Multivariate HW

Sarah Kell

February 8, 2018

```
library(vegan)

library(dummies)

data(dune)
data(dune.env) #dataframe of 20 observations (sites) on five variables
?dune

head(dune.env)

##    A1 Moisture Management      Use Manure
## 1 2.8        1         SF Haypastu      4
## 2 3.5        1         BF Haypastu      2
## 3 4.3        2         SF Haypastu      4
## 4 4.2        2         SF Haypastu      4
## 5 6.3        1         HF Hayfield      2
## 6 4.3        1         HF Haypastu      2
```

1. Conduct an indirect ordination on the dune plant community. Specifically, visually examine a NMDS plot using the bray-curtis distance metric. Below is some code to help you develop a potential plot that emphasizes the role of the environmental variable “Moisture”.

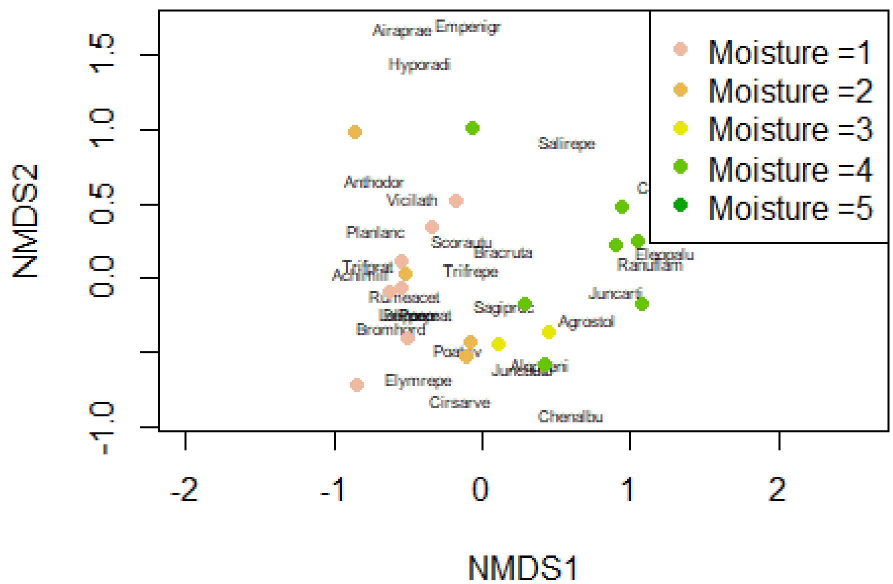
- Describe how you interpret the graphic- The dots represent the moisture level of the soil with 1 being the least and 5 being the most. The species distance to a moisture dot indicates the soil conditions in which it was found
- What is the goal of creating such a plot? You can get a quick idea of the moisture requirements for the 30 species
- Does this analysis suggest any interesting findings with respect to the dune vegetation? Species tend to either require little moisture or relatively moist soil

```
dune_mds = metaMDS(dune)

## Run 0 stress 0.1192678
## Run 1 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 0.02027031  max resid 0.0649565
## Run 2 stress 0.1192679
## Run 3 stress 0.1183186
## ... Procrustes: rmse 2.687289e-05  max resid 8.34341e-05
## ... Similar to previous best
## Run 4 stress 0.1183186
## ... Procrustes: rmse 4.248004e-05  max resid 0.000128741
```

```
## ... Similar to previous best
## Run 5 stress 0.1183186
## ... Procrustes: rmse 8.553937e-05  max resid 0.0002473271
## ... Similar to previous best
## Run 6 stress 0.2558641
## Run 7 stress 0.2378847
## Run 8 stress 0.1183186
## ... Procrustes: rmse 2.248173e-05  max resid 6.965894e-05
## ... Similar to previous best
## Run 9 stress 0.1192683
## Run 10 stress 0.1192678
## Run 11 stress 0.1192678
## Run 12 stress 0.1812938
## Run 13 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 3.488196e-06  max resid 1.114866e-05
## ... Similar to previous best
## Run 14 stress 0.1183186
## ... Procrustes: rmse 1.538435e-05  max resid 5.000909e-05
## ... Similar to previous best
## Run 15 stress 0.1183186
## ... Procrustes: rmse 1.799095e-06  max resid 5.199118e-06
## ... Similar to previous best
## Run 16 stress 0.1183186
## ... Procrustes: rmse 4.040116e-05  max resid 0.0001008035
## ... Similar to previous best
## Run 17 stress 0.1183186
## ... Procrustes: rmse 1.745177e-05  max resid 5.361259e-05
## ... Similar to previous best
## Run 18 stress 0.1192678
## Run 19 stress 0.1812957
## Run 20 stress 0.1192681
## *** Solution reached
```

```r
plot(dune_mds, type='n')
 text(dune_mds, 'sp', cex=.5)
 color_vect = rev(terrain.colors(6))[-1] # generate vector of colors
 points(dune_mds, 'sites', pch=19,
        col=color_vect[dune.env$Moisture])
 legend('topright', paste("Moisture =", 1:5, sep=''),
        col=color_vect, pch=19)
```



