



Klasyfikacja numeryczna

Czym jest klasyfikacja numeryczna?

- #Według niektórych badaczy (np. prof. Richard Telford) najprostsza metoda ordynacji
- #Dzieli zbiór danych na grupy
- #Ułatwia wyodrębnienie zasadniczych cech przedmiotów badania
- #Redukcja dużej liczby danych pierwotnych do kilku podstawowych kategorii
- #Zmniejszenie nakładu pracy i czasu analiz (mając 50 obiektów istnieje 10^{80} możliwych sposobów podziału obiektów)
- #Odkrycie nieznanej struktury danych – analiza bardzo przydatna dla taksonomów przy konstrukcji drzew filogenetycznych oraz dla fitosocjologów

Strategie

#Metoda najbliższego sąsiada (single)

- zaliczamy obiekt do tej klasy, do której należy większość z jego K najbliższych sąsiadów

#Metoda najdalszego sąsiada (complete)

- zaliczamy obiekt do tej klasy, do której należy większość z jego K najdalszych sąsiadów

#Metoda centroidów (centroid)

- Odległość między centroidami dwóch klas
- Centroid to punkt, który jest średnią wszystkich zmiennych w klasie

I inne...

Tworzenie dendrogramu

#Snowbeds1 – dane składu gatunkowego roślinności wyleżysk wysokogórskich

#legenda: m58k – dane z początku XX wieku;

m58m – dane powtórzone w 2015 roku

#poletka w kolumnach; gatunki w wierszach:

[illegible]

#transpozycja kolumn z wierszami

snowbeds2<-t(snowbeds1)

#teraz gatunki w kolumnach, a próby w wierszach:

```
> snowbeds2
      Abi.alb Ace.pse Aco.fir Aco.var Ade.all Ado.mos Aeg.pod Agr.sto Agr.alp Agr.can Agr.rup
m58k      0      0      0      0      0      0      0      0      0      0      0
m58n      0      0      0      0      0      0      0      0      0      0      0
m67k      0      0      0      0      0      0      0      0      0      0      0
      Agr.cap Alc.gla Alc.fla Alc.xan Aln.inc Ane.nar Ang.arc Ang.syl Ant.car Anx.alp Ara.alp
m58k      0      0      0      0      0      0      0      0      0      2      0
m58n      0      0      0      0      0      2      0      0      0      2      0
m67k      0      0      0      0      0      0      0      0      0      0      0
      Art.eri Aru.dio Asp.vir Ast.bel Asn.maj Ath.dis Ath.fil Ave.ver Bar.alp Bet.pub Ble.spi
m58k      0      0      0      0      0      0      0      0      0      0      0
m58n      0      0      0      0      0      0      0      2      0      0      0
m67k      0      0      0      0      0      0      0      0      0      0      0
      Bot.lun Cal.aru Cal.epi Cal.vil Cah.cor Cll.vul Cth.pal Cam.alp Cam.coc Cam.rot Car.ama
m58k      0      0      0      0      0      0      0      0      0      0      0
m58n      0      0      0      0      0      0      0      2      0      0      0
m67k      0      0      0      0      0      0      0      0      0      0      0
      Car.opi Car.fle Car.gla Car.imp Crd.bor Crd.hal Crd.neg Cdu.cri Crx.atr Crx.cut Crx.dig
m58k      0      0      0      0      0      0      2      0      0      0      0
m58n      0      0      0      0      0      0      0      0      0      0      0
m67k      0      0      0      0      0      0      2      0      0      0      0
```

#Stworzenie macierzy niepodobieństwa (obiekt typu `dist`)

```
library(vegan)
```

```
snowbeds.dist<-vegdist(snowbeds2, method="bray")
```

Usage

```
vegdist(x, method="bray", binary=FALSE, diag=FALSE,  
        na.rm = FALSE, ...)
```

Arguments

<code>x</code>	Community data matrix.
<code>method</code>	Dissimilarity index, partial match to "manhattan", "euclidean", "canberra", "clark", "bray", "kulczynski", "jaccard", "gower", "altGower", "morisita", "horn", "mountford", "raup", "binomial", "chao", "cao" or "mahalanobis".
<code>binary</code>	Perform presence/absence standardization before analysis using <u>decostand</u> .
<code>diag</code>	Compute diagonals.
<code>upper</code>	Return only the upper diagonal.
<code>na.rm</code>	Pairwise deletion of missing observations when computing dissimilarities.
<code>...</code>	Other parameters. These are ignored, except in <code>method="gower"</code> which accepts <code>range.global</code> parameter of <u>decostand</u> .

#Najczęściej używane w naukach biologicznych miary niepodobieństwa składu gatunkowego to:

- Wskaźnik niepodobieństwa Bray-Curtisa (dla danych binarnych/niebinarnych)
- Wskaźnik niepodobieństwa Sorensena (dla danych binarnych) – o nim później

snowbeds.dist

#macierz niepodobieństwa Bray-Curtisa pomiędzy próbami (każda z każdą)
#współczynnik niepodobieństwa mieści się pomiędzy 0 a 1,
gdzie 0 oznacza całkowite podobieństwo, a 1 całkowite niepodobieństwo
składu gatunkowego pomiędzy próbami

```
> snowbeds.dist
      m58k    m58n    m67k    m67n    m32k    m32n    m85k    m85n
m58n 0.6406250
m67k 0.3541667 0.7121212
m67n 0.4725275 0.6220472 0.2421053
m32k 0.4363636 0.6575342 0.4912281 0.5596330
m32n 0.6065574 0.5569620 0.6031746 0.5206612 0.5428571
m85k 0.3529412 0.5797101 0.5849057 0.6435644 0.4833333 0.5757576
m85n 0.5047619 0.5035461 0.6146789 0.6346154 0.5121951 0.5407407 0.3739130
m8ak 0.3465347 0.5912409 0.4285714 0.4600000 0.3277311 0.4809160 0.3153153 0.4385965
m8an 0.4653465 0.6058394 0.6190476 0.6000000 0.4957983 0.6183206 0.4414414 0.3508772
m30k 0.3800000 0.6617647 0.4807692 0.5959596 0.4915254 0.6000000 0.4000000 0.3982301
m30n 0.5841584 0.5620438 0.5047619 0.3600000 0.5798319 0.4045802 0.5855856 0.5614035
m122k 0.4608696 0.6026490 0.5462185 0.5789474 0.4135338 0.5172414 0.2960000 0.4687500
m122n 0.6160000 0.4285714 0.6899225 0.5967742 0.6503497 0.5225806 0.6000000 0.6086957
m100k 0.2323232 0.6444444 0.3592233 0.4285714 0.3675214 0.5658915 0.4311927 0.5178571
m100n 0.4925373 0.5529412 0.4202899 0.4135338 0.5526316 0.4390244 0.6111111 0.6190476
m107k 0.3877551 0.7164179 0.5882353 0.6701031 0.5000000 0.6562500 0.4259259 0.5135135
m107n 0.5407407 0.5204678 0.6690647 0.6567164 0.5294118 0.6121212 0.5310345 0.5000000
m45k 0.3333333 0.6956522 0.2264151 0.3663366 0.4500000 0.5151515 0.5357143 0.5826087
m45n 0.5125000 0.5510204 0.4634146 0.5345912 0.4943820 0.5368421 0.5058824 0.4450867
m21k 0.5897436 0.6315789 0.5365854 0.4805195 0.6250000 0.6111111 0.6136364 0.7142857
m21n 0.5833333 0.4393939 0.5800000 0.4315789 0.5964912 0.4285714 0.6226415 0.6513761
m25Bk 0.6969697 0.7843137 0.7142857 0.6615385 0.8571429 0.7916667 0.7105263 0.7468354
m25Bn 0.8208955 0.5922330 0.7746479 0.6363636 0.8823529 0.7731959 0.7922078 0.7750000
m25Ak 0.6666667 0.8333333 0.6875000 0.6271186 0.8461538 0.7777778 0.7142857 0.7808219
```

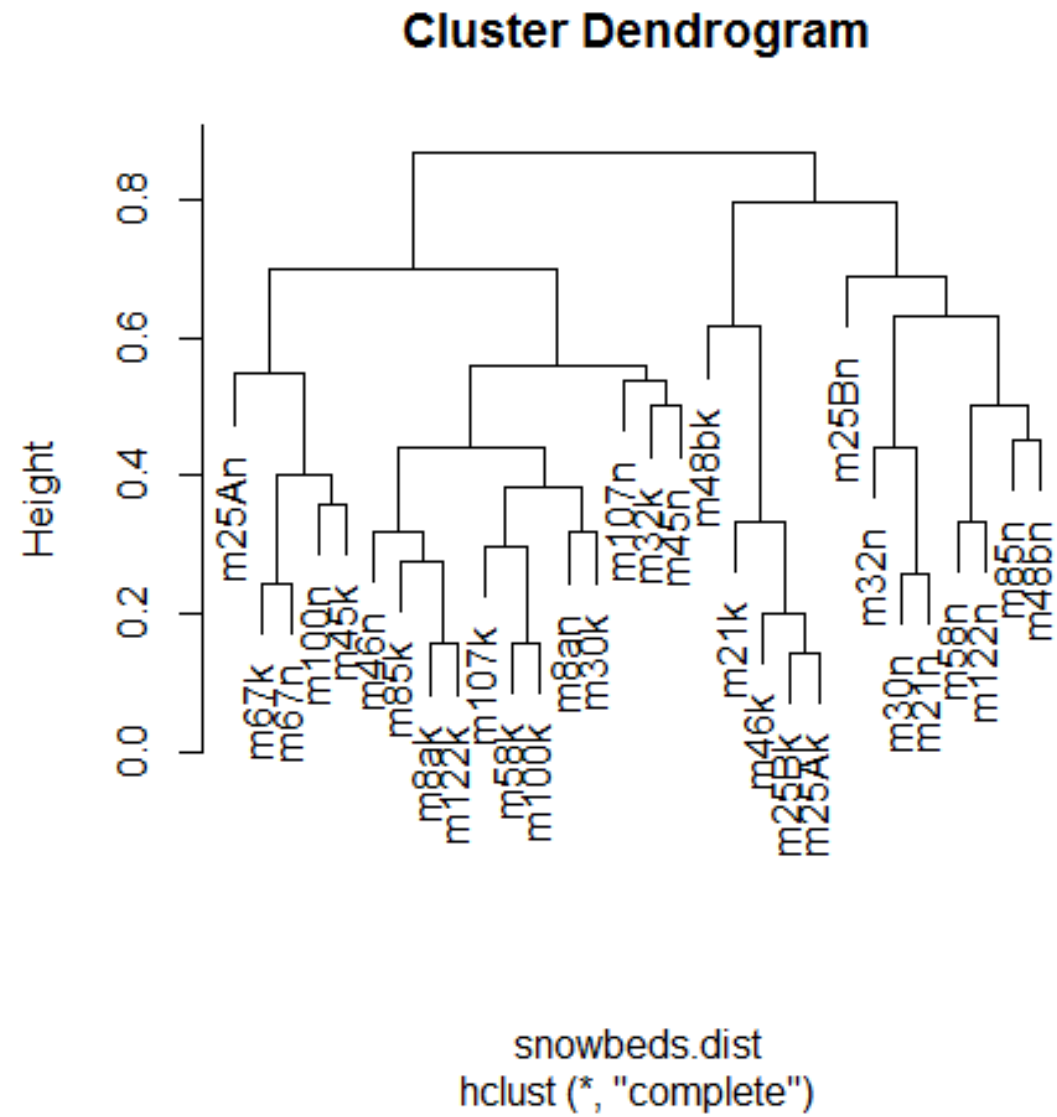

#tworzenie obiektu z dendrogramem

```
hc.snowbeds<-hclust(snowbeds.dist, "complete")
```

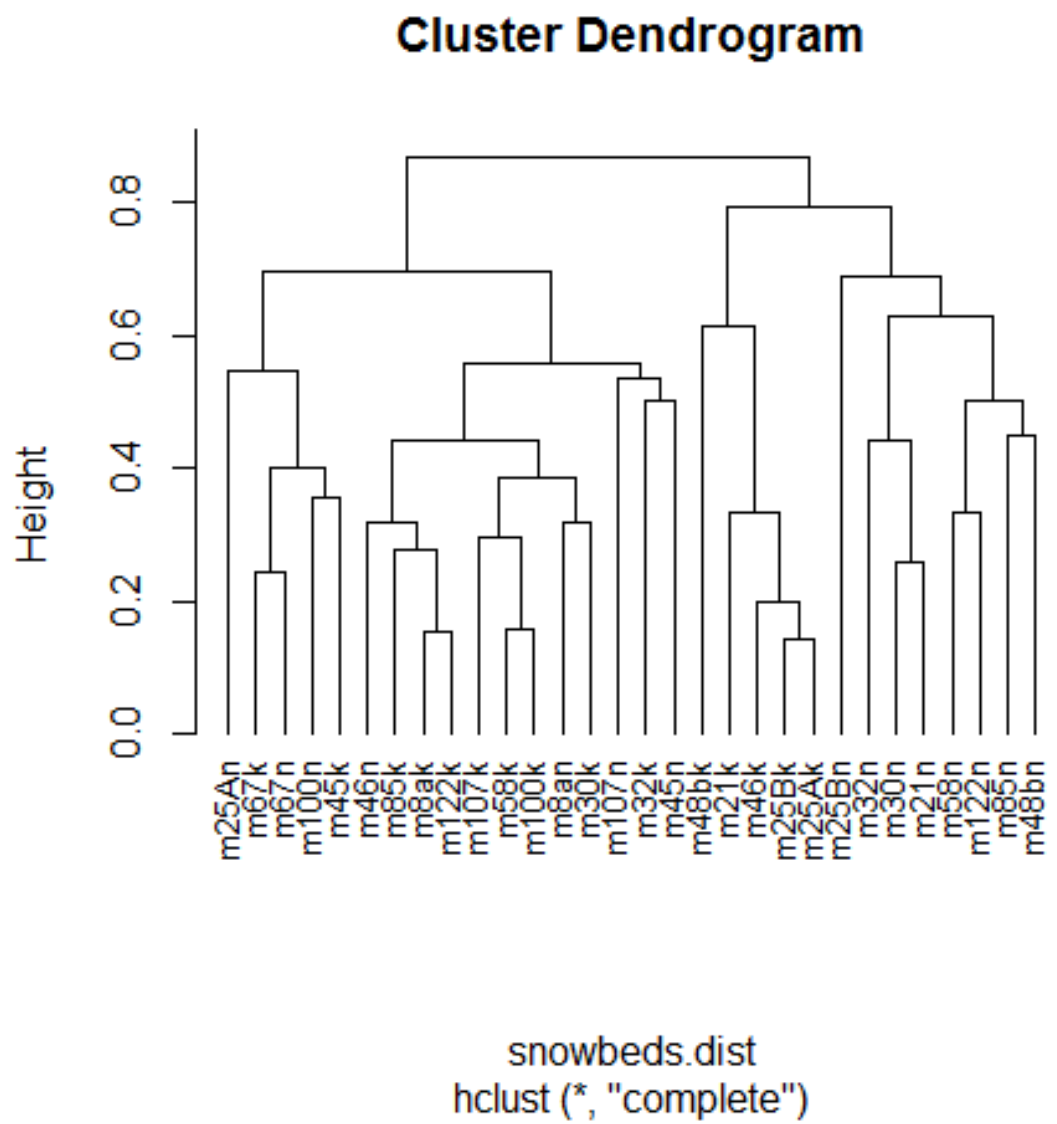
Arguments

<code>d</code>	a dissimilarity structure as produced by <code>dist</code> .
<code>method</code>	the agglomeration method to be used. This should be (an unambiguous abbreviation of) one of "ward.D", "ward.D2", "single", "complete", "average" (= UPGMA), "mcquitty" (= WPGMA), "median" (= WPGMC) or "centroid" (= UPGMC).
<code>members</code>	NULL or a vector with length size of <code>d</code> . See the 'Details' section.
<code>x</code>	an object of the type produced by <code>hclust</code> .
<code>hang</code>	The fraction of the plot height by which labels should hang below the rest of the plot. A negative value will cause the labels to hang down from 0.
<code>check</code>	logical indicating if the <code>x</code> object should be checked for validity. This check is not necessary when <code>x</code> is known to be valid such as when it is the direct result of <code>hclust()</code> . The default is <code>check=TRUE</code> , as invalid inputs may crash R due to memory violation in the internal C plotting code.

```
#tworzenie wykresu  
plot(hc.snowbeds)
```



#wyrównanie nazw prób do jednej linii i zmiana wielkości czcionki
`plot(hc.snowbeds, hang = -1, cex = 0.8)`



Wskaźnik Sorensena

```
snowbeds1<-read.table("snowbeds1.csv", sep=";", dec="," ,  
header=TRUE)
```

```
#Transpozycja kolumn z wierszami (w kolumnach gatunki, w wierszach  
powierzchnie badawcze/próby)
```

```
snowbeds2<-as.data.frame(t(snowbeds1))
```

```
#Konieczna transformacja do danych binarnych
```

```
snowbeds3<-decostand(snowbeds2, method='pa')
```

```
library(betapart)
beta0<-beta.pair(snowbeds3, index.family = 'sorensen')
```

Value

The function returns a list with three dissimilarity matrices. For `index.family="sorensen"` the three matrices are:

`beta.sim` `dist` object, dissimilarity matrix accounting for spatial turnover (replacement), measured as Simpson pair-wise dissimilarity

`beta.sne` `dist` object, dissimilarity matrix accounting for nestedness-resultant dissimilarity, measured as the nestedness-fraction of Sorensen pair-wise dissimilarity

`beta.sor` `dist` object, dissimilarity matrix accounting for total dissimilarity, measured as Sorensen pair-wise dissimilarity (a monotonic transformation of beta diversity)

For `index.family="jaccard"` the three matrices are:

`beta.jtu` `dist` dissimilarity matrix accounting for spatial turnover, measured as the turnover-fraction of Jaccard pair-wise dissimilarity

`beta.jne` `dist` object, dissimilarity matrix accounting for nestedness-resultant dissimilarity, measured as the nestedness-fraction of Jaccard pair-wise dissimilarity

`beta.jac` `dist` object, dissimilarity matrix accounting for beta diversity, measured as Jaccard pair-wise dissimilarity (a monotonic transformation of beta diversity)

#Wskaźnik Sorensena

beta0 jest listą zawierającą trzy poziomy
str(beta0)

```
> str(beta0)
List of 3
 $ beta.sim: 'dist' num [1:435] 0.5 0.278 0.412 0.333 0.5 ...
  ..- attr(*, "Labels")= chr [1:30] "m58k" "m58n" "m67k" "m67n" ...
  ..- attr(*, "Size")= int 30
  ..- attr(*, "call")= language as.dist.default(m = beta.sim)
  ..- attr(*, "Diag")= logi FALSE
  ..- attr(*, "Upper")= logi FALSE
 $ beta.sne: 'dist' num [1:435] 0.1087 0.038 0.0168 0.0813 0.0814 ...
  ..- attr(*, "Labels")= chr [1:30] "m58k" "m58n" "m67k" "m67n" ...
  ..- attr(*, "Size")= int 30
  ..- attr(*, "call")= language as.dist.default(m = beta.sne)
  ..- attr(*, "Diag")= logi FALSE
  ..- attr(*, "Upper")= logi FALSE
 $ beta.sor: 'dist' num [1:435] 0.609 0.316 0.429 0.415 0.581 ...
  ..- attr(*, "Labels")= chr [1:30] "m58k" "m58n" "m67k" "m67n" ...
  ..- attr(*, "Size")= int 30
  ..- attr(*, "call")= language as.dist.default(m = beta.sor)
  ..- attr(*, "Diag")= logi FALSE
  ..- attr(*, "Upper")= logi FALSE
```

beta0\$beta.sor - to nas interesuje

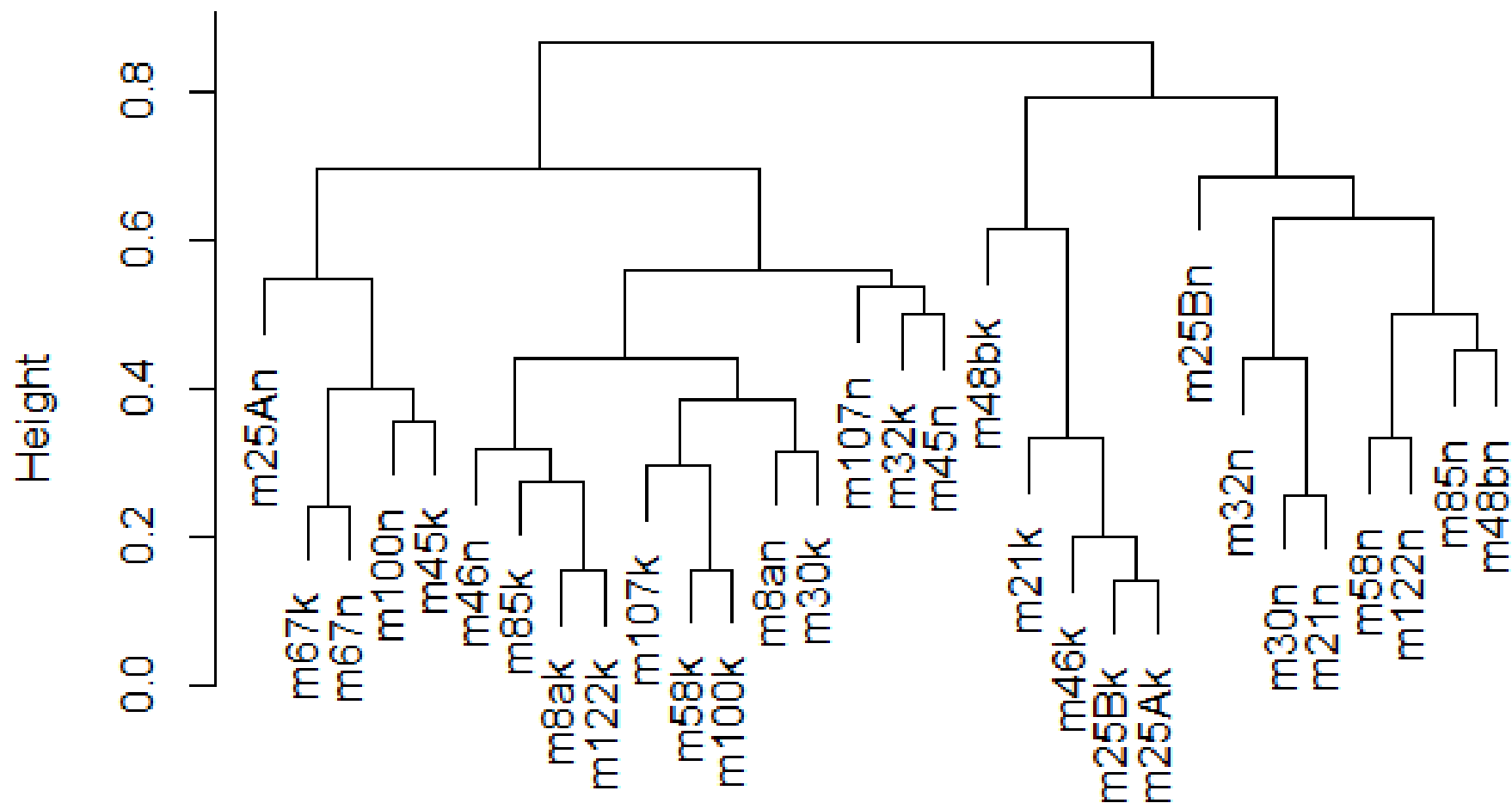
```
> beta0$beta.sor
```

	m58k	m58n	m67k	m67n	m32k	m32n	m85k	m85n	m8ak
m58n	0.6086957								
m67k	0.3157895	0.6666667							
m67n	0.4285714	0.6000000	0.2432432						
m32k	0.4146341	0.6470588	0.4883721	0.6000000					
m32n	0.5813953	0.5094340	0.6444444	0.5714286	0.5833333				
m85k	0.3000000	0.5200000	0.5238095	0.5897436	0.4666667	0.4893617			
m85n	0.4500000	0.4000000	0.6190476	0.6410256	0.5111111	0.4893617	0.3181818		
m8ak	0.2631579	0.5416667	0.4000000	0.4594595	0.3488372	0.5111111	0.2380952	0.4761905	
m8an	0.3846154	0.5510204	0.6097561	0.6315789	0.5000000	0.6521739	0.3488372	0.3953488	0.3170732
m30k	0.2631579	0.5833333	0.4500000	0.5675676	0.4883721	0.6000000	0.3333333	0.3809524	0.3000000
m30n	0.5555556	0.5217391	0.5263158	0.3714286	0.6097561	0.4418605	0.5500000	0.5500000	0.4210526
m122k	0.3953488	0.5849057	0.4666667	0.5238095	0.3750000	0.4800000	0.2765957	0.4468085	0.1555556
m122n	0.5454545	0.3333333	0.6086957	0.5348837	0.5918367	0.4117647	0.5416667	0.5000000	0.4347826
m100k	0.1578947	0.6250000	0.3500000	0.4594595	0.3488372	0.6000000	0.3333333	0.5238095	0.2000000
m100n	0.4509804	0.5737705	0.3962264	0.4000000	0.5714286	0.4827586	0.5636364	0.6000000	0.4716981
m107k	0.2972973	0.6595745	0.5897436	0.6666667	0.4761905	0.6818182	0.3658537	0.5121951	0.2307692
m107n	0.5294118	0.5081967	0.6981132	0.6800000	0.5357143	0.6206897	0.4909091	0.5636364	0.4339623
m45k	0.2682927	0.6470588	0.2558140	0.4000000	0.4347826	0.5416667	0.4666667	0.5555556	0.3953488
m45n	0.4925373	0.5064935	0.4782609	0.5454545	0.5000000	0.5945946	0.4647887	0.4647887	0.4782609
m21k	0.5714286	0.6842105	0.5333333	0.4074074	0.6363636	0.6000000	0.5625000	0.6875000	0.4000000
m21n	0.5428571	0.4666667	0.5675676	0.4117647	0.6000000	0.3809524	0.5897436	0.5897436	0.4594595
m25Bk	0.6000000	0.7714286	0.6296296	0.5000000	0.8000000	0.6875000	0.6551724	0.7241379	0.5555556
m25Bn	0.7600000	0.6000000	0.7037037	0.5833333	0.8666667	0.6875000	0.7241379	0.6551724	0.7037037
m25Ak	0.6000000	0.7714286	0.6296296	0.5000000	0.8000000	0.6875000	0.6551724	0.7241379	0.5555556

#Dendrogram z niepodobieństwem Sorensena

```
clust.soren<-hclust(beta0$beta.sor, "complete")  
plot(clust.soren)
```

Cluster Dendrogram



Metody ordynacji pośredniej

Jak działają?

