Canonical Correlation Analysis III

Mengqian LU

Canonical Coefficients

- Two sets of canonical coefficients (weights)
  - One set to combine the Xs
  - The other set to combine the Ys
  - Interpreted similarly to regression coefficients

F_1 = E1$Eigenvector[1,1] * X_1 + E1$Eigenvector[1,1] * X_2

F_2 = E1$Eigenvector[1,2] * X_1 + E1$Eigenvector[2,2] * X_2

G_1 = E2$Eigenvector[1,1] * Y_1 + E2$Eigenvector[1,1] * Y_2

G_2 = E2$Eigenvector[1,2] * Y_1 + E2$Eigenvector[2,2] * Y_2

Canonical Coefficients

- Two sets of canonical coefficients (weights)
  - One set to combine the Xs
  - The other set to combine the Ys
  - Interpreted similarly to regression coefficients
- Test statistical significance of canonical correlations
  - The maximum number of canonical correlations equals to the number of variables in the smaller set
  - Not all will be statistically significant
  - Not all statistical significant CCs will be meaningful
  - This can be done using *CCP* package in R by *p.asym()*, with options for test statistics, commonly use "Wilks" stand for Wilk's Lambda to be approximated by Chi-square distribution – Bartlett's Chi Square test

Example: head lengths of 1st and 2nd sons (IAMA 3.13.1)

```
> headsize
      head1 breadth1 head2 breadth2
 [1,]   191      155   179      145
 [2,]   195      149   201      152
 [3,]   181      148   185      149
 [4,]   183      153   188      149
 [5,]   176      144   171      142
 [6,]   208      157   192      152
 [7,]   189      150   190      149
 [8,]   197      159   189      152
 [9,]   188      152   197      159
[10,]   192      150   187      151
[11,]   179      158   186      148
[12,]   183      147   174      147
[13,]   174      150   185      152
[14,]   190      159   195      157
[15,]   188      151   187      158
[16,]   163      137   161      130
[17,]   195      155   183      158
[18,]   186      153   173      148
[19,]   181      145   182      146
[20,]   175      140   165      137
[21,]   192      154   185      152
[22,]   174      143   178      147
[23,]   176      139   176      143
[24,]   197      167   200      158
[25,]   190      163   187      150
```

```
headsize.std <- sweep(headsize, 2, apply(headsize, 2, sd), FUN="/")
R <- cor(headsize.std)
R11 <- R[1:2,1:2]
R12 <- R[3:4,1:2]
R21 <- R[1:2,3:4]
R22 <- R[3:4,3:4]
R11.inv <- solve(R11)
R22.inv <- solve(R22)

# compute E1 and E2
E1 <- R11.inv %*% R12 %*% R22.inv %*% R21
E2 <- R22.inv %*% R21 %*% R11.inv %*% R12

# compute eigenvalues and eigenvectors of E1 and E2:
eigen(E1)
eigen(E2)
```

$$\mathbf{E}_1 = \mathbf{R}_{11}^{-1}\mathbf{R}_{12}\mathbf{R}_{22}^{-1}\mathbf{R}_{21}, \quad \mathbf{E}_2 = \mathbf{R}_{22}^{-1}\mathbf{R}_{21}\mathbf{R}_{11}^{-1}\mathbf{R}_{12}$$

E_1 is p-by-p; E_2 is q-by-q.
p is the dimension of the predictors set
q is the dimension of the predictants set.

```
>eigen(E1)
$values
[1] 0.621781555 0.002887785
$vectors [,1] [,2]
[1,] -0.6947269 -0.7089828
[2,] -0.7192736 0.7052258
> eigen(E2)
$values
[1] 0.621781555 0.002887785
$vectors [,1] [,2]
[1,] 0.7424369 -0.7039264
[2,] 0.6699160 0.7102729
```

The canonical correlations express the association between the x and y variables after removal of the within-set correlation.

F_1 = -0.69*head1 - 0.72*breadth1;

F_2 = -0.70*head1 + 0.71*breadth1;

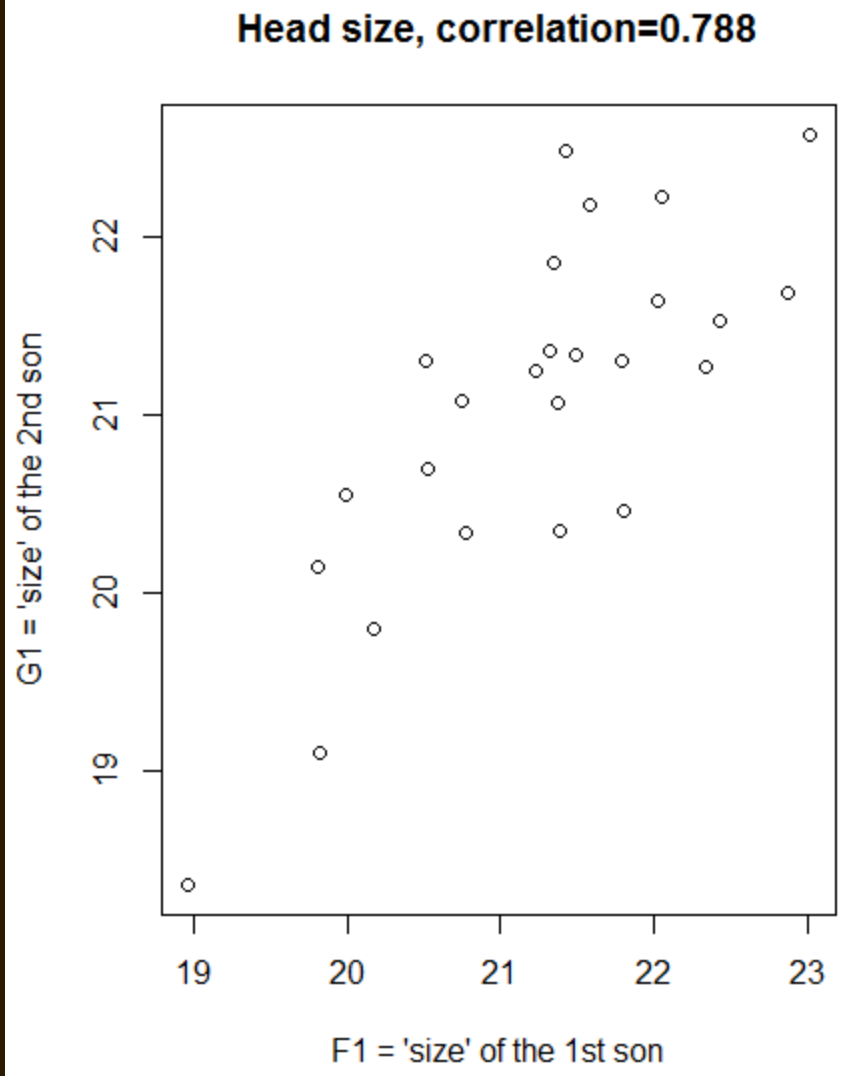
G_1 = +0.74*head2 + 0.70*breadth2;

