Statistics 5401
29. More than two groups

Gary W. Oehlert
School of Statistics
313B Ford Hall
612-625-1557
gary@stat.umn.edu

We are still doing classification, but now we have $g \geq 2$ populations. The basic ideas and approach are the same as for two groups, but the details are a little more complicated.
We will again construct classifiers to minimize the expected cost, or, when the costs are equal, minimize the total probability of misclassification.
Each population has a prior probability p_i and a density $f_i()$ for the covariates.
Think of the density $f_i()$ as the "probability" of x given the population i, and the prior p_i as the marginal probability of population i.
We need to compute $P(i|x)$, the posterior probability of π_i given data x. Use Bayes' theorem

$$P(i|x) = \frac{P(\pi_i \ \& \ x)}{P(x)} = \frac{f_i(x)\ p_i}{\sum_{j=1}^{g} P(j \ \& \ x)} = \frac{f_i(x)\ p_i}{\sum_{j=1}^{g} f_j(x)\ p_j}$$

This gives us the posterior in terms of quantities we "know".
We also need costs. Let $c(i|j)$ be the cost of classifying an object from π_j into π_i.

$$\begin{bmatrix} c(1|1) & c(1|2) & c(1|3) & \dots & c(1|g) \\ c(2|1) & c(2|2) & c(2|3) & \dots & c(2|g) \\ c(3|1) & c(3|2) & c(3|3) & \dots & c(3|g) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c(g|1) & c(g|2) & c(g|3) & \dots & c(g|g) \end{bmatrix}$$

Typically $c(i|i) = 0$, and off-diagonal costs are positive.
Now we have an object with data x and we need to classify it. What is the cost? If the object is really from π_j and we classify into π_i, then we incur cost $c(i|j)$.
Of course, we don't know that the object came from π_j (if we did, we wouldn't be classifying!). About the best we can do is use $P(j|x)$ to get an expected cost of misclassification.
Let $ECM(i|x)$ be the expected cost of misclassification when we classify an object with data x into π_i.

$$\begin{aligned} ECM(i|x) &= c(i|1)P(1|x) + \ldots + c(i|g)P(g|x) \\ &\propto c(i|1)f_1(x)p_1 + \ldots + c(i|g)f_g(x)p_g \end{aligned}$$

So for any x, we classify into the population π_i that minimizes $ECM(i|x)$.
If the diagonal costs ($c(i|i)$) are zero and all other costs are equal, then minimizing expected cost chooses π_i to minimize

$$\sum_{j \neq i} f_j(x) p_j$$

which is the same thing as choosing π_i to maximize

$$f_i(x)p_i \text{ or } \ln(f_i(x)) + \ln(p_i)$$

That is, when costs are equal, minimizing cost is the same as minimizing total probability of misclassification, so we choose to classify into the population with highest posterior probability.
Normal populations with equal variances.
In this case

$$f_i(x) = \frac{1}{\sqrt{(2\pi)^p|\Sigma|}} e^{-\frac{1}{2}(x-\mu_i)'\Sigma^{-1}(x-\mu_i)}$$

so that

$$\begin{aligned} ECM(i|x) &\propto c(i|1)f_1(x)p_1 + \ldots + c(i|g)f_g(x)p_g \\ &\propto \sum_{j=1}^{g} c(i|j)p_j e^{-\frac{1}{2}(x-\mu_j)'\Sigma^{-1}(x-\mu_j)} \end{aligned}$$

There's not a lot more simplification that we can do for general costs and priors.
Example: $p = 2$, $g = 3$, $\mu_1 = (0, 1)$, $\mu_2 = (.707, -.5)$, $\mu_3 = (-.707, -.5)$.

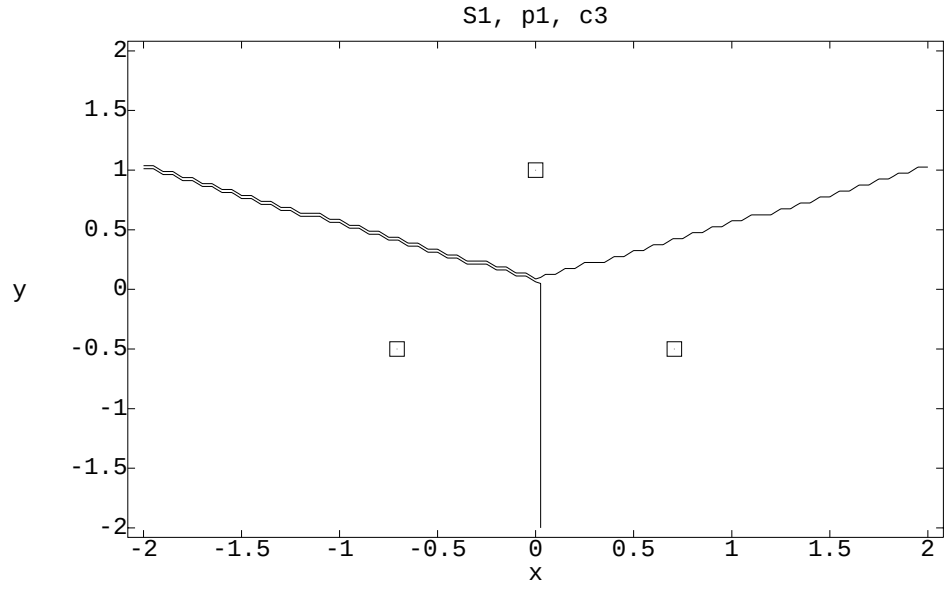
$$p \text{ vectors } \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}, \begin{bmatrix} 1/6 \\ 2/6 \\ 3/6 \end{bmatrix}, \begin{bmatrix} 3/6 \\ 2/6 \\ 1/6 \end{bmatrix}$$

