1D Model

Anne-Karin Cooke

## Experiments

1. alpha=0.6, lambda=0.1 # dry
2. alpha=1.5, lambda=0.1
3. alpha=1.5, lambda=1.0
4. alpha=0.6, lambda=1.0 # wet

In this version, Sandy Clay Loam was used.

# Model set up

In this model, salt enters the system through rainfall and from the groundwater table with the same concentration.

## Constants

Soil properties were derived from standard Australian soils in Neurotheta (Minasny and McBratney, 2002, as cited in Shah et al, 2011).

# Sandy Clay Loam  
 n<-0.367 # porosity  
 # more soil variables for evaporation & losses  
 # Hydraulic conductivity  
 K\_s<-52.08\*10 # mm/day  
 # campbell's b  
 b<-6.4069 # neurotheta sandy clay loam  
 # van Genuchten parameters  
# avg <- 0.0521  
# nvg <- 1.237  
 s\_fc<-0.2677/n # Field capacity  
 # This is the bubbling pressure  
 psi\_s\_bar<--1.2E-3 #  
   
 h1bar = -psi\_s\_bar   
 hb = psi\_s\_bar\*-10^5 # mm  
   
  
 Z =10000 # [mm] actual groundwater depth   
  
# parameters describing the soil  
 soilpar <- list(b = b, n = n, s\_fc = s\_fc, K\_s = K\_s,   
 psi\_s\_bar = psi\_s\_bar, h1bar = h1bar, hb = hb)  
 #................................................  
 # Vegetation 1 (Grass)  
 # paspalum secateum F-I and R-I, 2004  
  
 Zr = 400 # soil depth (mm) Check Also Table 2...Fernandez-Illescas and Rodriguez-Iturbe...2001  
  
  
 # parameters describing the root zone   
 vegpar <- list(Zr = Zr)  
  
  
# parameters describing plant dynamics and salt features  
  
 alpha\_i=1 #maximum infiltration rate per day (K\_s and therefore soil type dependency in balances function code)  
  
 k=12 # Saco et al, 2013  
 W0=0.2 # Saco et al, 2013  
 gmax=0.05 # Saco et al, 2013  
 k1=5 # Saco et al, 2013  
 c=10 # Saco et al, 2013  
 f= 0.8 # f is the soil salt leaching efficiency (whether some salt is retained)  
 ConcConst = 0.1 # ConcConst is the concentration of the salt in the infiltrating water in g/l  
 CM.gw = 0.1 # salt concentration in groundwater  
 d=0.24 # fraction of plant mortality  
   
 par <- list(alpha\_i=alpha\_i,k=k, W0=W0, gmax=gmax, k1=k1, c=c, f=f, ConcConst=ConcConst, CM.gw= CM.gw, d=d)

## Infiltration function

Infil <- function(h,P, par){  
   
 I=par$alpha\_i\*h\*(P+par$k\*par$W0)/(P+par$k)  
   
 return (I)  
}

## Water uptake function

WU <- function(M,P,par){   
# using Svir in here means scaling Svir back to M, easier to do at Svir in balances   
# WU=par$gmax\*((M\*(1+Svir))/(((M\*(1+Svir))+par$k1)))\*P   
 WU=par$gmax\*(M/((M+par$k1)))\*P   
   
 return(WU)  
}

## Vertical Flux function

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# based on Salvucci, 1993  
L\_n <- function(M,Z,soilpar,vegpar) {  
 Zr <- vegpar$Zr  
 hb <- - soilpar$psi\_s\_bar\*10^5 # (mm?)  
 #soilpar$s\_fc <- (Z/hb)^(-1/soilpar$b)  
   
 s=M/(soilpar$n\*vegpar$Zr)  
   
 psi = hb\*s^-soilpar$b  
   
 m=2 + 3/soilpar$b#5.64 # in Salvucci's paper it is called n, but I called it m here to not confuse it with porosity  
  
  
q <-((Z/hb)^(-m)-(psi/hb)^(-m))/(1+(psi/hb)^(-m)+(m-1)\*(Z/hb)^(-m))#/(soilpar$n\*Zr) #   
  
# Mass flux  
  
flux <- soilpar$K\_s\*q  
  
return(flux)  
}

Precip <- function(time,alpha,lambda,delta) {  
 # generate a vector of times between rainfall events > delta  
 f\_P<-floor(rexp(time,lambda\*exp(-delta/alpha))) # vector of times between rainfall occurrences (equation 4 & 8)  
 # generate a binary vector from this (impulse function)  
 binary.vec <- unlist(lapply(1:time,function(i) c(rep(0,f\_P[i]),1)))  
 R <- rexp(length(binary.vec),1/alpha)\*binary.vec   
 return(R[1:time])  
}