2. Use CCA to test hypotheses derived from MDS plot.

Specifically, carry out a test of the entire model (i.e., including all constrained axes) and also carry out tests at the scale of individual explanatory variables you included in your model if you included more than one variable. Plot your results.

Testing entire model - dune (y) ~ dune.env(x)

```
cca_dune = cca(dune ~ ., data = dune.env,scale=TRUE )
cca_dune
```

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

```
RsquareAdj(cca_dune) #AdjR2=0.23
```

```
## $r.squared
## [1] 0.7106267
##
## $adj.r.squared
## [1] 0.2389421
```

```
anova(cca_dune, permutations = 1000) #F=1.4325, p=0.02, good model fit
```

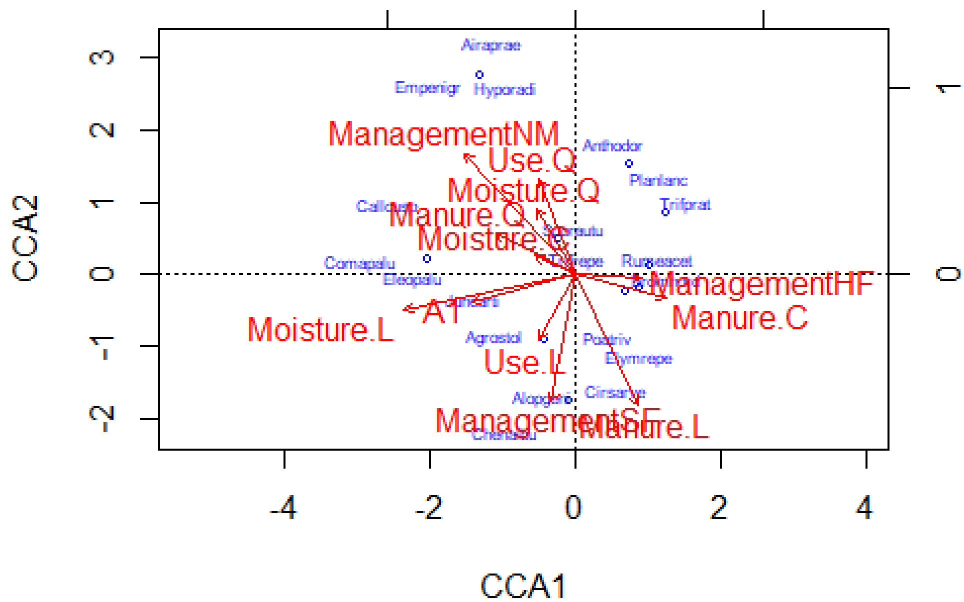
```
## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ A1 + Moisture + Management + Use + Manure,
data = dune.env, scale = TRUE)
##          Df ChiSquare      F  Pr(>F)
## Model    12    1.5032 1.4325 0.02098 *
## Residual  7    0.6121
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(cca_dune, by='margin', permutations = 1000) #No individual variables
are significant, A1 and moisture appear to be most important
```

```
## Permutation test for cca under reduced model
## Marginal effects of terms
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ A1 + Moisture + Management + Use + Manure,
data = dune.env, scale = TRUE)
##            Df ChiSquare      F Pr(>F)
## A1          1   0.11070 1.2660 0.2438
## Moisture    3   0.31587 1.2041 0.1968
## Management  2   0.15882 0.9081 0.5475
## Use         2   0.13010 0.7439 0.7852
## Manure      3   0.25490 0.9717 0.4855
## Residual    7   0.61210
```

Plotting Data

```
plot(cca_dune, type='n', scaling=1)
orditorp(cca_dune, display='sp', cex=0.5, scaling=1, col='blue')
text(cca_dune, display='bp', col='red')
```



