Supplementary material A: Ages and transit times as important diagnostics of model performance for predicting carbon dynamics in terrestrial vegetation models

Verónika Ceballos-Núñez

Contents

Motivation

One of the major controls on global C sources and sinks with respect to the atmosphere is the terrestrial vegetation (Canadell et al. 2007). Furthermore, the strength of the source/sink capacity of ecosystems is determined in large part by plant phenology and carbon allocation (CA) strategies (Lacointe 2000; Schiestl-Aalto et al. 2015). In fact, we can study the sink strength capacity of carbon in vegetation by quantifying canopy photosynthesis and C residence times (τ) in plant compartments (Luo et al. 2003).

There is uncertainty in how C allocation should be represented in models. Here, we analyze the influence of different allocation strategies on ecosystem level C cycling using the following metrics: 1) the age distribution of C in the system and in each compartment, 2) the transit time distribution of the system, and 3) the dynamics of radiocarbon for individual compartments.

Objectives and research questions

The main objective of this work was to assess the influence of three different CA schemes on the rates of C cycling in vegetation. These strategies are summarized in figure 1.

The research questions were the following:

- What is the effect of model structure for representing CA on overall cycling rates of carbon in vegetation?
- Is there empirical evidence that provides support for any of the proposed model structures?

This workflow shows how these questions can be answered.

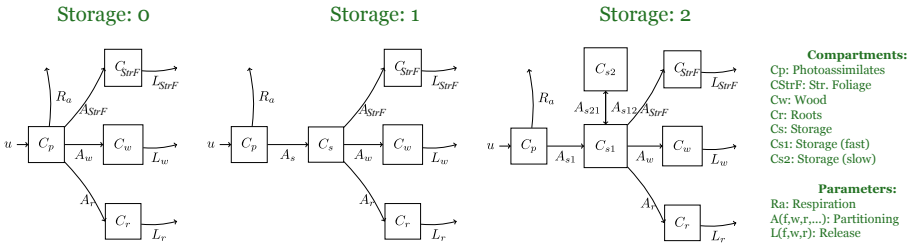


Figure 1: Three carbon allocation strategies in vegetation models. These strategies differ in the number of storage compartments, Storage: 0, Storage: 1, and Storage: 2. Adapted from [**?**]

Methods and results

First load `R` packages.

```
library(SoilR)
library(FME)
library(ggplot2)
library(latex2exp)
library(plyr)
library(Hmisc)
library(RColorBrewer)
library("matrixStats")
set.seed(15)
```

Preparation of the data sets with the environmental drivers and observations

In order to find appropriate parameter values for the implemented models, we use published data sets with field observations and measurements obtained from the following sources:

- Harvard Forest Environmental Measurement Site
- AmeriFlux US-Ha1 Harvard Forest EMS Tower (HFR1)
- US-Ha1: Harvard Forest EMS Tower (HFR1)
- Raw LAI, see also index

This data is processed in the following way:

1) Calculate the amount of carbon in wood, from the aboveground biomass inventory data

```
obs = read.csv("AMF_US-Ha1_BIF_LATEST.csv", stringsAsFactors = FALSE)
obs_w<-subset(obs,VARIABLE_GROUP=="GRP_AG_BIOMASS_TREE")
wood <-obs_w[grep("AG_BIOMASS_DATE|AG_BIOMASS_TREE$", obs_w$VARIABLE),]
value<-subset(wood[grep("AG_BIOMASS_TREE",wood$VARIABLE),])
value<-value[c("DATAVALUE")] # unit: gC m-2
date<-subset(wood[grep("AG_BIOMASS_DATE",wood$VARIABLE),])
date<-date[c("DATAVALUE")]
wood <-obs_w[grep("AG_BIOMASS_DATE|AG_BIOMASS_TREE_SPATIAL_VARIABILITY", obs_w$VARIABLE),]
sd<-subset(wood[grep("AG_BIOMASS_TREE_SPATIAL_VARIABILITY",wood$VARIABLE),])
sd<-sd[c("DATAVALUE")]
d_W <- cbind(date,value,sd)
names(d_W) <- c("time","mean","sd")
d_W<-d_W[order(d_W$time),]
d_W$time <- as.numeric(d_W$time)
d_W$mean <- as.numeric(d_W$mean)
d_W$sd <- as.numeric(d_W$sd)
d_W$max = as.numeric(d_W$mean)+as.numeric(d_W$sd)
d_W$min = as.numeric(d_W$mean)-as.numeric(d_W$sd)
dW <- data.frame("time" = d_W$time,"Cw" = d_W$mean,"sd" = d_W$sd)
```