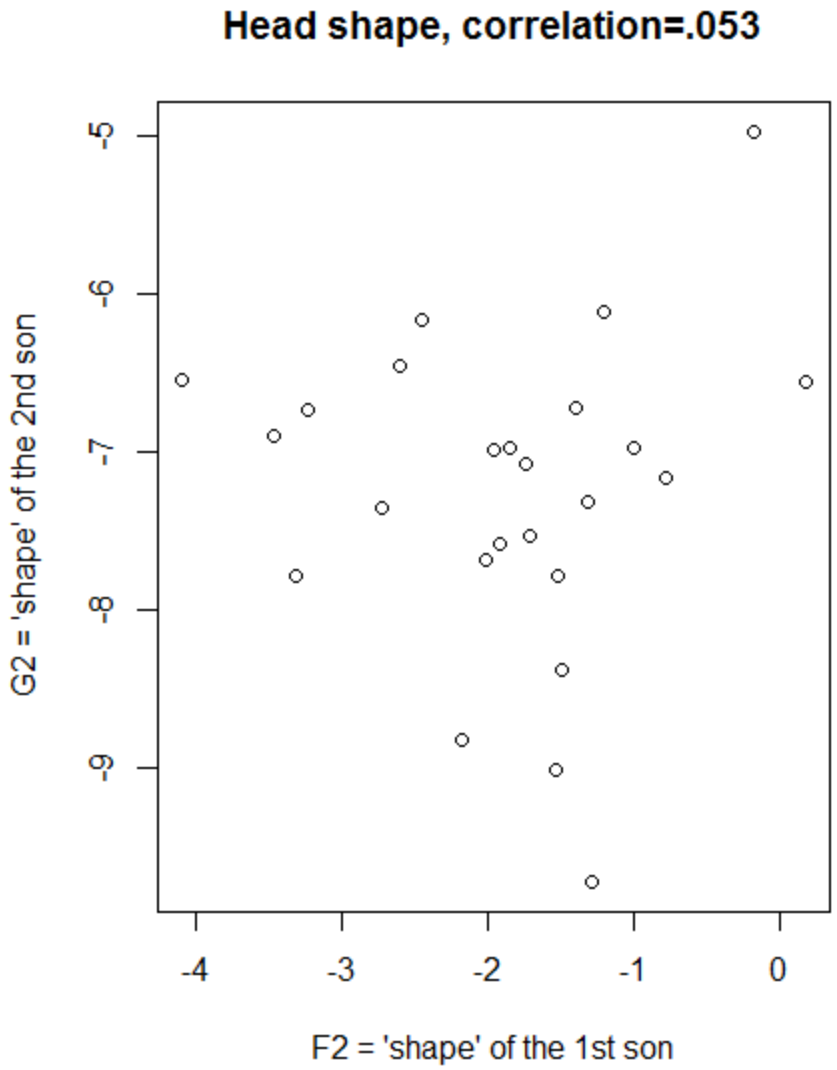
G_2 = -0.70*head2 + 0.71*breadth2.

 # Since we work with standardized data from the beginning , so
 # no correction of scaling of CC vectors is needed:
 # OPTIONAL:

```
# compute the canonical correlation vectors:
(a1 <- eigen(E1)$vectors[,1])
(a2 <- eigen(E1)$vectors[,2])
(b1 <- eigen(E2)$vectors[,1])
(b2 <- eigen(E2)$vectors[,2])
# correct the scaling of the canonical correlation vectors:
(a1 <- -1 * a1 / sqrt(t(a1) %*% R11 %*% a1))
(a2 <- -1 * a2 / sqrt(t(a2) %*% R11 %*% a2))
(b1 <- b1 / sqrt(t(b1) %*% R22 %*% b1))
(b2 <- -1 * b2 / sqrt(t(b2) %*% R22 %*% b2))
# check scaling:
t(a1) %*% R11 %*% a1
t(a2) %*% R11 %*% a2
t(b1) %*% R22 %*% b1
t(b2) %*% R22 %*% b2
```

```
# compute canonical correlation variables:
F1 <- headsize.std[,1:2] %*% a1   # size first son
F2 <- headsize.std[,1:2] %*% a2   # shape first son
G1 <- headsize.std[,3:4] %*% b1   # size second son
G2 <- headsize.std[,3:4] %*% b2   # shape second son

# check covariance matrix:
round(var(cbind(F1,F2,G1,G2)),3)

# plot canonical correlation variables:
par(mfrow=c(1,2))
plot(F1,G1,main="Head size, correlation=0.788", xlab="F1 =
'size' of the 1st son", ylab="G1 = 'size' of the 2nd son")
plot(F2,G2, main="Head shape, correlation=.053", xlab="u2 =
'shape' of the 1st son", ylab="G2 = 'shape' of the 2nd son")
```


Head size, correlation=0.788
G1 = 'size' of the 2nd son
F1 = 'size' of the 1st son
19
20
21
22
23


Head shape, correlation=.053
G2 = 'shape' of the 2nd son
F2 = 'shape' of the 1st son
-5
-6
-7
-8
-9
-4
-3
-2
-1
0

Loadings (structure coefficients)

- Questions to assist your interpretations:
  1. How well do the variates on either side relate to their own original variables?
  2. What is the associations between a variable and its respective canonical variate? – correlations
- Recall that: the pair of canonical variates with best correlation might not exactly interpretable
- Coefficients are for the computation of the variates, loadings are more for expressing the relationships of the variables to the construct (canonical variates)
  - ❑ Canonical communality coefficient
  - ❑ Canonical variate adequacy coefficient

Coefficients

- Canonical communality coefficient
  - Sum of the squared structure coefficients (loadings) across selected* canonical variates for a given variable
  - Measures how much variance of that original variable is explained or reproduced by the canonical variates

  * If looking at all variates, it shall equal one, typically we often want to check those retained for interpretation or prediction.
- Canonical variate adequacy coefficient
  - Average of all the squared structure coefficients (loadings) for one set of variables with respect to their canonical variate
  - Measures how well a resulted canonical variate represents the variance in that set of original variables.

