```
#An agroforestery model designed in March 2015 to simulate the soil organic carbon (SOC) dynamic designed by Remi Cardinael and Bertrand
Guenet.
#Published in Cardinael et al., 2018 Biogoesciences
#Any questions or suggestions please send emails to remi.cardinael@cirad.fr and/or bertrand.guenet@lsce.ipsl.fr.

#The time step of the model is 1year.
#The transport and decomposition models are based on Guenet et al., 2013 Biogeosciences. The moisture function is based on Moyano et al. 2012
Biogeosciences.
setwd('E:/Thema8/Agroforestry_model/Agroforestry_model/')
rm(list = ls())
print("*************************************")
print("Welcome!")
print("The model is starting")
#Read the parameter files

#Do we use priming?
priming="n"

#Do you want to run on the control plot?
control="n"

#Mean parameters (optimized coefficients)
PARAM=read.table('run_optim_p0_3pools.def',header=FALSE)

#Model parameter

#distance to the three in meter and resolution in m.

dist=6.5 # (middle of the agroforestry alley)
step_dist=0.1
d=as.data.frame(seq(step_dist,dist, by=step_dist))

#Soil depth in meter and resolution of the depth axis in m
depth=1.95
step_depth=0.1
z= as.data.frame(seq(-0.05,-depth,by=-step_depth))

#Tillage layer (in meter)
til_lay=-0.15

#Limit to crop roots development (in the alley)
limit_root_crop=-1.50

#Limit to grass roots development (in the tree row)
limit_root_grass=-1.50

#Duration of the agroforestry experiment in year
Agrof_length=18
time_Agrof_length=seq(1, Agrof_length)

#The limit from the tree where grass still grow (in meter)
limit_grass=1

#bulk density (kg m-3) from Cardinael et al., 2015 Geoderma
ta_bd<-z[,1]
for (i in 1:dim(z)[1]) {
  if (z[i,1]==as.character(-0.05)) {ta_bd[i]<-1.41}
  if (z[i,1]==as.character(-0.20)) {ta_bd[i]<-1.61}
  if (z[i,1]==as.character(-0.40)) {ta_bd[i]<-1.73}
  if (z[i,1]==as.character(-0.60)) {ta_bd[i]<-1.80}
  if (z[i,1]==as.character(-0.85)) {ta_bd[i]<-1.74}
  if (z[i,1]==as.character(-1.10)) {ta_bd[i]<-1.61}
  if (z[i,1]==as.character(-1.30)) {ta_bd[i]<-1.65}
  if (z[i,1]==as.character(-1.50)) {ta_bd[i]<-1.65}
  if (z[i,1]==as.character(-1.70)) {ta_bd[i]<-1.65}
  if (z[i,1]==as.character(-1.90)) {ta_bd[i]<-1.65}

  if (z[i,1]==as.character(-0.10)) {ta_bd[i]<-approx   (c(0.05,0.20),c(1.41,1.61), 0.10, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.15)) {ta_bd[i]<-approx   (c(0.05,0.20),c(1.41,1.61), 0.15, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.25)) {ta_bd[i]<-approx   (c(0.20,0.40),c(1.61, 1.73), 0.25, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.30)) {ta_bd[i]<-approx   (c(0.20,0.40),c(1.61, 1.73), 0.30, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.35)) {ta_bd[i]<-approx   (c(0.20,0.40),c(1.61, 1.73), 0.35, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.45)) {ta_bd[i]<-approx   (c(0.40,0.60),c(1.73, 1.80), 0.45, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.50)) {ta_bd[i]<-approx   (c(0.40,0.60),c(1.73, 1.80), 0.50, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.55)) {ta_bd[i]<-approx   (c(0.40,0.60),c(1.73, 1.80), 0.55, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.65)) {ta_bd[i]<-approx   (c(0.60,0.85),c(1.80, 1.74), 0.65, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.70)) {ta_bd[i]<-approx   (c(0.60,0.85),c(1.80, 1.74), 0.70, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.75)) {ta_bd[i]<-approx   (c(0.60,0.85),c(1.80, 1.74), 0.75, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.80)) {ta_bd[i]<-approx   (c(0.60,0.85),c(1.80, 1.74), 0.80, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.90)) {ta_bd[i]<-approx   (c(0.85,1.10),c(1.74, 1.61), 0.90, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.95)) {ta_bd[i]<-approx   (c(0.85,1.10),c(1.74, 1.61), 0.95, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.00)) {ta_bd[i]<-approx   (c(0.85,1.10),c(1.74, 1.61), 1.00, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.05)) {ta_bd[i]<-approx   (c(0.85,1.10),c(1.74, 1.61), 1.05, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.15)) {ta_bd[i]<-approx   (c(1.10,1.30),c(1.61, 1.65), 1.15, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.20)) {ta_bd[i]<-approx   (c(1.10,1.30),c(1.61, 1.65), 1.20, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.25)) {ta_bd[i]<-approx   (c(1.10,1.30),c(1.61, 1.65), 1.25, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.35)) {ta_bd[i]<-approx   (c(1.30,1.50),c(1.65, 1.65), 1.35, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.40)) {ta_bd[i]<-approx   (c(1.30,1.50),c(1.65, 1.65), 1.40, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.45)) {ta_bd[i]<-approx   (c(1.30,1.50),c(1.65, 1.65), 1.45, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.55)) {ta_bd[i]<-approx   (c(1.50,1.70),c(1.65, 1.65), 1.55, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.60)) {ta_bd[i]<-approx   (c(1.50,1.70),c(1.65, 1.65), 1.60, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.65)) {ta_bd[i]<-approx   (c(1.50,1.70),c(1.65, 1.65), 1.65, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.75)) {ta_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 1.75, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.80)) {ta_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 1.80, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.85)) {ta_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 1.85, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.95)) {ta_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 1.95, method="linear",rule = 2, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-2.00)) {ta_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 2.00, method="linear",rule = 2, f = 0, ties = mean)$y[1]}
}

ir_bd<-z[,1]
for (i in 1:dim(z)[1]) {
  if (z[i,1]==as.character(-0.05)) {ir_bd[i]<-1.23}
  if (z[i,1]==as.character(-0.20)) {ir_bd[i]<-1.60}
```

```
  if (z[i,1]==as.character(-0.40)) {ir_bd[i]<-1.67}
  if (z[i,1]==as.character(-0.60)) {ir_bd[i]<-1.77}
  if (z[i,1]==as.character(-0.85)) {ir_bd[i]<-1.71}
  if (z[i,1]==as.character(-1.10)) {ir_bd[i]<-1.55}
  if (z[i,1]==as.character(-1.30)) {ir_bd[i]<-1.64}
  if (z[i,1]==as.character(-1.50)) {ir_bd[i]<-1.64}
  if (z[i,1]==as.character(-1.70)) {ir_bd[i]<-1.65}
  if (z[i,1]==as.character(-1.90)) {ir_bd[i]<-1.65}

