```
' © Prof.Dr.Ludwig Paditz, 14.12.2016
' LogiDReg(listx,listy,A0,b0,c0,d0,μ0,I)
' logist. Regress. mit Levenberg-Marquardt-Verfahren
' A=ln(a)
' verbundene Datenlisten listx,listy
' Startparameter A0,b0,c0,d0
' Steuerparameter μ0

' Zahl der Iterationen ... I
' mehrere Iterationsschritte (μ0 wird verändert)

local x,y,a,b,c,d,D,μ,s1,s2,s3,s4
local SP,SP11,SP12,SP13,SP14,SP22,SP23,SP24,SP33,SP34,
SP44,F,FA,r₁,r₂,r₃,r₄
DelVar x,y,a,b,c,d,D,μ,ε
ClrText

Define F(x,y,a,b,c,d)=y-c/(1+e^(a-b*x))-d

Define r₁(x,a,b,c,d)=diff(F(x,y,a,b,c,d),a)
Define r₂(x,a,b,c,d)=diff(F(x,y,a,b,c,d),b)
Define r₃(x,a,b,c,d)=diff(F(x,y,a,b,c,d),c)
Define r₄(x,a,b,c,d)=diff(F(x,y,a,b,c,d),d)

'Anmerkung r₄(x,a,b,c,d)=-1

A0⇒a
b0⇒b
c0⇒c
d0⇒d
μ0⇒μ
dim(listx)⇒D

0⇒i
do
i+1⇒i
'print i

'Berechne trn(F'(listx,a,b,c,d))*F'(listx,a,b,c,d)+μ^2*[[1,0
,0,0],[0,1,0,0],[0,0,1,0],[0,0,0,1]]

approx(sum((approx(r₁(listx,a,b,c,d)))^2)+μ^2)⇒SP11
approx(sum(approx(r₁(listx,a,b,c,d))*approx(r₂(listx,a,b,c,d
))))⇒SP12
approx(sum(approx(r₁(listx,a,b,c,d))*approx(r₃(listx,a,b,c,d
))))⇒SP13
approx(sum(approx(r₁(listx,a,b,c,d))*(-1)))⇒SP14

approx(sum((approx(r₂(listx,a,b,c,d)))^2)+μ^2)⇒SP22
approx(sum(approx(r₂(listx,a,b,c,d))*approx(r₃(listx,a,b,c,d
))))⇒SP23
approx(sum(approx(r₂(listx,a,b,c,d))*(-1)))⇒SP24
```

```
approx(sum((approx(r3(listx,a,b,c,d)))^2)+μ^2)⇒SP33
approx(sum(approx(r3(listx,a,b,c,d))*(-1)))⇒SP34

approx(D+μ^2)⇒SP44

approx([[SP11,SP12,SP13,SP14],[SP12,SP22,SP23,SP24],[
SP13,SP23,SP33,SP34],[SP14,SP24,SP34,SP44]])⇒SP


'Invertieren und Schätzung für Korrektur vecs

approx(F(listx,listy,a,b,c,d))⇒FA
approx(-SP^(-1)·[[sum(approx(r1(listx,a,b,c,d))·FA)],[sum
(approx(r2(listx,a,b,c,d))·FA)],[sum(approx(r3(listx,a,b,c,d
))·FA)],[sum((-1)·FA)]])⇒vecs


'Definiere Parametersteuerung ρμ0=((‖F(a,b,c,d)‖²-‖F(a+s1,b+s
2,c+s3,d+s4)‖²)/(‖F(a,b,c,d)‖²-‖F(a,b,c,d)+F'*[[s1],[s2],
[s3],[s4]]‖²))

vecs[1,1]⇒s1
vecs[2,1]⇒s2
vecs[3,1]⇒s3
vecs[4,1]⇒s4

'Nenner in ρμ0
approx((sum(FA^2)-sum((FA+s1*approx(r1(listx,a,b,c,d))+s2
*approx(r2(listx,a,b,c,d))+s3*approx(r3(listx,a,b,c,d))+s4*(
-1))^2)))⇒ε
If abs(ε)<10^(-12)
Then
print [i,μ,ρμ0,ε]
print vecab
print MSerr
stop
Ifend

approx((sum(FA^2)-sum(approx(F(listx,listy,a+s1,b+s2,c+s3,
d+s4))^2))/ε)⇒ρμ0

'Startparameter a,b,c,d geeignet!

'iterative Korrektur von a,b,c,d mithilfe von s1,s2,s3,s4 in
Abhängigkeit von ρμ0:
If ρμ0≥0.2
Then
approx([[a],[b],[c],[d]]+vecs)⇒vecab
vecab[1,1]⇒a
vecab[2,1]⇒b
vecab[3,1]⇒c
vecab[4,1]⇒d
Ifend
```

```
'μ-Anpassungsheuristik halbieren, belassen, verdoppeln
If ρμ0>0.8
Then
μ/2⇒μ
ElseIf ρμ0<0.2
Then
2*μ⇒μ
Ifend

approx(sum(FA^2)/(D-2))⇒MSerr
print [i,μ,ρμ0,ε]
print vecab
print MSerr

LpWhile i<I

return
```

```
' © Prof.Dr.Ludwig Paditz, 12.05.2017
' ArctDReg(listx,listy,a0,b0,c0,d0,μ0,I)
' arctan-Regress. mit Levenberg-Marquardt-Verfahren
' verbundene Datenlisten listx,listy
' Startparameter a0,b0,c0,d0
' Steuerparameter μ0

' Zahl der Iterationen ... I
' mehrere Iterationsschritte (μ0 wird verändert)

local x,y,a,b,c,d,D,μ,s1,s2,s3,s4
local SP,SP11,SP12,SP13,SP14,SP22,SP23,SP24,SP33,SP34,
SP44,F,FA,r₁,r₂,r₃,r₄
DelVar x,y,a,b,c,d,D,μ,ε
ClrText

Define F(x,y,a,b,c,d)=y-a*tan⁻¹(c*(x-b))-d

Define r₁(x,a,b,c,d)=diff(F(x,y,a,b,c,d),a)
Define r₂(x,a,b,c,d)=diff(F(x,y,a,b,c,d),b)
Define r₃(x,a,b,c,d)=diff(F(x,y,a,b,c,d),c)
Define r₄(x,a,b,c,d)=diff(F(x,y,a,b,c,d),d)

'Anmerkung r₄(x,a,b,c,d)=-1

a0⇒a
b0⇒b
c0⇒c
d0⇒d
μ0⇒μ
dim(listx)⇒D

0⇒i
do
i+1⇒i
'print i

'Berechne trn(F'(listx,a,b,c,d))*F'(listx,a,b,c,d)+μ^2*[[1,0
,0,0],[0,1,0,0],[0,0,1,0],[0,0,0,1]]

approx(sum((approx(r₁(listx,a,b,c,d)))^2)+μ^2)⇒SP11
approx(sum(approx(r₁(listx,a,b,c,d))*approx(r₂(listx,a,b,c,d
))))⇒SP12
approx(sum(approx(r₁(listx,a,b,c,d))*approx(r₃(listx,a,b,c,d
))))⇒SP13
approx(sum(approx(r₁(listx,a,b,c,d))*(-1)))⇒SP14

approx(sum((approx(r₂(listx,a,b,c,d)))^2)+μ^2)⇒SP22
approx(sum(approx(r₂(listx,a,b,c,d))*approx(r₃(listx,a,b,c,d
))))⇒SP23
approx(sum(approx(r₂(listx,a,b,c,d))*(-1)))⇒SP24

approx(sum((approx(r₃(listx,a,b,c,d)))^2)+μ^2)⇒SP33
```

