Outline

K-means

Partition methods seek to divide examples into a pre-assigned number of clusters $C_1, \ldots, C_K$ where for all $k, k' \in \{1, \ldots, K\}$,

- $C_k \subset \{x_1, \ldots, x_n\}$
- $C_k \cap C_{k'} = \emptyset \quad \forall k \neq k'$
- $\cup_{k=1}^{K} C_k = \{x_1, \ldots, x_n\}$

For Euclidean space, we can assign a centre r_k to each cluster in order to measure within-cluster deviance

$$W_{C_k}(r_k) = \sum_{i: x_i \in C_k} \|x_i - r_k\|_2^2.$$

K-means

The overall objective is to choose both the cluster centres and allocation of points to minimize total within-cluster deviance given by

$$W = \sum_{k=1}^{K} W_{C_k}(r_k).$$

Given the contents of a cluster, simple differentiation of $W_{C_k}(r_k)$ with respect to r_k shows that within-cluster deviance is least when

$$r_k = \frac{1}{|C_k|} \sum_{i:x_i \in C_k} x_i,$$

where $|C_k| = \#\{i : x_i \in C_k\}$ is the number of members of cluster k.
The hard part is the combinatorial task of allocating points to clusters.

K-means

The K-means algorithm is a well-known method that heuristically minimizes W to partition $x_1, \dots, x_n$ into K clusters for some K.

1. Randomly fix K cluster centres $r_1, \dots, r_K$.
2. For each $i = 1, \dots, n$, assign each x_i to the cluster with the nearest centre,

$$x_i \in C_k \quad \Leftrightarrow \quad \|x_i - r_k\| \leq \|x_i - r_k'\| \quad \forall k' \neq k.$$

3. Move cluster centres $r_1, \dots, r_K$ to the average of the new clusters.
4. Repeat 2 and 3 until there is no change.
5. Return the partitions $C_1, \dots, C_K$ at the end.

Some Properties

Some notes about the K-means algorithm.

- **The algorithm stops in a finite number of iterations.** Between steps 2 and 3, W either stays constant -in which case we stop- or it decreases, this implies that we never revisit the same partition. As there are only finitely many partitions, the number of iterations cannot exceed this.
- **The K-means algorithm need not converge to a globally optimal assignment.** K-means is a heuristic search algorithm so it can get stuck at suboptimal configurations. The result depends on the starting configuration.
- **Other partition based methods.** There are many other partition based methods that employ related ideas for example K-medoids differs from K-means in requiring cluster centres r_i to be an observation x_i.