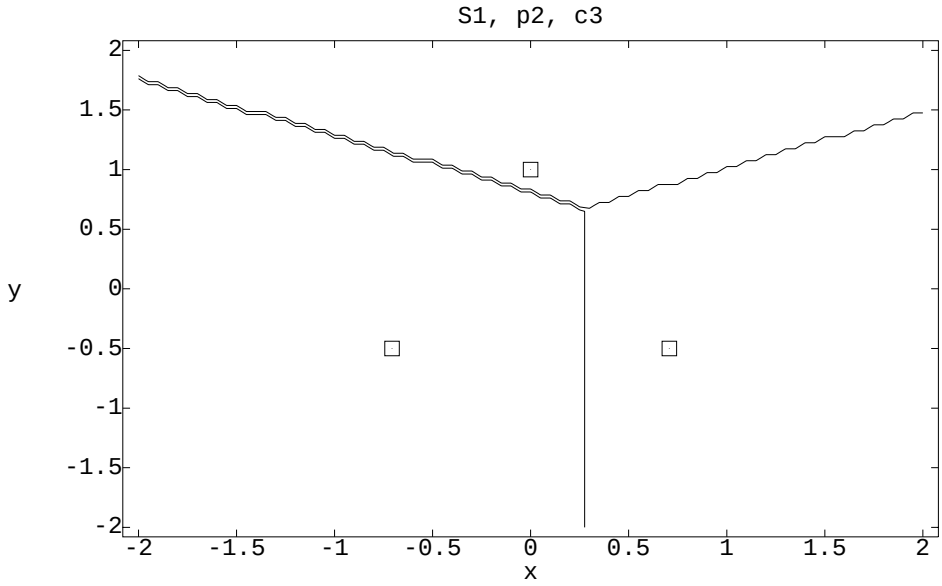
Costs

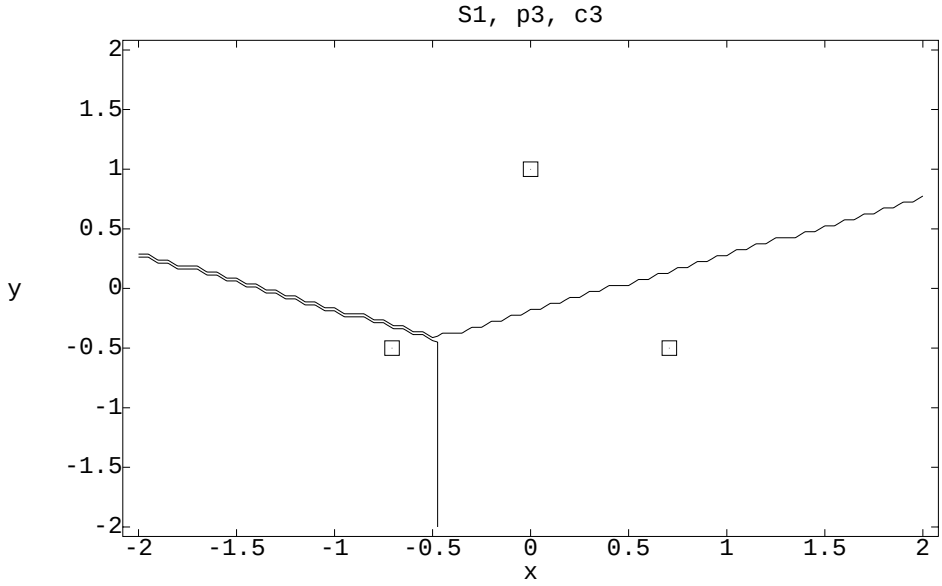
$$\begin{bmatrix} 0 & 2 & 3 \\ 1 & 0 & 3 \\ 1 & 2 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ 2 & 0 & 2 \\ 3 & 3 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

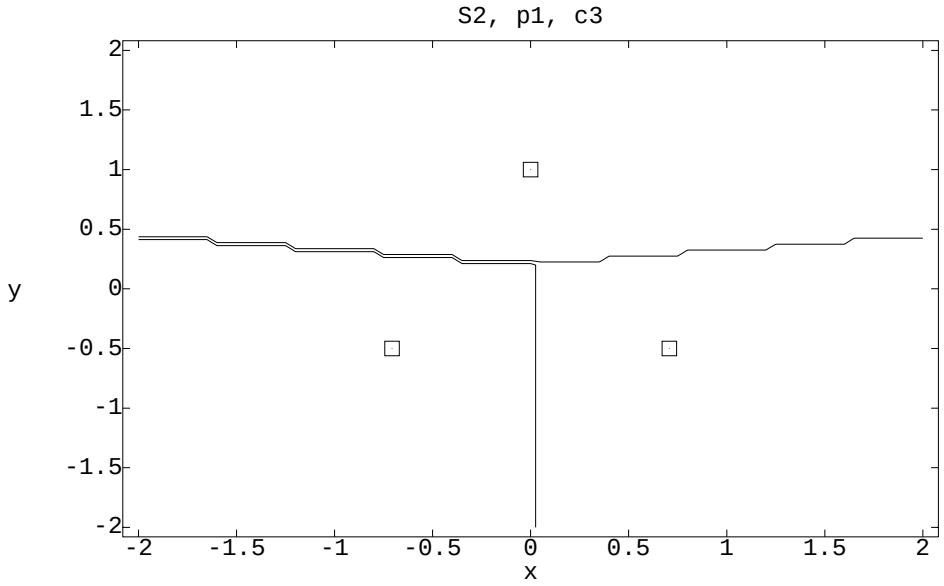
Σs

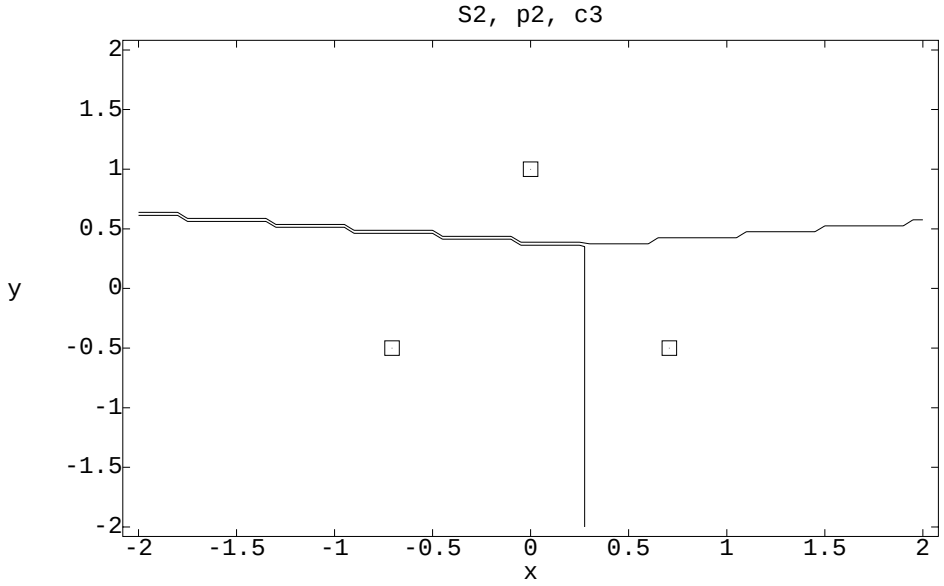
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & .25 \end{bmatrix}, \begin{bmatrix} 1 & .35 \\ .35 & .25 \end{bmatrix}$$

