*log(x)],x=0..400);
```

The resulting plot is given in Figure 6.4. On the screen, each curve is plotted with a different color. Observe that our plot does not seem to illustrate the expected behavior of the log function near $x = 0$. To get a more accurate plot, we can use the `numpoints` option. Try

```
> plot([sqrt(x),3*log(x)],x=0..400,numpoints=1000);
```

An alternative method for doing multiple plots is to use the `display` function in the *plots* package. Try

```
> with(plots):
> p1:=plot(sqrt(x),x=0..400):
> p2:=plot(3*log(x),x=0..400):
> display(p1,p2);
```

When defining `p1` and `p2`, use a colon unless you want to see all the points MAPLE uses to plot the functions. To see all the functions in the *plots* package, type

```
>  with(plots);
```

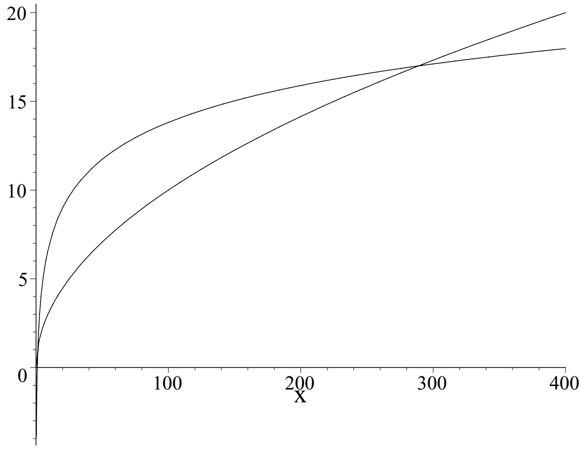


Figure 6.4 MAPLE plot of $y = \sqrt{x}$ and $y = 3\log x$.

6.1.4 Polar plots

To plot polar curves we use the `polarplot` function in the *plots* package. Use the command `polarplot(f(t), t=a..b)` to plot the polar curve $r = f(\theta)$. Try

```
> with(plots):
> polarplot(cos(5*t),t=0..2*Pi);
```

The resulting plot appears in Figure 6.5.

When you try this the first time you will notice the scale on the x-axis is different from that on the y-axis. To make the scales the same, hold the first mouse button on Projection and release on Constrained; or, better still, click on .

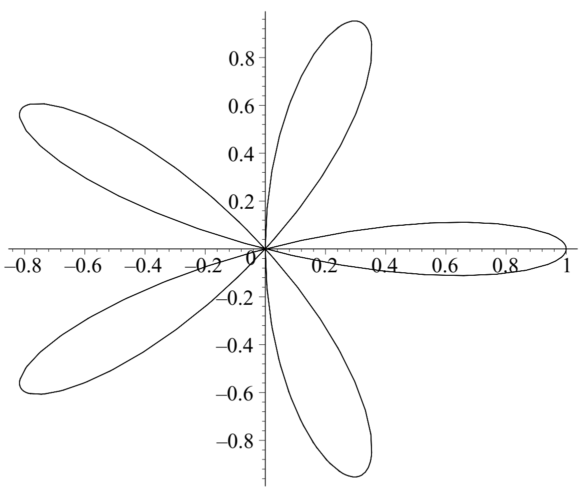


Figure 6.5 MAPLE plot of the polar curve $r = \cos 5\theta$.

We can also plot multiple polar curves. Try

```
> polarplot({cos(5*t),t},t=0..2*Pi);
```

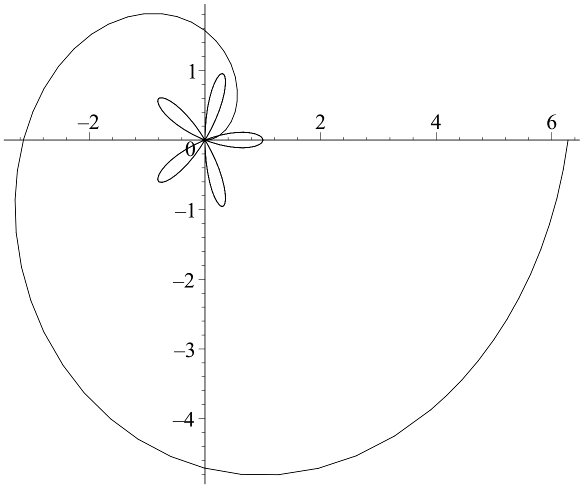


Figure 6.6 MAPLE plot of two polar curves.

You can use `polarplot(L,options)` where L is a list or set. If no range for the angle is specified, the default range $-\pi \le \theta \le \pi$ is taken.

There is another way to plot polar curves. Since $x = r\cos\theta$ and $y = r\sin\theta$, the polar curve $r = f(\theta)$ is given parametrically by

$$x = f(\theta)\,\cos\theta, \qquad y = f(\theta)\,\sin\theta.$$

For example, the polar curve $r = \cos 5\theta$ is given parametrically by

$$x = \cos 5\theta\,\cos\theta, \qquad y = \cos 5\theta\,\sin\theta,$$

so try

```
>  plot([cos(t)*cos(5*t),sin(t)*sin(5*t),t=0..2*Pi]);
```

You should obtain the same plot.

6.1.5 Plotting implicit functions

In Section 6.1.2 we used a parameterization to plot the curve $x^2 + 4y^2 = 1$. Alternatively, we can plot implicitly defined functions using the `implicitplot` command in the *plots* package. Try

```
>  with(plots):
>  implicitplot(x^2+4*y^2=1,x=-1..1,y=-1/2..1/2);
```

This should agree with what we obtained before.

6.1.6 Plotting points

In MAPLE, we plot the points

$$(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_n, y_n)$$

with the command `plot([[x1,y1],[x2,y2], ... ,[xn,yn]])`. Try

