

# The Moisture Response of Soil Heterotrophic Respiration: Interaction with Soil Properties

Moyano et al.

## Supplementary Figures:

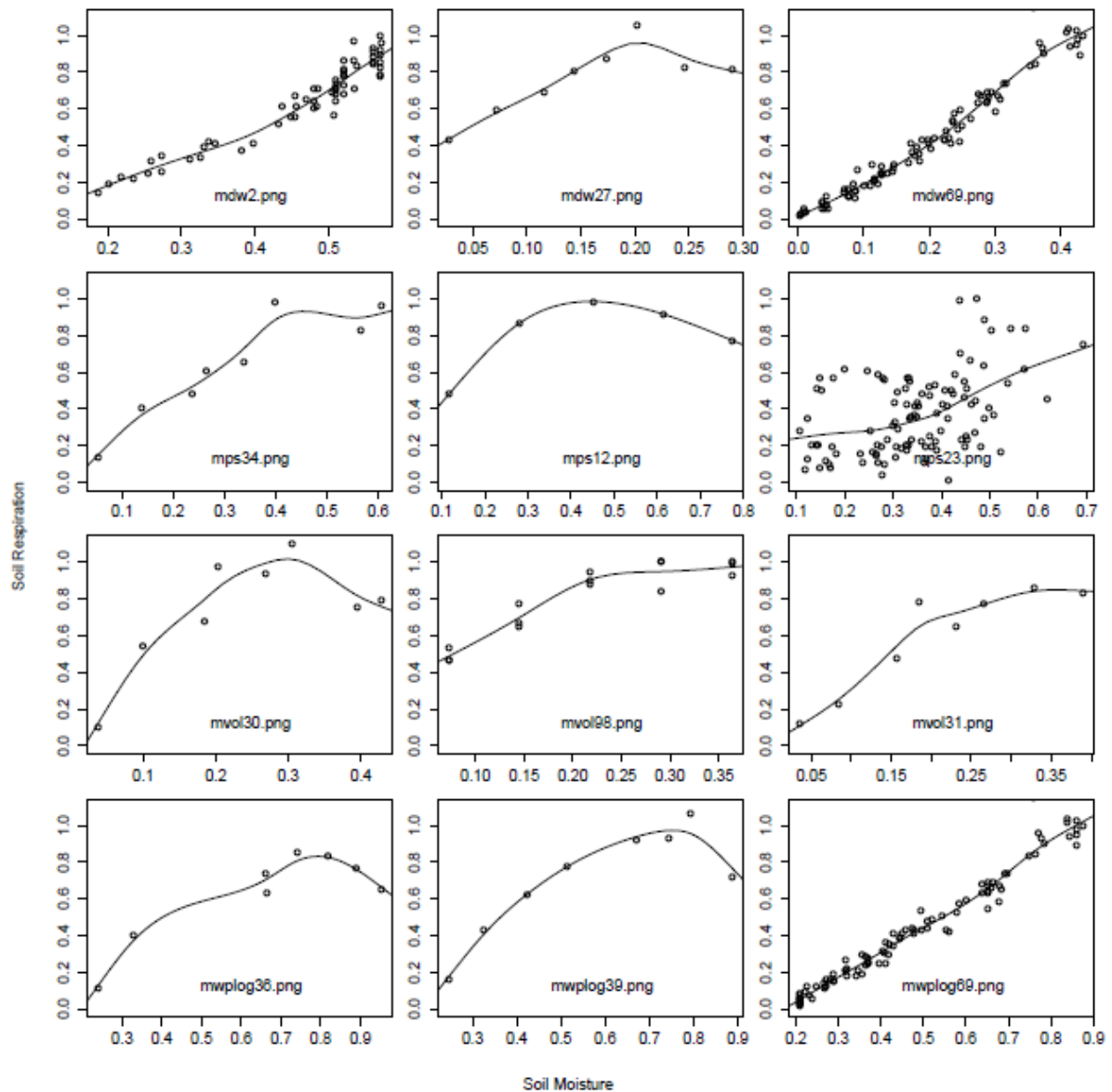


Figure S1: Examples of curve fitting using generalized additive models (GAMs) on datasets relating soil moisture to heterotrophic respiration. Units are not relevant so are not shown.

