restart;

Goal: contour plots of a function of 2 variables, *with the contours labeled on the plot.*

with(*Statistics*) :
with(*LinearAlgebra*) :
with(*ArrayTools*) :
with(*plots*) :

Raw data snippet:

$$chart1 := \begin{bmatrix}
10 & 0.1019 & 10383.58572 & 1.655 & 14.71813365 & 0.998691616 \\
11 & 9.07E-02 & 8226.470762 & 1.803 & 12.90034311 & 1.260564864 \\
12 & 8.08E-02 & 6528.624733 & 1.95 & 11.28046011 & 1.588389657 \\
13 & 7.20E-02 & 5183.987877 & 2.075 & 9.788142028 & 2.000390481 \\
14 & 6.41E-02 & 4106.236798 & 2.2 & 8.462283333 & 2.525426689 \\
15 & 5.71E-02 & 3260.402376 & 2.185 & 7.093692368 & 3.180589021 \\
16 & 5.08E-02 & 2582.66636 & 2.17 & 5.935732561 & 4.015230214 \\
17 & 4.53E-02 & 2052.085201 & 2.455 & 5.313314448 & 5.053396416 \\
18 & 4.03E-02 & 1624.086202 & 2.74 & 4.710047837 & 6.385129058 \\
19 & 3.59E-02 & 1288.806986 & 2.595 & 3.853921119 & 8.046200954 \\
20 & 3.20E-02 & 1021.439211 & 2.45 & 3.145470855 & 10.1523418 \\
21 & 2.85E-02 & 812.2481006 & 2.8 & 2.831644788 & 12.76703509 \\
22 & 2.54E-02 & 642.6209972 & 3.15 & 2.519501897 & 16.13703886 \\
23 & 2.26E-02 & 510.7588056 & 3.325 & 2.178968494 & 20.30312525 \\
24 & 2.01E-02 & 404.0090552 & 3.5 & 1.875083342 & 25.66774152 \\
25 & 1.79E-02 & 320.4092507 & 3.325 & 1.535917323 & 32.36485831 \\
26 & 1.59E-02 & 254.0830058 & 3.15 & 1.25626199 & 40.81343405 \\
27 & 1.42E-02 & 201.6395285 & 3.325 & 1.085227546 & 51.42840829 \\
28 & 1.26E-02 & 159.7692264 & 3.5 & 0.935076409 & 64.90611637 \\
29 & 1.13E-02 & 127.6897014 & 3.325 & 0.770381699 & 81.21250097 \\
30 & 1.00E-02 & 100.6006647 & 3.15 & 0.627045628 & 103.0808298 \\
31 & 8.90E-03 & 79.20981477 & 3.15 & 0.52412244 & 130.9181195 \\
32 & 7.95E-03 & 63.2023522 & 3.15 & 0.442484882 & 164.0761718 \\
33 & 7.10E-03 & 50.40988212 & 3.15 & 0.373452411 & 205.7136332 \\
34 & 6.30E-03 & 39.68990718 & 3.15 & 0.312146571 & 261.2754913 \\
35 & 5.60E-03 & 31.35992666 & 3.15 & 0.261595208 & 330.6767938 \\
36 & 5.00E-03 & 24.99994154 & 3.15 & 0.220700212 & 414.80097 \\
37 & 4.50E-03 & 20.24995265 & 3.15 & 0.188437145 & 512.0999629 \\
38 & 3.96E-03 & 15.68156333 & 3.15 & 0.155557345 & 661.2861091
\end{bmatrix} :$$

```
#This is the procedure that generates the output I need to display as a contour plot, with contour labels on the plot

procedure1:=proc(cur,len,Vdrop,chartvar);
     local Rpf,n_r,n_tr,n;
     Rpf:=Vdrop/(cur*len);
     n_r:=1:
     n_tr:=1;
     for n from 1 to RowDimension(chartvar)
     do
          if Rpf > chartvar[n,6]
          then
               n_r:=n;
          end if;
          if cur < chartvar[n,5]
          then
               n_tr:=n;
          end if;
     end do;
     chartvar[min(n_r,n_tr),1];
end proc:
```