  if (z[i,1]==as.character(-0.10)) {ir_bd[i]<-approx   (c(0.05,0.20),c(1.23,1.60), 0.10, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.15)) {ir_bd[i]<-approx   (c(0.05,0.20),c(1.23,1.60), 0.15, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.25)) {ir_bd[i]<-approx   (c(0.20,0.40),c(1.60, 1.67), 0.25, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.30)) {ir_bd[i]<-approx   (c(0.20,0.40),c(1.60, 1.67), 0.30, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.35)) {ir_bd[i]<-approx   (c(0.20,0.40),c(1.60, 1.67), 0.35, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.45)) {ir_bd[i]<-approx   (c(0.40,0.60),c(1.67, 1.77), 0.45, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.50)) {ir_bd[i]<-approx   (c(0.40,0.60),c(1.67, 1.77), 0.50, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.55)) {ir_bd[i]<-approx   (c(0.40,0.60),c(1.67, 1.77), 0.55, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.65)) {ir_bd[i]<-approx   (c(0.60,0.85),c(1.77, 1.71), 0.65, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.70)) {ir_bd[i]<-approx   (c(0.60,0.85),c(1.77, 1.71), 0.70, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.75)) {ir_bd[i]<-approx   (c(0.60,0.85),c(1.77, 1.71), 0.75, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.80)) {ir_bd[i]<-approx   (c(0.60,0.85),c(1.77, 1.71), 0.80, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.90)) {ir_bd[i]<-approx   (c(0.85,1.10),c(1.71, 1.55), 0.90, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.95)) {ir_bd[i]<-approx   (c(0.85,1.10),c(1.71, 1.55), 0.95, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.00)) {ir_bd[i]<-approx   (c(0.85,1.10),c(1.71, 1.55), 1.00, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.05)) {ir_bd[i]<-approx   (c(0.85,1.10),c(1.71, 1.55), 1.05, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.15)) {ir_bd[i]<-approx   (c(1.10,1.30),c(1.55, 1.64), 1.15, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.20)) {ir_bd[i]<-approx   (c(1.10,1.30),c(1.55, 1.64), 1.20, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.25)) {ir_bd[i]<-approx   (c(1.10,1.30),c(1.55, 1.64), 1.25, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.35)) {ir_bd[i]<-approx   (c(1.30,1.50),c(1.64, 1.64), 1.35, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.40)) {ir_bd[i]<-approx   (c(1.30,1.50),c(1.64, 1.64), 1.40, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.45)) {ir_bd[i]<-approx   (c(1.30,1.50),c(1.64, 1.64), 1.45, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.55)) {ir_bd[i]<-approx   (c(1.50,1.70),c(1.64, 1.65), 1.55, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.60)) {ir_bd[i]<-approx   (c(1.50,1.70),c(1.64, 1.65), 1.60, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.65)) {ir_bd[i]<-approx   (c(1.50,1.70),c(1.64, 1.65), 1.65, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.75)) {ir_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 1.75, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.80)) {ir_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 1.80, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.85)) {ir_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 1.85, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.95)) {ir_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 1.95, method="linear",rule = 2, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-2.00)) {ir_bd[i]<-approx   (c(1.70,1.90),c(1.65, 1.65), 2.00, method="linear",rule = 2, f = 0, ties = mean)$y[1]}
}

tr_bd<-z[,1]
for (i in 1:dim(z)[1]) {
  if (z[i,1]==as.character(-0.05)) {tr_bd[i]<-1.10}
  if (z[i,1]==as.character(-0.20)) {tr_bd[i]<-1.49}
  if (z[i,1]==as.character(-0.40)) {tr_bd[i]<-1.71}
  if (z[i,1]==as.character(-0.60)) {tr_bd[i]<-1.73}
  if (z[i,1]==as.character(-0.85)) {tr_bd[i]<-1.68}
  if (z[i,1]==as.character(-1.10)) {tr_bd[i]<-1.55}
  if (z[i,1]==as.character(-1.30)) {tr_bd[i]<-1.63}
  if (z[i,1]==as.character(-1.50)) {tr_bd[i]<-1.64}
  if (z[i,1]==as.character(-1.70)) {tr_bd[i]<-1.62}
  if (z[i,1]==as.character(-1.90)) {tr_bd[i]<-1.64}

  if (z[i,1]==as.character(-0.10)) {tr_bd[i]<-approx   (c(0.05,0.20),c(1.10,1.49), 0.10, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.15)) {tr_bd[i]<-approx   (c(0.05,0.20),c(1.10,1.49), 0.15, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.25)) {tr_bd[i]<-approx   (c(0.20,0.40),c(1.49, 1.71), 0.25, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.30)) {tr_bd[i]<-approx   (c(0.20,0.40),c(1.49, 1.71), 0.30, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.35)) {tr_bd[i]<-approx   (c(0.20,0.40),c(1.49, 1.71), 0.35, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.45)) {tr_bd[i]<-approx   (c(0.40,0.60),c(1.71, 1.73), 0.45, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.50)) {tr_bd[i]<-approx   (c(0.40,0.60),c(1.71, 1.73), 0.50, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.55)) {tr_bd[i]<-approx   (c(0.40,0.60),c(1.71, 1.73), 0.55, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.65)) {tr_bd[i]<-approx   (c(0.60,0.85),c(1.73, 1.68), 0.65, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.70)) {tr_bd[i]<-approx   (c(0.60,0.85),c(1.73, 1.68), 0.70, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.75)) {tr_bd[i]<-approx   (c(0.60,0.85),c(1.73, 1.68), 0.75, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.80)) {tr_bd[i]<-approx   (c(0.60,0.85),c(1.73, 1.68), 0.80, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.90)) {tr_bd[i]<-approx   (c(0.85,1.10),c(1.68, 1.55), 0.90, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-0.95)) {tr_bd[i]<-approx   (c(0.85,1.10),c(1.68, 1.55), 0.95, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.00)) {tr_bd[i]<-approx   (c(0.85,1.10),c(1.68, 1.55), 1.00, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.05)) {tr_bd[i]<-approx   (c(0.85,1.10),c(1.68, 1.55), 1.05, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.15)) {tr_bd[i]<-approx   (c(1.10,1.30),c(1.55, 1.63), 1.15, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.20)) {tr_bd[i]<-approx   (c(1.10,1.30),c(1.55, 1.63), 1.20, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.25)) {tr_bd[i]<-approx   (c(1.10,1.30),c(1.55, 1.63), 1.25, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.35)) {tr_bd[i]<-approx   (c(1.30,1.50),c(1.63, 1.64), 1.35, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.40)) {tr_bd[i]<-approx   (c(1.30,1.50),c(1.63, 1.64), 1.40, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.45)) {tr_bd[i]<-approx   (c(1.30,1.50),c(1.63, 1.64), 1.45, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.55)) {tr_bd[i]<-approx   (c(1.50,1.70),c(1.64, 1.62), 1.55, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.60)) {tr_bd[i]<-approx   (c(1.50,1.70),c(1.64, 1.62), 1.60, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.65)) {tr_bd[i]<-approx   (c(1.50,1.70),c(1.64, 1.62), 1.65, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.75)) {tr_bd[i]<-approx   (c(1.70,1.90),c(1.62, 1.64), 1.75, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.80)) {tr_bd[i]<-approx   (c(1.70,1.90),c(1.62, 1.64), 1.80, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.85)) {tr_bd[i]<-approx   (c(1.70,1.90),c(1.62, 1.64), 1.85, method="linear",rule = 1, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-1.95)) {tr_bd[i]<-approx   (c(1.70,1.90),c(1.62, 1.64), 1.95, method="linear",rule = 2, f = 0, ties = mean)$y[1]}
  if (z[i,1]==as.character(-2.00)) {tr_bd[i]<-approx   (c(1.70,1.90),c(1.62, 1.64), 2.00, method="linear",rule = 2, f = 0, ties = mean)$y[1]}
