

Erste Schritte

Eingabe von Werten, Grundrechenarten, Variablennamen usw.

```
Eingabe :  
2+3  
Ausgabe :  
[1] 5  
Es wird alles als Vektor betrachtet.  
[1] ist der Index der ersten Vektorkomponente in der Zeile.
```

Erzeugung des Vektors der Zahlen 1 bis 20

```
1:20  
  
Ergebnis:  
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15  
[16] 16 17 18 19 20
```

alternativ: Aufruf der Funktion seq (sequence = Folge)

```
seq(2, 6)
```

```
Ergebnis:  
[1] 2 3 4 5 6  
seq(2, 6, by=0.5)
```

```
Ergebnis:  
[1] 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0
```

weitere Grundrechenarten

```
> 2*3  
[1] 6  
> 2/3  
[1] 0.6666667  
> 0.5*7  
[1] 3.5
```

für **Kommentar** bis zum Zeilenende

Die „üblichen“ Konstanten und Funktionen:

```
> pi # dies ist die Kreiszahl  
[1] 3.141593  
> sin(pi/2)  
[1] 1  
> exp(2.5)  
[1] 12.18249  
> log(10)  
[1] 2.302585  
> log10(10)  
[1] 1  
> log2(8)  
[1] 3
```

Zuweisungen (durch <- oder =)

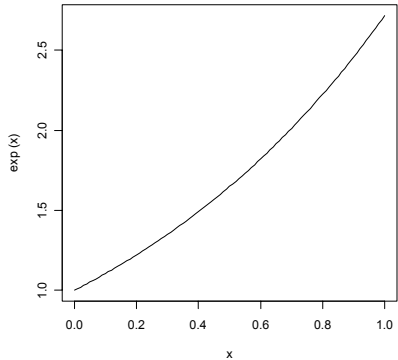
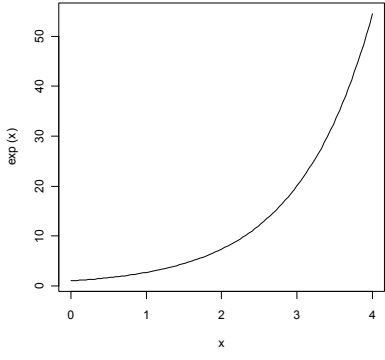
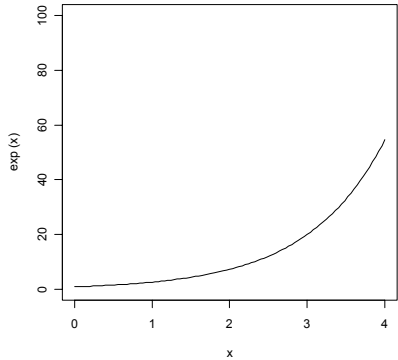
```
> a<-3  
> a  
[1] 3  
Ausgabe des Ergebnisses durch Aufruf des Variablennamens  
  
> 4->b # so geht es auch  
> b  
[1] 4
```

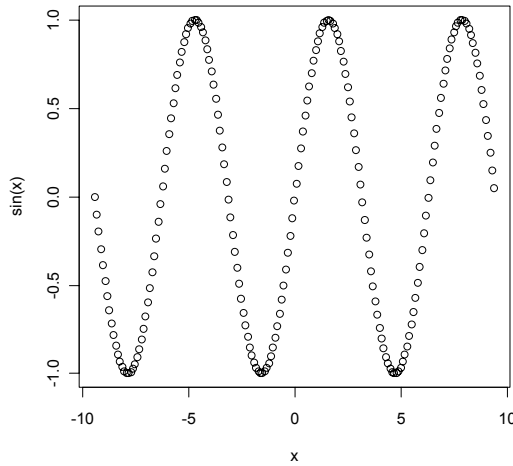
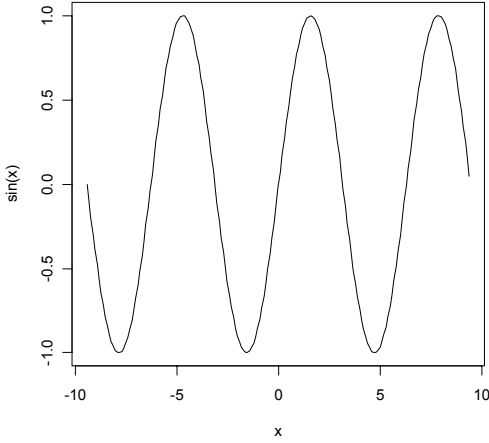
```
> a+b
[1] 7
```

Variablenamen dürfen als Sonderzeichen nur einen Punkt enthalten
 drei.plus.vier <- 3+4