Redundancy

- Questions to assist checking redundancy:
  1. How strongly do the original variables in one set relate to the canonical variates on the other side?
  2. How much of the average proportion of variance of the original variables in one set may be predicted from the variables in the other set – High redundancy suggests potentially high ability to predict
     - Product of the mean squared structure coefficient, i.e., the canonical adequacy coefficient for a canonical variate times the squared canonical correlation coefficient.
     - Canonical correlation reflects the % of variance in the predictant canonical variate explained by the predictor canonical variate

➢ Redundancy has to do with assessing the effectiveness of the canonical analysis in capturing the variance of the original variables

CCA in R

`cancor()`

data(*LifeCycleSavings*): savings ratio from 1960-1970

sr	numeric	aggregate personal savings
pop15	numeric	% of population under 15
pop75	numeric	% of population over 75
dpi	numeric	real per-capita disposable income
ddpi	numeric	% growth rate of dpi

```
pop = LifeCycleSavings[, 2:3] # select pop15 and pop75
oec = LifeCycleSavings[, -(2:3)] # select all the others

cancor(pop, oec)
```

CCA in R

```
cancor(pop, oec)
$cor
[1] 0.8247966 0.3652762

$xcoef
[,1]         [,2]
pop15 -0.009110856 -0.03622206
pop75  0.048647514 -0.26031158

$ycoef
[,1]           [,2]           [,3]
sr   0.0084710221  3.337936e-02 -5.157130e-03
dpi  0.0001307398 -7.588232e-05  4.543705e-06
ddpi 0.0041706000 -1.226790e-02  5.188324e-02

$xcenter
pop15   pop75
35.0896  2.2930

$ycenter
sr       dpi      ddpi
9.6710 1106.7584    3.7576
```

cancor()

0.08(sr)+0.0001(dpi)+0.004(ddpi) = -0.009(pop15)+0.048(pop75)

1st Canonical variate function

CCA in R

`matcor()&cc()` in package(CCA)

pakcage(CCA) provides extension to the cancor()

- cancor() has limited outputs, package(CCA) provides a complete list of outputs in terms of both numerical and graphical
- package(CCA) enhances handling of missing values
- package (CCA) can perform regularized CCA when n<max(p,q).

data(*nutrimouse*): from the Toxicology & Pharmacology Laboratory

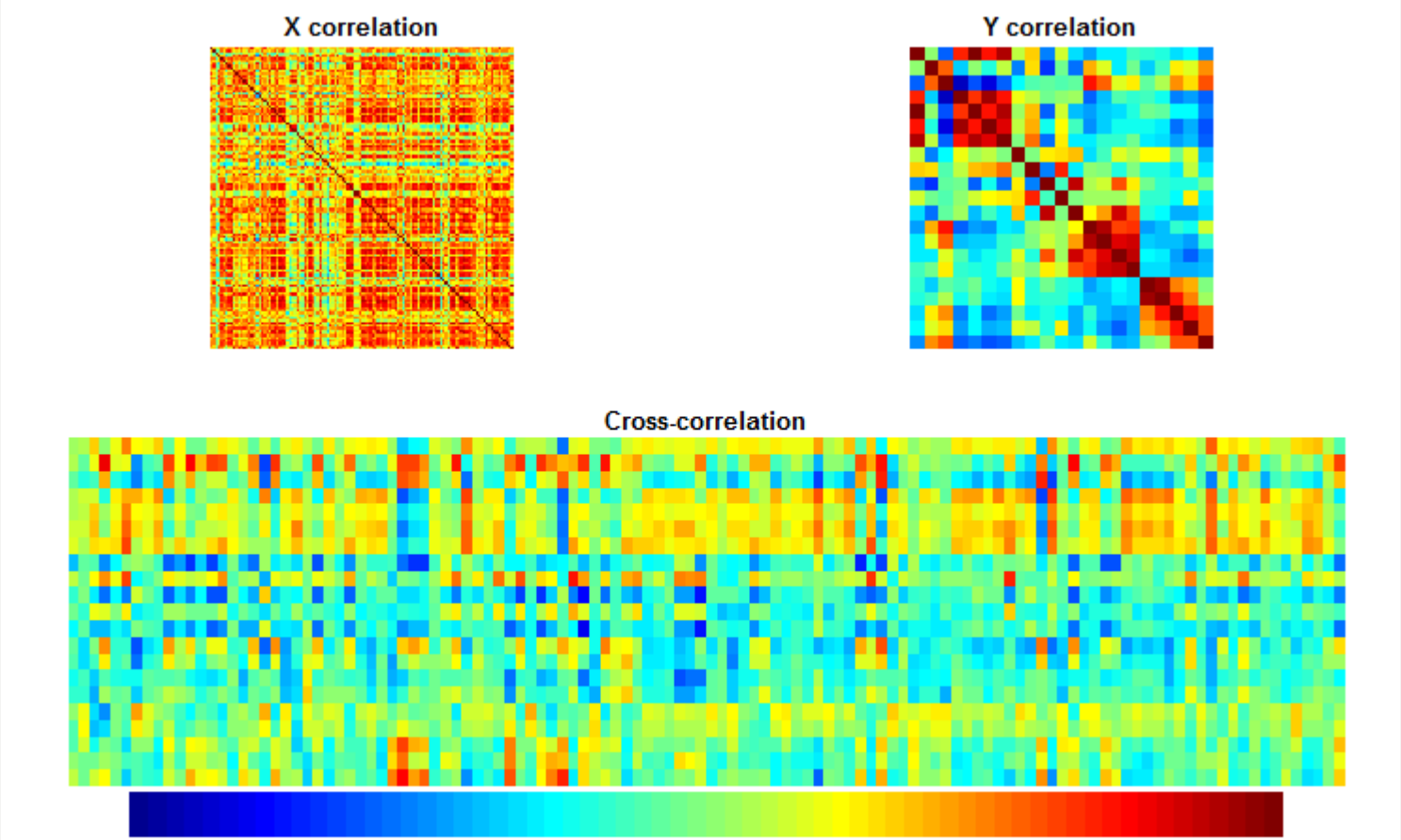
A list containing the following components:

- gene: data frame (40 * 120) with numerical variables
- lipid: data frame (40 * 21) with numerical variables
- diet: factor vector (40)
- genotype: factor vector (40)

CCA in R

matcor()&cc()

```
X = as.matrix(nutrimouse$gene)
Y = as.matrix(nutrimouse$lipid)
correl = matcor(X, Y)
img.matcor(correl, type = 2)
```



Correlation matrices for: X variables (upper-left), Y variables (upper-right), cross-correlation X × Y (bottom). Increasing values are translated into colors from blue (negative correlation) to red (positive correlation).

