

```
require(data.table)

require(geoR)
require(RandomFields)

require(truncnorm)

require(foreach)

require(parallel)
require(doSNOW)

dimen <- 600 #Anzal Agenten 600 für endgültige Simulation

ni <- dimen^2

a <- 0.001

#####Version with Parallelisation#####

fill <-function (){
  erzeugeNachbarn <- function(dimensions = c(10, 10), n_health = 2){
```

```
health <- colours()[1:n_health]
perc_empty <- 0.2

n_homes = prod(dimensions)

count_agents <- floor(n_homes * (1 - perc_empty))

health <- c("empty", health)

probabilities <- c(perc_empty, rep((1 - perc_empty)/(length(health) - 1),
                                times = length(health) - 1))

Nachbarn <-< data.table(id = 1:prod(dimensions), #Id für Feld auf Grid
                      Id_Agenten = 1:prod(dimensions), #Id für Agenten
                      x = rep(1:dimensions[1],
                              dimensions[2]),
                      y = rep(1:dimensions[2],
                              each = dimensions[1]),
                      Stressor = rep(NA, prod(dimensions)),
                      health = sample(x = health, #Health ist nich Health....dient nur zur Visualisierung des
plots
                                   size = n_homes,
                                   prob = probabilities,
                                   replace = TRUE),
                      Diff = 0,
                      bewohnt = rep(NA, prod(dimensions)),
                      age = rep(NA, prod(dimensions)),
                      gesundheit = rep(NA, prod(dimensions)),
```

```
        Verhalten = rep(NA, prod(dimensions)),
        Verhaltensunterschied=0)
}

set.seed(42^2)

erzeugeNachbarn(dimensions = c(dimen, dimen), n_health = 2)

n<- nrow(Nachbarn)

model1 <- RMexp(var=0.60, scale=6) + RMnugget(var=0.4) # Die Werte schwanken bei Varianz 60
zwischen -15 und 10, bei Varianz 10 zwischen -6 und 6

simu1a <- RFsimulate(model1, x= Nachbarn[,x], y=Nachbarn[,y])

model2 <- RMexp(var=0.6*(17^2), scale=6) + RMnugget(var=0.4*(17^2))

simu2a <- RFsimulate(model2, x= Nachbarn[,x], y=Nachbarn[,y])

Nachbarn[, age:= ifelse( Nachbarn$health == 'empty', NA, round(Alter1))] #Alter ist zufällig verteilt,
mit SOEP Daten hinterlegt und gerundet
```

```
#bewohnt

Nachbarn[, bewohnt := ifelse ( Nachbarn$health == 'empty', 0, 1)] #Indikator ob Haus leer ist oder
nicht

Stress <- rnorm(ni,0,0.2)

Nachbarn[,Stressor:= Stress+as.vector(simu1a)]

verhalten2 <- rtruncnorm(n=ni, a=1, b=99, mean=50, sd=17)

Nachbarn[,verhalten:= ifelse( Nachbarn$health == 'empty', NA, verhalten2+as.vector(simu2a) )]
#+as.vector(simu2a)

Nachbarn[,verhalten:= ifelse(Nachbarn$verhalten < 0,0, Nachbarn$verhalten+0)]

alsgesundheit1<- rtruncnorm(n=ni, a=10.7, b=74.87, mean=49.85, sd=10.18762 ) # SOEP Daten für
Pcs eingefügt

Nachbarn[,gesundheit:= ifelse( Nachbarn$health == 'empty', NA, alsgesundheit1)]
}

system.time(fill())
```

```
CS_Anpassung <- function(n=1){
```

```
  for(CS_Anpassung in 1:n) {
```

```
    f<- Nachbarn[,gesundheit]+(-0.26863)+Nachbarn[,Stressor*1]+runif(1,-0.5,0.5);
```

```
    Nachbarn[,gesundheit:= f];
```

```
    Nachbarn[,gesundheit:= ((Nachbarn[,gesundheit]-  
