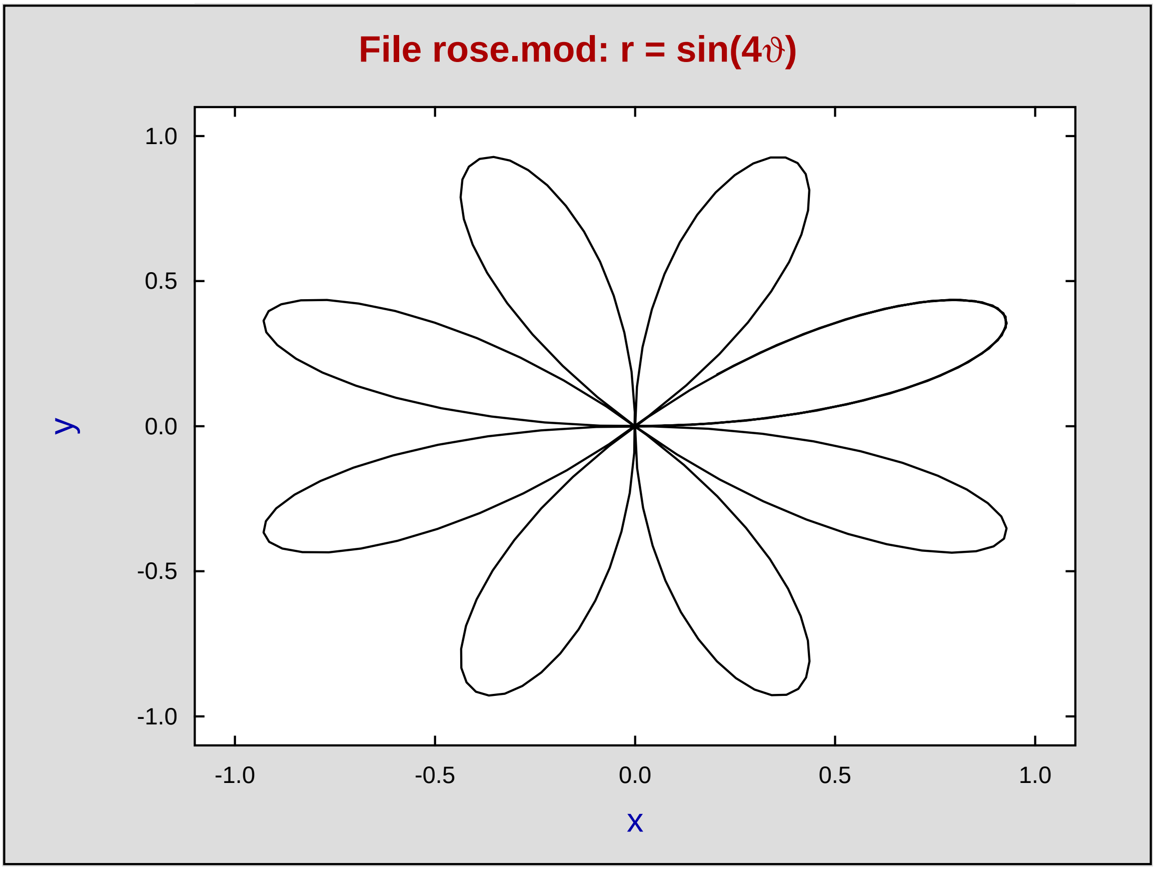
Tutorials and worked examples for simulation,
curve fitting, statistical analysis, and plotting.
http://www.simfit.org.uk

Plotting models expressed in parametric form is often required,e.g. when the model cannot be expressed in standard form. SimFIT programs **simplot** and **usermod** can plot models in the following forms:
Plane curves as $r = r(\theta)$, Plane curves as $x = x(t)$, $y = y(t)$, or Space curves as $x = x(t)$, $y = y(t)$, $z = z(t)$.

Plane curves as $r = r(\theta)$: Example 1

As an example consider the test file `rose_e.mod` for the eight leaved rose plotted below.