Example: Crabs

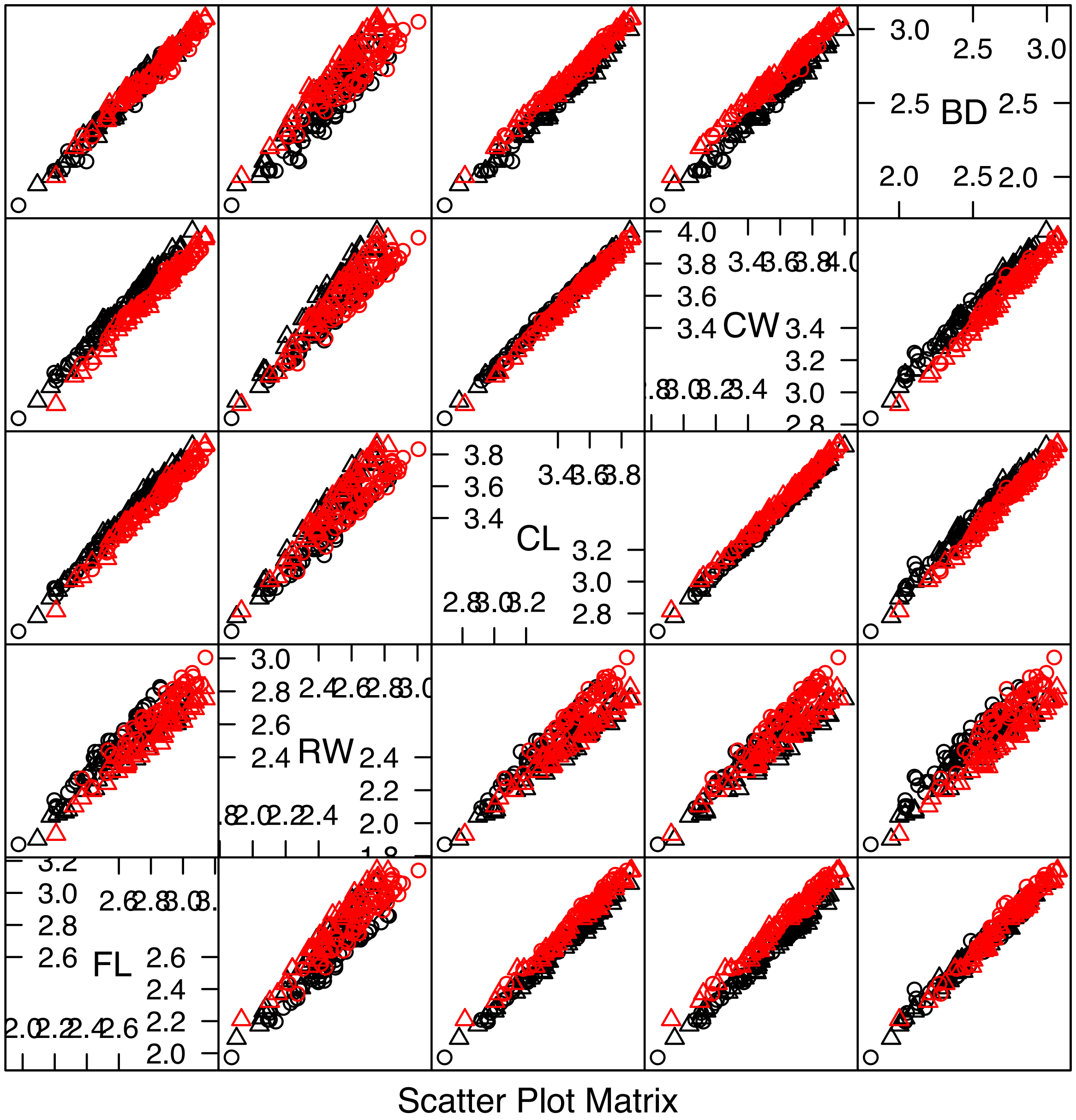
Looking at the Crabs data again.

```
library(MASS)
library(lattice)
data(crabs)

splom(~log(crabs[,4:8]),
    col=as.numeric(crabs[,1]),
    pch=as.numeric(crabs[,2]),
    main="circle/triangle is gender, black/red is species")
```

