Blatt 4

Johannes Burr

25 Mai 2020

```
source("po20_blatt04_code.R")
```

Attraktionsgebiete

Attraktionsgebiete sidn definiert als die Region, von der aus ein lokales Optimum erreichbar sind. Im folgenden wird die Annäherung im Zielfunktionsraum betrachten. Ein lokales Minimum wird also nicht durch seine Stelle, sondern durch seinen Zielfunktionswert eindeutig definiert.

Vorgehen

Zum Ermitteln der Attraktionsgebiete wird die entsprechende Optimierungsfunktion von allen Gitterpunkten aus gestartet und die korrespondierenden Werte in einem Dataframe gespeichert.

Danach werden übereinstimmende lokale Optima identifiziert, indem zuerst der Dataframe nach der fx-Spalte geordnet wird. Über diese Spalte wird dann ein übereinstimmender Marker notiert, wenn die Funktionswerte identisch sind.

Kompasssuche

```
dat = data.frame(matrix(ncol=6,nrow=0))
colnames(dat) = c("X1", "X2", "X1_opt", "X2_opt", "Fx","Marker")

for (i in 1:2601){
  dat[i,1] = grid[i,1]
  dat[i,2] = grid[i,2]
  dat[i,3] = round(kompasssuche3(f=f,x0=as.numeric(grid[i,]))$x[1],
                           digits=2)
  dat[i,4] = round(kompasssuche3(f=f,x0=as.numeric(grid[i,]))$x[2],
                         digits=2)
  dat[i,5] = round(kompasssuche3(f=f,x0=as.numeric(grid[i,]))$fx,
                           digits=2)
}

data = dat[order(dat[5]),]


mark=1
```

```
for (j in 1:2600){
  if (data[j,5]==data[(j+1),5]){
    data[j,6] <- mark
    data[j+1,6] <- mark
  }
  else {

    data[j,6]=mark
    mark = mark +1
    }
}
```

Wahrscheinlichkeiten

Nun werden die Wahrscheinlichkeiten des Auffindes des jeweiligen lokalen Minimums geschätzt. Dies wird im diskreten Raum geschehen. Dh die Anzahl der Gitterpunkte, von denen aus das Optimum erreicht wurde, wird durch die Gesamtzahl geteilt. Dies approximiert den Flächeninhalt im 2D-Plot.

```
# get the points with the same Marker = same f(x)
ks_1 = data[data$Marker==1,]
ks_2 = data[data$Marker==2,]
ks_3 = data[data$Marker==3,]
ks_4 = data[data$Marker==4,]

#divide the amount with the total number
p_ks1 = dim(ks_1)[1] / 2601
p_ks2 = dim(ks_2)[1] / 2601
p_ks3 = dim(ks_3)[1] / 2601
p_ks4 = dim(ks_4)[1] / 2601
```

f(x) Wahrscheinlichkeit -16.86 0.331 -3.99 0.313 15.97 0.150 34.15 0.207

Damit korrespondieren die geschätzten Wahrscheinlichkeiten ungefähr mit der optischen Einschätzung der Flächeninhalten. Diese unterscheiden sich von lokalem Optimum zum nächsten nur relativ wenig. Scheinbar bleibt die Kompasssuche recht schnell im lokalen Minimum stecken.

