Estatística computacional - Estatística - ce223 (www.leg.ufpr.br/ce223)
Prof. Walmes Marques Zeviani - Departamento de Estatística - UFPR
Terceira avaliação - Otimização, expressão regular e operações com datas (21/06/2011)
Acadêmico: Gabarito da Prova Tirei Dez Matrícula: GRR2011xxxx

Instruções:
- Você pode consultar a internet e o cartão de referência do R. Você não pode consultar o caderno nem trocar informações com colegas;
- Todas as questões devem ser respondidas na folha de prova, com os códigos empregados e os resultados obtidos;
- É altamente recomendável que você faça a prova no computador do laboratório (linux). Isso porque a importação de dados pode gerar problemas devido a codificação de caracteres. Caso faça a prova no seu computador pessoal, a responsabilidade de fazer a importação total e correta dos conjuntos de dados é sua.

1. (2.5) Considere os dados de liberação de potássio da `aula19.R` (`klib`, `tempo`) e o modelo de regressão não linear

$$f(t) = A \cdot (1 - \mathrm{e}^{-0{,}693 \cdot t/V}), \qquad t > 0, \quad A > 0, \quad V > 0, \tag{1}$$

em que A é o parâmetro assíntota e V é o parâmetro tempo de meia vida. Obtenha:
a) o gráfico da função $f(t)$ no domínio de t presente nos dados considerando valores arbitrários de A e V;
b) o gráfico de dispersão dos dados com `klib ~ tempo`;
c) use a `optim()` para estimar os parâmetros A e V pelo método de mínimização da soma de quadrados dos desvios. Adapte a função `fun.sqr()`;
d) faça o gráfico de dispersão e adicione a função $f(t)$ usando os valores estimados de A e V.

```
> #-------------------------------------------------------------------------------------------
> # dados presentes na aula19.R e amplitude de variação do tempo de 0 à 300 dias
>
> klib <- c(51.03, 57.76, 26.60, 60.65, 87.07, 64.67, 91.28, 105.22, 72.74, 81.88, 97.62,
+           90.14, 89.88, 113.22, 90.91, 115.39, 112.63, 87.51, 104.69, 120.58, 114.32,
+           130.07, 117.65, 111.69, 128.54, 126.88, 127.00, 134.17, 149.66, 118.25, 132.67,
+           154.48, 129.11, 151.83, 147.66, 127.30)
> tempo <- rep(c(15, 30, 45, 60, 75, 90, 120, 150, 180, 210, 240, 270), each=3)
> range(tempo)

[1]  15 270

> #-------------------------------------------------------------------------------------------
> # a) gráfico da função f(t) com valores abrtitrários de A (140) e V (50)
>
> A <- 140; V <- 50
> curve(A*(1-exp(-0.693*x/V)), 0, 300)
> #-------------------------------------------------------------------------------------------
> # b) gráfico de dispersão dos dados
>
> plot(klib~tempo)
> curve(A*(1-exp(-0.693*x/V)), add=TRUE)
> #-------------------------------------------------------------------------------------------
> # c) usando optim() para otimizar e adaptar a função fun.sqr()
>
> fun.sqr <- function(parms, k, t){
+   ## params 1 e 2 correspondem a A e V
+   ## k são valores observados de potássio, t é o tempo
+   ## função retorna a soma de quadrados dos resíduos
+   ft <- parms[1]*(1-exp(-0.693*t/parms[2])) # modelo f(t) = A*(1-exp(-0.693*t/V))
+   sqr <- sum((k-ft)^2)
+   return(sqr)
+ }
> op <- optim(c(140, 50), fun.sqr, k=klib, t=tempo)
> op$par # estimativas de mínimos quadrados dos parâmetros A e V

[1] 132.6958  33.1860

> #-------------------------------------------------------------------------------------------
> # d) gráfico com os valores estimados de A e V
>
> A <- op$par[1]; V <- op$par[2]
> plot(klib~tempo, xlab="Tempo (dias)", ylab="Potássio liberado (mg/dm³)")
> curve(A*(1-exp(-log(2)*x/V)), add=TRUE, col=2)
> #-------------------------------------------------------------------------------------------
```

