## PCA (rda function) output

```
prin_comp<-rda(community)
summary(prin_comp)
                   Inertia     Rank
Total              1.432
Unconstrained      1.432   10
Inertia is variance

Eigenvalues for unconstrained axes:
     PC1      PC2      PC3      PC4      PC5      PC6      PC7      PC8      PC9     PC10
0.654125 0.462758 0.146771 0.136343 0.014029 0.008240 0.004094 0.003824 0.001463 0.000673
```

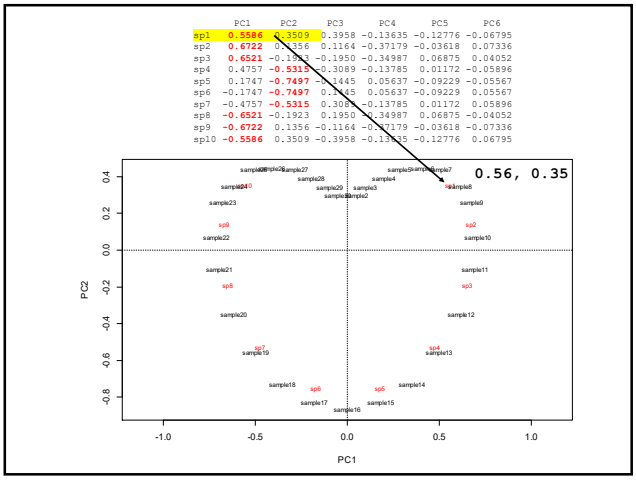
- Within the object returned by the function rda(), results are within CA object (?cca.object):
- prin_comp$CA contains
  - eig: eigenvalues
  - u: site scores
  - v: variable scores (eigenvectors)
  - u.eig: site scores scaled by eigenvalues
  - v.eig: variable scores scaled by eigenvalues
  - Tot.chi: total inertia
  - Xbar: centered and scaled data matrix

## PCA (rda function) output

- v: variable scores or loadings, listed as “species scores” by rda function
  - Represent the relationship between each variable and the axes.

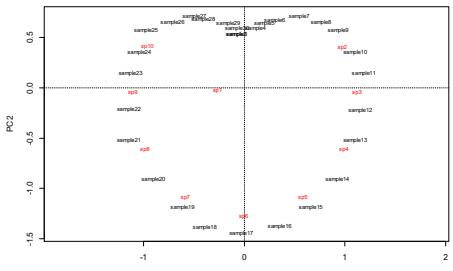
| | PC1 | PC2 | PC3 | PC4 | PC5 | PC6 |
|---|---|---|---|---|---|---|
| sp1 | **0.5586** | 0.3509 | 0.3958 | -0.13635 | -0.12776 | -0.06795 |
| sp2 | **0.6722** | 0.1356 | 0.1164 | -0.37179 | -0.03618 | 0.07336 |
| sp3 | **0.6521** | -0.1923 | -0.1950 | -0.34987 | 0.06875 | 0.04052 |
| sp4 | 0.4757 | **-0.5315** | -0.3089 | -0.13785 | 0.01172 | -0.05896 |
| sp5 | 0.1747 | **-0.7497** | -0.1445 | 0.05637 | -0.09229 | -0.05567 |
| sp6 | -0.1747 | **-0.7497** | 0.1445 | 0.05637 | -0.09229 | 0.05567 |
| sp7 | -0.4757 | **-0.5315** | 0.3089 | -0.13785 | 0.01172 | 0.05896 |
| sp8 | **-0.6521** | -0.1923 | 0.1950 | -0.34987 | 0.06875 | -0.04052 |
| sp9 | **-0.6722** | 0.1356 | -0.1164 | -0.37179 | -0.03618 | -0.07336 |
| sp10 | **-0.5586** | 0.3509 | -0.3958 | -0.13635 | -0.12776 | 0.06795 |

- Look for where each variable has its highest (greatest magnitude) loading. Remember that early axes are more important (explain more variation).