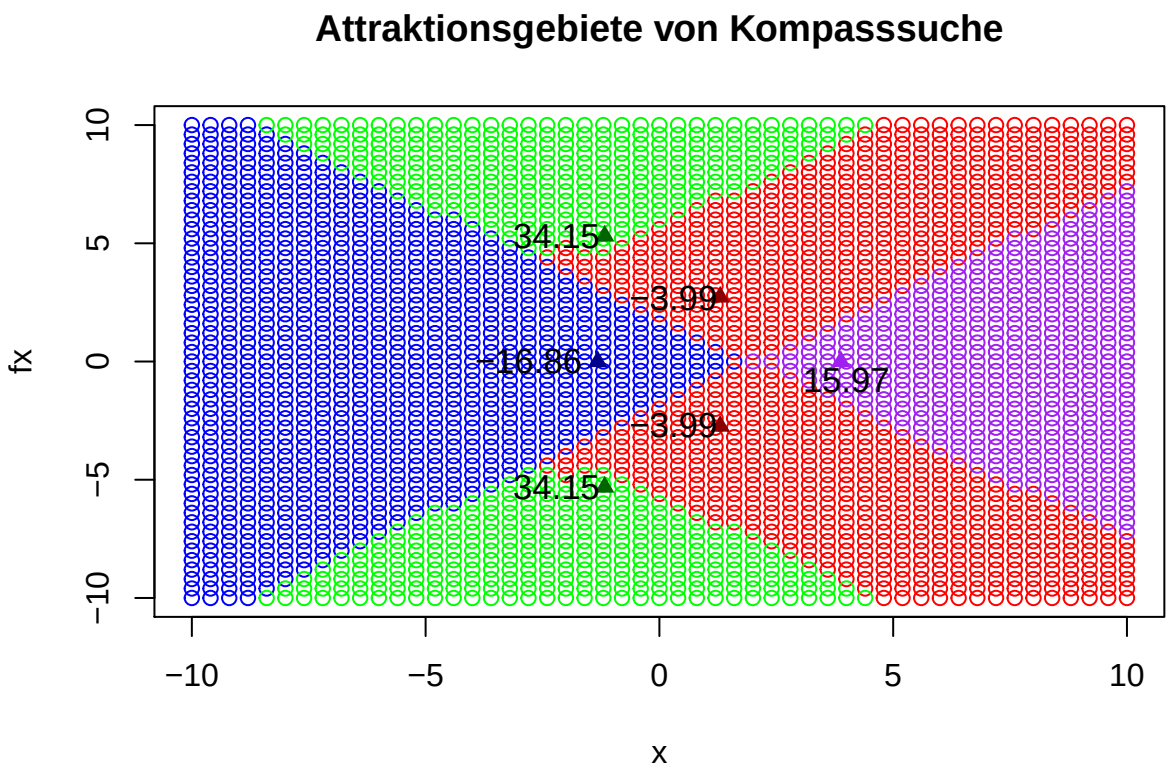
Plot

```
plot(x = 1, xlim = c(-10, 10), ylim = c(-10, 10), xlab = "x", ylab = "fx",
     main =("Attraktionsgebiete von Kompasssuche"),
     type = "n")
points(x=ks_1$X1, y=ks_1$X2, col="blue")
points(x=ks_2$X1,y=ks_2$X2, col="red")
points(x=ks_3$X1, y=ks_3$X2, col="purple")
points(x=ks_4$X1, y=ks_4$X2, col="green")

points(x=-1.32,y=0,col='dark blue',pch=17)
points(x=1.3,y=-2.73,col='dark red',pch=17)
points(x=1.3,y=2.73,col='dark red',pch=17)
points(x=3.88,y=0.00,col='purple',pch=17)
points(x=-1.17,y=-5.31,col='dark green',pch=17)
```

```
points(x=-1.17,y=5.31,col='dark green',pch=17)

text(-2.8, 0, "-16.86", cex=1.1)
text(0.3, -2.7, "-3.99", cex=1.1)
text(0.3, 2.7, "-3.99", cex=1.1)
text(-2.2, -5.3, "34.15", cex=1.1)
text(-2.2, 5.3, "34.15", cex=1.1)
text(4, -0.8, "15.97", cex=1.1)
```



3D-Plot

```
plot3d(
  x=ks_1$X1,
  y=ks_1$X2,
  z=apply(ks_1[1:2],1,f),
  size=5,
  col="blue",

  xlab="x",
  ylab="y",
  zlab="f(x,y)"
)
```

```
plot3d(
  x=ks_2$X1,
  y=ks_2$X2,
  z=apply(ks_2[1:2],1,f),
  size=5,
  col="red",
  add=TRUE,

)
plot3d(
  x=ks_3$X1,
  y=ks_3$X2,
  z=apply(ks_3[1:2],1,f),
  size=5,
  col="purple",
  add=TRUE,
)

plot3d(
  x=ks_4$X1,
  y=ks_4$X2,
  z=apply(ks_4[1:2],1,f),
  size=5,
  col="green",
  add=TRUE,
)


points3d(x=-1.32,y=0,-16.86,col='dark blue',
         size=10)
points3d(x=1.3,y=-2.73,-3.99,col='dark red',
         size=10)
points3d(x=1.3,y=2.73,-3.99,col='dark red',
         size=10)
points3d(x=3.88,y=0.00,15.97,col='purple',
         size=10)
points3d(x=-1.17,y=-5.31,34.15,col='dark green',
         size=10)
points3d(x=-1.17,y=5.31,34.15,col='dark green',
         size=10)
```

Die Attraktionsgebiete sind insgesamt zusammenhängend und dadurch relativ nachvollziehbar. So ziehen sich recht klare diagonale Grenzen durch die Gitterzone, wodurch die Attraktionsgebiete getrennt sind. Durch die Symmetrie der Funktion entstehen außerdem zwei lokale Minima an den Rändern (grün) mit exakt gleichen Funktionswerten, die deshalb hier als gleich verstanden werden. Diese klare Struktur entsteht durch die rigide Art, mit der die Kompasssuche die Richtung, der sie im jeweiligen Schritt folgen wird, wählt.

