Msc Eduardo Verri
Dra Marise Miranda

CÁLCULO EM APLICAÇÕES GENÉRICAS COMO MODELOS

Como falamos em regressão linear os modelo tende a regredir uma única reta que melhor represente o conjunto de dados. Isto quer dizer, que o modelo de muitas possíveis retas de no mínimo 2 pontos, tendem a derivar para uma única reta que melhor represente todas.

Contexto 1: Futebol
Problema: A cobrança de falta da cena abaixo é dada por uma parábola, com altura no ponto máximo desconhecida e distância estimada.

CET – Boletim Técnico 62 - Cobrança de Pênalti página 15.
Souza, Cláudio Pires e Albuquerque de Circuito fechado de televisão e vídeo digital: Partes I e II: História e Tecnologia/ Cláudio Pires e Albuquerque de Souza, São Paulo, 2019 251 p. : il. color

O objetivo é **ensinar uma ML(Machine learning)** a executar o cálculo da altura alcançada e a distância percorrida pela bola. Não está sendo considera a força exercida no chute, apenas a trajetória da bola que foi filmada.

A equação do chute em cobrança de falta é dada pelo modelo parabólico demostrado na figura. A bola chutada tem uma altura na curvatura e a distância que vai bater no chão pela primeira vez.

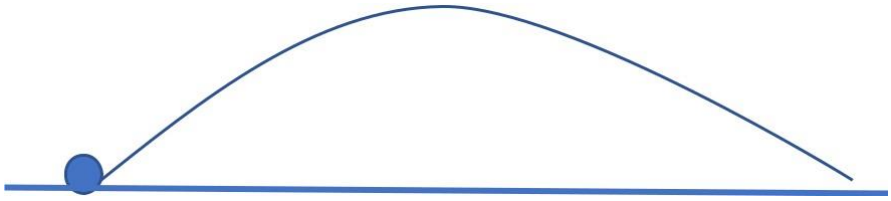
$$F(x) = - x^{**}2 -2^{*} x +1$$

A trajetória da bola confirma a equação
É uma equação parabólica

PARTE 5 – Cálculo - Modelagem

Boca da parábola para baixo
Ponto máximo da parábola é a altura máxima que a bola atinge na

curvatura Ponto do primeiro impacto pós chute é o ponto final da parábola.



A derivada explica qual distância percorrida pela bola e qual altura que a bola alcançará.

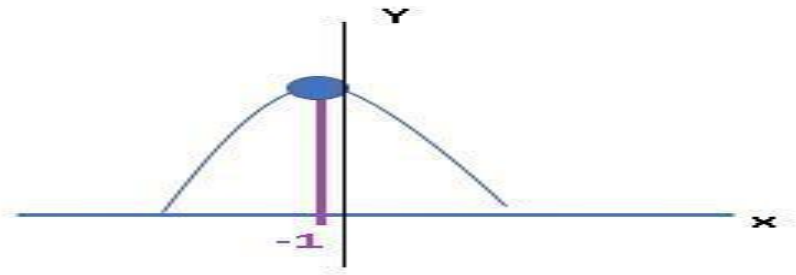
Derivando a função:

F'(x) = -2*x -2 → -2*x -2 =0 → - x = 2/ 2 → x = -1

Substituindo

F(-1) = - ((-1)**2) -2* (-1) +1 = -1 +2 +1 = 2

Par ordenado x,y será (-1,2)



Verificamos que o ponto de máximo é igual a 2, portanto a altura máxima que a bola alcançou foi 2 metros.
Agora vamos calcular a distância percorrida. Calcular as raízes da parábola.

F(x) = - x**2 -2* x +1

a= -1
b= -2
c= 1

A derivada

- x1 = (-b -sqrt(b^2 - 4ac))/2a
- x2 = (-b +sqrt(b^2 - 4ac))/2a

> a<- -1

PARTE 5 – Cálculo - Modelagem

```
> b<- -2
> c<- 1

> x1<- (-b -sqrt(b^2 -4*a*c))/(2*a)
> x2<- (-b +sqrt(b^2 -4*a*c))/(2*a)
> y<- function(x){a*x**2+b*x +c}
> data.frame("Root"=c(x1,x2), "Value"=c(y(x1),y(x2)))

1  0.4142136 -2.220446e-16
2 -2.4142136 0.000000e+00
 > abs(x1)
[1] 0.4142136
> abs(x2)
[1] 2.414214
```

A distância derivada percorrida da trajetória da bola, entre o ponto inicial do chute ao pênalti e a primeira queda bola, é de 2,82 metros.

Resultado:
Altura máxima que a bola alcança é de 2 metros, e a distância alcançada pelo chute do penalti foi de 2,82 metros de comprimento.

Contexto 2: ANÁLISE DA VARIÂNCIA (ANOVA) ★ ★ ★ ★ ★

Análise de variância é a técnica estatística que permite avaliar afirmações sobre as médias de populações. A análise visa, fundamentalmente, verificar se existe uma diferença significativa entre as médias e se os fatores exercem influência em alguma variável dependente.

Variância: é a variação das médias em relação a cada dado, veja o exemplo abaixo:

Tomemos 8 medidas iniciais da tabela Diamons (data set nativo do R) a coluna price. Calculamos a média dessas 8 medidas e depois subtraimos cada medida de price pela média. Vamos ter uma variação dessa diferença, esta se denomina a variância. Se é muito alta ou baixa, comparada outro conjunto distinto com 8 medidas diferentes na mesma coluna da tabela.

Price	diferença = (price - média)	eleva ao quadrado
326	-6,1	37,5
326	-6,1	37,5
327	-5,1	26,3
334	1,9	3,5
335	2,9	8,3
336	3,9	15,0
336	3,9	15,0
337	4,9	23,8
média	**332,1**	**20,9 VARIÂNCIA**

PARTE 5 – Cálculo - Modelagem

Vamos analisar o dataset nativo do R – Diamonds

Este data set necessita de pacotes como o ggplot, ggplot2 através do package tidyverse.