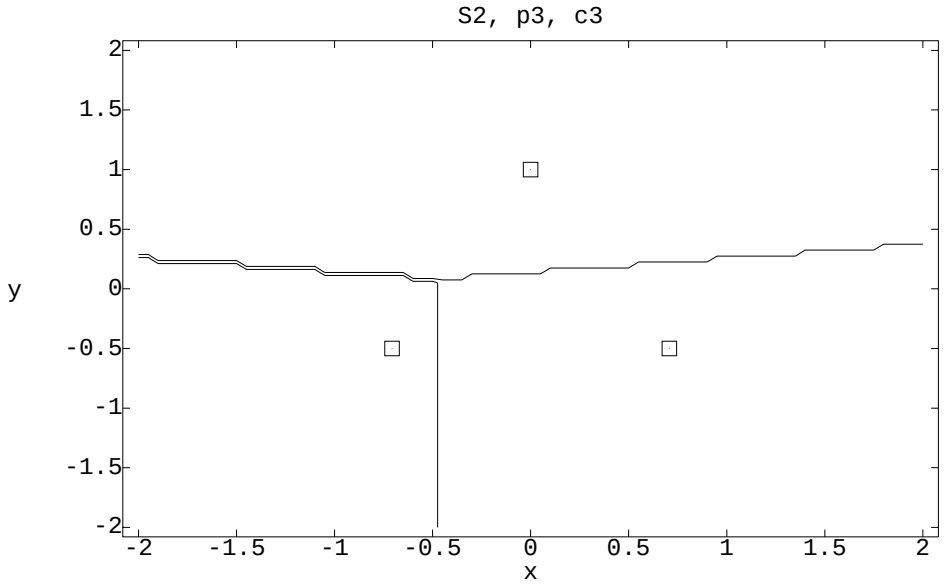


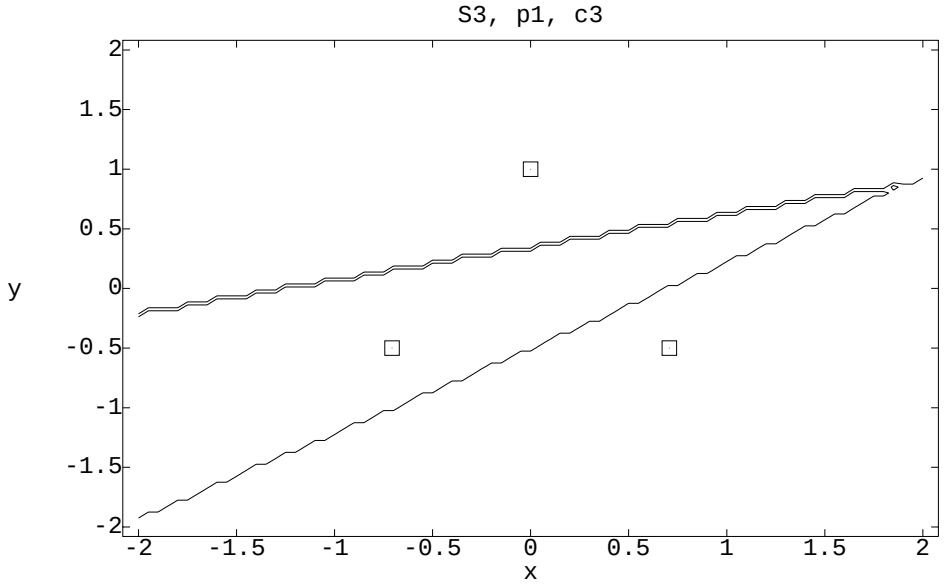

S1, p2, c3
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

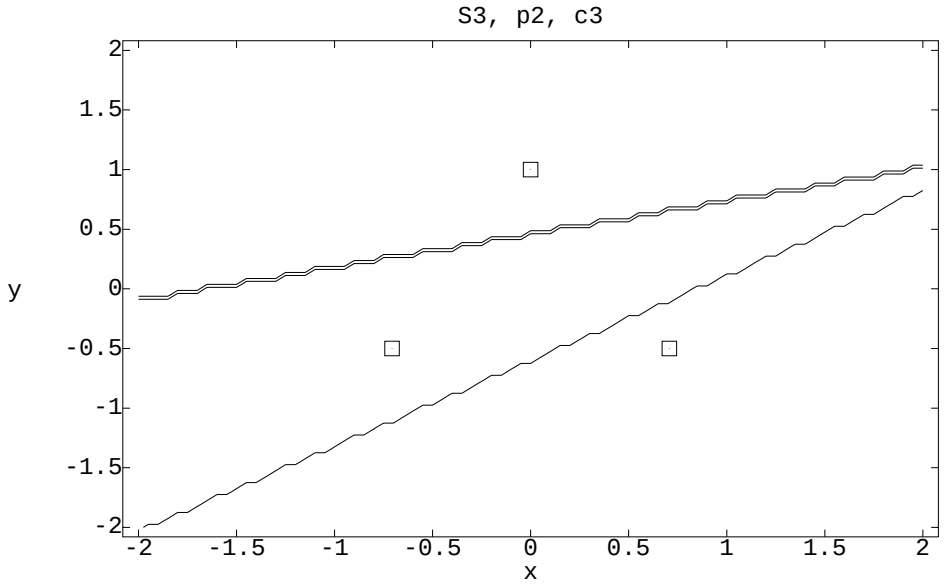

S1, p3, c3
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

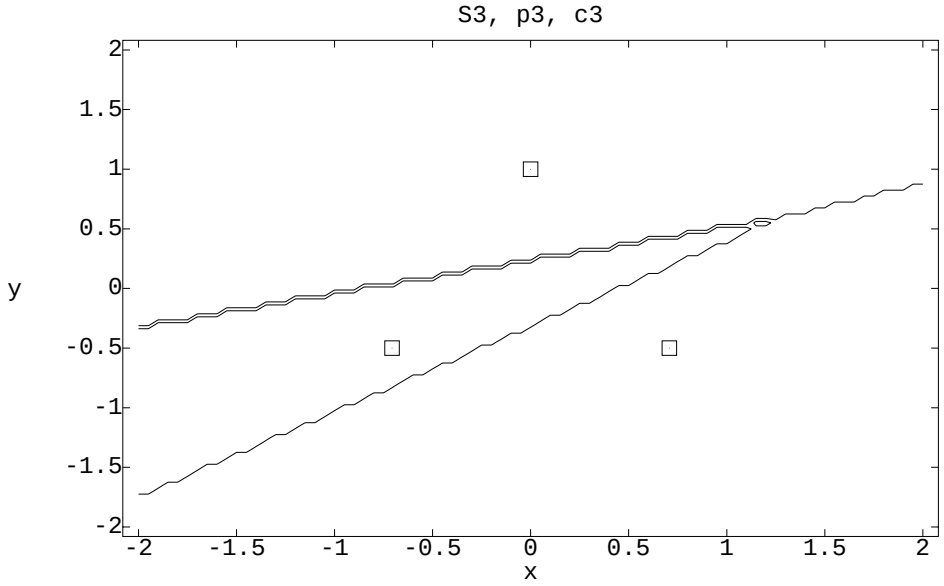

S2, p1, c3
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

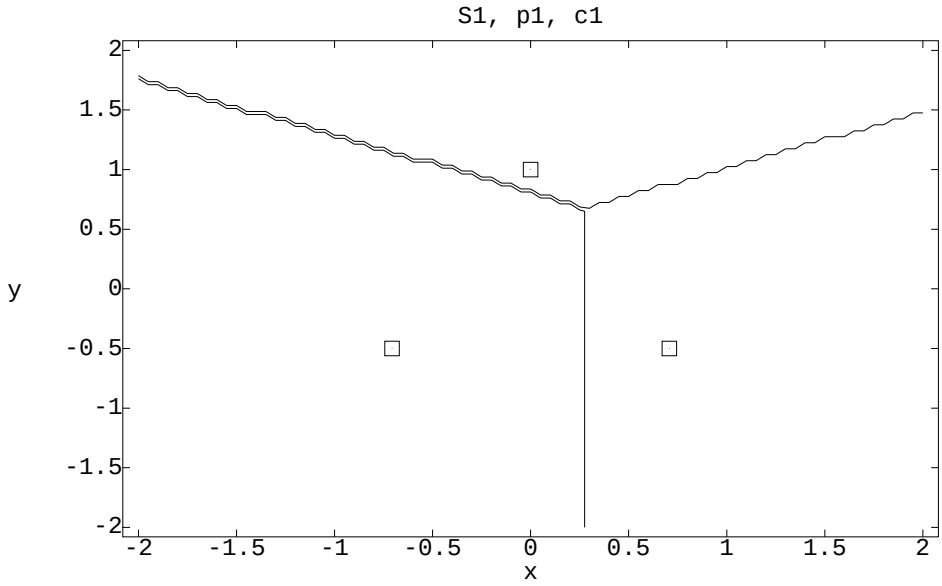