Example: Crabs

circle/triangle is gender, black/red is species



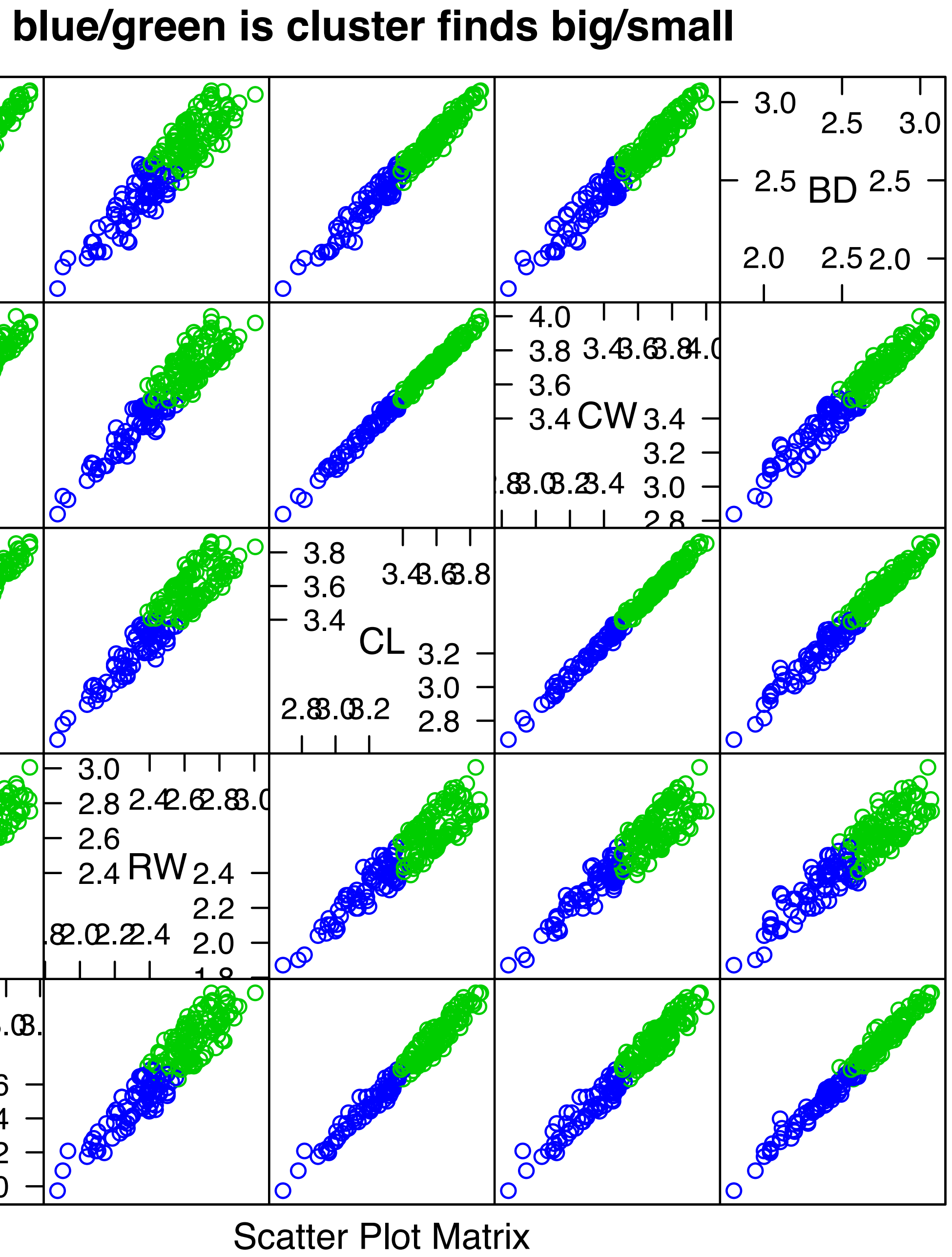
Example: Crabs

Apply kmeans with 2 clusters and plot results.

```
cl <- kmeans( log(crabs[,4:8]), 2, nstart=1, iter.max=10)

splom(~log(crabs[,4:8]),
      col=cl$cluster+2,
      main="blue/green is cluster finds big/small")
```