Aqui, vemos que há 10 variáveis no total (três fatores ordenados, um inteiro e 6 numéricas). Um bônus adicional de trabalhar com um conjunto de dados integrado é que a documentação com descrições e explicações adicionais está disponível na página de ajuda

```
> ?diamonds
```

Aqui está o que sabemos sobre o conjunto de dados de diamantes:

- Este conjunto de dados contém informações sobre 53.940 diamantes de corte redondo. Como nós sabemos? Cada linha de dados representa um diamante diferente e há 53.940 linhas de dados (consulte a página de ajuda ?diamonds)
- Existem 10 variáveis que medem várias informações sobre os diamantes. Observe que esses nomes de variáveis estão em letras minúsculas. Podemos ter uma visão rápida dos nomes das variáveis usando:

```
> names(diamonds)
 [1] "carat"  "cut"  "color"  "clarity"
 [5] "depth" "table" "price" "x"
 [9] "y"  "z"
```

- Existem 3 variáveis com uma estrutura factor order: cut, color, e clarity. Um fator ordenado organiza os valores categóricos em uma ordem de classificação de baixo para alto. Por exemplo, existem 5 categorias de cortes de diamante com “Fair” sendo o grau mais baixo de corte para o ideal sendo o grau mais alto (Fair, Good, Very Good, Premium, Ideal).
- Há 6 variáveis que são de estrutura numérica: carat, depth, table, x, y,z
- Existe 1 variável que possui uma estrutura inteira: price

Variável	Descrição	Valores
preço	preço em dólares americanos	$326-$18,823
quilate	peso do diamante	0.2-5.01
cortar	qualidade do corte	Fair, Good, Very Good, Premium, Ideal
cor	cor de diamante	J (worst) to D (best)
clareza	medição de quão claro é o diamante	I1 (pior), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (melhor)
x	comprimento em mm	0-10.74
Y	largura em mm	0-58.9
com	profundidade em mm	0-31.8
profundidade	porcentagem de profundidade total	43-79
tabela	largura do topo do diamante em relação ao ponto mais largo	43-95

```
> library(ggplot2)
```

```
> data("diamonds")
> niceDiamonds <- diamonds[diamonds$cut =="Premium" | diamonds$cut =="Ideal", ]
```

PARTE 5 – Cálculo - Modelagem

```
> View(diamonds)
```

diamonds

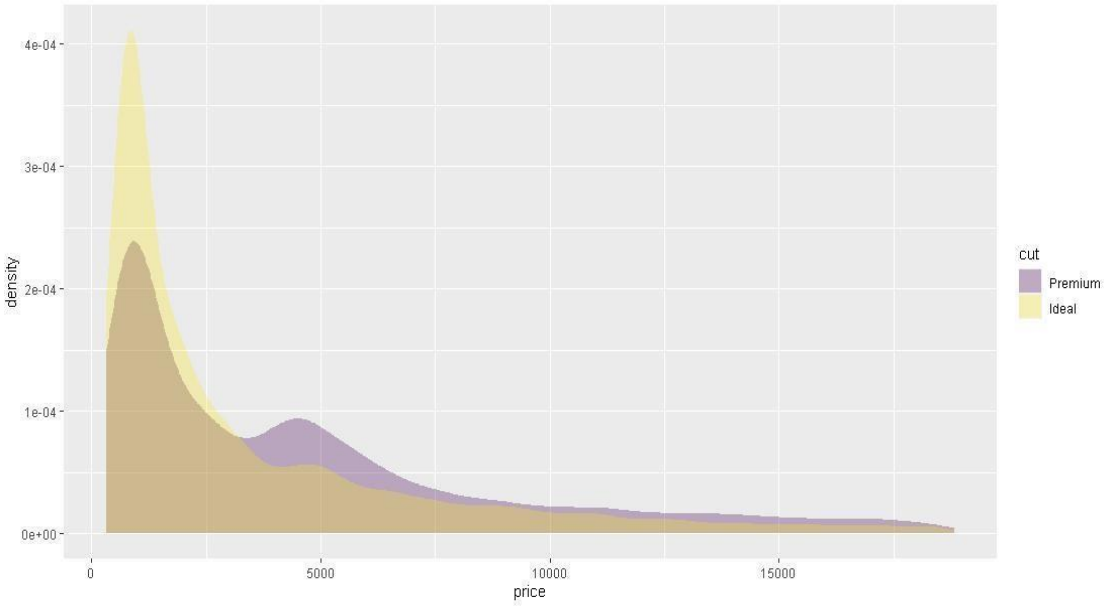
Filter

	carat	cut	color	clarity	depth	table	price	x	y	z
1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
6	0.24	Very Good	J	VVS2	62.8	57.0	336	3.94	3.96	2.48
7	0.24	Very Good	I	VVS1	62.3	57.0	336	3.95	3.98	2.47
8	0.26	Very Good	H	SI1	61.9	55.0	337	4.07	4.11	2.53

```
> summary((niceDiamonds))
```

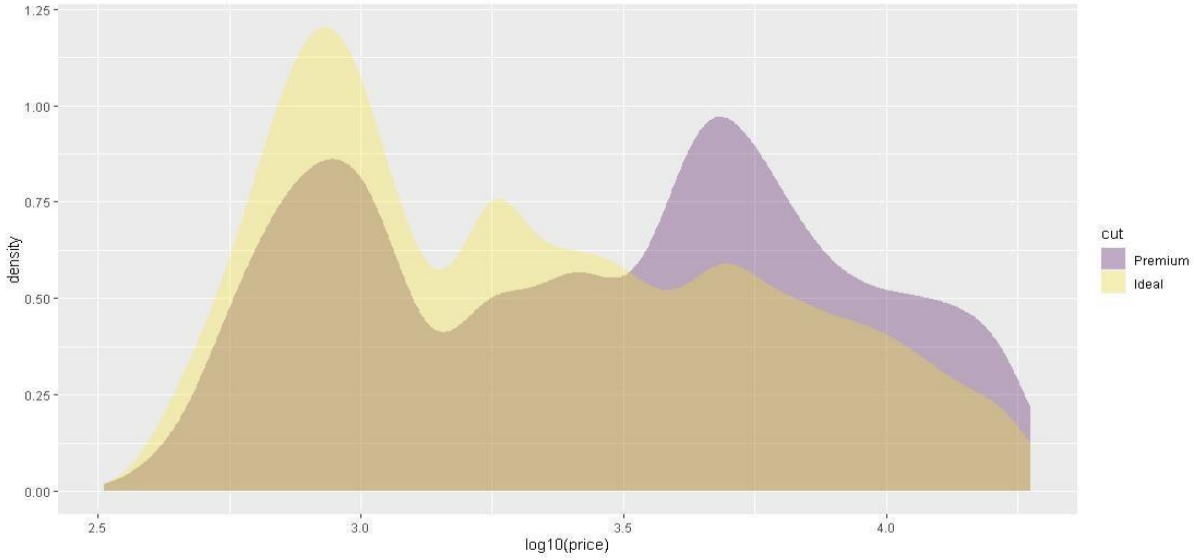
```
     carat               cut        color        clarity     
 Min.   :0.2000   Fair      :    0   D:4437   VS2     :8428  
 1st Qu.:0.3800   Good      :    0   E:6240   SI1     :7857  
 Median :0.6200   Very Good:    0   F:6157   VS1     :5578  
 Mean   :0.7766   Premium   :13791   G:7808   SI2     :5547  
 3rd Qu.:1.0500   Ideal     :21551   H:5475   VVS2    :3476  
 Max.   :4.0100                      I:3521   VVS1    :2663  
                                     J:1704   (other):1793  
     depth           table           price             x         
 Min.   :43.00   Min.   :43.00   Min.   :  326   Min.   : 0.000  
 1st Qu.:61.10   1st Qu.:56.00   1st Qu.:  929   1st Qu.: 4.650  
 Median :61.70   Median :57.00   Median : 2178   Median : 5.490  
 Mean   :61.54   Mean   :57.04   Mean   : 3897   Mean   : 5.689  
 3rd Qu.:62.20   3rd Qu.:58.00   3rd Qu.: 5364   3rd Qu.: 6.560  
 Max.   :66.70   Max.   :63.00   Max.   :18823   Max.   :10.140  

       y                z         
 Min.   : 0.000   Min.   :0.000  
 1st Qu.: 4.650   1st Qu.:2.850  
 Median : 5.490   Median :3.380  
 Mean   : 5.686   Mean   :3.497  
 3rd Qu.: 6.550   3rd Qu.:4.030  
 Max.   :58.900   Max.   :8.060  
```

```
> ggplot(niceDiamonds, aes(x=price, fill=cut))
+ geom_density(alpha = .3, color=NA)
```