## R-Code

```
# =====
# ===== R Code for the Analysis of Soil Moisture-Respiration Data =====
# =====
# Note: this analysis also includes % water holding capacity as a measure: not in the paper given the low
number of datasets

# clear working space
rm(list=ls())
# ===== Prepare Data
# Read in the data files (MRD = moisture respiration data; DD = data description; funs = function indexes)
MD<-read.csv("MRD.txt")
DD<-read.csv("DD.txt")
funs<-read.csv("funs.txt")
# create the description data.frame by aggregating and binding
DD1<-aggregate(x = MD, by = list(MD$id), FUN = "mean")
DD1<-subset(DD1,select=c(soil.t,bd:pd,co2.dw))
DD<-cbind(DD, DD1)
rm(DD1)
# Transform the mwplog to a practical scale (kPa/5)
MD$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=mo
d.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+l(xdata^2)) else
      if (funid==3) mod <- lm (ydata~xdata+l(xdata^2)+l(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=pr
ed.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 high organic carbon soils

library(MASS)
#===== Prepare the data
lmdata<-list()
DDsub<- subset(DD, select=c(id,ecosystem,bd,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 intervals)
moist<-rep(1:110,each=length(DD[,1])); moist<-moist/100 # moisture variable: repeat each value by the
amount of datasets
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 outliers
  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(3,4,5,8,9)) {z<-as.vector(!is.na(Data[,i])); x<-x*z}
  # x[Data$ecosystem=="Forest"]<-0 # select ecosystem?
  x[Data$c.org>0.05]<-0 # remove high c.org soils
  Data<-Data[x==1,]
  lmdata[j]<-list(Data)
}

```

```

# Fit Linear Models
lm.soilmin<-list()
lm.moist<-list()
for (j in c(1,2,3,4,5)) {
  lmmod<-stepAIC(lm(pcsr~moist+l(moist^2)+l(moist^3), data=lmdata[[j]]),k=log(length(lmdata[[j]]$pcsr)))
  lm.moist[j]<-list(lmmod)
  lmmod<-stepAIC(lm(pcsr~moist+l(moist^2)+l(moist^3)+moist*bd+c.org+moist*clay,
data=lmdata[[j]]),k=log(length(lmdata[[j]]$pcsr)))
  lm.soilmin[j]<-list(lmmod)
}
rm(i,lmmod,rm,rsd,Data,DDrep,x,z,j)

# =====
#===== 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)) {z<-as.vector(!is.na(Data[,i])); x<-x*z}
  # 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+l(moist^2)+l(moist^3),
data=lmdata.texorg[[j]]),k=log(length(lmdata.texorg[[j]]$pcsr)))
  lm.moist.texorg[j]<-list(lmmod)
  lmmod<-stepAIC(lm(pcsr~moist+l(moist^2)+l(moist^3)+moist*clay+c.org,
data=lmdata.texorg[[j]]),k=log(length(lmdata.texorg[[j]]$pcsr)))
  lm.texorg[j]<-list(lmmod)
}
rm(i,j,z,x,rsd,rm,Data,lmmod,DDrep)

# =====
#===== Multiple Linear Regression =====

```

```

#===== linear model for high c.org soils

library(MASS)

lm.hcorg<-list() # a list to save fitted models
lmdata.hcorg<-list()

#===== Prepare the data
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,bd,c.org,maxtime,soil.t,co2.dw)) # 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])
  rmsd<-mean(pcsr,na.rm=TRUE); rsd<-sd(pcsr,na.rm=TRUE); pcsr[pcsr>rms+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(4,8,9)) {z<-as.vector(!is.na(Data[,i])); x<-x*z}
# x[Data$ecosystem=="Unknown"]<-0 # select ecosystem?
x[Data$c.org<0.05]<-0 # remove low c.org soils
  Data<-Data[x==1,]
  lmdata.hcorg[j]<-list(Data)
}
rm(Data,j,x,z,rms,rsd,DDrep)

#===== Fit Linear Models
for (j in c(1,2,3,4,5)) {
  lmod<-stepAIC(lm(pcsr~moist+l(moist^2)+l(moist^3),
data=lmdata.hcorg[[j]],k=log(length(lmdata.hcorg[[j]]$pcsr)))
  lm.hcorg[j]<-list(lmod)
}
rm(j,i,lmod)

# =====
# ===== Plot PCSR values: FIGURE 1 =====

#===== Get data ready
# calculate the mean PCSR
mPCSR<-lapply(PCSR, function(x) apply(x,2,function (y) mean(na.exclude(y))))
# calculate stdev
PCSR.sd<-lapply(PCSR, function(x) sd(x,na.rm=TRUE))
# calculate the +/- 1 stdev values
stdev.u<-list()
stdev.l<-list()
for (i in 1:5) {
  stdev.u[i]<-list(mPCSR[[i]]+PCSR.sd[[i]])
  stdev.l[i]<-list(mPCSR[[i]]-PCSR.sd[[i]])
}
# calculate the confidence intervals

