```
##############################################

Reinhard Hoepfner     SoSe19

Statistik mit Rechneruebungen fuer Lehramtsstudierende

Darstellung von Funktionen  R --> R

25.04.2019

##############################################



##############################################
#
#  wie malt man sich in R bilder reellwertiger funktionen ?
#
##############################################



############################################################
# eine fertig vordefinierte funktion (beispiel sinus):
############################################################

x <- seq(-8.5, 8.5, 0.5) ; x ;
 # befehl erzeugt grobes gitter 'x' von argumenten
 # zwischen -5 und +5 mit schrittweite 0.25
 # help(seq) ;
 # erklaerung aller alternativen moeglichkeiten
 # erklaerungen mit 'q' verlassen


y <- sin(x) ; y;
 # erzeugt liste 'y' von funktionswerten zu den argumenten 'x'
 # dem i-ten wert x[i] des x-gitters
 # entspricht als funktionswert der i-te wert y[i] des y-gitters

plot(x,y) ;
 # scatterplot mit standardmaessig eingestellten symbolen
 # help(plot) ;

plot(x,y, type="p") ;
 # scatterplot wie oben
plot(x,y, type="b") ;
 # punkte und verbindungslinien
plot(x,y, type="l") ;
 # stueckweise linearer plot
plot(x,y, type="h") ;
 # histogrammaehnlicher plot


 #  eine schoene darstellung erfordert
 # ein feines gitter von x-argumenten
x <- seq(-5, 5, length.out=1000 ) ;
y <- sin(x) ;
plot(x,y, type="l", lwd=2) ;
```

```
 # gitterstrukturen einfuegen :
abline( h=1, lty=2 ) ;
abline( h=0, lty=2 ) ;
abline( h=-1, lty=2 ) ;

liste <- seq( -3*pi/2, 3*pi/2, pi/2 ) ;
for(i in 1:length(liste)) abline( v=liste[i], lty=2, col=4 ) ;



##########################################
# einfache plots
##########################################

x <- seq(-8.5, 8.5, 0.5) ;
y <- sin(x) ;


 # symbole fuer scatterplots
m <- 1 ;
 # danach sukzessive
plot(x, y, pch=m ) ; m <- m+1 ;
 # etwa 20 symbole verfuegbar


 # farben fuer scatterplots
m <- 1 ;
 # danach sukzessive
plot(x, y, pch=6, col=m) ; m <- m+1 ;
 # 8 farben verfuegbar


 # symbolgroesse fuer scatterplots
plot(x, y, pch=8) ;
plot(x, y, pch=8, col=3, lwd=4) ;
 # oder
plot(x, y, pch=8, cex=0.5 ) ;
plot(x, y, pch=8, cex=2.5 ) ;




########################################
# erste graphikversuche
########################################

x <- seq(-5, 5, length.out=1000 ) ;
y <- sin(x) ;

plot( c(-5,5), c(-1.25,1.25), type="n", xlab="argumente x", ylab="funktionswerte y",
main="sinusfunktion" ) ;
 # oeffnet einen rechteckigen rahmen frei vorzugebender Groesse,
 # achsenbeschriftungen kann man zwischen die anfuehrungszeichen setzen

lines( x, y ) ;
  # einsetzen: stueckweise linearer plot

  # oder mit symbol, farbe, groesse
lines( x, y, lty=2, col=4, lwd=2 ) ;
```

```
 # herumspielen mit linien verschiedenen typs und verschiedener dicke
plot( c(-5,5), c(-1.25,1.25), type="n", xlab="argumente x", ylab="funktionswerte y",
main="sinusfunktion" ) ;
lines( x, y, lty=4, col=4, lwd=2 ) ;