S2, p2, c3
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

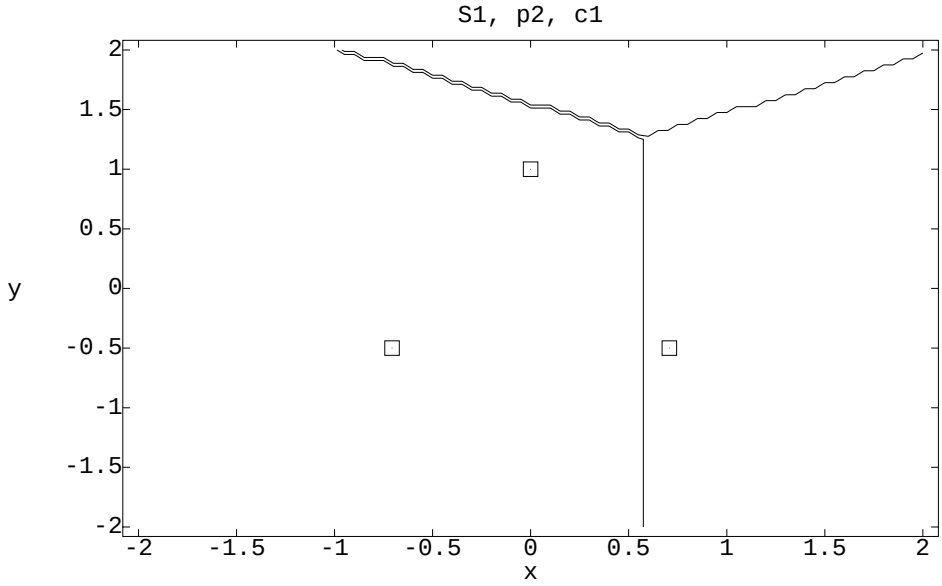

S2, p3, c3
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
y
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
x

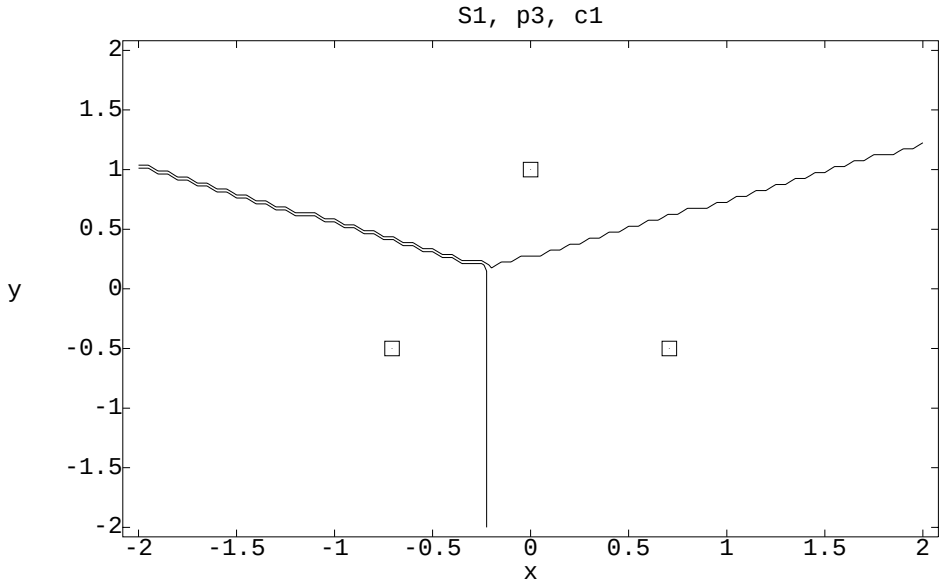

S3, p1, c3
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
y
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
x