- #Nie są celem samym w sobie
- #Grupują obiekty w określonym porządku/kolejności
- #Dane wielowymiarowe redukowane do niskowymiarowych
- #Zakładają, że istnieje bazowa struktura danych
- #Różnice w składzie gatunkowym tłumaczone przez zmienne (charakteryzujące strukturę ugrupowań organizmów lub pośrednio opisujące parametry środowiska) według prostego modelu odpowiedzi. Metody ordynacyjne służą identyfikacji tej podstawowej zależności
- #Wiele możliwych zmiennych. Które są ważne?
- #Dają globalny, holistyczny obraz, w przeciwieństwie do regresji, która daje lokalny, indywidualistyczny i redukcjonistyczny pogląd

Zanim zaczniemy...

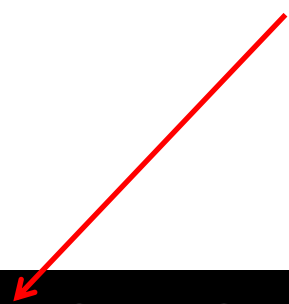
#Pusta komórka w prawym górnym rogu naszej tabelki jest ważna

#Jeśli będzie tam jakaś zawartość, R wywali błąd

#Jak powiedzieć R, że ta komórka jest pusta?

#Jeśli zrobimy csv z naszego pliku exela i wrzucimy w R, będzie on wyglądał tak:

#Puste pole z exela R musi czymś wypełnić (dla R nie jest to wartość NA, bo mu tego nie powiedzieliśmy), więc domyślnie wrzuca X, którego musimy się pozbyć




```
> epi
  X A10h A10 A11h A11 A1h A1 A2h A2 A3h A3 A4h A4 A5h A5 A6h A6 A7h A7 A8h A8 A9h A9 B10h B10 B11h B11 B1h B1
1 Acr.gem 0 0 2 1 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0
2 Aly.var 1 0 0 2 1 1 2 0 2 1 2 2 2 1 1 0 1 1 1 2 2 1 3 1 1 1 1 0
3 Ani.bif 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0
```

Jak usunąć X?

#Klikamy prawym przyciskiem myszy w nasz plik csv i wybieramy opcję „edytuj”

#Plik otworzy się w notatniku



```
;A10h;A10;A11h;A11;A1h;A1;A2h;A2;A3h;A3;A  
h;N8;O1h;O1;O2h;O2;O3h;O3;P1h;P1  
Acr.gem;0;0;2;1;0;1;1;0;0;0;1;0;0;0;0;0;0  
Aly.var;1;0;0;2;1;1;2;0;2;1;2;2;2;1;1;0;1  
Ani.bif;0;0;0;0;0;0;0;1;0;1;0;0;0;0;0;0;0  
Art.art;1;0;1;0;1;1;2;0;0;1;2;1;1;0;1;0;1  
Art.atr;0;0;0;0;0;1;0;1;1;1;0;0;0;0;0;0;0  
Art.did;1;2;0;1;1;2;1;1;0;1;1;1;1;0;1;1;1  
Art.rad;0;0;0;0;1;0;0;0;0;0;0;0;0;0;0;0;0
```

#Potem usuwamy średnik w lewym górnym rogu

Jak powinna wyglądać nasza ramka danych (macierz), zanim każemy R zrobić ordynację?

#Dla nas bardziej intuicyjne jest, przedstawienie tabelki tak:

#W kolumnach poletka

#W wierszach gatunki

> epi																														
	A10h	A11h	A1h	A2h	A3h	A4h	A5h	A6h	A7h	A8h	A9h	B10h	B11h	B1h	B2h	B3h	B4h	B5h	B6h	B7h	B8h	B9h	C10h	C11h	C1h					
Acr.gem	0	2	0	1	0	1	0	0	0	0	1	0	0	1	0	0	0	1	0	0	1	1	2	0	0					
Aly.var	1	0	1	2	2	2	2	1	1	1	2	3	1	1	1	2	2	0	0	1	3	2	2	2	0					
Ani.bif	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0					
	C2h	C3h	C4h	C5h	C6h	C7h	C8h	C9h	D10h	D11h	D1h	D2h	D3h	D4h	D5h	D6h	D7h	D8h	D9h	E10h	E11h	E1h	E2h	E3h	E4h	E5h				
Acr.gem	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0				
Aly.var	0	2	1	0	2	1	1	2	1	1	0	0	2	1	2	1	0	2	2	0	0	0	0	1	1	2				
Ani.bif	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
	E6h	E7h	E8h	E9h	F10h	F11h	F1h	F2h	F3h	F4h	F5h	F6h	F7h	F8h	F9h	G10h	G11h	G1h	G2h	G3h	G4h	G5h	G6h	G7h	G8h	G9h				
Acr.gem	0	0	0	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0				
Aly.var	1	1	2	0	0	2	0	0	0	2	1	0	1	1	1	3	3	0	0	0	0	0	0	1	3	2				
Ani.bif	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
	H10h	H11h	H1h	H2h	H3h	H4h	H5h	H6h	H7h	H8h	H9h	I10h	I11h	I1h	I2h	I3h	I4h	I5h	I6h	I7h	I8h	I9h	K10h	K11h	K1h					
Acr.gem	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0					
Aly.var	0	0	0	0	0	0	0	0	3	3	0	0	0	0	0	0	1	1	3	2	2	1	1	1	0					
Ani.bif	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
	K2h	K3h	K4h	K5h	K6h	K7h	K8h	K9h	L10h	L11h	L1h	L2h	L3h	L4h	L5h	L6h	L7h	L8h	L9h	M10h	M11h	M1h	M2h	M3h	M4h	M5h				
Acr.gem	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0				
Aly.var	0	0	0	1	0	0	2	2	1	2	0	0	0	1	2	3	1	3	3	1	0	0	0	0	3	2				
Ani.bif	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

#Niemniej jednak, jeśli wrzucimy tabelkę w takiej postaci, dla R będzie to nieczytelne

Błąd może i nie wyskoczy, ale cała analiza będzie zrobiona nieprawidłowo i na odwrót

Co trzeba zrobić?

#Zamienić kolumny z wierszami:

```
epi.t<-t(epi)
```

```
> epi.t
  Acr.gem Aly.var Ani.bif Art.art Art.atr Art.did Art.rad Art.spa Art.vin Ath.rua Ath.spe Bac.arc Bac.bec
A10h      0      1      0      1      0      1      0      0      0      0      0      0      0
A11h      2      0      0      1      0      0      0      1      0      0      0      0      1
A1h       0      1      0      1      0      1      1      0      0      0      0      0      0
A2h       1      2      0      2      0      1      0      1      1      0      0      0      0
A3h       0      2      0      0      1      0      0      0      0      0      0      0      1
A4h       1      2      0      2      0      1      0      1      0      0      0      0      0
  Bac.lau Bac.rub Bct.dry Bia.glo Bry.fus Bry.imp Bue.dis Bue.eru Bue.gri Bue.sch Cal.ads Cal.gla Cal.sal
A10h      0      0      0      0      0      0      0      0      0      0      0      0      0
A11h      0      0      0      0      0      1      0      0      0      0      0      0      1
A1h       0      0      0      0      0      0      1      0      0      0      0      0      0
A2h       0      0      0      0      0      0      0      0      0      0      0      0      0
A3h       0      0      0      0      0      0      0      0      0      1      0      0      0
A4h       1      0      0      0      0      1      0      0      1      1      0      0      0
  Cal.vir Clp.pyr Can.xan Crb.ant Cet.cet Cha.bra Cha.bru Cha.chl Cha.chr Cha.fer Cha.fur Cha.gra Cha.pha
A10h      0      0      2      0      1      0      0      0      0      2      0      0      0
A11h      0      0      1      0      0      0      0      0      0      3      0      0      2
A1h       0      0      1      0      1      0      0      0      1      1      0      0      2
A2h       0      0      1      0      1      0      0      0      0      2      0      1      2
A3h       0      0      2      0      1      0      0      0      1      2      0      0      2
A4h       0      0      1      0      1      0      0      0      1      2      0      0      2
```

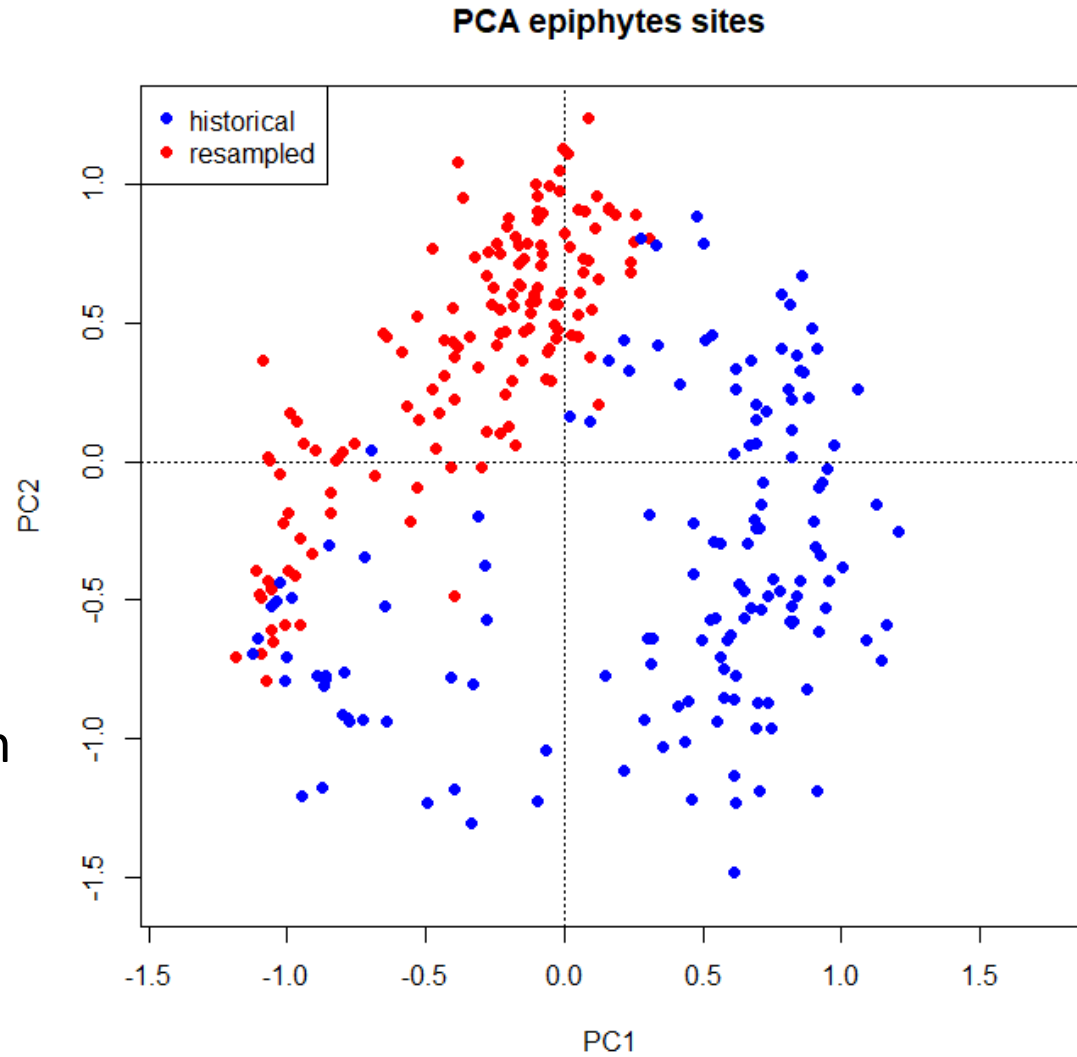
#Teraz jest OK, możemy ruszać

PCA (**P**rincipal **C**omponent **A**nalysis)

#Wyliczenie głównych
składowych – sztucznych
zmiennych; loadings

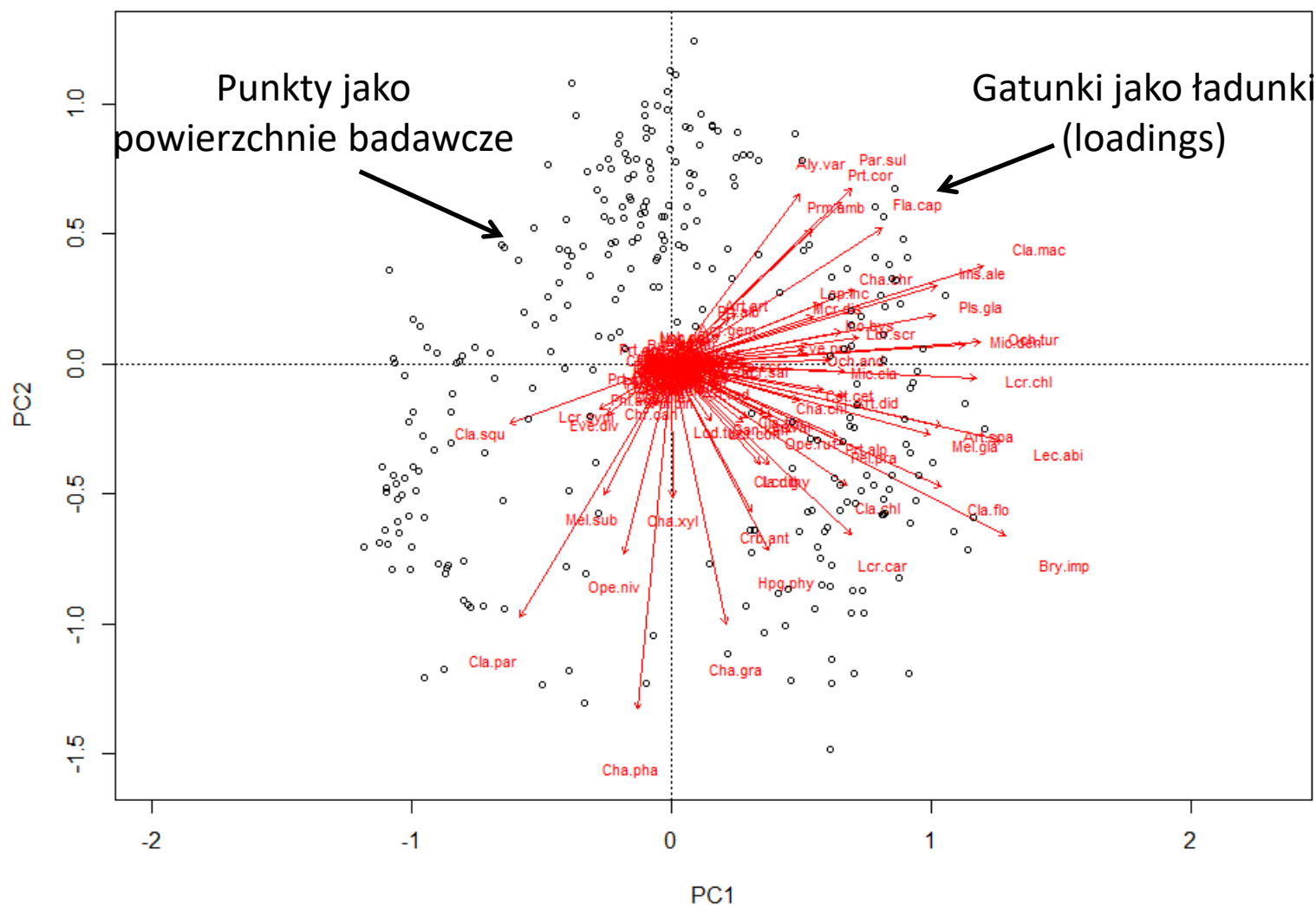
#Zmienne te to liniowe
kombinacje cech,
skorelowane ze sobą

#Kilka głównych składowych
pozwala wyjaśnić większość
zmienności



PCA w R

```
library(vegan)
pca1<-rda(epi.t)
biplot(pca1, choices = c(1, 2), type = c("text", "points"))
```



#Czy w tym przypadku przedstawianie gatunków jako „ładunki” ma sens?

#Oczywiście, że ma, ale wykres nieczytelny ze względu na dużą liczbę gatunków

#Rozwiązanie? – zmniejszenie wpływu gatunków rzadkich (występujących w niskich pokryciach) na wyniki.

#Robi to funkcja `vegan::downweight()`

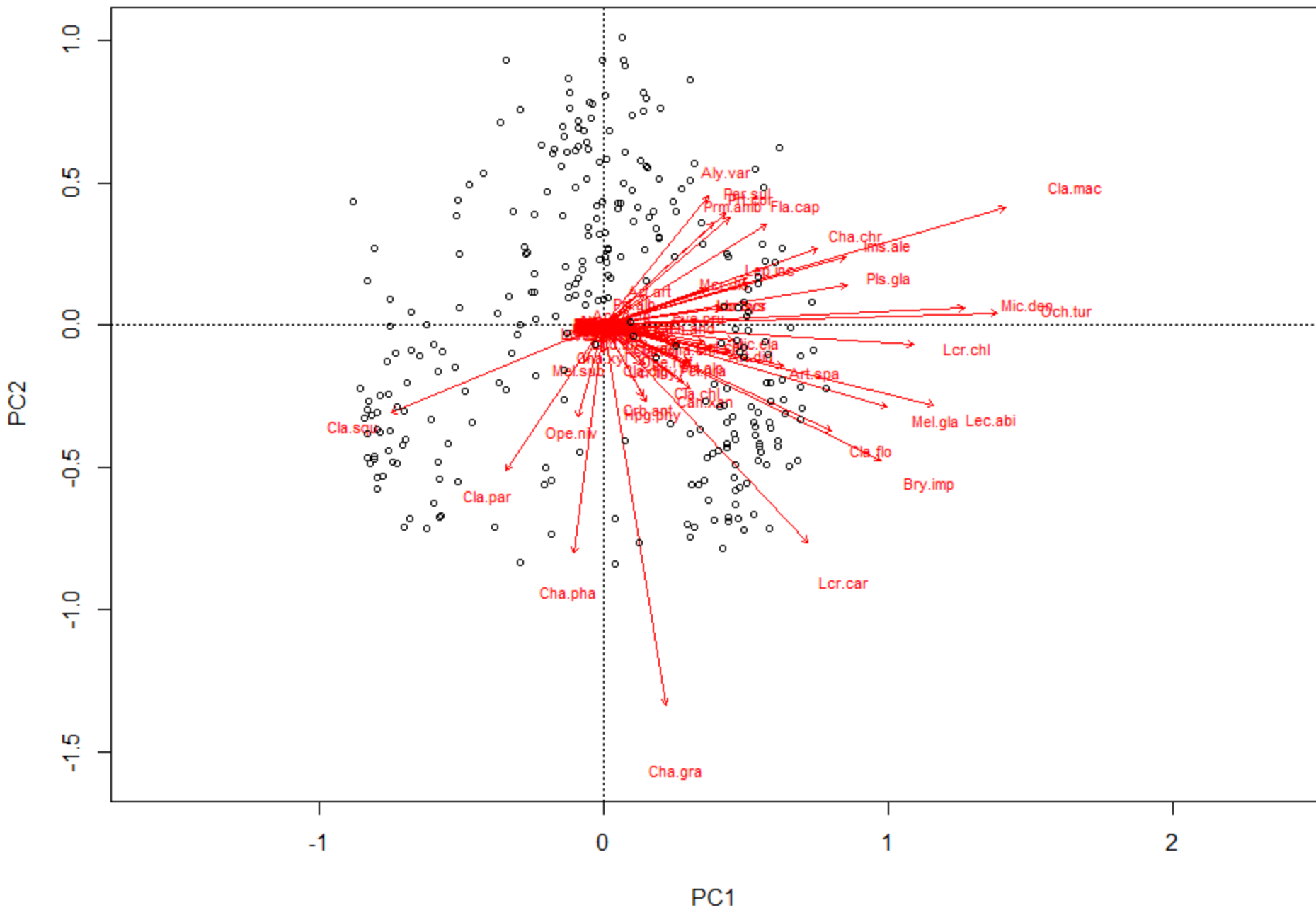
#Downweighting przeprowadzamy na zbiorze wyjściowym:

```
epi.down<-downweight(epi.t, fraction = 1)
```

```
pca.down<-rda(epi.down)
```

```
biplot(pca.down, choices = c(1, 2),  
       type = c("text", "points"))
```

Lepiej?



#Może i lepiej, ale wykres wciąż wydaje się nieczytelny

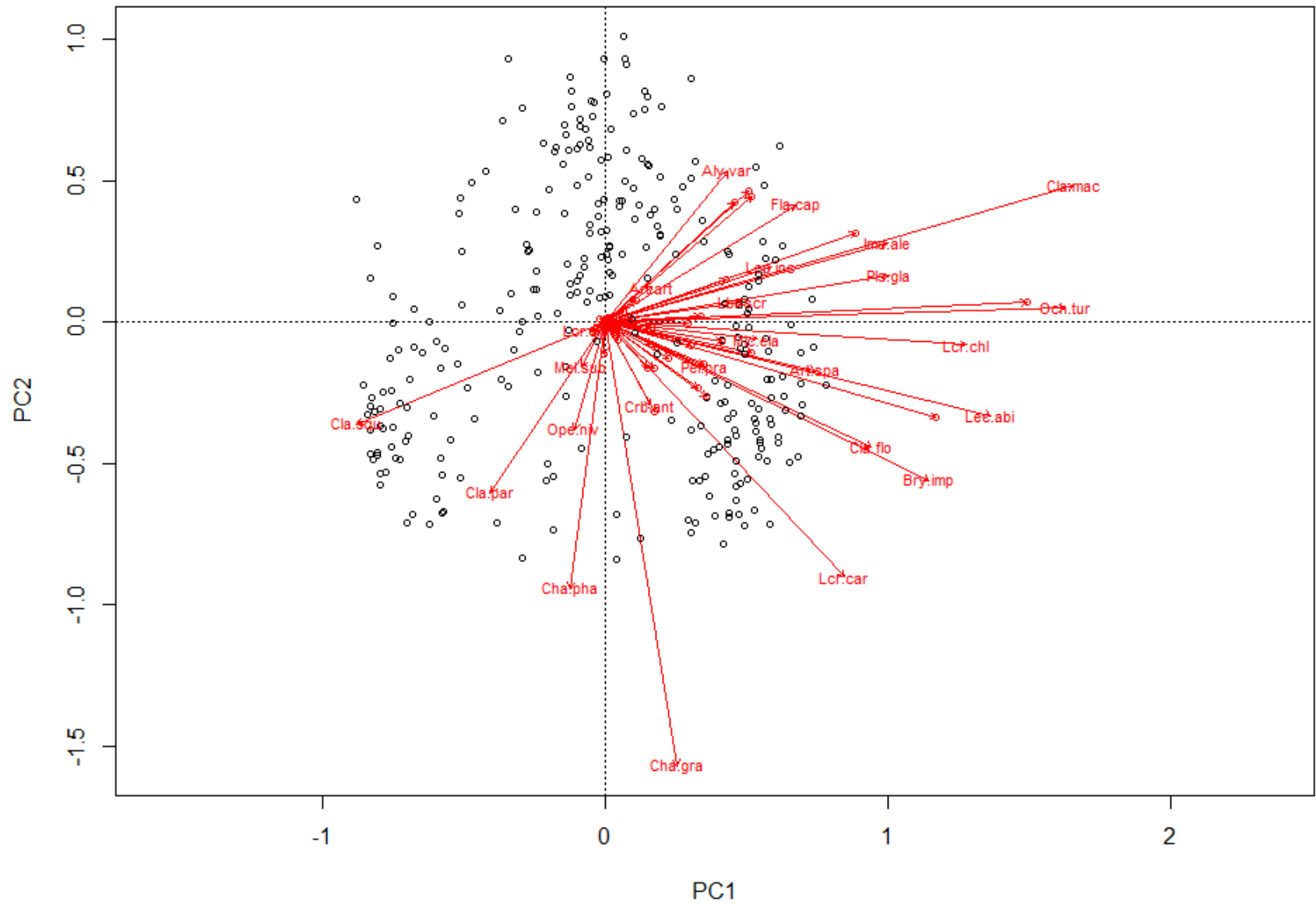
#Inne rozwiązanie – ręczne usunięcie gatunków „zbitych” w chmurę w środku układu współrzędnych, ale trzeba o tym napisać, prezentując rycinę w publikacji

Albo jeszcze inne (dla leniwych) – funkcja `vegan::orditorp()`, która automatycznie wyrzuca te gatunki, które się pokrywają:

```
biplot(pca.down, choices = c(1, 2))  
orditorp (pca.down, display = 'sp', col="red")
```

