```
#############################################################
#
#    Reinhard Hoepfner -- Statistik mit Rechneruebungen fuer Lehramtsstudierende -- SoSe19
#
#    Uebungsblatt 3
#
#    Konfidenzintervalle fuer Lokations- und Skalenmodelle
#    erzeugt von Doppelexponentialverteilung
#
#    fuer verschiedene Paare 'richtig skalierender' Schaetzer
#
#    21.06.19
#
###########################################################3



########################################
#
#   doppelexponentialverteilung und
#   doppelexponentialverteilte zufallszahlen
#
########################################


meine_quantile <- function(dat){
 abline( v=min(dat), col=2, lty=3 ) ;
 abline( v=quantile(dat,0.025), col=2, lty=4 ) ;
 abline( v=quantile(dat,0.05), col=2, lty=2 ) ;
 abline( v=quantile(dat,0.1), col=2, lty=6 ) ;
 abline( v=quantile(dat,0.9), col=2, lty=6 ) ;
 abline( v=quantile(dat,0.95), col=2, lty=2 ) ;
 abline( v=quantile(dat,0.975), col=2, lty=4 ) ;
 abline( v=max(dat), col=2, lty=3 ) ;
 } ; # ende funktionsdef
 # quantile irgendeines datensatzes


bilde_doppelexp_dichte <- function(x, m, c){ dexp( abs( (x-m)/c ) ) / (2*c)  } ;
  # funktion symmetrisiert exponentialverteilungen (fertig in R)
  # zu doppelexponentialverteilungen


bilde_doppelexp_zv <- function(N, m, c){
  yy <- rexp( N, rate=1/c ) ;
  vz <- 2*( rbinom(N,1,0.5) - 0.5 ) ; # vorzeichen plus minus
  vz*yy + m*rep(1,N) ;
  } ; # funktion erzeugt doppelexponentialverteilte zv
  # mit lokationsparameter  m  und skalenparameter  s


  # test
N <- 5000 ;
m <- 2.75 ;
c <- 1.33 ;
leg1 <- "histogramm von %%%% doppelexponentialvert zv, lokationspar , skalenpar &&&&" ;
leg1 <- gsub( "%%%%", N, leg1 ) ;
leg1 <- gsub( "", round(m,2), leg1 ) ;
leg1 <- gsub( "&&&&", round(c,2), leg1 ) ;

zv1 <- bilde_doppelexp_zv( N, m, c ) ;
hist( zv1, probability=T, nclass=150, col=8, ylim=c( 0, 1.1/(2*c) ), main=leg1 )  ;

xx1 <- seq( min(zv1)-1, max(zv1)+1, length.out=1000 ) ;
yy1 <- bilde_doppelexp_dichte( xx1, m, c ) ;
lines( xx1, yy1, col=2,lwd=2, lty=6 ) ;
meine_quantile( zv1 ) ;
```

```
#########################################
#
#  unterschied zwischen doppelexponentialverteilung und
#  normalverteilung mit denselben parametern
#
########################################


normalquantile <- function(m,c){
 abline( v=m-3*c, col=6, lty=3 ) ;
 abline( v=qnorm(0.025,m,c), col=6, lty=4 ) ;
 abline( v=qnorm(0.05,m,c), col=6, lty=2 ) ;
 abline( v=qnorm(0.1,m,c), col=6, lty=6 ) ;
 abline( v=qnorm(0.9,m,c), col=6, lty=6 ) ;
 abline( v=qnorm(0.95,m,c), col=6, lty=2 ) ;
 abline( v=qnorm(0.975,m,c), col=6, lty=4 ) ;
 abline( v=m+3*c, col=6, lty=3 ) ;
 } ; # ende funktionsdef


par( mfrow=c(2,1) ) ;
#
hist( zv1, probability=T, nclass=150, col=8, ylim=c( 0, 1.1/(2*c) ), main=leg1 )  ;
lines( xx1, yy1, col=2, lwd=2, lty=6, xlab="", ylab="" ) ;
meine_quantile( zv1 ) ;
#
yy2 <- dnorm( xx1, m, c ) ;
plot( range(zv1), range(yy2), type="n", xlab="", ylab="") ;
lines( xx1, yy2, col=6, lwd=2, lty=2 ) ;
normalquantile(m,c) ;
#
par( mfrow=c(1,1) ) ;

 # gestalt deutlich anders als fuer eine normalverteilung
 # mit denselben lokations- und skalenparametern
 # ganz besonders in den tails



##########################################