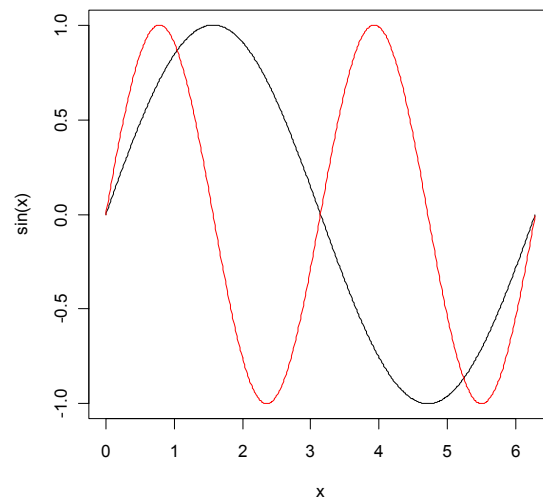
2D-Grafiken in R

Standardfunktionen

| | | |
|---|--|---|
| <pre>plot(exp)</pre> <p>automatische Wahl des Bereichs</p> | <pre>plot(exp,xlim=c(0,4))</pre> <p>x-Werte von 0 bis 4</p> | <pre>plot(exp,xlim=c(0,4), ylim=c(0,100))</pre> <p>y- Werte von 0 bis 100</p> |
|  |  |  |

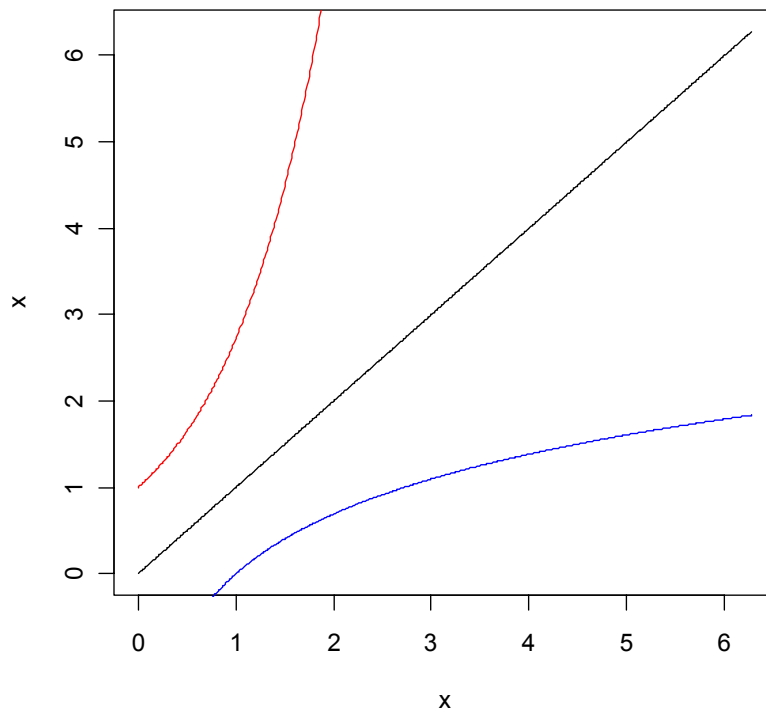
| | |
|---|--|
| <pre>x<-seq(-3*pi,3*pi,0.1)</pre> <pre>plot(x,sin(x))</pre> | <pre>plot(x,sin(x),type="l")</pre> <p>Liniendarstellung</p> |
|  |  |

```
x<-seq(0,2*pi,0.01)
plot(x,sin(x),type="l")
lines(x,sin(2*x), col="red") # hinzufuegen von Linien
```



```
plot(x,x,type="l")
points(x,exp(x),type="l",col="red") # hinzufuegen von Punkten oder Linien
lines(x,log(x), col="blue") # hinzufuegen von Linien
```