#Rozwiązanie dobre gdy liczba gatunków jest niewielka

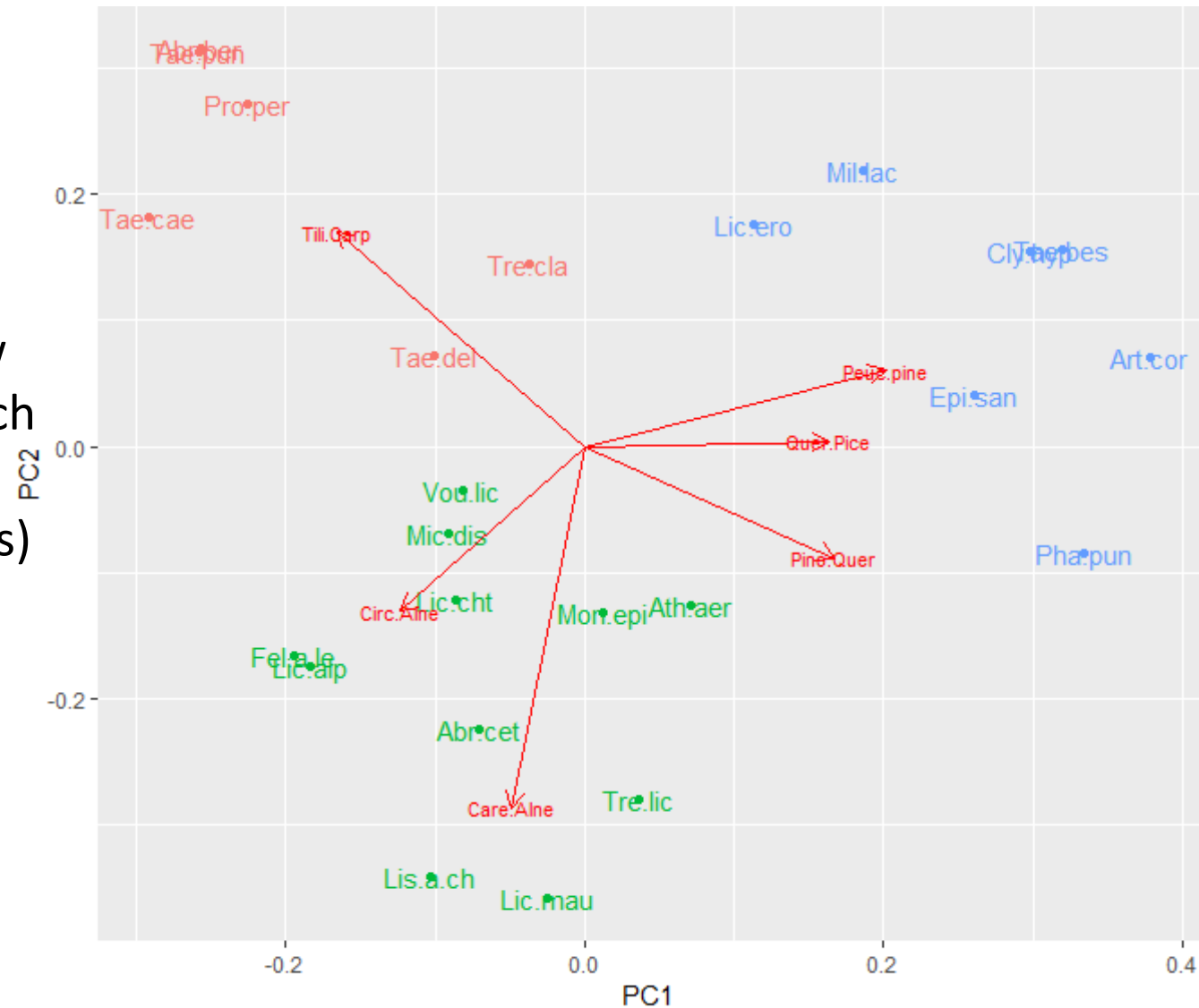
A teraz?



#Duża plastyczność PCA – jako loadings można pokazywać zarówno gatunki (species), jak i powierzchnie (sites)

#Inny zbiór danych:

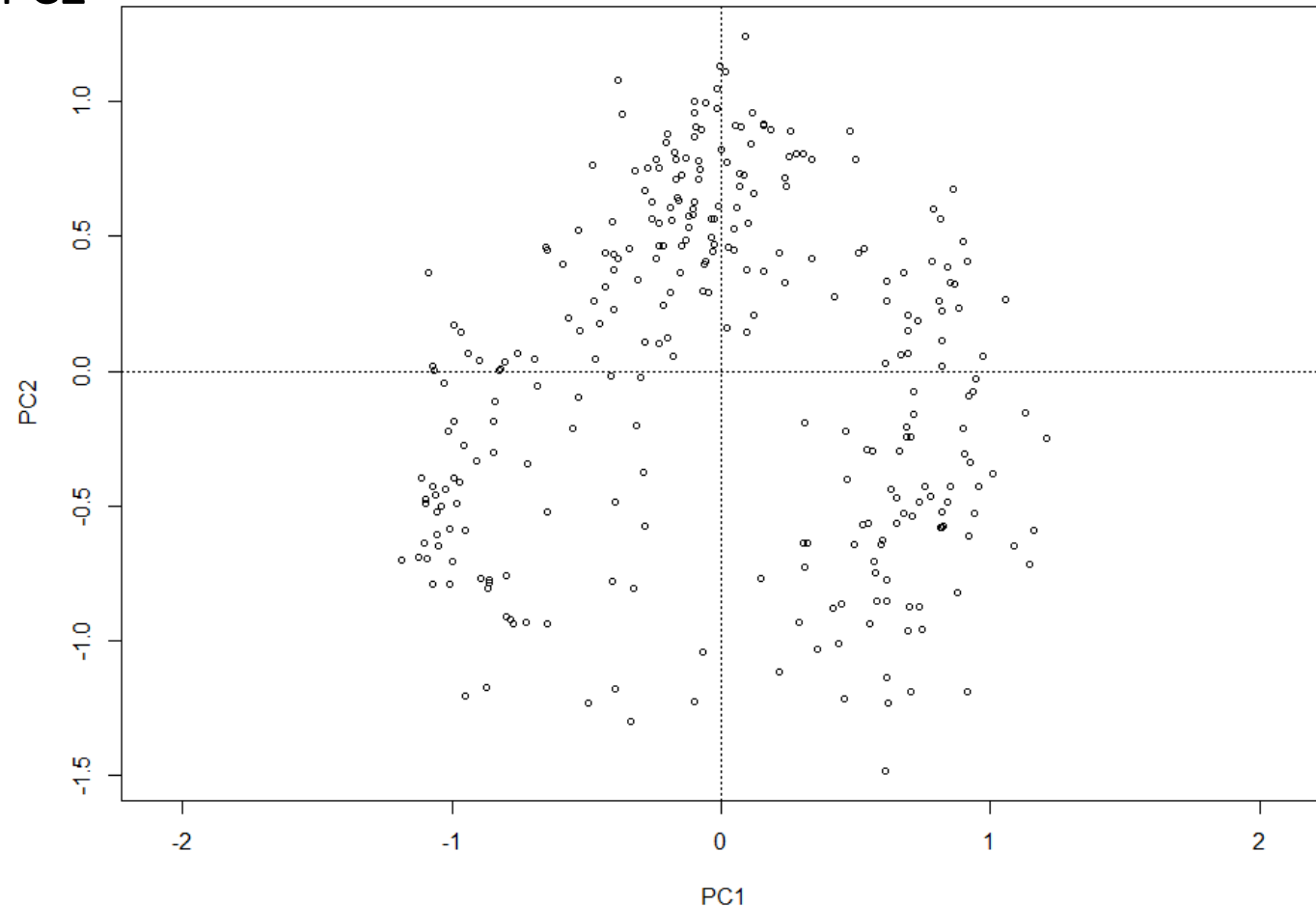
#Punkty z nazwami to współrzędne gatunków grzybów naporostowych (species), a loadings to zbiorowiska leśne (sites)



#Wróćmy do naszych porostów

#Na wykresie widzimy, że wydzielili nam się dwie chmury punktów – jak to interpretować?

#Wiemy, że badaliśmy zmiany składu gatunkowego porostów w czasie (dane historyczne z lat 80 vs. dane aktualne z 2014 r.), można przypuszczać, że to czas był główną zmienną wpływającą na rozrzut punktów, a główny podział nastąpił wzdłuż osi PC1 i (lub) PC2



#No to wrzucimy sobie czas jako faktor na PCA.

#Jak to zrobić – pokolorujemy punkty na czerwono i niebiesko:

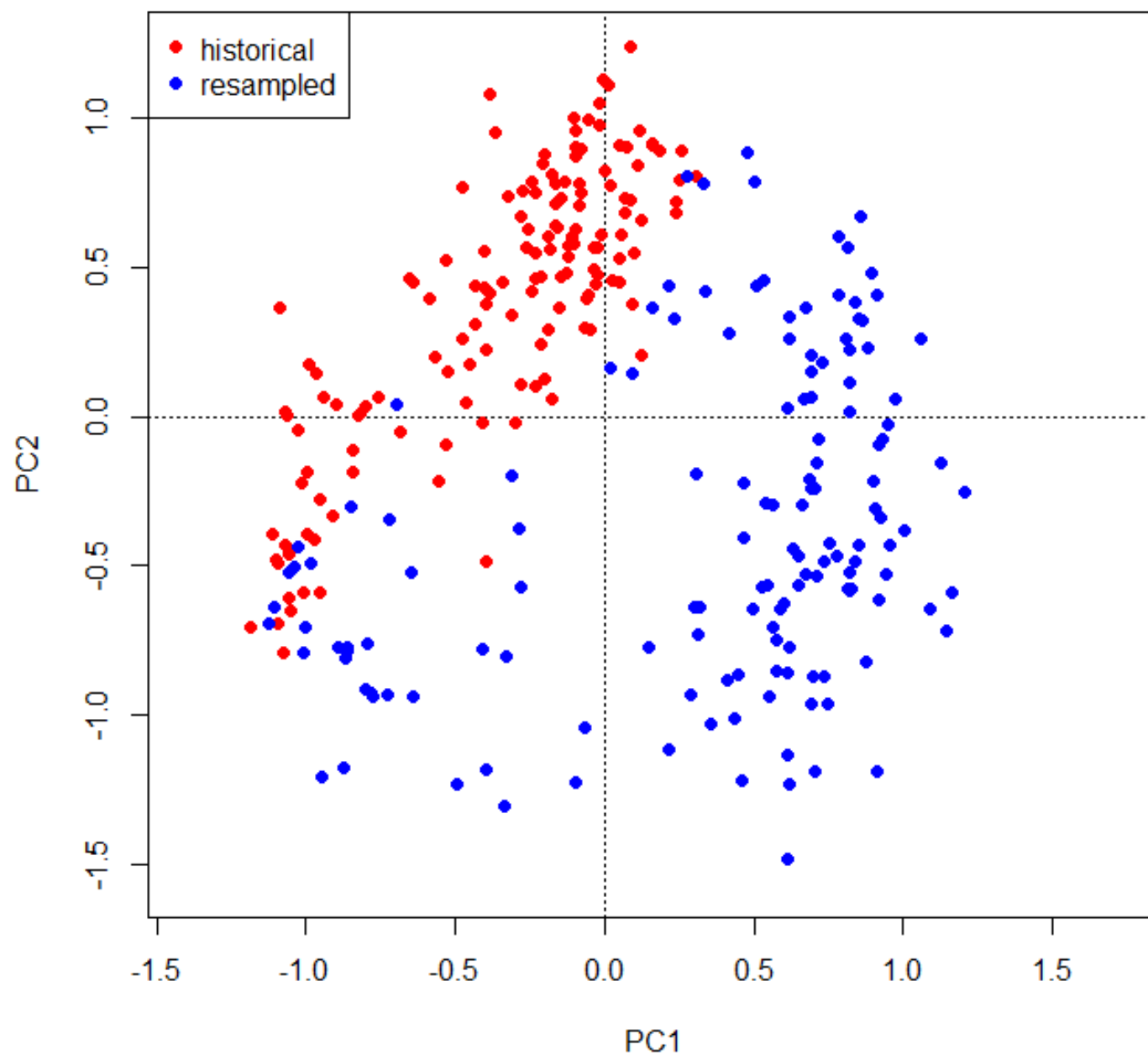
names1<-rep(c("h","n"), 144) #obiekt niosący informację, które poletko jest historyczne (h), a które powtórzone (n)

```
plot(pca1, type='n', xlab="PC1", ylab="PC2", main='PCA epiphytes sites')
points(scores(pca1)$sites[,1][which(names1=="h")],
       scores(pca1)$sites[,2][which(names1=="h")], col="red", pch=19)
points(scores(pca1)$sites[,1][which(names1=="n")],
       scores(pca1)$sites[,2][which(names1=="n")], col="blue", pch=19)
legend('topleft', c('historical', 'resampled'), col=c('red', 'blue'),
      pch=c(19,19), cex=1)
```

scores(pca1)\$sites[,1]#ekstrakcja współrzędnych punktów dla PC1

scores(pca1)\$sites[,2]#ekstrakcja współrzędnych punktów dla PC2

PCA epiphytes sites



PCA w ggplot2

#Bazowe funkcje wizualizujące ordynacje w vegan dają brzydkie obrazki
#Trzeba założyć, że zarówno recenzent, jak i czytelnik będą wzrokowcami –
cóż... chyba wszyscy kochamy obrazki
Ponadto, kod przedstawiony 2 slajdy wcześniej nieintuicyjny oraz przydługi
#Dlatego zrobmy to PCA w ggplot – robi się łatwiej i szybciej
#Plus o wiele więcej możliwości „tunningu” naszego obrazka
#Zaczynamy od ekstrakcji współrzędnych poletek (punktów na wykresie):

```
scores(pca1)$sites[,1]#ekstrakcja współrzędnych punktów dla PC1  
scores(pca1)$sites[,2]#ekstrakcja współrzędnych punktów dla PC2
```

```
scores.pca<-as.data.frame(cbind(scores(pca1)$sites[,1],  
scores(pca1)$sites[,2]))
```

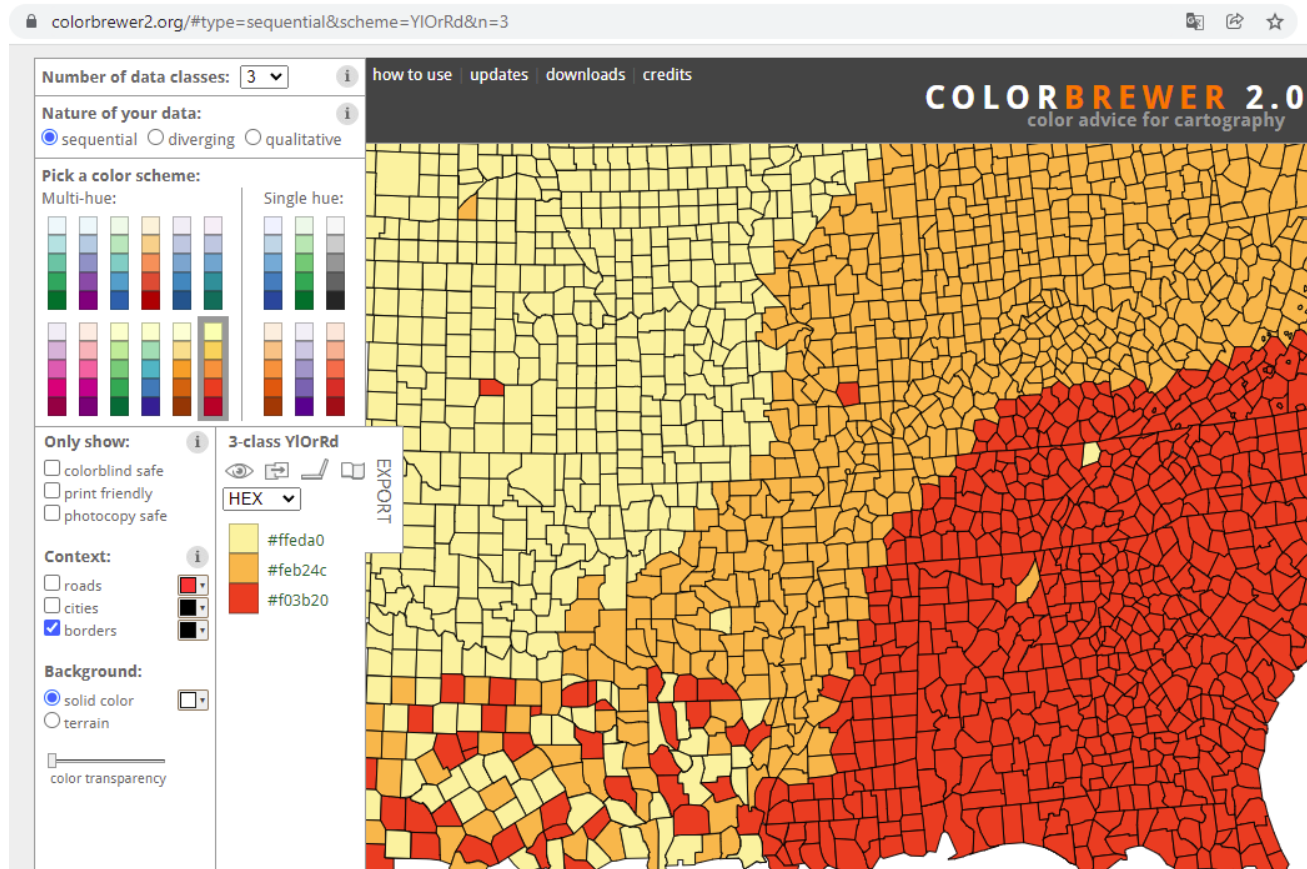
```
names(scores.pca)<-c('PCA1', 'PCA2')
```

#Tak samo współrzędne można ekstrahować dla gatunków – zamiast „sites”
dajemy „species”

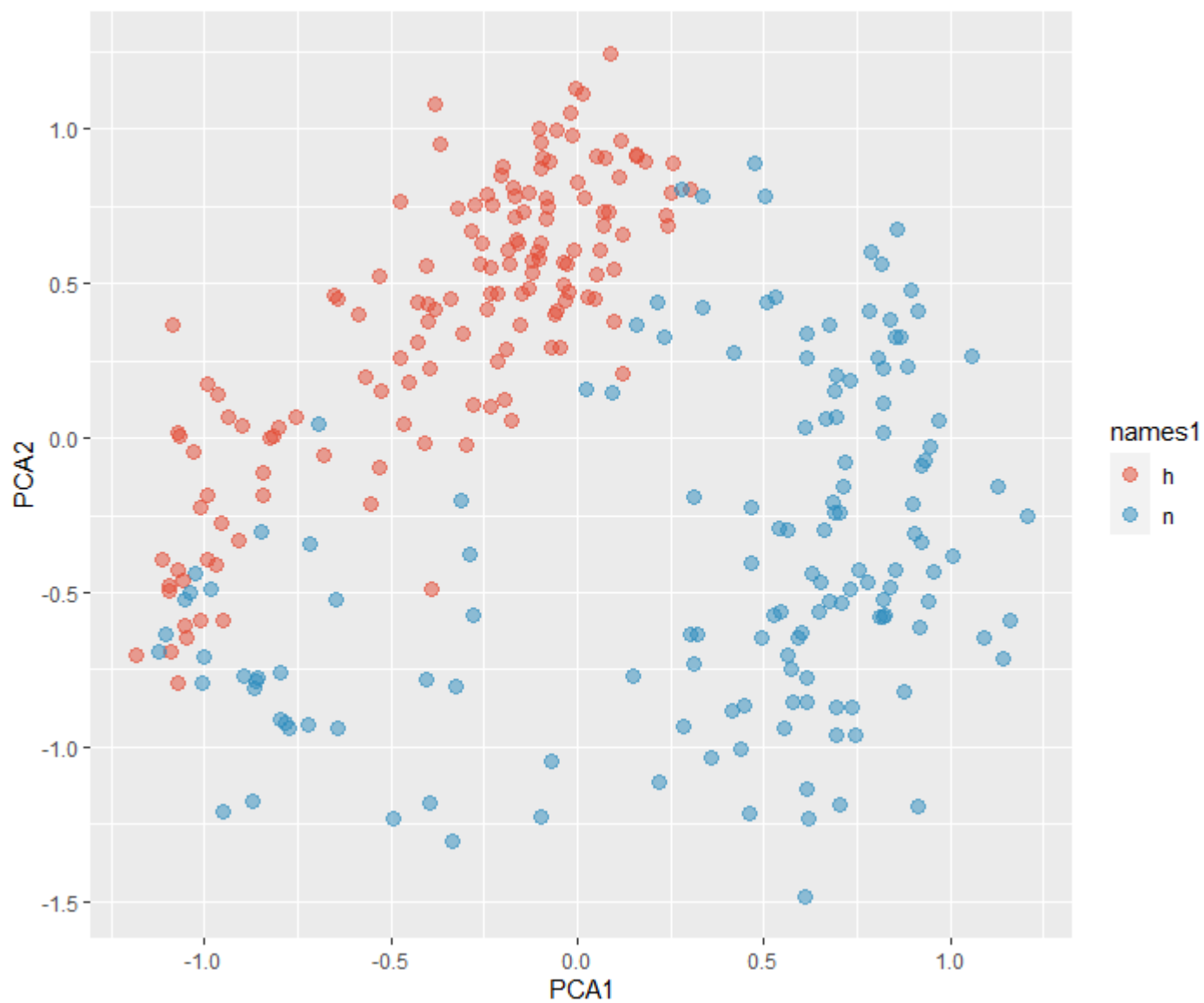
A potem...

`names1<-rep(c("h","n"), 144) #obiekt niosący informację, które
poletko jest historyczne (h), a które powtórzone (n)`

```
ggplot(scores.pca)+geom_point(aes(x=PCA1, y=PCA2,  
col=names1),alpha=0.5,size=3,shape=19)+  
scale_colour_manual(values=c("#e34a33", "#2b8cbe"))
```



Efekt



Co może nam się nie podobać?

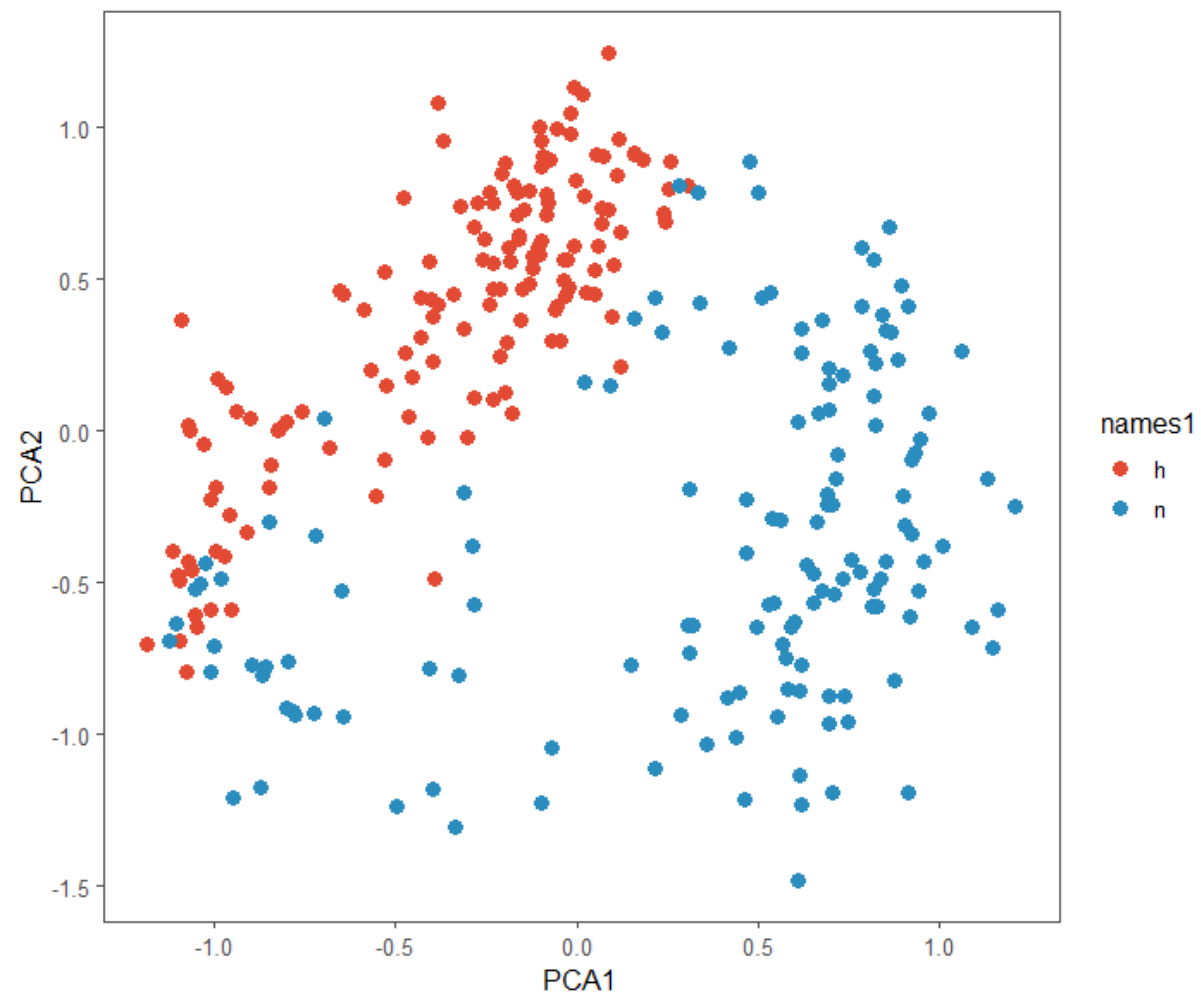
#Półprzezroczyste punkty – usuwamy „alpha=0.5”

```
ggplot(scores.pca)+geom_point(aes(x=PCA1, y=PCA2,  
col=names1),size=3,shape=19)+  
scale_colour_manual(values=c("#e34a33", "#2b8cbe"))
```

#Szare tło, które usuwamy tak:

```
library(ggthemes)
```

```
ggplot(scores.pca)+geom_point(aes(x=PCA1, y=PCA2,  
col=names1),size=3,shape=19)+  
scale_colour_manual(values=c("#e34a33",  
"#2b8cbe"))+theme_few()
```



Co możemy dodać?