#
#    definiere drei verschiedene schaetzer fuer das paar
#    (m,c) = (lokationsparameter, skalenparameter )
#
##########################################


  # schaetzer (Mi,Ci) fuer das paar (m,c) , die dieselben
  # skalierungseigenschaften wie ( emp mw, sqrt(emp var) ) besitzen
  # und daher zur festlegung von konfidenzintervallen
  # zu vorgegebenem konfidenzkoeffizient benutzt werden koennen

  # lokationsparameter
M1 <- function(daten) median(daten) ;
  # median
M2 <- function(daten) median(daten) ;
  # median
M3 <- function(daten) mean(daten) ;
  # empirischer mittelwert

  # skalenparameter
C1 <- function(daten) mean( abs( daten - M1(daten) ) ) ;
  # MLE im doppelexponentialverteilungsmodell
C2 <- function(daten) M1( abs( daten - median(daten) ) ) ;
  # komplett nichtparametrischer schaetzer : medianabstand
C3 <- function(daten) sqrt(var(daten)) ;
  # empirische varianz
```

```
########################################
#
#    testdatensatz
#
########################################

testdat <- c(
  6.0490067, 20.0324319, 8.0917892, 5.3001702, 8.4231110, 8.4178589,
  7.9412088, 6.0004114, 7.6231672, 3.2640157, 7.9500893, 7.1471330,
  -3.7385962, 13.1849629, 15.3080767, 6.6531491, 9.1205368, 0.9610968,
  4.9916255, 11.5371074, 3.1072696, 5.7450847, 5.0906057, 9.5253568,
  4.7945269, 4.0560317, 10.1374325
  ) ; # der datensatz aus aufgabe 3.1 d)

  # MLE im doppelexponentialverteilungsmodell
c( M1(testdat), C1(testdat) ) ;
#   [1] 7.147133 3.148842

  # komplett nichtparametrischer schaetzer : median und medianabstand
c( M2(testdat), C2(testdat) ) ;
#   [1] 7.147133 2.056527

  # emp mw und emp var fuer L^2-lokationsmodelle
c( M3(testdat), C3(testdat) ) ;
#   [1] 7.285728 4.524473

  # ein 'wahrer wert' ist fuer den testdatensatz natuerlich unbekannt
hist( testdat, breaks=seq(min(testdat)-5,max(testdat)+5,length.out=15), prob=T ) ;
xx3 <- seq( min(testdat), max(testdat), length.out=1000 ) ;
lines( xx3, bilde_doppelexp_dichte( xx3, M1(testdat), C1(testdat) ) ) ;
  # der testdatensatz passt also zumindest ungefaehr
  # zu einer doppelexponentialverteilung


#######################################################
#
#   lokations- und skalenexperiment
#   erzeugt von der doppelexponentialverteilung
#
#   empirische verteilungen fuer die statistiken
#   r_n,  v_n,  w_n  aus aufgabe 3.3 b)
#
#   analog zu  t-verteilungen im normalverteilungsmodell
#   werden diese gebaut aus 'richtig skalierenden' schaetzern
#   fuer lokations- und skalenparameter
#
########################################################


  # empirische verteilung von  T1 := v_n
  # benutzt komponenten des MLE im doppelexponentialverteilungsmodell
emp_law_T1 <- function(n,N){
  hilf <- 1:N ;      # startwerte zum ueberschreiben
  for ( i in 1:N ) {
     dat <- bilde_doppelexp_zv( n, 0, 1 ) ;
     hilf[i] <- sqrt(n) * M1(dat) / C1(dat) ;
     } ; # ende der i schleife
  hilf ; } ; # ende funktionsdef, argumente sind
  # n = stichprobenlaenge des anvisierten datensatzes
  # N = zahl der simulationslaeufe zur simulation der verteilung

N <- 10000 ;
n <- length(testdat) ;  n ;
hilf1 <- emp_law_T1(n,N) ;
summary( hilf1 ) ;