```

```

cint<-lapply(PCSR, function(x) apply(x,2,function (y) qt(0.05,sum(!is.na(y))-1,lower.tail=0) *
sd(na.exclude(y))/sqrt(sum(!is.na(y)))) )
uconf<-list()
lconf<-list()
stdev<-list()
for (i in 1:5) {
  uconf[i]<-list(mPCSR[[i]]+cint[[i]])
  lconf[i]<-list(mPCSR[[i]]-cint[[i]])
}
# ===== Get plot variables ready =====
moist<-rep(1:110,each=length(DD[,1]))
# labels for the axis
col<-260
# color specific datasets?
# col<-rep(DD$c.org,110)
# col[col>0.05]<-32
# col[col<=0.05]<-260
#==== Make plots =====
pdf(file="figure1.pdf",bg="white",height=6.5,width=3.5)
par(mfrow=c(4,1),mar=c(0,0,0,0), oma=c(4.4,4,2.5,1), cex=0.7, font.main=1, bg = "white", tcl=0.3)
atxlab<-c(0,20,40,60,80,100); xlab <-c(0.0,0.2,0.4,0.6,0.8,1.0)
atylab <-c(0.6,0.7,0.8,0.9,1.0,1.1,1.2,1.3,1.4,1.5,1.6); ylab <-c(0.6,NA,0.8,NA,1.0,NA,1.2,NA,1.4,NA,1.6)
for (i in c(1,2,3,5)) {
  yl<-c(0.9,1.5)
  ydata<-as.vector(PCSR[[i]][,1:110])
  plot(ydata~moist, type="p", yaxt="n", xaxt="n", xlab = "", ylab = "", col="grey70",cex=0.5, lwd=0.5, ylim=yl)
  lines(rep(1,110),lty=1,col="grey30",lwd=0.8)
  lines(mPCSR[[i]], type="l", col="black", lwd=0.8)
  lines(stdev.l[[i]],lty=2, lwd=0.7)
  lines(stdev.u[[i]],lty=2,lwd=0.7)
  # lines(uconf[[i]],lty=2)
  # lines(lconf[[i]],lty=2)
  axis(2, at=atylab, labels=ylab, padj=1)
  # text (x=55,y=c(1.47,1.47,1.47,NA,1.47)[i], labels=list(expression(paste("a
(",theta[m],")")),expression(paste("b (",theta[v],")")),expression(paste("c
(",theta[s],")")),NA,expression(paste("d (",psi[log],")")))[i]), cex=1, font=2)
  text (x=110,y=c(1.47,1.47,1.47,NA,1.47)[i], labels=c("a","b","c",NA,"d")[i], cex=1, font=2)
  text (x=55,y=c(1.47,1.47,1.47,NA,1.47)[i],
labels=list(expression(theta[m]),expression(theta[v]),expression(theta[s]),NA,expression(psi[log])))[i]), cex=1.1,
font=2)
  if (i==5) {
    title (xlab="Soil Moisture", outer=TRUE, line=1.5)
    title (ylab="Proportional Response of Soil Respiration", outer=TRUE, line=2)
    axis(1, at=atxlab, labels=xlab, padj=-1.1)
  }
}
dev.off()
rm(yl, col, moist, i, PCSR.sd, cint, lconf, mPCSR, stdev, stdev.l, stdev.u, uconf, ydata, atylab, ylab)

# =====
# ===== PLOT Correlation coefficients: FIGURE 2 =====

#==== Make lists for assigning names
color<-c("black","blue","orange3", "brown")
#===== get the parameters of the models