#Linie łączące ze sobą poletka historyczne i powtórzone

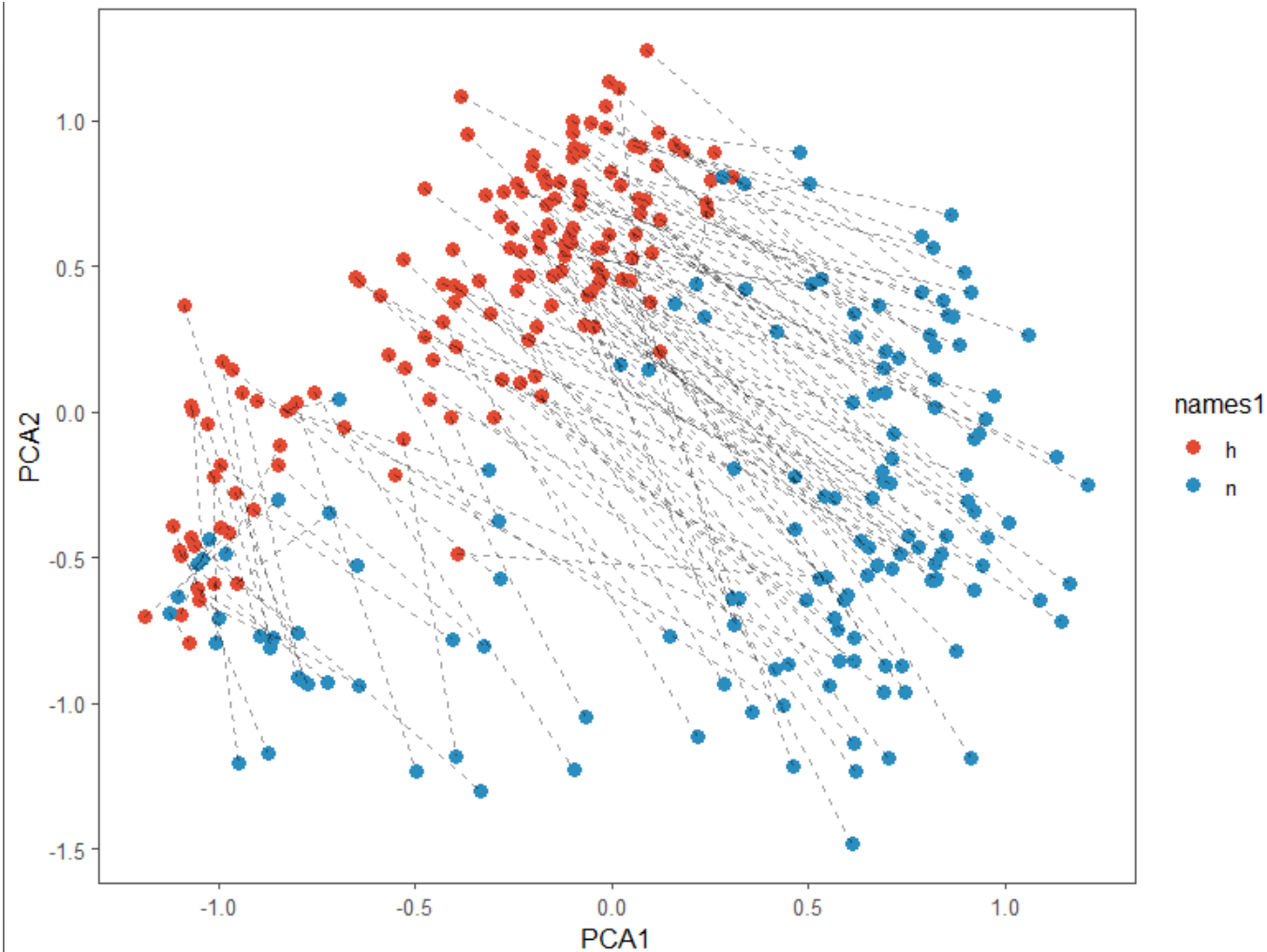
#Czasem przydatne – pokazują, czy zmiany w składzie gatunkowym są kierunkowe, czy też nie

#zaczynamy od stworzenia obiektu z plot.id (ja zrobiłem z ręki w exelu):

```
> poletko.id
[1] 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9 10 10 11 11 12 12
[25] 13 13 14 14 15 15 16 16 17 17 18 18 19 19 20 20 21 21 22 22 23 23 24 24
[49] 25 25 26 26 27 27 28 28 29 29 30 30 31 31 32 32 33 33 34 34 35 35 36 36
[73] 37 37 38 38 39 39 40 40 41 41 42 42 43 43 44 44 45 45 46 46 47 47 48 48
[97] 49 49 50 50 51 51 52 52 53 53 54 54 55 55 56 56 57 57 58 58 59 59 60 60
[121] 61 61 62 62 63 63 64 64 65 65 66 66 67 67 68 68 69 69 70 70 71 71 72 72
[145] 73 73 74 74 75 75 76 76 77 77 78 78 79 79 80 80 81 81 82 82 83 83 84 84
[169] 85 85 86 86 87 87 88 88 89 89 90 90 91 91 92 92 93 93 94 94 95 95 96 96
[193] 97 97 98 98 99 99 100 100 101 101 102 102 103 103 104 104 105 105 106 106 107 107 108 108
[217] 109 109 110 110 111 111 112 112 113 113 114 114 115 115 116 116 117 117 118 118 119 119 120 120
[241] 121 121 122 122 123 123 124 124 125 125 126 126 127 127 128 128 129 129 130 130 131 131 132 132
[265] 133 133 134 134 135 135 136 136 137 137 138 138 139 139 140 140 141 141 142 142 143 143 144 144
>
```

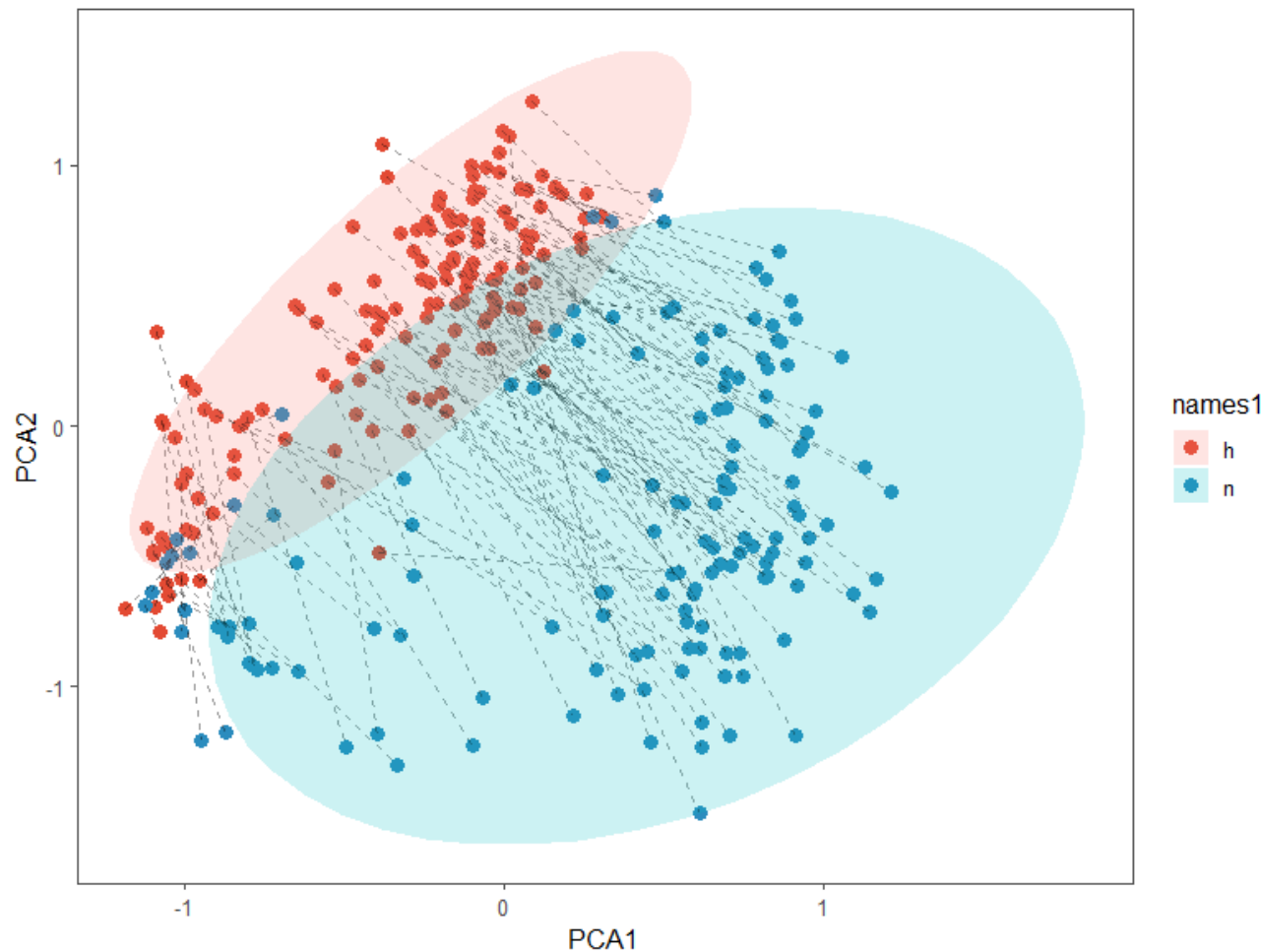
A teraz pora na ggplot

```
ggplot(scores.pca)+geom_point(aes(x=PCA1, y=PCA2,  
col=names1),size=3,shape=19)+  
  scale_colour_manual(values=c("#e34a33", "#2b8cbe"))+  
geom_line(aes(x=PCA1, y=PCA2, group=poletko.id), alpha=0.4,  
linetype='dashed')+theme_few()
```



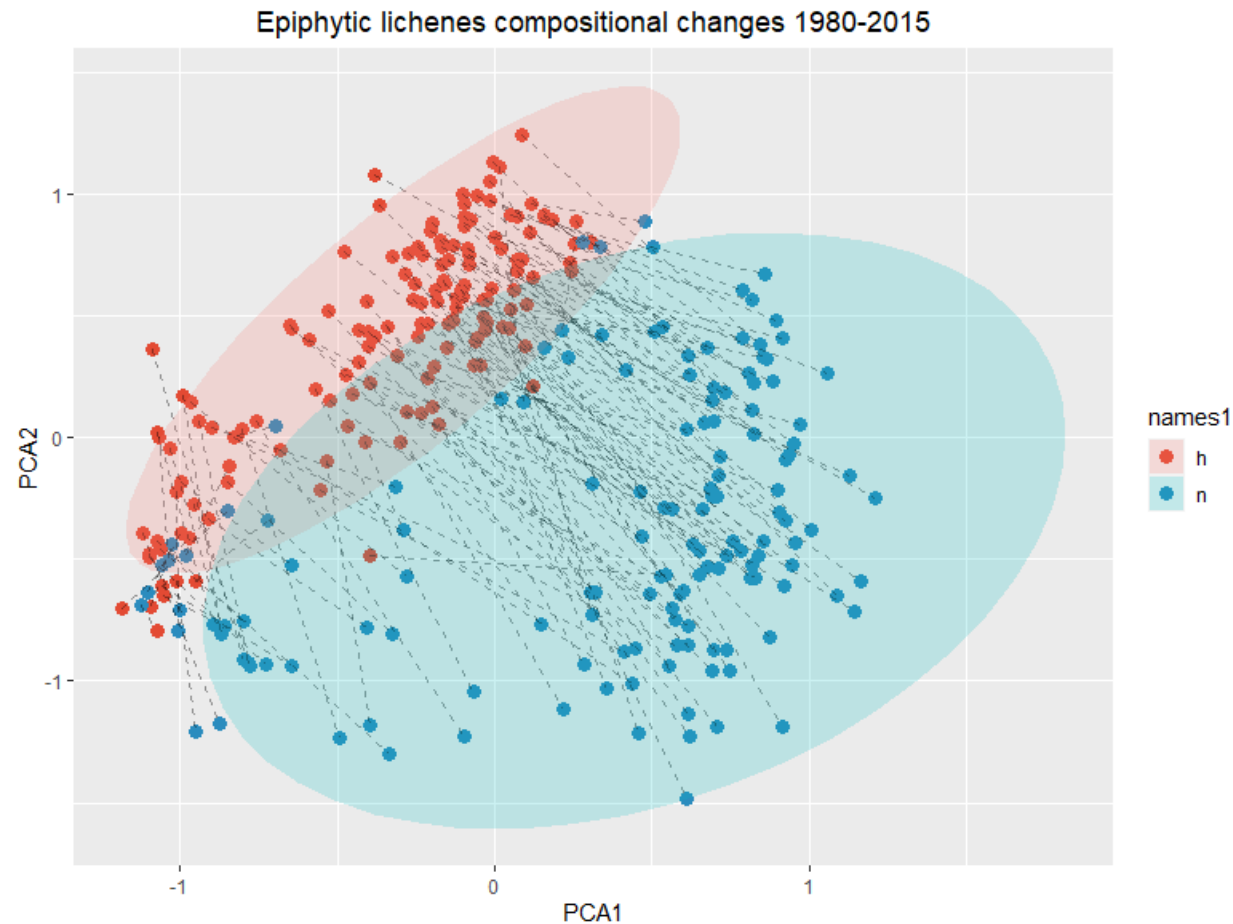
Mało? A może by tak narysować elipsy...

```
ggplot(scores.pca)+geom_point(aes(x=PCA1, y=PCA2,  
col=names1),size=3,shape=19)+scale_colour_manual(values=c("#e34a33",  
"#2b8cbe"))+geom_line(aes(x=PCA1, y=PCA2, group=poletko.id), alpha=0.4,  
linetype='dashed')+stat_ellipse(aes(x=PCA1,y=PCA2,fill=names1),  
geom="polygon", level=0.95, alpha=0.2)+theme_few()
```



Dodajmy jeszcze tytuł obrazkowi

```
ggplot(scores.pca)+geom_point(aes(x=PCA1, y=PCA2,  
col=names1),size=3,shape=19)+scale_colour_manual(values=c("#e34a33",  
"#2b8cbe"))+geom_line(aes(x=PCA1, y=PCA2, group=poletko.id), alpha=0.4,  
linetype='dashed')+stat_ellipse(aes(x=PCA1,y=PCA2,fill=names1),geom="polygon",  
level=0.95, alpha=0.2)+ggtitle("Epiphytic lichenes compositional  
changes 1980-2015")+theme(plot.title = element_text(hjust = 0.5))
```

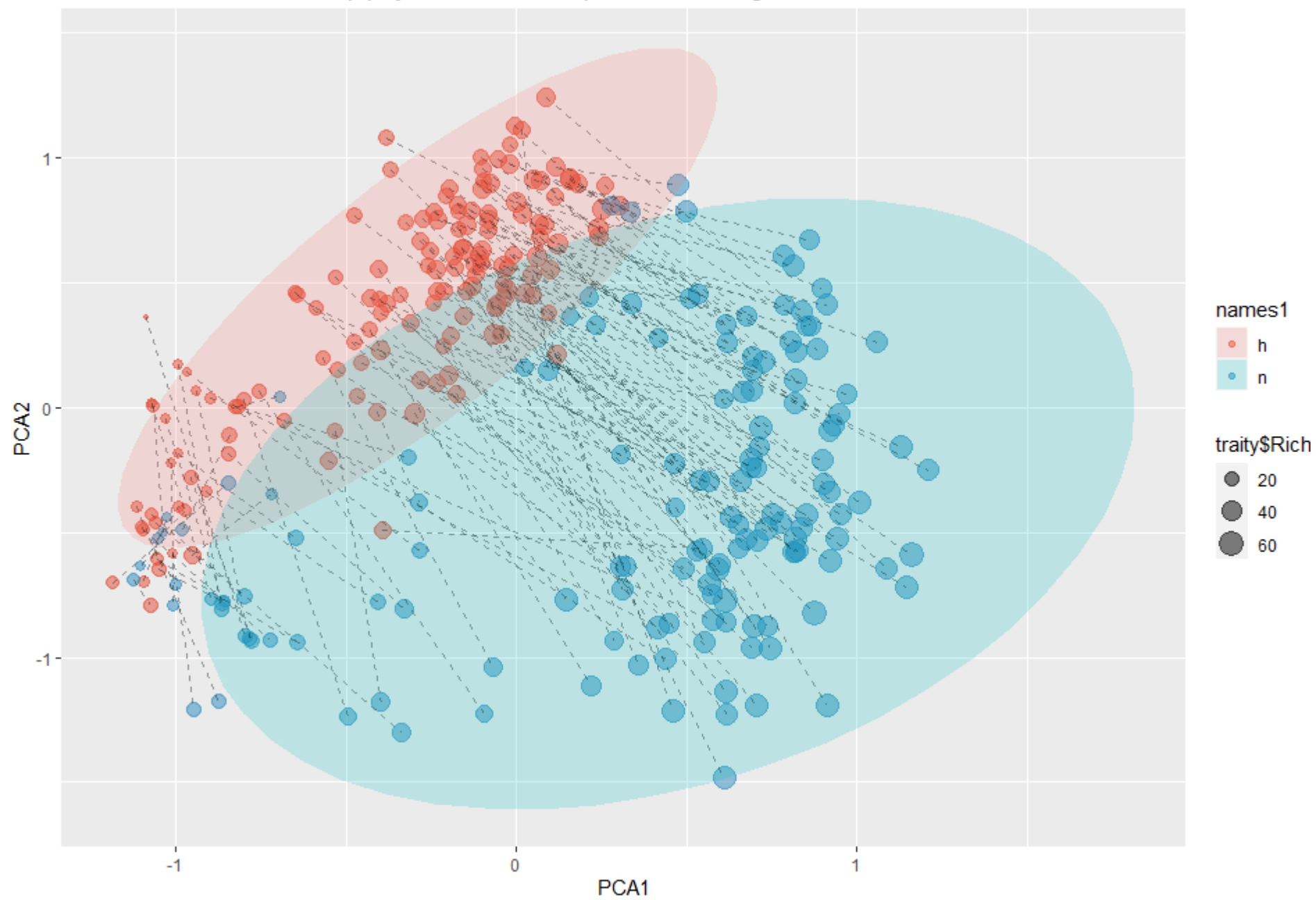


Co jeszcze można zrobić?

#Np. bogactwo gatunkowe poletek (obiekt „sp.rich” zawiera 2 kolumny: plot id oraz liczbę gatunków per plot) wyrazić za pomocą różnych rozmiarów punktów:

```
ggplot(scores.pca)+geom_point(aes(x=PCA1, y=PCA2, col=names1,  
size=sp.rich),alpha=0.5,shape=19)+scale_colour_manual(values=c("#e34a33",  
"#2b8cbe"))+geom_line(aes(x=PCA1, y=PCA2, group=poletko.id), alpha=0.4,  
linetype='dashed')+stat_ellipse(aes(x=PC1,y=PC2,fill=names1),geom="polygo  
n", level=0.95, alpha=0.2)+ggtitle("Epiphytic lichenes compositional  
changes 1980-2015")+theme(plot.title = element_text(hjust = 0.5))
```

Epiphytic lichenes compositional changes 1980-2015



Czy zmiany w czasie rzeczywiście ważne?

#Wobec tego, jak obliczyć procent zmienności wyjaśnionej przez dwie pierwsze osie ordynacyjne?

#Wejdźmy sobie w obiekt, w którym siedzi nasze PCA

```
> pcal
Call: rda(X = epi.t)

              Inertia Rank
Total              46.06
Unconstrained    46.06  145
Inertia is variance

Eigenvalues for unconstrained axes:
   PC1    PC2    PC3    PC4    PC5    PC6    PC7    PC8
13.822  5.888  1.807  1.467  1.264  1.116  0.949  0.895
(Showing 8 of 145 unconstrained eigenvalues)
```

Total inertia = 46.06

Eigenvalue PC1 = 13.822

Eigenvalue PC1/Total inertia
= 0.300 (30%)

Total inertia = 46.06

Eigenvalue PC2 = 5.888

Eigenvalue PC2/Total inertia
= 0.127 (13%)

#Ale...

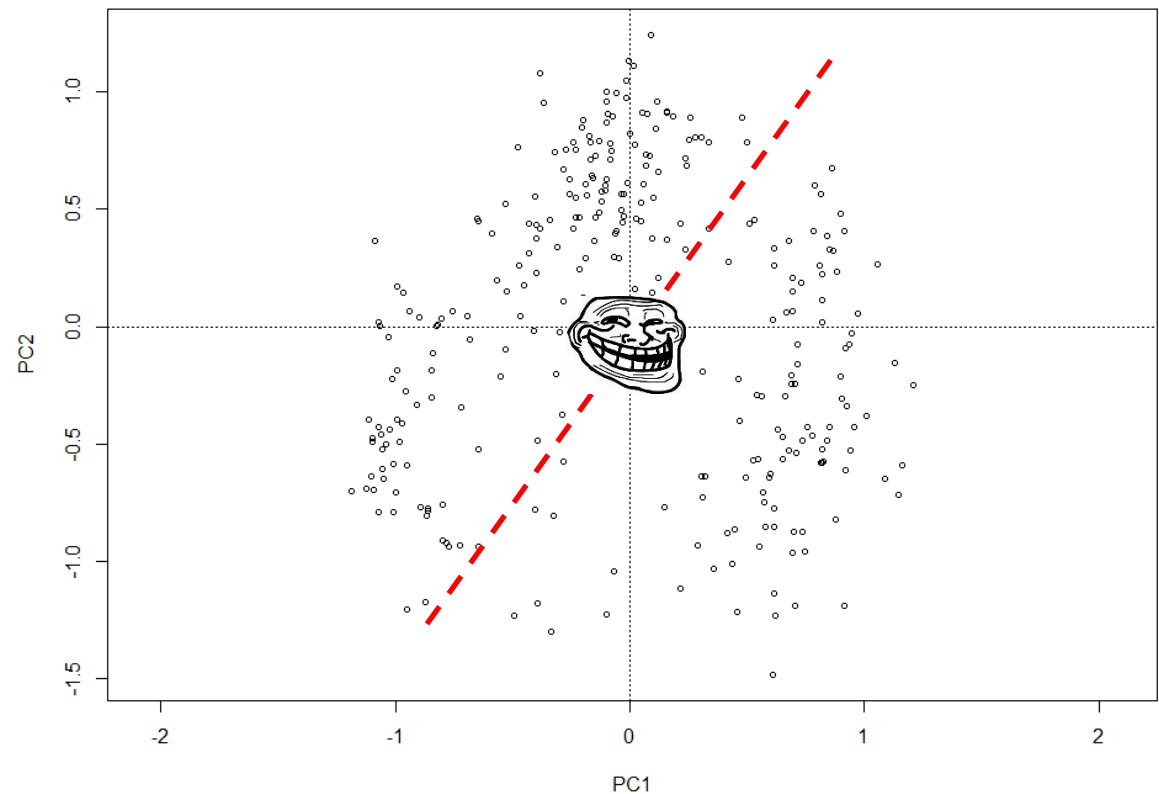
#Na wykresie widać, że ten podział wzdłuż PC1 i PC2 mógł być tylko częściowy

#Co teraz?

#Źle przeprowadzona analiza? Co robić?