Try a few values to make sure the procedure is returning the expected results-

$$procedure1(12, 3 \cdot 3.2808, 250, chart1) = 11$$
$$procedure1(2.5, 2 \cdot 3.2808, 250, chart1) = 21$$

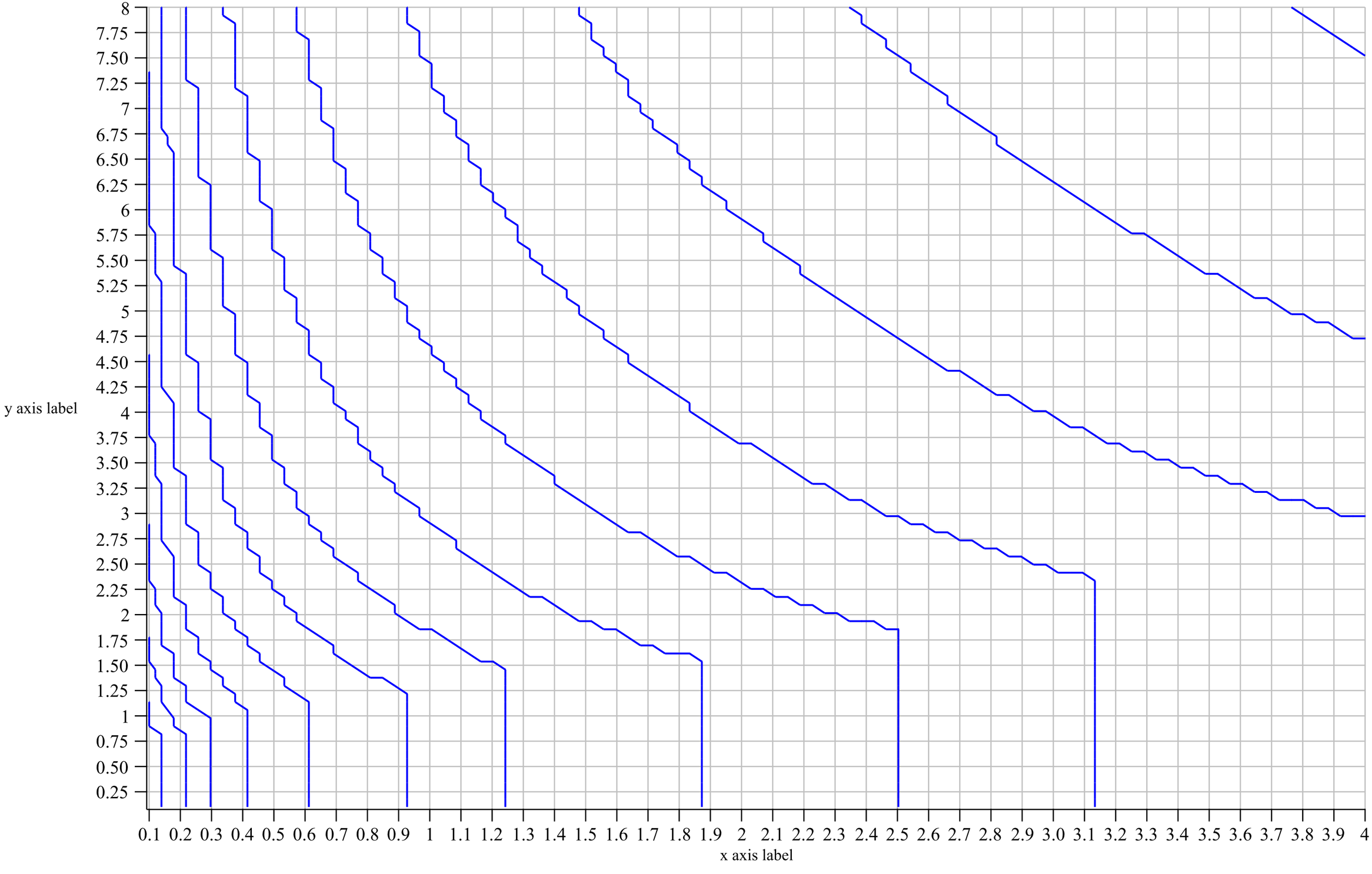
Results are as expected.