Classical CCA: n > p+q

CCA in R

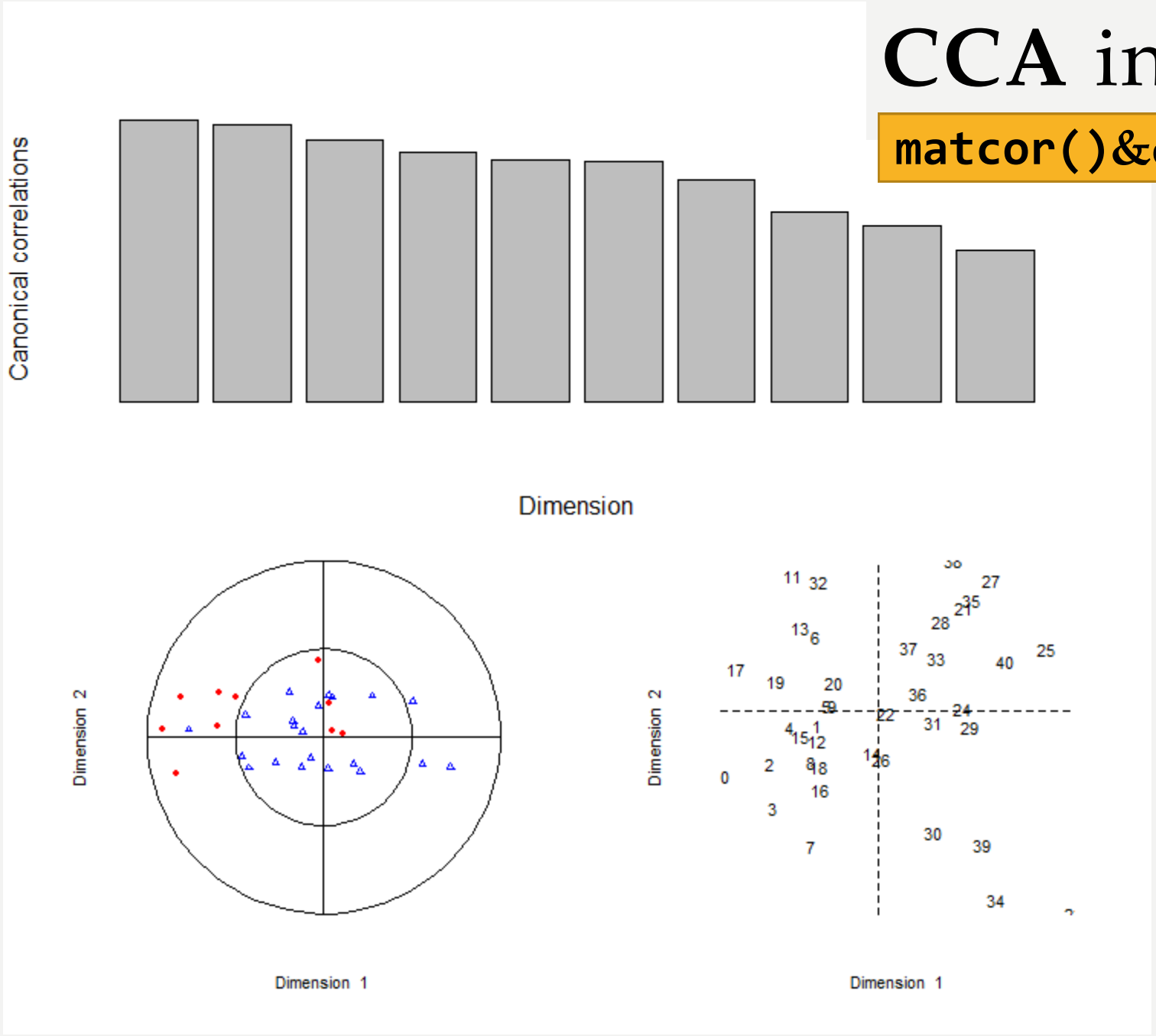
matcor()&cc()

```
Xr = as.matrix(nutrimouse$gene[, sample(1:120, size = 10)])
res.cc = cc(Xr, Y)
par(mar=c(4,4,4,4))
barplot(res.cc$cor, xlab = "Dimension",ylab = "Canonical
correlations", names.arg = 1:10, ylim = c(0,1))
plt.cc(res.cc)
```

1. Randomly choose 10 genes among the 120 → Xr;
2. CCA on Xr (40-by-10) and Y (40-by-21)
3. Barplot of the canonical correlations (remember scree plot?)
4. Original variables and units are plotted on the first two canonical variates (remember similar presentations in PCA, MDS and EFA etc. ?)