1) Calculate the amount of carbon in foliage, from LAI and LMA, based on the formula $Cf = lma * lai[g/m^2]$ or $Cf = lai * lma * canopyarea[g]$ (Sitch et al. (2003)). The Leaf Mass per Area (LMA) $[gdrymatter/cm^2]$, which is the inverse of the specific leaf area (SLA), was obtained from Bartlett et al. (2011).

```
obs_l = read.csv("lai_98_15.csv")
LMA = read.csv("LMA.csv", stringsAsFactors = FALSE)
lma <- mean(LMA[,4])
lma <- lma*1e4 # unit: g/m^2
# Correct for issues with values in Bartlett et. al, 2011, and convert dry mass to C:
lma <- lma/(10*2)
obs_l$mean <- obs_l$LAI * lma
year_l = obs_l$year
year_l = unique(year_l)
d_F <- data.frame()
for(i in year_l){
    sub_obs <- subset(obs_l, year == i)
    df_sub <- data.frame(time = i, mean = mean(sub_obs$mean), sd = sd(sub_obs$mean))
    d_F <- rbind(d_F,df_sub)
}
#see also: http://www.srs.fs.usda.gov/pubs/ja/ja_lockhart015.pdf
d_F$max = as.numeric(d_F$mean)+as.numeric(d_F$sd)
d_F$min = as.numeric(d_F$mean)-as.numeric(d_F$sd)
dF <- data.frame("time" = d_F$time,"Cf" = d_F$mean,"sd" = d_F$sd)
```