leg1 <- "empirische verteilung der statistik T1 := v_n  im doppelexp.vert.modell" ;
hist( hilf1, prob=T, nclass=0.02*N, main=leg1, xlab="", ylab="" ) ;
```

```
  # empirische verteilung von  T2 := r_n
  # komplett nichtparametrisch, benutzt
  # median und medianabstand
emp_law_T2 <- function(n,N){
  hilf <- 1:N ;   # startwerte zum ueberschreiben
  for ( i in 1:N ) {
     dat <- bilde_doppelexp_zv( n, 0, 1 ) ;
     hilf[i] <- sqrt(n) * M2(dat) / C2(dat) ;
     } ; # ende der i schleife
  hilf ; } ; # ende funktionsdef, argumente sind
  # n = stichprobenlaenge des anvisierten datensatzes
  # N = zahl der simulationslaeufe zur simulation der verteilung

hilf2 <- emp_law_T2(n,N) ;
summary( hilf2 ) ;
leg2 <- "empirische verteilung der statistik  T2 := r_n  im doppelexp.vert.modell" ;
hist( hilf2, prob=T, nclass=0.02*N, main=leg2, xlab="", ylab="" ) ;


 # empirische verteilung von  T3 := w_n
 # benutzt mw und standardabweichung
emp_law_T3 <- function(n,N){
  hilf <- 1:N ;      # startwerte zum ueberschreiben
  for ( i in 1:N ) {
     dat <- bilde_doppelexp_zv( n, 0, 1 ) ;
     hilf[i] <- sqrt(n) * M3(dat) / C3(dat) ;
     } ; # ende der i schleife
  hilf ; } ; # ende funktionsdef, argumente sind
  # n = stichprobenlaenge des anvisierten datensatzes
  # N = zahl der simulationslaeufe zur simulation der verteilung


hilf3 <- emp_law_T3(n,N) ;
summary( hilf3 ) ;
leg3 <- "empirische verteilung der statistik  T3 := w_n  im doppelexp.vert.modell" ;
hist( hilf3, prob=T, nclass=0.02*N, main=leg3, xlab="", ylab="" ) ;


  # modell kommt von der doppelexponentialverteilung
  # also ist das keine t verteilung
  # aber ist trotzdem nahe daran
x2 <- seq(-5,5,0.001 ) ;
y2 <- dt( x2, n-1 ) ;
lines( x2, y2, col=6 ) ;


####################################
#
#   vergleich der verteilungen  r_n,  v_n,  w_n
#
####################################


leg1 <- "verteilung von  T1 = v_n  fuer n= simuliert in N=%%%% laeufen im doppelexp.vert.-
loc+scale-modell \n v_n  gebildet aus komponenten des MLE im doppelexponentialverteilungs-
loc+scal-modell" ;
leg1 <- gsub( "", n, leg1 ) ;
leg1 <- gsub( "%%%%", N, leg1 ) ; leg1 ;

leg2 <- "verteilung von  T2 = r_n  fuer n= simuliert in N=%%%% laeufen im doppelexp.vert.-
loc+scal-modell \n r_n  gebildet komplett nichtparametrisch aus median und medianabstand" ;
leg2 <- gsub( "", n, leg2 ) ;
leg2 <- gsub( "%%%%", N, leg2 ) ; leg2 ;

leg3 <- "verteilung von  T3 = w_n  fuer n= simuliert in N=%%%% laeufen im
```

```
doppelexponentialverteilungs- loc+scal-modell \n w_n  gebildet wie im normalverteilungsmodell aus
mittelwert und standardabweichung" ;
leg3 <- gsub( "", n, leg3 ) ;
leg3 <- gsub( "%%%%", N, leg3 ) ; leg3 ;


m <- min( c(hilf1,hilf2,hilf3) ) - 0.1 ; m ;
M <- max( c(hilf1,hilf2,hilf3) ) + 0.1 ; M ;
meinebreaks <- seq( round(m,2), round(M,2), 0.05 ) ;
par( mfrow=c(3,1) ) ;
hist( hilf1, prob=T, breaks=meinebreaks, xlim=c(m,M), main=leg1, xlab="", ylab="" ) ;
meine_quantile(hilf1) ;
hist( hilf2, prob=T, breaks=meinebreaks, xlim=c(m,M), main=leg2, xlab="", ylab="" ) ;
meine_quantile(hilf2) ;
hist( hilf3, prob=T, breaks=meinebreaks, xlim=c(m,M), main=leg3, xlab="", ylab="" ) ;
meine_quantile(hilf3) ;
par( mfrow=c(1,1) ) ;



######################################
