LINKÖPINGS UNIVERSITET
Institutionen för datavetenskap
STIMA
Josef Wilzén

2020-06-09
Programmering i R, 7.5 hp
732G33 och 732G83

Tentamen i Programmering i R, 7.5 hp

Skrivtid: 8.00-12.00

Betygsgränser: Tentamen omfattar totalt 20 poäng. 12 poäng ger Godkänt,
16 poäng ger Väl godkänt.

Skriv dina lösningar i **fullständig och läsbar kod**. Kommentera din kod och använd en god kodstil.
Kommentera direkt i din R-fil när något behöver förklaras eller diskuteras.
Eventuella grafer som skapas under tentans gång behöver **INTE** skickas in för rättning,
det räcker med att skicka in den kod som producerar figurerna.

Instruktioner

- Se filen "**Information om hemtenta i 732G33 och 732G83.pdf**" för regler och detaljer kring tentan, där finns bland annat information om:
  - Inlämning
  - Kontakt med lärare
  - Hjälpmedel
- När ni är klara med tentan: På Lisam kan ni se ert Anonym-id, vilket är ert unika tenta-id. Detta ska ni använda när ni lämnar in er fil, ni namnger filen på formen [Anonym-id].txt Till exempel om ni har Anonym-id: 12345, så ska er inlämnade fil ha namnet 12345.txt Ni ska alltså lämna in en .txt-fil och inte en .R-fil. Eftersom det är en anonym tenta så ska denna fil ska **inte** innehålla ert namn eller Liu-ID.
- Era lösningar ska innehålla lämpliga kommenater, där ni förklarar de övergripande dragen och de viktiga stegen i er kod. Ni behöver inte föklara alla detaljer. Koden ska även ha en god kodstil. T.ex. så ska ni ha lämplig indentering och bra variabelnamn. Lösningar med fungerade kod men med brister i kommenater eller kodstil får poängavdrag.
- **Spara era lösningar ofta, om R kraschar kan kod förloras.**
- Tentan består av 5 uppgifter som ger 4 poäng vardera.

Uppgifter

Uppgift 1: Kontrollstrukturer och datastrukturer

Del A 2p

Nu ska funktionen `while_calc(n,type)` skapas. Funktionen ska kunna utgår från den numeriska vektorn `n`, och om `type=''sum''` så ska summan av alla elementen i vektorn beräknas. Om `type=''prod''` så ska produkten av alla elementen i `n` beräknas. Beräkningarna ska ske "för hand" med hjäp av en while-loop. Inga funktioner som direkt beräknar summor eller produkter är tillåtna i denna uppgift, t.ex. `sum()` och `prod()` är inte tillåtna. Se exemplen nedan för hur funktionen ska fungera.

```
a<-while_calc(n = 1:3,type = "sum")
a

[1] 6

b<-while_calc(n = 1:3,type = "prod")
b

[1] 6

while_calc(n = 10:9,type = "prod")

[1] 90

while_calc(n = 10:9,type = "sum")

[1] 19

while_calc(n = c(2,4,2,10,3,3),type = "sum")

[1] 24

while_calc(n = c(2,4,2,10,3,3),type = "prod")

[1] 1440
```

Del B 2p

Nu ska ni skapa en funktion som kan genera en vektor, en matris eller en lista. Funktionen ska hete `data_struct(type)`, där `type` är en textsträng. `type` kan antingen vara "vector", "matrix" eller "list", och ska då ska funktionen retunera en förutbestämnd datastruktur, se exemplen nedan för det exakta innehållet i de olika fallen. Om `type` anges till en annan textsträng än de nämnda så ska ett felmeddelande gerneras: "not supported". Se testfallen nedan.

```
A<-data_struct(type="vector")
A
```

```
 [1] 3 3 3 3 2 2 2 2 1 1 1 1
```

```
data_struct(type="matrix")
```

```
     [,1] [,2] [,3] [,4]
[1,] "A"  "D"  "G"  "J"
[2,] "B"  "E"  "H"  "K"
[3,] "C"  "F"  "I"  "L"
```

```
data_struct(type="list")
```

```
$e1
[1]  1 20 87

$e1
[1]  TRUE FALSE FALSE FALSE

$e3
 [1]   100   400   900  1600  2500  3600  4900  6400  8100 10000
```

```
data_struct(type = "hej")
```

```
Error in data_struct(type = "hej"): not supported
```

```
data_struct(type = "data.frame")
```

```
Error in data_struct(type = "data.frame"): not supported
```

```
data_struct(type = "xyz")
```

```
Error in data_struct(type = "xyz"): not supported
```