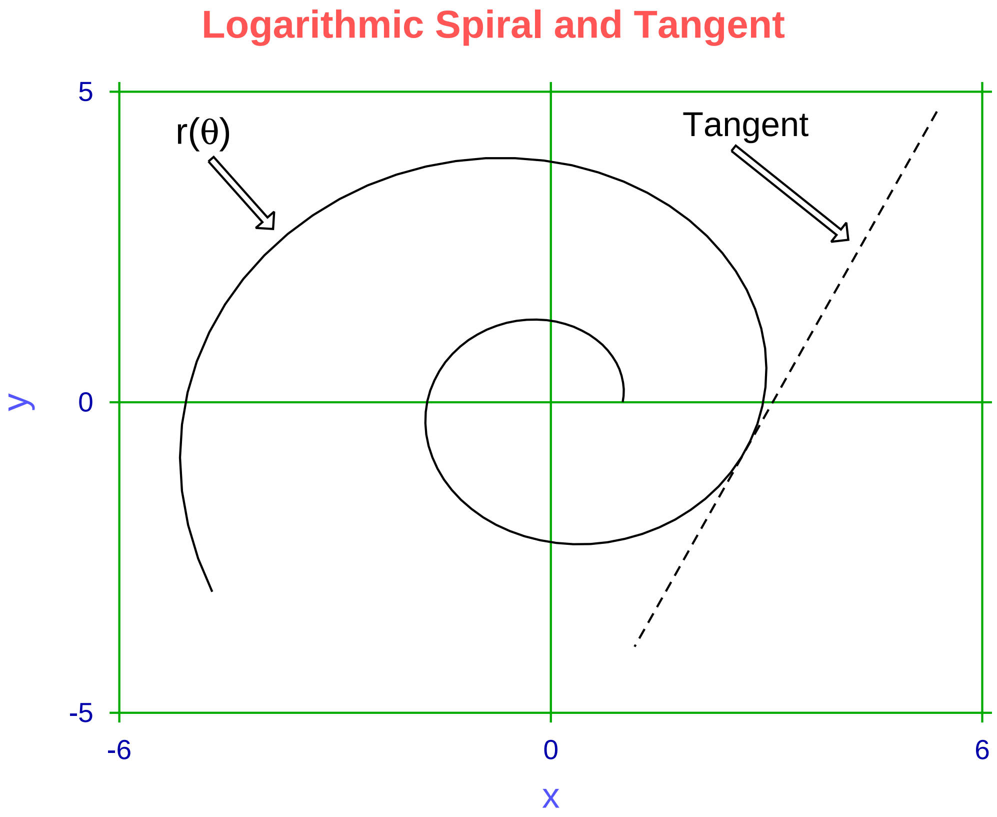
```
  %
Example: Eight leaved rose
r = A*sin(4*theta): where theta = x, r = f(1) and A = p(1)
 %
 1 equation
 1 variable
 1 parameter
 %
 begin{expression}
 f(1) = p(1)sin(4x)
 end{expression}
 %
```



Plane curves as $r = r(\theta)$: Example 2

The next plot illustrates the logarithmic spiral defined in SimFIT model file `camalot_e.mod`.

```
%
Model: Logarithmic Spiral
   r = A*exp(theta*cot(p(1))): where theta = x, r = f(1)
   A = amplitude scaling factor
p(1) = angle in radians between radius vector and tangent
 %
 1 equation
 1 variable
 1 parameter
 %
 begin{expression}
 f(1) = exp{x/tan[p(1)]}
 end{expression}
 %
```



This profile is used in rock climbing camming devices such as Camalots and Friends to maintain a constant angle α between the radius vector for the spiral and the tangent to the curve, defined in `tangent.mod` as