Implementation of the three model structures

For the initial simulations we use parameter values based on previous work done at the study site (Fox et al. 2009). Since we will implement the three models using the `SoilR` function `GeneralModel_14`, we need to prepare the arguments of the function:

```
# Duration of model run (years)
t_start=1951
t_end=2009
sequence=seq(t_start, t_end)
Age = seq(0,200,by=0.01)

# Initial parameters
input=1400 #unit: gC m^-2 yr^-1
Ra=0.5; Af=0.3; Ar=0.3; Aw=0.3; Lf=0.05*365; Lr=0.008*365; Lw=(1e-05)*365; As=1.8; As12=0.2; As21=0.0076
parsC=c(Ra=0.5, Af=0.3, Ar=0.3, Aw=0.3, Lf=0.05*365, Lr=0.008*365, Lw=(1e-05)*365, As=1.8, As12=0.2, As21=0.0076)
parsB=parsC[parsC != c(As12,As21)]
parsA=parsB[parsB != As]

# Upper and lower boundaries of the parameter values
tl=c(Lf = 1e-04, Lr = 1e-04, Lw = 1e-06) # unit: day-1
tu=c(Lf = 0.1, Lr = 0.01, Lw = 0.01) # unit: day-1
turn_l=(tl*365) # unit: yr-1
turn_u=(tu*365) # unit: yr-1
pA_l=c(Ra = 0.2, Af = 0.01, Ar = 0.01, Aw = 0.01, turn_l)
pA_u=c(Ra = 0.7, Af = 0.5, Ar = 0.5, Aw = 0.5, turn_u)
pB_l=c(pA_l, As = 0.1)
pB_u=c(pA_u, As = 10)
pC_l=c(pB_l, As12 = 0.001, As21 = 0.001)
pC_u=c(pB_u, As12 = 1, As21 = 1)

# Matrices of C cycling parameters for each model
A0 = function(pars){
    A = matrix(
      c((-pars["Ra"]-pars["Af"]-pars["Aw"]-pars["Ar"]), 0,  0,  0,
            pars["Af"],       -pars["Lf"], 0,  0,
            pars["Aw"],        0, -pars["Lw"], 0,
            pars["Ar"],        0,  0, -pars["Lr"]),
      byrow=TRUE,
      nrow=4,
      ncol=4)
    return(A)
}

At = function(pars){
  A=A0(pars)
```

```
  At=new(Class="BoundLinDecompOp",
    t_start,
    t_end,
    function(t0){A}
  )
  return(At)
}

Ab0 = function(pars){
    Ab = matrix(
      c(-pars["Ra"]-pars["As"],       0,        0,      0,      0,
         pars["As"], (-pars["Af"]-pars["Aw"]-pars["Ar"]), 0,      0,      0,
         0,       pars["Af"],   (-pars["Lf"]), 0,      0,
         0,       pars["Aw"],        0, (-pars["Lw"]), 0,
         0,       pars["Ar"],        0,      0, (-pars["Lr"])),
      byrow=TRUE,
      nrow=5,
      ncol=5)
    return(Ab)
}

Atb = function(pars){
  Ab=Ab0(pars)
  Atb=new(Class="BoundLinDecompOp",
    t_start,
    t_end,
    function(t0){Ab}
  )
  return(Atb)
}

Ac0 = function(pars){
    Ac = matrix(
      c(-pars["Ra"]-pars["As"],         0,            0,      0,       0,       0,
         pars["As"], (-pars["As12"]-pars["Af"]-pars["Aw"]-pars["Ar"]), pars["As21"], 0, 0, 0,
              0,         pars["As12"],      (-pars["As21"]), 0,       0,       0,
              0,         pars["Af"],            0, (-pars["Lf"]),  0,       0,
              0,         pars["Aw"],            0,      0, (-pars["Lw"]),   0,
              0,         pars["Ar"],            0,      0,       0,  (-pars["Lr"])),
      byrow=TRUE,
      nrow=6,
      ncol=6)
    return(Ac)
}

Atc = function(pars){
  Ac=Ac0(pars)
  Atc=new(Class="BoundLinDecompOp",
    t_start,
    t_end,
    function(t0){Ac}
  )
  return(Atc)
}

# Input vectors (input * partitioning coefficients)
s = function(pars){
  s = matrix(nrow=4,ncol=1,input*c(1,0,0,0))
  return(s)
}

inputs = function(pars){
  s=s(pars)
  inputs=inputFluxes=new(
    "TimeMap",
    t_start,
```

```
    t_end,
    function(t0){s}
  )
  return(inputs)
}

sb = function(pars){
  sb = matrix(nrow=5,ncol=1,input*c(1, 0, 0, 0, 0))
  return(sb)
}

inputsb = function(pars){
  sb=sb(pars)
  inputsb=inputFluxes=new(
    "TimeMap",
    t_start,
    t_end,
    function(t0){sb}
  )
  return(inputsb)
}

sc = function(pars){
  sc = matrix(nrow=6,ncol=1,input*c(1, 0, 0, 0, 0, 0))
  return(sc)
}

inputsc = function(pars){
  sc=sc(pars)
  inputsc=inputFluxes=new(
    "TimeMap",
    t_start,
    t_end,
    function(t0){sc}
  )
  return(inputsc)
}

# Initial C stocks
c0=c(100, 100, 5000, 1000) # Cp, Cf, Cw, Cr
c0b=c(100, 560, 100, 5000, 1000) # Cp, Cs, Cf, Cw, Cr
c0c=c(100, 280, 280, 100, 5000, 1000) # Cp, Cs1, Cs2, Cf, Cw, Cr

# Set conditions to predict the radiocarbon, based on the bomb-spike
F0=ConstFc(c(0, 0, 0, 0),"Delta14C")
F0b=ConstFc(c(0,0, 0, 0, 0),"Delta14C")
F0c=ConstFc(c(0,0,0, 0, 0, 0),"Delta14C")
AtmFc=BoundFc(Hua2013$NHZone2[,1:2], format="Delta14C")
th=5730
k=log(0.5)/th #note that k is negative and has the unit y^-1
```

Then, the models can run via the following functions:

```
modelA = function(pars){
     model=GeneralModel_14(t=sequence, A=At(pars), ivList=c0, initialValF=F0,
     inputFluxes=inputs(pars), inputFc=AtmFc, di=k, pass = TRUE)
     return(model)
}

modelB = function(pars){
     model=GeneralModel_14(t=sequence, A=Atb(pars), ivList=c0b, initialValF=F0b,
     inputFluxes=inputsb(pars), inputFc=AtmFc, di=k, pass = TRUE)
     return(model)
}

modelC = function(pars){
```

```
    model=GeneralModel_14(t=sequence, A=Atc(pars), ivList=c0c, initialValF=F0c,
    inputFluxes=inputsc(pars), inputFc=AtmFc, di=k, pass = TRUE)
    return(model)
}
```

Parameter estimation

In order to further constrain the models, we calculated the amount of C in Roots and Storage based on shoot:root ratio = 1:5 and NSC from Wood (see pg 5 Richardson et al. (2013))

```
dR <- data.frame(time=dF[,1],Cr=dF[,2]*5,sd=dF[,3]*5)
dS <- data.frame(time=dW[,1],Cs=dW[,2]*(22.4*4*0.5/0.4)/1000,sd=dW[,3]*(22.4*4*0.5/0.4)/1000)
```

