Drawing ellipses in MacAnova

There are several ways to draw ellipses in MacAnova.

The defining equation for an ellipse centered at \( \mathbf{x}_0 = [x_{10}, x_{20}]' \) and with shape matrix \( \mathbf{Q} \)

\[
\mathbf{Q} = \begin{bmatrix} q_{11} & q_{12} \\ q_{12} & q_{22} \end{bmatrix}
\]

is

\[
q_{11}(x_1 - x_{10})^2 + 2q_{12}(x_1 - x_{10})(x_2 - x_{20}) + q_{22}(x_2 - x_{20})^2 = K^2
\]

where \( q_{jk} \) are the elements of \( \mathbf{Q}^{-1} \).

When you solve for \( x_2 \) in terms of \( x_1 \), you get the following equation:

\[
x_2 = x_{20} - q_{12}(x_1 - x_{10})/q_{22} \pm \left\{ \frac{K^2}{q_{22}} - (q_{11}q_{22} - (q_{12})^2)(x_1 - x_{10})^2/(q_{22})^2 \right\}^{1/2}
\]

The + and – signs go with the top and bottom halves of the ellipse, respectively.

This works only when the expression in \( \{ \ldots \} \) is nonnegative. This us the case only when \( x_{10} - K\sqrt{q_{11}} \leq x_1 \leq x_{10} + K\sqrt{q_{11}} \), that is \( |x_1-x_{10}| \leq K\sqrt{q_{11}} \). When \( |x_1-x_{10}| > K\sqrt{q_{11}} \), no real number \( x_2 \) satisfies the equation.

Similarly, you can express \( x_1 \) in terms of \( x_2 \) by

\[
x_1 = x_{10} - q_{12}(x_2 - x_{20})/q_{11} \pm \left\{ \frac{K^2}{q_{11}} - (q_{11}q_{22} - (q_{12})^2)/(q_{11})^2 \right\}^{1/2}
\]

For this to be meaningful, \( x_2 \) must satisfy \( x_{20} - K\sqrt{q_{22}} \leq x_2 \leq x_{20} + K\sqrt{q_{22}} \), that is \( |x_2-x_{20}| \leq K\sqrt{q_{22}} \).

Here I apply this in MacAnova:

```r
Cmd> Q <- matrix(vector(25,10.5,10.5,9),2); Q
          (1,1)  25  10.5 Positive definite symmetric matrix
          (2,1)  10.5   9
Cmd> x0 <- vector(30,40) # center
Cmd> K <- sqrt(invchi(.95, 2)); K # constant defining size
(1) 2.4477
Cmd> Qinv <- solve(Q)
Cmd> x1min <- x0[1] - K*sqrt(Q[1,1]) # minimum possible value for x1
Cmd> x1max <- x0[1] + K*sqrt(Q[1,1]) # maximum possible value for x1
```
Drawing ellipses in MacAnova

Cmd> `x1 <- x1min + (x1max - x1min) * run(0,100)/100` # values for x1

x1 now contains 101 equally spaced values from x1min to x1max. Now I computed x2 values for the top and bottom of the ellipse.

Cmd> `x2top <- x0[2] - Qinv[1,2] * (x1 - x0[1]) / Qinv[2,2] + \sqrt(K^2/Qinv[2,2] - Qinv[1,1]*Qinv[2,2] - Qinv[1,2]^2)*(x1 - x0[1])^2/Qinv[2,2]^2)`

Cmd> `x2bottom <- x0[2] - Qinv[1,2] * (x1 - x0[1]) / Qinv[2,2] - \sqrt(K^2/Qinv[2,2] - Qinv[1,1]*Qinv[2,2] - Qinv[1,2]^2)*(x1 - x0[1])^2/Qinv[2,2]^2)`

Vectors x2top and x2bottom now contain the x2 coordinates for the top and bottom halves of the ellipse.

Cmd> `lineplot(x1,x2top,show:F)` # draw but don't show top half
Cmd> `addlines(x1,x2bottom,ymin:?,title:"Ellipse",xlab:"x1",ylab:"x2")`
Drawing ellipses in MacAnova

It's much easier to do this using macro ellipse.