## Plant growth function

Gr <- function(M,P,par) {   
   
 Gr = par$c\*WU(M,P,par)  
   
 return(Gr)  
}

## Plant mortality function

Mo <- function(P,M,Svir,par) {  
 # needs to be M/Svir because both are "large" numbers  
 # you want a number ~1 for multiplication, or <0.1 for addition  
 Mo = P\*(par$d\*(M/Svir))  
   
 return(Mo)  
  
}

## balances function

balances <- function(Rain, par, plotit=T,  
 soilpar,  
 vegpar){  
   
  
   
# Storage vectors for the daily steps are initialized.  
  
 M <- rep(0,length(Rain)) # soil moisture [mm]  
 h <- rep(0,length(Rain)) # infiltration depth [mm]  
 P <- rep(0,length(Rain)) #biomass density []  
 CM<- rep(0,length(Rain)) # Salt concentration in soil water in g/L or g/mm  
 SmI<- rep(0,length(Rain)) # Salt mass in infiltrating water [g]  
 SmM <- rep(0,length(Rain)) # Salt mass in soil water [g]  
 In <- rep(0,length(Rain)) # infiltration [mm]  
  
 Svir <- rep(0,length(Rain)) # virtual saturation  
 flux<- rep(0,length(Rain)) # drainage and capillary rise flux, according to sign  
  
  
  
# Initial values to start the simulation.  
  
   
 M[1] <- 10  
 h[1] <- 10   
 P[1] <- 30  
 CM[1]<- 0  
 Svir[1] <- M[1]  
  
  
  
# We decided to split the numerical calculations for the daily into 12 substeps.  
  
 deltat <- 12 # split in 12 increments  
  
   
# Storage vectors for the substeps are initialized.  
  
 M\_sub <- rep(0,deltat)  
 h\_sub <- rep(0,deltat)  
 I\_sub <- rep(0,deltat)  
 WU\_sub <-rep(0,deltat) # Water uptake in mm  
 P\_sub <- rep(0,deltat)   
 Gr\_sub <- rep(0,deltat) # Growth of biomass  
 Mo\_sub<- rep(0,deltat) # Mortality of biomass  
 SmI\_sub <- rep(0,deltat)   
 SmM\_sub<- rep(0,deltat)   
 CM\_sub<- rep(0,deltat)   
 Svir\_sub <- rep(0,deltat) # virtual saturation  
  
 flux\_sub<-rep(0,deltat) # calculates leakage loss without evaporation loss  
  
 U\_salt <-rep(0,deltat) # Salt mass rising  
 L\_salt <-rep(0,deltat) # salt mass drained  
  
  
   
 timeincr= 1/deltat  
   
 for (t in 2:length(Rain)){  
   
 for (tt in 1:(deltat-1)) {  
   
 h.old <- ifelse(tt==1,h[t-1],h\_sub[tt])  
 P.old <- ifelse(tt==1,P[t-1],P\_sub[tt])  
 M.old <- ifelse(tt==1,M[t-1],M\_sub[tt])  
 SmI.old <-ifelse(tt==1,SmI[t-1],SmI\_sub[tt])  
 CM.old <-ifelse(tt==1,CM[t-1],CM\_sub[tt])  
 Svir.old <-ifelse(tt==1,Svir[t-1],Svir\_sub[tt])  
  
  
 # Balance for water depth on soil  
 h\_sub[tt+1] <- h.old + ifelse(tt==1,(10\*Rain[t]),0)   
 #- Infil(h.old, P.old,par)\*timeincr  
  
 # Infiltration  
 par$alpha\_i <- ifelse(h\_sub[tt+1]<soilpar$K\_s\*timeincr, 1,  
 (1-(h\_sub[tt+1]-soilpar$K\_s\*timeincr)/h\_sub[tt+1]))  
 # Calculate infiltration and recalculate h\_sub   
 I\_sub[tt] <- Infil(h.old, P.old,par)\*timeincr  
 h\_sub[tt+1] <- h\_sub[tt+1] - I\_sub[tt]   
  
# Now do all plant uptake and growth  
# water uptake by plants: include infiltration in available water  
  
 WU\_sub[tt] <- WU(M=Svir.old,P.old,par)\*timeincr   
   
 # growth rate  
 Gr\_sub[tt] <- Gr(M=Svir.old, P.old,par)\*timeincr   
 # Mortality  
 Mo\_sub[tt]<- Mo(P.old,M=M.old, Svir=Svir.old, par)\*timeincr  
 # calculate plant biomass balance  
 P\_sub[tt + 1] <- P.old + Gr\_sub[tt]- Mo\_sub[tt]   
  