2. (3.0) Importe os dados de endereço (`email`) e senha (`password`) de usuários em um sistema de cadastro. Usando expressões regulares e a função `grep()` obtenha:

```
senha <- read.table("http://www.leg.ufpr.br/~walmes/ensino/ce223-2011-01/senha.txt",
                    header=TRUE, sep="\t", stringsAsFactors=FALSE, quote="")
```

a) um gráfico de barras indicando o número de e-mails do `@yahoo`, `@hotmail`, `@gmail`, `@msn` e `@live`. Considere os restantes na categoria `outros`;
b) usando a função `nchar()`, que retorna o número de caracteres de cada elemento de um vetor, faça um gráfico de barras da frequência absoluta do tamanho das senhas. Use `table()`;
c) faça um gráfico de barras indicando a frequência de ocorrência de cada uma das 26 letras mínusculas do alfabeto. Use o vetor `letters`, a funções `sapply()`, `grep()` e `length()`;
d) faça um gráfico de barras indicando a frequência de ocorrência de cada um dos números de 0 à 9;
e) qual o percentual de senhas que começam com letra e terminam com número?
f) quantas senhas iniciam com os digitos do número π, 31415?

```
> #-------------------------------------------------------------------------------------------
> # importando dados pelo site ou localmente
>
> senha <- read.table("http://www.leg.ufpr.br/~walmes/ensino/ce223-2011-01/senha.txt",
+                     header=TRUE, sep="\t", stringsAsFactors=FALSE, quote="")
> str(senha)

'data.frame':        35571 obs. of  2 variables:
 $ email   : chr  "1@grandhaven.us" "10coleen@bellsouth.net" "121cabana@animail.net" "123@11.com" ...
 $ password: chr  "sweeps" "roxy123" "a0adam" "123" ...

> #-------------------------------------------------------------------------------------------
> # a) usando expressão regular para pegar o número de e-mail de cada serviço de e-mail
> # solução 1 (lusitana)
>
> y <- length(grep("@yahoo.", senha$email))
> g <- length(grep("@gmail.", senha$email))
> h <- length(grep("@hotmail.", senha$email))
> l <- length(grep("@live.", senha$email))
> m <- length(grep("@msn.", senha$email))
> o <- nrow(senha)-y-g-h-l-m
> c(y,g,h,l,m,o)

[1]  9392  5184  4740   306  1116 14833

> barplot(c(yahoo=y,gmail=g,hotmail=h,live=l,msn=m,outros=o),
+         xlab="Serviço de e-mail", ylab="Número de usuários")
> #-------------------------------------------------------------------------------------------
> # solução 2 (sulmatogrossense)
>
> se <- c(yahoo="@yahoo.", gmail="@gmail.", hotmail="@hotmail.", live="@live.", msn="@msn.")
> nu <- sapply(se, function(er) length(grep(er, senha$email))) # ou
> nu <- sapply(sapply(se, grep, x=senha$email), length)
> nu <- c(nu, outros=nrow(senha)-sum(nu))
> nu

  yahoo   gmail hotmail    live     msn  outros
   9392    5184    4740     306    1116   14833

> barplot(sort(nu, dec=TRUE), xlab="Serviço de e-mail", ylab="Número de usuários")
> nu <- sort(nu)
> lab <- paste(names(nu), " (", prettyNum(100*prop.table(nu), digits=3), ")", sep="")
> pie(nu, labels=lab)
> mtext(text="Propoção de usuários de cada serviço de e-mail (%)", font=2)
> #-------------------------------------------------------------------------------------------
> # b) gráfico de barras do tamanho das senhas usando nchar() e table()
>
> ts <- nchar(senha$password)
> ts <- table(ts)
> barplot(ts, xlab="Número de caracteres na senha", ylab="Número de senhas")
> #-------------------------------------------------------------------------------------------
> # c) frequência de cada uma das letras minúsculas do alfabeto
>
> fminu <- sapply(letters, function(l) length(grep(l, senha$password)))
> barplot(fminu)                        # barras na ordem das letras
> barplot(sort(fminu, decreasing=TRUE), # barras na ordem das frequencias
+         xlab="Letra minúscula do alfabeto", ylab="Número de senhas")
> #-------------------------------------------------------------------------------------------
> # d) frequência de cada um dos digitos de 0 à 9
>
> fdigi <- sapply(as.character(0:9), function(l) length(grep(l, senha$password)))
> barplot(fdigi)
> barplot(sort(fdigi, decreasing=TRUE),
+         xlab="Digitos de 0 à 9", ylab="Número de senhas")
> #-------------------------------------------------------------------------------------------
> # e) percentual de senhas que começam (^) com letras ([A-Za-Z]) e terminam com número
```

```
> # ([0-9]) com qualquer quantidade de qualquer caractere no meio (.*)
>
> Az09 <- grep("^[A-Za-z].*[0-9]$", senha$password, value=TRUE)
> str(Az09)
 chr [1:14391] "roxy123" "microsoft123" "yosemite71" "teddy15" ...
> 100*length(Az09)/length(senha$password) # em porcentagem
[1] 40.45711
> #-------------------------------------------------------------------------------------------
> # f) senhas que iniciam (^) com pi
>
> grep("^31415", senha$password, value=TRUE)
[1] "314159" "314159" "314159" "314159"
>
> #-------------------------------------------------------------------------------------------
> NA
> NA
```