```
> L := [[0,0],[1,1],[2,3],[3,2],[4,-2]]:
> plot(L);
```

The resulting plot is given in Figure 6.7.

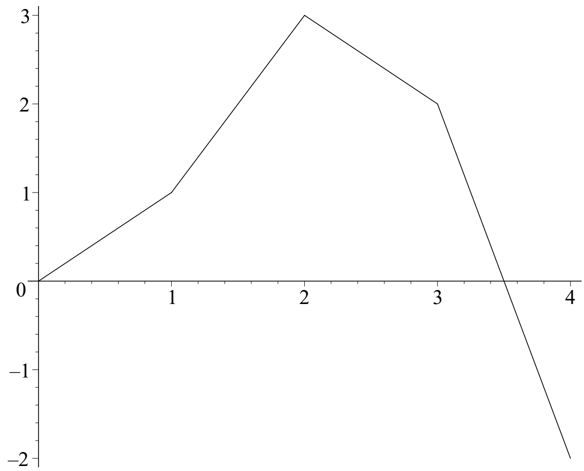


Figure 6.7 MAPLE plot of some data points.

Notice that MAPLE (by default) has drawn lines between the points. To plot the points and nothing but the points, try

```
> plot(L, style=point, symbol=circle);
```

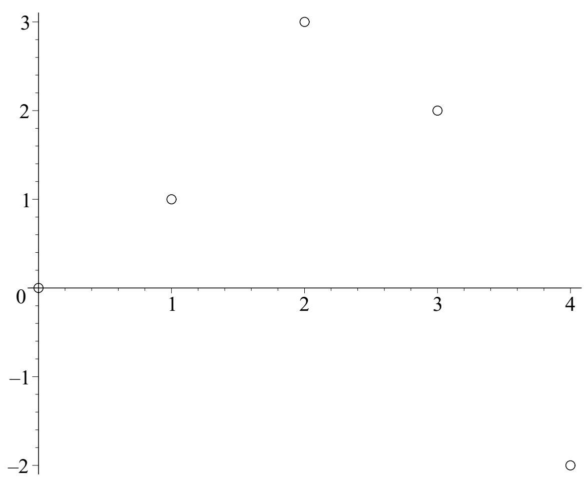


Figure 6.8 MAPLE plot of some unconnected data points.

The points correspond to circles. Try plotting this without the `symbol=circle` option.

6.1.7 Title and text in a plot

To put a title above a plot, we use the option `title`. Try

```
> p1:=plot([sqrt(x),3*log(x)],x=0..400,
    title=`The Square Root and log functions`):
> display(p1);
```

To add text to a plot, we use the `texplot` and `display` functions in the *plots* package. Try

```
> p2:=textplot([[360,16,`y=3log(x)`],[130,10,`y=sqrt(x)`]]):
> display(p1,p2);
```

`textplot([x1,y1,string])` creates a plot with *string* positioned at (x_1, y_1).

We add a legend to a plot. Try

```
> plot([sqrt(x),3*log(x)],x=0..400,
    title="The Square Root \n and log functions",
    legend=["y=sqrt(x)","y=3log x"]);
```

We can add Greek letters and other symbols to plots using the Symbol font. Below is a table showing Greek letters with corresponding Roman letters.

a	b	c	d	e	f	g	h	i	j	k	l	m	n
α	β	χ	δ	ϵ	ϕ	γ	η	ι	φ	κ	λ	μ	ν

o	p	q	r	s	t	u	v	w	x	y	z
o	π	θ	ρ	σ	τ	υ	ϖ	ω	ξ	ψ	ζ

A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	B	X	Δ	E	Φ	Γ	H	I	ϑ	K	Λ	M	N

O	P	Q	R	S	T	U	V	W	X	Y	Z
O	Π	Θ	P	Σ	T	Y	ς	Ω	Ξ	Ψ	Z

To produce μ at the point $(1, 1)$ in a plot, we try:

```
> with(plots):
> textplot([1,1,`m`], font=[SYMBOL,12]);
```

As an illustration, we will plot two normal curves with means μ and $\mu*$. We need to load the *stats* package so we can plot normal density functions. We will discuss the *stats* package in more detail in Chapter 16.

```
> with(stats):
> with(plots):
> xaxis:=plot([[-5,0],[7,0]]):
> meanl1:=plot([[0,0],[0,0.42]],linestyle=2):
```

```
> meanl2:=plot([[1,0],[1,0.42]],linestyle=2):
> p1:=plot(statevalf[pdf,normald[0,1]](t),t=-5..5):
> p2:=plot(statevalf[pdf,normald[1,1]](t),t=-4..5):
> t1:=textplot([0,-0.02,m],font=[SYMBOL,12],'align=BELOW'):
> t2:=textplot([1,-0.02,"m*"],font=[SYMBOL,12],'align=BELOW'):
> display(xaxis,p1,p2,t1,t2,meanl1,meanl2,
    view=[-5..7,-0.02..0.42], axes=none);
```

The resulting plot appears below in Figure 6.9.

`xaxis` gives a horizontal line corresponding to the x-axis. The two vertical dotted lines are `meanl1` and `meanl2`, indicating the two means μ and $\mu*$. The two normal curves are given by `p1` and `p2`. The *stats* function `statevalf[pdf,normald[`μ`,`σ`]]` computes values of the normal density function with mean μ and standard deviation σ. See Chapter 16 for more details. The symbols μ and $\mu*$ were placed in their correct positions using `textplot`. The `align` option in `textplot` can take the values `ABOVE`, `BELOW`, `RIGHT`, or `LEFT`. See `?plots[textplot]` for more details.

Other keyboard characters give different symbols when using symbol font:

@	$	^	'	|
$\cong$	$\exists$	$\perp$	$\ni$	$\therefore$

Try

```
> textplot([1,1,`@`],font=[SYMBOL,12]);
```

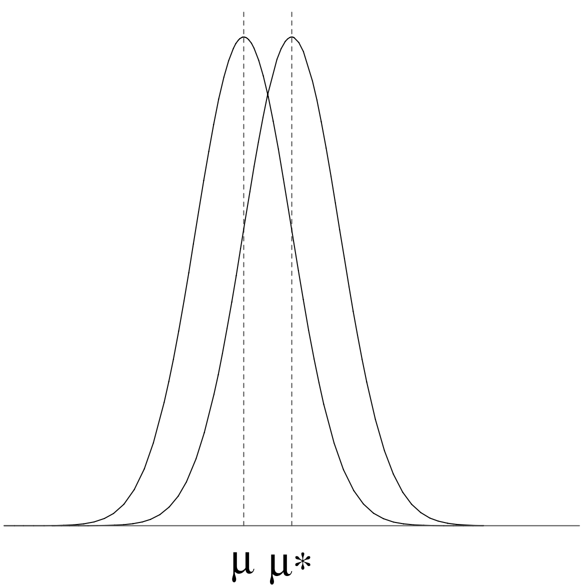


Figure 6.9 Two normal curves.

Other symbols are encoded as character numbers using `convert([`n`],bytes)`. Here n is an integer satisfying $32 \le n \le 126$, $161 \le n \le 254$. Try