```

```
# Define some functions to get model parameters from the list (lmsep) of lists of models, using sapply-mapply
below
```

```
lmfun.r2<- function(A) { sapply(A, function (x) {y <- summary(x); return (as.numeric(y[8]))}) }
lmfun.coef <- function (A) { sapply(A, function (x) {return (x[1])}) }
lmfun.f <- function (A) { sapply(A, function (x) {y <- summary(x); return (y[10])}) }
lmfun.p <- function (A) { sapply(A, function (x) {if (length(x)>2) 1-pf(x["value"],x["numdf"],x["dendf"]) else
return (NA)}) }
lmfun.cc <- function (A,B) { mapply (function (x,y) {sqrt(x)*sign(as.numeric(y[2]))}, A, B) }
lmsep.r2 <- lapply (lmsep, lmfun.r2)
lmsep.coef <- lapply (lmsep, lmfun.coef)
lmsep.f <- lapply (lmsep, lmfun.f)
lmsep.p <- lapply (lmsep, lmfun.p)
lmsep.cc <- mapply (FUN=lmfun.cc, lmsep.r2, lmsep.coef, SIMPLIFY = FALSE)
lmsep.p1 <- lmsep.p
```

```
# ===== Make Plots
```

```
moistlab <- list(expression(paste("Gravimetric Moisture (" ,theta[m],")")),expression(paste("Volumetric
Moisture (" ,theta[v],")")),expression(paste("Water Saturation (" ,theta[s],")")), "Fraction of Water Holding
Capacity",expression(paste("Water Potential (" ,psi[log],")")))
```

```
colval<-lmsep.p
```

```
for (i in 1:30) {
```

```
  lmsep.p1[[i]][lmsep.p1[[i]]>0.05] <- NA
  colval[[i]][colval[[i]]>0.1] <- 1
  colval[[i]][colval[[i]]<=0.1] <- 24
  colval[[i]][colval[[i]]==24]<-rep(c(163,257,26,552,501,142),times=5)[i]
}
```

```
pdf(file="figure2.pdf", bg="white", width=6, height=5)
```

```
par(mfrow=c(2,2),mar=c(3.5,2,0,0), oma=c(0,2,1,2),font.main=1, bg = "white",tcl=0.3)
```

```
for (i in c(1,7,13,25)) {
```

```
  par(cex=0.7,lwd=0.7)
  if (i<7) xlc<-c(0,70) else if (i>6 & i<13) xlc<-c(0,60) else xlc<-c(0,100)
  xscale<-seq(2,100,2)
  plot(xl,c(-1,1),type="n",axes=FALSE,xlab="",ylab="")
  par(cex=0.5,lwd=0.4)
  points(xscale, lmsep.cc[[i]],col="gray10", pch=21, bg=colors()[colval[[i]]])
  points(xscale, lmsep.cc[[i+1]],col="green3", pch=21,bg=colors()[colval[[i+1]]])
  points(xscale, lmsep.cc[[i+2]],col="blue", pch=21,bg=colors()[colval[[i+2]]])
  points(xscale, lmsep.cc[[i+3]],col="red", pch=21,bg=colors()[colval[[i+3]]])
  points(xscale, lmsep.cc[[i+4]],col="orange3", pch=21,bg=colors()[colval[[i+4]]])
  points(xscale, lmsep.cc[[i+5]],col="gold", pch=21,bg=colors()[colval[[i+5]]])
```

```
  par(cex=0.7,lwd=0.7)
```

```
  lines(rep(0,100), col="grey")
```

```
  box()
```

```
  xls<-seq(0,1,0.2) #if (i==25) xls<-seq(-5,0,1)
```

```
  axis(2,at=c(-1,0,1),labels=c(-1,0,1), lwd=0.7,padj=1.5)
```

```
    axis(1,at=seq(0,100,20),labels=xls, lwd=0.7, padj=-1.5)
```

```
  if (i==1) title (ylab="Correlation Coefficient", outer=1,line=0)
```

```
    title(main="",xlab=moistlab[[ceiling(i/6)]], line=1.5)
```

```
  text (x=1, y=0.95, labels=rep(c("a","b","c",NA,"d"),each=6)[i], cex=1.1, font=2)
```

```
}
```

```
while (dev.cur()>1) dev.off()
```

```
rm(i,color,moistlab,xl,xls,xscale, colval)
```

```
rm(lmsep.cc, lmsep.coef, lmsep.f, lmsep.p, lmsep.r2, lmfun.cc, lmfun.coef, lmfun.f, lmfun.p,
```

```
lmfun.r2,lmsep.p1)
```

```
# =====
```



```
# ===== PLOT CURVES: FIGURE 3 =====
```

```
# ===== model and measures used
```

```
model<-lm.soilmin
```

```
m1<-1
```

```
m2<-2
```

```
m3<-3
```

```
m4<-5
```

```
# ===== create dataframe for newdata
```

```
simdata<-data.frame(moist=seq(0.02,1.2,0.02))
```

```
for (j in c(m1,m2,m3,m4)) {
```

```