---

3. (1.5) Considere as seis expressões regulares a seguir

```
"^[a-z]+$"   "^[A-Z]+$"   "^[0-9]+$"   "^\\d+$"   "^\\W+$"   "^[^aeiou]+$"
```

a) o que cada uma delas está selecionando?
b) obtenha o número de senhas que combine com cada uma dessas expressões regulares.

```
> #-------------------------------------------------------------------------------------------
> # a) o significado das expressões:
> # "^[a-z]+$": começam (^) e terminam ($) e contém só, no mínimo uma (+), minúscula [a-z]
> # "^[A-Z]+$": começam (^) e terminam ($) e contém só, no mínimo uma (+), maiúscula [A-Z]
> # "^[0-9]+$": começam (^) e terminam ($) e contém só, no mínimo um (+), um número [0-9]
> # "^\\d+$": começam (^) e terminam ($) e contém só, no mínimo um (+), um número \\d
> # "^\\W+$": começam (^) e terminam ($) e contém só, no mínimo um (+), não alfanumérico \\W
> # "^[^aeiou]+$": começam (^) e terminam ($) sem conter vogal
>
> #-------------------------------------------------------------------------------------------
> # b) número de senhas que combinam com essas expressões regulares
>
> er <- c(minúsuculas="^[a-z]+$", maiúsculas="^[A-Z]+$", números="^[0-9]+$", números="^\\d+$",
+         nãoalfanuméricos="^\\W+$", nãovogais="^[^aeiou]+$")
> sapply(er, function(r) length(grep(r, senha$password)))
     minúsuculas       maiúsculas          números          números
           15541              344             1660             1660
nãoalfanuméricos        nãovogais
               2             4278
> #-------------------------------------------------------------------------------------------
```

---

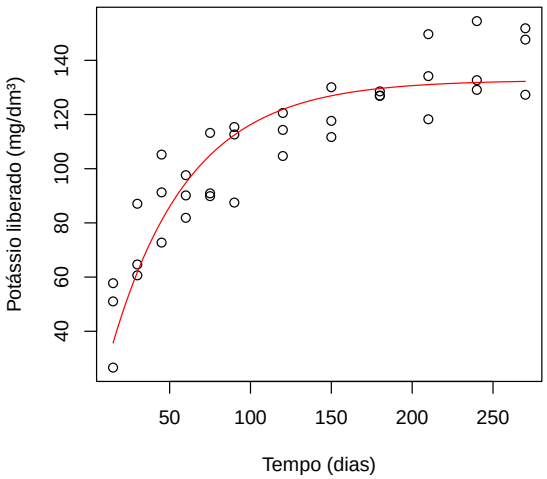
4. (3.0) Considere a série metereológica de temperatura (`temp`) e umidade relativa (`urel`) registrada em intervalos de hora durante um ano.