#
#   konfidenzintervalle fuer den lokationsparameter
#
######################################


  # T1 = v_n gebildet aus komponenten des MLE
  # im doppelexponentialverteilungsmodell
conf_int_T1 <- function(daten,beta){
  n <- length(daten) ;
  hilf <- emp_law_T1(n,10000) ;
  q_oben <- -quantile( hilf, (1-beta)/2 ) ;
  q_unten <- -quantile( hilf, 1-(1-beta)/2 ) ;
  M <- M1(daten) ;
  C <- C1(daten) ;
  c( M + q_unten*C/sqrt(n) , M + q_oben*C/sqrt(n) ) ;
  } ; # ende funktionsdef und ausgabe konfidenzintervall
  # zum konfidenzkoeffizienten beta
  # fuer den datensatz daten


  # T2 = r_n gebildet komplett nichtparametrisch
  # aus median und medianabstand
conf_int_T2 <- function(daten,beta){
  n <- length(daten) ;
  hilf <- emp_law_T2(n,10000) ;
  q_oben <- -quantile( hilf, (1-beta)/2 ) ;
  q_unten <- -quantile( hilf, 1-(1-beta)/2 ) ;
  M <- M2(daten) ;
  C <- C2(daten) ;
  c( M + q_unten*C/sqrt(n) , M + q_oben*C/sqrt(n) ) ;
  } ; # ende funktionsdef und ausgabe konfidenzintervall
  # zum konfidenzkoeffizienten beta
  # fuer den datensatz daten


  # T3 = w_n gebildet mit mw und standardabweichung
  # wie im Normalverteilungsmodell
conf_int_T3 <- function(daten,beta){
  n <- length(daten) ;
  hilf <- emp_law_T3(n,10000) ;
  q_oben <- -quantile( hilf, (1-beta)/2 ) ;
  q_unten <- -quantile( hilf, 1-(1-beta)/2 ) ;
  M <- M3(daten) ;
  C <- C3(daten) ;
  c( M + q_unten*C/sqrt(n) , M + q_oben*C/sqrt(n) ) ;
```

```
  } ; # ende funktionsdef und ausgabe konfidenzintervall
  # zum konfidenzkoeffizienten beta
  # fuer den datensatz daten


##########################################
#
#   anwendung auf den testdatensatz
#
#########################################


 # boxplot zur darstellung des datensatzes
boxplot( testdat , range=0.5, col=5 ) ;
title( main = "testdatensatz" ) ;

 # erinnerung: voreinstellungen in 'boxplot' :
abline( h = min(testdat), lty=2, col=4 ) ;
abline( h = max(testdat), lty=2, col=4 ) ;
abline( h = median(testdat), lty=2, col=4 ) ;
abline( h = quantile(testdat,0.75), lty=2, col=4 ) ;
abline( h = quantile(testdat,0.25), lty=2, col=4 ) ;




beta <- 0.9 ;
leg1 <- "int aus komponenten des MLE" ;
leg2 <- "int aus median und medianabstand" ;
leg3 <- "int  aus mw und standardabweichung" ;
xx <- rep(0,2) ;
zz1 <- conf_int_T1( testdat, beta ) ; zz1 ;
zz2 <- conf_int_T2( testdat, beta ) ; zz2 ;
zz3 <- conf_int_T3( testdat, beta ) ; zz3 ;

par( mfrow=c(1,4) ) ;
#
boxplot( testdat , range=0.5, col=5 ) ;
title( main = "testdatensatz" ) ;
#
plot( range(xx), range(testdat), type="n", xlab="", ylab="", main=leg1 ) ;
lines( xx, zz1, lwd=4, col=6 ) ;
abline( h=zz1[1], lty=2, col=6 ) ;
abline( h=zz1[2], lty=2, col=6 ) ;
#
plot( range(xx), range(testdat), type="n", xlab="", ylab="", main=leg2 ) ;
lines( xx, zz2, lwd=4, col=6 ) ;
abline( h=zz2[1], lty=2, col=6 ) ;
abline( h=zz2[2], lty=2, col=6 ) ;
#
plot( range(xx), range(testdat), type="n", xlab="", ylab="", main=leg3 ) ;
lines( xx, zz3, lwd=4, col=6 ) ;
abline( h=zz3[1], lty=2, col=6 ) ;
abline( h=zz3[2], lty=2, col=6 ) ;
#
par( mfrow=c(1,1) ) ;



####################################
#
#   ende aufgabe 3.1
#   21.06.19
#
####################################
```