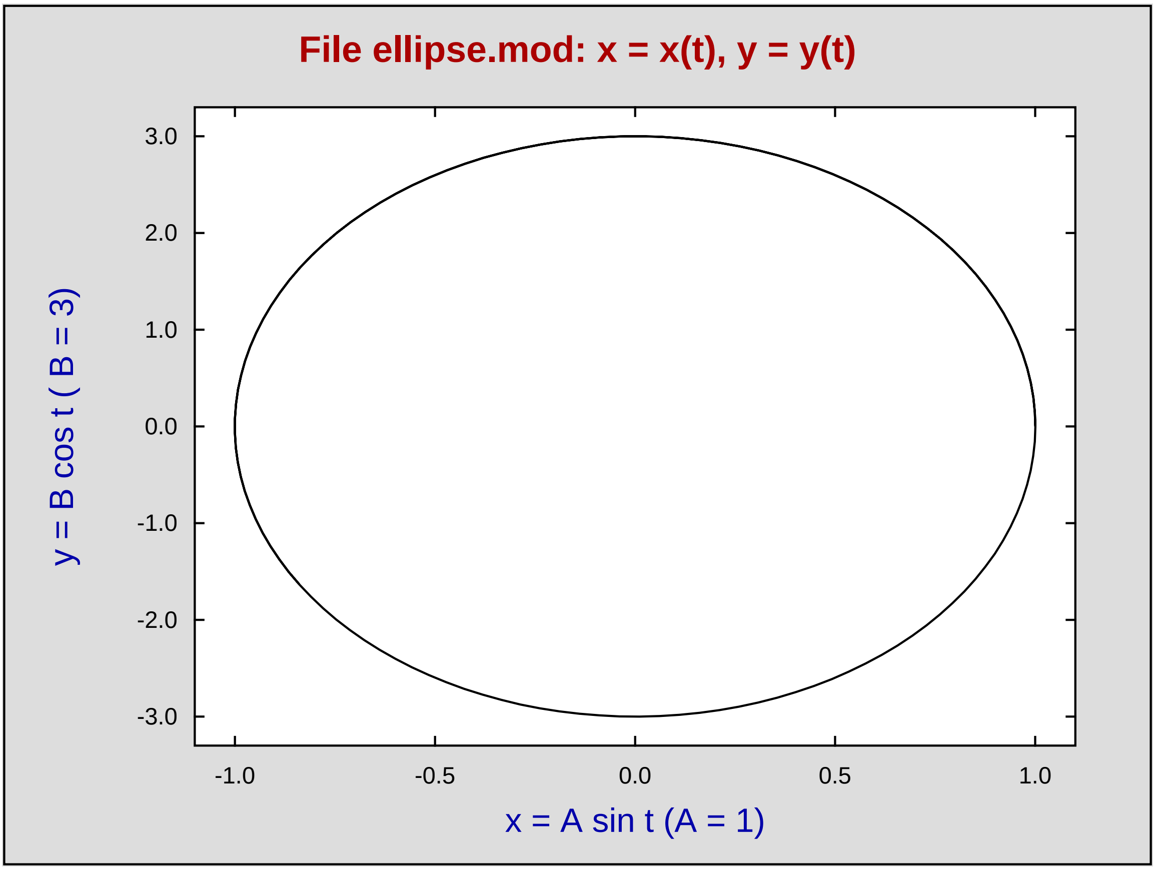
$$r = \frac{A\exp(\theta_0 \cot\alpha)[\sin\theta_0 - \tan(\theta_0 + \alpha)\cos\theta_0]}{\sin\theta - \tan(\theta_0 + \alpha)\cos\theta}.$$

The figure above used $\alpha = p(1) = 1.4, \theta_0 = P(2) = 6$ and **usermod** to generate individual figures over the range $0 \leq \theta = x \leq 10$, then **simplot** plotted the ASCII text coordinates simultaneously, a technique that can be used to overlay any number of curves.

Plane curves as $x(t)$, $y(t)$

The test file `ellipse_e.mod` defines an ellipse as now listed, followed by a plot where the eccentricity, which is evident from the ranges of the axes, is introduced by choosing $A = 1$ and $B = 3$.