BFGS

```
bfg = data.frame(matrix(ncol=6,nrow=0))
colnames(bfg) = c("X1", "X2", "X1_opt", "X2_opt", "Fx","Marker")
```

```
for (i in 1:2601){
  bfg[i,1] = grid[i,1]
  bfg[i,2] = grid[i,2]
  bfg[i,3] = round(optim(method="BFGS",fn=f,par=as.numeric(grid[i,]))$par[1],
                   digits=2)
  bfg[i,4] = round(optim(method="BFGS",fn=f,par=as.numeric(grid[i,]))$par[2],
                   digits=2)
  bfg[i,5] = round(optim(method="BFGS",fn=f,par=as.numeric(grid[i,]))$value,
                   digits=2)
}

bfgs = bfg[order(bfg[5]),]


mark=1

for (j in 1:2600){
  if (bfgs[j,5]==bfgs[(j+1),5]){
    bfgs[j,6] <- mark
    bfgs[j+1,6] <- mark
  }
  else {

    bfgs[j,6]=mark
    mark = mark +1
  }
}
```

Schätzen der Wahrscheinlichkeiten

```
bf_op1 = bfgs[bfgs$Marker == 1,]
bf_op2 = bfgs[bfgs$Marker == 2,]
bf_op3 = bfgs[bfgs$Marker == 3,]
bf_op4 = bfgs[bfgs$Marker == 4,]

(p_bf1 = dim(bf_op1)[1] / 2601)
```

```
## [1] 0.4901961
```

```
(p_bf2 = dim(bf_op2)[1] / 2601)
```

```
## [1] 0.4306036
```

```
(p_bf3 = dim(bf_op3)[1] / 2601)
```

```
## [1] 0.05074971
```

```r
(p_bf4 = dim(bf_op4)[1] / 2601)
```

```
## [1] 0.0284506
```

```r
p_bf4
```

```
## [1] 0.0284506
```

Opt Wahrscheinlichkeit -16.86 0.490 -3.99 0.431 15.97 0.051 34.15 0.028

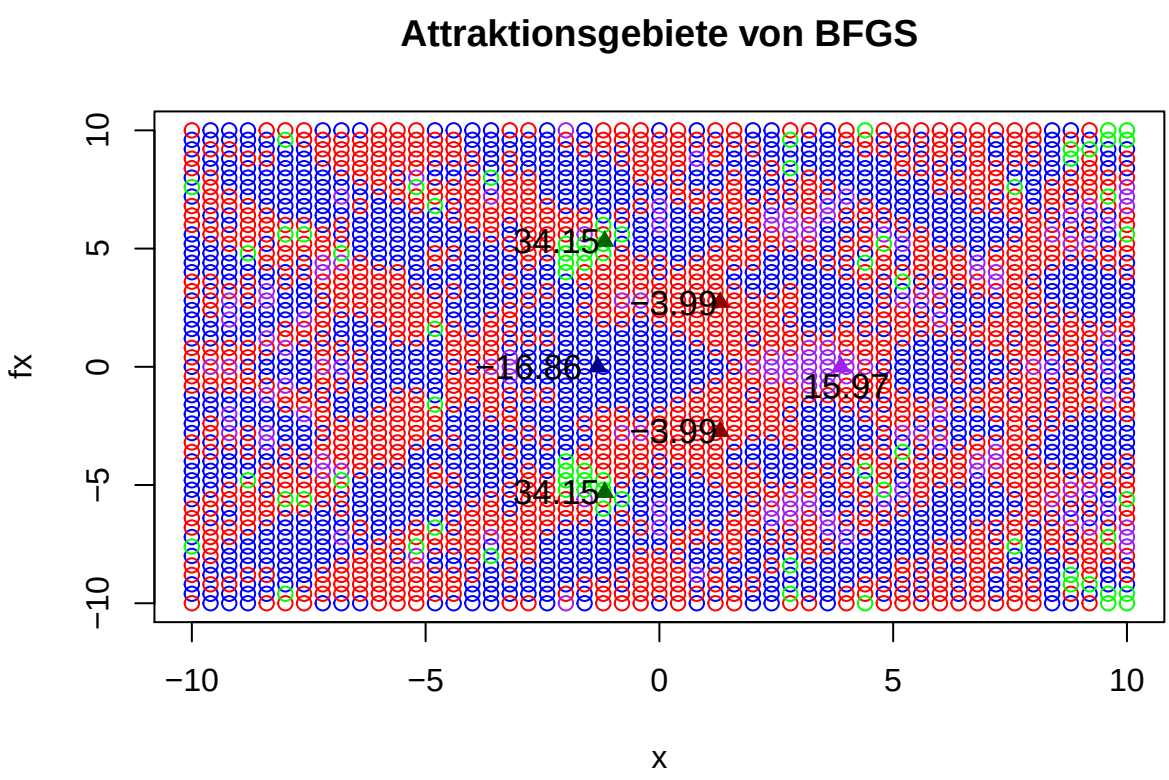
Somit findet BFGS grundsätzlich die gleichen Optima wie die Kompasssuche, allerdings mit höherer Warscheinlichkeit die beiden besseren, sowie insbesondere das globale Minimum mit p=0.49.