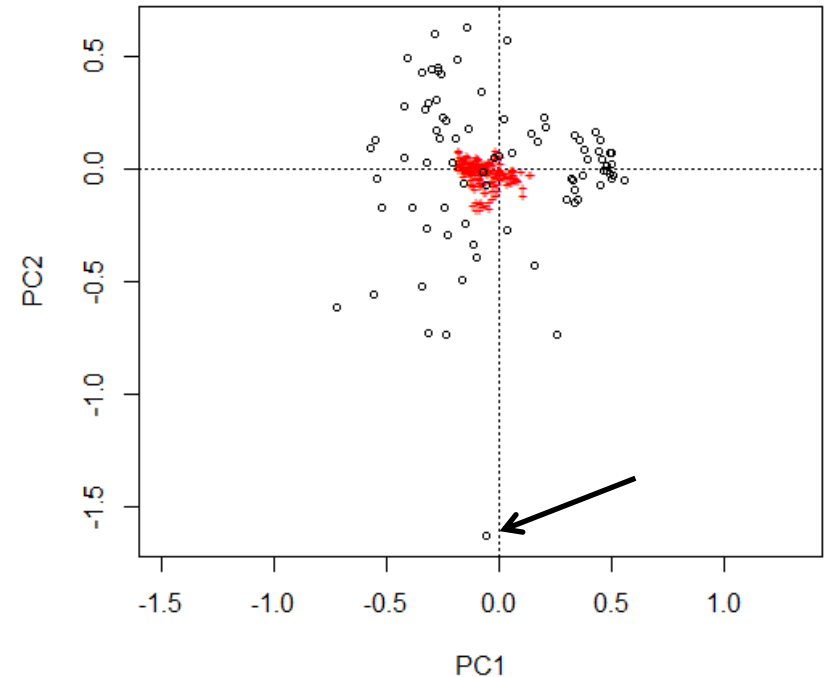
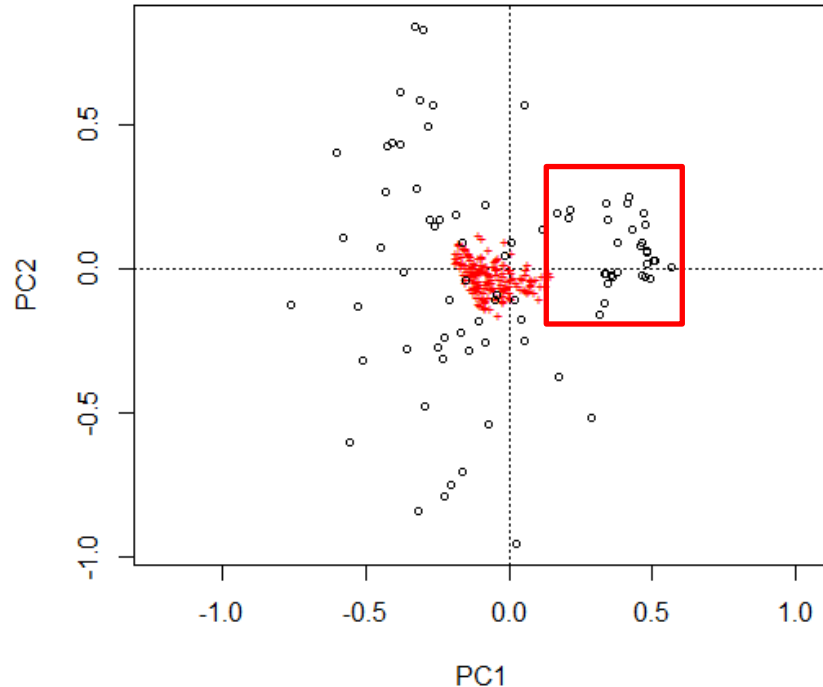
#Można zostawić tak jak jest, albo, jeżeli recenzent to wyłapie, być może kazać będzie zrobić to samo PCA wzdłuż osi pierwszej (PC1) i trzeciej (PC3) lub 2 i 3

#Albo można transformować dane...



Po co transformować dane?

- #Gdy mamy do czynienia z pomiarami w obrębie jednej cechy, pomiędzy którymi są duże różnice
- #Gdy obecne są powierzchnie odstające (outliery)
- #Gdy nasze dane skupiają się w jednym obszarze przestrzeni ordynacyjnej
- #Gdy rozrzut danych surowych nie w pełni satysfakcjonujący sposób oddaje wzorzec lub różnice, które można interpretować ekologicznie



Metody transformacji danych

`decostand(x, method="...")`

The function offers following standardization methods for community data:

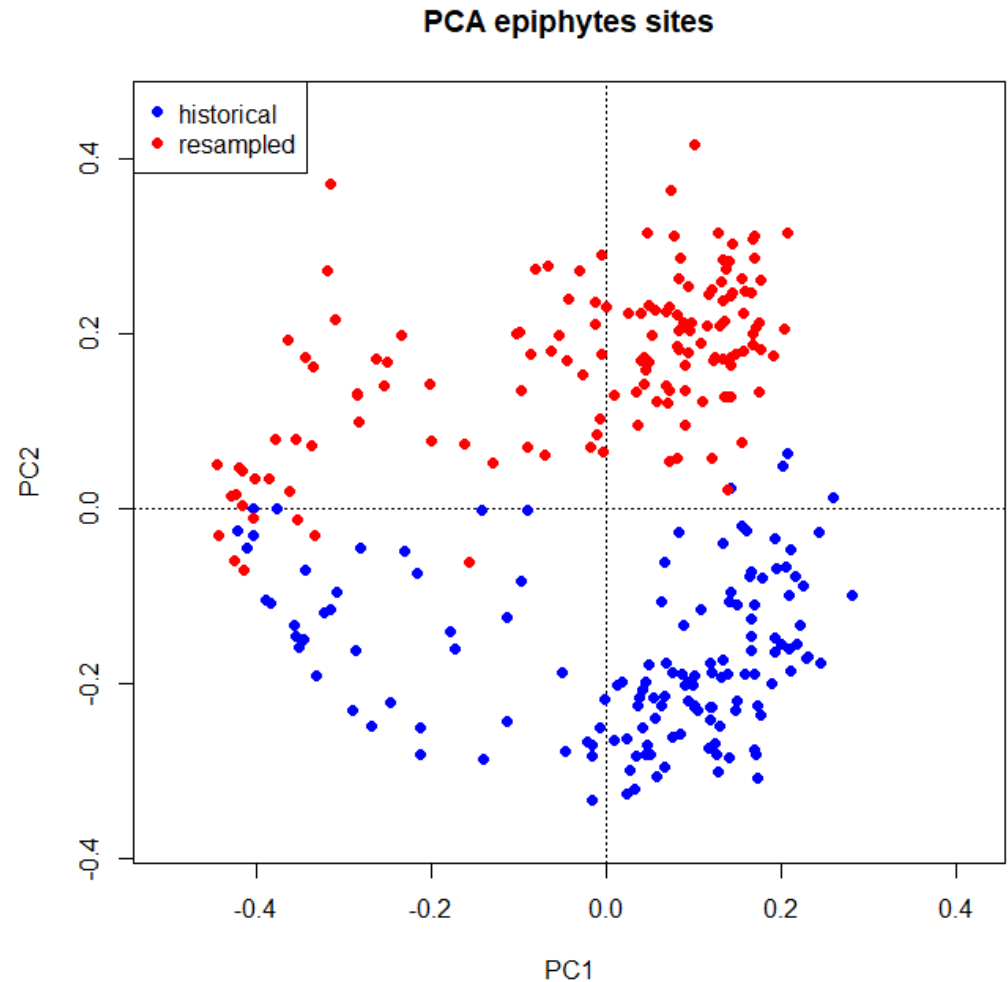
- `total`: divide by margin total (default `MARGIN = 1`).
- `max`: divide by margin maximum (default `MARGIN = 2`).
- `frequency`: divide by margin total and multiply by the number of non-zero items, so that the average of non-zero entries is one (Oksanen 1983; default `MARGIN = 2`).
- `normalize`: make margin sum of squares equal to one (default `MARGIN = 1`).
- `range`: standardize values into range 0 ... 1 (default `MARGIN = 2`). If all values are constant, they will be transformed to 0.
- `rank`, `rrank`: `rank` replaces abundance values by their increasing ranks leaving zeros unchanged, and `rrank` is similar but uses relative ranks with maximum 1 (default `MARGIN = 1`). Average ranks are used for tied values.
- `standardize`: scale `x` to zero mean and unit variance (default `MARGIN = 2`).
- `pa`: scale `x` to presence/absence scale (0/1).
- `chi.square`: divide by row sums and square root of column sums, and adjust for square root of matrix total (Legendre & Gallagher 2001). When used with the Euclidean distance, the distances should be similar to the Chi-square distance used in correspondence analysis. However, the results from `cmdscale` would still differ, since CA is a weighted ordination method (default `MARGIN = 1`).
- `hellinger`: square root of `method = "total"` (Legendre & Gallagher 2001).
- `log`: logarithmic transformation as suggested by Anderson et al. (2006): $\log_b(x) + 1$ for $x > 0$, where b is the base of the logarithm; zeros are left as zeros. Higher bases give less weight to quantities and more to presences, and `logbase = Inf` gives the presence/absence scaling. Please note this is *not* $\log(x+1)$. Anderson et al. (2006) suggested this for their (strongly) modified Gower distance (implemented as `method = "altGower"` in `vegdist`), but the standardization can be used independently of distance indices.

Normalizacja

```
porosten.norm<-decostand(epi.t, method="normalize")  
pca.porosten.norm <-rda(porosten.norm)
```

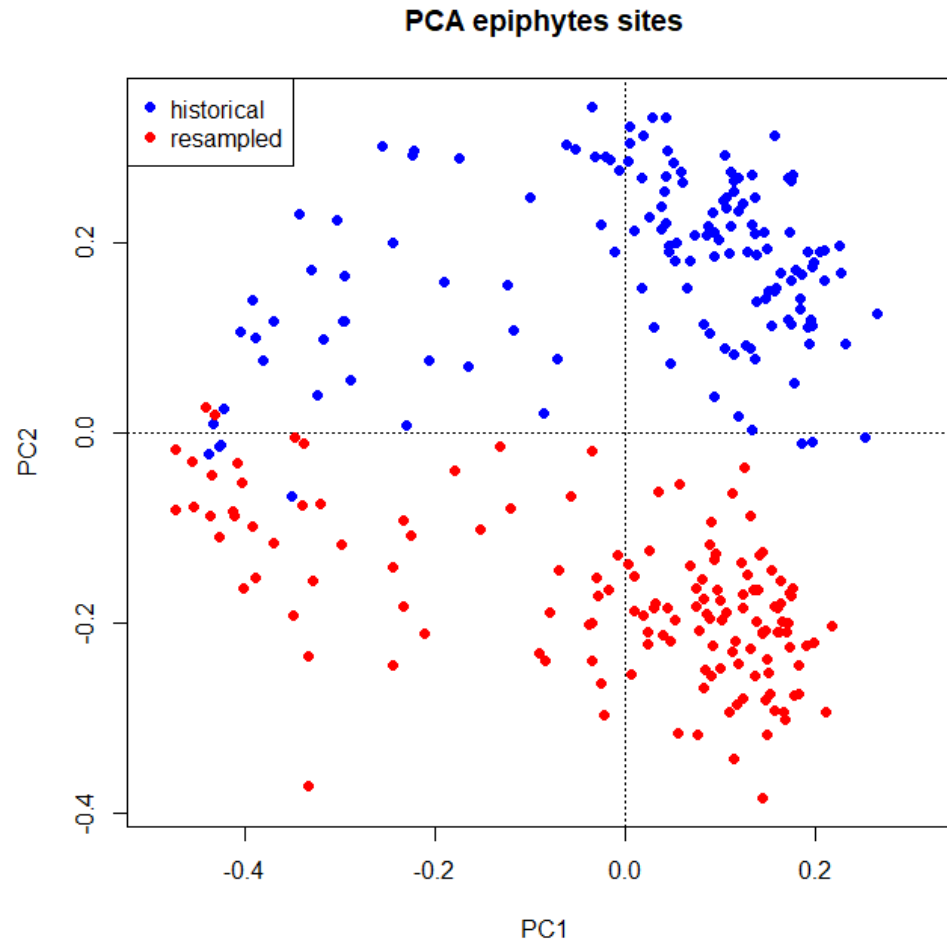
#Kod do stworzenia obrazka w
ggplot2 taki sam, jak poprzednio

#Pamiętajmy o ponownej
ekstrakcji współrzędnych
i podmianie obiektów



Spierwiastkowanie

```
porosten.hell<-decostand(epi.t, method="hellinger")  
pca.porosten.hell <-rda(porosten.hell)
```



Która metoda transformacji lepsza?

#Nie ma lepszej i gorszej

#Najczęściej używane to: range, normalize, standardize, log i pa

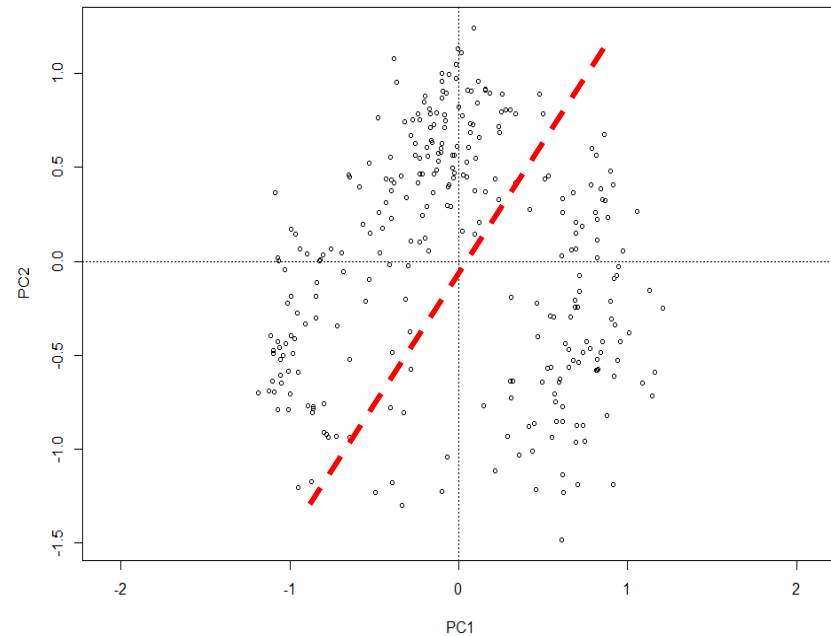
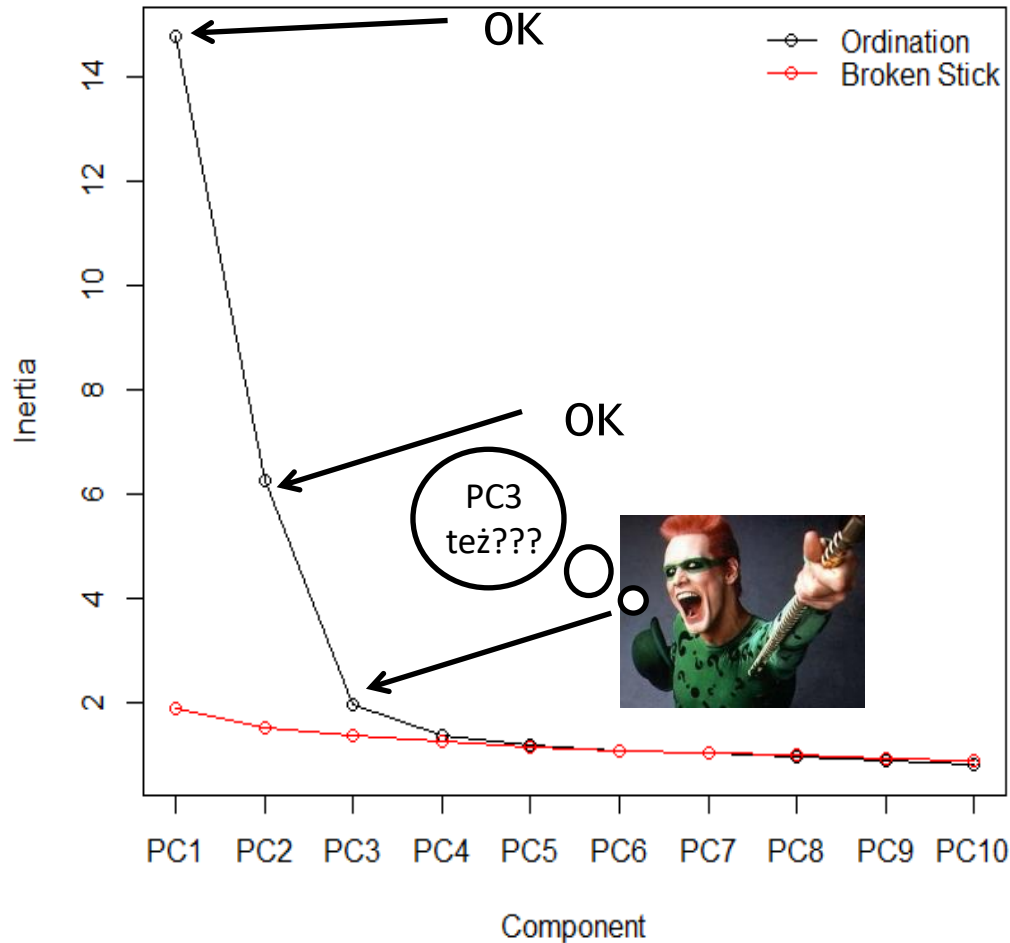
#Metodę transformacji zawsze dobieramy do danych, z jakimi pracujemy

#Najlepiej intuicyjnie przeprowadzić kilka rodzajów transformacji i wybrać taki wykres, który najlepiej tłumaczy różnice czy dany proces w kontekście ekologicznym

#Transformacji danych nie trzeba przeprowadzać, jeżeli rozrzut surowych danych w przestrzeni ordynacyjnej jest satysfakcjonujący dla badacza

Diagnoza poprawności analizy (metoda złamanego kijka)

```
screepplot(pca1, type='lines', bstick=TRUE)
```

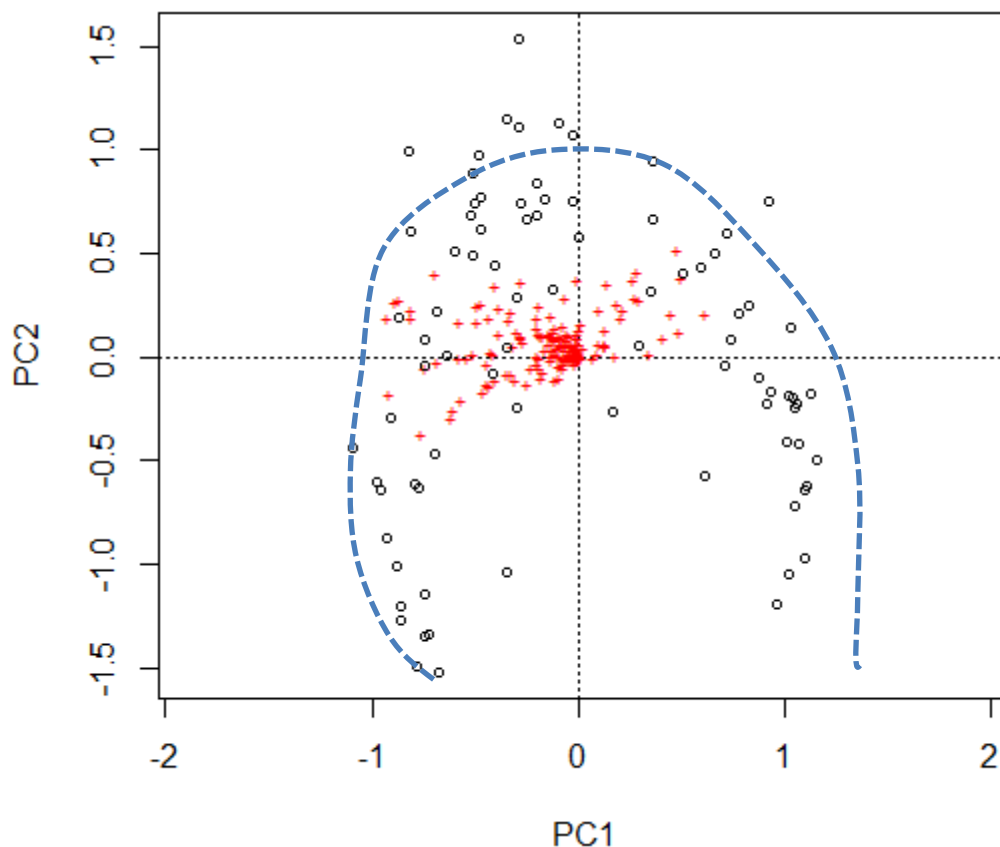
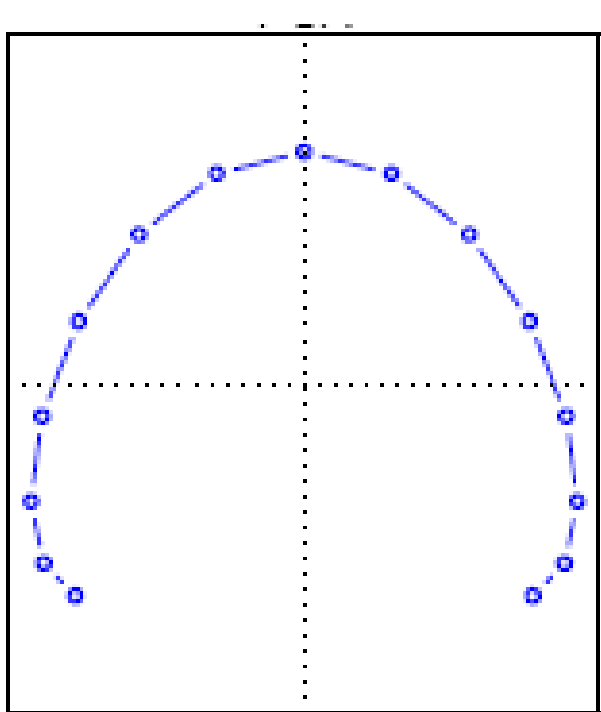


Artefakt PCA: efekt podkowy (horseshoe effect)

#Gdy gradient jest za długi, dane układają się w charakterystyczny sposób, przypominający podkowę.

#Przyczyna: nieprawidłowe uporządkowanie danych wzdłuż pierwszej osi (dane upakowywane „na siłę” na końcach gradientu).

#Wtedy PCA nie jest odpowiednią metodą do analizy zbioru danych

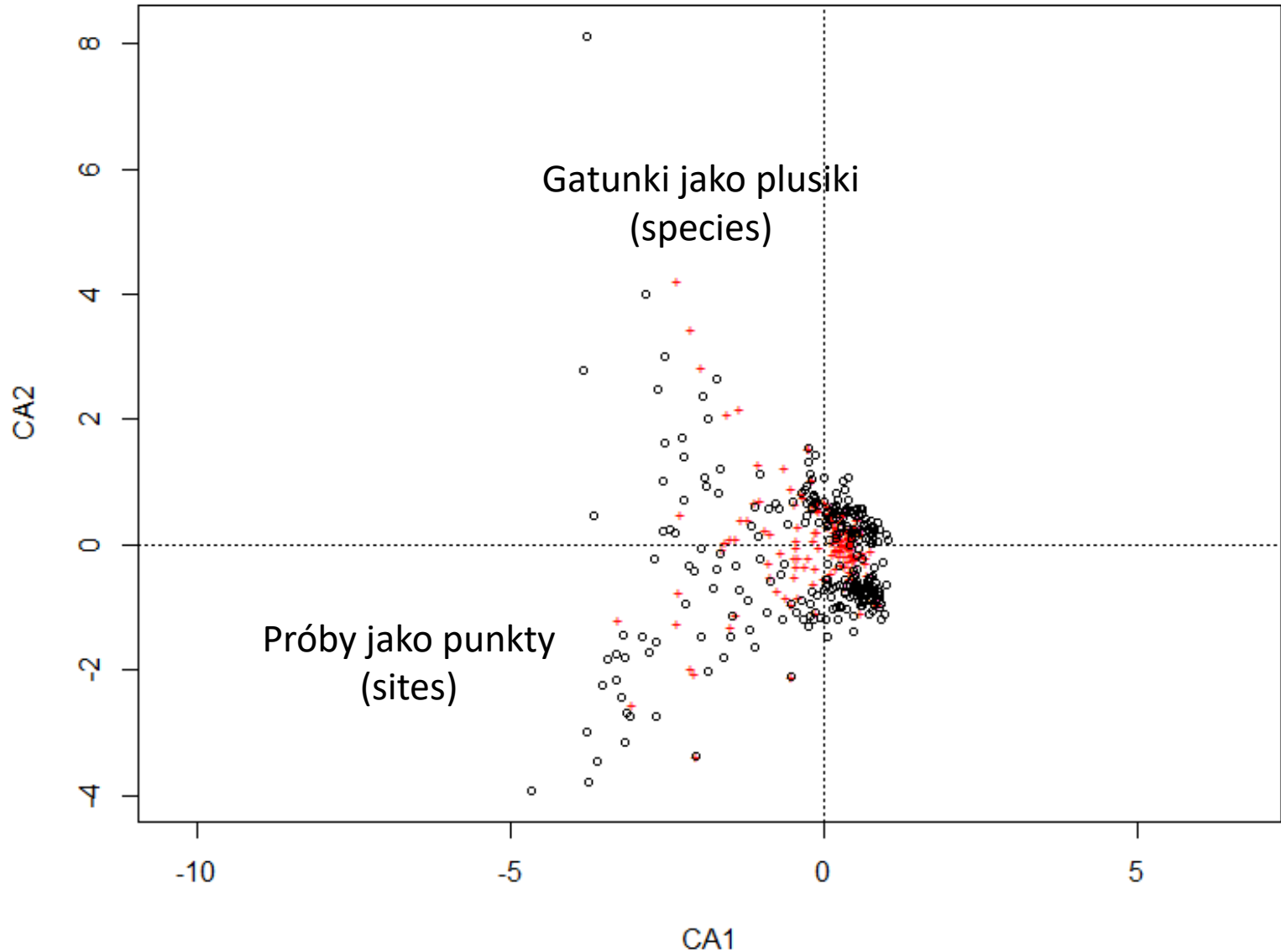


CA (**C**orrespondence **A**nalysis)

- #Do analizy krótkich gradientów – jak obliczyć długość gradientu, będzie potem
- #Nie wylicza głównych składowych, lecz przyjmuje arbitralne wskaźniki
- #Dla gatunków w obrębie poletek wylicza średnie ważone wartości
- #Następnie wylicza nowe wartości uśredniając je, by miały jednakowe wagi (korespondowały ze sobą)

CA w R

```
ca1<-cca(dane)  
plot(ca1, display=c("sites", "species"))
```



CA na danych z porostów w ggplot2

#Ekstrakcja współrzędnych poletek:

```
scores.ca<-as.data.frame(cbind(scores(ca1)$sites[,1],  
scores(ca1)$sites[,2]))
```