}

bd<-matrix(ncol=dim(d)[1],nrow=dim(z)[1])

for (i in 1:dim(d)[1]) {
  if (d[i,1]<= limit_grass) {bd[,i]<-tr_bd} else {bd[,i]<-ir_bd}
}

#Soil temperature (in K)
temp<--0.8931*(-z[,1])+288.24

#The soil moisture profile (in volumetric fraction)
moist_profil=0.0476*(-z[,1])+0.284

#pH (in pH units) profile
ph<-0.1603*log(-z[,1]) + 8.3608

#Clay content profile from Cardinael et al., 2015 Geoderma
af_clay<-z[,1]
ta_clay<-z[,1]
```

```
for (i in 1:dim(z)[1]) {
  if (z[i,1]==as.character(-0.05)) {af_clay[i]<-0.1754}
  if (z[i,1]==as.character(-0.20)) {af_clay[i]<-0.17029}
  if (z[i,1]==as.character(-0.40)) {af_clay[i]<-0.17762}
  if (z[i,1]==as.character(-0.60)) {af_clay[i]<-0.25004}
  if (z[i,1]==as.character(-0.85)) {af_clay[i]<-0.3092}
  if (z[i,1]==as.character(-1.10)) {af_clay[i]<-0.32209}
  if (z[i,1]==as.character(-1.30)) {af_clay[i]<-0.33695}
  if (z[i,1]==as.character(-1.50)) {af_clay[i]<-0.34204}
  if (z[i,1]==as.character(-1.70)) {af_clay[i]<-0.3399394}
  if (z[i,1]==as.character(-1.90)) {af_clay[i]<-0.3316413}

  if (z[i,1]==as.character(-0.10)) {af_clay[i]<-approx   (c(0.05,0.20),c(0.1754,0.17029), 0.10, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.15)) {af_clay[i]<-approx   (c(0.05,0.20),c(0.1754,0.17029), 0.15, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.25)) {af_clay[i]<-approx   (c(0.20,0.40),c(0.17029, 0.17762), 0.25, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.30)) {af_clay[i]<-approx   (c(0.20,0.40),c(0.17029, 0.17762), 0.30, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.35)) {af_clay[i]<-approx   (c(0.20,0.40),c(0.17029, 0.17762), 0.35, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.45)) {af_clay[i]<-approx   (c(0.40,0.60),c(0.17762, 0.25004), 0.45, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.50)) {af_clay[i]<-approx   (c(0.40,0.60),c(0.17762, 0.25004), 0.50, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.55)) {af_clay[i]<-approx   (c(0.40,0.60),c(0.17762, 0.25004), 0.55, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.65)) {af_clay[i]<-approx   (c(0.60,0.85),c(0.25004, 0.3092), 0.65, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.70)) {af_clay[i]<-approx   (c(0.60,0.85),c(0.25004, 0.3092), 0.70, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.75)) {af_clay[i]<-approx   (c(0.60,0.85),c(0.25004, 0.3092), 0.75, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.80)) {af_clay[i]<-approx   (c(0.60,0.85),c(0.25004, 0.3092), 0.80, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.90)) {af_clay[i]<-approx   (c(0.85,1.10),c(0.3092, 0.32209), 0.90, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.95)) {af_clay[i]<-approx   (c(0.85,1.10),c(0.3092, 0.32209), 0.95, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.00)) {af_clay[i]<-approx   (c(0.85,1.10),c(0.3092, 0.32209), 1.00, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.05)) {af_clay[i]<-approx   (c(0.85,1.10),c(0.3092, 0.32209), 1.05, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.15)) {af_clay[i]<-approx   (c(1.10,1.30),c(0.32209, 0.33695), 1.15, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.20)) {af_clay[i]<-approx   (c(1.10,1.30),c(0.32209, 0.33695), 1.20, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.25)) {af_clay[i]<-approx   (c(1.10,1.30),c(0.32209, 0.33695), 1.25, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.35)) {af_clay[i]<-approx   (c(1.30,1.50),c(0.33695, 0.34204), 1.35, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.40)) {af_clay[i]<-approx   (c(1.30,1.50),c(0.33695, 0.34204), 1.40, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.45)) {af_clay[i]<-approx   (c(1.30,1.50),c(0.33695, 0.34204), 1.45, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.55)) {af_clay[i]<-approx   (c(1.50,1.70),c(0.34204, 0.3399394), 1.55, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.60)) {af_clay[i]<-approx   (c(1.50,1.70),c(0.34204, 0.3399394), 1.60, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.65)) {af_clay[i]<-approx   (c(1.50,1.70),c(0.34204, 0.3399394), 1.65, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.75)) {af_clay[i]<-approx   (c(1.70,1.90),c(0.3399394, 0.3316413), 1.75, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.80)) {af_clay[i]<-approx   (c(1.70,1.90),c(0.3399394, 0.3316413), 1.80, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.85)) {af_clay[i]<-approx   (c(1.70,1.90),c(0.3399394, 0.3316413), 1.85, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.95)) {af_clay[i]<-approx   (c(1.70,1.90),c(0.3399394, 0.3316413), 1.95, method="linear",rule = 2, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-2.00)) {af_clay[i]<-approx   (c(1.70,1.90),c(0.3399394, 0.3316413), 2.00, method="linear",rule = 2, f = 0, ties =
mean)$y[1]}
}

for (i in 1:dim(z)[1]) {
  if (z[i,1]==as.character(-0.05)) {ta_clay[i]<-0.1785269}
  if (z[i,1]==as.character(-0.20)) {ta_clay[i]<-0.1730538}
  if (z[i,1]==as.character(-0.40)) {ta_clay[i]<-0.1773226}
  if (z[i,1]==as.character(-0.60)) {ta_clay[i]<-0.2426882}
  if (z[i,1]==as.character(-0.85)) {ta_clay[i]<-0.3069355}
  if (z[i,1]==as.character(-1.10)) {ta_clay[i]<-0.3256022}
  if (z[i,1]==as.character(-1.30)) {ta_clay[i]<-0.3305161}
  if (z[i,1]==as.character(-1.50)) {ta_clay[i]<-0.3286292}
  if (z[i,1]==as.character(-1.70)) {ta_clay[i]<-0.3283146}
  if (z[i,1]==as.character(-1.90)) {ta_clay[i]<-0.3126136}

  if (z[i,1]==as.character(-0.10)) {ta_clay[i]<-approx   (c(0.05,0.20),c(0.1785269,0.1730538), 0.10, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.15)) {ta_clay[i]<-approx   (c(0.05,0.20),c(0.1785269,0.1730538), 0.15, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.25)) {ta_clay[i]<-approx   (c(0.20,0.40),c(0.1730538, 0.1773226), 0.25, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.30)) {ta_clay[i]<-approx   (c(0.20,0.40),c(0.1730538, 0.1773226), 0.30, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.35)) {ta_clay[i]<-approx   (c(0.20,0.40),c(0.1730538, 0.1773226), 0.35, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.45)) {ta_clay[i]<-approx   (c(0.40,0.60),c(0.1773226, 0.2426882), 0.45, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.50)) {ta_clay[i]<-approx   (c(0.40,0.60),c(0.1773226, 0.2426882), 0.50, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.55)) {ta_clay[i]<-approx   (c(0.40,0.60),c(0.1773226, 0.2426882), 0.55, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.65)) {ta_clay[i]<-approx   (c(0.60,0.85),c(0.2426882, 0.3069355), 0.65, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
```

```