Plot

```r
plot(x = 1, xlim = c(-10, 10), ylim = c(-10, 10), xlab = "x", ylab = "fx",
     main =("Attraktionsgebiete von BFGS"),
     type = "n")
points(x=bf_op1$X1, y=bf_op1$X2, col="blue")
points(x=bf_op2$X1,y=bf_op2$X2, col="red")
points(x=bf_op3$X1, y=bf_op3$X2, col="purple")
points(x=bf_op4$X1, y=bf_op4$X2, col="green")

points(x=-1.32,y=0,col='dark blue',pch=17)
points(x=1.3,y=-2.73,col='dark red',pch=17)
points(x=1.3,y=2.73,col='dark red',pch=17)
points(x=3.88,y=0.00,col='purple',pch=17)
points(x=-1.17,y=-5.31,col='dark green',pch=17)
points(x=-1.17,y=5.31,col='dark green',pch=17)

text(-2.8, 0, "-16.86", cex=1.1)
text(0.3, -2.7, "-3.99", cex=1.1)
text(0.3, 2.7, "-3.99", cex=1.1)
text(-2.2, -5.3, "34.15", cex=1.1)
text(-2.2, 5.3, "34.15", cex=1.1)
text(4, -0.8, "15.97", cex=1.1)
```



3D-Plots

```
plot3d(
  x=bf_op1$X1,
  y=bf_op1$X2,
  z=apply(bf_op1[1:2],1,f),
  size=5,
  col="blue",
  main="Attraktionsgebiete von BFGS",
  xlab="x",
  ylab="y",
  zlab="f(x,y)"
)

plot3d(
  x=bf_op2$X1,
  y=bf_op2$X2,
  z=apply(bf_op2[1:2],1,f),
  size=5,
  col="red",
  add=TRUE,

)
plot3d(
```

```
  x=bf_op3$X1,
  y=bf_op3$X2,
  z=apply(bf_op3[1:2],1,f),
  size=5,
  col="purple",
  add=TRUE,
)

plot3d(
  x=bf_op4$X1,
  y=bf_op4$X2,
  z=apply(bf_op4[1:2],1,f),
  size=5,
  col="green",
  add=TRUE,
)


points3d(x=-1.32,y=0,-16.86,col='dark blue',
         size=10)
points3d(x=1.3,y=-2.73,-3.99,col='dark red',
         size=10)
points3d(x=1.3,y=2.73,-3.99,col='dark red',
         size=10)
points3d(x=3.88,y=0.00,15.97,col='purple',
         size=10)
points3d(x=-1.17,y=-5.31,34.15,col='dark green',
         size=10)
points3d(x=-1.17,y=5.31,34.15,col='dark green',
         size=10)
```

Auch hier stimmt die Grafik mit der intuitiven Interpretation der Auffindenswahrscheinlichkeiten überein: BFGS findet meistens das globale Minimum (blau) oder die zweit beste Lösung (rot), aber nurnoch sehr selten eine der anderen lokalen Optima. Die Interpretation der Struktur fällt hier aber deutlich schwieriger, weil die Attraktionsgebiete viel zerlüfteter und unzusammenhängend sind. Teilweise gibt es einen einzelnen grünen Gitterpunkt inmitten eines roten Clusters. Da ich nicht weiß, wie optims BFGS im Inneren funktioniert, kann ich nicht erklären, woran das liegt.