- Repeat the same analysis, this time with species 1 randomized (sampled without replacement)

| | PC1 | PC2 | PC3 | PC4 | PC5 | PC6 |
|---|---|---|---|---|---|---|
| sp1 | **-0.259840** | **-0.01249** | -1.018452 | 0.66267 | -0.35883 | -0.01944 |
| sp2 | 0.978973 | 0.41127 | 0.324628 | 0.51865 | 0.36163 | -0.18337 |
| sp3 | 1.121840 | -0.04099 | 0.458173 | 0.39966 | -0.14316 | 0.04228 |
| sp4 | 0.993676 | -0.60484 | 0.284719 | 0.08457 | -0.45551 | 0.11026 |
| sp5 | 0.585036 | -1.08172 | 0.007771 | -0.14915 | -0.33593 | -0.04522 |
| sp6 | -0.003748 | -1.26824 | -0.119410 | -0.09780 | 0.04745 | -0.16286 |
| sp7 | -0.584121 | -1.08015 | 0.066570 | 0.12869 | 0.34952 | -0.02035 |
| sp8 | -0.984533 | -0.60254 | 0.320699 | 0.41218 | 0.21392 | 0.10098 |
| sp9 | -1.101661 | -0.03827 | 0.540038 | 0.32122 | -0.20250 | 0.03227 |
| sp10 | -0.959497 | 0.41386 | 0.453136 | -0.05608 | -0.57776 | -0.17342 |



## How do you know what "good" loadings are?

*Ecology*, 84(9), 2003, pp. 2347–2363
© 2003 by the Ecological Society of America

GIVING MEANINGFUL INTERPRETATION TO ORDINATION AXES: ASSESSING LOADING SIGNIFICANCE IN PRINCIPAL COMPONENT ANALYSIS

PEDRO R. PERES-NETO,[1] DONALD A. JACKSON, AND KEITH M. SOMERS
*Department of Zoology, University of Toronto, Toronto, Ontario, Canada M5S 3G5*

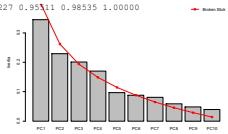
- Axis score-variable correlations are difficult to interpret
- Convention is some arbitrary eigenvalue cut off values
- Permutation approach (see script and paper) to assess "significance"
  - Randomizes matrix (sample without replacement), calculates loadings, builds distribution

## How well does the analysis find and summarize structure?

```
Partitioning of variance:
             Inertia Proportion
Total          1.432          1
Unconstrained  1.432          1

Eigenvalues, and their contribution to the variance

Importance of components:
                          PC1    PC2    PC3     PC4     PC5     PC6     PC7     PC8     PC9     PC10
Eigenvalue            0.6541 0.4628 0.1468 0.13634 0.01403 0.00824 0.004094 0.003924 0.001463 0.0006734
Proportion Explained  0.4567 0.3231 0.1025 0.09519 0.00979 0.00575 0.002860 0.002670 0.001020 0.0004700
Cumulative Proportion 0.4567 0.7798 0.8822 0.97743 0.98723 0.99298 0.995840 0.998510 0.999530 1.0000000
```

Randomize the same data matrix (sample without replacement) and repeat the analysis.

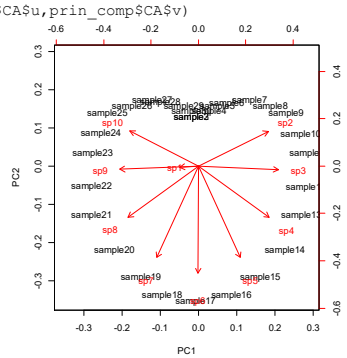
```
Partitioning of variance:
             Inertia Proportion
Total          1.368          1
Unconstrained  1.368          1

Eigenvalues, and their contribution to the variance

Importance of components:
                          PC1    PC2    PC3    PC4     PC5     PC6     PC7     PC8     PC9     PC10
Eigenvalue            0.3119 0.2264 0.2029 0.1547 0.12884 0.11558 0.09405 0.07230 0.04137 0.02005
Proportion Explained  0.2280 0.1655 0.1483 0.1131 0.09417 0.08448 0.06874 0.05284 0.03024 0.01466
Cumulative Proportion 0.2280 0.3935 0.5418 0.6549 0.74905 0.83353 0.90227 0.95511 0.98535 1.00000
```