  # ===== predict and simulate changing clay
```

```
  claymat<-matrix(nrow=10,ncol=60)
```

```
  for (k in 1:10) {
```

```
    # simdata$ecosystem<-as.factor("Cultivated")
```

```
    simdata$bd<-1.2
```

```
    simdata$c.org<-0.02
```

```
    simdata$clay<-k/10
```

```
    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)
```

```
    claymat[k,]<-y
```

```
    # scale values to start at 0
```

```
    if (j!=5) {
```

```
      ind<-which.max(claymat[k,])
```

```
      claymat[k,1:ind]<-(claymat[k,1:ind]-min(claymat[k,1:ind]))
```

```
      claymat[k,1:ind]<-claymat[k,1:ind]/max(claymat[k,1:ind]) }
```

```
  }
```

```
  # ===== predict and simulate changing bd
```

```
  bdmat<-matrix(nrow=10,ncol=60)
```

```
  for (k in 1:10) {
```

```
#    simdata$ecosystem<-as.factor("Cultivated")
```

```
    simdata$bd<-k/10+0.5
```

```
    simdata$c.org<-0.02
```

```
    simdata$clay<-0.3
```

```
    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)
```

```
    bdmat[k,]<-y
```

```
    # scale values to start at 0
```

```
    if (j!=5) {
```

```
      ind<-which.max(bdmat[k,])
```

```
      bdmat[k,1:ind]<-(bdmat[k,1:ind]-min(bdmat[k,1:ind]))
```

```
      bdmat[k,1:ind]<-bdmat[k,1:ind]/max(bdmat[k,1:ind]) }
```

```
  }
```

```
}
```

```
# ===== predict and simulate changing corg
```

```
corgmat<-matrix(nrow=10,ncol=60)
```

```
for (k in 1:10) {
```

```
#    simdata$ecosystem<-as.factor("Cultivated")
```

```
    simdata$bd<-1.2
```

```

simdata$c.org<-k/200
simdata$clay<-0.3
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])
}
}
# Make Plot =====
# moistnames<-list(expression(theta[m]), expression(theta[v]),
expression(theta[s]),"WHC",expression(psi[log]))
moistnames <- list("Gravimetric Moisture", "Volumetric Moisture", "Water Saturation", "Fraction of Water
Holding Capacity", "Water Potential")

if (j==m1) {
  pdf(file="figure3.pdf", bg="white", width=5, height=4)
  par(mfcol=c(3,4),mar=c(0,0,0,0), oma=c(4.6,4.6,2.5,1), cex=0.5, font.main=1, bg = "white",tcl=0.3,lwd=0.7)
}
x<-seq(0.02,1.2,0.02)
y<-seq(0.02,1.2,0.02)
xmax <- 1; if (j==1 | j==2) {xmax<-0.6} else if (j==5) {xmax<-1}
xmin <- 0
if (j==m1) letind<-1; if (j==m2) letind<-4; if (j==m3) letind<-7; if (j==m4) letind<-10
letter<-c("a","e","i","b","f","j","c","g","k","d","h","l")
if (j==5) xpos<-0.7; if (j==1 | j==2) xpos<-0.3; if (j==3) xpos<-0.5
atxlab<-c(0.0,0.2,0.4,0.6,0.8,1.0)
if (j==5) xlab<-c(-5,-4,-3,-2,-1,0) else xlab <-c(0.0,0.2,0.4,0.6,0.8,1)# if (j==1) xlab <-c(0.0,0.2,0.4,0.6,NA,NA,NA)
else
  atylab <-c(0.2,0.4,0.6,0.8,1.0)
  ylab <-c(0.2,0.4,0.6,0.8,1.0)
  if (j==m1) a<-0.03; if (j==m2) a <-0.35; if (j==m3) a <-0.65; if (j==m4) a <- 0.945
  cx<-1.1

z<-claymat
plot (z[1,]~simdata$moist, type="n", bg="white", yaxt="n", xaxt="n", xlab = "", ylab = "", ylim=c(0,1),
xlim=c(xmin,xmax))
for (i in 1:length(z[,1])) {c<-253-i*9; lines(z[i,]~simdata$moist,type="l", lwd=1, lty=1, col=colors()[c]) }
if (j==m1) axis(2, at=atylab, labels=ylab, padj=1, lwd=0.7)
text (x=xpos, y=0.1, labels=letter[letind], font=2, cex=cx)

z<-corgmat
plot (z[1,]~simdata$moist,type="n", bg="white", yaxt="n", xaxt="n", yaxt="n", xlab = "", ylab = "", ylim=c(0,1),
xlim=c(xmin,xmax))
for (i in 1:length(z[,1])) {c<-253-i*9; lines(z[i,]~simdata$moist,type="l", lwd=1, lty=1, col=colors()[c]) }
if (j==m1) axis(2, at=atylab, labels=ylab, padj=1, lwd=0.7)
text (x=xpos, y=0.1, labels=letter[letind+1],cex=cx, font=2)