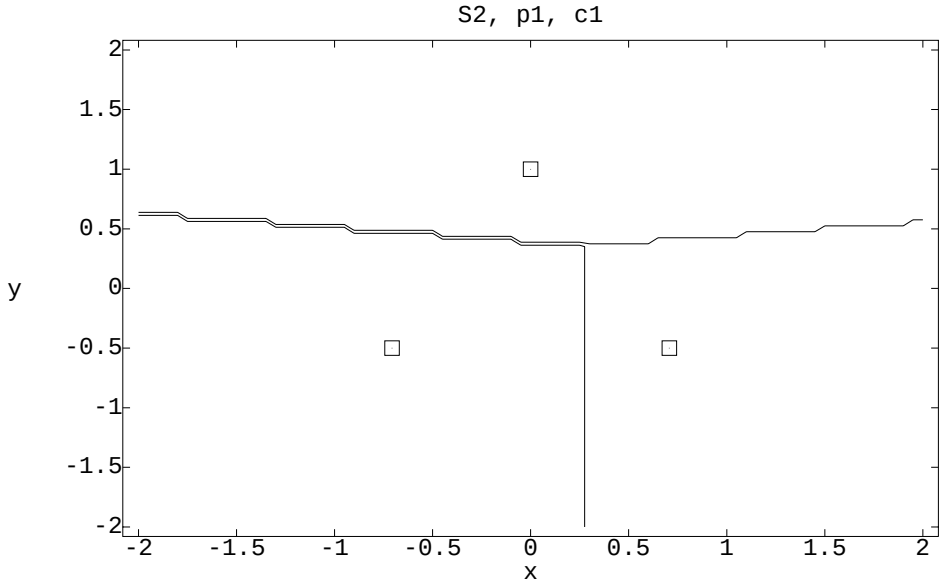

S3, p2, c3
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

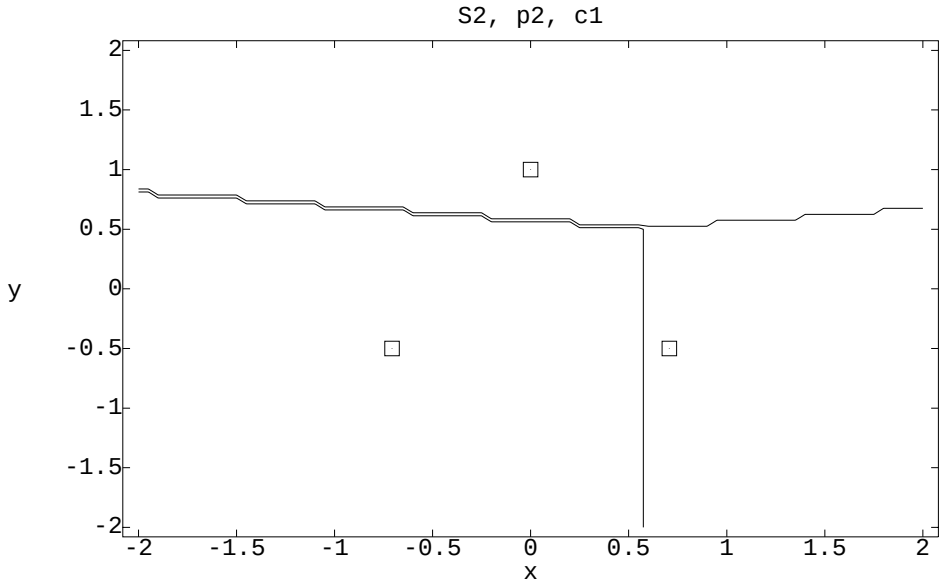

S3, p3, c3
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

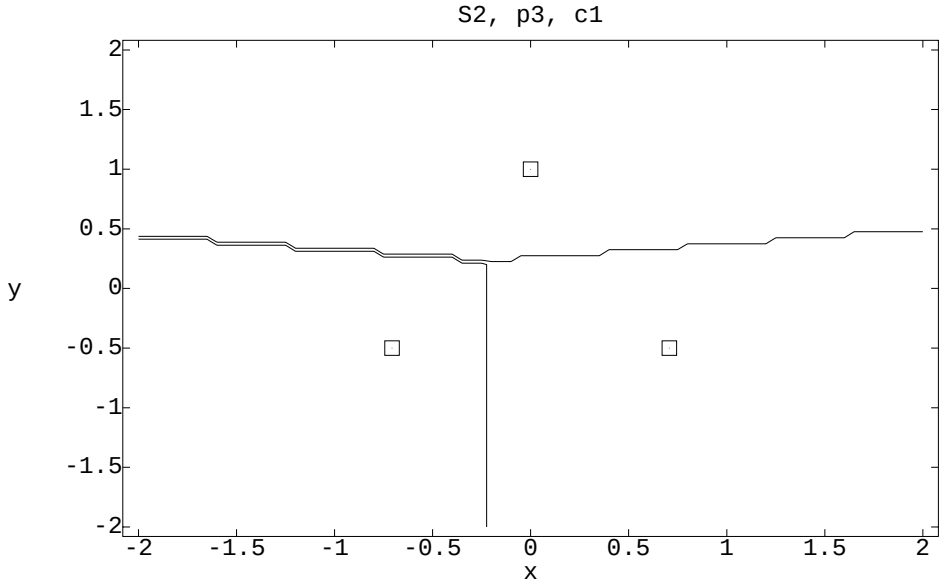

S1, p1, c1
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
y
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
x

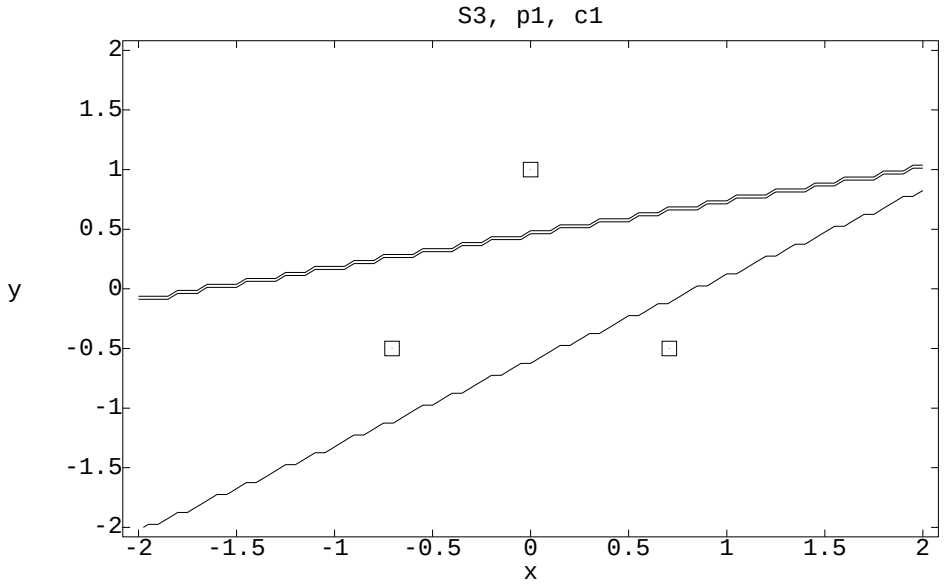

S1, p2, c1
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
y
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
x

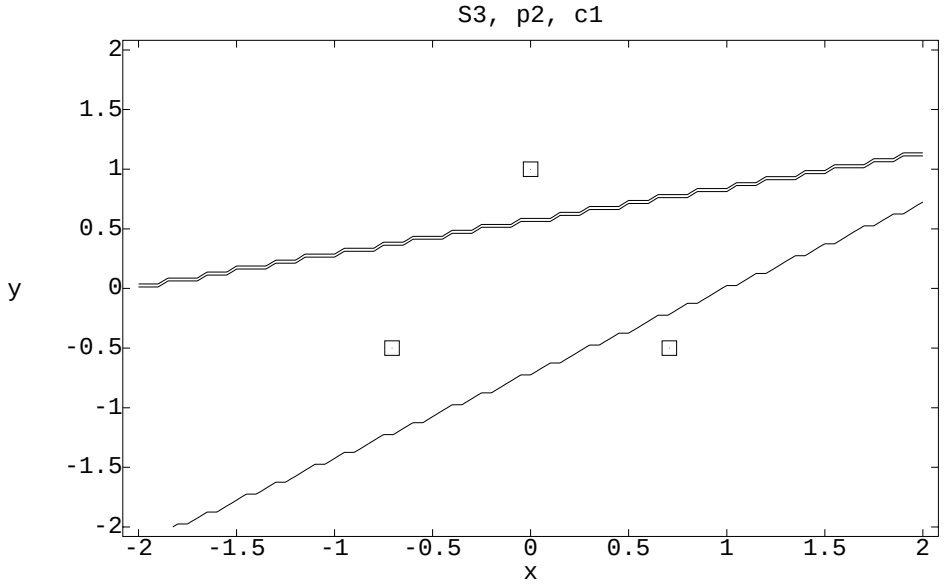

S1, p3, c1
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
y
-2 -1.5 -1 -0.5 0 0.5 1 1.5 2
x

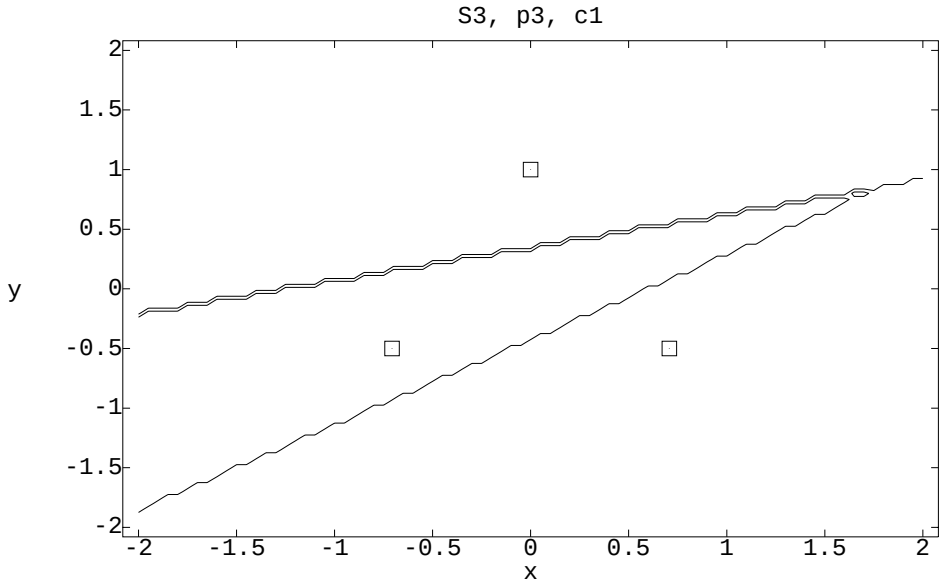

S2, p1, c1
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
y
-2 -1.5 -1 -0.5 0 0.5 1 1.5 2
x