## Biplot function

- Function biplot (stats package, should load automatically) will plot sample points and loadings/scores as vectors.

```
prin_comp<-rda(community,scaled=T)
biplot(prin_comp$CA$u,prin_comp$CA$v)
```




## PCA

- Sample scores are calculated from centered and scaled data multiplied by the eigenvector matrix.
- In an rda object, these are returned as $CA$u, the centered and scaled data is returned as $CA$Xbar.

```
Xbar (29x10):
                sp1          sp2          sp3          sp4
sample2   1.950211228 -0.692074135 -0.692074135 -0.692074135
sample3   1.860177537 -0.692074135 -0.692074135 -0.692074135
sample4  -0.692074135 -0.008200361 -0.692074135 -0.692074135
sample5  -0.692074135  0.629068547 -0.692074135 -0.692074135
sample6   1.596212114  1.176303763 -0.008200361 -0.692074135
sample7  -0.692074135  1.596212114  0.629068547 -0.692074135
sample8  -0.692074135  1.860177537  1.176303763 -0.008200361
sample9  -0.692074135  1.950211228  1.596212114  0.629068547
```

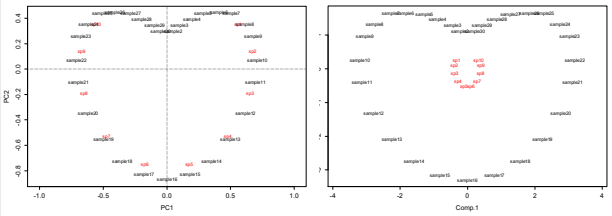
%*% = sample scores (29x10)

```
Eigenvectors (10x10)
               [,1]         [,2]         [,3]        [,4]
 [1,]  0.097799707 -0.005574267  0.710938526  0.60503086
 [2,] -0.368469932  0.183598130 -0.226609013  0.47353761
 [3,] -0.422243182 -0.018299122 -0.319831541  0.36489940
 [4,] -0.374004178 -0.270009949 -0.198750294  0.07721397
 [5,] -0.220198463 -0.482895556 -0.005424313 -0.13617713
 [6,]  0.001410515 -0.566163776  0.083355083 -0.08929034
 [7,]  0.219853824 -0.482195989 -0.046469557  0.11749890
 [8,]  0.370562733 -0.268984385 -0.223866728  0.37633225
 [9,]  0.414647983 -0.017083175 -0.376977741  0.29328474
[10,]  0.361139794  0.184752479 -0.316315399 -0.05119929
```

- Because they are centered and scaled, PCA points are centered at the origin.

## Scores function

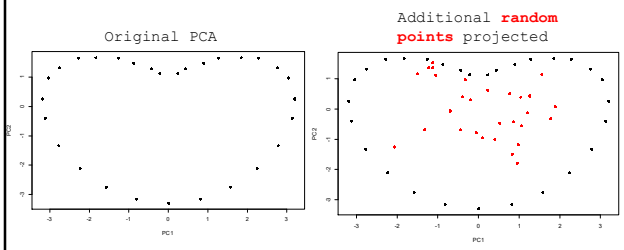
- Different software/functions will scale points and variable scores differently.
- The scores function (vegan) returns scores for samples and variables.
- scores(prin_comp) – returns an object with sample and site scores




## Projecting Scores

- This also means that the PCA can serve as a model. New data can be "projected" into PCA space without changing the underlying model (re-running the PCA).
- Function predict will project new data (must have the same variables with the same names)

```
predict(prin_comp,new_data)
```

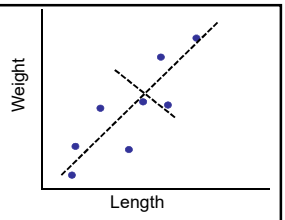


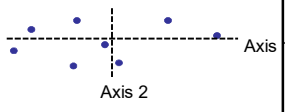

## Table 9.1 from Legendre and Legendre (1998)

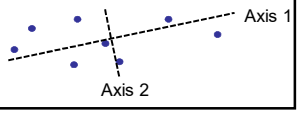
| Method | Distance | Variables |
|---|---|---|
| PCA | Euclidean | Quantitative, linear relationships assumed, beware of double-zeroes |
| PCoA | Any | Quantitative, qualitative or mixed |
| NMDS | Any | Quantitative, qualitative or mixed |
| CA | $X^2$ | Non-negative, quantitative or binary |
| FA | Euclidean | Quantitative, linear relationships assumed, beware of double-zeroes |