O gráfico anterior tem uma cauda longa que pode impactar na análise.
Vamos usar uma escala logarítimica para ajustar a representação.

```
> ggplot(niceDiamonds, aes(x=log10(price), fill=cut))
+ geom_density(alpha = .3, color=NA)
```



O command em R para calcular a anova é “aov”.

```
> modelanova<-aov(price ~ cut, data=diamonds)

> modelanova
Call:
   aov(formula = price ~ cut, data = diamonds)


Terms:
                       cut    Residuals
Sum of Squares  11041745359 847431390159
Deg. of Freedom            4        53935

Residual standard error: 3963.847
Estimated effects may be unbalanced
```

Essa função nos retorna alguns elementos, que não serão discutidos nesta disciplina. Mas basicamente temos, as informações a seguir. A análise é mais complexa porque temos duas variáveis.

Em *Call* temos a fórmula que o R usou para executar a ANOVA

- Em *Terms* temos algumas estatísticas importantes, sendo a primeira coluna referente as análises dentro dos grupos e a segunda coluna referente a análises entre dos grupos
  - *Sum of Squares* : soma dos quadrados
  - *Df*: graus de liberdade

- *Residual standard error*: Erro padrão dos resíduos. Calculado a partir da raiz quadrada da divisão entre a soma dos quadrados dos resíduos e seus graus de liberdade.

Estudo de caso 3 – Escala Logarítimica dos dados.

Vamos simular a renda de pessoas pela técnica da distribuição normal relacionada com uma matriz de variância, cujo desvio padrão está em 0.7. Em probabilidade, uma variável aleatória X tem a distribuição log-normal quando o seu logaritmo tem a distribuição normal.

```
> income<-rlnorm(4000, meanlog = 4, sdlog = 0.7)
```

n → número de conjuntos de dados a serem simulados

meanlog → o vetor médio dos logs

varlog → a matriz de variância

n → 4000

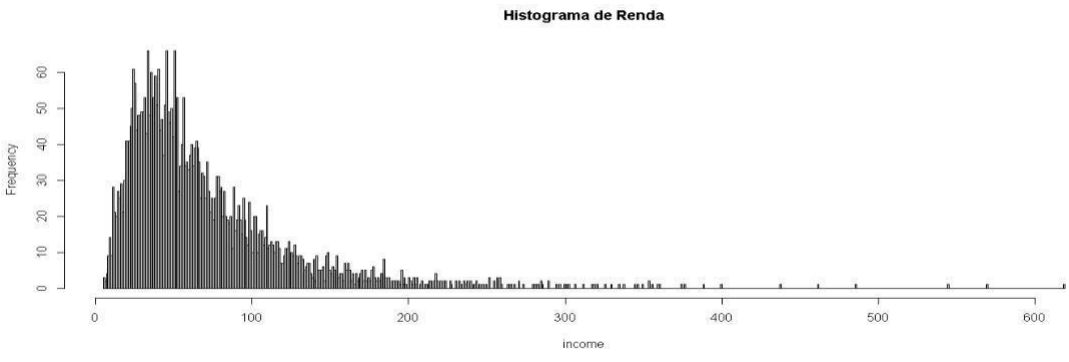
meanlog → 4

```
sdlog (desvio padrão log)  → 0.7 [1]  71.918097 102.595865 36.564606 33.533520 35.323583
 [6] 125.960270 24.423949 141.849652 19.753351 135.408282
[11]  36.113258 45.100219 30.861145 60.593085 44.268427
[16]  6.848002 91.514340 120.995083 28.616467 50.834962
[21]  25.971201 79.225080 123.641466 24.154900 20.505430
[26]  63.299273 87.806924 24.844690 90.754553 56.350803
[31]  34.700803 139.885656 184.773651 30.554776 25.410576
[36] 112.532598 210.331411 27.582399 52.305776 258.394293
[41] 108.476428  11.893030 90.206871 55.411893 174.496700
[46]  75.975992 79.282248 40.165094 13.399906 55.813584
[51]  10.124895 69.878078 120.650111 285.929505 48.461370
[56]  38.476928 47.153299 31.633495 39.041218 92.545523
[61]   9.228903 20.229490 47.336830 23.736524 149.341743
[66]  74.379133 48.769123 44.116733 239.388638 106.807811
```

```
> summary(income)
```

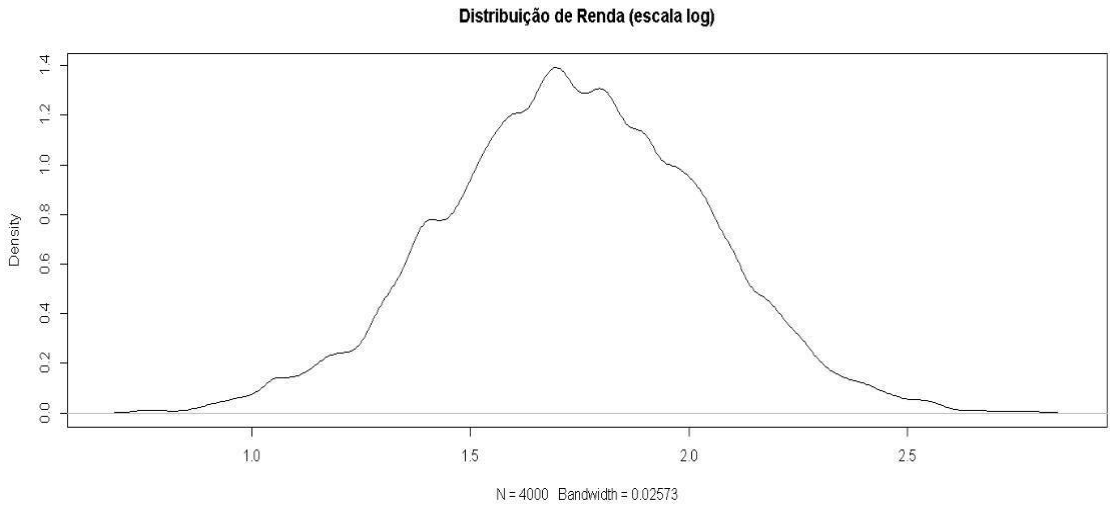
```
   Min. 1st Qu. Median     Mean 3rd Qu.     Max.
  4.494 34.549 55.361 71.023 88.869 612.521
```

```
> hist(income, breaks=500, xlab="income", main="Histograma de Renda")
```