```
> with(plots):
> textplot([0,0,convert([192], bytes)], font=[SYMBOL,12],
  axes=none);
```

ℵ

To view more of these symbols define `chardisplay`:

```
> with(plots):
>   chardisplay:=n -> display(textplot([0,0,convert([n],bytes)],
  font=[SYMBOL,12]),axes=none):
```

Now try `chardisplay(n)` for different values of n.

```
> chardisplay(169);
```

6.1.8 Plotting options

The plotting options are given after the function and ranges in the `plot` command. The following information is taken from the MAPLE help pages. See `?plot[options]`.

`adaptive`

If set to false, disables the use of adaptive plotting.

`axes`

Specifies the type of axes, one of: `FRAME`, `BOXED`, `NORMAL`, and `NONE`.

`axesfont=l`

Font for the labels on the tick marks of the axes, specified in the same manner as `font`.

`color=n`

Allows the user to specify the color of the curves to be plotted. The spelling *colour* may also be used. See `?plot,color` for details.

`coords=name`

Indicates that a parametric plot is in the coordinate system specified by `name`. See `?plot[coords]` for more information about the choices of coordinate system.

`discont=s`

Setting `s` to `true` forces plot to first call the function `discont` to determine the discontinuities of the input and then break the horizontal axis into appropriate intervals where the expression is continuous.

`filled=truefalse`

If the filled option is set to true, the area between the curve and the x-axis is given a solid color. This option is valid only with the following commands: `plot`, `contourplot`, `implicitplot`, `listcontplot`, `polarplot`, and `semilogplot`.

font=l

Font for text objects in the plot; l is a list [family, style, size], where family is one of TIMES, COURIER, HELVETICA, and SYMBOL. For TIMES, style may be one of ROMAN, BOLD, ITALIC, or BOLDITALIC. For HELVETICA and COURIER, style may be omitted or select one of BOLD, OBLIQUE, or BOLDOBLIQUE. SYMBOL does not accept a style option. The final value, size, is the point size to be used. As an example, try font=[HELVETICA,12].

labels=[x,y]

This option specifies labels for the axes. The values of x and y must be strings. The default labels are the names of the variables used in the plotting function.

labeldirections=[x,y]

This option specifies the direction in which labels are printed along the axes. The values of x and y must be HORIZONTAL or VERTICAL. The default direction of any labels is HORIZONTAL.

labelfont=l

Font for the labels on the axes of the plot, specified in the same manner as font.

legend=s

A legend for a plot can be specified by either a string or a list of strings. When more than one curve is being plotted, they must be specified as a list and there must be a legend for each curve.

linestyle=n

Controls the dash pattern used to render lines in the plot. When n=1, the line is solid. For n=2 the style is dot, n=3 gives dash, and n=4 gives dash-dot.

numpoints=n

Specifies the minimum number of points to be generated (the default is n = 50). Note: plot employs an adaptive plotting scheme that automatically does more work when the function values do not lie close to a straight line. Hence plot will often generate more than the minimum number of points.

resolution=n

Sets the horizontal display resolution of the device in pixels (the default is n = 200). The value of n is used to determine when the adaptive plotting scheme terminates. A higher value will result in more function evaluations for non-smooth functions.

sample

Supplies a list of parameter values to be used for the initial sampling of the function(s). When coupled with adaptive=false, this option allows explicit control over the function evaluations performed by plot.

`scaling`

Controls the scaling of the graph. Either `CONSTRAINED` or `UNCONSTRAINED`. Default is `UNCONSTRAINED`. `CONSTRAINED` means the same scale is used on both axes.

`style=s`

The interpolation style must be one of `LINE`, `POINT`, `PATCH`, or `PATCHNOGRID`. The default is `LINE`. `POINT` style plots points only, `LINE` interpolates between the points, `PATCH` uses the patch style for plots containing polygons, and `PATCHNOGRID` is the `PATCH` style without the grid lines.

`symbol=s`

Symbol for points in the plot, `s` is one of `BOX`, `CROSS`, `CIRCLE`, `POINT`, and `DIAMOND`.

`symbolsize=n`

The size (in points) of a symbol used in plotting can be given by a positive integer. This does not affect the symbol `POINT`. The default symbol size is 10.

`thickness=n`

Thickness of lines in the plot; `n` should be 0, 1, 2, or 3. 0 is the default thickness.

`tickmarks=[m,n]`

This option specifies that a reasonable number of points no less than `m` and `n` should be marked along the x-axis and y-axis, respectively. Both `m` and `n` must be either a positive integer or the name *default*. If tickmarks are desired along only one axis, use `xtickmarks` or `ytickmarks` instead.

`title=t`

The title for the plot. `t` must be a character string. The default is no title. You can create multiline titles for standard plots. Use the characters "\n" in the character string to obtain a line break in the title.

`titlefont=l`

Font for the title of the plot, specified in the same manner as `font`.

`view=[xmin..xmax, ymin..ymax]`

This option indicates the minimum and maximum coordinates of the curve to be displayed on the screen. The default is the entire curve.

`xtickmarks=n`

Indicates that a reasonable number of points no less than `n` should be marked along the horizontal axis; `n` must be a positive integer or a list. If `n` is a list, then the list of values is used to mark the axis; the corresponding option `ytickmarks=n` can be used to specify the minimum number of divisions along the vertical axis, or a list of values used to mark the vertical axis.

6.1.9 Saving and printing a plot

There are several ways to save a plot. Any plot that is part of a worksheet will be saved when the worksheet is saved. See Sections 9.2 and 9.3. The `plotsetup` function can be used to save a plot as a file suitable for other drivers. This is done by specifying the `plotdevice` variable. Common settings for `plotdevice` are

`bmp`	Windows BMP file
`cps`	Color Postscript file
`gif`	GIF image file
`ps`	encapsulated Postscript file
`jpeg`	24-bit color JPEG file
`hpgl`	HP GL file

Here is an example.