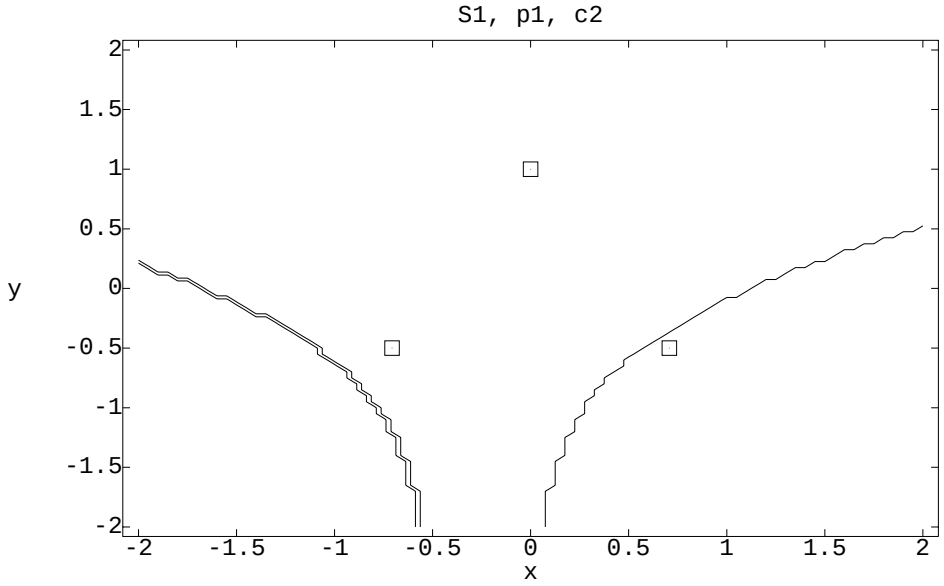

S2, p2, c1
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

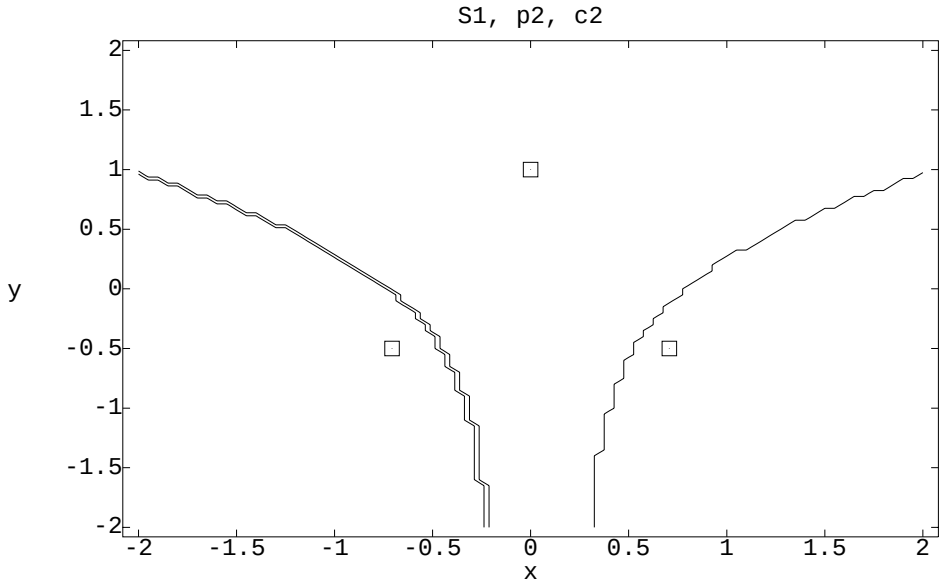

S2, p3, c1
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

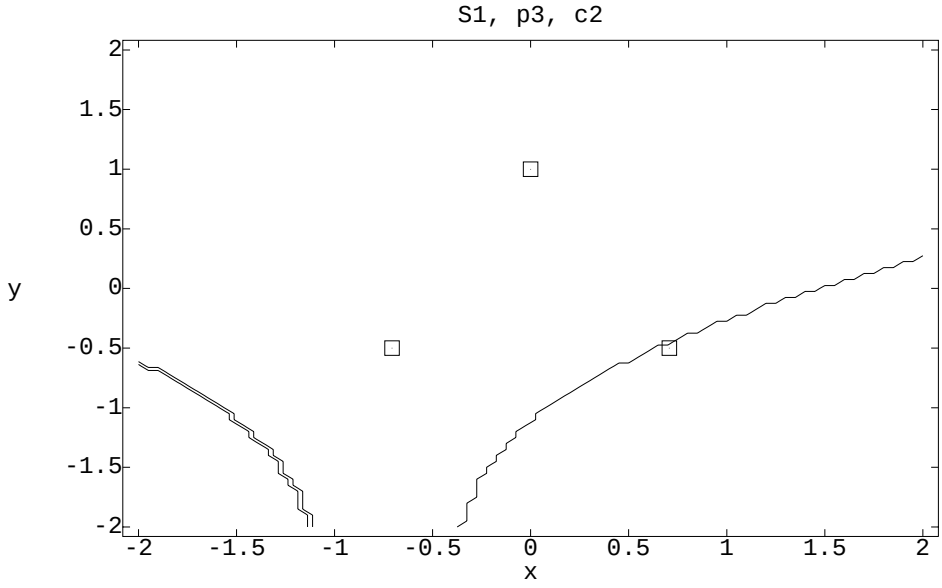

S3, p1, c1
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

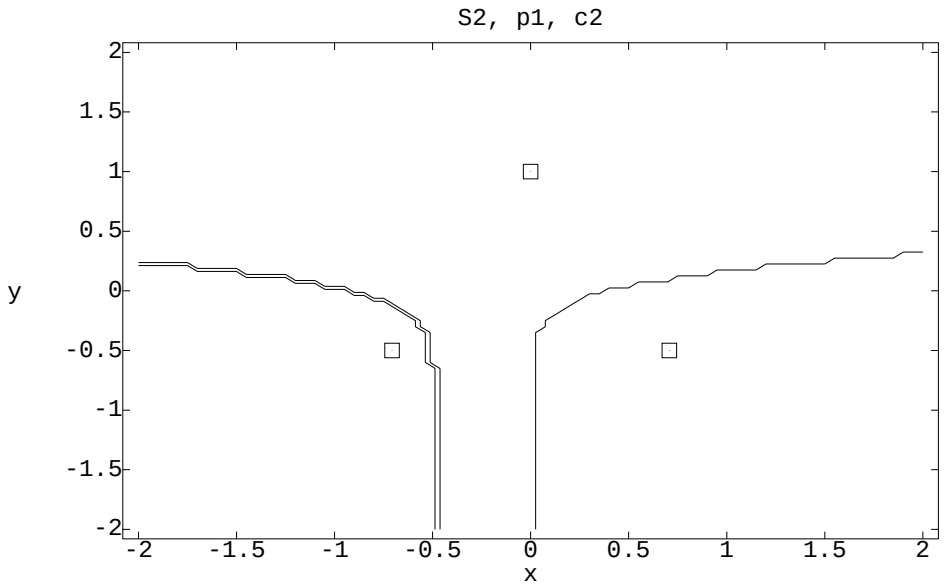

S3, p2, c1
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

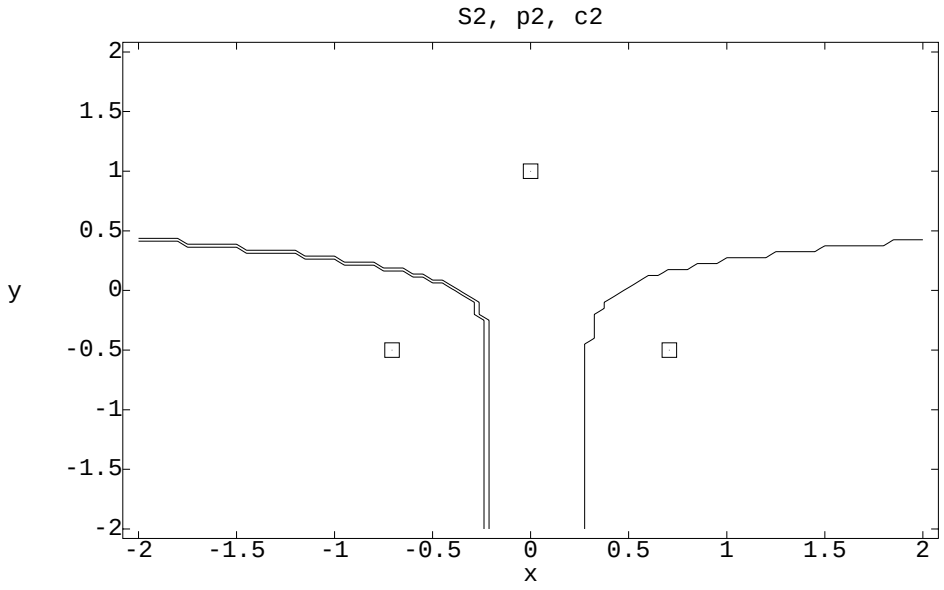

S3, p3, c1
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

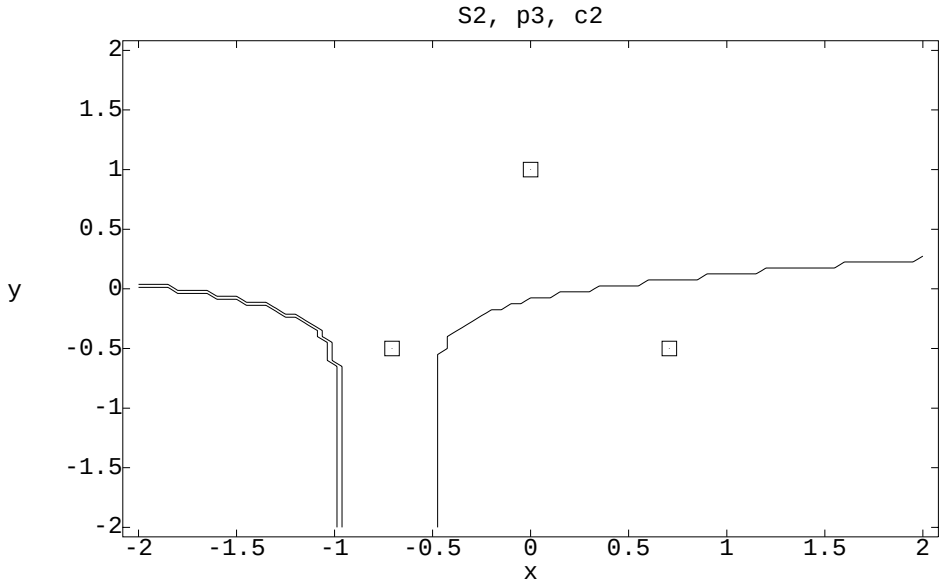

S1, p1, c2
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

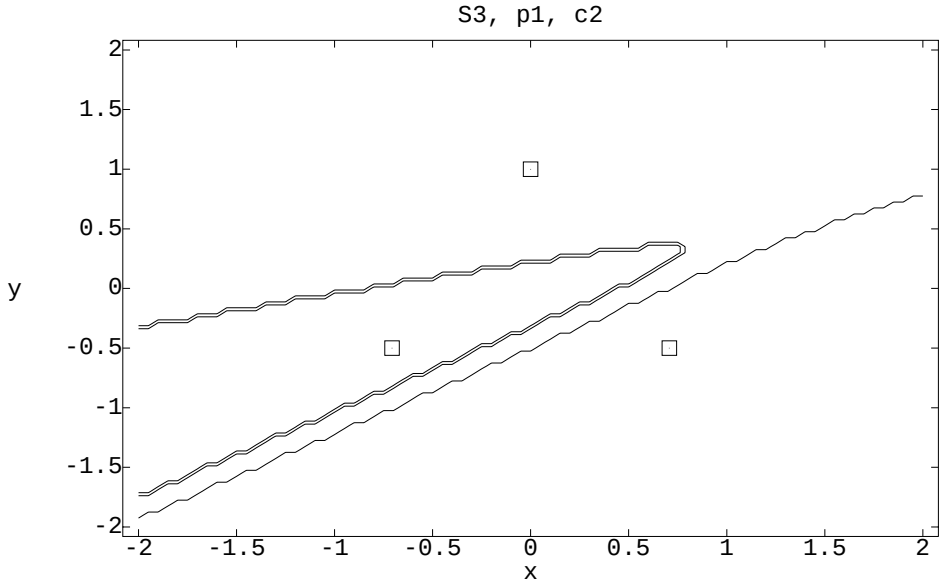

S1, p2, c2
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

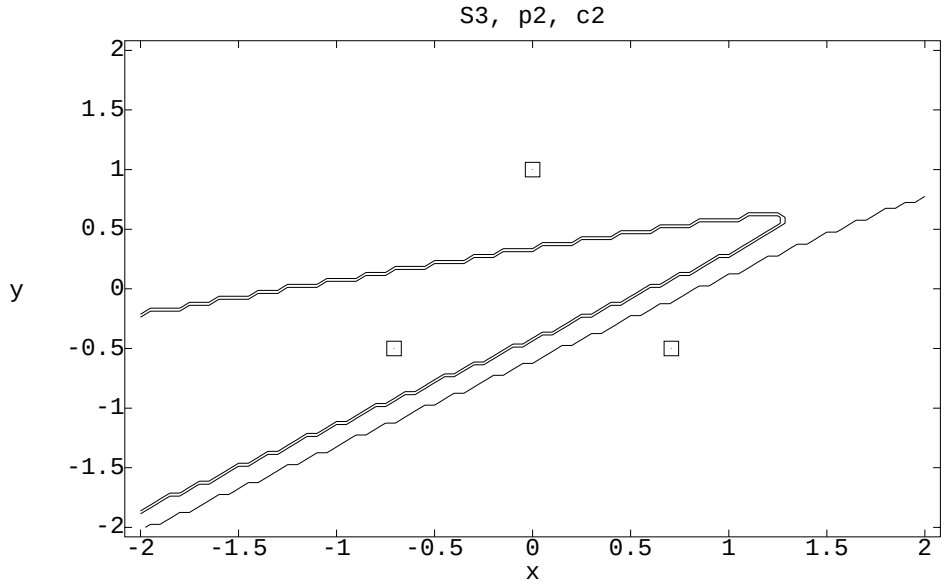

S1, p3, c2
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

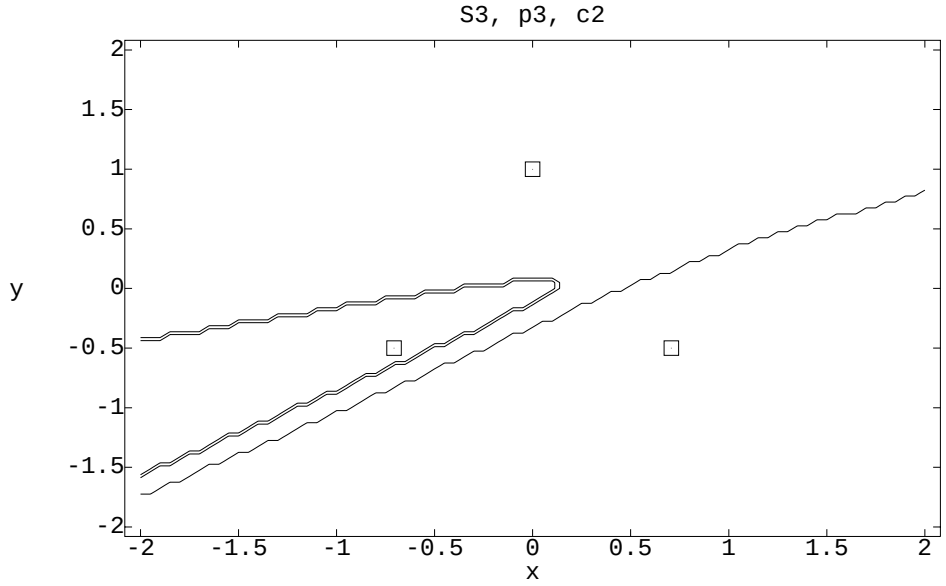

S2, p1, c2
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2


S2, p2, c2
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2


S2, p3, c2
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2


S3, p1, c2
y
x
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2





Note that when $c(i|j)$ does not depend on i (for $i \neq j$), then the regional boundaries are linear.
In this case you can absorb the cost into the prior.
In practice, we have to use sample quatities $\overline{\mathbf{x}}_i$ and $\mathbf{S}$.
When the costs are all equal (or absorbed into the prior), you classify into the population with maximal posterior probability $P(j|x) \propto p_j f_j(x)$ or log posterior probability $\log(p_j) + \log(f_j(x))$.
For multivariate normals with equal variance, the log posterior to maximize is

$$k + \ln(p_j) - (x - \mu_j)'\Sigma^{-1}(x - \mu_j)/2$$

where k is a constant. Alternatively, minimize