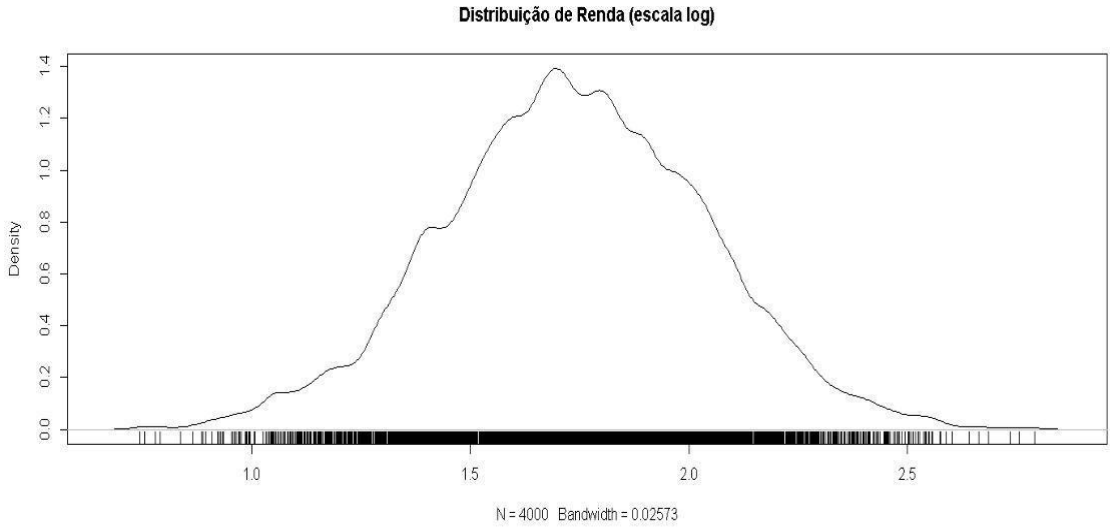


PARTE 5 – Cálculo - Modelagem

```
> plot(density(log10(income), adjust=0.5),
+     main="Distribuição de Renda (escala log)")
```



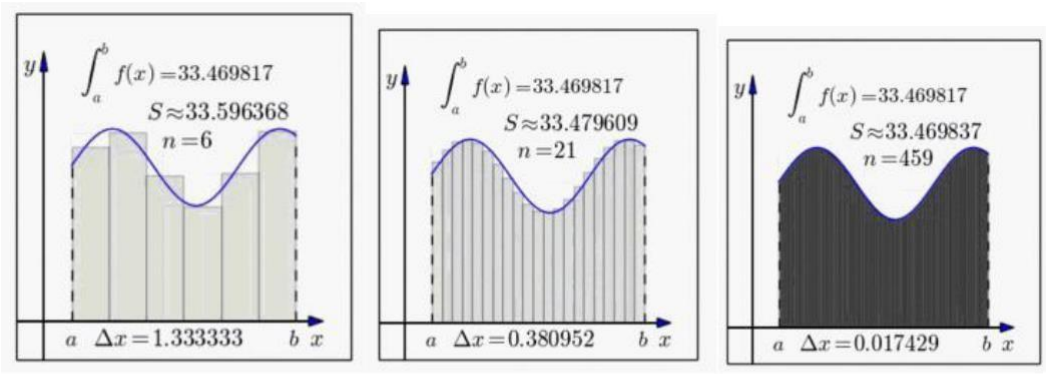

```
> rug(log10(income))
```



Estudo de Caso: Integração

No cálculo, a **integral** de uma função foi criada originalmente para determinar a área sob uma curva no plano cartesiano, como exemplo, usada para determinar a posição em todos os instantes de um objeto, se for conhecida a sua velocidade instantânea em todos os instantes.

PARTE 5 – Cálculo - Modelagem



$$\int_0^\infty \frac{1}{(x+1)\sqrt{x}} dx$$

```
> integrand <- function(x) {1/((x+1)*sqrt(x))}
> integrate(integrand, lower = 0, upper = Inf)
```
3.141593 with absolute error < 2.7e-05

A resposta numérica dessa integral é o valor de 3,141593 e um pequeno erro de $2.7*10^{-5}$

Outro exemplo:

Calcular o modelo integrativo da equação:

$$\int_{-1.96}^{1.96} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$$

```
> f <- function(x) {1/sqrt(2*pi)*exp(-x^2/2)}
> integrate(f, lower = -1.96, upper = 1.96)
```
0.9500042 with absolute error < 1e-11

Mais um exemplo: Integral tripla $\rightarrow$ usada para calcular volume

$$\int_0^{\frac{1}{2}} \int_0^{\frac{1}{2}} \int_0^{\frac{1}{2}} \frac{2}{3}(x_1 + x_2 + x_3) dx_1 dx_2 dx_3$$

```
> library(cubature)
```
Warning message:
package ‘cubature’ was built under R version 4.0.3
```
> f <- function(x) { 2/3 * (x[1] + x[2] + x[3]) } # "x" is vector
> adaptIntegrate(f, lowerLimit = c(0, 0, 0), upperLimit = c(0.5, 0.5,
0.5)) $integral
```
[1] 0.0625

$error
[1] 1.387779e-17

PARTE 5 – Cálculo - Modelagem

$functionEvaluations
[1] 33

$returnCode
[1] 0

Estudo de Caso: Análise Canônica – Agregação, Segmentação e Clusters. Correlação Canônica.

A análise de correlação canônica (CCA) é um método estatístico exploratório multidimensional que opera com o mesmo princípio da análise de componentes principais. O objetivo principal da abordagem de correlação canônica é a exploração de correlações de amostra entre dois conjuntos de variáveis quantitativas observadas nas mesmas unidades experimentais.

A seguir, consideraremos o mesmo conjunto de dados com o qual já trabalhamos antes. Os dados consistem em dois conjuntos de dados, onde cada conjunto de dados representa medições em algumas localidades fluviais específicas (64-65) na República Tcheca. O primeiro conjunto de dados contém medições de métricas biológicas para cada localidade (17 métricas e táxons diferentes) e o conjunto de dados secon contém medições de concentrações químicas e valores nas mesmas localidades (7 covariáveis registradas).