```
>  plotsetup(ps, plotoutput=‘plot.ps‘,
     plotoptions=‘portrait, noborder‘);
>  plot(sin(x),x=-2*Pi..2*Pi);
>  interface(plotdevice=inline);
```

In this session, a plot of $y = \sin(x)$ was written to the Postscript file *plot.ps*, in portrait style with no surrounding border. The `interface` function was used so that any future plot will be within the worksheet. Otherwise, if `plotsetup` is not changed, any future plot will overwrite the file *plot.ps*.

A plot may be printed as part of the worksheet using the menu. Alternatively, it can be saved as a file and printed using a graphics driver. For example, try

```
>  plotsetup(hpgl, plotoutput=‘plot.hp‘,plotoptions=‘laserjet‘);
```

when printing a plot with an HP Laserjet printer. For more information, use the help commands `?plotsetup`, `?plot[device]`.

A plot may be also saved using the Export menu. Click on a plot in the worksheet that you want to save and then click on Export. A menu should appear:

Drawing Exchange Format (DXF)...
Encapsulated Postscript (EPS)...
Graphics Interchange Format (GIF)...
JPEG File Interchange Format (JPG)...
Persitence of Vision (POV)...
Windows Bitmap (BMP)...
Windows Metafile (WMF)...

Select your favorite file format. A **Save As** window should appear. Type an appropriate file name in the File name box and click on `Save`.

6.1.10 Other plot functions

We describe briefly the other two-dimensional plotting functions available in the *plots* package. Don't forget to load the *plots* package.

```
> with(plots):
```

complexplot

Suppose $f(t)$ is a complex-valued function, say

$$f(t) = u(t) + i\,v(t),$$

where $u(t)$ and $v(t)$ are real-valued functions. Then the function `complexplot(f(t),t=a..b)` will plot the curve given parametrically by

$$x = u(t), \qquad v(t), \qquad a \le t \le b.$$

```
> complexplot(exp(I*x),x=0..2*Pi);
```

conformal

Suppose $f(z)$ is a complex-valued function, then the function `conformal (f(z),z=z1..z2)` will plot the image of a rectangular grid under the mapping $w = f(z)$. The complex numbers `z1` and `z2` determine two corners in the rectangular grid. More details and examples for this function will be given in Section 11.6.

```
> conformal(sin(z),z=-1-I..1+I);
```

The resulting plot appears below in Figure 6.10.

coordplot

The function `coordplot(coord,rangelist,eqns)` plots *graph paper* of the specified `coord`inate system. The available coordinate systems are `bipolar`, `cardioid`, `cartesian`, `cassinian`, `elliptic`, `hyperbolic`, `invcassinian`, `invelliptic`, `logarithmic`, `logcosh`, `maxwell`, `para- bolic`, `polar`, `rose`, and `tangent`. For a description of these coordinate systems see `?coords`. `rangelist` is a list of two coordinate ranges, and `eqns` are optional equations that modify the plot. See `?plots[coord]` for more details.

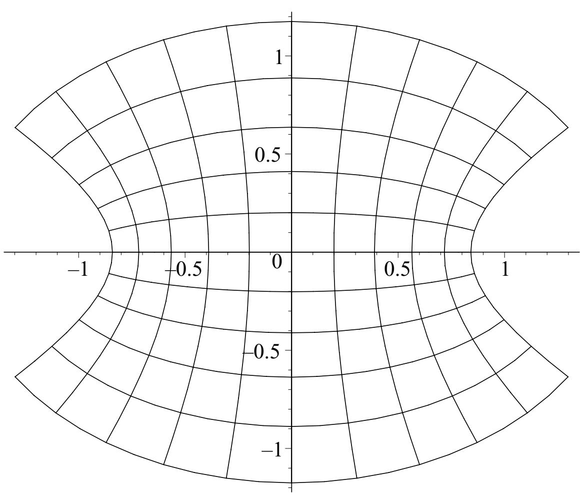


Figure 6.10 The conformal mapping $w = \sin z$.

```
> coordplot(polar,[0..2,0..2*Pi],labelling=true,
    grid=[5,13], view=[-2..2,-2..2],scaling=constrained);
```

The resulting plot appears below in Figure 6.11.

fieldplot

The function `fieldplot([f(x,y),g(x,y),x=a..b,y=c..d)` plots the two-dimensional vector field

$$\vec{F}(x,y) = f(x,y)\,\vec{i} + g(x,y)\,\vec{j},$$

where $a \le x \le b$, and $c \le y \le d$. Let's plot the direction field

$$\vec{F}(x,y) = -y\,\vec{i} + x\,\vec{j},$$

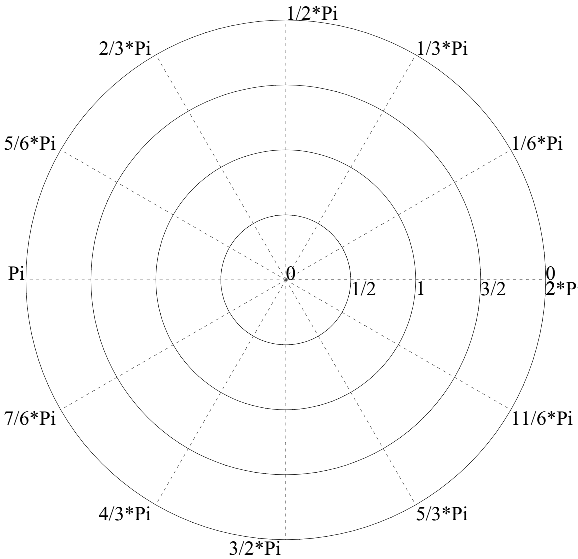


Figure 6.11 Polar graph paper via coordplot.