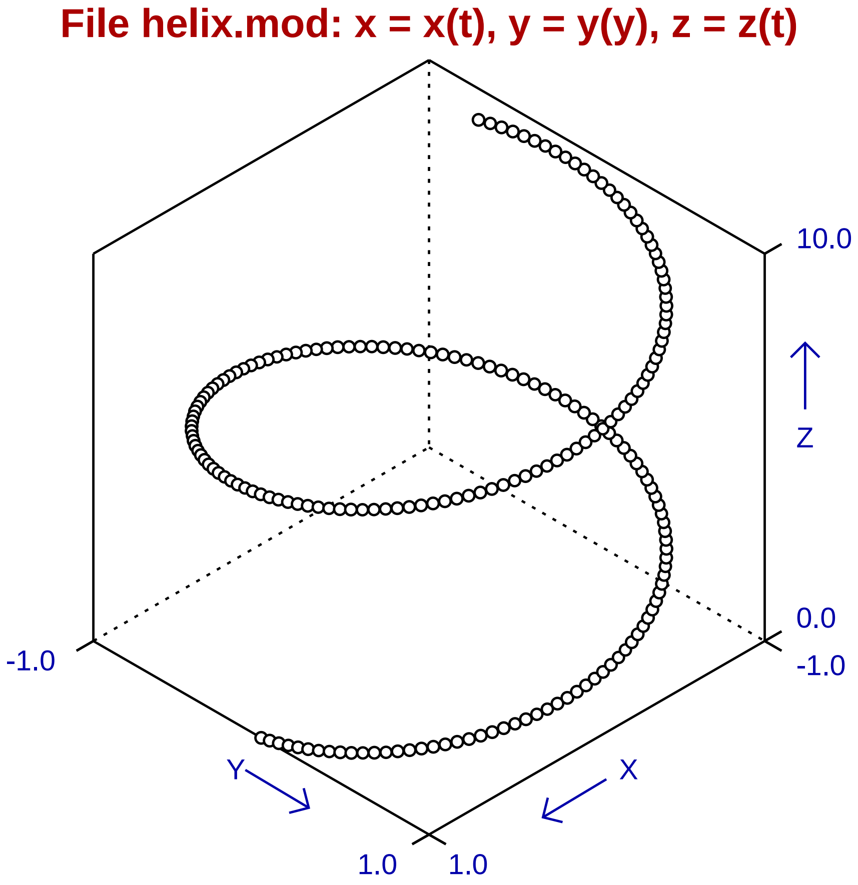
```
 %
Example: the ellipse
         X = A*cos(t), Y = B*sin(t)
         where: t = x, A = p(1), B = p(2)
                and X(t) = f(1), Y(t) = f(2)
 %
 2 equations
 1 variable
 2 parameters
 %
 begin{expression}
 f(1) = p(1)cos(x)
 f(2) = p(2)sin(x)
 end{expression}
 %
```



Space curves as $x(t), y(t), z(t)$

Note that, as with all parametric equations, program **simplot** transforms the parametric curve into sensible plotting parameters so that, as in this case with `helix_e.mod`, only minor editing of the data ranges is required.

```
 %
Example: the helix
         X = A*cos(t), Y = B*sin(t), Z = C*t
         where: t = x, A = p(1), B = p(2), C = p(3)
                and X(t) = f(1), Y(t) = f(2), Z(t) = f(3)
 %
 3 equations
 1 variable
 3 parameters
 %
 begin{expression}
 f(1) = p(1)cos(x)
 f(2) = p(2)sin(x)
 f(3) = p(3)x
 end{expression}
 %
```



Two dimensional families of curves

Users may need to plot families of curves indexed by parameters. For instance, diffusion of a unit mass of substance from an instantaneous plane source is described by the equation