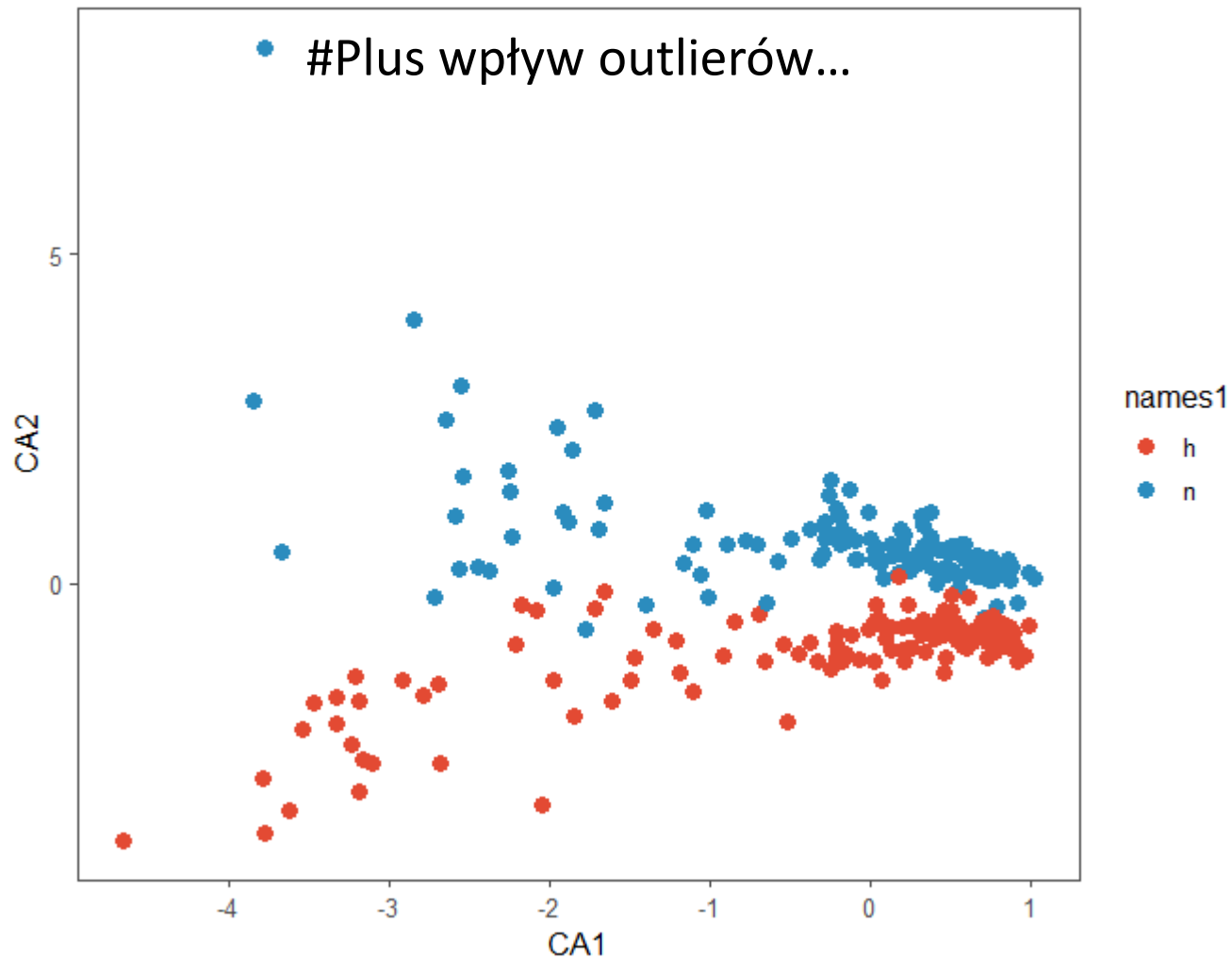
```
names(scores.ca)<-c('CA1', 'CA2')
```

#wykres:

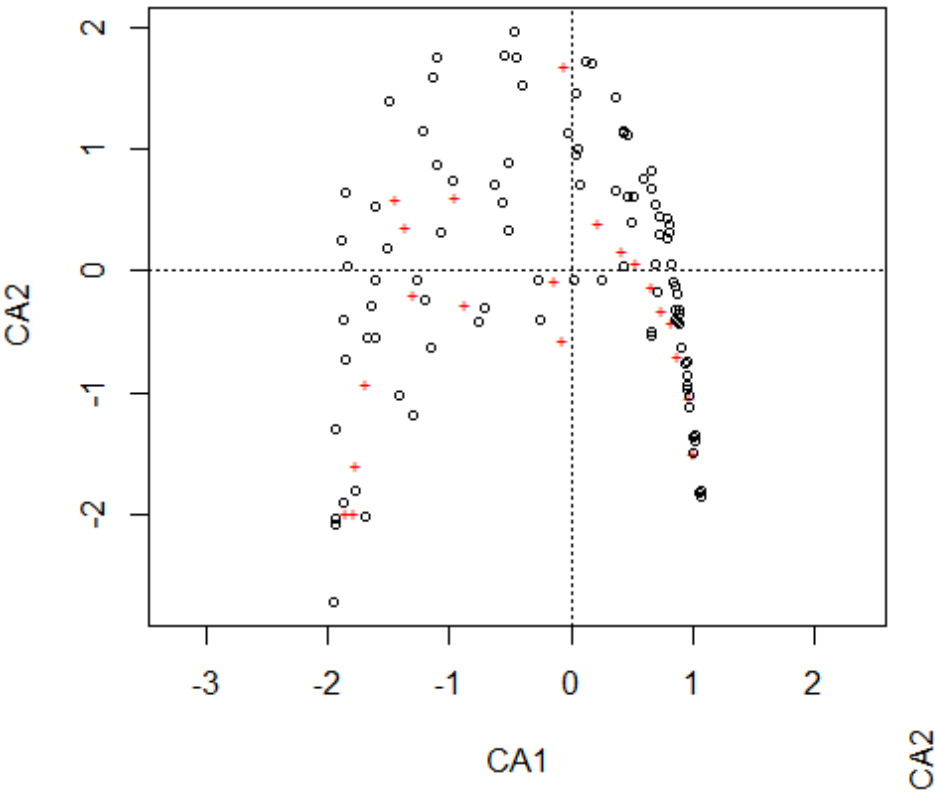
```
ggplot(scores.ca)+geom_point(aes(x=CA1, y=CA2,  
col=names1),size=3,shape=19)+  
scale_colour_manual(values=c("#e34a33",  
"#2b8cbe"))+theme_few()
```

#Dla porostów lepsza wydawała się PCA

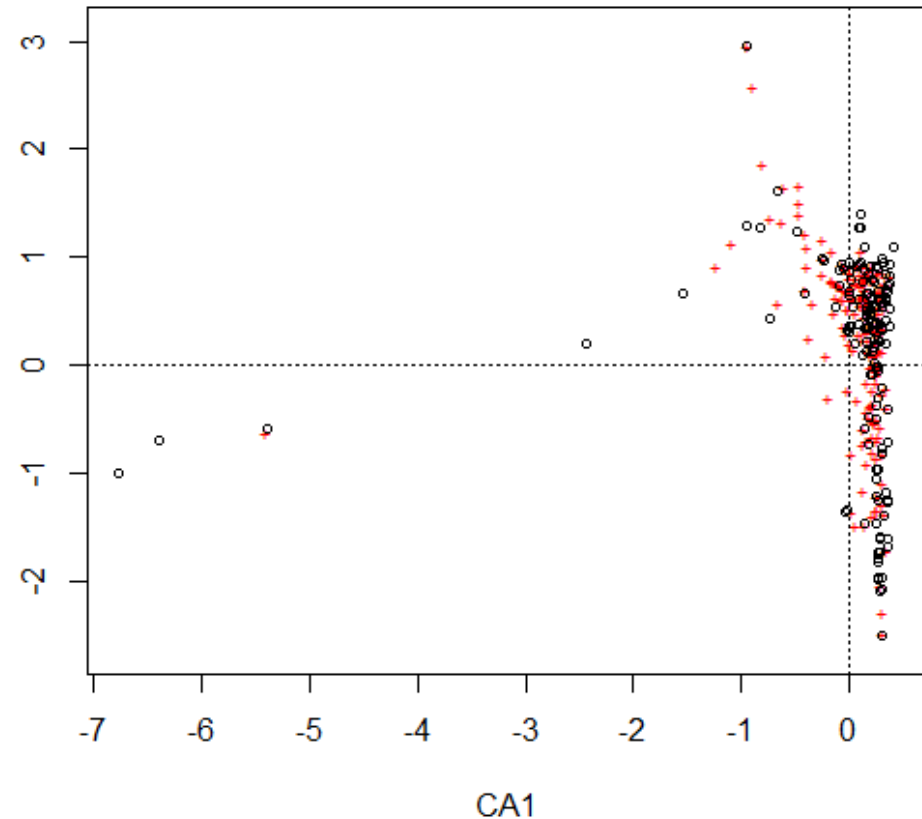
#Przy CA mamy upakowanie poletek na końcu gradientu



Częsty artefakt przy CA: efekt łuku (arch effect)



#Przy porostach nie było, ale punkty upakowane przy końcu gradientu
#Jak uniknąć efekt łuku? – Zrobić DCA

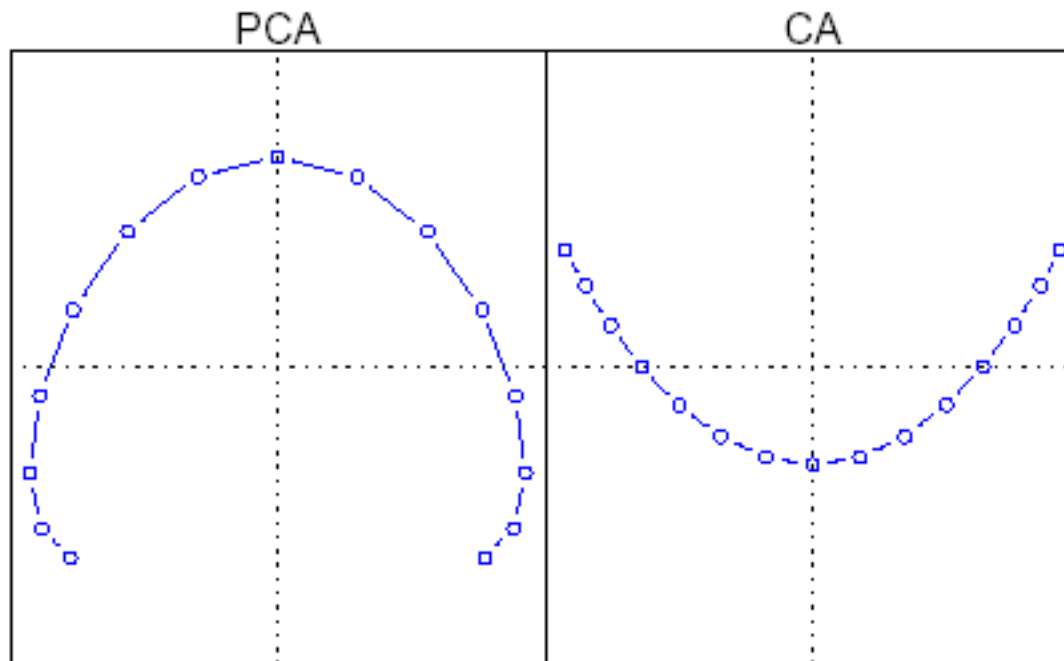


DCA (**D**etrended **C**orrespondence **A**nalysis)

#Detrending usuwa „fałszywą” krzywiznę w ordynacji pojedynczych „silnych” gradientów

#Reskalowanie służy prawidłowemu rozmieszczaniu prób „upakowanych” na końcach gradientu

#Współrzędne DCA wyrażone w jednostkach odchyłeń standardowych, a nie w jednostkach abstrakcyjnych (jak przy PCA i CA)



Kiedy używać PCA, a kiedy CA/DCA?

Najpierw przeprowadź analizę DCA i zobacz jaka jest długość gradientu wzdłuż DCA 1

Jeżeli gradient jest krótszy niż 3(2; 2.5) SD użyj PCA

Jeżeli gradient jest dłuższy niż 3(2; 2.5) SD użyj CA lub DCA

Jeżeli w PCA widać „podkowę”, zrób CA

Jeżeli w CA widać „efekt łuku”, zrób DCA

Jeśli wynik nadal nie jest satysfakcjonujący –
pokombinuj z transformacjami danych i porównaj
wyniki różnych ordynacji

dane w lasy.legowe zawierają spisy roślin naczyniowych runa leśnego na 144 powierzchniach (10x10 m każda) w lasach łęgowych

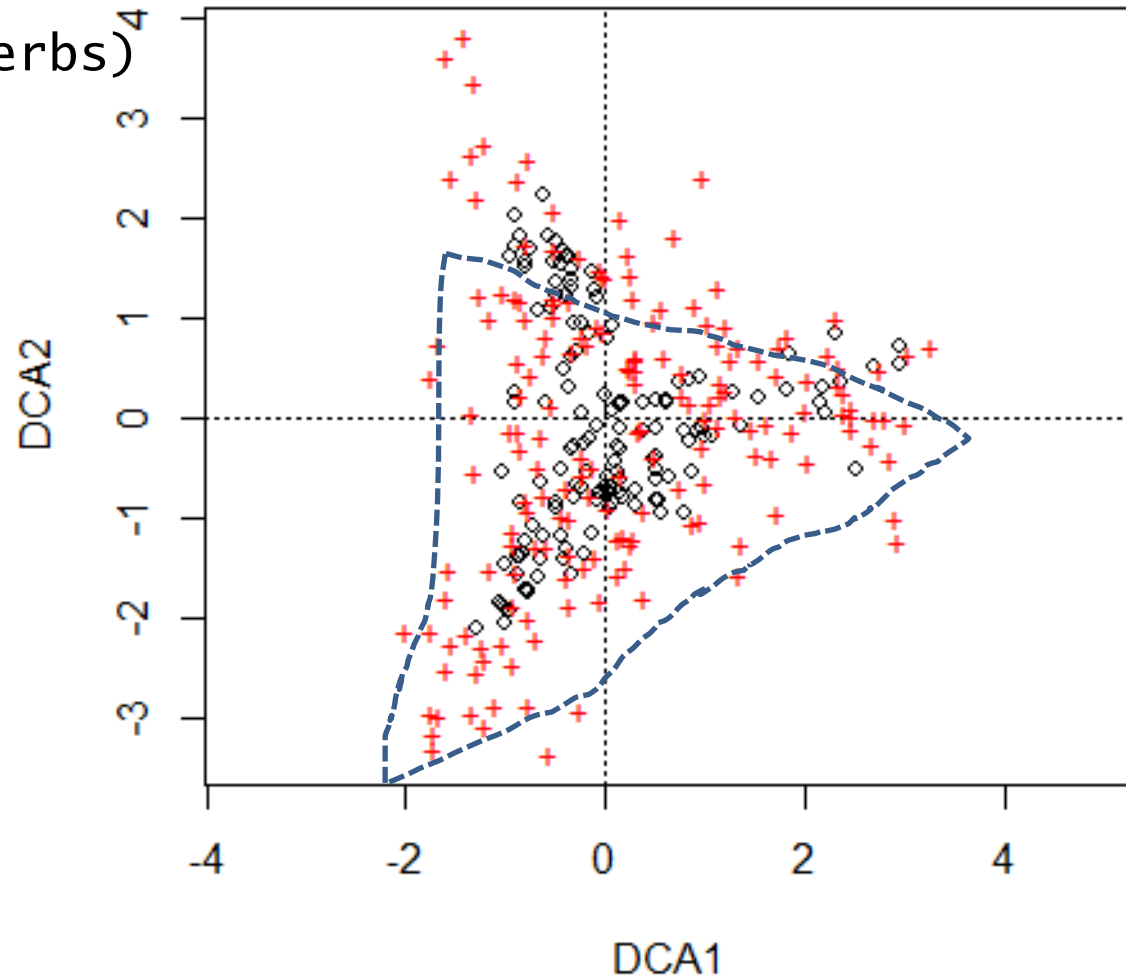
```
legi.dca<-decorana(lasy.legowe)  
legi.dca
```

```
> legi.dca  
  
call:  
decorana(veg = lasy.legowe)  
  
Detrended correspondence analysis with 26 segments.  
Rescaling of axes with 4 iterations.  
  
              DCA1    DCA2    DCA3    DCA4  
Eigenvalues    0.5667  0.5865  0.4240  0.3070  
Decorana values 0.7212  0.5274  0.3865  0.2791  
Axis lengths   4.2485  4.3550  3.5772  3.1609
```

#Długością gradientu jest długość pierwszej osi ordynacyjnej (DCA1). W tym przypadku jest to 4.2485 SD > 2SD, czyli trzeba zrobić DCA, bo gradient bardzo długi

DCA w R

```
dca1<-decorana(samples.herbs)  
plot(dca1, disp="both")
```



#Artefakt przy DCA – efekt trójkąta – gradient za długi
#Alternatywa – analiza DCCA (niedostępna w R, ale dostępna w CANOCO)
#Inna opcja – zrobić NMDS (o nim później)

DCA w ggplot2

#Kod ggplot2 taki sam, jak przy PCA/CA

#Jedyna różnica tkwi w ekstrakcji współrzędnych poletek:

#Poprzednio funkcja scores wypluwała listę, złożoną z 2 poziomów:

\$species

\$sites

#przy DCA jest ramka danych:

```
> scores(dca1)
```

	DCA1	DCA2	DCA3	DCA4
A10h	-0.2027879547	-0.34999702	0.0864932074	6.319943e-02
A11h	-0.5132556311	-0.21540632	-0.0595387368	1.206227e-01
A1h	-0.3192267751	-0.23792235	-0.0676953175	1.045466e-02
A2h	-0.4449501918	-0.33102039	-0.1403017623	1.807098e-01
A3h	-0.4326424484	-0.21225550	0.0257742945	-5.487861e-02
A4h	-0.4685358130	-0.29781081	-0.0787295668	6.296208e-02

#Współrzędne gatunków wyciągamy tak:

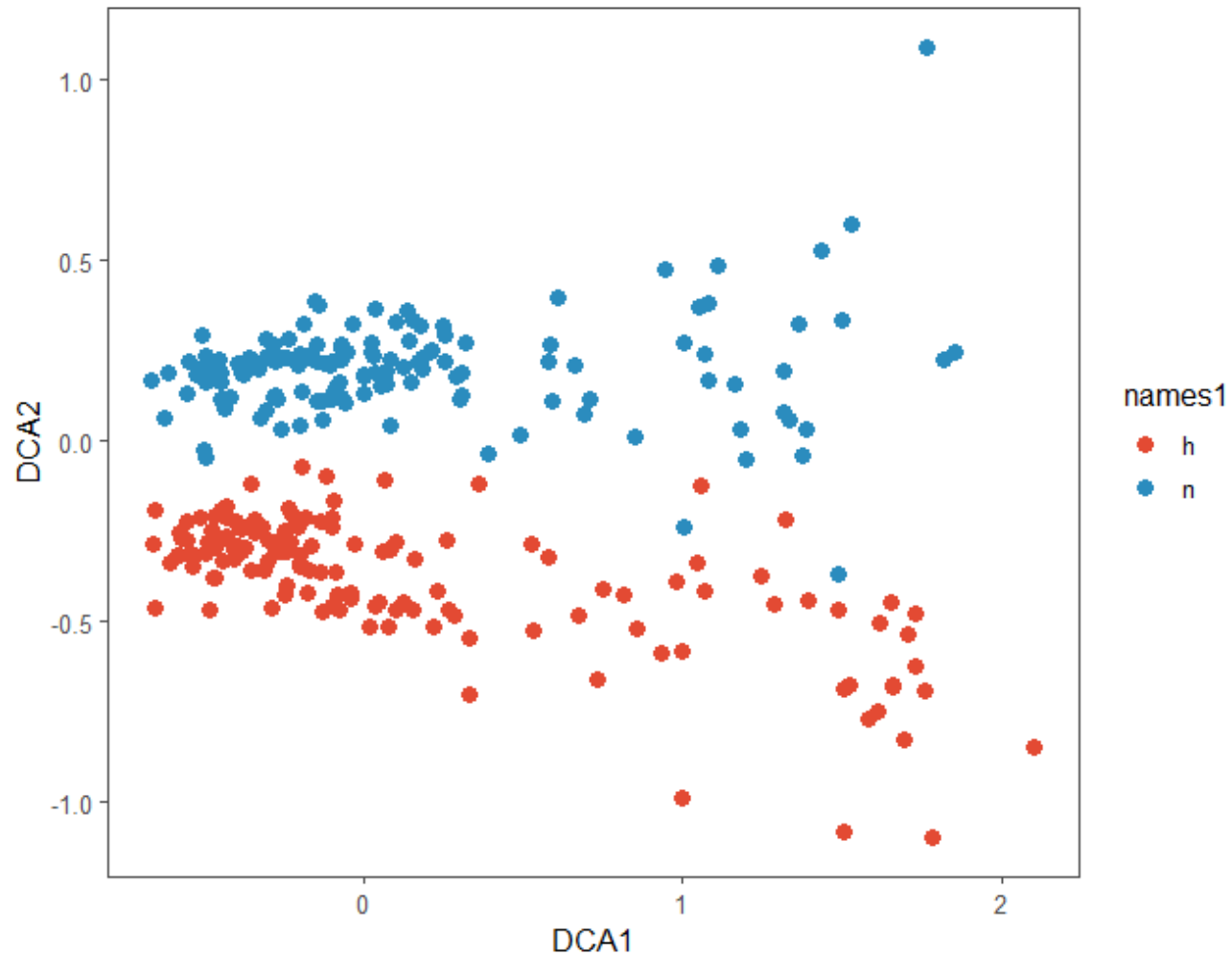
```
scores.dca<-as.data.frame(scores(dca1, choices=c(1,2),  
display="species"))
```

	DCA1	DCA2
Acr.gem	-1.280105250	-0.53763471
Aly.var	-1.478125181	-1.12948755
Ani.bif	-2.048024321	1.38732551
Art.art	-1.041170737	-1.08349774
Art.atr	-1.401570787	-0.01608485
Art.did	-0.417661704	0.51791830
Art.rad	-0.217613129	0.51142205
Art.spa	-0.934152213	1.10163264

#Natomiast współrzędne poletek tak:

```
scores.dca<-as.data.frame(scores(dca1, choices=c(1,2), display="sites"))
```

```
ggplot(scores.dca)+geom_point(aes(x=DCA1, y=DCA2,  
col=names1),size=3,shape=19)+ scale_colour_manual(values=c("#e34a33",  
"#2b8cbe"))+theme_few()
```



Dodawanie wektorów zmiennych pasywnych do PCA/CA/DCA

#Wróćmy do porostów i naszego PCA

env.df1 – ramka danych z charakterystykami wymagań ekologicznych gatunków

```
> env.df1
```

	EIV_L	EIV_T	EIV_K	EIV_F	EIV_N	EIV_R
1	4.500000	5.160000	4.375000	4.250000	3.125000	3.718750
2	4.555556	5.081633	4.634921	4.241379	3.158730	3.857143
3	4.450980	5.475000	4.215686	4.346939	2.921569	4.294118
4	4.333333	5.290909	4.269841	4.180328	3.253968	4.571429
5	4.350000	5.441176	4.300000	4.243243	2.925000	4.100000
6	4.193548	5.388889	4.274194	4.152542	3.225806	4.403226
7	4.519231	5.275000	4.211538	4.240000	3.134615	4.288462
8	4.385965	5.380000	4.245614	4.054545	3.350877	4.403509
9	4.490566	5.404762	4.339623	4.039216	3.226415	4.169811
10	4.245283	5.425532	4.188679	4.226415	3.094340	4.320755
11	4.431373	5.292683	4.254902	4.208333	3.058824	4.196078
12	4.466667	5.360000	4.266667	4.169492	3.116667	4.350000
13	4.425926	5.279070	4.148148	4.301887	3.092593	4.203704
14	4.562500	5.342105	4.354167	4.108696	3.166667	4.208333
15	4.552632	5.206897	4.473684	4.194444	3.052632	4.026316
16	4.242424	5.309091	4.121212	4.241935	3.075758	4.227273


```
vektory<-envfit(pca1, env.df1,  
permutations = 999, strata = NULL, choices=c(1,2))
```

```
> vektory  
  
***VECTORS  
  
          PC1      PC2      r2 Pr(>r)  
EIV_L -0.80043 -0.59943 0.7238 0.001 ***  
EIV_T  0.68031  0.73292 0.8814 0.001 ***  
EIV_K -0.79688 -0.60414 0.8553 0.001 ***  
EIV_F  0.90500  0.42541 0.1571 0.001 ***  
EIV_N  0.76611 -0.64271 0.0566 0.003 **  
EIV_R  0.81655  0.57728 0.8890 0.001 ***  
---  
signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
Permutation: free  
Number of permutations: 999
```

#Wszystkie 6 zmiennych istotnie wyjaśnia różnice w składzie gatunkowym porostów. EIV-R i EIV_T najbardziej (największe r2), a w najmniejszym stopniu EIV_N (najmniejsze r2)

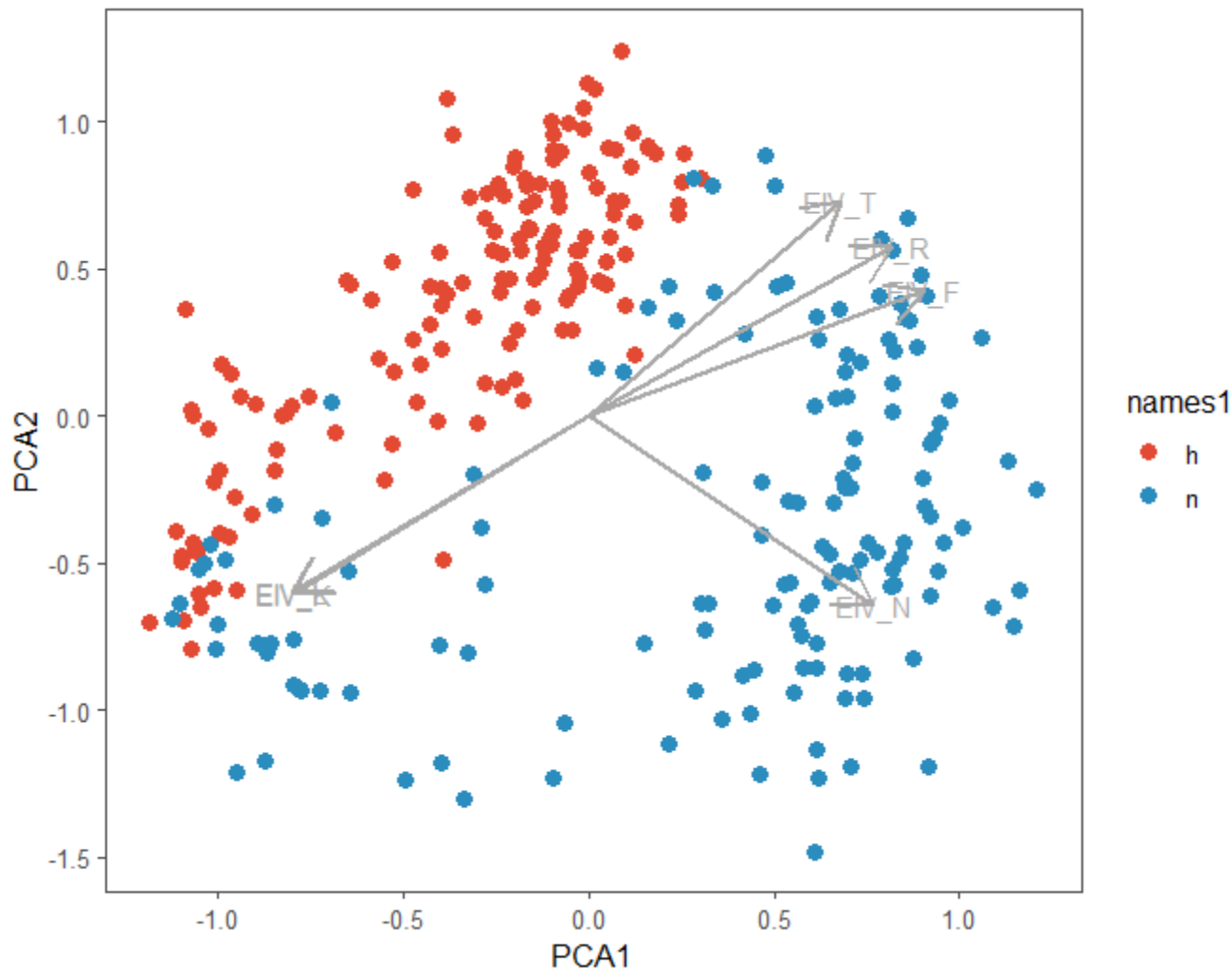
#Pytanie: czy którykolwiek z wektorów może nam powiedzieć coś na temat zmian udziału gatunków o danych wymaganiach ekologicznych w czasie?