  if (z[i,1]==as.character(-0.70)) {ta_clay[i]<-approx   (c(0.60,0.85),c(0.2426882, 0.3069355), 0.70, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.75)) {ta_clay[i]<-approx   (c(0.60,0.85),c(0.2426882, 0.3069355), 0.75, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.80)) {ta_clay[i]<-approx   (c(0.60,0.85),c(0.2426882, 0.3069355), 0.80, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.90)) {ta_clay[i]<-approx   (c(0.85,1.10),c(0.3069355, 0.3256022), 0.90, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-0.95)) {ta_clay[i]<-approx   (c(0.85,1.10),c(0.3069355, 0.3256022), 0.95, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.00)) {ta_clay[i]<-approx   (c(0.85,1.10),c(0.3069355, 0.3256022), 1.00, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.05)) {ta_clay[i]<-approx   (c(0.85,1.10),c(0.3069355, 0.3256022), 1.05, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.15)) {ta_clay[i]<-approx   (c(1.10,1.30),c(0.3256022, 0.3305161), 1.15, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.20)) {ta_clay[i]<-approx   (c(1.10,1.30),c(0.3256022, 0.3305161), 1.20, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.25)) {ta_clay[i]<-approx   (c(1.10,1.30),c(0.3256022, 0.3305161), 1.25, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.35)) {ta_clay[i]<-approx   (c(1.30,1.50),c(0.3305161, 0.3286292), 1.35, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.40)) {ta_clay[i]<-approx   (c(1.30,1.50),c(0.3305161, 0.3286292), 1.40, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.45)) {ta_clay[i]<-approx   (c(1.30,1.50),c(0.3305161, 0.3286292), 1.45, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.55)) {ta_clay[i]<-approx   (c(1.50,1.70),c(0.3286292, 0.3283146), 1.55, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.60)) {ta_clay[i]<-approx   (c(1.50,1.70),c(0.3286292, 0.3283146), 1.60, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.65)) {ta_clay[i]<-approx   (c(1.50,1.70),c(0.3286292, 0.3283146), 1.65, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.75)) {ta_clay[i]<-approx   (c(1.70,1.90),c(0.3283146, 0.3126136), 1.75, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.80)) {ta_clay[i]<-approx   (c(1.70,1.90),c(0.3283146, 0.3126136), 1.80, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.85)) {ta_clay[i]<-approx   (c(1.70,1.90),c(0.3283146, 0.3126136), 1.85, method="linear",rule = 1, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-1.95)) {ta_clay[i]<-approx   (c(1.70,1.90),c(0.3283146, 0.3126136), 1.95, method="linear",rule = 2, f = 0, ties =
mean)$y[1]}
  if (z[i,1]==as.character(-2.00)) {ta_clay[i]<-approx   (c(1.70,1.90),c(0.3283146, 0.3126136), 2.00, method="linear",rule = 2, f = 0, ties =
mean)$y[1]}
}

#Clay function coming from ORCHIDEE
#For this site we do not take that into account the effect of clay on decomposition since we define the decomposition
#rate of SOM in function of depth. The effect of depth is probably due to clay differences.
ta_clay_func<-1-0.75* ta_clay
af_clay_func<-1-0.75* af_clay


#Duration of the spinup in year
spin_length=5000

#Decomposition rate of Wheat roots (yr-1)
kw=3.03

#Decomposition rate of Walnut tree roots (yr-1)
kt=2.46

#C concentration in wheat roots (g g-1)
C_cont_root=0.3514

#C concentration in wheat straw (g g-1)
C_cont_straw=0.4332

#Non_exported straw (0-1)
Non_exp_straw=0.25

#Straw:yield ratio of wheat (unitless)
St_Yi_ratio=1.03

#Above Ground Biomass:yield ratio of wheat (unitless)
AB_Yi_ratio=2.45

#Root:shoot ratio of wheat (unitless)
Ro_Sh_ratio=0.79

#Decomposition rate of of slow SOM (1/residence time)
ks<-z[,1]
for (i in 2:(dim(z)[1]+1)) {ks[i-1]<-(PARAM[1,1]/100)*exp(-1.455*(-z[i-1,1]))*365.25}
#ks<-1/100

#Decomposition rate of of passive SOM (1/residence time)
kp<-z[,1]
for (i in 2:(dim(z)[1]+1)) {kp[i-1]<-(PARAM[2,1]/100)*exp(-1.455*(-z[i-1,1]))*365.25}

#Q10 value
Q10=2.0

#The moisture fonction are based on Moyano et al but to save computing time we used the measured stocks as input of the fonction and not the
ones calculated by the model.

#Moyano et al 2012 Biogoesciences moisture function
moyano<-function(SOM,clay){


  # ====================================================================
  # ===== R Code for the Analysis of Soil Moisture-Respiration Data =====
  # ====================================================================
  # Note: this script  also analyzes water holding capacity: not in the paper given the low number of datasets
  # Note added by Remi Cardinael and Bertrand Guenet, the original script by Moyano et al was a bit simplified (but not the calculation)
  # to improve the computing time.
```

```
  # ===== Prepare Data
  # Read in the data files (MRD = moisture respiration data; DD = data description; funs = function indexes)
  MD<-read.csv("MRD.csv")
  DD<-read.csv("DD.csv")
  funs<-read.csv("funs.csv")
  # create the description data.frame by aggregating and binding
  DD1<-aggregate(x = MD, by = list(MD$id), FUN = "mean")
  DD1<-subset(DD1,select=c(soil.t,bd:pd,co2.dw))
  DD<-cbind(DD, DD1)
  rm(DD1)
  # Transform the mwplog to a practical scale
  MD$mwplog<-(MD$mwplog)/5*(-1)+1
  # Subset some variables to work with
  MDsub <- subset(MD, select=c(id,mdw,mvol,mps,mwhc,mwplog,min.rel))
  # mdw=gravimetric moisture, mvol=volumetric, mps=water saturation, mwhc=water holding capacity moisture, mwplog=log water potential
  # min.rel=relative carbon mineralization (note: C min. was previously normalized although doing this does not affect the analysis)

  # ===== Fit smooth curves to each dataset
  library(gam)
  mod.mdw<-list()
  mod.mvol<-list()
  mod.mwhc<-list()
  mod.mwplog<-list()
  mod.mps<-list()
  models<-list(mod.mdw=mod.mdw,mod.mvol=mod.mvol,mod.mps=mod.mps,mod.mwhc=mod.mwhc,mod.mwplog=mod.mwplog) # Make a list object for the models
  rm(mod.mdw,mod.mvol,mod.mps,mod.mwhc,mod.mwplog)
  for (i in 1:length(DD$id)) {
    WD <- MDsub[MDsub$id==i,] # create a working data frame
    ydata<-WD$min.rel
    for (j in 1:5) {
      xdata<-WD[,j+1]
      funid<-funs[i,j]
      # fit the functions:
      if (funid==1) { mod <- lm (ydata~xdata)} else
        if (funid==2) mod <- lm (ydata~xdata+I(xdata^2)) else
          if (funid==3) mod <- lm (ydata~xdata+I(xdata^2)+I(xdata^3)) else
            if (funid==7) {mod <- gam (ydata~s(xdata)); funs[i,j]<-7} else  mod <- NA
      models[[j]][i]<-list(mod) # save the model in a list
    }
  }
  rm(mod,i,j,funid,xdata,ydata,WD)

  # ===== predict values and store in a matrix
  # Need objects: DD, MDsub, models, funs
  # create matrices of length length(DD$id) by 110 and put into a list
  pred.mdw<-array(NA,c(length(DD$id),400))
  pred.mvol<-array(NA,c(length(DD$id),110))
  pred.mps<-array(NA,c(length(DD$id),110))
  pred.mwhc<-array(NA,c(length(DD$id),110))
  pred.mwplog<-array(NA,c(length(DD$id),110))
  pred<-list(pred.mdw=pred.mdw,pred.mvol=pred.mvol,pred.mps=pred.mps,pred.mwhc=pred.mwhc,pred.mwplog=pred.mwplog)
  rm(pred.mdw,pred.mvol,pred.mps,pred.mwhc,pred.mwplog)
  # get predicted values and store in matrix at the correct positions
  for (i in 1:length(DD$id)) {
    for (j in 1:5) {
      if (funs[i,j] > 0) {      # if there is a function fitted for this dataset then...