Die erste Grafik legt den Bereich für die x- und y-Werte fest



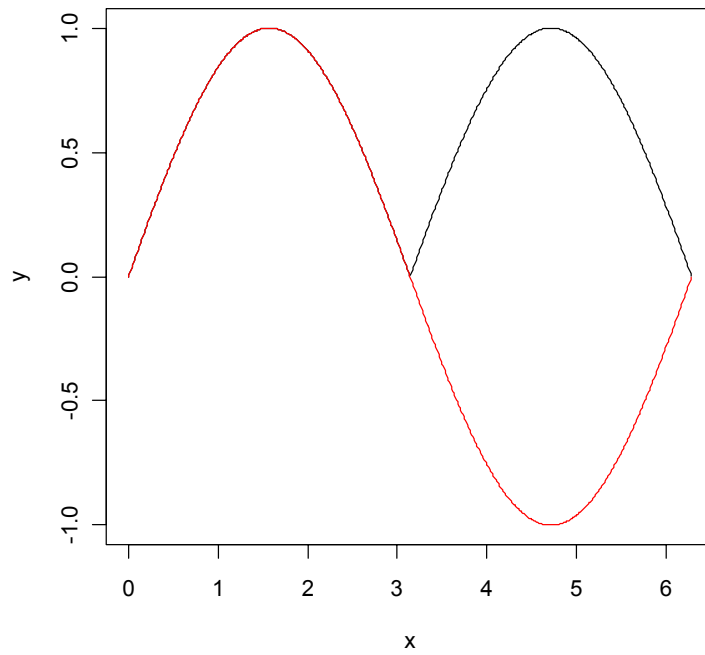
Festlegen des Bereichs für beide Funktionen:

Der Bereich für die y-Werte wird durch Minimum und Maximum der beiden Datenreihen bestimmt:

```
y<-abs(sin(x))  
z<-sin(x)
```

```
mi<-min(y,z)  
ma<-max(y,z)
```

```
plot(x,y,ylim=c(mi,ma),type="l")  
points(x,z,type="l",col="red")
```



3D-Grafiken in R

1. Erzeugen von Vektoren a und b:

```
a<-seq(-5,5,len=4)
b<-seq(-3,2,len=3)

> a
[1] -5.000000 -1.666667  1.666667  5.000000
> b
[1] -3.0 -0.5  2.0
```

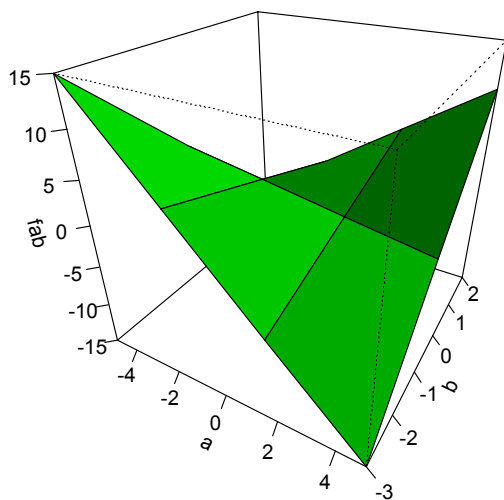
Fuer jede Kombination von Komponenten aus den Vektoren a und b wird ein
Wert der Funktion $f(a,b)=a*b$ berechnet und in der Matrix fab gespeichert:

```
fab<-outer(a,b,function(a,b) a*b)
```

```
> fab
      [,1]      [,2]      [,3]
[1,]    15  2.500000 -10.000000
[2,]     5  0.8333333 -3.333333
[3,]    -5 -0.8333333  3.333333
[4,]   -15 -2.500000  10.000000
```

3D-Darstellung:

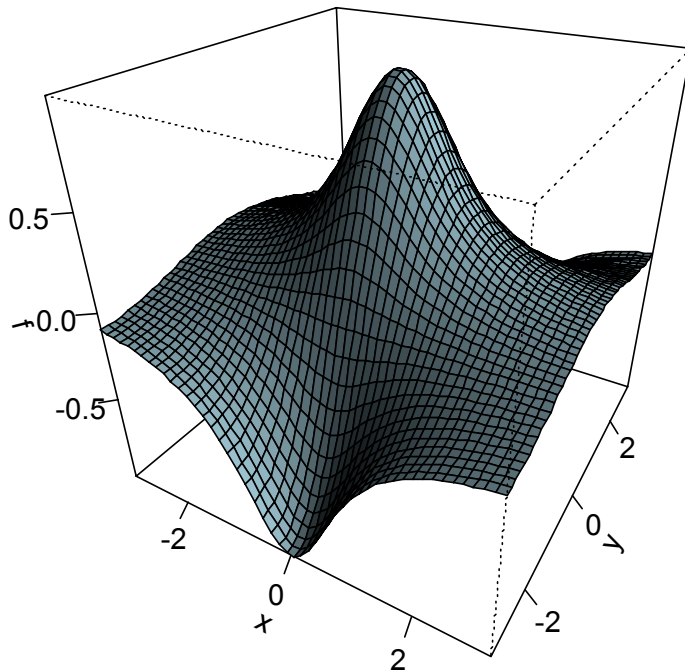
```
persp(a, b, fab, theta = 30, phi = 30, col = "green",
      ltheta = 120, shade = 0.75, ticktype = "detailed")
```