```
> fieldplot([-y,x],x=-1..1,y=-1..1);
```

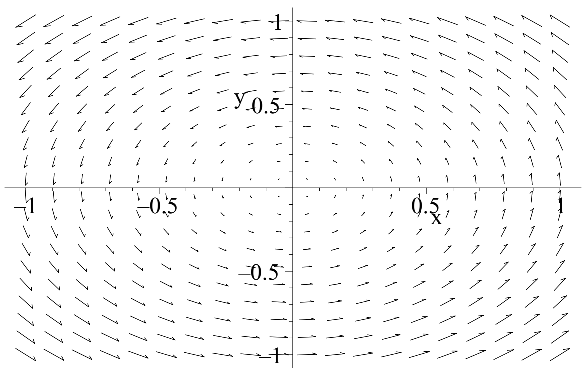


Figure 6.12 2D direction field.

inequal

The function inequal(ineqs,x=a..b,y=c..d,options) will plot regions defined by *linear inequalities* in the variables x and y over the specified ranges. We plot the regions specified by the inequalities

$$x - y \leq 0, \quad x + y \leq 1, \quad 5 + 2x \geq y,$$

where $-6 \leq x \leq 3$, and $-6 \leq y \leq 6$. The intersection is colored red and elsewhere is colored yellow.

```
> inequal( { x-y<=0,x+y<=1,5+2*x>=y}, x=-6..3,y=-6..6,
   optionsfeasible=(color=red),optionsexcluded=(color=yellow));
```

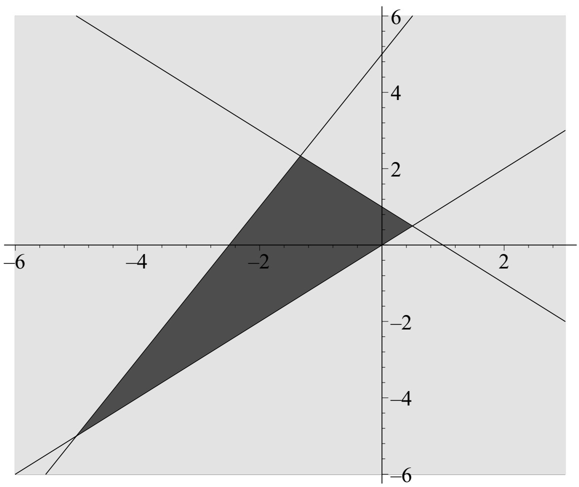


Figure 6.13 Graphing inequalities using inequal.

Warning: This function has some bugs for regions specified with strict inequalities. For example, try

```
> inequal( { x-y<0,x+y<1,5+2*x>y}, x=-6..3,y=-6..6,
optionsfeasible=(color=red),optionsexcluded=(color=blue));
```

logplot

The function logplot(f(x),x=a..b) creates a plot of the function $f(x)$ ($a \leq x \leq b$) with a logarithmic scale on the y axis. Try

```
> logplot(tan(x),x=0..1.55);
```

pareto

The pareto function plots a Pareto diagram of specified frequencies. For more information see ?plots[pareto].

pointplot

The function pointplot(L) plots a list or set of points L. It is basically equivalent to the command plot(L,style=point).

polygonplot

If L is a list of points, the function polygonplot(L) creates a plot of a polygon whose vertices are these points.

```
> L := [[0,1],[1,1],[1/2,1/2]]:
> polygonplot(L);
```

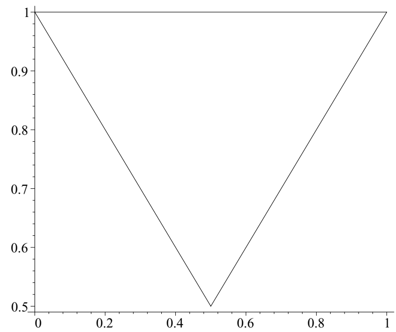


Figure 6.14 A polygon plot of a triangle.

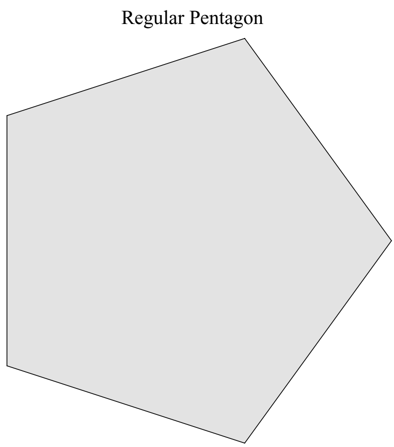


Figure 6.15 A polygon plot of a pentagon.

Observe that we plotted the triangle with vertices $(0,1)$, $(1,1)$, $(1/2,1/2)$. In general, straight lines connect the points of L, and then the last point in L is connected to the first point. We can add color with the color option.

```
> ngon := n -> [seq([ cos(2*Pi*i/n), sin(2*Pi*i/n) ],
    i = 1..n)]:
> polygonplot(ngon(5),scaling=constrained,axes=none,
    title="Regular Pentagon",color=yellow);
```

The resulting plot is given above in Figure 6.15. The function ngon(n) returns n equally spaced points on the unit circle. We plotted a regular polygon by applying the polygonplot function to the list of five points returned by ngon(5). Below we define a function nstar for plotting an n-pointed star using polygonplot.