 # 1. Update soil moisture with infiltration  
 M\_sub[tt + 1] <- M.old + I\_sub[tt] - WU\_sub[tt]   
   
 # 2. calculate leaching and capillary rise amount  
 flux\_sub[tt+1]<-do.call(L\_n,list(M=M\_sub[tt+1],Z=Z,soilpar=soilpar,vegpar=vegpar))  
  
# 3. final adjust soil moisture for leaching or capillary rise  
   
 M\_sub[tt + 1] <- M\_sub[tt + 1] + flux\_sub[tt+1]\*timeincr  
  
  
# calculate saltbalance  
  
# Salt leaching  
 L\_salt[tt+1] <- ifelse(flux\_sub[tt+1]<0,  
 par$f\*CM\_sub[tt+1]\*flux\_sub[tt+1]\*timeincr,0)  
 # leaching of salt  
  
# salt upflow  
 U\_salt[tt+1] <- ifelse(flux\_sub[tt+1]>0,   
 par$CM.gw\*flux\_sub[tt+1]\*timeincr,0)   
 # rise of salt  
  
# salt mass coming in with infiltration  
 SmI\_sub[tt+1]<- SmI.old + I\_sub[tt]\*par$ConcConst   
  
#salt mass in soil  
 SmM\_sub[tt+1] <- SmI\_sub[tt+1] + U\_salt[tt+1] - L\_salt[tt+1]  
# Calculate salt concentration in the soil  
 CM\_sub[tt+1]<- (SmM\_sub[tt+1]/M\_sub[tt+1])\*(1/58.44) #   
   
# Virtual saturation (Shah et al., 2012), here in [mm] to be in the same unit as M  
 Svir\_sub[tt + 1]<-soilpar$n\*vegpar$Zr\*((soilpar$h1bar\*10^-1)^(1/soilpar$b))\*  
 ((soilpar$h1bar\*10^-1)\*(M\_sub[tt + 1]/  
 (soilpar$n\*vegpar$Zr))^(-soilpar$b)  
 +(3.6\*CM\_sub[tt + 1]))^(-1/soilpar$b)  
  
 }   
   
# Aggregating the substep results to daily values.  
  
 P[t] = P\_sub[deltat]  
 M[t] = M\_sub[deltat]  
 h[t] = h\_sub[deltat]  
 CM[t] = CM\_sub[deltat]  
 SmM[t] = SmM\_sub[deltat]   
 SmI[t]=SmI\_sub[deltat]  
 In[t]= sum(I\_sub)  
 flux[t] = sum(flux\_sub)  
 Svir[t] = Svir\_sub[deltat]  
  
}  
  
  
# Plotting  
   
if (plotit==T) {   
 plot(M, type="l",ylim=c(-1,10),xlim=c(0,time),xlab=("time [d]"), main=paste(" lambda=", lambda[j],"alpha=", alpha[i]))  
 points(Rain\*10, type="h", col="skyblue")  
   
 lines(h,type="l", col="blue")  
 abline(h=0, col="Gray50",lwd=2,lty=2)  
  
 lines(SmM,type="l", col="red")  
 lines(CM,type="l", col="purple")  
 lines(P/10,type="l", col="green")  
 lines(flux,type="l", col="orange")  
   
   
# legend("topright", title="Variables",cex=1, pt.cex=0.2, c("Moisture [mm]","Rainfall [mm]\*10","overland flow depth[mm] ","salt mass in soil water [g]", "salt concentration in soil water [g/l]", "Plant biomass density [g/m^2]/10"),  
# col=c("black","skyblue","blue","red","purple","green"),lty=1)  
#   
}  
  
Out <- data.frame(P=P,M=M,h=h, CM=CM, SmM=SmM, In=In, flux=flux, Svir=Svir)  
return(Out)  
}