$$(x - \mu_j)'\Sigma^{-1}(x - \mu_j) - \ln(p_j) - k$$

If all the p_i's are equal, then we classify into the population which has its mean closest to x, where close is measured using a Σ^{-1} type scaling. The boundaries for this are linear.

If the p_is are not all equal, the boundaries are still linear, but they move away from the means belonging to populations with high probability.
General computations. Let $\mathbf{C}$ be the $g \times g$ matrix of costs, with $C_{ij} = c(i|j)$.
Let $\mathbf{W}$ be the $g \times n$ matrix of posterior probabilities (rows for groups, columns for objects).
$\mathbf{E} = \mathbf{CW}$ is $g \times n$, and $\mathbf{E}_{i,j}$ is the expected cost of classifying object j into class i. Classify object j into the group (row) with the smallest $\mathbf{E}_{i,j}$.

```
Cmd> X <- matrix(vecread(""),5)'
Read from file "~/JW5data/T6-13.DAT"

Cmd> gps <- X[,5]

Cmd> X <- X[,-5]

Cmd> qout <- discrimquad(gps,X)

Cmd> c1 <- matrix(vector(0,1,1,2,0,2,3,3,0),3)

Cmd> c2 <- c1'

Cmd> c3 <- matrix(vector(0,1,1,1,0,1,1,1,0),3)

Cmd> c1
(1,1)           0           2           3
(2,1)           1           0           3
(3,1)           1           2           0
```

Misclassifying a π_3 costs a lot.
Misclassifying into π_3 costs a lot.

```
Cmd> c2
(1,1)           0           1           1
(2,1)           2           0           2
(3,1)           3           3           0

Cmd> c3
(1,1)           0           1           1
(2,1)           1           0           1
(3,1)           1           1           0
```

Equal costs.

```
Cmd> p1 <- vector(1,1,1)/3

Cmd> p2 <- vector(1,2,3)/6
```

```
Cmd> p3 <- vector(3,2,1)/6

Cmd> pp1 <- probsquad(X,qout,prior:p1)

Cmd> pp1 <- pp1’

Cmd> pp2 <- probsquad(X,qout,prior:p2)

Cmd> pp2 <- pp2’

Cmd> pp3 <- probsquad(X,qout,prior:p3)

Cmd> pp3 <- pp3’

Cmd> tmp <- c1%*%pp1

