```
##############################################
#
#  Reinhard Hoepfner
#
#  Praktikum zur 'Grundlagen der Stochastik'
#
#  Empirische Verteilungsfunktionen
#
#  normalverteilte ZV -- Konvergenz der emp VF
#
#  05.11.18
#
##############################################



##############################################
#
#   empirische verteilungsfunktionen vorgefertigt in R
#
##############################################

 # siehe :  help(ecdf) ;
 # siehe :  help(rnorm) ; help(dnorm) ;  help(pnorm) ;


 # erzeugen von daten
daten <- rnorm( 10000, 0, 1 ) ;
 # siehe : help(rnorm)


 # ansehen mit
yleg <- "N(0,1) hat 99.7 prozent der masse in (-3,3)" ;
leg <- "meine daten : 10000 iid standardnormalverteilte zufallszahlen" ;
plot( c(1,length(daten)), c( -4.5, 4.5 ), type="n",
 ylab=yleg, xlab="", main=leg ) ;
points( 1:length(daten), daten, col=2, pch=18, cex=0.5 ) ;
abline( h=-3, lty=2, col=3 ) ;
abline( h=3, lty=2, col=3 ) ;


 # bereite vor : verteilungsfunktion standardnormalverteilung
xstep <- 0.01 ;
x <- seq( -5, 5, xstep ) ;
y <- pnorm( x, 0, 1) ;
plot( x, y, type="l", lwd=2, col=4, ylab="", xlab=yleg,
 main="verteilungsfunktion standardnormalverteilung" ) ;
abline( v=-3, lty=2, col=3 ) ;
abline( v=3, lty=2, col=3 ) ;
 # siehe help(pnorm) ;
```

```
 # vorgefertigt in R existiert 'ecdf'
 # die empirische verteilungsfunktion in R,
 # nichts anderes als die aufsteigend geordneten beobachtungen
 # kombiniert mit einem programm zum zeichnen einer sprungfunktion
 # mit werten in ]0,1] und fertig definierten sprunghoehen

plot( ecdf(daten[1:50]) ) ;
# lines(x, y, col=4) ;

plot( ecdf(daten[1:500]) ) ;
# lines(x, y, col=4) ;

 # man kann die dicken punkte wegnehmen
 # (automatisch bei hinreichend vielen daten)
plot( ecdf(daten[1:500]), pch="" ) ;
# plot( ecdf(daten[1:5000]), pch="" ) ;

 # man kann die vertikalen einzeichnen lassen oder nicht
plot( ecdf(daten[1:50]), pch="", verticals=T,
 main="individualisierter titel", xlab="", ylab="" ) ;

plot( ecdf(daten[1:50]), pch="", verticals=F,
 main="individualisierter titel", xlab="", ylab="" ) ;



 # ecdf ruft selber wieder ein programm  stepfun
 # zum zeichnen einer sprungfunktion auf
 # siehe : help(stepfun) ;
plot( ecdf(daten[1:50]) ) ;
 # macht dasselbe wie
plot.stepfun( daten[1:50], pch=19, verticals=F ) ;
abline( h=0, lty=2, col=8 ) ;
abline( h=1, lty=2, col=8 ) ;
 # programm zum zeichnen einer sprungfunktion
lines(x, y, col=4) ;




#####################################
#
#   aehnliches haette man auch selbst schreiben koennen
#
#####################################


N <- 25 ;
xx <- sort(daten[1:N]) ;
 # aufsteigende anordnung der daten
yy <- seq( 1/N, 1, 1/N ) ;
plot( range(daten), c(0,1), type="n",
  main="empirische verteilungsfunktion") ;
lines( xx, yy, type="s" ) ; # sprungfunktion
```

```
points( xx, yy, pch=19 ) ; # punkte dazu
lines(x, y, col=4) ;
 # unschoen : es fehlt das rechte und das linke ende



 # nehme zwei kuenstliche punkte weit aussen dazu
 # um die emp vf 'weit links' auf  0
 # und 'weit rechts' auf 1 zu setzen

xx1 <- c( min(xx)-10, xx, max(xx)+10 ) ;
yy1 <- c( 0, yy, 1 ) ;
plot( range(daten), c(0,1), type="n", xlab="",
 ylab="", main="empirische verteilungsfunktion") ;
lines( xx1, yy1, type="s") ; # sprungfunktion
points( xx1, yy1, pch=19, cex=0.7 ) ; # die punkte dazu
lines(x, y, col=4) ; # die normaldichte zum vergleich




####################################
#
#     'der zufall und sein gesetz' fuer emprische verteilungsfunktionen :
#
#     fuer grosse zahl von beobachtungen
#     sieht man das dem zufall zugrundeliegende gesetz
#     immer deutlicher, aber nie ganz
#
#     empirische verteilungsfunktionen fuer die
#     ersten  N  daten, konvergenz N -> \infty
#
####################################


# daten <- rnorm( 10000, 0, 1 ) ;

par( mfrow=c(2,2) ) ;
n_list <- c( 25, 100, 500, 5000 ) ;

for( n in n_list ) {
  xleg <- "nach den ersten &&&& beobachtungen" ;
  xleg <- gsub( "&&&&", n , xleg ) ;
  plot( ecdf(daten[1:n]), xlab=xleg,
      ylab="", main="", pch="", verticals=T ) ;
  lines(x, y, col=4)  ;
  }  ;

par( mfrow=c(1,1) ) ;
title("normalverteilte zufallszahlen: konvergenz der empirischen verteilungsfunktionen \n") ;
```

```
#####################################
#
#   datenabhaengig kann in der letzten graphik die skalierung
#   der x-achse in jeder teilgraphik anders ausfallen
#
#   hier eine langsamere aber individuellere variante
#
#####################################


relhf <- function(daten,x){
  round( length( daten[ daten<=x ] )/length(daten) , 5 ) ;
  } ; # funktion berechnet die relative haeufigkeit
  # von beobachtungen kleiner oder gleich  x
  # im datensatz  daten

# daten <- rnorm( 10000, 0, 1 ) ;

c( relhf(daten,-3) , 1-relhf(daten,3) , relhf(daten,-3)+1-relhf(daten,3)  ) ;



maleempdf <- function( daten, n, xmin, xmax, xfeinheit ){
  n <- min( n, length(daten) ) ;
  dat <- daten[1:n] ;
  x <- seq( xmin, xmax, xfeinheit ) ;
  y <- x ; # startwert zum ueberschreiben
  for (i in 1:length(y) ) y[i] <- relhf(dat,x[i]) ;
  leg <- "nach den ersten  beobachtungen" ;
  leg <- gsub( "", n, leg ) ;
  plot( c(xmin,xmax), c(0,1), type="n", xlab=leg, ylab="") ;
  lines( x , y, type="s", col=2 ) ; # funktion wird als sprungfunktion rot gemalt
  title(main="empirische verteilungsfunktion") ;
  } ; #  funktion berechnet die empirische verteilungsfunktion
  #  zu den ersten  N  beobachtungen im datensatz  daten  und malt sie
  # auf das interval  (xmin,xmax)  mit schrittweite  xfeinheit

maleempdf( daten, 50, -5, 5, 0.001 ) ;



# daten <- rnorm( 10000, 0, 1 ) ;

par( mfrow=c(2,2) ) ;
n_list <- c( 25, 100, 500, 5000 ) ;

for( n in n_list ) {
  maleempdf( daten, n, -5, 5, 0.001 ) ;
  lines(x, y, col=4)  ;
  } ;
par( mfrow=c(1,1) ) ;


#################################  05.11.18
#########################
```