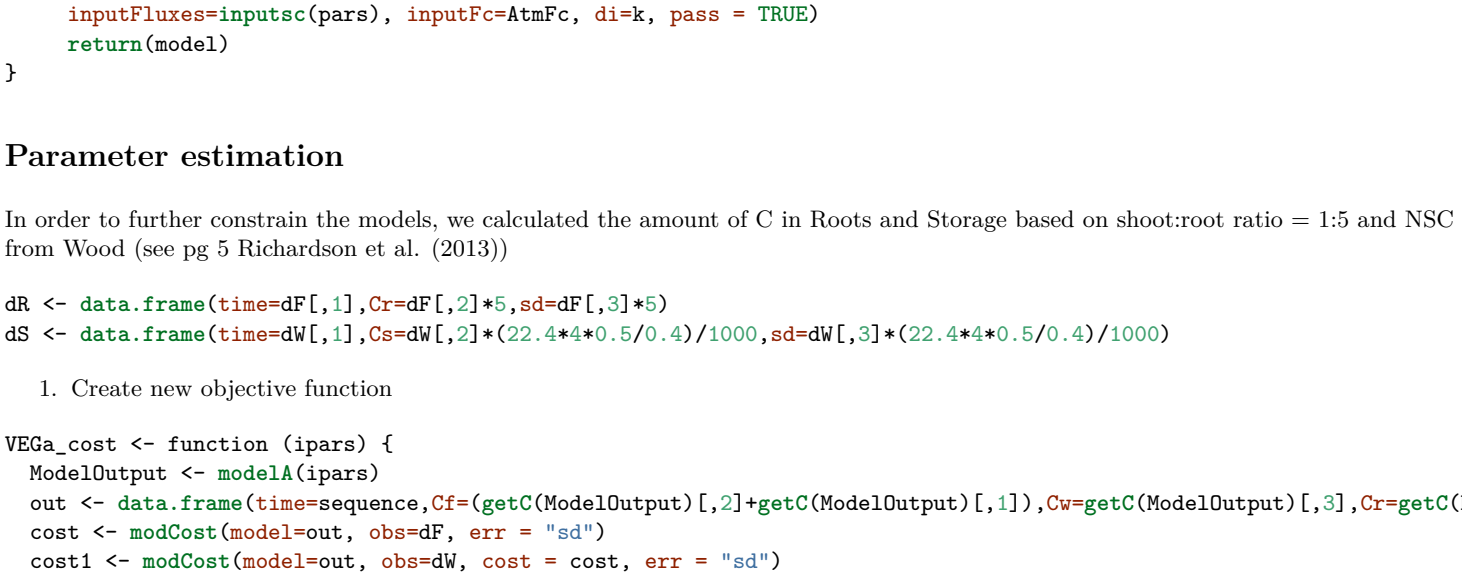
1. Create new objective function

```
VEGa_cost <- function (ipars) {
  ModelOutput <- modelA(ipars)
  out <- data.frame(time=sequence,Cf=(getC(ModelOutput)[,2]+getC(ModelOutput)[,1]),Cw=getC(ModelOutput)[,3],Cr=getC(ModelOu
  cost <- modCost(model=out, obs=dF, err = "sd")
  cost1 <- modCost(model=out, obs=dW, cost = cost, err = "sd")
  return(modCost(model=out, obs=dR, cost = cost1, err = "sd"))
}

VEGb_cost <- function (ipars) {
  ModelOutput <- modelB(ipars)
  out <- data.frame(time=sequence,Cs=getC(ModelOutput)[,2],Cf=(getC(ModelOutput)[,3]+getC(ModelOutput)[,1]),Cw=getC(ModelOu
  cost <- modCost(model=out, obs=dF, err = "sd")
  cost1 <- modCost(model=out, obs=dW, cost=cost, err = "sd")
  cost2 <- modCost(model=out, obs=dR, cost = cost1, err = "sd")
  return(modCost(model=out, obs=dS, cost = cost2, err = "sd"))
}

VEGc_cost <- function (ipars) {
  ModelOutput <- modelC(ipars)
  out <- data.frame(time=sequence,Cs=(getC(ModelOutput)[,2]+getC(ModelOutput)[,3]),Cf=(getC(ModelOutput)[,4]+getC(ModelOutp
  cost <- modCost(model=out, obs=dF, err = "sd")
  cost1 <- modCost(model=out, obs=dW, cost=cost, err = "sd")
  cost2 <- modCost(model=out, obs=dR, cost = cost1, err = "sd")
  return(modCost(model=out, obs=dS, cost = cost2, err = "sd"))
}
```