L-BFGS-B

```
lb = data.frame(matrix(ncol=6,nrow=0))
colnames(lb) = c("X1", "X2", "X1_opt", "X2_opt", "Fx","Marker")

for (i in 1:2601){
  lb[i,1] = grid[i,1]
  lb[i,2] = grid[i,2]
  lb[i,3] = round(optim(method="L-BFGS-B",fn=f,par=as.numeric(grid[i,]))$par[1],digits=2)
  lb[i,4] = round(optim(method="L-BFGS-B",fn=f,par=as.numeric(grid[i,]))$par[2],digits=2)
  lb[i,5] = round(optim(method="L-BFGS-B",fn=f,par=as.numeric(grid[i,]))$value,digits=2)
}
```

```r
#order with f(x)
lbfgs = lb[order(lb[5]),]


#mark rows with equal f(x)
mark=1

for (j in 1:2600){
  if (lbfgs[j,5]==lbfgs[(j+1),5]){
    lbfgs[j,6] <- mark
    lbfgs[j+1,6] <- mark
  }
  else {

    lbfgs[j,6]=mark
    mark = mark +1
  }
}
```

```r
lf_op1 = lbfgs[lbfgs$Marker==1,]
lf_op2 = lbfgs[lbfgs$Marker==2,]
lf_op3 = lbfgs[lbfgs$Marker==3,]
lf_op4 = lbfgs[lbfgs$Marker==4,]


(p_lf1 = dim(lf_op1)[1] /2601)
```

```
## [1] 0.4448289
```

```r
(p_lf2 = dim(lf_op2)[1] /2601)
```

```
## [1] 0.4006151
```

```r
(p_lf3 = dim(lf_op3)[1] /2601)
```

```
## [1] 0.07381776
```

```r
(p_lf4 = dim(lf_op4)[1] /2601)
```

```
## [1] 0.08073818
```

```r
View(lbfgs)
```

Optimum Wahrscheinlichkeit

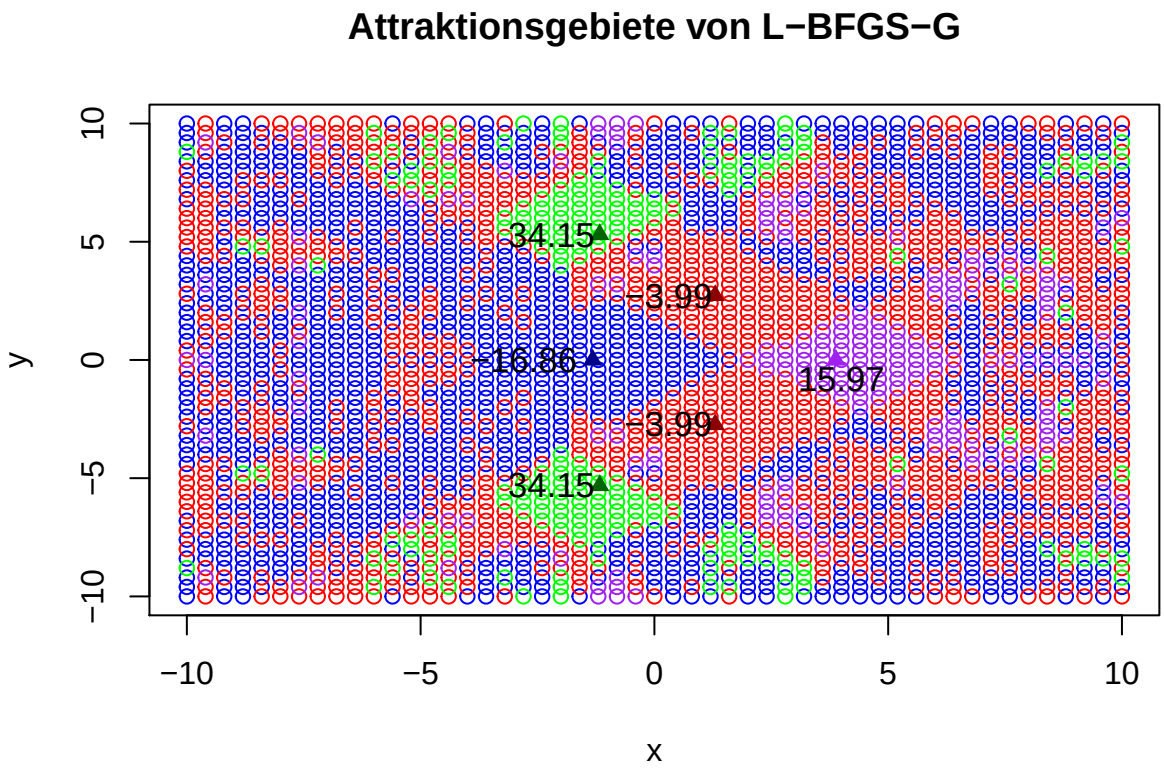
-16.86 0.445 -3.99 0.401 15.97 0.074 34.15 0.081