Example: Crabs



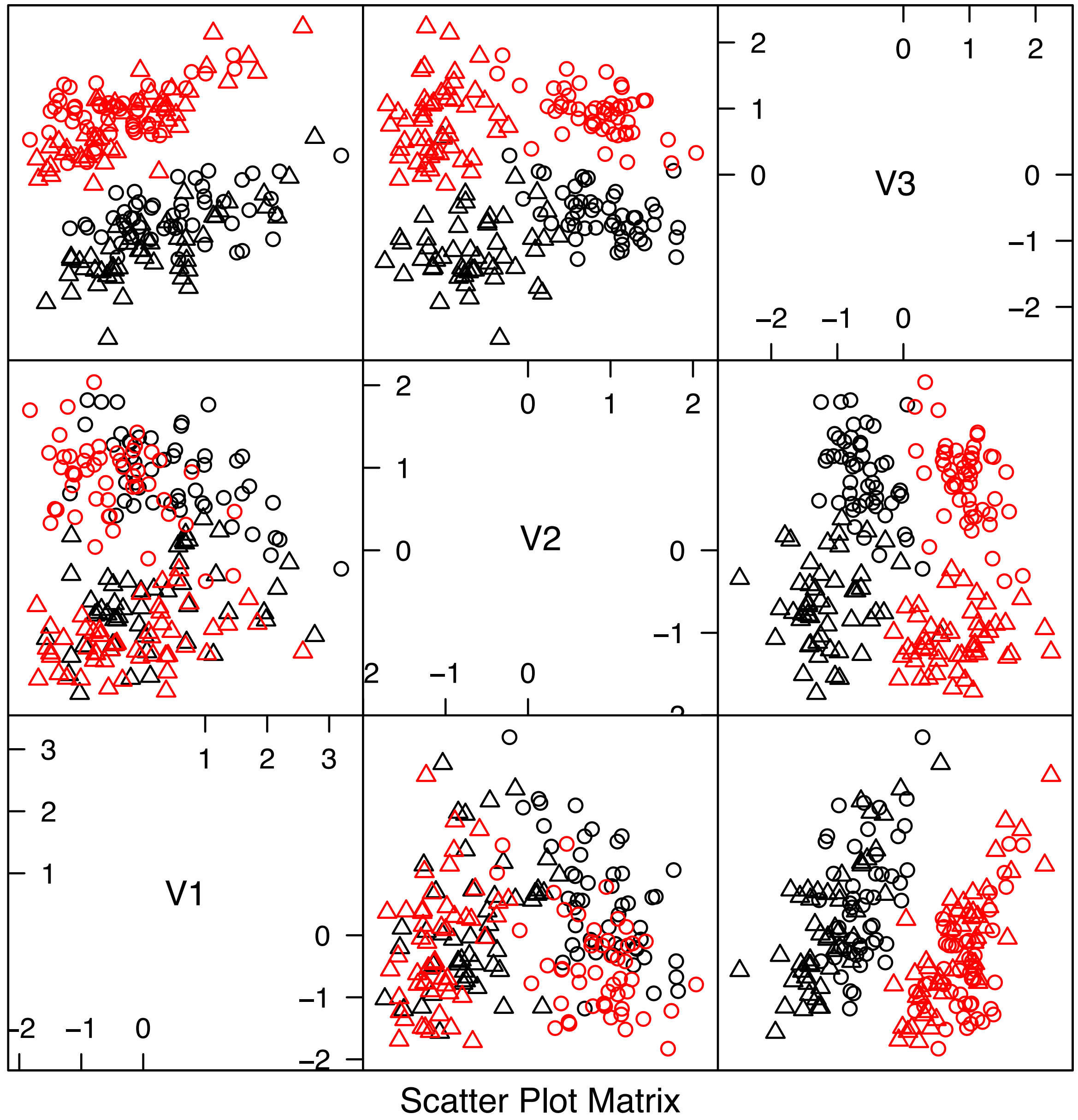
Example: Crabs

Sphere the data.

```
pcp <- princomp( log(crabs[,4:8]) )
spc <- pcp$scores %*% diag(1/pcp$sdev)
splom( ~spc[,1:3],
    col=as.numeric(crabs[,1]),
    pch=as.numeric(crabs[,2]),
    main="circle/triangle is gender, black/red is species")
```