CCA in R

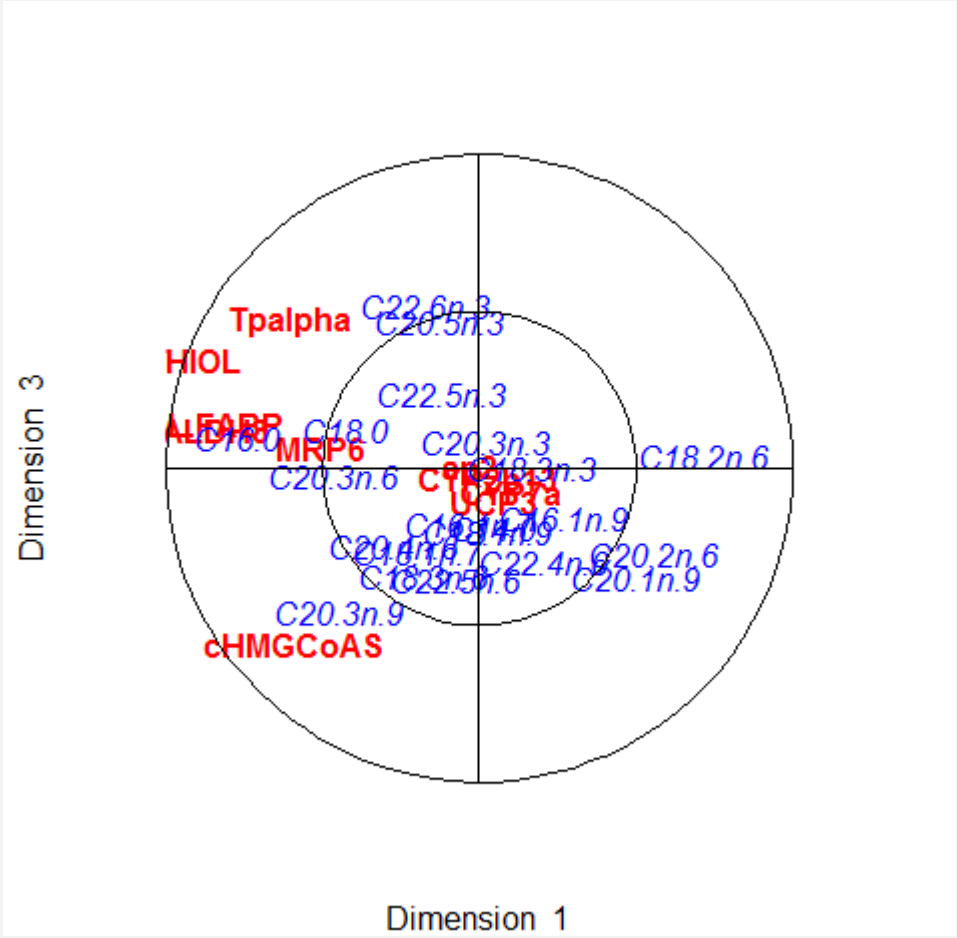
`matcor()&cc()`



CCA in R

matcor()&cc()

```
plt.cc(res.cc,d1=1,d2=3,type="v",var.label=TRUE)
```



CCA in R

`estim.regul()`
`rcc()`

Regularized CCA

Gene (X): 40-by-120; Lipid (Y): 40-by-21

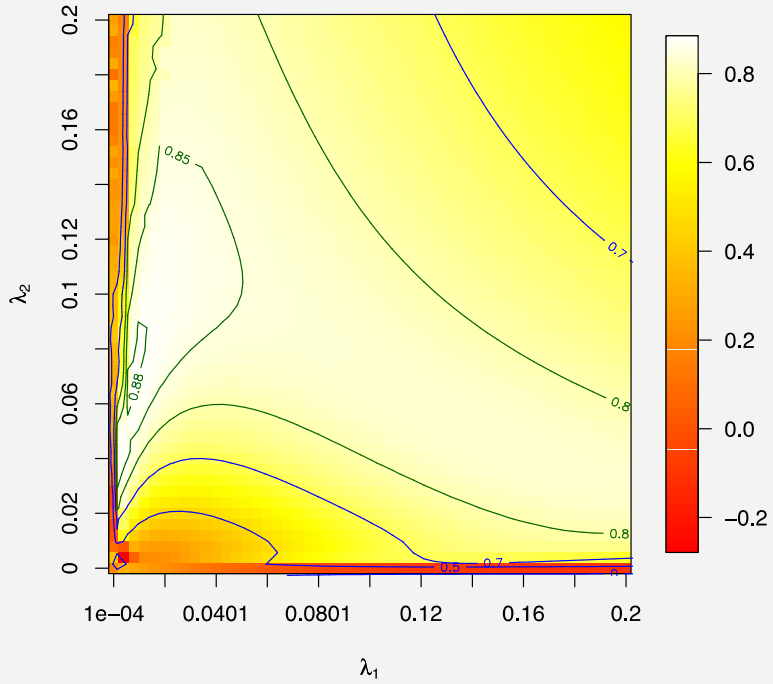
Correlation matrix: S_{XX} and S_{YY}

RCCA: $\sum_{XX}(\lambda_1) = S_{XX} + \lambda_1 I_p; \sum_{YY}(\lambda_2) = S_{YY} + \lambda_2 I_q$

How to set 'good' values for the regularization parameters?

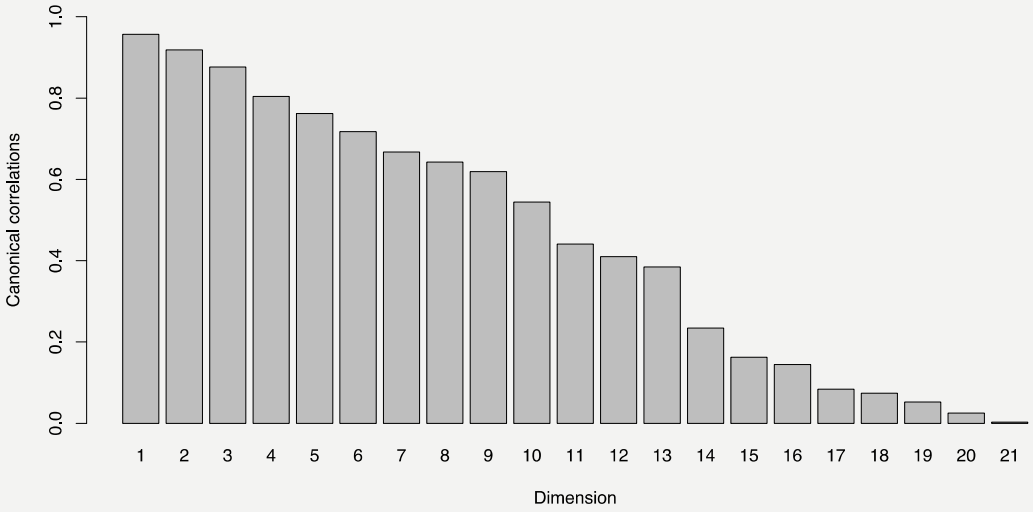
- Leave-one-out cross validation by `estim.regul()`
- Default grid is 5 equally-spaced points [0.001,1]

```
res.regul = estim.regul(X, Y, plt = TRUE, grid1 =
seq(0.0001, 0.2, l=51), grid2 = seq(0, 0.2, l=51))
# λ1 = 0.008096;
# λ2 = 0.064;
# CV-score = 0.8852923
contour(res.regul$grid1, res.regul$grid2, res.regul$mat, add = TRUE,
levels = c(0,0.5,0.7), col = "blue")
contour(res.regul$grid1, res.regul$grid2, res.regul$mat, add = TRUE,
levels = c(0.8,0.85,0.88), col = "darkgreen")
```