        x0<-min(MDsub[MDsub$id==i,j+1],na.rm=TRUE)
        x1<-max(MDsub[MDsub$id==i,j+1],na.rm=TRUE)
        x0<-ceiling(x0*100)/100 # round up at the second decimal
        x1<-ceiling(x1*100)/100
        xv<-seq(x0,x1,0.01) # create a sequence of values used to predict
        yv<-predict(models[[j]][[i]], list(xdata=xv))   # predict using the function fitted on the dataset
        im <- array(c(rep(i,length(xv)),seq(x0*100,x1*100,1)), dim=c(length(xv),2)) # create an index matrix (im) to position values in the
matrix.
        pred[[j]][im]<-yv       # assign the predicted values using the index matrix
      }
    }
  }
  rm(i, im, j, x0,x1, xv, yv)

  # ===== Calculate the Proportional Change in Soil Respiration (PCSR)
  # Here we take values in the predicted tables at moisture x and divide it by the value at x-0.01
  # giving the change in respiration at each 0.01 range
  prePCSR <- list()
  PCSR <- list()
  for (i in 1:5) {
    A<-pred[[i]]
    B<-cbind(A[,2:ncol(A)],NA)
    C<-(B/A)
    prePCSR[i]<-list(C)
  }
  # average to get the correct PCSR at the given moisture point
  for (i in 1:5) {
    A<-prePCSR[[i]]
    B<-cbind(NA,A[,1:ncol(A)-1])
    C<-(A+B)/2
    PCSR[i]<-list(C)
  }
  PCSRnames <- list("PCSR.mdw","PCSR.mvol","PCSR.mps","PCSR.mwhc","PCSR.mwplog")
  names(PCSR)<-PCSRnames
  rm(PCSRnames,A,B,C,i,prePCSR)

  # ========================================================
  # ===== Linear Regressions with Soil Properties
  # Make lists for assigning names
  moistnames <- list("mdw","mvol","mps","mwhc","wplog")
  varnames <- list("bd", "corg", "ps", "clay", "silt", "sand")
  lmnames <-list()
  lst <- 1
  for (j in 1:5) {
    for (i in 1:6) {
      lmnames[lst] <- list(paste("lm", moistnames[[j]],varnames[[i]],sep="."))
      lst <- lst+1
    }
```

```
  }
  # Subset some variables to work with
  DDsub <- subset(DD, select=c(id,soil.t:sand,co2.dw,maxtime))
  DDsub[DDsub$c.org>0.05 | is.na(DDsub$c.org),]<-NA # select either mineral or organic soils
  # explore each variable separately: fit a linear regression at points along the moisture range
  lmfunc <- function (A) {if (sum(as.vector(!is.na(A))*as.vector(!is.na(DDsub[,i])))>2) {lm(A~DDsub[,i], na.action=na.exclude)} else return
(NA)}
  lmsep<-list()
  lst <- 1
  for (j in 1:5) {
    for (i in 3:8) {
      ind <- seq(1,99,2) # select a subset of moisture points for applying regressions
      PCSR.tmp <- PCSR[[j]][,ind]
      x <- apply(PCSR.tmp, 2, lmfunc)
      lmsep[lst]<- list(x)
      lst <- lst+1
    }
  }
  names(lmsep) <- lmnames
  rm(lmfunc, moistnames,varnames,lmnames, x,i,j,lst,PCSR.tmp,ind)


  # ========================================================
  #====== Multiple Linear Regression =====
  #====  Model excluding bulk density and organic soils

  library(MASS)
  #======= Prepare the data
  lmdata.texorg<-list()
  moist<-rep(1:110,each=107); moist<-moist/100 # the moisture variable: repeat each value 107 times (the amount of datasets)
  DDsub<- subset(DD, select=c(id,ecosystem,c.org,clay,soil.t,maxtime)) # subset variables used
  DDrep<-DDsub
  for (i in 1:109) {DDrep<-rbind(DDrep,DDsub)} # repeat each case 110 time (the amount of moisture points)
  DDrep<-cbind(DDrep,moist)
  rm(i,DDsub,moist)
  for (j in c(1,2,3,4,5)) {
    pcsr<-as.vector(PCSR[[j]][,1:110])
    rm<-mean(pcsr,na.rm=TRUE); rsd<-sd(pcsr,na.rm=TRUE); pcsr[pcsr>rm+3*rsd]<-NA # remove extreme values?
    Data<-DDrep
    Data<-cbind(DDrep,pcsr)
    rm(pcsr)
    # Make a vector (x) that marks all rows where all variables are present, subset accordingly
    x<-rep(1,length(Data[,1]))