Example: Crabs

circle/triangle is gender, black/red is species

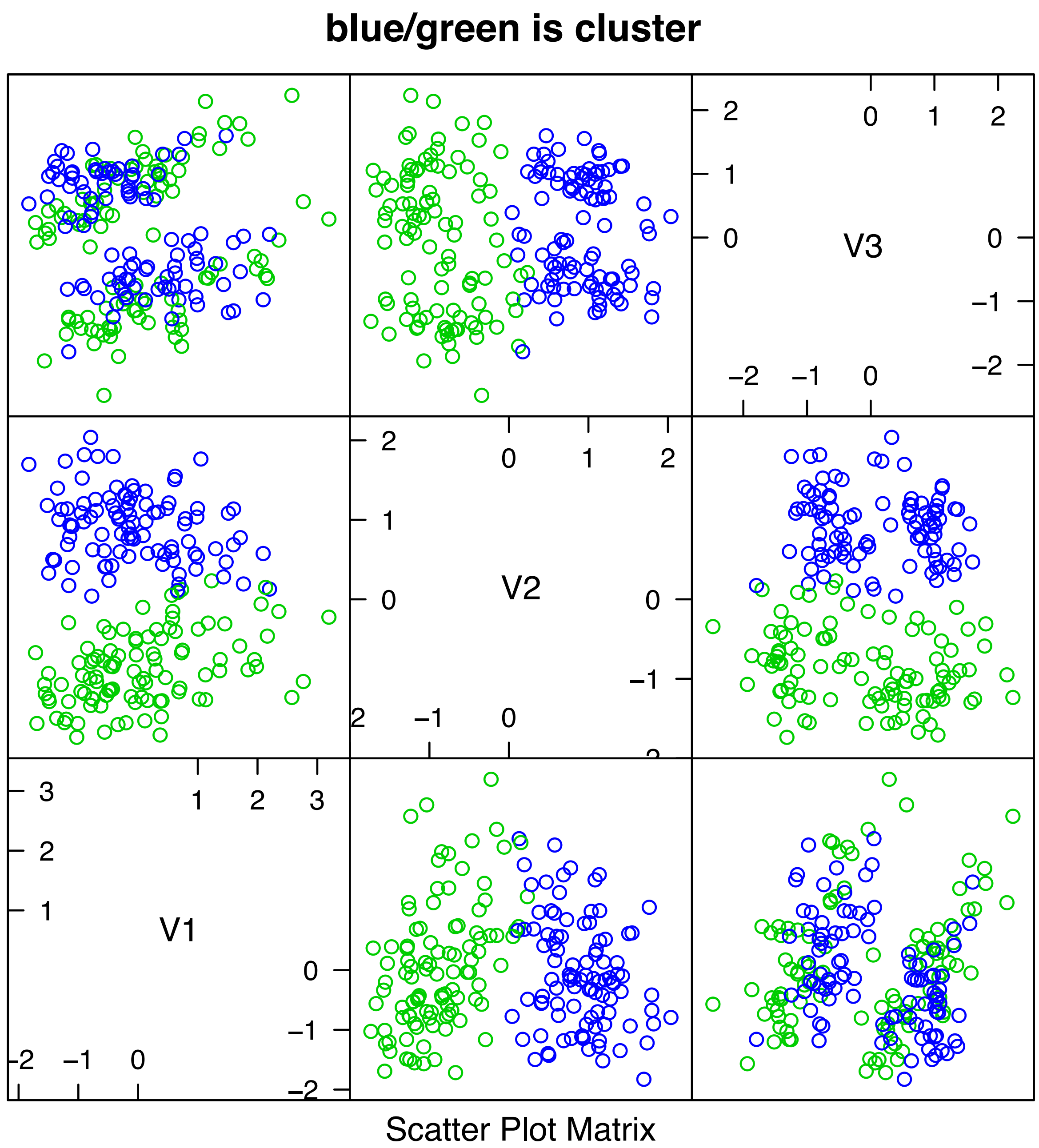


Example: Crabs

And apply K-means again.