```
> npt :=(r,i,n) -> [r*cos(2*Pi*i/n),r*sin(2*Pi*i/n)];
```

$$npt := (r, i, n) \mapsto [r\cos(2\,\frac{\pi\,i}{n}), r\sin(2\,\frac{\pi\,i}{n})]$$

```
> shard:=(i,n,col)->polygonplot([npt(1,i,n),npt(2,(2*i+1),
    2*n),npt(1,i+1,n)],color=col):
> nstar:=(n,col)->display(seq(shard(i,n,col),i=1..n),
    scaling=constrained, axes=none):
> nstar(17,blue);
```

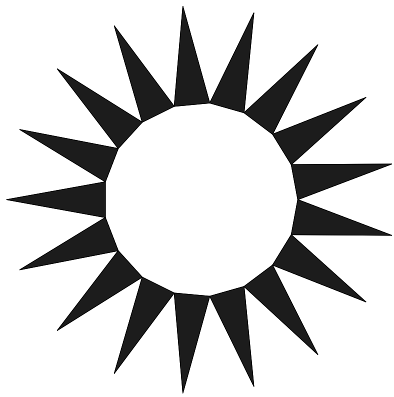


Figure 6.16 A polygon plot of a 17 pointed star.

The function `nstar(n,color)` should plot an n-pointed star with the specified color. It is defined in terms of the two functions `npt` and `shard`. `npt(r,i,n)` returns the ith point in a sequence of n equally spaced points on the circle of radius r. `shard(i,n,col)` plots a triangle corresponding to the ith point of the star.

<u>`semilogplot`</u>

The function `semilogplot(f(x),x=a..b)` creates a plot of the function $f(x)$ ($a \le x \le b$) with a logarithmic scale on the x axis. Try

```
> semilogplot({sqrt(x),log(x)},x=0.1..100);
```

<u>`setoptions`</u>

This function sets global options for two-dimensional plots. These become default for all subsequent 2D plots in the same MAPLE session. See `?plot[options]` for a list of options.

```
> setoptions(title=`Semilog plot of Sqrt and Log`,
    axes=BOXED);
> semilogplot({sqrt(x),log(x)},x=0.1..100);
```

To remove these options, do

```
> setoptions(title=``, axes=normal);
```

6.2 Three-dimensional plotting

The syntax for plotting an expression (or function) in two variables (say x, y) is `plot3d(f(x,y), x=a..b,y=c..d)`. For example, to plot the function $z = e^{-(x^2+y^2-1)^2}$ for $-2 \leq x, y \leq 2$, we use the command

```
> plot3d(exp(-(x^2 + y^2-1)^2), x=-2..2, y=-2..2);
```

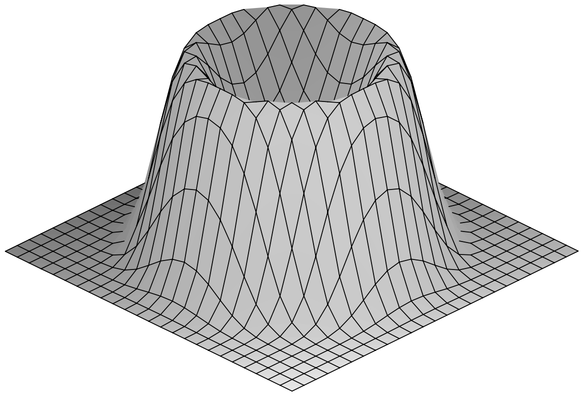


Figure 6.17 A plot of the function $z = e^{-(x^2+y^2-1)^2}$.

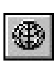
Observe (as before with two-dimensional plotting) that the plot appears in the worksheet. Now try clicking on the plot. Notice the appearance of the Style, Colour, Axes, Projection, and Animation menus. The context bar has also changed. There should be a pair of small windows labelled ϑ and ϕ, each containing the number 45. This pair of numbers refers to a point in spherical coordinates and corresponds to the orientation of the plot. There should also be 13 new buttons. Try clicking on each button to see its effect.

Specifies orientation.

Render the plot using the polygon patch style with gridlines.

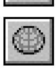
Render the plot using the polygon patch style.

Render the plot using the polygon patch and contour style.



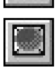
Render the plot using the hidden line removal style.

Render the plot using the contour style.



Render the plot using the wireframe style.

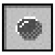
Render the plot using the point style.



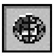
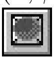
Draw the plot axes as an enclosed box.

 Draw the plot axes as an exterior frame.

 Draw the plot axes in traditional form.

 Suppress the drawing of plot axes.

 Use the same scale on each axis.

Now, hold the first mouse button down on the plot and at the same time move it around. Notice how the plot rotates as you move the mouse, and notice that the value of (ϑ, ϕ) changes. Below in Figure 6.18 is a plot obtained by clicking on  and  and selecting $(\vartheta, \phi) = (22, 67)$.

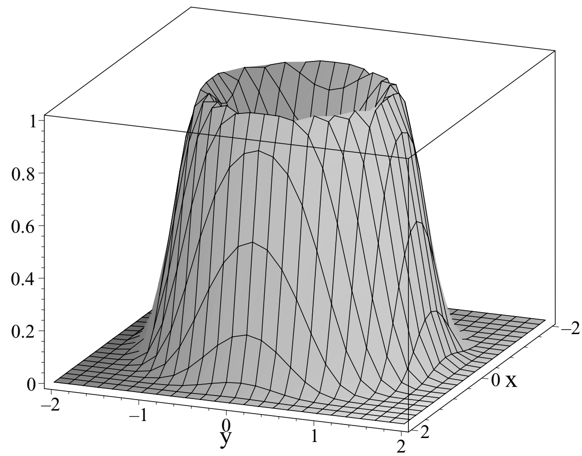
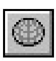


Figure 6.18 A MAPLE plot with boxed axes.

Now, try clicking  to see some hidden detail of the plot. You might use the `grid` option to increase the number of contours plotted. Try

```
> plot3d(exp(-(x^2 + y^2)^2), x=-2..2,y=-2..2, grid=[50,50]);
```

This time try clicking the right mouse button on the plot. A context menu should appear:

Copy
Style ▶
Color ▶
Axes ▶
Projection ▶
Animation ▶
Export As ▶

Select Color. Another menu appears.

✓ XYZ XY Z Z (Hue) Z (Grayscale) No Coloring Default Coloring
No Lighting User Lighting Light Scheme 1 Light Scheme 2 Light Scheme 3 Light Scheme 4

Select Light Scheme 1. Notice how the coloring of the plot changes. Try out some other selections.

Now let's plot something simpler such as a plane. Remember that the equation of a plane takes the form

$$ax + by + cz = d.$$

To plot such a plane, we solve for z and plot the resulting function of x and y. As an example, we plot the plane

$$2x + 3y + 2z = 6.$$

Solving for z, we find that we must plot the function $f(x, y) = 3 - x - 3y/2$.