    for (i in c(3,4,7,8)) {AA<-as.vector(!is.na(Data[,i])); x<-x*AA}
    # x[Data$ecosystem=="Forest"]<-0 # select ecosystem?
    x[Data$c.org>0.05]<-0 # remove high c.org soils
    Data<-Data[x==1,]
    lmdata.texorg[j]<-list(Data)
  }
  # Fit Linear Models
  lm.moist.texorg<-list()
  lm.texorg<-list()
  for (j in c(1,2,3,4,5)) {
    lmmod<-stepAIC(lm(pcsr~moist+I(moist^2)+I(moist^3), data=lmdata.texorg[[j]]),k=log(length(lmdata.texorg[[j]]$pcsr)),trace=FALSE)
    lm.moist.texorg[j]<-list(lmmod)
    lmmod<-stepAIC(lm(pcsr~moist+I(moist^2)+I(moist^3)+moist*clay+c.org,
data=lmdata.texorg[[j]]),k=log(length(lmdata.texorg[[j]]$pcsr)),trace=FALSE)
    lm.texorg[j]<-list(lmmod)
  }
  rm(i,j,AA,x,rsd,rm,Data,lmmod,DDrep)




  # ====== model and measures used
  model<-lm.texorg
  m1<-1
  m2<-2
  m3<-3
  m4<-5
  # ====== create dataframe for newdata

  for (j in c(m2)) {
    # ====== predict and simulate changing corg
    corgmat<-matrix(nrow=dim(z)[1],ncol=60)
    for (k in 1:as.numeric(length(SOM))) {
      #     simdata$ecosystem<-as.factor("Cultivated")
      simdata<-data.frame(moist=seq(0.02,1.2,0.02))
      simdata$c.org<-(SOM[k])/1000 # To convert from g kg-1 to g g-1
      simdata$clay<-clay[k]
      x<-predict(model[[j]], newdata=simdata)
      y<-vector(length=length(x))
      for (i in 1:length(y)){
        if (i==1) {y[i]<-x[i]} else {y[i]<-y[i-1]*x[i]}
      }
      y<-y/max(y)
      corgmat[k,]<-y
      # scale values to start at 0
      if (j!=5) {
        ind<-which.max(corgmat[k,])
        corgmat[k,1:ind]<-(corgmat[k,1:ind]-min(corgmat[k,1:ind]))
        corgmat[k,1:ind]<-corgmat[k,1:ind]/max(corgmat[k,1:ind])

      }

    }
    return(corgmat)
  }
}

#SOC stocks are from Cardinael et al., 2015 Geoderma
#For control
SOM <-
```

```
c(9.34,13.23,13.23,11.81,11.81,10.17,10.17,10.11,10.11,10.11,9.65,9.65,10.06,10.06,10.22,10.22,9.69,9.69,8.33,8.33)*0.1/(ta_bd*step_depth)
clay<-ta_clay
moist_f<-moyano(SOM,clay)
hum_soil<-seq(0.02,1.2,0.02)
mf_ctrl<-rep(1,dim(z)[1])

for (j in 1:dim(z)[1])  {
  k<-which.min(abs(hum_soil-moist_profil[j]))
  mf_ctrl[j]<-moist_f[j,k]
}
for (j in 1:dim(z)[1]) {if (mf_ctrl[j] > 1) {mf_ctrl[j]<-1} else {mf_ctrl[j]<-mf_ctrl[j]}}

#For tree line
SOM <-
c(21.56,15.63,15.63,12.14,12.14,10.47,10.47,10.79,10.79,10.79,9.95,9.95,10.27,10.27,10.10,10.10,9.22,9.22,8.30,8.30)*0.1/(ta_bd*step_depth)
clay<-af_clay
moist_f<-moyano(SOM,clay)
hum_soil<-seq(0.02,1.2,0.02)
mf_tl<-rep(1,dim(z)[1])

for (j in 1:dim(z)[1])  {
  k<-which.min(abs(hum_soil-moist_profil[j]))
  mf_tl[j]<-moist_f[j,k]
}
for (j in 1:dim(z)[1]) {if (mf_tl[j] > 1) {mf_tl[j]<-1} else {mf_tl[j]<-mf_tl[j]}}

#For inter row
SOM <- c(9.78,14.04,14.04,12.04,12.04,10.22,10.22,10.43,10.43,10.43,9.72,9.72,9.82,9.82,9.82,9.82,9.05,9.05,7.76,7.76)*0.1/(ta_bd*step_depth)
clay<-af_clay
moist_f<-moyano(SOM,clay)
hum_soil<-seq(0.02,1.2,0.02)
mf_ir<-rep(1,dim(z)[1])

for (j in 1:dim(z)[1])  {
  k<-which.min(abs(hum_soil-moist_profil[j]))
  mf_ir[j]<-moist_f[j,k]
}
for (j in 1:dim(z)[1]) {if (mf_ir[j] > 1) {mf_ir[j]<-1} else {mf_ir[j]<-mf_ir[j]}}

#Yield of decomposed FOM that goes to SOM (0-1)
e=PARAM[5,1]

#DBH (m)
DBH<-0.0157* time_Agrof_length - 0.0391
for (k in 1: Agrof_length)     {if (DBH[k]<0) {DBH[k]<-0.01}}

#Yield for wheat in the control plot (t DM ha-1, average over 20 years)
Yield_ctrl<-3.79

#Yield for wheat in the AF plot (t DM ha-1)
#Here we defined b as the minimum between a and 1 with a being the function of yield reduction
#depending on DBH
a<-DBH
b<-DBH

a[]<-((-93.33*DBH[]+100)/100)
for (k in 1: Agrof_length)  {b[k]<-min(1,a[k])  }

Yield_AF<-matrix(ncol=dim(d)[1],nrow= Agrof_length)

#Here, the coefficient were obtained based on data regression, but the function underestimates the
#yield by 20%, that's why we multiplied by 1.2

for (k in 1:dim(d)[1])     {
  Yield_AF[1: Agrof_length,k]<- 1.2*((4.39*d[k,1]+64.57)/100)*Yield_ctrl*b[1: Agrof_length]
}

#mortality rate for tree, crop and grass roots (year)
mr_tree=2.2 #(cf Germon et al. 2016 Plant and Soil)
mr_crop=1
mr_grass=1

#Aboveground total inputs (tC ha-1)
Input_leaves= 3.04568*DBH

Input_grass_ab<-d[,1]
for (k in 1:dim(d)[1])     {
  if (d[k,1]<=limit_grass)   {
    Input_grass_ab[k]=2.13
  }
  else                              {
    Input_grass_ab[k]= 0
  }
}

Input_crop_ab_spin<-d[,1]
Input_crop_ab<-matrix(ncol=dim(d)[1],nrow= Agrof_length)

Input_crop_ab_spin[]<-Yield_ctrl* St_Yi_ratio* Non_exp_straw* C_cont_straw

for (k in 1:dim(d)[1])     {
  if (d[k,1]<=limit_grass)   {
    Input_crop_ab[,k]=0
  }
  else                              {
    Input_crop_ab[,k]= Yield_AF[,k]* St_Yi_ratio* Non_exp_straw* C_cont_straw
  }
}


#Belowground total inputs (tC ha-1)
Input_TR<-matrix(ncol=dim(d)[1],nrow= Agrof_length)
Input_CR_SPIN<-d[,1]
Input_CR<-matrix(ncol=dim(d)[1],nrow= Agrof_length)
```

```
Input_GR<-d[,1]

#To obtain the relationship between belowground inputs and DBH, we just do a linear regression between the inputs in 2012 after 20 years of
#agroforestery and a point of 0 inputs at t0.
for (t in 1: Agrof_length) {
  for (k in 1:dim(d)[1])     {
    if (d[k,1]<=limit_grass) {
      Input_TR[t,k]=3.6886*DBH[t]
    }
    else                           {
      Input_TR[t,k]=4.7005*DBH[t]*exp(-0.27531*(d[k,1]))
    }
  }
}
for (k in 1:dim(d)[1])     {
  Input_CR_SPIN[k]= Yield_ctrl*AB_Yi_ratio* Ro_Sh_ratio* C_cont_root
  Input_GR[k]=1 #We do not have the data to estimate the total roots input, therefore we fitted a profile on the stock on the first layer and
we directly applied this function depending on stock. To facilitate model developpement we artificially fixed our inputs to 1.
}

for (k in 1:dim(d)[1])     {
  if (d[k,1]<=limit_grass)   {
    Input_CR[,k]=0
  }
  else                           {
    Input_CR[,k]= Yield_AF[,k]* AB_Yi_ratio* Ro_Sh_ratio* C_cont_root
  }
}

#Transport parameters
v=PARAM[3,1]/10000.
v_slow=v

Dt=PARAM[4,1]/10000.