Make a second function of just the first two variables, that calls the original procedure, specifically for contour plotting. Add the ".0001" to help contour plot function converge on just a single line rather than wide swatches of equal values

$$procedure11 := (cur, len) \rightarrow procedure1(cur, len * 3.2808, 250, chart1) + .0001 :$$

Contour plot below (for fast rendering, until ready to do high quality print, make the "grid" values small, so result is chunky-looking here).

$$contourplot(procedure11, .1..4, .1..8, size = [1, \text{"golden"}], color = \text{"blue"}, axis = [gridlines = [30, color = grey]], labels = [\text{"x axis label"}, \text{"y axis label"}], grid = [100, 100], contours = [10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38])$$


y axis label
8
7.75
7.50
7.25
7
6.75
6.50
6.25
6
5.75
5.50
5.25
5
4.75
4.50
4.25
4
3.75
3.50
3.25
3
2.75
2.50
2.25
2
1.75
1.50
1.25
1
0.75
0.50
0.25
0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 3.0 3.1 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4
x axis label

I need to see the contour values but unfortunately **there seems to be no way to display the contour values on the plot** without hovering over individual contour lines from within the Maple app. This of course won't work for any type of file output like PDFs, for example.

I found a clever program on a Maple forum to create contour plots with the contour values displayed. https://www.mapleprimes.com/posts/202222-Contour-Curves-With-Labels.