```
>  plot3d(3 - x - 3*y/2,x=0..3,y=0..2,axes=normal,
     orientation=[20,60], view=[0..4,0..3,0..4]);
```

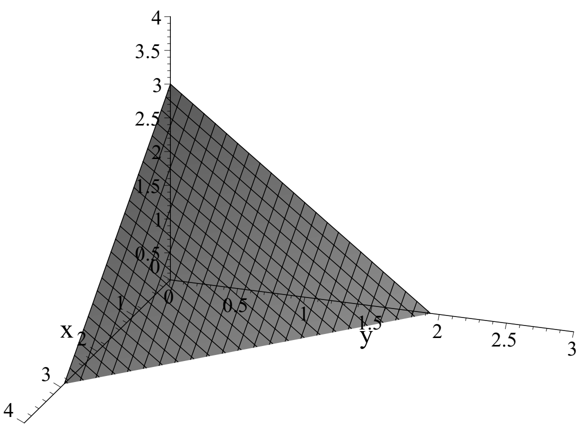


Figure 6.19 Plot of a plane.

The option `axes=normal` gave the usual x-, y- and z-axes. The option `orientation=[20,60]` set $\vartheta = 20$ and $\phi = 60$. The `view` option restricted the range for each variable as $0 \le x \le 4$, $0 \le y \le 3$, $0 \le z \le 4$. This way we were able to plot that portion of the plane that lies in the first octant (i.e., x, y, $z \ge 0$).

6.2.1 Parametric plots

To plot the surface parameterized by

$$x = f(u, v), \quad y = g(u, v), \quad z = h(u, v),$$

where $a \le u \le b$, $c \le v \le d$; use the command `plot3d([f(u,v), g(u,v), h(u,v)], u=a..b, v=c ..d)`. For example, the hyperboloid

$$x^2 + y^2 - z^2 = 1,$$

may be parameterized by

$$x = \sqrt{1+u^2}\,\cos t, \quad y = \sqrt{1+u^2}\,\sin t, \quad z = u,$$

where $-\infty < u < \infty$ and $0 \le t \le 2\pi$. Try

```
> plot3d([sqrt(1+u^2)*cos(t),sqrt(1+u^2)*sin(t),u],
    u=-1..1, t=0..2*Pi);
```

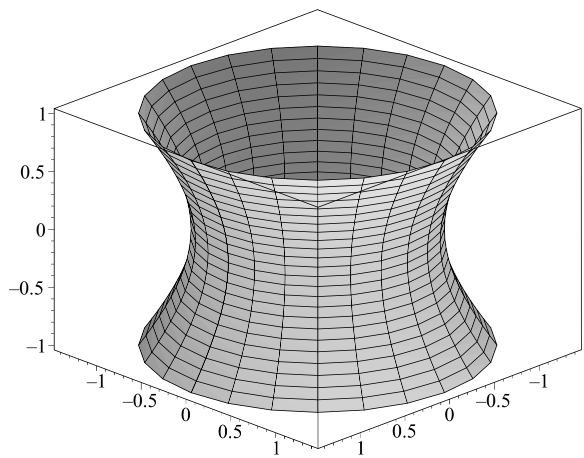


Figure 6.20 MAPLE plot of a hyperboloid.

A plot with $(\vartheta, \phi) = (45, 60)$ is given above in Figure 6.20.

6.2.2 Multiple plots

To plot the two functions

$$z = e^{-x^2-y^2},$$
$$z = x + y + 1,$$

try

```
> plot3d({exp(-x^2-y^2),x+y+1},x=-2..2, y=-1..1);
```

with $(\vartheta, \phi) = (120, 45)$. As with two-dimensional plotting, multiple three-dimensional plots can be produced using the `display` function in the *plots* package. Try

```
> with(plots):
> p1:=plot3d(exp(-x^2-y^2),x=-2..2, y=-1..1):
> p2:=plot3d(x+y+1,x=-2..2,y=-1..1):
> display(p1,p2);
```

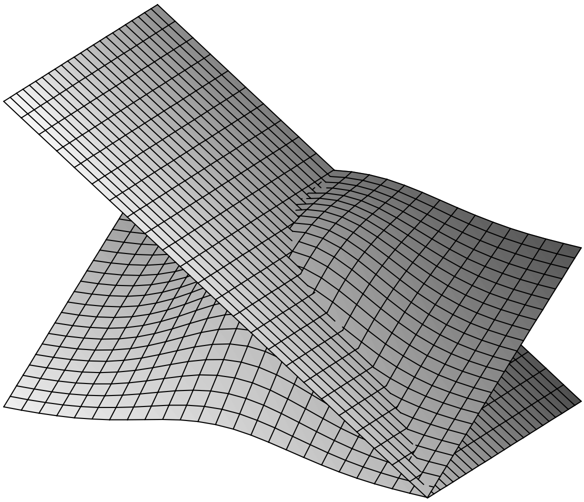


Figure 6.21 Two intersecting surfaces.

6.2.3 Space curves

To plot the space curve

$$x = f(t), \quad y = g(t), \quad z = h(t),$$

where $a \leq t \leq b$, we use the `spacecurve` function in the *plots* package. The command is `spacecurve([f(t),g(t),h(t)],t=a..b)`. We plot the helix

$$x = \cos t, \quad y = \sin t, \quad z = t.$$

Try