mean(na.omit(Nachbarn$gesundheit)))/sd(na.omit(Nachbarn$gesundheit)))]
```

```
  }
```

```
  return(Nachbarn)
```

```
}
```

```
iterate <- function(n = 1){
```

```
  set.seed(NULL)
```

```
  how_many_neighbors55 <-function(x_1=Nachbarn[i,x], y_1=Nachbarn[i,y]){ #ca 10 sek bei 1600  
Agenten, bildet mittelwert der Nachbarn
```

```
  ver<-0
```

```
  count <-0
```

```
  rand <- dimen
```

```
# summe <- 0

if(Nachbarn[i,8]==1){

  if(Nachbarn[i,3] > 1) {

    ifelse(Nachbarn[x==(x_1-1)& y==y_1 & health !="empty"],( (ver<- ver+ Nachbarn[x==(x_1-1)&
y==y_1,verhalten]) &( count <- count+1)) ,count <- count+0 )

    if (Nachbarn[i,4] > 1)

      ifelse(Nachbarn[x==(x_1-1)& y==(y_1-1)& health !="empty"], ((ver<- ver+Nachbarn[x==(x_1-1)&
y==(y_1-1),verhalten])&( count <- count+1)),count <- count+0 )

      if (Nachbarn[i,4] < rand)

        ifelse(Nachbarn[x==(x_1-1)& y==(y_1+1)& health !="empty"], ((ver<- ver+Nachbarn[x==(x_1-
1)& y==(y_1+1),verhalten])&( count <- count+1)),count <- count+0 )

    }

    if(Nachbarn[i,3] < rand) {

      ifelse(Nachbarn[x==(x_1+1)& y==y_1 & health !="empty"], ((ver<- ver+Nachbarn[x==(x_1+1)&
y==y_1,verhalten])&( count <- count+1)),count <- count+0 )

      if (Nachbarn[i,4] > 1)

        ifelse(Nachbarn[x==(x_1+1)& y==(y_1-1)& health!="empty"], ( ( ver<-
ver+Nachbarn[x==(x_1+1)& y==(y_1-1),verhalten])&( count <- count+1)),count <- count+0 )

        if (Nachbarn[i,4] < rand)

          ifelse(Nachbarn[x==(x_1+1)& y==(y_1+1)& health !="empty"], ( (ver<-
ver+Nachbarn[x==(x_1+1)& y==(y_1+1),verhalten])&( count <- count+1)),count <- count+0 )

    }

    if(Nachbarn[i,4] > 1) ifelse(Nachbarn[x==x_1& y==(y_1-1)& health!="empty"], ((ver<-
ver+Nachbarn[x==x_1& y==(y_1-1),verhalten])&( count <- count+1)),count <- count+0 )

    if(Nachbarn[i,4] < rand) ifelse(Nachbarn[x==x_1& y==(y_1+1)& health !="empty"], ((ver<-
ver+Nachbarn[x==x_1& y==(y_1+1),verhalten])&( count <- count+1)),count <- count+0 )

  }

  if(ver ==0 & count==0){mean_verhalten <- 0} else{mean_verhalten <- ver/count}
```

```
for (iterate in 1:n){

  Nachbarn[, age:= age +1]
  emptyIDs <- Nachbarn[health == "empty", id]
  notemptyIDs <- Nachbarn[health != "empty",id]
  prozent_umzug <- 0.01
  z <- nrow(Nachbarn)
  z <- na.omit(z)
  u <- sample(notemptyIDs, (prozent_umzug*z))

  oldIDs <- Nachbarn[u, id]

  newIDs <- sample(x = c(emptyIDs, oldIDs),
    size = length(oldIDs),
    replace = F)

  transition <- data.table(origin = oldIDs,
    oldhealth = Nachbarn[id %in% oldIDs, health],
    oldage = Nachbarn[id %in% oldIDs, age],
    oldverhalten= Nachbarn[id %in% oldIDs, verhalten],
    oldID_Agenten = Nachbarn [id %in% oldIDs, Id_Agenten],
    oldDiff =Nachbarn[id %in% oldIDs, Diff],
    oldbewohnt =Nachbarn[id %in% oldIDs, bewohnt],
    oldgesundheit = Nachbarn[id %in% oldIDs, gesundheit],
    target = newIDs)
```

```
x <- Nachbarn[id %in% transition$origin]
x[,health := "empty"]
x[,age := NA]
x[,verhalten := NA]
x[, gesundheit :=NA]
x[, Id_Agenten := NA]
x[, bewohnt := 0]
x[, Diff := 0]
```

```
y <- Nachbarn[id %in% transition$target]
y[, health:= transition$oldhealth]
y[,age := transition$oldage]
y[,verhalten := transition$oldverhalten]