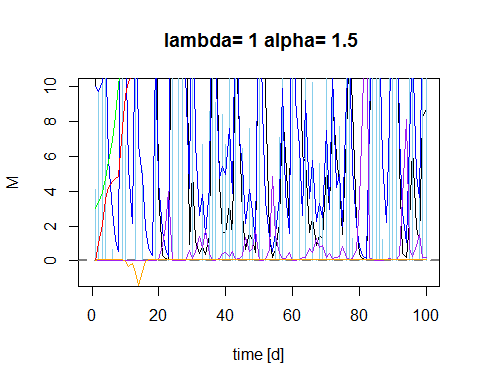
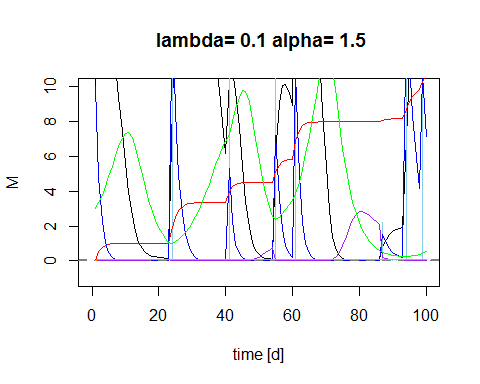
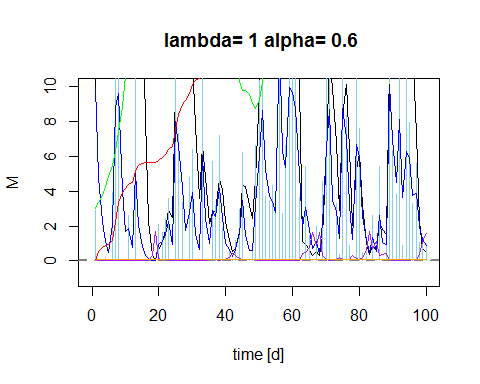
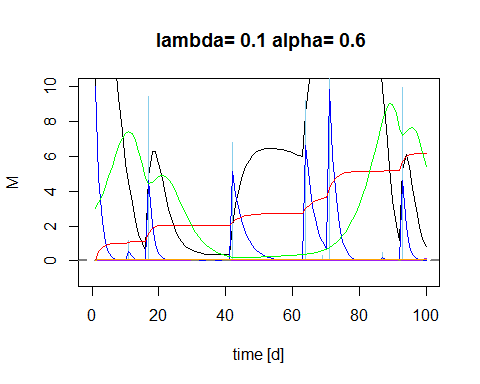
## Changing climates

Alpha and lambda are set to low and high values, respecitively.

alpha <- c(0.6,1.5)   
lambda <- c(0.1,1)  
  
Store <- list()  
sub\_store <- list()  
time <- 100  
delta <- 0  
set.seed(1000)

This loop generates rainfall with the distribution properties alpha[i] and lambda[j]. For each alpha[i] and each lambda[j] the soil water balance function is executed and the results stored in a nested list.

#source("Rainfall.R")  
  
  
for (i in 1:length(alpha)) {  
   
for (j in 1:length(lambda)) {  
 # generate the rainfall  
 Rain <- Precip(time,alpha[i],lambda[j],delta)  
 Rainlist <- data.frame(Precip(time,alpha[i],lambda[j],delta))   
   
 sub\_store[[j]] <-data.frame(alpha\_o=rep(alpha[i],time),  
 lambda\_o=rep(lambda[j],time),  
 balances(Rain,plotit=T, par=par,  
 soilpar, vegpar))  
 }  
 Store[[i]] <- sub\_store  
}



## Results

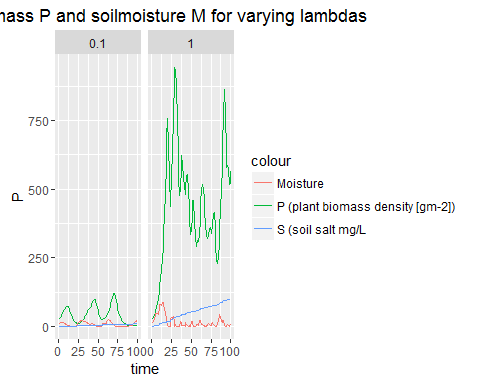
Plotting M and P for different lambdas

require(ggplot2)

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.3.1

lambda\_sum <- do.call(rbind,Store[][[2]])  
lambda\_sum$time <- rep(1:time,length(lambda))  
  
pl <- ggplot(lambda\_sum,aes(x=time,y=P, colour="P (plant biomass density [gm-2])")) + geom\_line()  
pl <- pl + geom\_line(aes(x=time,y=M, colour="Moisture"))   
pl <- pl + facet\_wrap(~lambda\_o)   
pl + ggtitle("Plant biomass P and soilmoisture M for varying lambdas") + geom\_line(aes(x=time, y=SmM, colour= "S (soil salt mg/L")) + theme(plot.title = element\_text(lineheight=.4))



Plotting Soil water salt concentration for different alphas and lambdas

lambda\_sum\_all <- do.call(rbind,do.call(rbind,Store))  
lambda\_sum\_all$time <- rep(rep(1:time,length(lambda)),length(alpha))  
  
  
pa <- ggplot(lambda\_sum\_all,aes(x=time,y=SmM,col=as.factor(lambda\_o))) + geom\_line(linetype=1)   
# pa <- pa + + scale\_color\_gradient(low="blue", high="red")  
pa <- pa + facet\_wrap(~alpha\_o) + ggtitle("Soil water salt mass [g] for varying alphas and lambdas") + theme(plot.title = element\_text(lineheight=.4))  
pa