Cmd> help(ellipse:vector("usage","plotting_ellipse"))
Subtopic 'usage' of help on 'ellipse'
You can use ellipse() to compute and optionally draw an ellipse with shape defined by a specified positive definite matrix and centered at a specified point

ellipse(K, Q [,x0] [,graphics keywords]) computes xvals and yvals, the x- and y-coordinates of points on the ellipse defined by the equation

\[(x - x0)' \%*% solve(Q) \%*% (x - x0) = K^2\]

The value returned is structure(x:xvals,y:yvals [,graphics keywords]).

K > 0 must be a REAL scalar and Q must be a 2 by 2 REAL positive definite symmetric matrix. If x0 is an argument, it must be a REAL vector of length 2. Otherwise, rep(0,2) is used for x0.

Subtopic 'plotting_ellipse' of help on 'ellipse'
The ellipse can be plotted by

Cmd> result <- ellipse(K, Q [,x0] [,graphics keywords])
Cmd> lineplot(keys:result)

ellipse(K, Q [,x0], draw:T [,graphics keywords]) draws the ellipse directly and doesn't return the coordinates as a value. If the ellipse is to be added to an existing graph, include add:T as an argument.

So let's use ellipse():

Cmd> coords <- ellipse(K,Q,x0) # compute coordinates for ellipse
coords is a structure with two components, x and y:

Cmd> compnames(coords) # coords includes both x and y
(1) "x"
(2) "y"
You can actually just use `coords` itself instead of its x and y components separately. The following produces the identical plot.

```
Cmd> lineplot(coords,title:"Ellipse",xlab:"x1",ylab:"x2")
```

With keyword phrase `draw:T` and other graphics keyword phrases, `ellipse()` will draw the ellipse itself. The following single command will reproduce the preceding plot:

```
Cmd> ellipse(K,Q,x0,title:"Ellipse",xlab:"x1",ylab:"x2",draw:T)
```

By including `add:T`, you can add the ellipse to an existing plot. In the following lines, I generated and made a scatter plot of a bivariate normal sample with $\mu = x_0$ and $\Sigma = Q$, and then drew the following contour of the normal distribution:

$$
\sigma^{11}(x_1 - \mu_1)^2 + 2\sigma^{12}(x_1 - \mu_1)(x_2 - \mu_2) + \sigma^{22}(x_2 - \mu_2)^2 = K^2
$$

where $\sigma^{ij}$ are elements of $\Sigma^{-1}$, and $K^2 = \chi^2_{2(.95)} = 5.991$ is a probability point of $\chi^2$ on 2 degrees of freedom. With this value of $K$, the contour encloses 95% of the population.
Drawing ellipses in MacAnova

Cmd> sigma <- Q; mu <- x0
Cmd> n <- 100; y <- rmvnorm(n, sigma, mu)  # N_2(mu,sigma) sample

rmvnorm(n, sigma, mu) computes a multivariate normal sample of size n, variance matrix sigma and mean vector mu. Look at the help for full details.

Cmd> covar(y)  # sample mean vector and variance matrix
WARNING: searching for unrecognized macro covar near covar(
component: n
(1)         100
component: mean
(1,1)      29.888      39.957  Pretty close to \( \mu \)
component: covariance
(1,1)      24.066      9.974  Pretty close to \( \Sigma \)
(2,1)       9.974      8.4978

Cmd> plot(y[,1],y[,2],symbol:"\1",title:"Bivariate normal sample",
xlab:"y1",ylab:"y2",show:F)  # make scatter plot but don't show it
Cmd> ellipse(K,sigma,mu,draw:T,add:T,xmin:?,xmax:?,ymin:?,ymax:?,
title:"Bivariate normal sample and contour",xlab:"x1",ylab:"x2")

Keyword phrases xmin:?,xmax:?,ymin:? and ymax:? ensure the limits of the graph includes all the points and the contour. Without them, you might find some of the the contour was cut off by the frame.