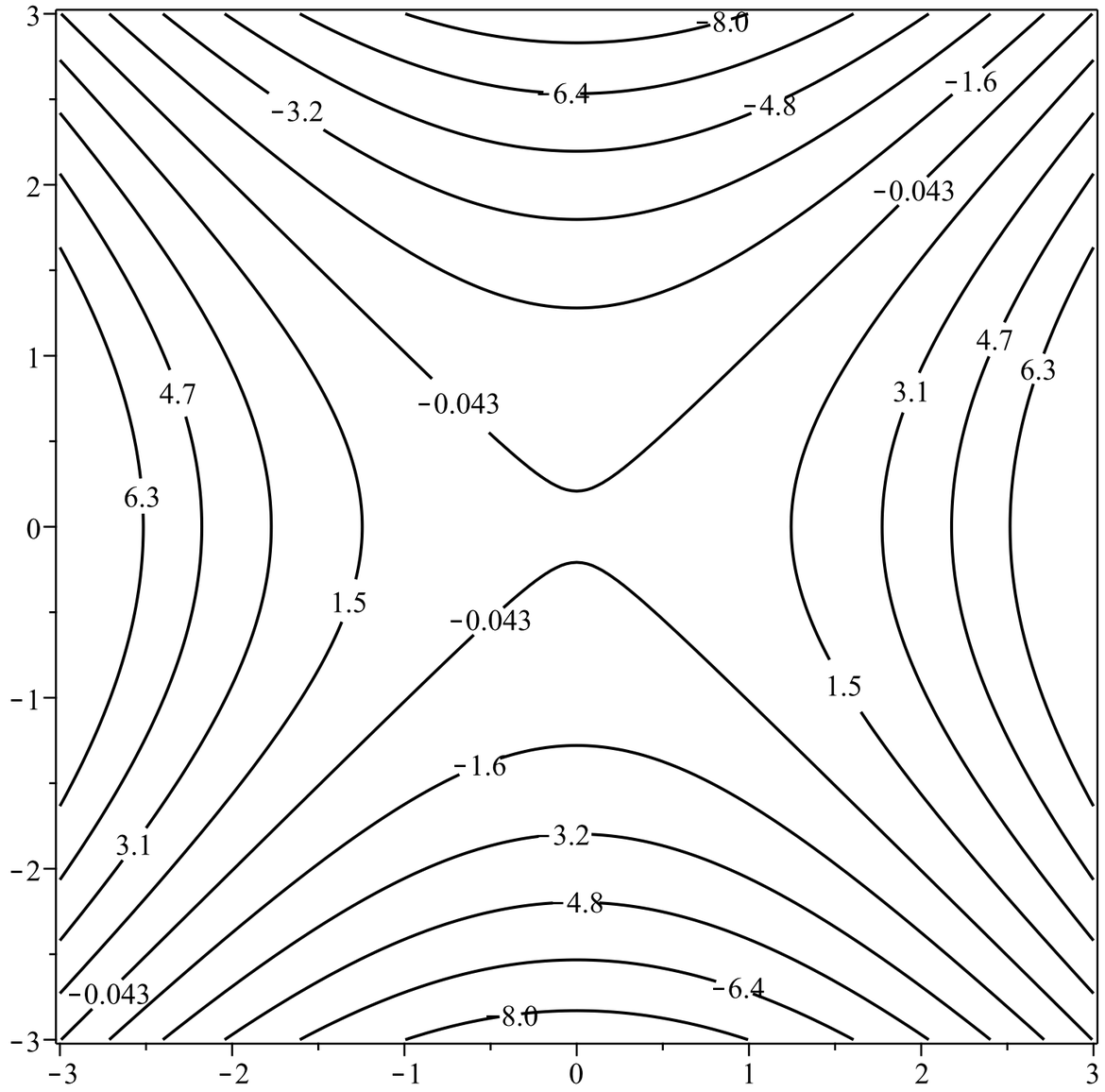
I am not very familiar with Maple programming at this point, so much of what is done here I don't really understand well.

```
ContoursWithLabels := proc(Expr, Range1::(range(realcons)), Range2::(range(realcons)), Number::posint := 8, S::(set(realcons)) := { }, GraphicOptions::list := [color = black, axes = box, size = [.5, 1]], Coloring::`=` := NULL)
    local r1, r2, L, f, L1, h, S1, P, P1, r, M, C, T, p, p1, m, n, A, B, E;
    uses plots, plottools;
        f := unapply(Expr, x, y);
        if S = { } then
            r1 := rand(convert(Range1, float));
            r2 := rand(convert(Range2, float));
            L := [seq([r1( ), r2( )], i = 1 ..205)];
            L1 := convert(sort(select(a → type(a, realcons), [seq(f(op(t)), t = L)]), (a, b) → is(abs(a) < abs(b))), set);
            h := (L1[ − 6] − L1[1]) / Number;
            S1 := [seq(L1[1] + 1 * h / 2 + h * (n − 1), n = 1 ..Number)];
        else
            S1 := convert(S, list);
        fi;
        print(Contours = evalf[2](S1));
        r := k → rand(20 ..k − 20);
        M := [ ];
        T := [ ];
        for C in S1 do
            P := implicitplot(Expr = C, x = Range1, y = Range2, op(GraphicOptions), gridrefine = 3);
            P1 := [getdata(P)];
            for p in P1 do
                p1 := convert(p[3], listlist);
                n := nops(p1);
                if n < 500 then
                    m := `if`(40 < n, r(n)( ), round(n / 2));
                    M := `if`(40 < n, [op(M), p1[1 ..m − 11], p1[m + 11 ..n]], [op(M), p1]);
                    T := [op(T), [op(p1[m]), evalf[2](C)]]
                else
                    if 500 <= n then
                        h := floor(n / 2);
                        m := r(h)( );
                        M := [op(M), p1[1 ..m − 11], p1[m + 11 ..m + h − 11], p1[m + h + 11 ..n]];
                        T := [op(T), [op(p1[m]), evalf[2](C)], [op(p1[m + h]), evalf[2](C)]]
                    fi; fi;
                od; od;
                A := plot(M, op(GraphicOptions));
                B := textplot(T);
                if Coloring = NULL then
                    E := NULL
                else
                    E := densityplot(Expr, x = Range1, y = Range2, op(rhs(Coloring)))
                fi;
                display(E, A, B);
            end proc:
```

Below is an example use case of the program. It generates a render that illustrates exactly what I am trying to do- i.e. show the contour values on the plot.