```

```

z<-bdmat
plot (z[,1]~simdata$moist,type="n", bg="white", yaxt="n", xaxt="n", yaxt="n", xlab = "", ylab = "", ylim=c(0,1),
xlim=c(xmin,xmax))
for (i in 1:length(z[,1])) {c<-253-i*9; lines(z[,i]~simdata$moist,type="l", lwd=1, lty=1, col=colors()[c]) }
if (j==m1) axis(2, at=atylab, labels=ylab, padj=1, lwd=0.7)
text (x=xpos, y=0.1, labels=letter[letind+2], cex=cx, font=2)
axis(1, at=atxlab, labels=xlab, padj=-1, lwd=0.7)

title(xlab = moistnames[j], outer=TRUE, adj=a, line=2, font.lab=1, cex.lab=cx)
title(ylab = "Relative Soil Respiration", outer=TRUE, cex.lab=cx, font.lab=1, line=2.1)

}
dev.off()
#
rm(m1,m2,m3,xpos,ylab,xmax,xmin,xlab,z,moistnames,j,i,c,k,a,letter,letind,ind,atxlab,atylab,model,simdata,x,
y,claymat,bdmat,corgmat,cx)

# =====
# ===== PLOT Respiration Functions: FIGURE 4 =====
# plots model of UK soil series area against reduction functions from other models

for (j in c(3,5)) {
  model<-lm.texorg
  moistlab <- list(expression(paste("Gravimetric Moisture (" ,theta[m],")")),expression(paste("Volumetric
Moisture (" ,theta[v],")")),expression(paste("Water Saturation (" ,theta[s],")")), "Fraction of Water Holding
Capacity",expression(paste("Water Potential (" ,psi[log], ")"))))
  UKsoils<-read.csv("UKsoils.csv")
  data<-UKsoils[UKsoils$c.org<=0.05,]
  data<-data[order(data$area, decreasing=1),]
  RRmatrix<-matrix(nrow=length(data[,1]),ncol=120)
  # create dataframe for newdata
  simdata<-data.frame(moist=seq(0.01,1.2,0.01))
  for (k in 1:length(data[,1])) {
#   simdata$ecosystem<-rep(as.factor(data[k,1]),120)
    simdata$c.org<-rep(data[k,2],120)
    simdata$clay<-rep(data[k,3],120)
    # predict and simulate respiration
    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)
    RRmatrix[k,]<-y
    # to make all curves start at 0
    if (j!=5) {
      ind<-which.max(RRmatrix[k,])
      RRmatrix[k,1:ind]<-(RRmatrix[k,1:ind]-min(RRmatrix[k,1:ind]))
      RRmatrix[k,1:ind]<-RRmatrix[k,1:ind]/max(RRmatrix[k,1:ind])
    }
  }
}
# ===== PLOTS =====
mini<-1 # select the range in the data
maxi<-100
mini2<-mini/100