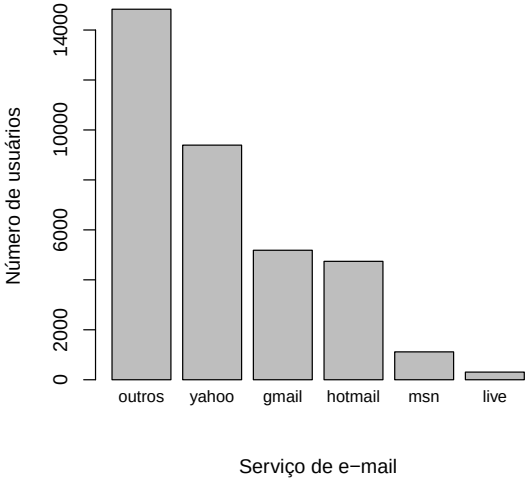
```
met <- read.table("http://www.leg.ufpr.br/~walmes/ensino/ce223-2011-01/metereo.txt",
                  header=TRUE, sep="\t", stringsAsFactors=FALSE)
```

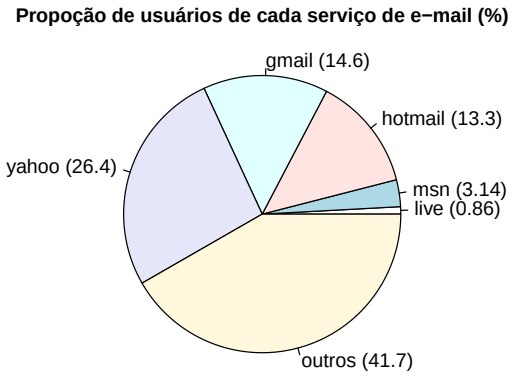
a) qual o formato usado para representar a data?
b) crie uma nova coluna para um objeto de classe de data. Use `as.POSIXlt()`;
c) em quais datas observou os valores máximos e mínimos de temperatura e umidade?
d) qual a diferença em horas entre a data da temperatura mínima e máxima?
e) faça o gráfico da série de umidade;
f) faça o gráfico da série de temperatura média diária. Use `tapply()` e `as.Date()`.

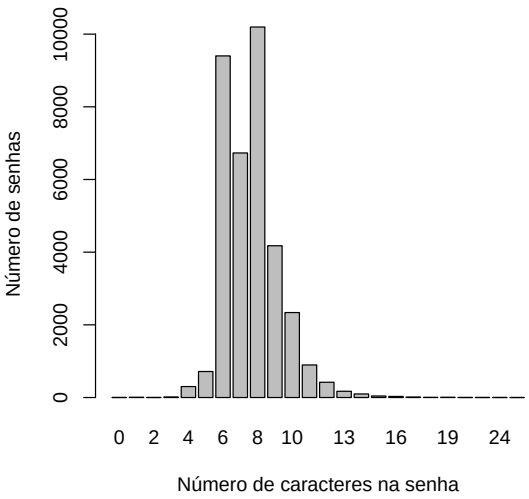
```
> #-------------------------------------------------------------------------------------------
> # importando os dados da web ou localmente
>
> met <- read.table("http://www.leg.ufpr.br/~walmes/ensino/ce223-2011-01/metereo.txt",
+                   header=TRUE, sep="\t", stringsAsFactors=FALSE)
> str(met)
'data.frame':        8773 obs. of  3 variables:
 $ data: chr  "20/06/2010 01:00" "20/06/2010 02:00" "20/06/2010 03:00" "20/06/2010 04:00" ...
 $ temp: num  22.4 22.1 21.3 20.9 19.5 18.9 18.3 20.8 24.6 27.4 ...
 $ urel: int  63 62 64 66 73 75 79 70 53 41 ...
> #-------------------------------------------------------------------------------------------
> # a) formato de data é dd/mm/aaaa H:M, em formato R é %d/%m/%Y %H:%M
>
> met$data[1:4]
```