#Distribution of mineralized C in the different pools
frac_SA=PARAM[6,1]
frac_AS=PARAM[7,1]

print("****************************************")
print("The parameters were read succesfully")

#Roots profil of tree (% of the total root mass) - from Cardinael et al. 2015 Pland and Soil
profil_TR_R<-matrix(ncol=dim(d)[1]+1,nrow=dim(z)[1]+1)
for (i in 1:dim(d)[1]) {profil_TR_R[1,i+1]<-d[i,]}
for (i in 1:dim(z)[1]) {profil_TR_R[i+1,1]<-z[i,]}

for (j in seq(z[1,1],z[dim(z)[1],1], by=-step_depth)){
  for (i in 1:dim(z)[1]+1) {
    if (profil_TR_R[i,1]==as.character(j)){
      for (k in 1:dim(d)[1]+1)     {
        if (profil_TR_R[1,k]<=limit_grass) {
          profil_TR_R[i,k]<-13.9169*exp((-1.38609)*(-z[i-1,1]))
        }
        else                                                                                                    {
          profil_TR_R[i,k]<-(10.306589-1.1505*(d[k-1,1]))*exp((-1.10282+0.18586*(d[k-1,1]))*(-z[i-1,1]))


        }
        profil_TR_R[i,k]<-profil_TR_R[i,k]/100 #Conversion from % to proportion
      }
    }
  }
}




#Roots profil of crop roots (% of the total root mass)
profil_CR_R<-matrix(ncol=dim(d)[1]+1,nrow=dim(z)[1]+1)
for (i in 1:dim(d)[1]) {profil_CR_R[1,i+1]<-d[i,]}
for (i in 1:dim(z)[1]) {profil_CR_R[i+1,1]<-z[i,]}

for (j in seq(z[1,1],z[dim(z)[1],1], by=-step_depth)){
  for (i in 1:dim(z)[1]+1) {
    if (profil_CR_R[i,1]==as.character(j)){
      for (k in 1:dim(d)[1]+1)     {
        profil_CR_R[i,k]<-26.443*exp((-2.6)*(-z[i-1,1]))
        profil_CR_R[i,k]<-profil_CR_R[i,k]/100 #Conversion from % to proportion
        if (profil_CR_R[i,1]<limit_root_crop) {profil_CR_R[i,k]<-0} # no more crop roots below 1.5m
      }
    }
  }
}
profil_CR_R_SPIN<-profil_CR_R
for (i in 1:dim(z)[1]+1) {
  for (k in 1:dim(d)[1]+1)     {
    if (profil_CR_R[1,k]<=limit_grass) {profil_CR_R[i,k]<-0} # no crop on the tree line
  }
}

#Roots profil of graminae under the tree line (% of the total root mass)
profil_GR_R<-matrix(ncol=dim(d)[1]+1,nrow=dim(z)[1]+1)
for (i in 1:dim(d)[1]) {profil_GR_R[1,i+1]<-d[i,]}
for (i in 1:dim(z)[1]) {profil_GR_R[i+1,1]<-z[i,]}

for (j in seq(z[1,1],z[dim(z)[1],1], by=-step_depth)){
  for (i in 1:dim(z)[1]+1) {
    if (profil_GR_R[i,1]==as.character(j)){
      for (k in 1:dim(d)[1]+1)     {
        if (profil_GR_R[1,k] <= limit_grass) {
          profil_GR_R[i,k]<-0.4365*exp(-3.121*(-z[i-1,1]))
        }
        else                                                        {
```

```
          profil_GR_R[i,k]<-0
        }
      }
    }
  }
}

for (i in 1:dim(z)[1]+1) {if (profil_GR_R[i,1]<limit_root_grass) {profil_GR_R[i,]<-0}}

print("************************************")
print("The root profiles were defined successfully")


##############################3 pools model#####################################
############################numerical simulation#################################

library("deSolve")

#############with 3 pools#########################
modelp3difft <- function(t, initial_state,parms){
  with (as.list(parms),
        {
          A <- initial_state[1:dim(z)[1]]
          S <- initial_state[(dim(z)[1]+1):((2*dim(z)[1]))]
          P <- initial_state[((2*dim(z)[1])+1):((3*dim(z)[1]))]

          #Fluxes in z direction
          FluxA <- Dt * (c(0,A)) / dz -D * diff(c(0,A,0)) / dz - c(0,Dmix) * diff(c(0,A,0)) / dz
          FluxS <- Dt * (c(0,S)) / dz -D_slow * diff(c(0,S,0)) / dz - c(0,Dmix) * diff(c(0,S,0)) / dz
          FluxP <- Dt * (c(0,P)) / dz -D_slow * diff(c(0,P,0)) / dz - c(0,Dmix) * diff(c(0,P,0)) / dz
          FluxA[1]=0.
          FluxS[1]=0.
          FluxP[1]=0.

          #Reaction
          Import<-import_tree_be*mr_tree + import_grass_be*mr_grass + import_crop_be*mr_crop
          dA=-diff(FluxA) + (e*ks*S*frac_SA + e*kp*P - kf* A* clay_func)* mf* tf  + Import
          dS=-diff(FluxS) + (frac_AS*kf*e*A*clay_func - ks*S)* mf* tf
          dP=-diff(FluxP) + ((1-frac_SA)*e*ks*S + (1-frac_AS)*kf*e*A*clay_func - kp*P)* mf* tf

          list(c(dA=dA,dS=dS,dP=dP
          ))
        })
}


initial_state<- c(rep(0,dim(z)[1]),rep(0,dim(z)[1]),rep(0,dim(z)[1]))
times <- seq(0,spin_length,by=1)


print("************************************")
print("Simulation starting")

out_spinup<-matrix(ncol=dim(d)[1],nrow=3*dim(z)[1])
out_intermediate<-matrix(nrow= spin_length,ncol=3*dim(z)[1])

#Calculation of the input
import_tree_ab=0
import_tree_be=0
import_crop_ab= Input_crop_ab_spin[1]
import_crop_be= profil_CR_R_SPIN[2:(dim(z)[1]+1),1+1]*Input_CR_SPIN[1]
import_grass_ab=0
import_grass_be=0

import_crop_be[1]=import_crop_be[1]+ import_crop_ab

#Mixing effect of tillage
Dmix<-rep(0,dim(z)[1])
for (j in 1:dim(z)[1])  { if (abs(z[j,1])<=abs(til_lay)) {Dmix[j]<-50} else {Dmix[j]<-0}}

#Moisture function on decomposition
mf<-mf_ctrl

#Calculation of the soil temperature function
tf<-Q10^((temp-304.15)/10)
for (j in 1:dim(z)[1]) {if (tf[j] > 1) {tf[j]<-1} else {tf[j]<-tf[j]}}

#Calculation of the clay function
if (control=="y") {clay_func<-ta_clay_func} else {clay_func<-af_clay_func}

#Lauching the simulation
kf=((import_tree_be)/(import_tree_be+import_crop_be+import_grass_be))*kt + ((import_crop_be+import_grass_be)/(import_tree_be
+import_crop_be+import_grass_be))*kw

kf[1]=((import_tree_ab+import_tree_be[1])/(import_tree_ab+import_tree_be[1]+