plot( c(-5,5), c(-1.25,1.25), type="n", xlab="argumente x", ylab="funktionswerte y",
main="sinusfunktion" ) ;
lines( x, y, col=2, lwd=3 ) ;



##############################################
# jetzt eine hinreichend gute graphik malen
##############################################


r <- 4 ;
 # gewuenschte anzahl von dezimalen festlegen
x <- seq( -round(3*pi/2 , r), round(3*pi/2 , r), 10^(-r) ) ;
y <- sin(x) ;


# plot mit adaptiven titeln
xleg <- "argument  x  mit schrittweite  %%%%" ;
xleg <- gsub("%%%%", 10^(-r), xleg) ; xleg ;
xrange <- c( -5, 5 ) ;
yrange <- c( -1.25, 1.25 ) ;
plot( xrange, yrange, type="n", xlab=xleg, ylab="funktionswerte" ) ;
lines( x, y, col=2, lwd=3 ) ;
title(main="ein etwas schoenerer versuch mit sinusfunktion") ;
abline( h = 0, lty=2 ) ;
for( i in (-3):3 ) abline( v = i*pi/2, lty=2 ) ;


 # legenden an einen bestimmten platz setzen
xwert <- pi/6 ;
ywert <- sin(xwert) ;
abline( v=xwert, lty=2, lwd=2, col=2 ) ;
abline( h=ywert, lty=2, lwd=2, col=2 ) ;
 # hier schreibt man die linke obere ecke vor
leg <- " pi / 6 = %%%% \n sin( %%%% ) =  \n\n" ;
leg <- gsub( "%%%%", round(xwert,r), leg ) ; leg ;
leg <- gsub( "", round(ywert,r), leg ) ; leg ;
legend( xwert, ywert, leg, bg=5) ;
 # legende mit linker oberer ecke
 # an den eingegebenen koordinaten einfuegen
 # viele viele moeglichkeiten, siehe
 # help(legend) ;


 # oder man will 2 graphiken auf eine Seite setzen
par( mfrow=c(2,1) ) ;
x <- seq( -4*pi, 4*pi, 0.001 ) ;
y1 <- sin(x) ;
y2 <- cos(x) ;
plot( range(x), c(-1.1,1.1), type="n", xlab="", ylab="",
main="erste graphik: sinus") ;
lines( x, y1, col=2, lwd=2 ) ;
```

```
abline( h=0, lty=2) ; abline( v=0, lty=2) ;
plot( range(x), c(-1.1,1.1), type="n", xlab="", ylab="",
main="zweite graphik: cosinus") ;
lines( x, y2, col=4, lwd=2 ) ;
abline( h=0, lty=2) ; abline( v=0, lty=2) ;
par( mfrow=c(1,1) ) ;



#########################################################################
#
#   eine selbstdefinierte Sprungfunktion
#
#   (Beispiel f(x):=-0.5 falls x<0.3, f(x):=+1.2 sonst)
#
#########################################################################


demo_sprungfunktion <- function(x){ ifelse ( x<0.3, -0.5, 1.2 ) }
  # funktion im argument 'x',
  # falls x < 0.3 ist, gibt sie den wert -0.5 heraus, ansonsten 1.2
  # help(ifelse) ;

x <- seq( -10, 10, 0.01 ) ;
y <- demo_sprungfunktion(x) ;

plot(x,y) ;

plot(x,y,type="b") ;

plot(x,y,type="s") ;
points( 0.3, 1.2, pch=19, lwd=4 ) ;
  # zeichnet die Funktion als Sprungfunktion
  # und hebt funktionswert an der sprungstelle hervor


 # gewuenschte rechtsstetigkeit deutlich machen
x1 <- x[y<0] ; y1 <- y[y<0] ;
x2 <- x[y>0] ; y2 <- y[y>0] ;

plot( c(-10,10), c(-0.75,1.5), type="n", xlab="",
main="eine selbstdefinierte sprungfunktion", ylab="") ;
 # vorgegebener rahmen

lines( x1, y1, lwd=2 ) ;
lines( x2, y2, lwd=2 ) ;
points( min(x2), min(y2), pch=19, cex=1.5 ) ;
points( max(x1), max(y1), pch=1, cex=1.5 ) ;


############################ ende 25.04.2019 ############################
```