A ideia é relacionar de alguma forma os dois conjuntos de dados a fim de explicar quais bio métricas podem correlacionar quais concentrações químicas. Uma maneira de responder a isso é aplicar a abordagem de correlação canônica explicada abaixo.

```
> rm(list = ls())
>    bioData <- read.csv("http://msekce.karlin.mff.cuni.cz/~maciak/NMST539/bioData.csv",
header = T)
> bioData
```

1 BecvChor 2.57620 31.96916 0.37453
2 BecvOsek 2.43686 28.12037 0.27790
3 BecvTrou 2.10235 21.30691 0.31422
4 BelaBosk 1.53297 37.12914 0.48181
5 BilyPoto 1.79499 27.21495 0.37100
6 BlatTova 2.90209 16.04854 0.30351
7 BrezJaro 2.70210 18.20885 0.27896
8 BrodVice 2.14583 19.97223 0.25092
9 BrtnStri 1.86586 27.28535 0.36365
10 BystByst 1.69639 36.63515 0.41381
11 DesnSudk 1.78585 35.27455 0.47687
12 DrevLuto 2.69870 22.97771 0.32069
13 DrevOtro 2.77197 12.81706 0.22601

```
>  chemData <- read.csv("http://msekce.karlin.mff.cuni.cz/~maciak/NMST539/chemData.csv",
header = T)
```

PARTE 5 – Cálculo - Modelagem

```
> chemData

  Kod_Canoco Tepl_max X.O2 BSK5  Kond N.NH4
1  BecvChor      20.9 107.0 1.3 33.70 0.090
2  BecvOsek      21.8 105.0 1.2 37.00 0.060
3  BecvTrou      22.1 106.0 1.4 43.90 0.070
4  BelaBosk      16.4 104.0 1.1 22.10 0.020
5  BilyPoto      18.9 104.0 1.9 30.40 0.100
6  BrezJaro       18.4 84.0 2.5 83.80 0.300
```

```
> head(bioData)
  Kod_Canoco SaprInd   Lital    RETI  EPTAbu
1   BecvChor 2.57620 31.96916 0.37453 26.90136
2   BecvOsek 2.43686 28.12037 0.27790 23.32888
3   BecvTrou 2.10235 21.30691 0.31422 36.49575
4   BelaBosk 1.53297 37.12914 0.48181 38.49802
5  BilyPoto 1.79499 27.21495 0.37100 23.26918
6  BlatTova 2.90209 16.04854 0.30351 8.64187
     Marg Metaritr JepAbu Epiritral
1 7.10911 16.25647 14.89330  7.68869
2 5.80443 14.14831 11.06343  6.96659
3 6.14622 11.21903 30.23011  5.92213
```

```
> head(chemData)
  Kod_Canoco Tepl_max X.O2 BSK5 Kond N.NH4 N.NO3 Pcelk
1  BecvChor      20.9 107 1.3 33.7 0.09 1.55 0.077
2  BecvOsek      21.8 105 1.2 37.0 0.06 2.32 0.063
3  BecvTrou      22.1 106 1.4 43.9 0.07 2.49 0.080
4  BelaBosk      16.4 104 1.1 22.1 0.02 2.05 0.029
5  BilyPoto      18.9 104 1.9 30.4 0.10 4.48 0.131
6  BrezJaro       18.4  84 2.5 83.8 0.30 3.63 0.237
```

Devem paarecer mensagens de worning. Há uma localidade que está faltando o conjunto de dados de medições químicas. Identificamos esta localidade e não a consideramos para uma análise posterior. Ambos os conjuntos de dados são alinhados em relação aos nomes das localidades - a primeira coluna em cada conjunto de dados.

```
> ind <- match(chemData[,1], bioData[,1])
> data <- data.frame(bioData[ind, ], chemData[, 2:8])
> X <- data[,2:9]
> Y <- data[,19:25]
```

Uma boa ferramenta gráfica para visualizar os dois conjuntos de dados com relação à estrutura de correlação geral (dentro e entre) está disponível no pacote R 'CCA' (Análise de Correlação Canônica). Os pacotes precisam ser instalados primeiro **(install.packages("CCA"))** e depois pode-se usar as funções **matcor( )**e **img.matcor( )**exibir graficamente a matriz de correção.

$$cor(\mathcal{X}, \mathcal{E}) = \begin{pmatrix} \Sigma_{XX} \Sigma_{XY} \\ \Sigma_{YX} \Sigma_{YY} \end{pmatrix}$$

PARTE 5 – Cálculo - Modelagem

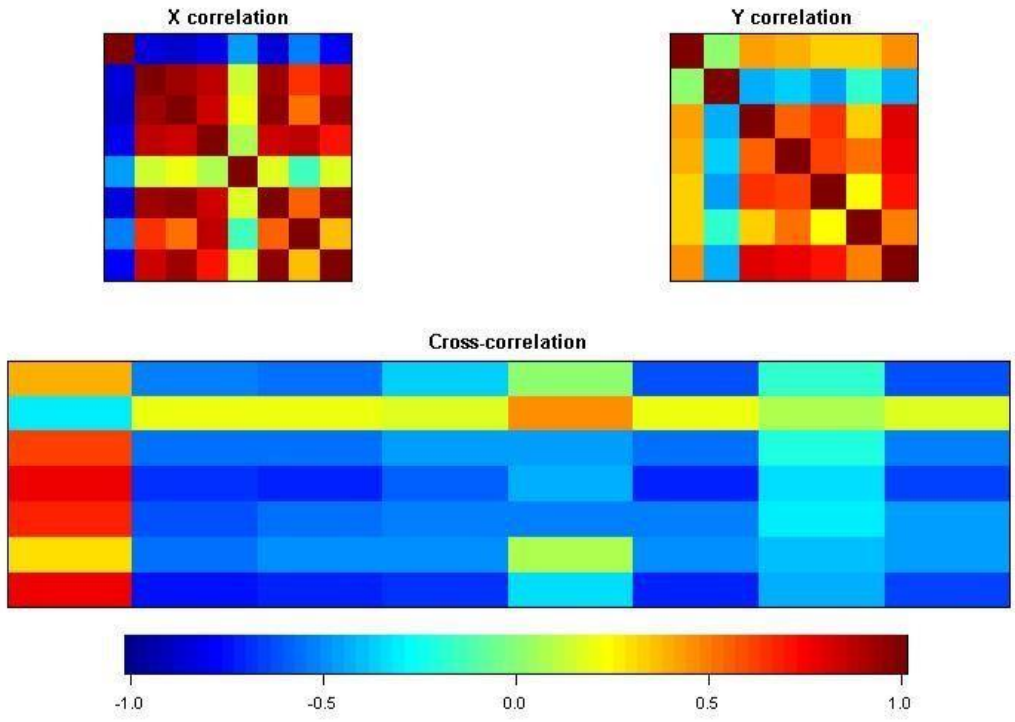
Vá em Tools para installar o Package CCA.

```
> install.packages("CCA")
```

package ‘dotCall64’ successfully unpacked and MD5 sums checked
package ‘spam’ successfully unpacked and MD5 sums checked
package ‘maps’ successfully unpacked and MD5 sums checked
package ‘fda’ successfully unpacked and MD5 sums checked
package ‘fields’ successfully unpacked and MD5 sums checked
package ‘CCA’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
C:\Users\Marise Miranda\AppData\Local\Temp\RtmpM7bJM5\downloaded_packages

```
> library("CCA")
> correl <- matcor(X, Y )
> img.matcor(correl, type = 2)
```



Em seguida, podemos usar diretamente a **cancor()**função para obter **correlações canônicas** para os conjuntos de dados X e Y. Outra opção é usar a função **cc( )**(do pacote R 'CCA'). Observe que os resultados são equivalentes, embora não idênticos.

```
> cc1 <- cancor(X, Y)
> cc2 <- cc(X, Y)
> cc1$cor
```