import_crop_ab+import_crop_be[1]+import_grass_ab+import_grass_be[1]))*kt +
((import_crop_ab+import_crop_be[1]+import_grass_ab+import_grass_be[1])/(import_tree_ab+import_tree_be[1]+
import_crop_ab+import_crop_be[1]+import_grass_ab+import_grass_be[1]))*kw

for (j in 1:dim(z)[1]) {if (is.na(kf[j])){kf[j]<-kw}}

parms <- c(e=e,ks=ks,mr_tree=mr_tree,mr_grass=mr_grass,mr_crop=mr_crop,kf=kf,D=v,cr=c,D_slow=v_slow,dz= step_depth,Dt=Dt, ad=1, mf =
mf,Dmix=Dmix, tf=tf,frac_AS=frac_AS,frac_SA=frac_SA,kp=kp)
initial_state<-c(rep(0,dim(z)[1]),rep(0,dim(z)[1]),rep(0,dim(z)[1]))

out<- ode.1D(y=initial_state, time=times, func=modelp3difft,parms=parms,nspec=2)
nb_param<-10


out_spinup[,1]<-out[spin_length,1:((3*dim(z)[1]))+1]
for (i in 1:dim(d)[1])  {out_spinup[,i]<-out_spinup[,1] }


out_for_optim<-rep(0,10)
```

```
out_for_optim[1]<-out_spinup[21,1]+out_spinup[41,1]
out_for_optim[2]<-out_spinup[22,1]+out_spinup[23,1]+out_spinup[42,1]+out_spinup[43,1]
out_for_optim[3]<-out_spinup[24,1]+out_spinup[25,1]+out_spinup[44,1]+out_spinup[45,1]
out_for_optim[4]<-out_spinup[26,1]+out_spinup[27,1]+out_spinup[46,1]+out_spinup[47,1]
out_for_optim[5]<-out_spinup[28,1]+out_spinup[29,1]+out_spinup[30,1]+out_spinup[48,1]+out_spinup[49,1]+out_spinup[50,1]
out_for_optim[6]<-out_spinup[31,1]+out_spinup[32,1]+out_spinup[51,1]+out_spinup[52,1]
out_for_optim[7]<-out_spinup[33,1]+out_spinup[34,1]+out_spinup[53,1]+out_spinup[54,1]
out_for_optim[8]<-out_spinup[35,1]+out_spinup[36,1]+out_spinup[55,1]+out_spinup[56,1]
out_for_optim[9]<-out_spinup[37,1]+out_spinup[38,1]+out_spinup[57,1]+out_spinup[58,1]
out_for_optim[10]<-out_spinup[39,1]+out_spinup[40,1]+out_spinup[59,1]+out_spinup[60,1]

out_for_optim<-out_for_optim*c(1,1/2,1/2,1/2,1/2,1/3,1/2,1/2,1/2,1/2,1/2,1/2) #(conversion in kg m3 for the optimization)
print("out_for_optim")
print(out_for_optim)

write.table(out_for_optim,'out_Mik_p0.txt',row.names=FALSE,col.names=FALSE)

print("**************************************")
print("Spinup finished")

#run agroforestery
out_final<-matrix(ncol=dim(d)[1],nrow=3*dim(z)[1])
out_intermediate_Agrof<-matrix(nrow= Agrof_length,ncol=3*dim(z)[1])
out_total <- array(0, dim=c(dim(d)[1], Agrof_length, 3*dim(z)[1]))
for (i in 1:dim(d)[1]) {
  if (i==1){print("**************************************");print("We start the band");print(i)} else {print("We start the band");print(i)}
  for (t in 1: Agrof_length) {
    if (t==1){print("**************************************");print("We start the year");print(t)} else {print("We start the
year");print(t)}

    #Calculation of the input
    if (control=="y") {
      import_tree_ab=0
      import_tree_be=0
      import_crop_ab= Input_crop_ab_spin[1]
      import_crop_be= profil_CR_R_SPIN[2:(dim(z)[1]+1),1+1]*Input_CR_SPIN[1]
      import_grass_ab=0
      import_grass_be=0

      import_crop_be[1]=import_crop_be[1]+ import_crop_ab
    }
    else {
      import_tree_ab= Input_leaves[time_Agrof_length [t]]
      import_tree_be=profil_TR_R[2:(dim(z)[1]+1),i+1]*Input_TR[time_Agrof_length [t],i]
      import_crop_ab= Input_crop_ab[t,i]
      import_crop_be=profil_CR_R[2:(dim(z)[1]+1),i+1]*Input_CR[t,i]
      import_grass_ab=Input_grass_ab[i]
      import_grass_be=profil_GR_R[2:(dim(z)[1]+1),i+1]*Input_GR[i]

      import_crop_be[1]=import_crop_be[1]+ import_crop_ab
      import_tree_be[1]=import_tree_be[1]+ import_tree_ab/mr_tree #we divided by mr_tree to take into account that leaves are product only
once per year
      import_grass_be[1]=import_grass_be[1]+ import_grass_ab
    }

    #Mixing effect of tillage
    Dmix<-rep(0,dim(z)[1])
    for (j in 1:dim(z)[1])      { if (abs(z[j,1])<=abs(til_lay)) {Dmix[j]<-50} else {Dmix[j]<-0}}
    if (control=="n") {    if (d[i,] <= limit_grass) {Dmix[]<-0} }

    #Calculation of the soil moisture function
    if (d[i,] <= limit_grass) {mf<-mf_tl} else{mf<-mf_ir}
    if (control=="y") {mf<-mf_ctrl} else {mf<-mf}
    #Calculation of the clay function
    clay_func<-af_clay_func

    #Lauching the simulation
    times <- seq(0,Agrof_length,by=1)

    kf=((import_tree_be)/(import_tree_be+import_crop_be+import_grass_be))*kt + ((import_crop_be+import_grass_be)/(import_tree_be
+import_crop_be+import_grass_be))*kw

    kf[1]=((import_tree_ab+import_tree_be[1])/(import_tree_ab+import_tree_be[1]+
import_crop_ab+import_crop_be[1]+import_grass_ab+import_grass_be[1]))*kt +
((import_crop_ab+import_crop_be[1]+import_grass_ab+import_grass_be[1])/(import_tree_ab+import_tree_be[1]+
import_crop_ab+import_crop_be[1]+import_grass_ab+import_grass_be[1]))*kw

    for (j in 1:dim(z)[1]) {if (is.na(kf[j])){kf[j]<-kw}}

    parms <- c(e=e,ks=ks,mr_tree=mr_tree,mr_grass=mr_grass,mr_crop=mr_crop,kf=kf,D=v,cr=c,D_slow=v_slow,dz= step_depth,Dt=Dt, ad=1, mf =
mf,Dmix=Dmix, tf=tf,frac_AS=frac_AS,frac_SA=frac_SA,kp=kp)
    if (t==1){initial_state<-out_spinup[,i]} else {initial_state<-out_agrof[2,2:(3*dim(z)[1]+1)]}

    out_agrof<- ode.1D(y=initial_state, time=times, func=modelp3difft,parms=parms,nspec=2)
    nb_param<-10
    out_intermediate_Agrof[t,]<-out_agrof[2,2:((3*dim(z)[1]+1))]
  }
  out_final[,i]<-out_intermediate_Agrof[Agrof_length,1:((3*dim(z)[1]))]
  out_total[i,,]<-out_intermediate_Agrof[1:Agrof_length,1:((3*dim(z)[1]))]
}
print("**************************************")
print("Simulation finished")
```