#Aby na nie odpowiedzieć, nałożmy te zmienne na diagram PCA
#Najpierw stwórzmy obiekt zawierający dane o współrzędnych początku i końca tych wektorów (na wykresie będą to strzałki):

```
gg.strzałki<-data.frame(PCA1=vektory$vectors$arrows[,1],  
PCA2=vektory$vectors$arrows[,2],  
lab=rownames(epi.envi$vectors$arrows),  
p=epi.envi$vectors$pvals)
```

#A teraz dodajmy ten obiekt do kodu z obrazkiem:

```
ggplot(scores.pca)+geom_point(aes(x=PCA1, y=PCA2,  
col=names1),size=3,shape=19)+  
scale_colour_manual(values=c("#e34a33", "#2b8cbe"))+  
geom_segment(data=gg.strzałki,aes(x=0, xend=PCA1, y=0,  
yend=PCA2),arrow = arrow(length =  
unit(0.6, 'cm')),color='darkgrey', size=.8)+  
geom_text(data=gg.strzałki, aes(x=PCA1,y=PCA2,label=lab),  
color="darkgrey")+theme_few()
```



#No i co?

#Nic – brak wyraźnych zmian pod względem udziału gatunków o określonych wymaganiach ekologicznych w czasie (no... może oprócz EIV_N)

#Wniosek – Zmiany w składzie gatunkowym są kierunkowe, ale mogą zależeć od innych czynników, np. typ zbiorowiska leśnego...

NMDS (**N**on-metric **M**ulti**D**imensional **S**caling)

#Skalowanie metryczne (**PCoA**; funkcja `capscale` lub `cmdscale`) zakłada liniową (metryczną) relację między odległościami ordynacyjnymi a realnymi

#Skalowanie niemetryczne (**NMDS**; funkcja `metaMDS`) rozszerza ten warunek po to, aby znaleźć jakąkolwiek relację

#Metoda bardzo pomocna, gdy inne metody (PCA, CA, DCA) zawodzą

#Ponieważ `metaMDS` pracuje z liczbami pseudolosowymi, aby uzyskać stabilne wyniki, przed wykonaniem analizy należy „zasiać ziarno”, gdzie liczba w nawiasie może być dowolna:

```
set.seed(15266)
```

#Domyślnie NMDS używa odległości Bray-Curtisa jako miarę niepodobieństwa składu gatunkowego poszczególnych poletek
#W przypadku, gdy użycie odległości Bray-Curtisa generuje niesatysfakcjonujące wyniki, można użyć innych miar niepodobieństwa, które definiuje argument „distance”

```
epi.mds<-metaMDS(epi.t, distance="bray")
```

```
#Lub
```

```
epi.mds<-metaMDS(epi.t, distance=„euclidean")
```

```
#Lub
```

```
epi.mds<-metaMDS(epi.t, distance=„jaccard")
```

```
#itd.
```

#Duża różnorodność metod – ściąga w funkcji „vegdist” z pakietu „vegan”

Usage

```
vegdist(x, method="bray", binary=FALSE, diag=FALSE, upper=FALSE,  
        na.rm = FALSE, ...)
```

Arguments

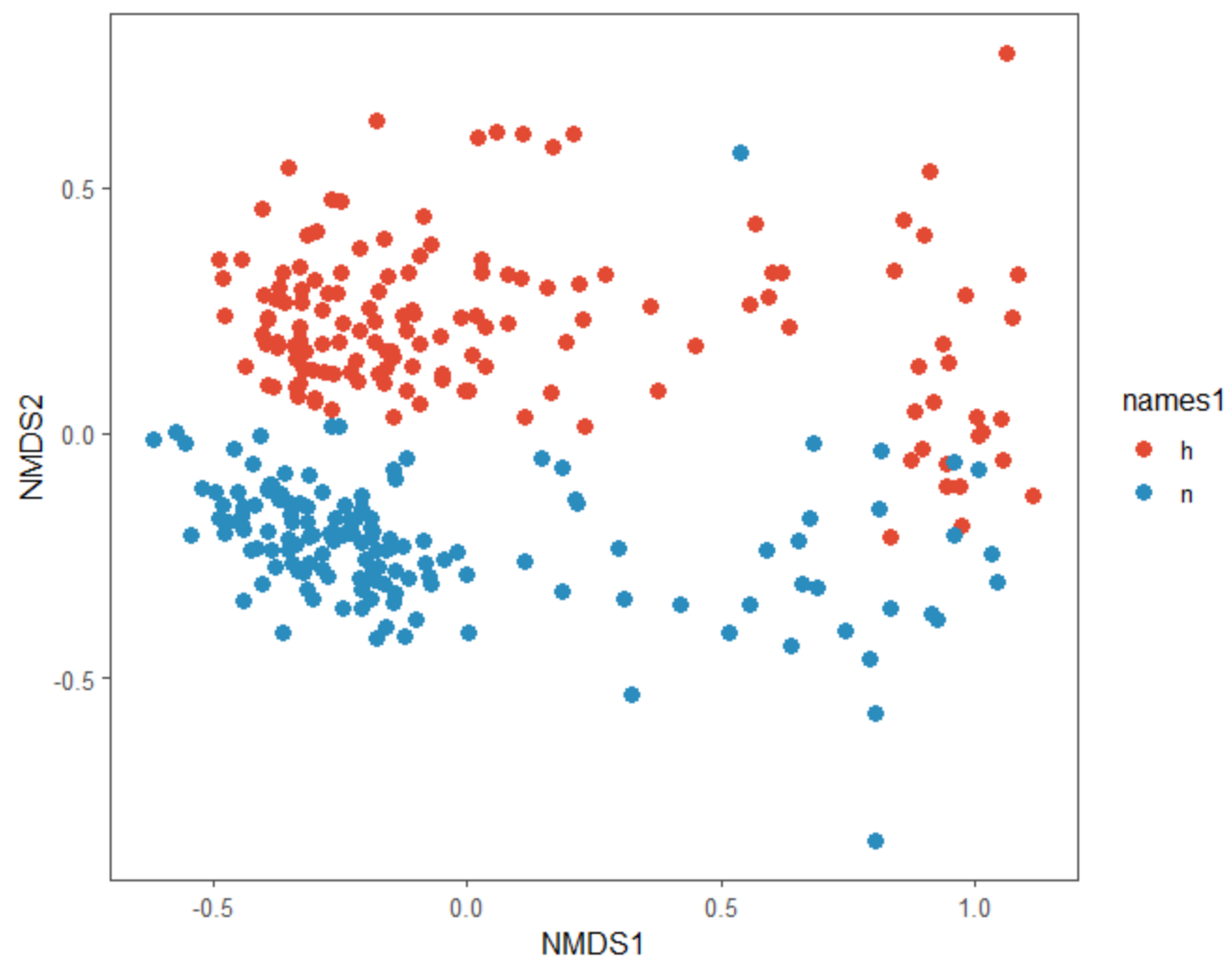
x	Community data matrix.
method	Dissimilarity index, partial match to "manhattan", "euclidean", "canberra", "clark", "bray", "kulczynski", "jaccard", "gower", "altGower", "morisita", "horn", "mountford", "raup", "binomial", "chao", "cao", "mahalanobis", "chisq" or "chord".

Wykres w ggplot2

#ekstrakcja współrzędnych poletek/gatunków tak samo, jak przy DCA

```
scores.mds<-as.data.frame(scores(epi.mds, choices=c(1,2),  
display="sites"))
```

```
ggplot(scores.mds)+geom_point(aes(x=NMDS1, y=NMDS2,  
col=names1),size=3,shape=19)+  
scale_colour_manual(values=c("#e34a33",  
"#2b8cbe"))+theme_few()
```



Diagnoza poprawności analizy NMDS

#Dwie opcje:

#Funkcja „goodness” z pakietu „vegan” – goodness of fit (GOF) - pokazuje jak bardzo wynik analizy zafałszowuje realną zmienność kompozycji gatunkowej plotek badawczych

#Interpretacja:

Duże wartości (większe niż 0.2) mówią o dużym zafałszowaniu

Małe wartości (mniejsze niż 0.01/0.001) mówią o niskim zafałszowaniu i dobrym dopasowaniu analizy:

```
good1<-goodness(epi.mds)
max(good1)
min(good1)
mean(good1)
```

```
> good1<-goodness(mds1)
> good1
[1] 0.005294803 0.006824213 0.008415028 0.006600813 0.005294803 0.010744912
[7] 0.012319171 0.009184311 0.012901856 0.010306458 0.011953330 0.005035505
[13] 0.012932511 0.005874651 0.011482612 0.014066376 0.005024934 0.030617995
[19] 0.009633527 0.005513087 0.006100058 0.009195815 0.013785275 0.012011908
[25] 0.010109707 0.008920893 0.026002306 0.023454563 0.018233214 0.012572743
```

```
> mean(good1)
[1] 0.01083353
> max(good1)
[1] 0.03061799
> min(good1)
[1] 0.005024934
> mean(good1)
[1] 0.01083353
```

#good1 – pokazuje GOF dla każdego poletka

#W publikacji można podać minimalne, maksymalne i średnie wartości GOF

#Druga opcja:

#Obliczenie wartości tzw. stresu („stress values”)

#Stress values mierzą różnice w odległościach pomiędzy poletkami w przestrzeni ordynacyjnej zredukowanej do dwóch wymiarów (osi NMDS1 i NMDS2) w porównaniu do odległości w całkowitej przestrzeni wielowymiarowej

#Bardzo małe wartości (mniejsze niż 0.001) mówią o tym, że dwie pierwsze osie wyjaśniają większość zmienności

#Bardzo duże wartości (większe niż 0.2) mówią, że rozmieszczenie naszych poletek wzdłuż 2 pierwszych osi jest losowe i nie ma żadnego gradientu zmienności czy różnic w składzie gatunkowym:

```
> epi.mds$stress  
[1] 0.1335867
```

#W naszym przypadku jest w miarę OK

Kiedy używać PCA, a kiedy CA/DCA?

Najpierw przeprowadź analizę DCA i zobacz jaka jest długość gradientu
wzdłuż DCA 1

Jeżeli gradient jest krótszy niż 3(2; 2.5) SD użyj PCA

Jeżeli gradient jest dłuższy niż 3(2; 2.5) SD użyj CA lub DCA

Jeżeli w PCA widać „podkowę”, zrób CA

Jeżeli w CA widać „efekt łuku”, zrób DCA

Jeśli wynik nadal nie jest satysfakcjonujący – pokombinuj z transformacjami
danych i porównaj wyniki różnych ordynacji

Jeśli wynik nadal nie satysfakcjonuje – analiza ostatniej szansy NMDS +
transformacje + testowanie różnych miar odległości (niepodobieństwa)

Metody ordynacji bezpośredniej

Ordynacja i regresja w jednym

#Przedstawiają zmienne środowiskowe w sposób **aktywny** (dodanie do analizy zmiennych środowiskowych modyfikuje rozmieszczenie punktów w przestrzeni ordynacyjnej)

#Przy ordynacji pośredniej **pasywne** nakładanie zmiennych (po dodaniu do analizy zmiennych środowiskowych brak modyfikacji rozmieszczenia punktów w przestrzeni ordynacyjnej)

Redundancy analysis (**RDA**) \equiv constrained or canonical **PCA**

Canonical correspondence analysis (**CCA**) \equiv constrained **CA**

CCA (**C**anonical **C**orrespondence **A**nalysis)

Dobra do analizy długich gradientów (>3SD; opcjonalnie 2 lub 2.5SD)

Co nam daje:

#Odległości pomiędzy próbami (**site scores**), czyli podstawowy gradient zmienności

#Odległości między gatunkami (**species scores**), czyli lokalizacja optimum występowania gatunków w przestrzeni w zależności od lokalizacji prób (sites)

#Dodatkowo – odległości środowiskowe (**biplot scores**), które definiują przestrzeń gradientu

CCA w R

#W kolumnach obiektu ponds.spp zawarto nazwy 48 gatunków niesporczaków (wyrażone kodami), stwierdzone w 30 próbach wody (wiersze). Dane w komórkach zawierają biomasę każdego gatunku w każdej próbie

```
> ponds.spp
```

	AC001A	AC013A	AC013E	AM011A	AM012A	AS001A	AU002A	AU003B	CC001A	CC002A	CC9997	CM004A	CO001A	CY002A	CY003A
4	0.00	0.55	0.00	0.74	0.92	1.66	4.60	0.00	0.00	0.00	0.00	0.00	0.18	1.11	0.00
7	0.36	3.40	0.00	1.07	8.05	0.36	0.00	0.00	2.15	3.40	0.00	0.00	3.94	1.97	3.04
31	0.90	1.08	0.00	0.90	5.39	0.00	0.00	0.00	0.00	0.18	0.18	0.00	0.72	0.00	0.00
34	0.17	0.52	0.00	0.69	0.35	0.00	0.00	0.00	9.69	7.96	15.23	0.00	0.52	3.46	2.77
37	0.00	6.84	0.00	2.54	2.34	0.19	0.00	0.00	0.00	0.00	0.00	0.00	2.15	0.59	0.19
42	0.18	0.91	0.00	0.00	0.73	0.36	14.03	0.00	0.00	0.00	0.00	0.00	0.91	0.00	0.00
50	1.60	10.02	0.00	0.00	0.00	2.40	0.00	0.00	0.20	0.20	0.00	0.00	2.40	25.65	1.20
53	0.56	2.64	0.00	0.94	0.00	2.83	9.60	0.00	0.00	2.26	0.00	0.00	0.00	10.92	0.94
57	31.38	1.38	0.00	0.34	0.34	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.17
58	0.51	7.78	0.00	0.34	0.00	1.18	0.34	0.00	0.00	0.00	0.00	0.00	4.06	1.69	1.35
65	0.00	0.35	0.00	0.00	0.00	0.71	0.00	0.18	0.00	9.73	3.19	0.00	0.35	16.11	29.56
69	1.70	5.59	0.00	0.00	0.85	0.00	0.00	0.00	0.00	3.22	3.05	0.00	1.02	12.37	1.86
73	2.19	3.03	0.84	1.35	0.00	0.00	0.00	0.00	0.00	6.90	0.00	0.00	1.51	7.41	2.02

#Obiekt ponds.env zawiera zbiór parametrów fizykochemicznych wody, również obliczone dla każdej z 30 prób

```
> colnames(ponds.env)
```

[1]	"pH"	"conductivity"	"Alkalinity"	"TP"	"SiO2"	"NO3"	"Na"
[8]	"K"	"Mg"	"Ca"	"Cl"	"SO4"	"Chla"	"Secchi"
[15]	"Maxdept"						

#Pełny zapis CCA:

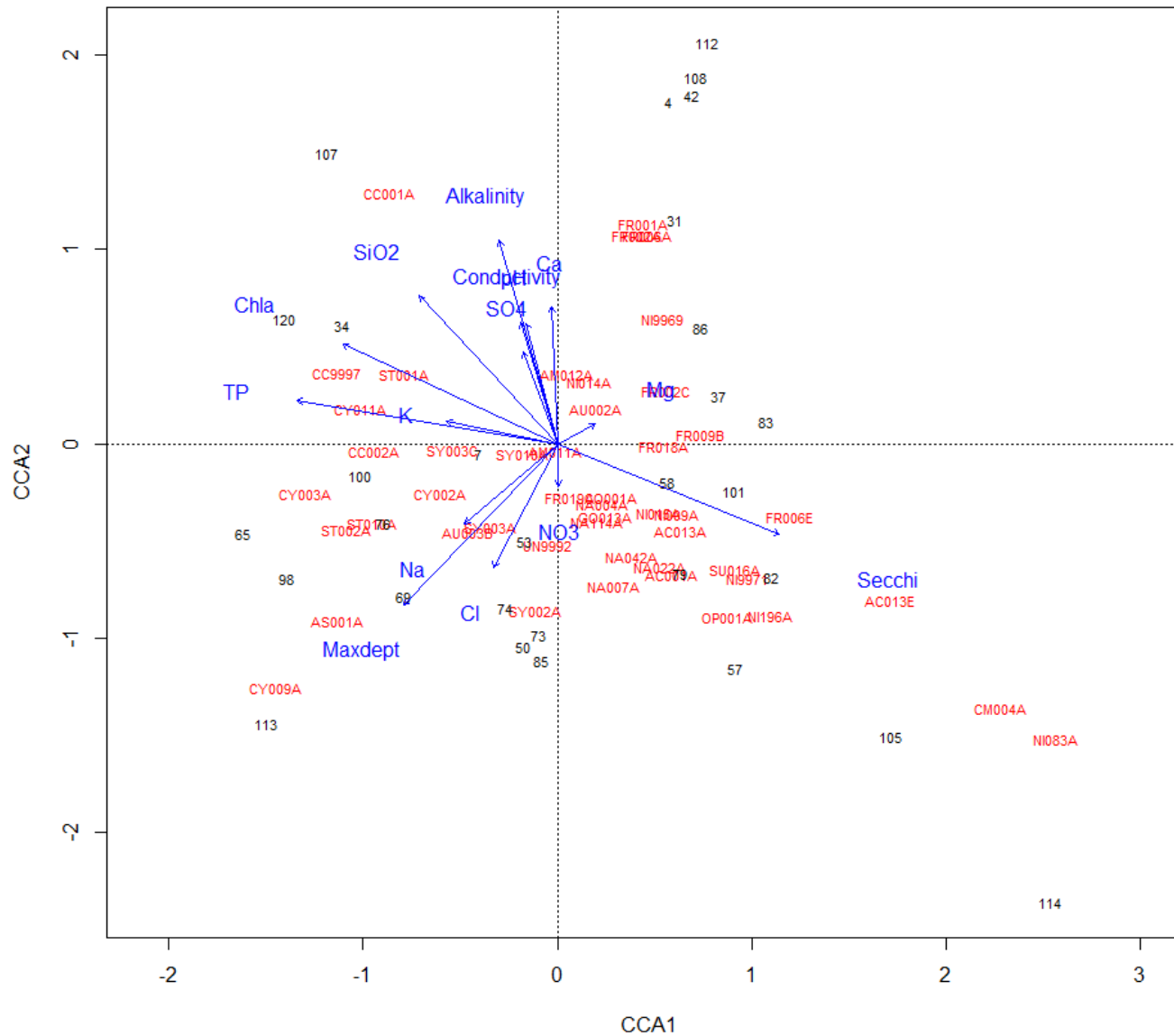
```
ponds.cca<-cca(ponds.spp~pH+Conductivity+  
Alkalinity+TP+SiO2+NO3+Na+K+Mg+Ca+Cl+S04+  
Chla+Secchi+Maxdept, data=ponds.env)
```

#Skrócona forma zapisu

```
ponds.cca<-cca(ponds.env~., data=ponds.env)
```

Wykres

plot(ponds.cca)



CCA triplot:

Ciągłe zmienne
środowiskowe
(strzałki)

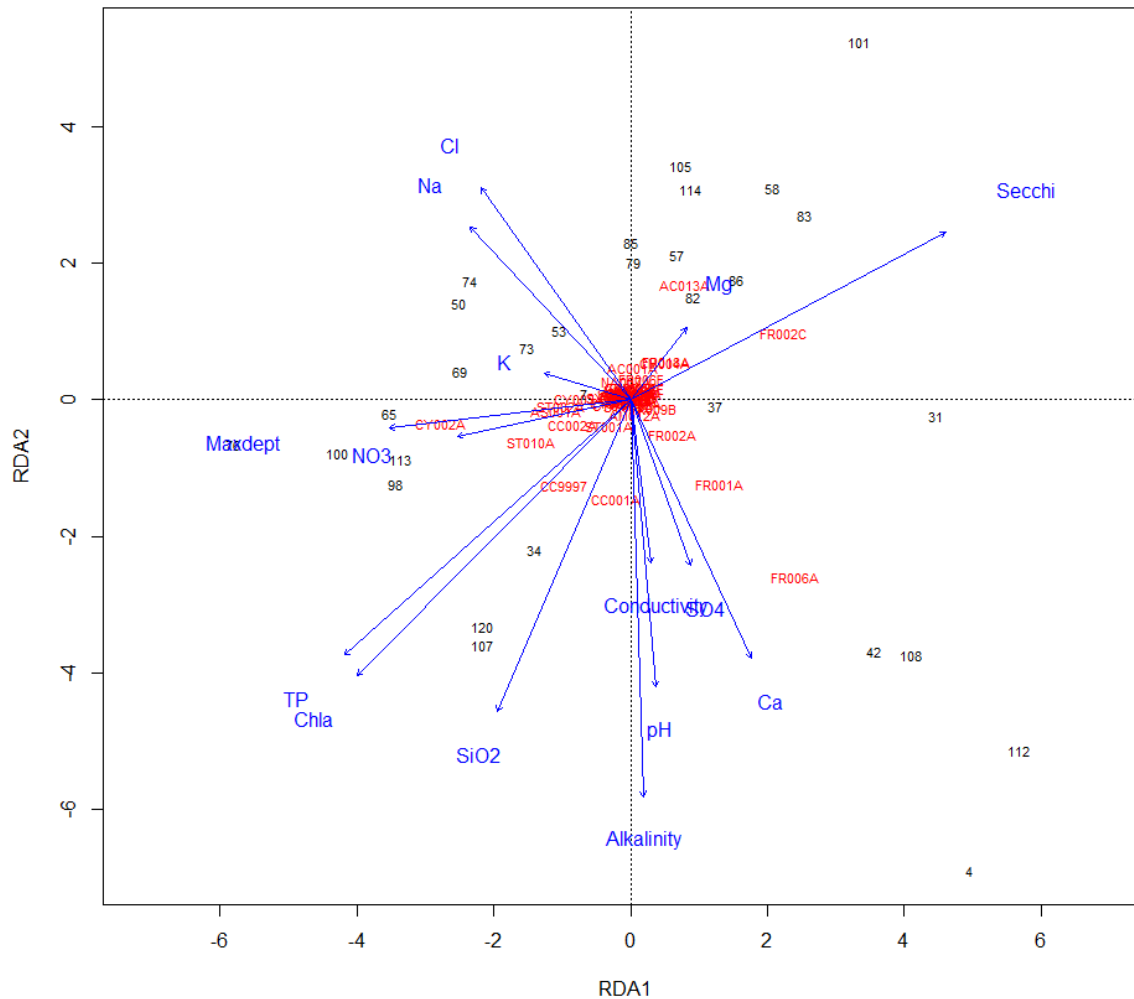
Próby jako czarne
liczby

Gatunki na czerwono

RDA (ReDundancy Analysis)

Dobra do analizy krótkich gradientów (<3SD; opcjonalnie 2 lub 2.5SD)

```
ponds.rda<-rda(ponds.spp~., data=ponds.env)
plot(ponds.rda)
```



Ustalenie ważności zmiennych

#Variance Inflation Factors – duże wartości VIF dla zmiennych świadczą o dużej sile korelacji tych zmiennych z innymi

```
vif.cca(ponds.cca)
```

```
> vif.cca(ponds.cca)
```

pH	Conductivity	Alkalinity	TP	SiO2
7.238890	30.262396	16.344538	10.386463	5.018683
NO3	Na	K	Mg	Ca
2.179886	43.888206	8.870264	23.694856	6.633363
Cl	SO4	Chla	Secchi	Maxdept
36.752869	18.663534	3.755310	2.165980	2.168597

#Wyrzucamy z zestawu cechy o największych wartościach VIF (powyżej 10)

#Dlaczego powyżej 10?