[1] 0.87496109 0.70496586 0.64579693 0.48750752 0.24205299 0.11563830
0.04481277 > cc2$cor

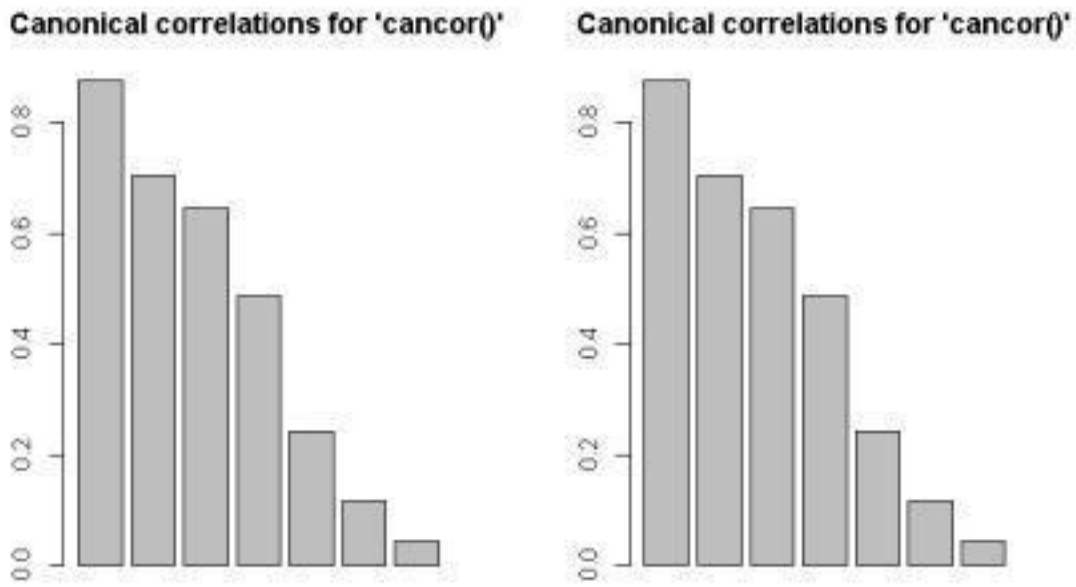
PARTE 5 – Cálculo - Modelagem

[1] 0.87496109 0.70496586 0.64579693 0.48750752 0.24205299 0.11563830

0.04481277 > par(mfrow = c(1,2))

> barplot(cc1$cor, main = "Canonical correlations for 'cancor()'", col = "gray")

>barplot(cc1$cor, main = "Canonical correlations for 'cancor()'", col = "gray")



No entanto, as combinações lineares reais estimadas das covariáveis originais em ambos os conjuntos de dados XX e YYsão diferentes. Compare o seguinte:

> cc1$xcoef

	[,1]	[,2]	[,3]	[,4]	[,5]
SaprInd	0.162796345	0.089714234	-0.161737589	-0.491927540	-0.186875530
Lital	-0.003800701	0.001797172	-0.030965801	-0.007214761	0.017517637
RETI	0.324209881	-0.055178843	0.693084411	-1.283570658	-2.507634813
EPTAbu	-0.002149837	-0.005077822	-0.009027554	0.008345945	0.005867073
Marg	-0.011912887	-0.037901448	-0.009666364	-0.065884327	-0.025639954
Metaritr	-0.012221662	0.001551030	0.043003790	0.012178243	-0.003013795
JepAbu	0.005168862	0.006308244	0.004048909	-0.009547646	-0.019794148
Epiritral	0.010432924	0.023391927	-0.013976612	-0.021306313	0.018722542

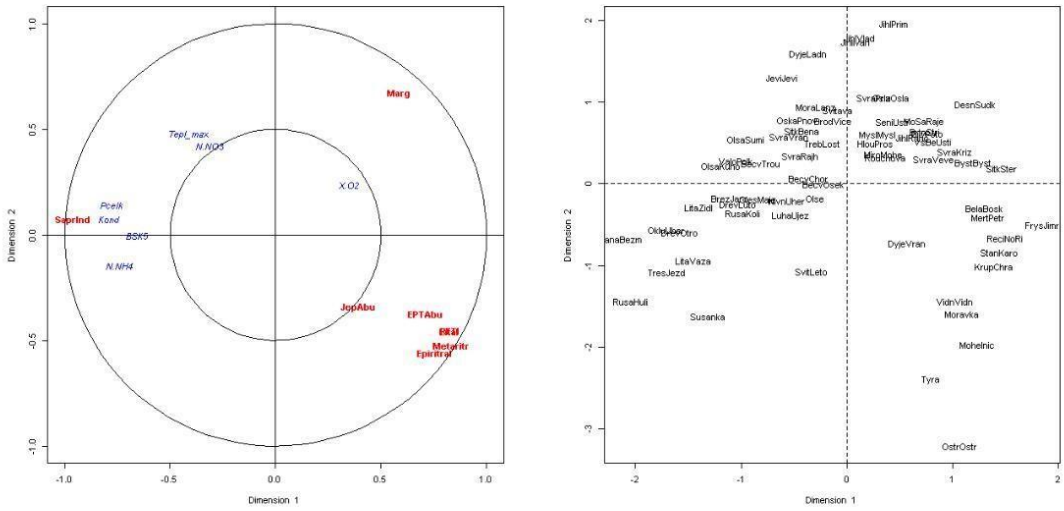
	[,6]	[,7]	[,8]
SaprInd	-0.098864028	0.06897063	-0.355850803
Lital	0.004465308	-0.01327305	0.008943729
RETI	3.047758363	1.14371548	0.285767761
EPTAbu	-0.005401486	0.01277878	-0.026098676
Marg	-0.029969261	0.01102501	-0.002459819
Metaritr	-0.027620185	-0.04963561	-0.056954743
JepAbu	-0.005346595	-0.00253416	0.030139906
Epiritral	-0.022760565	0.04740075	0.064087475