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
(1,1)            6           12           12
(2,1)            2           12           16
(3,1)            2            4           24

Cmd> tmp <- c1%*%pp2

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
(1,1)            4           10           16
(2,1)            0           13           17
(3,1)            0            4           26

Cmd> tmp <- c1%*%pp3

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
(1,1)            9           15            6
(2,1)            4           21            5
(3,1)            3           10           17

Cmd> tmp <- c2%*%pp1

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
```

```
(1,1)           28            1            1
(2,1)           25            4            1
(3,1)           21            2            7

Cmd> tmp <- c2%*%pp2

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
(1,1)           25            1            4
(2,1)           24            4            2
(3,1)           15            3           12

Cmd> tmp <- c2%*%pp3

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
(1,1)           29            1            0
(2,1)           26            4            0
(3,1)           24            2            4

Cmd> tmp <- c3%*%pp1

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
(1,1)           12           10            8
(2,1)            9           12            9
(3,1)            4            8           18

Cmd> tmp <- c3%*%pp2

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
(1,1)            6           12           12
(2,1)            2           12           16
(3,1)            2            4           24

Cmd> tmp <- c3%*%pp3

Cmd> preds <- vector(grade(tmp)[1,])

Cmd> tabs(,gps,preds)
(1,1)           23            4            3
```

(2,1)	20	9	1
(3,1)	14	8	8