approx(sum(approx(r_3(listx, a, b, c, d))*(−1)))⇒SP34

approx(D+μ^2)⇒SP44

approx([[SP11, SP12, SP13, SP14], [SP12, SP22, SP23, SP24], [SP13, SP23, SP33, SP34], [SP14, SP24, SP34, SP44]])⇒SP

'Invertieren und Schätzung für Korrektur vecs

approx(F(listx, listy, a, b, c, d))⇒FA
approx(−SP^(−1)·[[sum(approx(r_1(listx, a, b, c, d))·FA)], [sum(approx(r_2(listx, a, b, c, d))·FA)], [sum(approx(r_3(listx, a, b, c, d))·FA)], [sum((−1)·FA)]])⇒vecs

'Definiere Parametersteuerung $\rho\mu_0$=((‖F(a, b, c, d)‖²−‖F(a+s1, b+s2, c+s3, d+s4)‖²)/(‖F(a, b, c, d)‖²−‖F(a, b, c, d)+F'*[[s1], [s2], [s3], [s4]]‖²))

vecs[1, 1]⇒s1
vecs[2, 1]⇒s2
vecs[3, 1]⇒s3
vecs[4, 1]⇒s4

'Nenner in $\rho\mu_0$
approx((sum(FA^2)−sum((FA+s1*approx(r_1(listx, a, b, c, d))+s2*approx(r_2(listx, a, b, c, d))+s3*approx(r_3(listx, a, b, c, d))+s4*(−1))^2)))⇒ε
If abs(ε)<10^(−12)
Then
print [i, μ, $\rho\mu_0$, ε]
print vecab
print MSerr
stop
Ifend

approx((sum(FA^2)−sum(approx(F(listx, listy, a+s1, b+s2, c+s3, d+s4))^2))/ε)⇒$\rho\mu_0$

'Startparameter a, b, c, d geeignet!