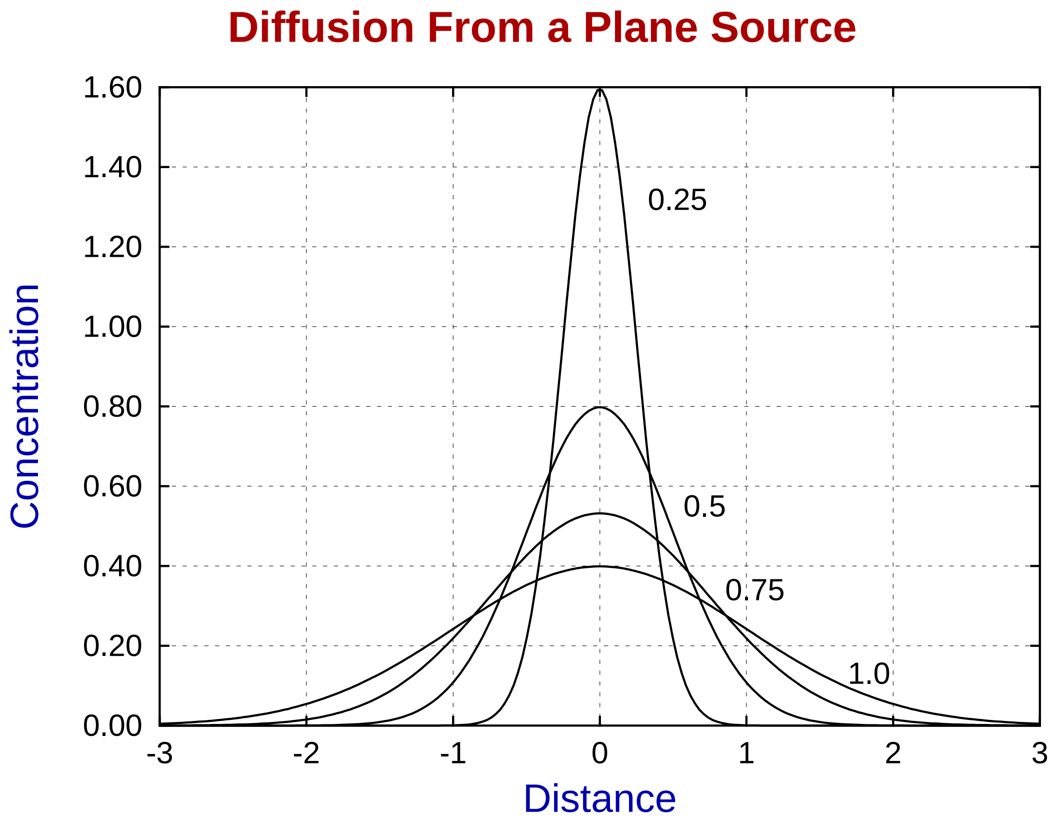
$$f(x) = \frac{1}{2\sqrt{\pi Dt}} \exp\left(\frac{-x^2}{4Dt}\right)$$

which is, of course, a normal distribution with $\mu = 0$ and $\sigma^2 = 2Dt$, where D is the diffusion constant and t is time, so that $2Dt$ is the mean square distance diffused by molecules in time t. Now it is easy to plot the concentration $f(x)$ predicted by this equation as a function of distance x and time t given a diffusion constant D, by simulating the equation using **makdat**, saving the curves to a library file or project archive, then plotting the collected curves. However, there is a much better way using program **usermod** which has the important advantage that families of curves indexed by parameters can be plotted interactively. This is a more powerful technique which provides numerous advantages and convenient options when simulating systems to observe the behavior of the profiles as the indexing parameters vary.

The next figure shows the above equation plotted (in arbitrary units) using the model parameters

$$p_i = 2Dt_i, \text{ for } i = 1, 2, 3, 4$$

to display the diffusion profiles as a function of time. The plot was created using model file `family2d.mod`, which simply defines four identical equations corresponding to the diffusion equation but with four different parameters p_i. Program **usermod** was then used to read in the model, simulate it for the parameter values indicated, then plot the curves simultaneously.

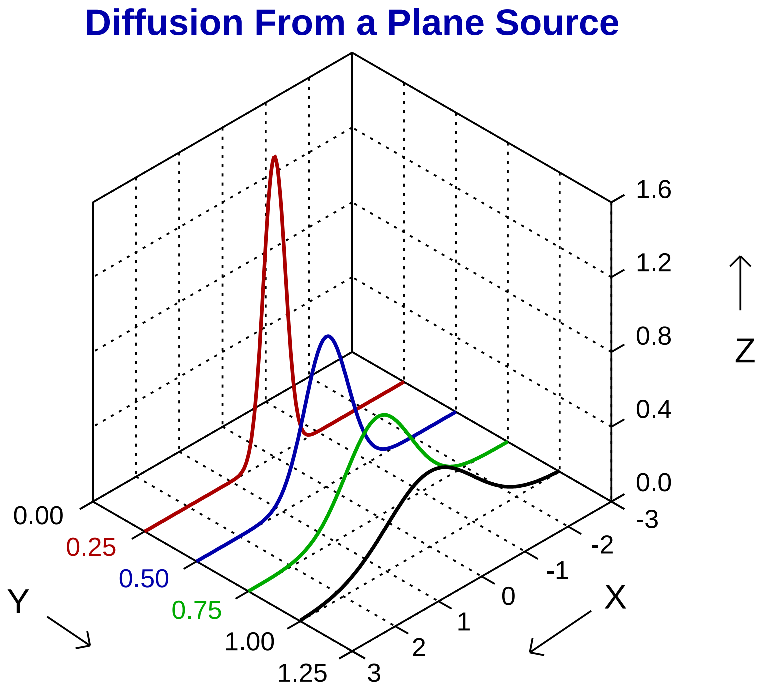


Three dimensional families of curves

Users may need to plot families of curves indexed by parameters in three dimensions. To show how this is done, the diffusion equation dealt with previously is reformulated, using $y = \sqrt{2Dt}$, as