Uppgift 2: Spektraltätheter för autoregressiva modeller 4p

Autoregresiva (AR) modeller är en vanlig klass av tidseriemodeller. För att kunna tolka en AR-modell enklare så kan dess spektraltäthet[1] beräknas i frekvensdomänen. Spektraltätheten för en AR(0) ges av:

$$S(f) = \sigma^2$$

Där σ^2 är variansen. Spektraltätheten för en AR(1) ges av:

$$S(f) = \frac{\sigma^2}{1 + \varphi_1^2 - 2\varphi_1 cos(2 \cdot \pi \cdot f)}$$

Där f är frekvensen och φ_1 är den första AR-parametern.
Spektraltätheten för en AR(2) ges av:

$$S(f) = \frac{\sigma^2}{1 + \varphi_1^2 + \varphi_2^2 - 2\varphi_1(1 - \varphi_2)cos(2 \cdot \pi \cdot f) - 2\varphi_2 cos(4 \cdot \pi \cdot f)}$$

Där f är frekvensen och φ_1 och φ_2 är de två AR-parameterarna. Se figur 1 för exempel på olika spektraltärheter med olika värden på φ.

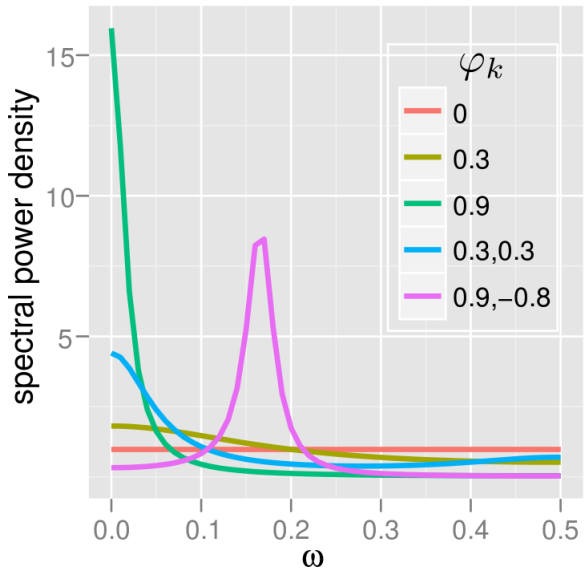


Figure 1: Exempel på olika spektraltätheter med olika värden på φ. Notera att frekvenserna kallas ω här istället för f.

Ni ska i denna uppgift skapa en funktion som heter `my_spec(f,sigma2,phi)` som ska kunna beräkna spektraltätheter för AR(0), AR(1) och AR(2) modeller. Argument:

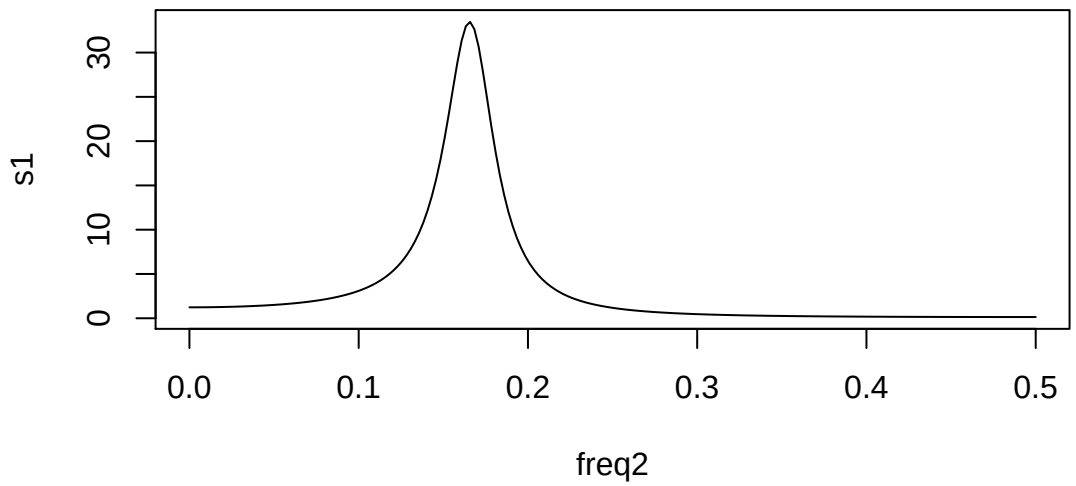
- `f`: är en numerisk vektor med frekvenser, mellan 0 och 0.5
- `sigma2`: är numeriskt värde som representerar variansen σ^2

[1] Mer info finns här.

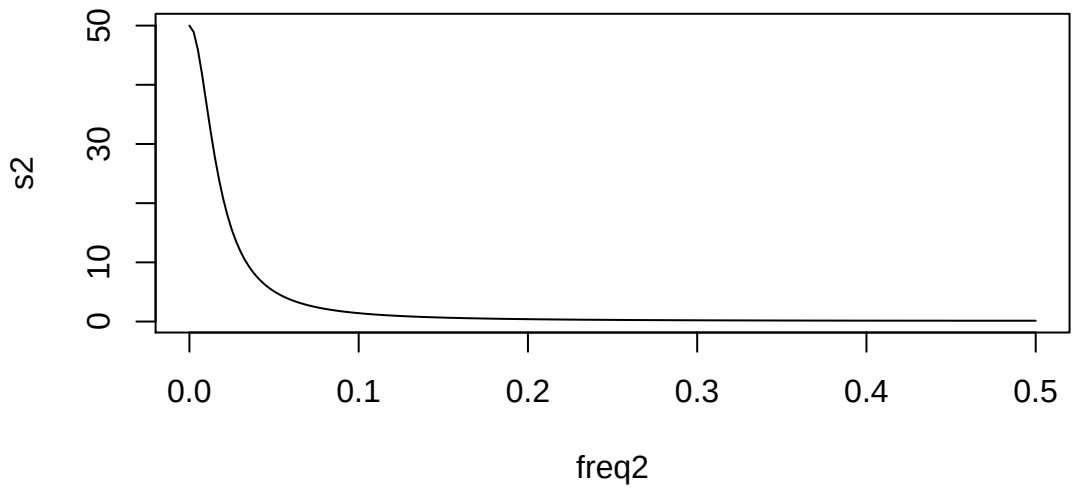
- `phi`: är en numerisk vektor där elementen är de olika värdena på φ.

Om `phi=NULL` då ska spektraltätheten för en AR(0) beräknas. Om `phi` är en numerisk vektor av längd ett så ska spektraltätheten för en AR(1) beräknas. Om `phi` är en numerisk vektor av längd två så ska spektraltätheten för en AR(2) beräknas. Om `phi` är en numerisk vektor som har mer än två element så ska funktionen generera ett fel med valfritt meddelande. Funktionen ska returnera en vektor med beräknade spektraltätheter av samma längd som `f`. Se testfallen nedan.