ContoursWithLabels$(x^2 - y^2, -3..3, -3..3, 10)$;

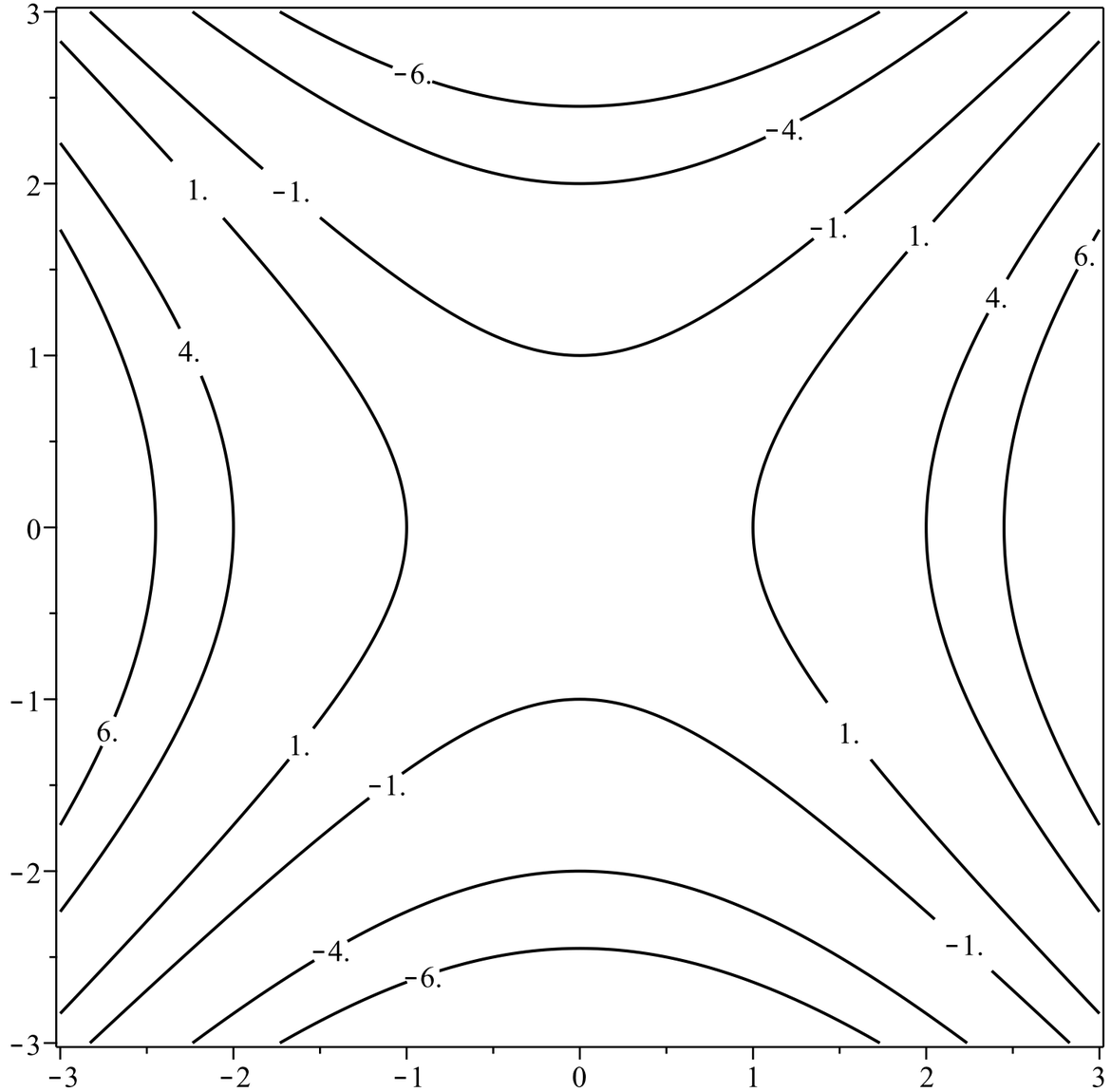
Contours = [−8.0, −6.4, −4.8, −3.2, −1.6, −0.043, 1.5, 3.1, 4.7, 6.3]



Or pass it a set of specific contours to use

ContoursWithLabels$(x^2 - y^2, -3..3, -3..3, \{-1, -4, -6, 1, 4, 6\})$;

Contours = [−6., −4., −1., 1., 4., 6.]



However this won't work with my procedure passed as the expression (below). **This is what I need help getting to work-**

ContoursWithLabels(*procedure11*, .1 ..4, .1 ..4)
Error, (in ContoursWithLabels) invalid subscript selector