Models looking at individual variables

```
cca_dune2 = cca(dune ~ A1 + Moisture, data = dune.env,scale=TRUE)
cca_dune2

## Call: cca(formula = dune ~ A1 + Moisture, data = dune.env, scale =
## TRUE)
##
##                 Inertia Proportion Rank
## Total            2.1153     1.0000
## Constrained      0.7437     0.3516    4
## Unconstrained    1.3715     0.6484   15
## Inertia is mean squared contingency coefficient
##
## Eigenvalues for constrained axes:
##   CCA1   CCA2   CCA3   CCA4
## 0.4314 0.1350 0.1066 0.0706
##
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7    CA8    CA9   CA10
## 0.3843 0.2140 0.1601 0.1226 0.0989 0.0902 0.0751 0.0605 0.0539 0.0456
```

```
##    CA11   CA12   CA13   CA14   CA15
## 0.0209 0.0154 0.0125 0.0096 0.0081
```

```r
RsquareAdj(cca_dune2) #AdjR2 = 0.187, decreased from entire model
```

```
## $r.squared
## [1] 0.3516042
##
## $adj.r.squared
## [1] 0.1873396
```

```r
anova(cca_dune2, permutations = 1000) #F = 2.0335, p=0.003 better model fit
than all variables
```

```
## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ A1 + Moisture, data = dune.env, scale = TRUE)
##          Df ChiSquare      F   Pr(>F)
## Model     4   0.74374 2.0335 0.000999 ***
## Residual 15   1.37153
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(cca_dune2, by='margin', permutations = 1000)
```

```
## Permutation test for cca under reduced model
## Marginal effects of terms
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ A1 + Moisture, data = dune.env, scale = TRUE)
##          Df ChiSquare      F   Pr(>F)
## A1        1   0.11543 1.2624 0.207792
## Moisture  3   0.51898 1.8920 0.003996 **
## Residual 15   1.37153
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Soil A1 horizon only

```r
cca_dune3 = cca(dune ~ A1, data = dune.env, scale=TRUE)
cca_dune3
```

```
## Call: cca(formula = dune ~ A1, data = dune.env, scale = TRUE)
##
##               Inertia Proportion Rank
## Total          2.1153     1.0000
## Constrained    0.2248     0.1063    1
## Unconstrained  1.8905     0.8937   18
## Inertia is mean squared contingency coefficient
```

```
##
## Eigenvalues for constrained axes:
##    CCA1
## 0.22476
##
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7    CA8
## 0.4073 0.3938 0.2519 0.1662 0.1314 0.0975 0.0898 0.0747
## (Showed only 8 of all 18 unconstrained eigenvalues)
```

```r
RsquareAdj(cca_dune3) #AdjR2 = 0.058,
```

```
## $r.squared
## [1] 0.1062564
##
## $adj.r.squared
## [1] 0.05859232
```

```r
anova(cca_dune3, permutations = 1000) #F = 2.14, p=0.03
```

```
## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ A1, data = dune.env, scale = TRUE)
##          Df ChiSquare    F  Pr(>F)
## Model     1   0.22476 2.14 0.03097 *
## Residual 18   1.89050
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Moisture only

```r
cca_dune4 = cca(dune ~ Moisture, data = dune.env, scale=TRUE)
cca_dune4
```

```
## Call: cca(formula = dune ~ Moisture, data = dune.env, scale =
## TRUE)
##
##                 Inertia Proportion Rank
## Total            2.1153     1.0000
## Constrained      0.6283     0.2970    3
## Unconstrained    1.4870     0.7030   16
## Inertia is mean squared contingency coefficient
##
## Eigenvalues for constrained axes:
##   CCA1   CCA2   CCA3
## 0.4187 0.1330 0.0766
##
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7    CA8    CA9   CA10
```

```
## 0.4098 0.2259 0.1761 0.1234 0.1082 0.0908 0.0859 0.0609 0.0566 0.0467
##   CA11   CA12   CA13   CA14   CA15   CA16
## 0.0419 0.0201 0.0143 0.0099 0.0085 0.0080

RsquareAdj(cca_dune4) #AdjR2 = 0.171,

## $r.squared
## [1] 0.2970359
##
## $adj.r.squared
## [1] 0.1713802

anova(cca_dune4, permutations = 1000) #F = 2.25, p=0.0009  BEST FIT

## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ Moisture, data = dune.env, scale = TRUE)
##          Df ChiSquare      F   Pr(>F)
## Model     3   0.62831 2.2536 0.001998 **
## Residual 16   1.48695
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Management only