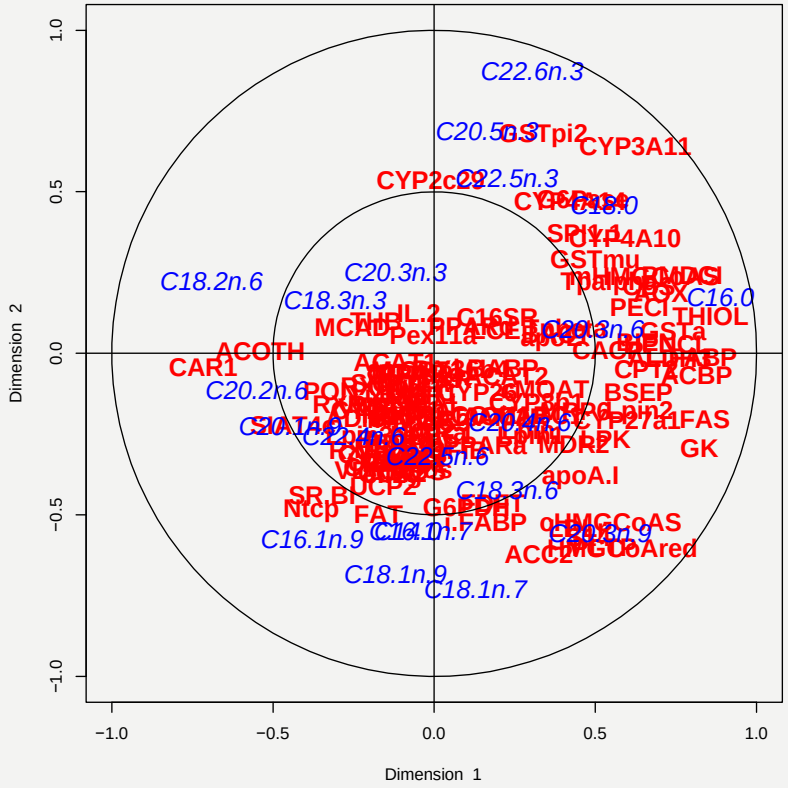


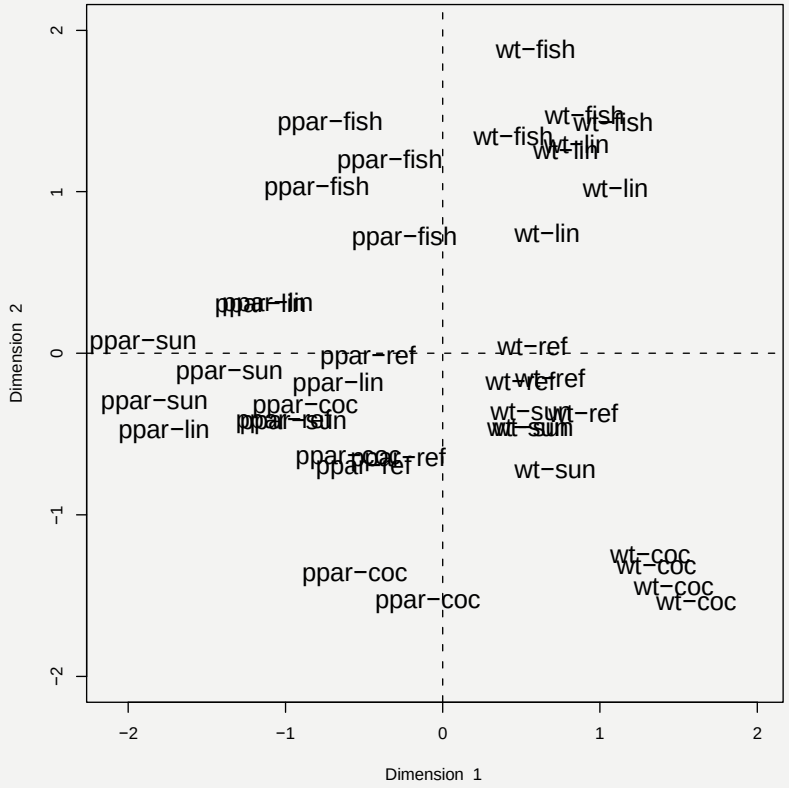
CV-score for λ_1 and λ_2 on a 51 × 51 grid defined by equally-spaced discretization points on the region: $0.0001 \leq \lambda_1 \leq 0.2$ and $0 \leq \lambda_2 \leq 0.2$. Two kinds of contour plots are also displayed for values equal to {0-0.5-0.7} (in blue) and to {0.8-0.85-0.88} (in green).

```
# Regularized CCA
res.rcc = rcc(X, Y, 0.008096, 0.064)
barplot(res.rcc$cor, xlab = "Dimension",ylab = "Canonical
correlations", names.arg = 1:21, ylim = c(0,1))
plt.cc(res.rcc, var.label = TRUE,ind.names =
paste(nutrimouse$genotype, nutrimouse$diet, sep = "-"))
```



Martin et al. (2007)





"CCA" in R

`cca()` in package(vegan)

- pakcage(vegan) developed for community ecologists; thus a good collection of ordination methods for the field;
  - PCA, MDS, FA and more…
- But it has a lot of functions & associated visualization tools for us to use for other data.

data(*dune*): Vegetation and Environment in Dutch Dune Meadows

- The *dune* meadow vegetation data has cover class values of 30 species on 20 sites. The corresponding environmental data frame *dune.env* is a data frame of 20 observations on 5 variables

`CCA` in package(vegan) stands for Canonical Correspondence Analysis

MDS in package(vegan)