## Axis Rotation

- PCA constructs the new variables (axes) to maximize variance explained. Loadings are used to relate variables to axes.
- What if you want the relationship between variables and axes to be clearer?
- You can have the analysis "rotate" axes to maximize loadings with variables.
- The tradeoff is that less overall variance will be explained



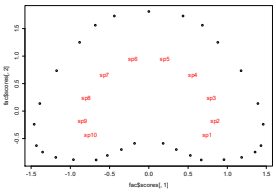



## Factor Analysis

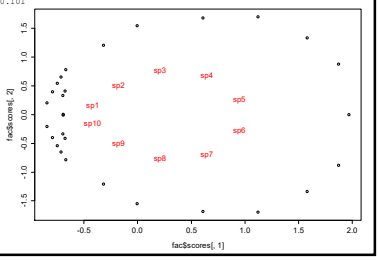
- Function factanal()
  - Data matrix
  - Number of factors
  - Method of calculating scores
  - Type of rotation (none = PCA)
- First, try factor analysis with no rotation. Results look very similar to a PCA

```
>fac<-factanal(community,6,scores="regression",rotation="none")
>fac

Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
SS loadings     4.523   3.215   1.033   0.939   0.096   0.044
Proportion Var  0.452   0.322   0.103   0.094   0.010   0.004
Cumulative Var  0.452   0.774   0.877   0.971   0.981   0.985
```



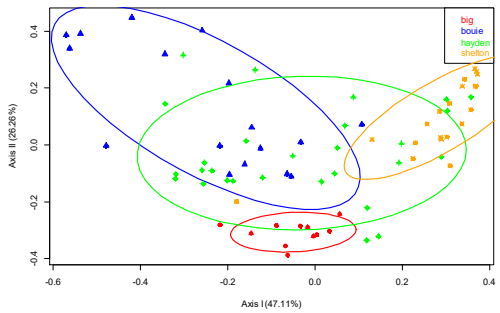

## Factor Analysis

```
> fac<-factanal(community,6,scores="regression",rotation="varimax")
> fac

                Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
SS loadings       3.144   2.820   2.731   0.932   0.174   0.051
Proportion Var    0.314   0.282   0.273   0.093   0.017   0.005
Cumulative Var    0.314   0.596   0.869   0.963   0.980   0.985

Loadings:
      Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
sp1   -0.419   0.153   0.849   0.121   0.244
sp2   -0.178   0.507   0.683   0.441   0.130
sp3    0.210   0.762   0.394   0.460
sp4    0.647   0.687   0.147   0.211          -0.101
sp5    0.950   0.276                          -0.111
sp6    0.950  -0.276                           0.111
sp7    0.647  -0.687  -0.147   0.211           0.101
sp8    0.210  -0.762  -0.394   0.460
sp9   -0.178  -0.507  -0.683   0.441   0.130
sp10  -0.419  -0.153  -0.849   0.121   0.244
```




## Sample Grouping

- Function dataEllipse (car package) will add a centroid and confidence ellipse to a group of points
- 75% confidence ellipses for groups (creek_factor):
  - `dataEllipse(prin_coord$points[,1:2],add=T,levels=0.75,group=creek_factor)`




## Sample Grouping

- Another approach is to plot minimum convex polygons for groups of samples.
- Function chull (grDevices package) returns the points that form a polygon for a groups of samples.

```
> chull(prin_coord$points[1:15,1:2])
[1]  2  1  3  4  5  7 10 11 12 13 14 15
```

- Function polygon (graphics package) will add a polygon to your plot.

```
grp1<-prin_coord$points[creek_factor=="big",1:2]
polygon(grp1[chull(grp1),1:2],col=rgb(1,0,0,0.3))
```