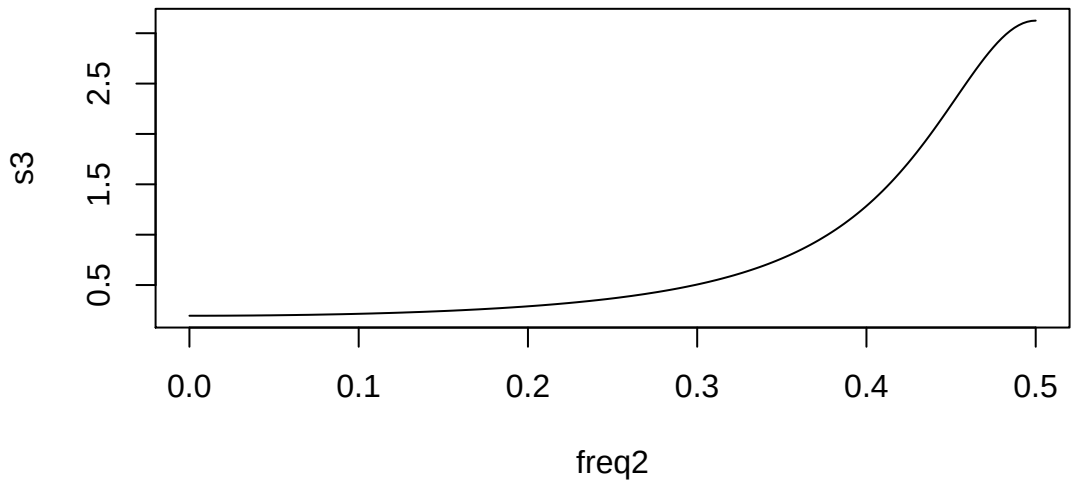
```
freq1<-seq(0,0.5,length=5)

my_spec(f = freq1,sigma2 = 2,phi = c(1,2,3))

Error in my_spec(f = freq1, sigma2 = 2, phi = c(1, 2, 3)): phi is too long

my_spec(f = freq1,sigma2 = 2,phi = NULL)

[1] 2 2 2 2 2

my_spec(f = freq1,sigma2 = 2,phi = 0.7)

[1] 22.222222  3.999596  1.342282  0.806468  0.692042

my_spec(f = freq1,sigma2 = 2,phi = c(0.7,-0.1))

[1] 12.500000  4.865522  1.538462  0.772516  0.617284

freq2<-seq(0,0.5,length=200)
s1<-my_spec(f = freq2,sigma2 = 1,phi = c(0.9,-0.8))
plot(freq2,s1,t="l")
```



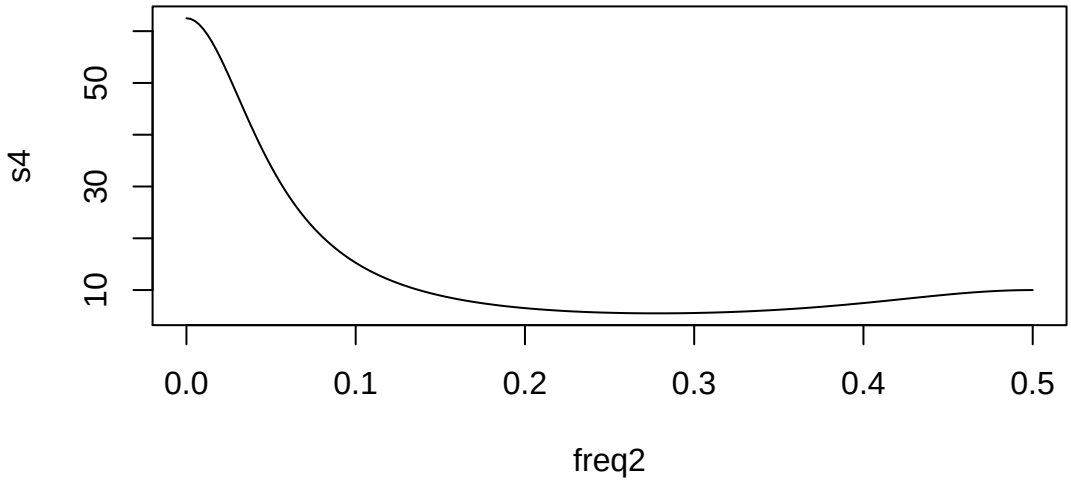
```r
s2<-my_spec(f = freq2,sigma2 = 0.5,phi = c(0.9))
plot(freq2,s2,t="l")
```