- Non-metric MDS
  - Dissimilarity: *vegdist(,…)*
    "manhattan", "euclidean", "canberra", "bray", "kulczynski","jaccard", "gower", "altGower", "morisita", "horn", "mountford", "raup" , "binomial", "chao","cao" or "mahalanobis"

```
data(dune)
dis = vegdist(dune)
m = monoMDS(dis, model = "loc")
```

loc: non-metric MDS with separate regressions for each point

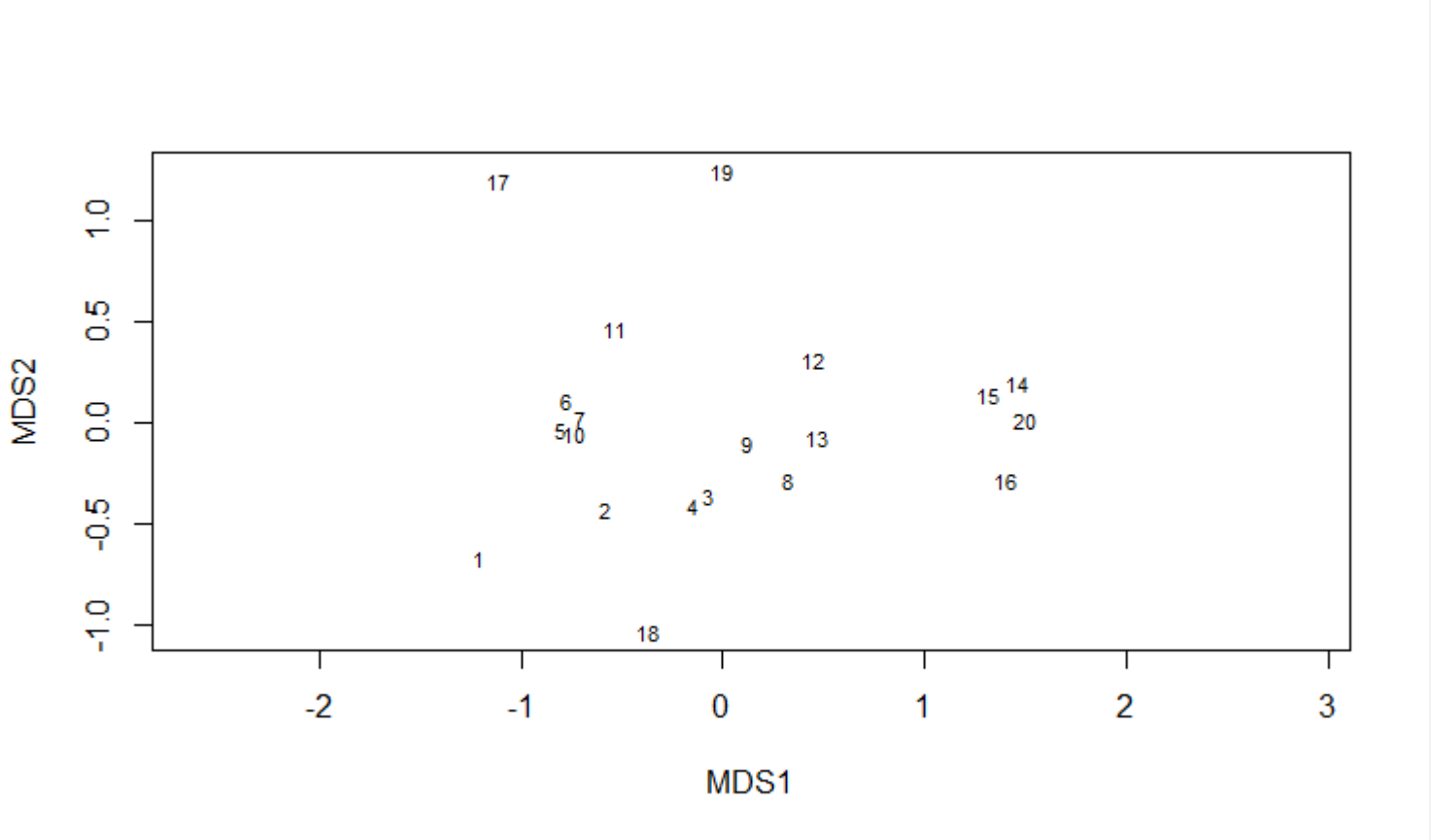
MDS model: "global" is normal non-metric MDS with a monotone regression, "local" is non-metric MDS with separate regressions for each point, "linear" uses linear regression, and "hybrid" uses linear regression for dissimilarities below a threshold in addition to monotone regression.

MDS in package(vegan)

```
> m

Call:
  monoMDS(dist = dis, model = "loc")

Local non-metric Multidimensional Scaling

20 points, dissimilarity ‘bray’, call ‘vegdist(x =
dune)’

Dimensions: 2
Stress:     0.1232248
Stress type 1, weak ties
Scores scaled to unit root mean square, rotated to
principal components
Stopped after 71 iterations: Stress nearly unchanged
(ratio > sratmax)
> plot(m)
```

monoMDS has some rotation options to assist your initial diagnosis

MDS in package(vegan)



Let's see another better wrapper for non-metric MDS in *vegan*

`metaMDS()` in package(vegan)

metaMDS() – THE POWERFUL WRAPPER

1. Use adequate dissimilarity measures (function *vegdist*),
2. Run NMDS several times with random starting configurations – avoid local minimum,
3. Compares results, w.r.t. minimizing sum of squared differences (function *procrustes*),
4. Stop after finding twice a similar minimum stress solution,
5. Scale & rotate the solution,
6. Add species scores to the configuration as weighted averages (function *wascores*): before we only see the 20 sites

A COMPLETE ANALYSIS

metaMDS() in package(vegan)