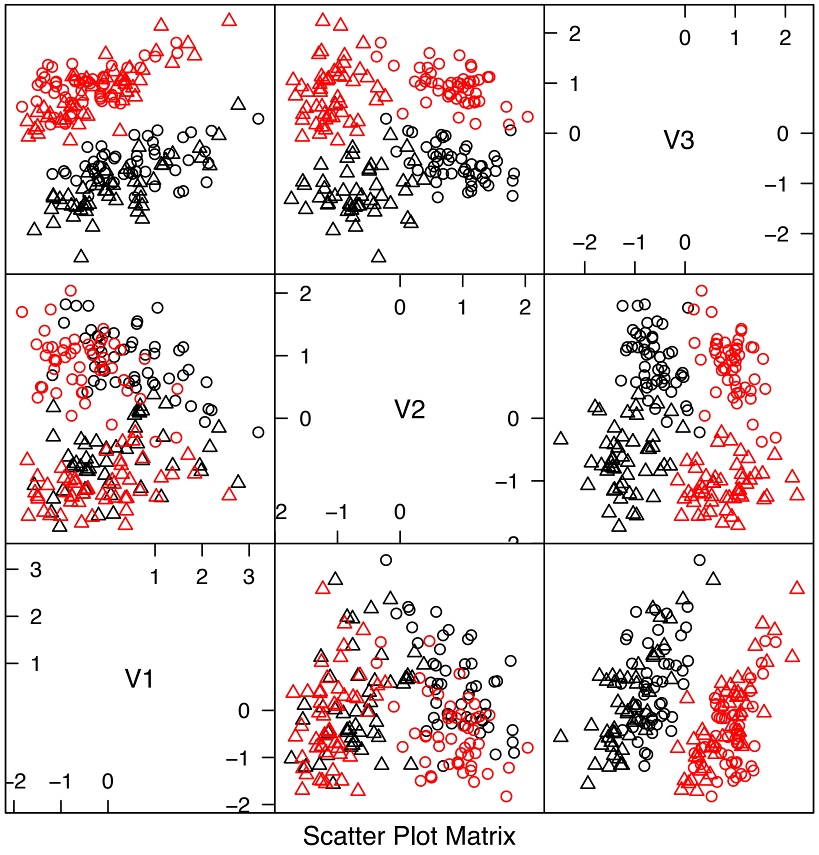
```
cl <- kmeans(spc, 2, nstart=1, iter.max=20)
splom( ~spc[,1:3],
      col=cl$cluster+2, main="blue/green is cluster")
```