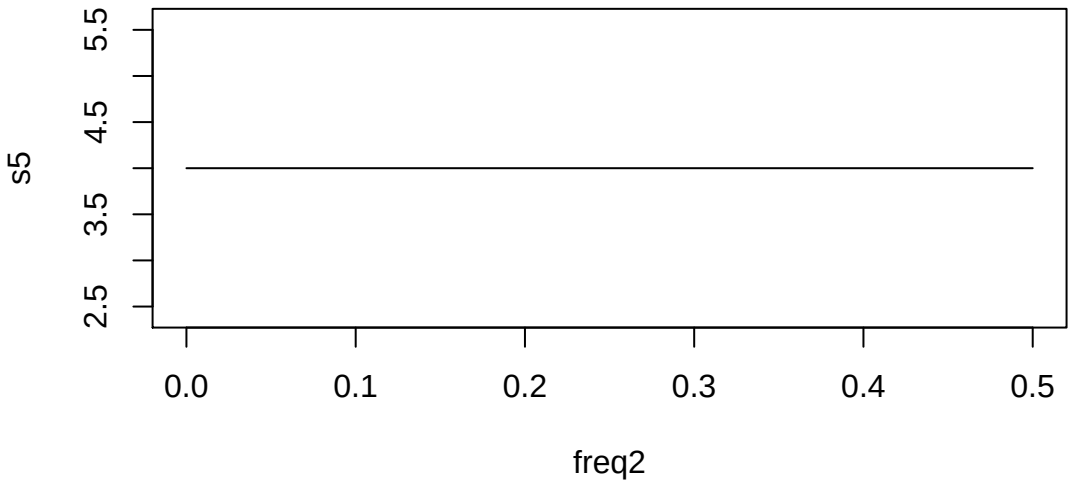

```r
s3<-my_spec(f = freq2,sigma2 = 0.5,phi = c(-0.6))
plot(freq2,s3,t="l")
```




```r
s4<-my_spec(f = freq2,sigma2 = 10,phi = c(0.3,0.3))
plot(freq2,s4,t="l")
```




```
s5<-my_spec(f = freq2,sigma2 = 4,phi = NULL)
plot(freq2,s5,t="l")
```



Uppgift 3: Strängar 4p

Nu ska en funktion skapas som kan derivera ett polynom analytiskt. Utgå från polynomet

$$y = f(x) = a \cdot x^n$$

Då ges derivatan av uttrycket

$$\frac{dy}{dx} = a \cdot n \cdot x^{n-1}$$

där a är en reel konstant och n är ett heltal. Till exempel om

$$y = f(x) = 8x^3$$

så är

$$\frac{dy}{dx} = 24 \cdot x^2$$

Skapa nu funktionen `ploy_derivative(f)`, som ska ta en textsträng som argument. Exempel på sträng: `f=''2x^3''`. Om f inte är en sträng så ska funktionen avbryta med valfritt felmeddelande. Funktionen ska returnera utrycket för derivatan som en texsträng. Utgå i från att variabeln alltid heter x. a antas vara postivt heltal och n är ett heltal mellan 1 och 9. a antas kunna bestå av godtyckligt antal tecken. Funktionen ska returnera en text-sträng på formen enligt testfallen nedan.

```
ploy_derivative(f = 33)

Error in ploy_derivative(f = 33): f is not a string!

a<-ploy_derivative(f = "3x^1")
str(a)

 chr "dy/dx=3x^0"

a

[1] "dy/dx=3x^0"

ploy_derivative(f = "2031x^2")

[1] "dy/dx=4062x^1"

ploy_derivative(f = "25x^3")

[1] "dy/dx=75x^2"

ploy_derivative(f = "8x^9")

[1] "dy/dx=72x^8"

ploy_derivative(f = "12345x^4")

[1] "dy/dx=49380x^3"
```

Uppgift 4: Indexera listor 4p

Ni ska nu skapa funktionen `list_index(X,index_list)`. `X` är en lista som ska indexeras. Vi utgår ifrån att alla element i `X` innehåller vektorer (som kan vara olika långa). `index_list` är en annan lista där varje element är ett index för motsvarande element i `X`. Så första elemetet i `index_list` ska alltså användas att indexera första elementet i `X` och så vidare.