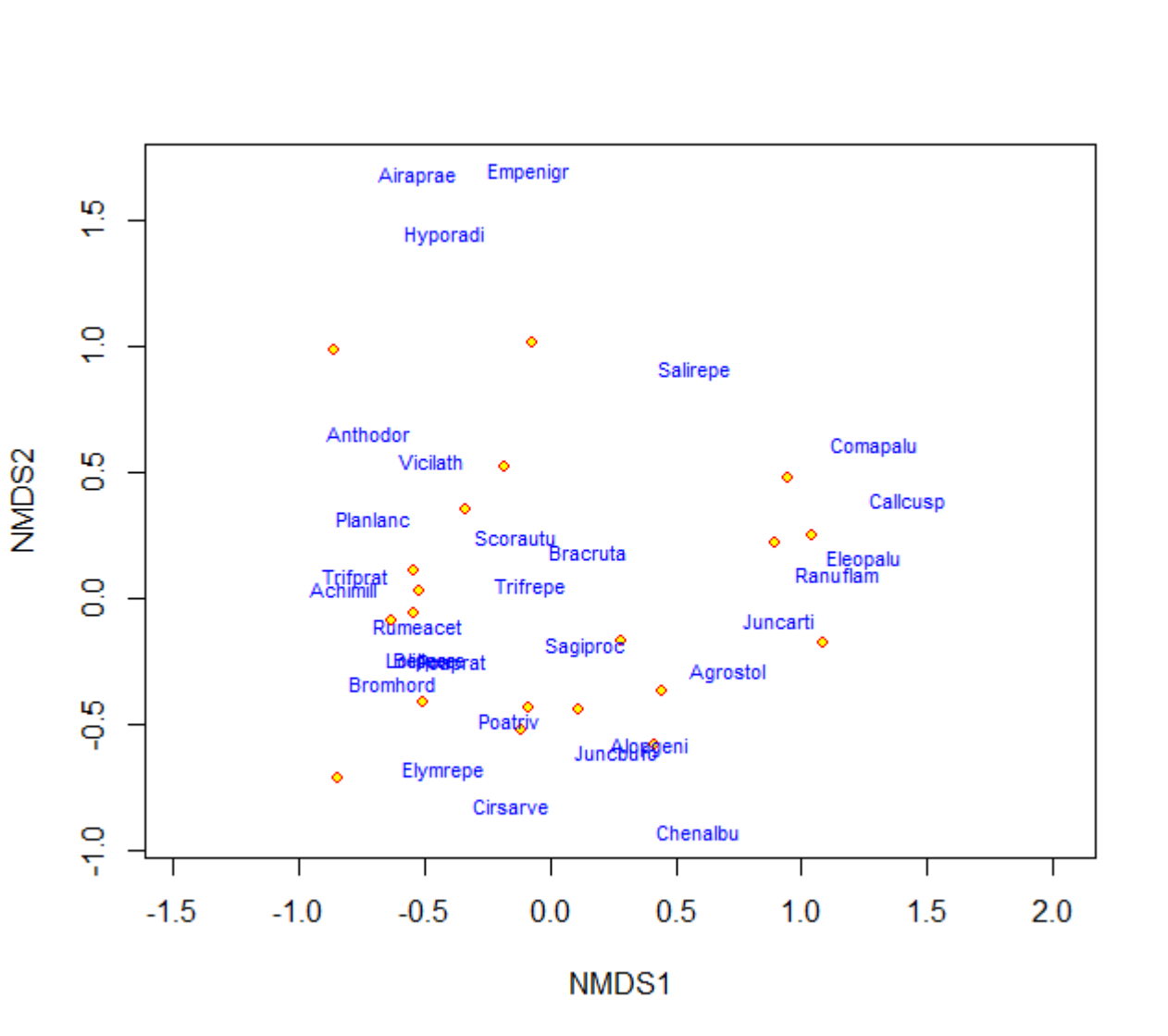
```
> ord <- metaMDS(dune)
Run 0 stress 0.1192678
Run 1 stress 0.1812935
Run 2 stress 0.330578
Run 3 stress 0.1183186
... New best solution
... procrustes: rmse 0.02026999  max resid 0.06495494
Run 4 stress 0.1922247
Run 5 stress 0.2361935
Run 6 stress 0.119269
Run 7 stress 0.1808916
Run 8 stress 0.1192679
Run 9 stress 0.1192679
Run 10 stress 0.1192685
Run 11 stress 0.1183186
... procrustes: rmse 2.444379e-06  max resid
7.278529e-06
*** Solution reached
```

metaMDS() in package(vegan)

```
> ord

Call:
metaMDS(comm = dune)

global Multidimensional Scaling using monoMDS

Data:     dune
Distance: bray

Dimensions: 2
Stress:     0.1183186
Stress type 1, weak ties
Two convergent solutions found after 11 tries
Scaling: centring, PC rotation, halfchange scaling
Species: expanded scores based on ‘dune’
```

CCA in R

`metaMDS()` in package(vegan)



"CCA" in R

cca() in package(vegan)

data(*dune*): Vegetation and Environment in Dutch Dune Meadows

- The *dune* meadow vegetation data has cover class values of 30 species on 20 sites. The corresponding environmental data frame *dune.env* is a data frame of 20 observations on the following 5 variables:

A1: a numeric vector of thickness of soil A1 horizon.

Moisture: an ordered factor with levels: 1 < 2 < 4 < 5.

Management: a factor with levels: BF (Biological farming), HF (Hobby farming), NM (Nature Conservation Management), and SF (Standard Farming).

Use: an ordered factor of land-use with levels: Hayfield < Haypastu < Pasture.

Manure: an ordered factor with levels: 0 < 1 < 2 < 3 < 4.

"CCA" in R

cca() in package(vegan)

```
> cca(dune ~ ., data=dune.env)
Call: cca(formula = dune ~ A1 + Moisture + Management + Use +
          Manure, data = dune.env)

Inertia Proportion Rank
Total         2.1153     1.0000
Constrained   1.5032     0.7106   12
Unconstrained 0.6121     0.2894    7
Inertia is mean squared contingency coefficient
Some constraints were aliased because they were collinear (redundant)

Eigenvalues for constrained axes:
  CCA1   CCA2   CCA3   CCA4   CCA5   CCA6   CCA7   CCA8   CCA9  CCA10
 0.4671 0.3410 0.1761 0.1532 0.0953 0.0703 0.0589 0.0499 0.0318 0.0260
 CCA11  CCA12
 0.0228 0.0108

Eigenvalues for unconstrained axes:
  CA1     CA2     CA3     CA4     CA5     CA6     CA7
0.27237 0.10876 0.08975 0.06305 0.03489 0.02529 0.01798
```

Canonical Correspondence Analysis (Optional topic)

- References provided on CourseWorks

SUMMARY
Things to check

- Number of canonical variate pairs
  - How many prove to be statistically/practically significant?
- Interpreting the canonical variates
  - Where is the meaning in the combinations of variables created?
- Importance of canonical variates
  - How strong is the correlation between the canonical variates?
  - What is the nature of the variate's relation to the individual variables in its own set? The other set?
- Canonical variate scores
  - If one did directly measure the variate, what would the subjects' scores be? (similar to the factor scores)

Limitations

1. By maximizing the correlation between the linear combinations of X and Y, there is a possibly of having the combination unexplainable.
2. Nonlinearity still causes problems: Classical → Regularized CCA
3. CCA is very sensitive to data involved
4. Correlation does not imply causality, necessary but not sufficient condition for causality.
5. It is more of a descriptive procedure, as being a simple correlation in the end.