Example: Crabs

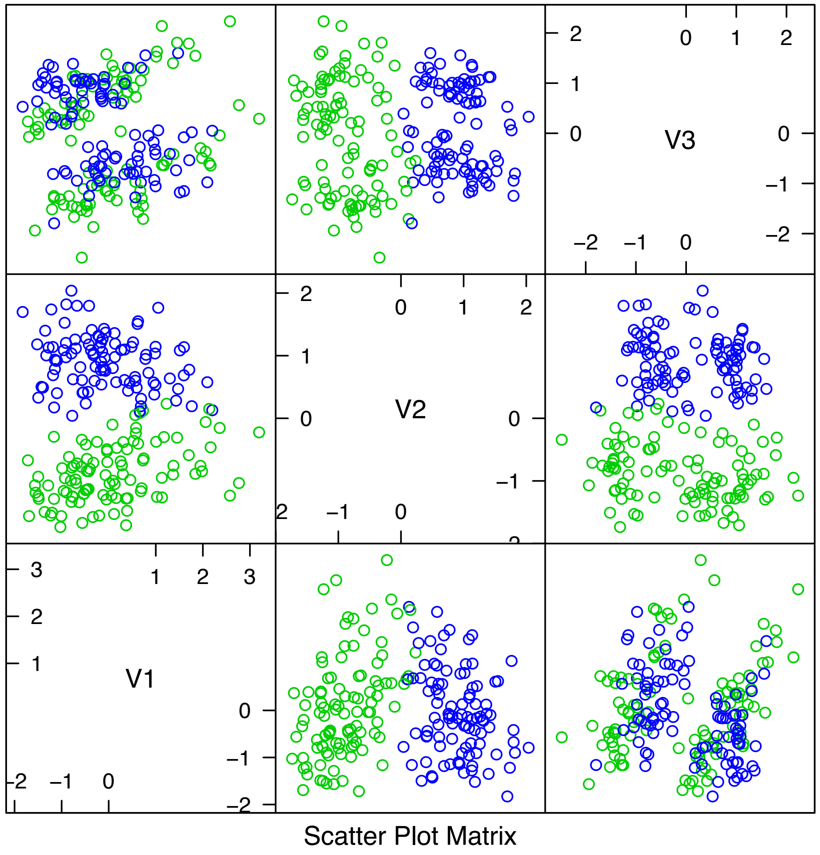


Example: Crabs

circle/triangle is gender, black/red is species



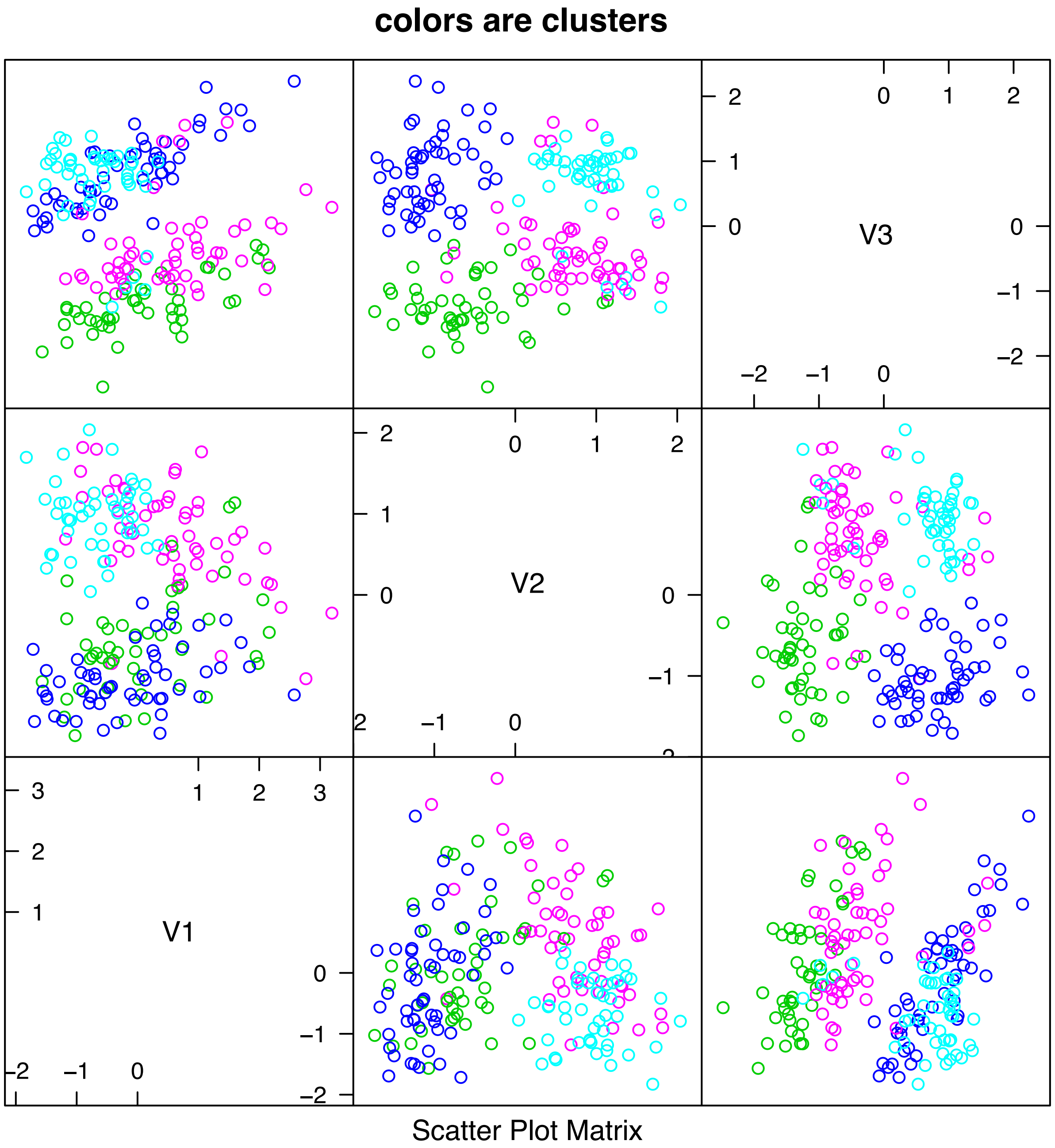
blue/green is cluster



Discovers gender difference...
Results depends crucially on sphering the data first.

Example: Crabs

Using 4 cluster centers.



Example: Crabs