```
cca_dune5 = cca(dune ~ Management, data = dune.env, scale=TRUE)
cca_dune5

## Call: cca(formula = dune ~ Management, data = dune.env, scale =
## TRUE)
##
##                Inertia Proportion Rank
## Total           2.1153     1.0000
## Constrained     0.6038     0.2855    3
## Unconstrained   1.5114     0.7145   16
## Inertia is mean squared contingency coefficient
##
## Eigenvalues for constrained axes:
##   CCA1   CCA2   CCA3
## 0.3186 0.1825 0.1027
##
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7    CA8    CA9   CA10
## 0.4474 0.2030 0.1630 0.1346 0.1294 0.0949 0.0790 0.0653 0.0500 0.0432
##   CA11   CA12   CA13   CA14   CA15   CA16
## 0.0387 0.0239 0.0177 0.0092 0.0080 0.0042

RsquareAdj(cca_dune5) #AdjR2 = 0.157,
```

```
## $r.squared
## [1] 0.285467
##
## $adj.r.squared
## [1] 0.157582
```

```
anova(cca_dune5, permutations = 1000) #F = 2.13, p=0.0009
```

```
## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ Management, data = dune.env, scale = TRUE)
##          Df ChiSquare      F   Pr(>F)
## Model     3   0.60384 2.1307 0.002997 **
## Residual 16   1.51143
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Land Use

```
cca_dune6 = cca(dune ~ Use, data = dune.env, scale=TRUE)
cca_dune6
```

```
## Call: cca(formula = dune ~ Use, data = dune.env, scale = TRUE)
##
##                 Inertia Proportion Rank
## Total            2.1153     1.0000
## Constrained      0.2507     0.1185    2
## Unconstrained    1.8645     0.8815   17
## Inertia is mean squared contingency coefficient
##
## Eigenvalues for constrained axes:
##    CCA1    CCA2
## 0.15249 0.09825
##
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7    CA8
## 0.4929 0.3104 0.2322 0.1755 0.1420 0.1030 0.0782 0.0758
## (Showed only 8 of all 17 unconstrained eigenvalues)
```

```
RsquareAdj(cca_dune6) #AdjR2 = 0.017,
```

```
## $r.squared
## [1] 0.1185381
##
## $adj.r.squared
## [1] 0.01883057
```

```
anova(cca_dune6, permutations = 1000) #F = 1.14, p=0.228 Not significant
```

```
## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ Use, data = dune.env, scale = TRUE)
##          Df ChiSquare      F Pr(>F)
## Model     2   0.25074 1.1431 0.2058
## Residual 17   1.86452
```

Manure

```
cca_dune7 = cca(dune ~ Manure, data = dune.env, scale=TRUE)
cca_dune7
```

```
## Call: cca(formula = dune ~ Manure, data = dune.env, scale = TRUE)
##
##                Inertia Proportion Rank
## Total           2.1153     1.0000
## Constrained     0.6116     0.2891    4
## Unconstrained   1.5037     0.7109   15
## Inertia is mean squared contingency coefficient
##
## Eigenvalues for constrained axes:
##   CCA1   CCA2   CCA3   CCA4
## 0.3472 0.1199 0.0984 0.0461
##
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7    CA8    CA9   CA10
## 0.4450 0.2475 0.1495 0.1308 0.1083 0.1008 0.0791 0.0650 0.0593 0.0327
##   CA11   CA12   CA13   CA14   CA15
## 0.0319 0.0249 0.0147 0.0086 0.0056
```

```
RsquareAdj(cca_dune7) #AdjR2 = 0.108,
```

```
## $r.squared
## [1] 0.2891171
##
## $adj.r.squared
## [1] 0.1092444
```

```
anova(cca_dune7, permutations = 1000) #F = 1.53, p=0.019
```

```
## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 1000
##
## Model: cca(formula = dune ~ Manure, data = dune.env, scale = TRUE)
##          Df ChiSquare      F  Pr(>F)
## Model     4   0.61156 1.5251 0.01798 *
## Residual 15   1.50370
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3. Do your two analyses agree with one another or complement one another or do these two analyses seem to be suggesting different take home messages? The two analyses agree in that moisture is the most important variable in determining species of dune vegetation.

*Which analysis do you find to be more useful? (NMDS plot vs cca) I like both equally as companions to one another. The NMDS plot provides a quick idea of the relevance of moisture and which moisture levels are most important. CCA provides more statistical relevance, in determing the F and p values of moisture.