2. Local Sensitivity Analysis (`sensFun`) and Collinearity of Parameter Sets (`collin`)

- For the model without storage compartment:

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
sensA = sensFun(f=VEGa_cost,parms = parsA)
CollinA = collin(sensA, parset = NULL, N = NULL, maxcomb = 5000)
CollinA = subset(CollinA, !collinearity == Inf)
plot(CollinA, log="y")
abline(h = 20, col = "red")
```



```
# select "identifiable" sets with 4 parameters
CollinA [CollinA[,"collinearity"] < 20 & CollinA[,"N"]==3,]
```

```
##    Ra Af Ar Aw Lf Lr Lw N collinearity
## 1   1  0  1  1  0  0  0 3          1.6
## 2   1  0  1  0  1  0  0 3          1.9
## 3   1  0  1  0  0  0  1 3          2.0
## 4   1  0  0  1  1  0  0 3          2.0
## 5   1  0  0  1  0  1  0 3          2.0
## 6   1  0  0  0  1  1  0 3          2.3
## 7   1  0  0  0  1  0  1 3          2.4
## 8   1  0  0  0  0  1  1 3          2.4
## 9   0  1  1  1  0  0  0 3          1.7
## 10  0  1  1  0  1  0  0 3          1.9
```

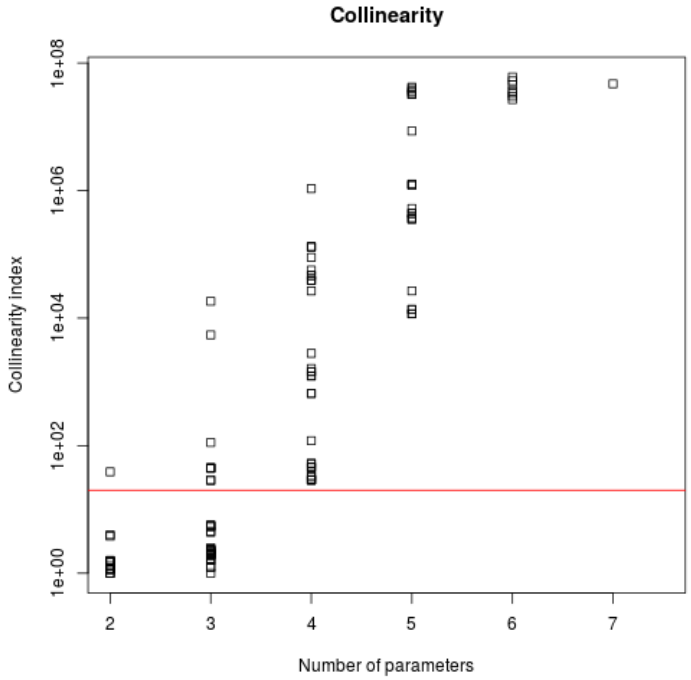


Figure 2: plot of chunk Collinearity_S0

```
## 11  0  1  1  0  0  0  1 3          2.1
## 12  0  1  0  1  1  0  0 3          2.0
## 13  0  1  0  1  0  1  0 3          2.1
## 14  0  1  0  0  1  1  0 3          2.2
## 15  0  1  0  0  1  0  1 3          2.3
## 16  0  1  0  0  0  1  1 3          2.5
## 17  0  0  1  1  1  0  0 3          1.6
## 18  0  0  1  1  0  1  0 3          4.4
## 19  0  0  1  1  0  0  1 3          4.5
## 20  0  0  1  0  1  1  0 3          5.3
## 21  0  0  1  0  1  0  1 3          1.3
## 22  0  0  1  0  0  1  1 3          5.5
## 23  0  0  0  1  1  1  0 3          1.2
## 24  0  0  0  1  1  0  1 3          5.7
## 25  0  0  0  1  0  1  1 3          5.7
## 26  0  0  0  0  1  1  1 3          1.0
```

- For the model with one storage compartment:

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
sensB = sensFun(f=VEGb_cost,parms = parsB)
CollinB = collin(sensB, parset = NULL, N = NULL, maxcomb = 5000)
CollinB = subset(CollinB, !collinearity == Inf)
plot(CollinB, log="y")
abline(h = 20, col = "red")

# select "identifiable" sets with 4 parameters
CollinB [CollinB[,"collinearity"] < 20 & CollinB[,"N"]==4,]
```

```
##    Ra Af Ar Aw Lf Lr Lw As N collinearity
## 1   1  1  1  1  0  0  0  0 4          2.9
## 2   1  1  1  0  0  0  1  0 4          2.9
## 3   1  1  0  1  0  1  0  0 4          2.9
## 4   1  1  0  0  0  1  1  0 4          3.0
## 5   1  0  1  1  1  0  0  0 4          1.5
## 6   1  0  1  1  0  1  0  0 4          5.1
## 7   1  0  1  1  0  0  1  0 4          5.3
## 8   1  0  1  1  0  0  0  1 4          1.3
```