PARTE 5 – Cálculo - Modelagem

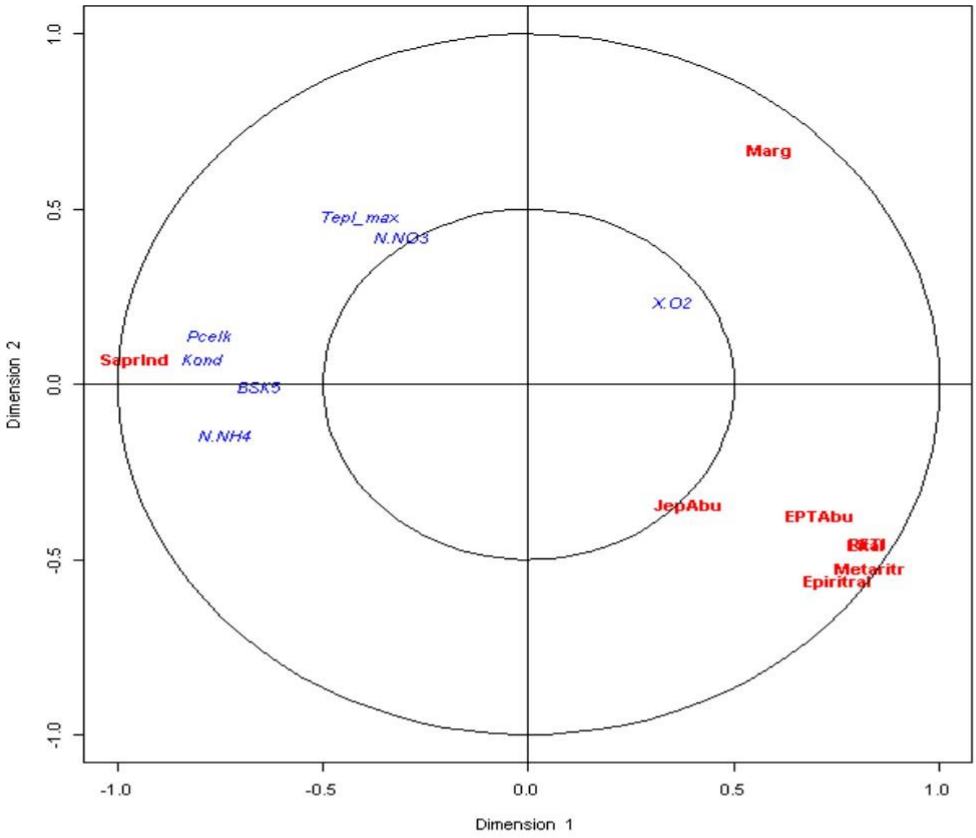
```
> cc2$xcoef
              [,1]        [,2]       [,3]        [,4]        [,5]
SaprInd     -1.29215593 -0.71208465 -1.28375231 -3.90455380 -1.48327854
Lital        0.03016713 -0.01426461 -0.24578343 -0.05726539  0.13904193
RETI        -2.57333615  0.43796849  5.50118696 -10.18802625 -19.90373428
EPTAbu       0.01706380  0.04030396 -0.07165399  0.06624389  0.04656845
Marg         0.09455561  0.30083342 -0.07672439 -0.52294063 -0.20351082
Metaritr     0.09700644 -0.01231092  0.34133200  0.09666181 -0.02392126
JepAbu      -0.04102657 -0.05007014  0.03213722 -0.07578209 -0.15711118
Epiritral   -0.08280877 -0.18566767 -0.11093592 -0.16911362  0.14860557
              [,6]        [,7]
SaprInd    -0.78470889 0.54743742
Lital       0.03544229 -0.10535158
RETI       24.19083206 9.07796019
EPTAbu     -0.04287297 0.10142843
Marg       -0.23787363 0.08750829
Metaritr   -0.21922842 -0.39397046
JepAbu     -0.04243728 -0.02011427
Epiritral  -0.18065638 0.37623175
```

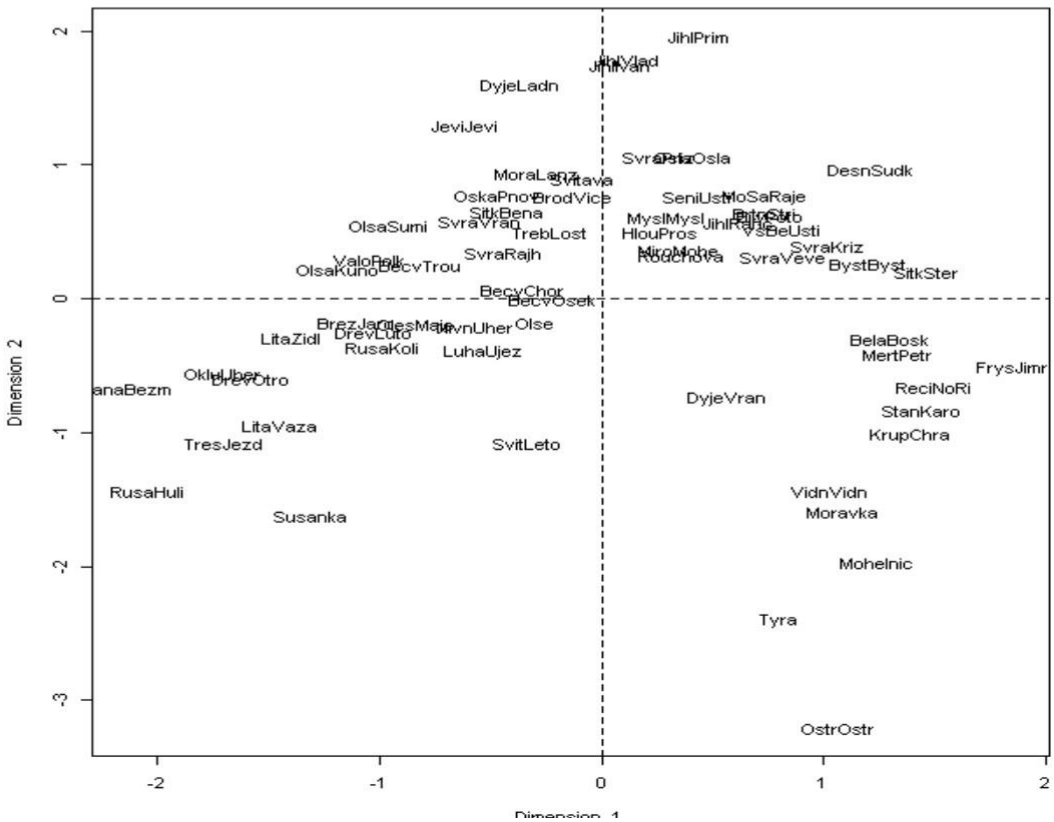
Usando o pacote 'CCA', pode-se também tirar vantagem de boas ferramentas gráficas que estão disponíveis para a análise de correlação canônica. A ideia é exibir correlações maximizadas entre variáveis transformadas do conjunto de dadosX e o conjunto de dados Y.

```
>plt.cc(cc2, var.label = TRUE, ind.names = data[,1])
```