Wieder die gleichen Optima wie zuvor, mit ähnlichen Wahrscheinlichkeiten wie BFGS, allerdings mit etwas geringerer Wahrscheinlichkeit fürs globale Minimum.

```r
plot(x = 1, xlim = c(-10, 10), ylim = c(-10, 10), xlab = "x", ylab = "y",
     main =("Attraktionsgebiete von L-BFGS-G"),
     type = "n")
points(x=lf_op1$X1, y=lf_op1$X2, col="blue")
points(x=lf_op2$X1,y=lf_op2$X2, col="red")
points(x=lf_op3$X1, y=lf_op3$X2, col="purple")
points(x=lf_op4$X1, y=lf_op4$X2, col="green")

points(x=-1.32,y=0,col='dark blue',pch=17)
points(x=1.3,y=-2.73,col='dark red',pch=17)
points(x=1.3,y=2.73,col='dark red',pch=17)
points(x=3.88,y=0.00,col='purple',pch=17)
points(x=-1.17,y=-5.31,col='dark green',pch=17)
points(x=-1.17,y=5.31,col='dark green',pch=17)

text(-2.8, 0, "-16.86", cex=1.1)
text(0.3, -2.7, "-3.99", cex=1.1)
text(0.3, 2.7, "-3.99", cex=1.1)
text(-2.2, -5.3, "34.15", cex=1.1)
text(-2.2, 5.3, "34.15", cex=1.1)
text(4, -0.8, "15.97", cex=1.1)
```



3D-Plots

```
plot3d(
  x=lf_op1$X1,
  y=lf_op1$X2,
  z=apply(lf_op1[1:2],1,f),
  size=5,
  col="blue",
  main="Attraktionsgebiete von L-BFGS-B",
  xlab="x",
  ylab="y",
  zlab="f(x,y)"
)

plot3d(
  x=lf_op2$X1,
  y=lf_op2$X2,
  z=apply(lf_op2[1:2],1,f),
  size=5,
  col="red",
  add=TRUE,

)
plot3d(
  x=lf_op3$X1,
  y=lf_op3$X2,
  z=apply(lf_op3[1:2],1,f),
  size=5,
  col="purple",
  add=TRUE,
)

plot3d(
  x=lf_op4$X1,
  y=lf_op4$X2,
  z=apply(lf_op4[1:2],1,f),
  size=5,
  col="green",
  add=TRUE,
)


points3d(x=-1.32,y=0,-16.86,col='dark blue',size=10)
points3d(x=1.3,y=-2.73,-3.99,col='dark red',size=10)
points3d(x=1.3,y=2.73,-3.99,col='dark red',
         size=10)
points3d(x=3.88,y=0.00,15.97,col='purple',
         size=10)
points3d(x=-1.17,y=-5.31,34.15,col='dark green',size=10)
points3d(x=-1.17,y=5.31,34.15,col='dark green',size=10)
```

Hier sind die Attraktionsgebiete etwas klarer umrissen als bei der Methode BFGS. So sind zum Beispiel um die jeweiligen lokalen Optima herum recht klare Vierecke zu erkennen, von wo aus immer das jweilige Minimum

erreicht wird. Das könnte daran liegen, dass der Algorithmus mit deutlich höherer Wahrscheinlichkeit in ein lokales Minimum läuft, wenn er in dessen Nähe ist.

Fazit

Das Globale Minimum mit f(x)=-16.86 findet die Methode BFGS am wahrscheinlichsten (p=0.49). Wird unbedingt die beste Lösung gesucht, sollte also BFGS verwendet werden. Reicht es aus, eine gute Lösung zu finden, könnte aber auch L-BFGS-G Vorteile haben, weil es mit geringerer Wahrscheinlichkeit in einem der beiden sehr schlechten lokalen Optima stecken bleibt (0.079 vs. 0.155)