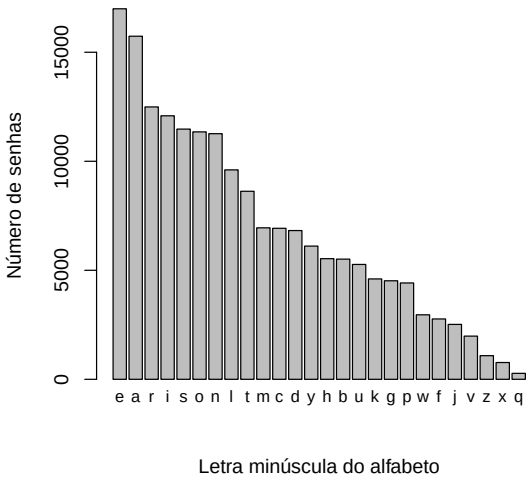
```
[1] "20/06/2010 01:00" "20/06/2010 02:00" "20/06/2010 03:00" "20/06/2010 04:00"
> #-----------------------------------------------------------------------------------------
> # b) nova coluna com classe do tipo data usando as.POSIXlt()
>
> met$d <- as.POSIXlt(met$data, format="%d/%m/%Y %H:%M")
> str(met)
'data.frame':        8773 obs. of  4 variables:
 $ data: chr  "20/06/2010 01:00" "20/06/2010 02:00" "20/06/2010 03:00" "20/06/2010 04:00" ...
 $ temp: num  22.4 22.1 21.3 20.9 19.5 18.9 18.3 20.8 24.6 27.4 ...
 $ urel: int  63 62 64 66 73 75 79 70 53 41 ...
 $ d   : POSIXlt, format: "2010-06-20 01:00:00" "2010-06-20 02:00:00" ...
> #-----------------------------------------------------------------------------------------
> # c) as datas dos valores mínimos e máximos de umidade e temperatura
>
> urange <- met$d[c(which.min(met$urel), which.max(met$urel))]; urange # para umidade
[1] "2010-09-13 14:00:00" "2010-12-23 05:00:00"
> trange <- met$d[c(which.min(met$temp), which.max(met$temp))]; trange # para temperatura
[1] "2010-08-06 06:00:00" "2010-09-21 14:00:00"
> #-----------------------------------------------------------------------------------------
> # d) diferença em horas entre das datas que de mínima e máxima temperatura
>
> difftime(trange[2], trange[1], units="hours")
Time difference of 1112 hours
> difftime(trange[2], trange[1], units="days")
Time difference of 46.33333 days
> difftime(trange[2], trange[1], units="weeks")
Time difference of 6.619048 weeks
> #-----------------------------------------------------------------------------------------
> # e) gráfico da série de umidade
>
> with(met,
+      plot(d, urel, type="l",
+           ylab="Umidade relativa (%)",
+           xlab="Tempo (horas, 2010-2011)")) # de umidade
> #-----------------------------------------------------------------------------------------
> # f) gráfico da temperatura média diária (extra: mínima e máxima)
>
> dia <- as.Date(format(met$d, format="%d/%m/%Y"), format="%d/%m/%Y") # ou
> length(unique(dia)) # número de dias únicos, cada dia tem 24 medidas
[1] 366
> tmed <- tapply(met$temp, dia, mean) # média da temperatura no dia
> tmin <- tapply(met$temp, dia, min)  # a mínima da temperatura no dia
> tmax <- tapply(met$temp, dia, max)  # a máxima da temperatura no dia
> ylim <- extendrange(range(c(tmax,tmin))) # amplitude de y para caber as séries
> dia <- as.POSIXlt(format(met$d, format="%d/%m/%Y"), format="%d/%m/%Y")
> plot(unique(dia), tmed, ylim=ylim, type="l",
+      xlab="Tempo (dias, 2010-2011)", ylab="Temperatura", xaxt="n")
> lines(unique(dia), tmin, col=2)
> lines(unique(dia), tmax, col=2)
> axis.POSIXct(1, at=seq(as.POSIXct("2010-06-01"), as.POSIXct("2011-07-01"), by="month"), format="%b")
> axis.POSIXct(1, at=seq(as.POSIXct("2010-06-01"), as.POSIXct("2011-07-01"), by="week"), format="%W",
+              labels=FALSE, tcl=-0.2)
> legend("bottomright", bty="n", legend=c("mínima e máxima","média"), lty=1, col=c(2,1), ncol=2)
> #-----------------------------------------------------------------------------------------
```