y[, gesundheit :=transition$oldgesundheit]
y[, Id_Agenten := transition$oldID_Agenten]
y[, bewohnt := transition$oldbewohnt]
y[, Diff := transition$oldDiff]
```

```
Result <- merge(x,y, all=TRUE)
```

```
Nachbarn[Result, on = .(id), health:= i.health]
Nachbarn[Result, on = .(id), age:= i.age]
Nachbarn[Result, on= .(id), verhalten := i.verhalten]
Nachbarn[Result, on = .(id), gesundheit:= i.gesundheit]
Nachbarn[Result, on= .(id), Id_Agenten := i.Id_Agenten]
Nachbarn[Result, on= .(id), bewohnt := i.bewohnt]
Nachbarn[Result, on= .(id), Diff := i.Diff]
```



```
cluster = makeCluster(4, type = "SOCK")
registerDoSNOW(cluster)

v <- foreach(i = 1:ni, export = c("Nachbarn", "dimen"), .packages = c("data.table"), .combine = rbind)
%doapar% {
  how_many_neighbors55()

}

stopCluster(cluster)

altes_verhalten <- Nachbarn[,verhalten];
Nachbarn[,Diff:= v- Nachbarn[,verhalten]];
c<- Nachbarn[,Diff*a];
d <- Nachbarn[,verhalten +c]+ runif(1,-0.5,0.5);
Nachbarn[,verhalten:= d ];
Nachbarn[,Verhaltensunterschied:= Nachbarn[,verhalten]- altes_verhalten];

f<- Nachbarn[,gesundheit]+Nachbarn[,Stressor*0.01]+ Nachbarn[,
Verhaltensunterschied*0.1]+runif(1,-0.5,0.5);#
Nachbarn[,gesundheit:= f];

v<-0;

Nachbarn[,gesundheit:= ((Nachbarn[,gesundheit]-
mean(na.omit(Nachbarn$gesundheit)))/sd(na.omit(Nachbarn$gesundheit)))]
```

```
}  
  
return(Nachbarn)  
  
}  
  
#####  
  
Nachbarn <- CS_Anpassung(n=5)  
Nachbarn <- iterate( n=10)  
  
for (anzahl_Varios in 1:20) {  
  
  test <- Nachbarn[ sample(nrow(Nachbarn), 0.1*ni),]  
  
  
  test <-na.omit(test)  
  
  name2 <-paste(c("Results",anzahl_Varios),collapse=".")  
  
  Nachbarn_geo1<-as.geodata(na.omit(cbind(test$x,test$y, test$verhalten)))  
  
  Nachbarn.variogram1<-variog(Nachbarn_geo1,estimator.type="classical", max.dist=30) #9
```

```
plot(Nachbarn.variogram1)

datafit1.ge<-variofit(Nachbarn.variogram1,c((var(na.omit(test$verhalten))),6),
cov.model="exponential")

datafit1.ge
lines(datafit1.ge)
title("Verhalten")

Nachbarn_geo2<-as.geodata(na.omit(cbind(test$x,test$y, test$gesundheit)))

Nachbarn.variogram2<-variog(Nachbarn_geo2,estimator.type="classical", max.dist=30) #9
plot(Nachbarn.variogram2)
datafit2.ge<-variofit(Nachbarn.variogram2,c(1,6), cov.model="exponential")
datafit2.ge
lines(datafit2.ge)
title("Gesundheit")

assign(name2, c(datafit2.ge[[1]],datafit2.ge[[2]], datafit2.ge[[8]],datafit1.ge[[1]],datafit1.ge[[2]],
datafit1.ge[[8]],mean(test$verhalten),var(test$verhalten),nrow(test)))

}

Result <- rbind(Results.1,Results.2, Results.3,Results.4, Results.5,
Results.6,Results.7,Results.8,Results.9,Results.10,Results.11,Results.12,Results.13,Results.14,Results.
15,Results.16,Results.17,Results.18,Results.19,Results.20)

Result[,3]<- Result[,2]/(Result[,1]+Result[,2])

Result[,7]<- Result[,6]/(Result[,5]+Result[,6])

colnames(Result) <- c("tausq_Gesundheit", "sigmaq(partial sill)_Gesundheit",
"RSV_Gesundheit","Range_Gesundheit","tausq_Verhalten", "sigmaq(partial sill)_Verhalten",
"RSV_Verhalten","Range_Verhalten", "Mean_Verhalten", " Varianz_Verhalten", "n")
```

```
save(Result, file=paste0("mitVerhaltensanfangscorrelation", "_Results_a_", a, ".RData", sep="."))
```