#Bo tak jest umownie przyjęte, ale niektórzy autorzy w finalnym modelu nie uwzględniają predyktorów już przy wartościach $VIF > 5$:

”To check for collinearity between independent variables, variance inflation factors (VIF) were calculated using the corvif function in the AED package in R”
- Zuur et al. (2009)

„VIF above 5 indicate high multicollinearity between independent variables (Sileshi 2014). While many biomass studies include variables that are highly correlated (e.g., diameter and height), we avoided this to ensure that the parameter estimates represented causal relationships as closely as possible -
Forrester et al. (2017)

#Wyrzucamy z zestawu cechy o największych wartościach VIF

```
colnames(ponds.env)
```

```
> colnames(ponds.env)
[1] "pH"          "conductivity" "Alkalinity"   "TP"
[5] "SiO2"        "NO3"          "Na"           "K"
[9] "Mg"          "Ca"           "Cl"           "SO4"
[13] "chl a"       "Secchi"       "Maxdept"
```

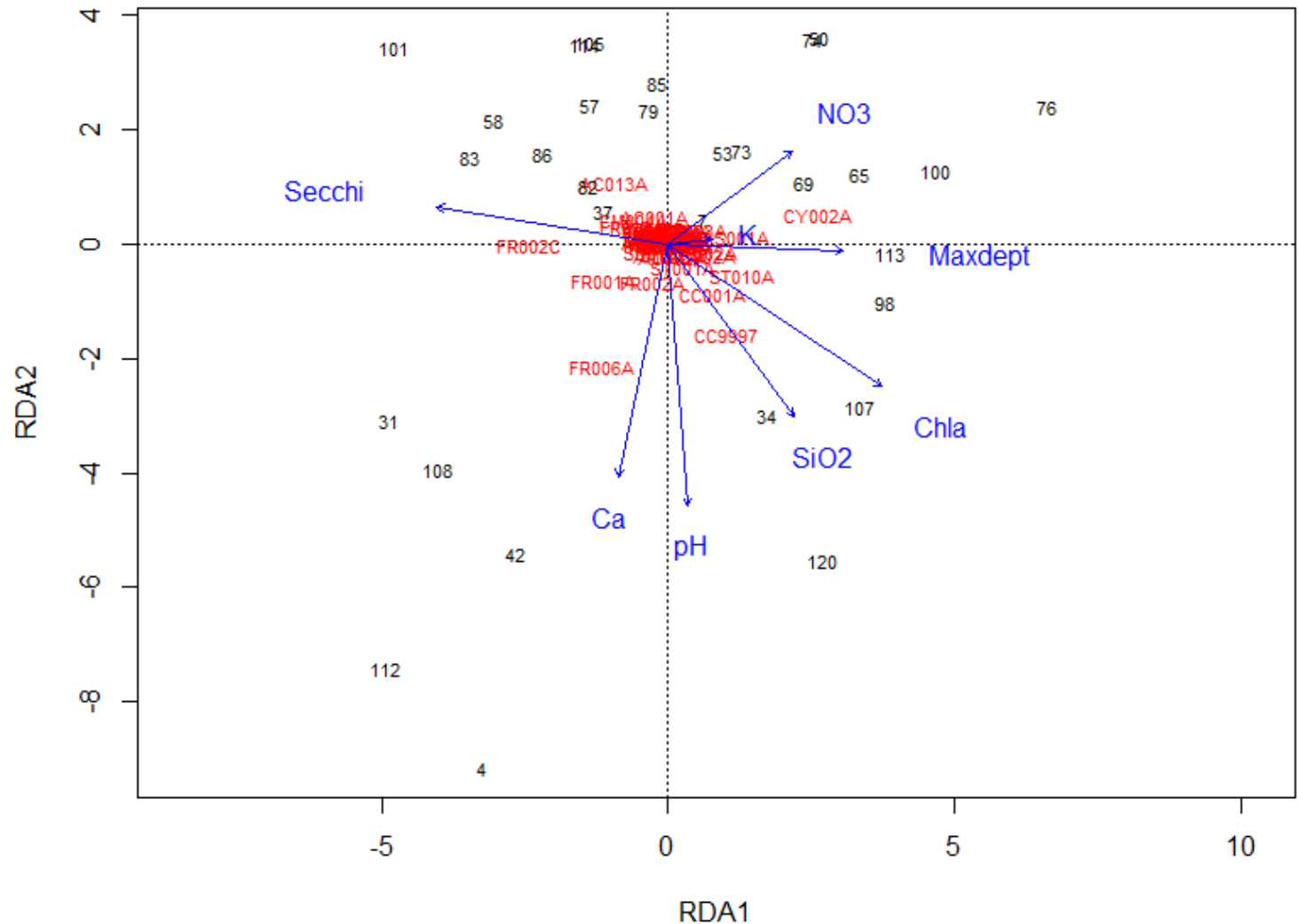
```
ponds2<-ponds.env[, -c(2,3,4,7,9,11,12)]
colnames(ponds2)
```

```
> colnames(ponds2)
[1] "pH"          "SiO2"        "NO3"         "K"           "Ca"          "chl a"       "Secchi"
[8] "Maxdept"
```

#Na podstawie VIF, cechy w ponds2 w największym stopniu mogą wpływać na biomasę niesporczaków.

#Wniosek z analizy VIF – do naszego CCA powinniśmy użyć mniej predyktorów
#Zróbmy więc CCA z mniejszą liczbą zmiennych:

```
ponds.cca2<-cca(ponds.spp~., data=ponds2)  
plot(ponds.cca2)
```



#Okej, ale czy nasz model jest istotny statystycznie?

#Istotność modelu można sprawdzić wykonując analizę PERMANOVA:

```
anova(ponds.cca2)
```

```
> anova(ponds.cca2)
Permutation test for rda under reduced model
Permutation: free
Number of permutations: 999

Model: rda(formula = ponds.spp ~ pH + SiO2 + NO3 + K + Ca + Chl.a + Secchi + Maxdepth, data = ponds2)
      Df Variance      F Pr(>F)
Model    8   400.01 1.5655 0.005 **
Residual 21   670.71
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#Okej, model istotny, ale jak sprawdzić, które zmienne istotnie wpływają na biomasę niesporczaków?


```
anova(ponds.cca.final, by='terms')
```

```
> anova(ponds.cca2, by='terms')
Permutation test for rda under reduced model
Terms added sequentially (first to last)
Permutation: free
Number of permutations: 999

Model: rda(formula = ponds.spp ~ pH + SiO2 + NO3 + K + Ca + Chla + Secchi + Maxdept, data = ponds2)
  Df Variance      F Pr(>F)
pH   1    43.22 1.3532 0.174
SiO2  1    43.50 1.3620 0.161
NO3   1    42.09 1.3178 0.184
K     1    26.11 0.8175 0.611
Ca    1    44.81 1.4029 0.165
Chla  1    58.33 1.8264 0.035 *
Secchi 1    41.23 1.2910 0.204
Maxdept 1  100.72 3.1535 0.003 **
Residual 21   670.71
---
signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#Wychodzi, że tylko dwa predyktory są istotne, ale czy ten model najlepiej opisuje wpływ właściwości fizykochemicznych na biomasę niesporczaków?

#Aby to sprawdzić, można przeprowadzić krokową selekcję zmiennych i na podstawie najmniejszego AIC wybrać model, który najlepiej opisuje zależność

#do tego służy `vegan::step()`

`step(ponds.cca2)`

NIEMNIEJ JEDNAK,
JEST TO
NIEBEZPIECZNE
NARZĘDZIE!!!

```
> step(ponds.cca2)
Start:  AIC=212.23
ponds.spp ~ pH + SiO2 + NO3 + K + Ca + Chla + Secchi + Maxdept

      Df    AIC
- pH      1 211.52
- K        1 211.71
- SiO2     1 211.79
- Chla     1 211.81
- Ca       1 211.99
<none>      212.23
- Secchi   1 212.44
- NO3      1 212.75
- Maxdept  1 214.43

Step:  AIC=211.52
ponds.spp ~ SiO2 + NO3 + K + Ca + Chla + Secchi + Maxdept

      Df    AIC
- K        1 210.98
- Chla     1 211.08
- SiO2     1 211.10
- Ca       1 211.46
<none>      211.52
- Secchi   1 211.67
- NO3      1 212.00
- Maxdept  1 213.69

Step:  AIC=210.98
ponds.spp ~ SiO2 + NO3 + Ca + Chla + Secchi + Maxdept

      Df    AIC
- SiO2     1 210.28
- Ca       1 210.34
- Chla     1 210.47
<none>      210.98
- Secchi   1 211.05
```

#Model finalny:

Step: AIC=209.06

ponds.spp ~ NO3 + Secchi + Maxdept

	Df	AIC
<none>		209.06
- NO3	1	209.18
- Secchi	1	210.17
- Maxdept	1	210.56

Call: rda(formula = ponds.spp ~ NO3 + Secchi + Maxdept, data = ponds2)

	Inertia	Proportion	Rank
Total	1070.7136	1.0000	
Constrained	228.5709	0.2135	3
Unconstrained	842.1428	0.7865	26

Inertia is variance

Eigenvalues for constrained axes:

RDA1	RDA2	RDA3
131.12	59.69	37.76

Eigenvalues for unconstrained axes:

PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
205.34	115.72	92.32	69.86	61.23	53.24	39.60	37.98

(Showing 8 of 26 unconstrained eigenvalues)

#Zróbmy więc CCA z trzema predyktorami, uwzględnionymi w modelu finalnym:

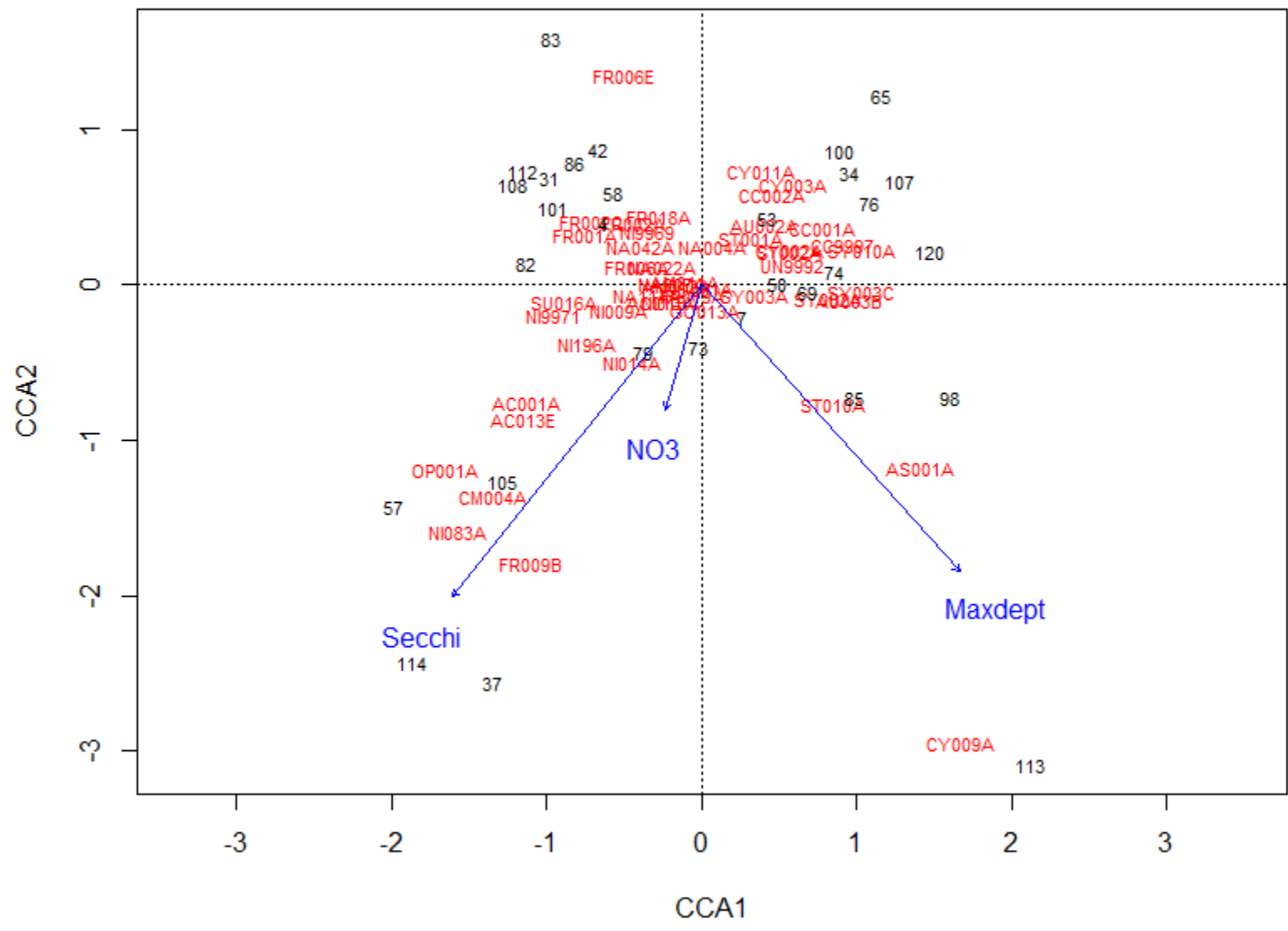
```
cca.final<-cca(ponds.spp~NO3 + Secchi + Maxdept,  
data=ponds2)  
plot(cca.final)
```

#i PERMANOVA

```
anova(cca.final, by="terms")
```

```
> anova(cca.final, by="terms")  
Permutation test for cca under reduced model  
Terms added sequentially (first to last)  
Permutation: free  
Number of permutations: 999  
  
Model: cca(formula = ponds.spp ~ NO3 + secchi + Maxdept, data = ponds2)  
      Df Chisquare      F Pr(>F)  
NO3      1      0.2302 1.2447  0.152  
Secchi    1      0.3617 1.9555  0.003 **  
Maxdept   1      0.4114 2.2245  0.001 ***  
Residual 26      4.8089  
---  
signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#Tylko przezroczność i głębokość wody istotnie wpływa na biomasę niesporczaków



#Ale ile procent zmienności wyjaśniają nasze modele?

#Obliczmy ich R2 i porównajmy ze sobą

```
> RsquareAdj(ponds.cca2)
$r.squared
[1] 0.3735895

$adj.r.squared
[1] 0.134957
```

```
> RsquareAdj(cca.final)
$r.squared
[1] 0.1726241

$adj.r.squared
[1] 0.07863976
```

#Rodzi się więc pytanie...

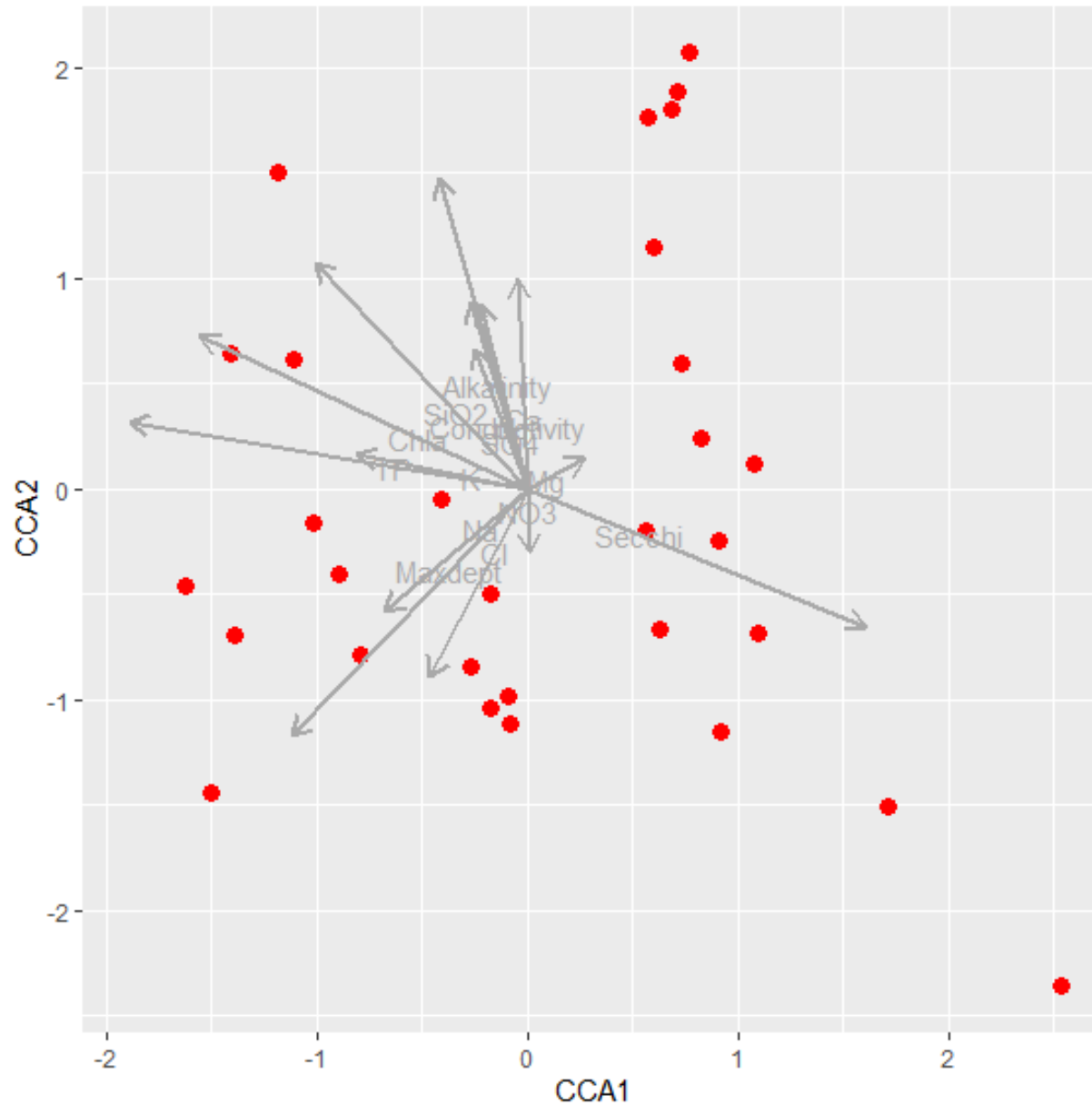
...jak byście mieli do wyboru model bardziej skomplikowany z większą ilością predyktorów i większym R2

...oraz model prostszy z mniejszą liczbą bardziej istotnych predyktorów, ale z mniejszym R2

...to co byście wybrali?



Tworzenie wykresu RDA/CCA w ggplot2



#Przeprowadzenie RDA/CCA

```
ponds.cca<-cca(ponds.spp~., data=ponds.env)
```

#Ekstrakcja współrzędnych dla pierwszej i drugiej osi ordynacyjnej (dla RDA i CCA tak samo)

```
sites<-as.data.frame(scores(ponds.cca)$sites)
```

#Ekstrakcja współrzędnych aktywnie nałożonych zmiennych środowiskowych wzdłuż CCA1 i CCA2

```
szczałki<-data.frame(CCA1=ponds.cca$CCA$biplot[,1],  
CCA2=ponds.cca$CCA$biplot[,2])
```


#Wykres

```
ggplot(sites)+  
  geom_point(aes(x=CCA1,y=CCA2), col="red", size=3)+  
  geom_segment(data=szcza1ki,  
              aes(x=0, xend=CCA1, y=0, yend=CCA2),  
              arrow = arrow(length = unit(0.3,'cm'))),  
              color='darkgray', size=.9)+  
  geom_text(data=as.data.frame(rownames(szcza1ki)),  
            aes(x=szcza1ki$CCA1,y=szcza1ki$CCA2,  
                label=rownames(szcza1ki)),  
            color="darkgray")
```

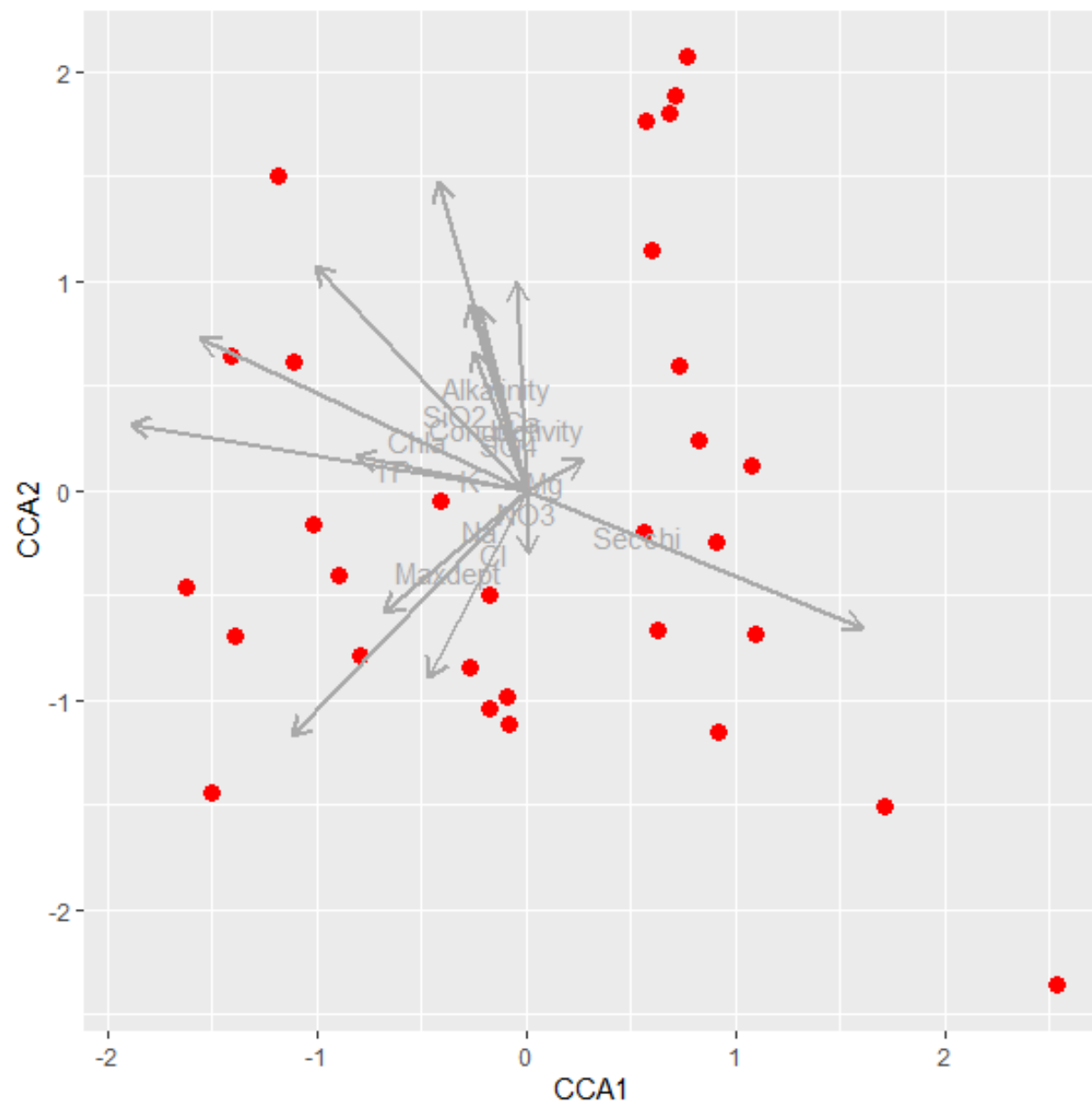


#Wykres nieczytelny?

#Co można zrobić?

#Pomnożyć współrzędne wektorów np. razy 3

```
ggplot(sites)+  
  geom_point(aes(x=CCA1,y=CCA2), col="red", size=3)+  
  theme_bw()+theme(panel.grid = element_blank())+  
  geom_segment(data=szczałki*3,  
               aes(x=0, xend=CCA1, y=0, yend=CCA2),  
               arrow = arrow(length = unit(0.3,'cm')),  
               color='darkgray', size=.9)+  
  geom_text(data=as.data.frame(rownames(szczałki)),  
            aes(x=szczałki$CCA1,y=szczałki$CCA2,  
                label=rownames(szczałki)),  
            color="darkgray")
```





BSS
BIAŁOWIESKA SZKOŁA STATYSTYKI