Funktionen ska först testa att `X` och `index_list` är listor, annars ska de avbrytas med felmeddelandet som ges av exemplen. Sen ska ni testa att `X` och `index_list` har lika många element, om inte ska funktionen avbrytas med felmeddelandet "list length mismatch".

Sen ska funktionen ta indexen i `index_list` och indexera motsvarande element i `X`. Sen ska den modiferade listan returneras. Notera att listan som returneras ska ha samma elmentnamn och ordning på elementen som `X`. Se exemplen nedan på hur funktionen ska fungera.

```
a1<-list((1:10)*3)
index1<-list(c(10,2,3))

# testa felmeddelanden:
list_index(X = a1,index_list = TRUE)

Error in list_index(X = a1, index_list = TRUE): index_list is not a list

list_index(X = "abc",index_list = index1)

Error in list_index(X = "abc", index_list = index1): X is not a list

list_index(X = a1,index_list = list(1,2,3,4))

Error in list_index(X = a1, index_list = list(1, 2, 3, 4)): list length mismatch

# fall 1
list_index(X = list(c(1,5,7)),index_list = list(c(1,3)))

[[1]]
[1] 1 7

# fall 2:
b1<-list_index(X = a1,index_list = index1)
b1

[[1]]
[1] 30  6  9

# fall 3:
data("trees")
a2<-as.list(trees)
index2<-list(1:3,31:28,c(3,4))
b2<-list_index(X = a2,index_list = index2)
str(b2)

List of 3
 $ Girth : num [1:3] 8.3 8.6 8.8
 $ Height: num [1:4] 87 80 80 80
 $ Volume: num [1:2] 10.2 16.4
```

```r
# fall 4:
data("AirPassengers")
a3<-list(a=month.name,b=1:5,c=as.vector(AirPassengers)[1:20],sqrt(1:10))
index3<-list(c(12,9),5:3,c(1,3,5,20),c(2,5,9))
b3<-list_index(X = a3,index_list = index3)
str(b3)
```

```
List of 4
 $ a: chr [1:2] "December" "September"
 $ b: int [1:3] 5 4 3
 $ c: num [1:4] 112 132 121 170
 $  : num [1:3] 1.41 2.24 3
```

Uppgift 5: Datum och Grafik 4p

Läs in datamaterialet "coffee_data.csv". Data innehåller information om antal sålda koppar på ett café för olika datum.

Del A

Gör en tidserieplot (med linje) för antalet sålda koppar kaffe. x-axeln ska ha rätt tidsskala och linjen ska vara röd. Detta ska göras med ggplot2.

Del B

Skapa grupperade boxplots för alla månader för antalet sålda koppar kaffe. Det ska alltså bli tolv boxplots brevid varandra i samma graf, och alla ska vara fyllda med blå färg. Den första boxploten ska vara baserad på alla värden för januari, den andra ska vara baserad på alla värden för februari osv. Det ska tydligt framgå i grafen vilken månad som hör till vilken boxplot. Detta ska göras med ggplot2.

Del C

Räkna ut medianen för antalet sålda koppar kaffe för alla vardagar och för alla helger (lördag och söndag). Spara dessa två värden i två variabler med lämpliga namn.

Kom ihåg: När ni är klara med tentan: På Lisam kan ni se ert Anonym-id, vilket är ert unika tenta-id. Detta ska ni använda när ni lämnar in er fil, ni namnger filen på formen [Anonym-id].txt Till exempel om ni har Anonym-id: A-12345, så ska er inlämnade fil ha namnet A-12345.txt Ni ska alltså lämna in en .txt-fil och inte en .R-fil. Eftersom det är en anonym tenta så ska denna fil ska **inte** innehålla ert namn eller Liu-ID.

Lycka till!