$$z(x, y) = \frac{1}{y\sqrt{2\pi}} \exp\left\{-\frac{1}{2}\left(\frac{x}{y}\right)^2\right\}$$

and is plotted in the next figure for the same parameter values used before, but now as sections through the surface of a function of two variables.



This is, of course, a case of a family of parametric space curves projected onto the fixed values of y. Now the model file `family3d.mod` was used by program **usermod** to create this figure, using the option to plot n sets of parametric space curves, but you should observe a number of important facts about this model file before attempting to plot your own families of space curves.

- There are 4 curves so there are 12 functions of 1 variable
- Functions $f(1)$, $f(4)$, $f(7)$, $f(10)$ are the parameter t, i.e. x
- Functions $f(2)$, $f(5)$, $f(8)$, $f(11)$ are the y values, i.e. $\sqrt{2Dt}$
- Functions $f(3)$, $f(6)$, $f(9)$, $f(12)$ are the z values, i.e. the concentration profiles

Finally, it is clear that n space curves require a model file that specifies $3n$ equations, but you should also realize that space curves cannot be plotted if there is insufficient variation in any of the independent variables, e.g. if all $y = k$, for some fixed parameter k.

The model files for diffusion curves

The model file `family2d.mod` defines two–dimensional curves as follows.

```
%
Normal pdf with mu = 0, sigma = p(i), i = 1, 2, ..., 4
to be used by program usermod to plot a 2D family of curves
for diffusion from a plane source where p(i) = sqrt(2Dt).
Try p(i) = sqrt(2Dt) = i/4, with -3 =< x =< 3.
%
4 equations
1 variable
4 parameters
%
1
pi
2
multiply
squareroot
divide
put(1)
x
p(1)
divide
2
power
0.5
multiply
negative
exponential
get(1)
multiply
p(1)
divide
f(1)
x
p(2)
divide
2
power
0.5
multiply
negative
exponential
get(1)
multiply
p(2)
divide
f(2)
x
p(3)
divide
2
power
0.5
multiply
negative
exponential
get(1)
```

```
multiply
p(3)
divide
f(3)
x
p(4)
divide
2
power
0.5
multiply
negative
exponential
get(1)
multiply
p(4)
divide
f(4)
%
```

The model file `family3d.mod` defines three–dimensional curves as follows.

```
%
4 space curves (12 equations) for x(i),y(i),z(i) as f(t).
Defines normal pdfs mu(i) = 0, sigma(i) = p(i), i=1,2,3,4
for program usermod to plot a 3D family of curves showing
diffusion from a plane source, where p(i) = sqrt(2Dt).
Try p(i) = sqrt(2Dt) (e.g. = i/4), with -3 =< t =< 3.
%
12 equations
1 variable
4 parameters
%
1
pi
2
multiply
squareroot
divide
put(1)                          store 1/srt(2*pi)
x
f(1)                             x(t) = x
p(1)
f(2)                             y(t) = p(1)
x
p(1)
divide
2
power
0.5
multiply
negative
exponential
get(1)
multiply
p(1)
divide
f(3)                             z(t) = function value
x
f(4)                             x(t) = x
```

```
p(2)
f(5)                           y(t) = p(2)
x
p(2)
divide
2
power
0.5
multiply
negative
exponential
get(1)
multiply
p(2)
divide
f(6)                           z(t) = function value
x
f(7)                           x(t) = x
p(3)
f(8)                           y(t) = p(3)
x
p(3)
divide
2
power
0.5
multiply
negative
exponential
get(1)
multiply
p(3)
divide
f(9)                           z(t) = function value
x
f(10)                          x(t) = x
p(4)
f(11)                          y(t) = p(4)
x
p(4)
divide
2
power
0.5
multiply
negative
exponential
get(1)
multiply
p(4)
divide
f(12)                           z(t) = function value
%
```