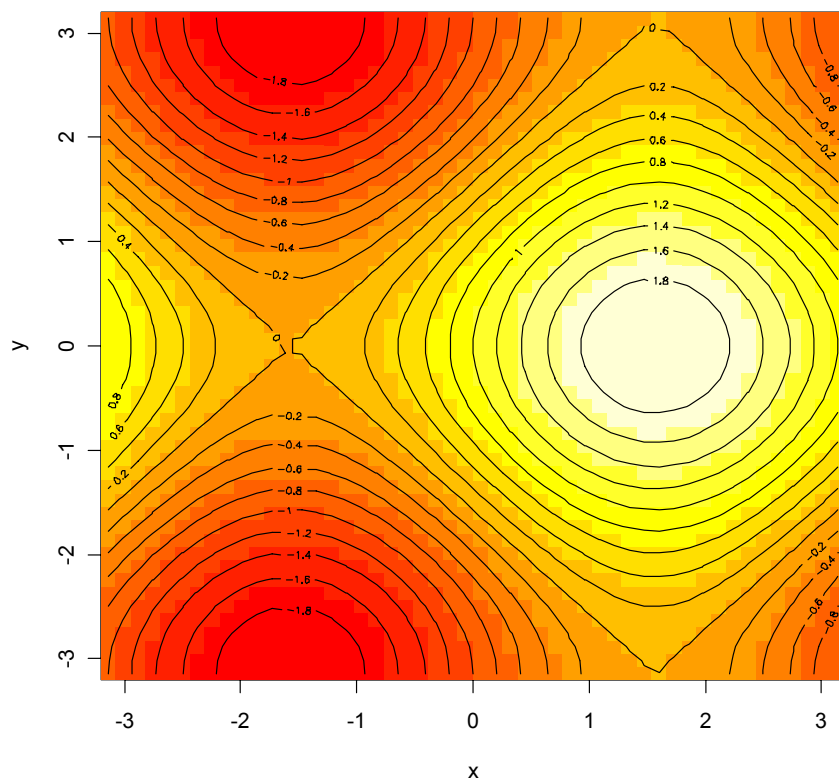
```
# dasselbe nochmal mit mehr Werten in x- und y-Richtung und einer anderen  
# Funktion:
```

```
x<-seq(-pi,pi,len=50)  
y<-x  
f<-outer(x,y,function(x,y) cos(y)/(1+x^2))
```

```
# 3D-Darstellung  
persp(x, y, f, theta = 30, phi = 30, col = "lightblue",  
+ ltheta = 120, shade = 0.75, ticktype = "detailed")
```



```
image(x,y,f) # Hoehen durch Farben darstellen  
contour(x,y,f,nlevels=25,add=T) # Hoehenlinien hinzufuegen
```



```

# More examples in demo(persp) !!
## -----

# (1) The Obligatory Mathematical surface.
# Rotated sinc function.

x <- seq(-10, 10, length= 30)
y <- x
f <- function(x,y) { r <- sqrt(x^2+y^2); 10 * sin(r)/r }
z <- outer(x, y, f)
z[is.na(z)] <- 1 # fehlende Werte auf 1 setzen

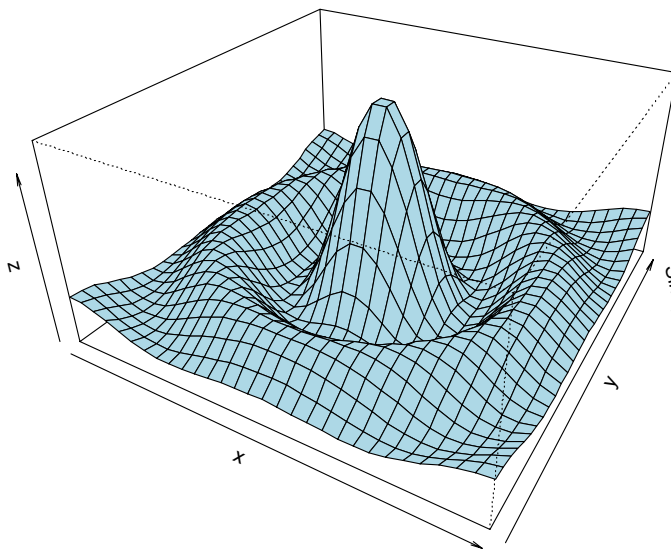
persp(x, y, z, theta = 30, phi = 30, expand = 0.5, col = "lightblue")
title("einfach")

x11()
persp(x, y, z, theta = 30, phi = 30, expand = 0.5, col = "green",
      ltheta = 120, shade = 0.75, ticktype = "detailed",
      xlab = "X", ylab = "Y", zlab = "Sinc( r )")

title("Nr. 2")

```

einfach



Nr. 2