```
> with(plots):
> spacecurve([cos(t),sin(t),t],t=0..4*Pi, numpoints=200,
   orientation=[22,60],axes=BOXED);
```

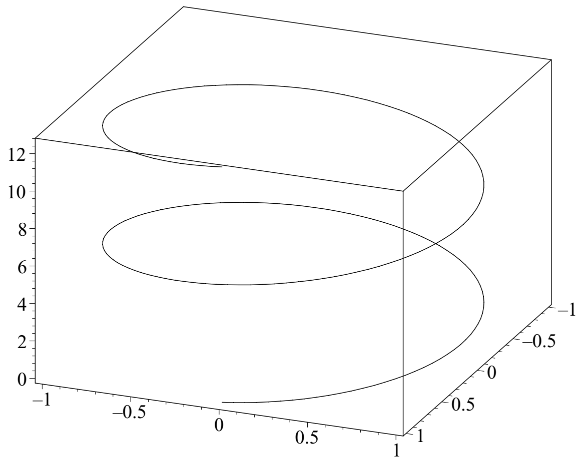


Figure 6.22 MAPLE plot of a helix.

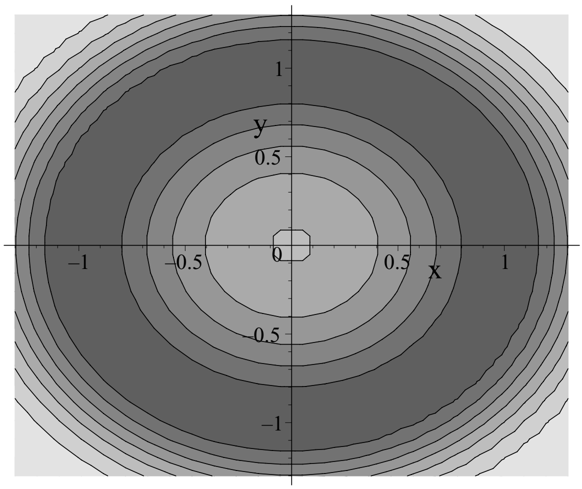


Figure 6.23 A contour plot.

6.2.4 Contour plots

The graph of a function of two variables may be visualized with a two-dimensional contour plot. To produce contour plots, we use the functions `contourplot` and `contourplot3d` in the *plots* package. `Contourplot3d` "paints" the contour plot on the corresponding surface. Try

```
>  with(plots):
>  contourplot(exp(-(x^2+y^2-1)^2), x=-(1.3)..(1.3),
     y=-(1.3)..(1.3), filled=true, coloring=[blue,red]);
```

The resulting plot is given above in Figure 6.23.

```
>  contourplot3d(exp(-(x^2+y^2-1)^2), x=-(1.3)..(1.3),
```

```
y=-(1.3)..(1.3), filled=true, coloring=[blue,red]);
```

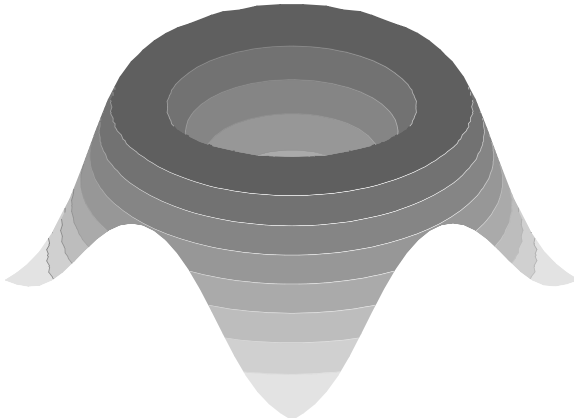


Figure 6.24 A 3D contour plot.

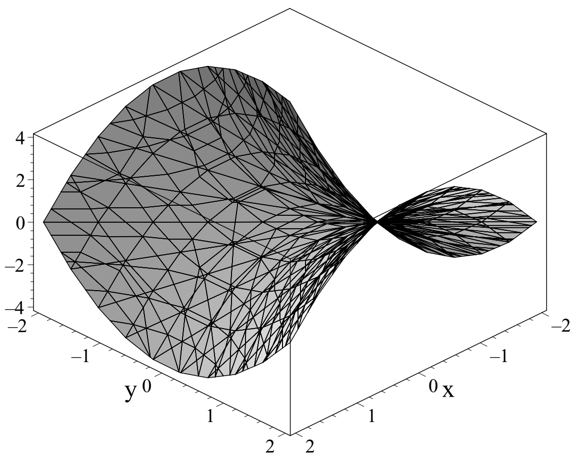


Figure 6.25 MAPLE plot of a hyperbolic paraboloid.

6.2.5 Plotting surfaces defined implicitly

To plot the surface defined implicitly by the equation

$$f(x, y, z) = c,$$

use the command `implicitplot3d(f(x,y,z)=c, x=a..b, y=d..e, z=g..h)` in the *plots* package. For example, to plot the hyperbolic paraboloid

$$y^2 - x^2 = z,$$

try

```
> with(plots):
> implicitplot3d(y^2 - x^2 = z, x=-2..2, y=-2..2,
   z=-4..4);
```

The resulting plot is given above in Figure 6.25.

In Section 6.2.1 we obtained a plot of the surface

$$x^2 + y^2 - z^2 = 1,$$

by using a parameterization. This time, try

```
> implicitplot3d(x^2 + y^2 - z^2 = 1, x=-1..1, y=-1..1,
   z=-1..1);
> implicitplot3d(x^2 + y^2 - z^2 = 1, x=-2..2, y=-2..2,
   z=-1..1);
```

Notice how care must be taken in choosing the range for each variable.

6.2.6 Title and text in a plot

A title or text may be inserted in a three-dimensional plot in the same way it was done in Section 6.1.7 for two-dimensional plots. Try

```
> with(plots):
> p1:=plot3d(exp(-(x^2+y^2-1)^2), x=-2..2,y=-2..2,
font=[TIMES,ROMAN,12],titlefont=[HELVETICA,BOLD,10],
title=`The surface z=exp(-(x^2+y^2-1)^2)`):
> p2:=textplot3d([0,1.1,1,`Circular Rim`], align=RIGHT,
   color=BLUE):
> display(p1,p2);
```

6.2.7 Three-dimensional plotting options

The options `axes`, `font`, `labels`, `labelfont`, `linestyle`, `numpoints`, `scaling`, `symbol`, `thickness`, `title`, `titlefont`, and `view` should work like they did for two-dimensional plotting (see Section 6.1.8). Other options are given below. This information was taken from the MAPLE help pages. See `?plot3d[options]`.

<u>`ambientlight=[r,g,b]`</u>

This option sets the red, green, and blue intensity of the ambient light for user-defined lighting. `r`, `g`, and `b` must be numeric values in the range 0 to 1.

<u>`axes=f`</u>

This option specifies how the axes are to be drawn, where `f` is one of `BOXED`, `NORMAL`, `FRAME`, and `NONE`. The default axis is `NONE`.