```

```

maxi2<-maxi/100
# === make data for plotting area
RRmin<-apply(RRmatrix[,mini:maxi],2,FUN=function(x){min(x)})
RRmax<-apply(RRmatrix[,mini:maxi],2,FUN=function(x){max(x)})
RRmax<-RRmax[seq(length(RRmax),1,-1)]
RRminmax<-c(RRmin,RRmax)
polymoist<-c(seq(mini2,maxi2,0.01),seq(maxi2,mini2,-0.01))
# === make plots
if (j==3) {
  pdf(file="figure4.pdf", bg="white", width=6, height=3)
  par(mfcol=c(1,2),mar=c(4,0,1,0), oma=c(0,4,0,1), cex=0.7, font.main=1, bg = "white", tcl=0.3,lwd=0.8)
  plot (simdata$moist~simdata$moist,type="n", xaxt="n", yaxt="n", xlab = "", ylab = "", ylim=c(0,1),
xlim=c(0,1),lwd=0.8) #xlim=c(0, max(data[[j]]$moist))#min(data[[j]]$moist)
  title(ylab = "Relative Soil Respiration", outer=1, adj=0.6, line=1.8)
  text (x=0.03, y=0.97, labels="a", cex=1.1, font=2)
} else {
  plot (simdata$moist~simdata$moist,type="n", yaxt="n", xaxt="n", xlab = "", ylab = "", ylim=c(0,1),
xlim=c(0,1.1)) #xlim=c(0, max(data[[j]]$moist))#min(data[[j]]$moist)
  text (x=0.03, y=0.97, labels="b", cex=1.1, font=2)
}
atxlab<-c(0.0,0.2,0.4,0.6,0.8,1.0)
if (j==1) xlab <-c(0.0,0.2,0.4,0.6,NA,NA) else if (j==5) xlab<-c(-5,-4,-3,-2,-1,0) else xlab <-c(0.0,
0.2,0.4,0.6,0.8,1.0)
atylab<-seq(0,1,0.2); ylab<-seq(0,1,0.2)
title(xlab = moistlab[j], line=1.5)
axis(1, at=atxlab,labels=xlab, padj=-1.5,lwd=0.8)
if (j==3) axis(2, at=atylab,labels=ylab, padj=1.5,lwd=0.8)
polygon(polymoist,RRminmax,col="grey", border=NA)
# === plot lines of other models
if (j!=5) {
  SMRFcandy<-function(x) {if (x<=0.5) {4*x*(1-x)} else {1}}
  RRcandy<-sapply(simdata$moist, FUN=SMRFcandy)
  SMRFbethy<-function(x){x}
  RRBethy<-sapply(simdata$moist, FUN=SMRFbethy)
  SMRFsimcycle<-function(x){min(x/(0.15+x)+0.2,(1-x)/(0.08+(1-x))+0.2)}
  RRsimcycle<-sapply(simdata$moist, FUN=SMRFsimcycle)
  SMRFcentury<-function(x) { 1 / (1 + 4 * exp(-6*x)) }
  RRcentury<- sapply(simdata$moist, FUN=SMRFcentury)
  SMRFrothc<-function(x) { min (0.2 + 0.8 * (x/0.556),1) }
  RRrothc<- sapply(simdata$moist, FUN=SMRFrothc)
  lines(RRcandy~simdata$moist,type="l", lwd=1.5, lty=1, col=1)
  lines(RRBethy~simdata$moist,type="l", lwd=1.5, lty=2, col=1)
  lines(RRsimcycle~simdata$moist,type="l", lwd=2, lty=3, col=1)
  lines(RRrothc~simdata$moist,type="l", lwd=2, lty=4, col=1)
} else {
  SMRFdaisy<-function(x) {if(x<1.2 & x>0.9){1-0.4*(x-0.9)/0.3} else if (x<=0.9 & x>0.7) {1} else if (x<=0.7 & x>
0.1) {1*(x+0.1)/0.8} else {0} } # use (-pF)+1/5+1 to calculate what x value is in pF
  RRdaisy<-sapply(simdata$moist, FUN=SMRFdaisy)
  SMRFsoilco2<-function(x) { if (x>0.8) {1} else if (x<=0.8 & x>-0.2) {x+0.2} else {0} } # use (-pF)+1/5+1 to
calculate what x value is in pF
  RRsoilco2<-sapply(simdata$moist, FUN=SMRFsoilco2)
  lines(RRdaisy~simdata$moist,type="l", lwd=1.5, lty=1, col=1)
  lines(RRsoilco2~simdata$moist,type="l", lwd=1.5, lty=2, col=1)
}
}
dev.off()

```

```
rm(model, mini, maxi, data, i, j, x, y, k, simdata, RRdaisy, RRsoilco2, RRmatrix, RRminmax, RRmin, RRmax,  
polymoist, moistlab, atxlab, xlab, ind, maxi2, mini2)
```