PARTE 5 – Cálculo - Modelagem





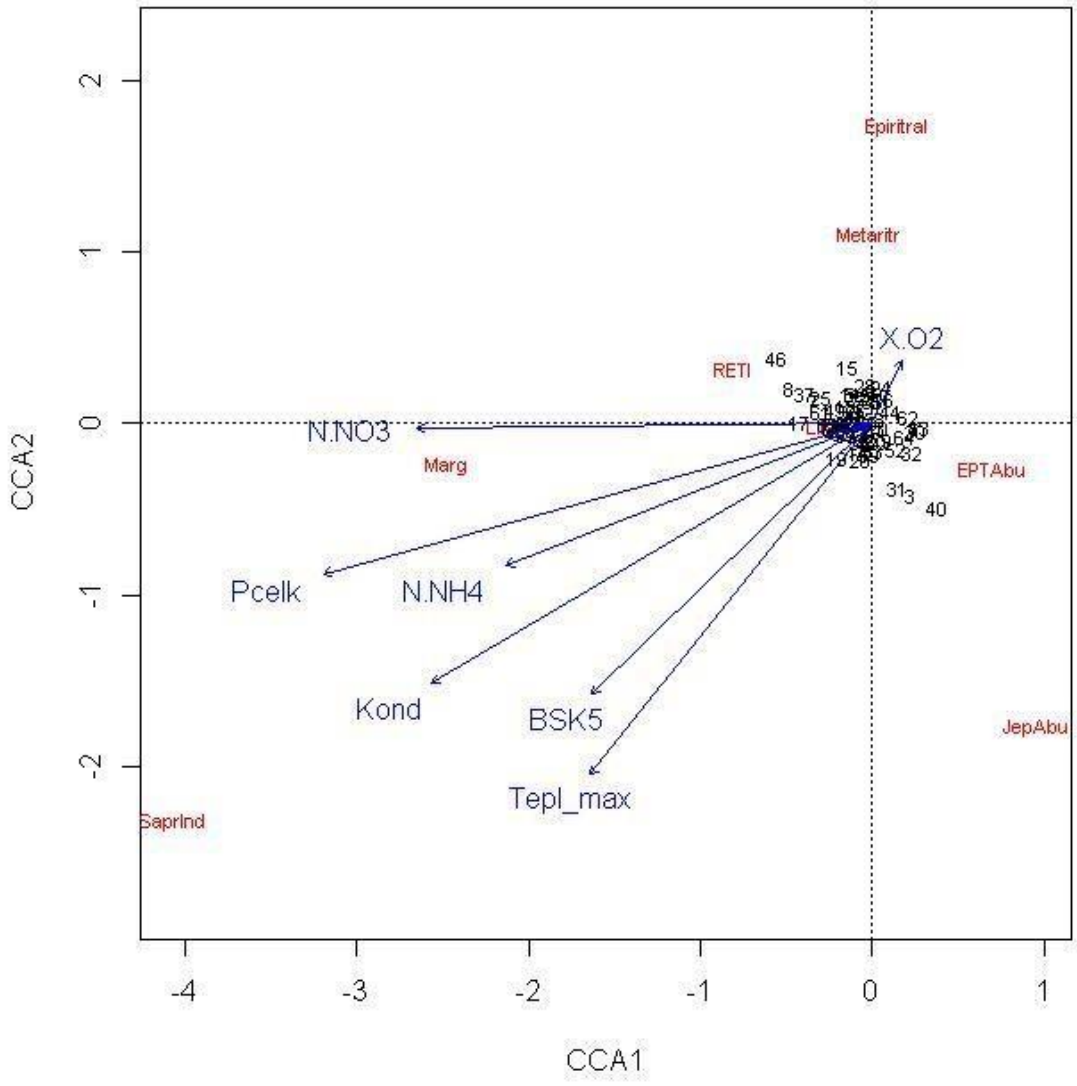
PARTE 5 – Cálculo - Modelagem

Uma extensão útil para a abordagem de correlação canônica clássica é a versão regularizada - chamada de análise de correlação canônica regularizada. É médio para cenários onde o número de parâmetrosp∈N é maior que o tamanho da amostra n ∈ n (geralmente p≫n) Há uma função R **rcc( )**disponível no pacote R 'CCA'.

Outra opção de como executar a correlação canônica em R é instalar um pacote R adicional - **pacote 'vegan' (usar install.packages("vegan")para instalação)** e aplicar a função cca(). Use a sessão de ajuda em R para obter mais detalhes sobre a própria função ou o pacote correspondente, respectivamente.

Em tools install vegan

```
> install.packages("vegan")
> library(vegan)
> cc3 <- cca(X, Y)
> plot(cc3, scaling = 1)
```

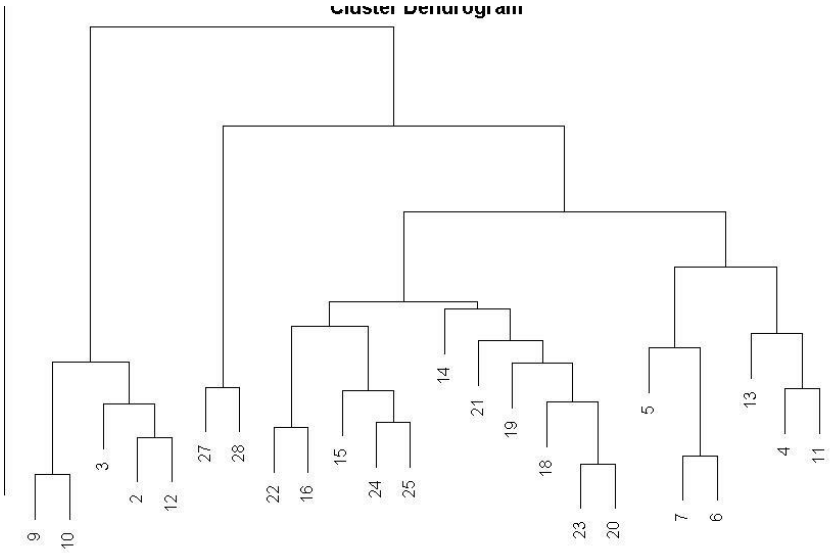


PARTE 5 – Cálculo - Modelagem

Estudo de Caso: Análise de agrupamentos (cluster analysis) por Distância Euclidiana

```
> data(varespec)
Warning message:
In data(varespec) : data set ‘varespec’ not found
> library(vegan)
```

```
1: package ‘vegan’ was built under R version 4.0.3
2: package ‘permute’ was built under R version 4.0.3
> data(varespec)
> View(data("varespec"))
> especie.dist<-vegdist(varespec,"euclidean")
> clusterUPGMA<-hclust(especie.dist,method="average")
> clusterUPGMA

Call:
hclust(d = especie.dist, method = "average")

Cluster method : average
Distance       : euclidean
Number of objects: 24

> plot(clusterUPGMA)
```



Estudo de Caso: Janela de Hamming

A distância de Hamming entre:

"**elab**ore" e "**melh**ore" é 4.
2**17**3**8**96 e 2**23**3**7**96 é 3.
1**1**011 e 1**0**011 é 1.

Neste terceiro exemplo, se a cadeia de bits A(11011) fosse emitida e chegasse ao receptor como A'(10011), poderia ser usada a operação XOR da seguinte maneira:

11011
XOR 10011
01000

```
> d <- hamming.distance(x)
> NN <- apply(d[test.set, design.set], 1, order)
> k <- 5
> pred <- apply(NN[, 1:k, drop=FALSE], 1, function(nn){
+ tab <- table(y[design.set][nn])
+ as.integer(names(tab)[which.max(tab)]) +
)
> table(pred, y[test.set])
```

```
pred 1 2
   120 5
   2 619
```

Este exemplo não plota é só pra vc ver como funciona.