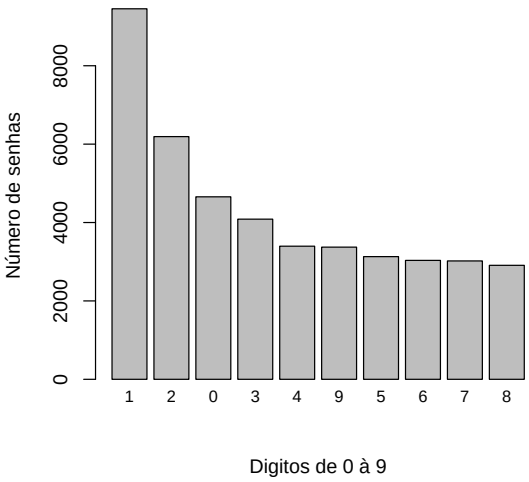

Potássio liberado (mg/dm³)
40
60
80
100
120
140
50
100
150
200
250
Tempo (dias)

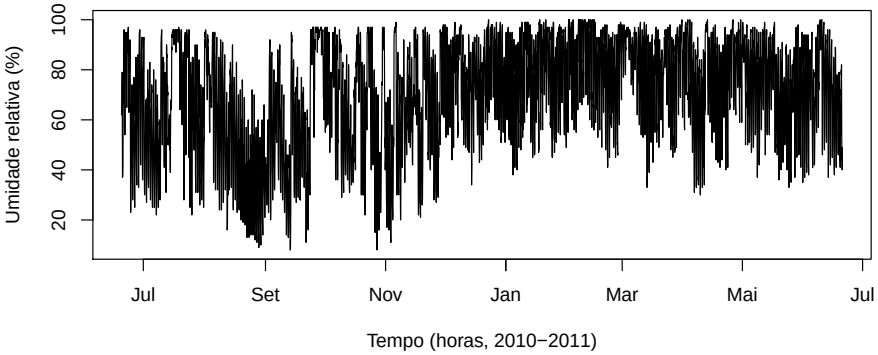

Número de usuários
0
2000
6000
10000
14000
outros
yahoo
gmail
hotmail
msn
live
Serviço de e−mail

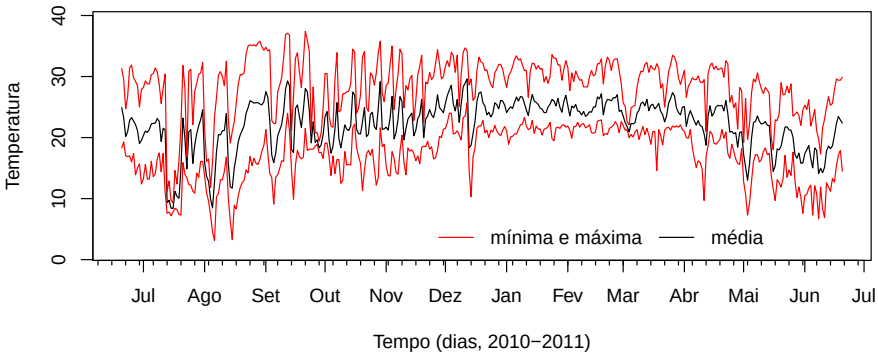

Propoção de usuários de cada serviço de e−mail (%)
gmail (14.6)
hotmail (13.3)
yahoo (26.4)
msn (3.14)
live (0.86)
outros (41.7)


Número de senhas
0
2000
4000
6000
8000
10000
0
2
4
6
8
10
13
16
19
24
Número de caracteres na senha


Número de senhas
0
5000
10000
15000
e a r i s o n l t m c d y h b u k g p w f j v z x q
Letra minúscula do alfabeto


Número de senhas
0
2000
4000
6000
8000
1
2
0
3
4
9
5
6
7
8
Digitos de 0 à 9


Umidade relativa (%)
20
40
60
80
100
Jul
Set
Nov
Jan
Mar
Mai
Jul
Tempo (horas, 2010−2011)


Temperatura
0
10
20
30
40
mínima e máxima
média
Jul
Ago
Set
Out
Nov
Dez
Jan
Fev
Mar
Abr
Mai
Jun
Jul
Tempo (dias, 2010−2011)