'iterative Korrektur von a, b, c, d mithilfe von s1, s2, s3, s4 in Abhängigkeit von $\rho\mu_0$:
If $\rho\mu_0$≥0.2
Then
approx([[a], [b], [c], [d]]+vecs)⇒vecab
vecab[1, 1]⇒a
vecab[2, 1]⇒b
vecab[3, 1]⇒c
vecab[4, 1]⇒d
Ifend

```
'μ-Anpassungsheuristik halbieren, belassen, verdoppeln
If ρμ₀>0.8
Then
μ/2⇒μ
ElseIf ρμ₀<0.2
Then
2*μ⇒μ
Ifend

approx(sum(FA^2)/(D-2))⇒MSerr
print [i,μ,ρμ₀,ε]
print vecab
print MSerr

LpWhile i<I

return
```

```
' © Prof.Dr.Ludwig Paditz, 17.05.2017
' PowRegLM(listx,listy,a0,b0,μ0,I)
' Potenz-Regress. mit Levenberg-Marquardt-Verfahren

' verbundene Datenlisten listx,listy
' Startparameter a0,b0
' Steuerparameter μ0

' Zahl der Iterationen ... I
' mehrere Iterationsschritte (μ0 wird verändert)

local x,y,a,b,D,μ,s1,s2
local SP,SP11,SP12,SP22,F,FA,r₁,r₂
DelVar x,y,a,b,D,μ,ε
ClrText

Define F(x,y,a,b)=y-a*x^b

Define r₁(x,a,b)=diff(F(x,y,a,b),a)
Define r₂(x,a,b)=diff(F(x,y,a,b),b)

a0⇒a
b0⇒b
μ0⇒μ
dim(listx)⇒D


0⇒i
do
i+1⇒i
'print i

'Berechne trn(F'(listx,a,b))*F'(listx,a,b)+μ^2*[[1,0],[0,1]
]

approx(sum((approx(r₁(listx,a,b)))^2)+μ^2)⇒SP11
approx(sum(approx(r₁(listx,a,b))*approx(r₂(listx,a,b))))⇒SP1
2

approx(sum((approx(r₂(listx,a,b)))^2)+μ^2)⇒SP22

approx([[SP11,SP12],[SP12,SP22]])⇒SP


'Invertieren und Schätzung für Korrektur vecs

approx(F(listx,listy,a,b))⇒FA
approx(-SP^(-1)·[[sum(approx(r₁(listx,a,b))·FA)],[sum(appr
ox(r₂(listx,a,b))·FA)]])⇒vecs


'Definiere Parametersteuerung ρμ₀=((‖F(a,b)‖²-‖F(a+s1,b+s2)‖²
)/(‖F(a,b)‖²-‖F(a,b)+F'*[[s1],[s2]]‖²))
```

```
vecs[1,1]⇒s1
vecs[2,1]⇒s2

'Nenner in ρμ₀
approx((sum(FA^2)-sum((FA+s1*approx(r₁(listx,a,b))+s2*app
rox(r₂(listx,a,b)))^2)))⇒ε
If abs(ε)<10^(-12)
Then
print [i,μ,ρμ₀,ε]
print vecab
print MSerr
stop
Ifend

approx((sum(FA^2)-sum(approx(F(listx,listy,a+s1,b+s2))^2))
/ε)⇒ρμ₀

'Startparameter a,b geeignet!

'iterative Korrektur von a,b mithilfe von s1,s2 in Abhängigkeit
 von ρμ₀:
If ρμ₀≥0.2
Then
approx([[a],[b]]+vecs)⇒vecab
vecab[1,1]⇒a
vecab[2,1]⇒b
Ifend

'μ-Anpassungsheuristik halbieren, belassen, verdoppeln
If ρμ₀>0.8
Then
μ/2⇒μ
ElseIf ρμ₀<0.2
Then
2*μ⇒μ
Ifend

approx(sum(FA^2)/(D-2))⇒MSerr
print [i,μ,ρμ₀,ε]
print vecab
print MSerr

LpWhile i<I

return
```