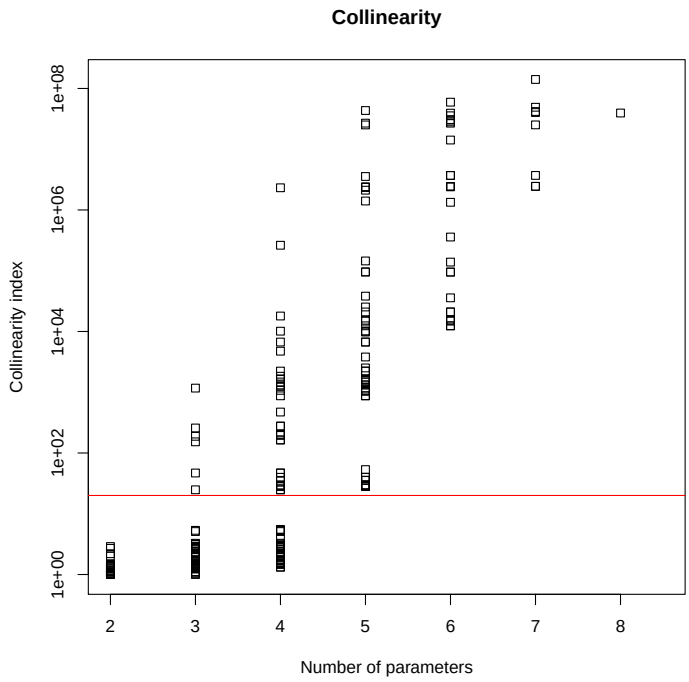


Figure 3: plot of chunk Collinearity_S1

```
## 9   1  0  1  0  1  0  1  0 4          1.7
## 10  1  0  1  0  0  1  1  0 4          5.3
## 11  1  0  1  0  0  0  1  1 4          1.6
## 12  1  0  0  1  1  1  0  0 4          1.7
## 13  1  0  0  1  0  1  1  0 4          5.5
## 14  1  0  0  1  0  1  0  1 4          1.5
## 15  1  0  0  0  1  1  1  0 4          2.2
## 16  1  0  0  0  0  1  1  1 4          1.9
## 17  0  1  1  1  1  0  0  0 4          1.3
## 18  0  1  1  1  0  0  0  1 4          1.5
## 19  0  1  1  0  1  0  1  0 4          1.6
## 20  0  1  1  0  0  0  1  1 4          1.7
## 21  0  1  0  1  1  1  0  0 4          1.5
## 22  0  1  0  1  0  1  0  1 4          1.7
## 23  0  1  0  0  1  1  1  0 4          1.9
## 24  0  1  0  0  0  1  1  1 4          2.0
## 25  0  0  1  1  1  1  0  0 4          2.0
## 26  0  0  1  1  1  0  1  0 4          2.2
## 27  0  0  1  1  1  0  0  1 4          2.8
## 28  0  0  1  1  0  1  0  1 4          2.4
## 29  0  0  1  1  0  0  1  1 4          2.6
## 30  0  0  1  0  1  1  1  0 4          2.6
## 31  0  0  1  0  1  0  1  1 4          3.3
## 32  0  0  1  0  0  1  1  1 4          2.9
## 33  0  0  0  1  1  1  1  0 4          2.8
## 34  0  0  0  1  1  1  0  1 4          3.2
## 35  0  0  0  1  0  1  1  1 4          3.0
## 36  0  0  0  0  1  1  1  1 4          4.1
```

- And for the model with two storage compartments:

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
sensC = sensFun(f=VEGc_cost,parms = parsC)
CollinC = collin(sensC, parset = NULL, N = NULL, maxcomb = 5000)
CollinC = subset(CollinC, !collinearity == Inf)
plot(CollinC, log="y")
abline(h = 20, col = "red")
```

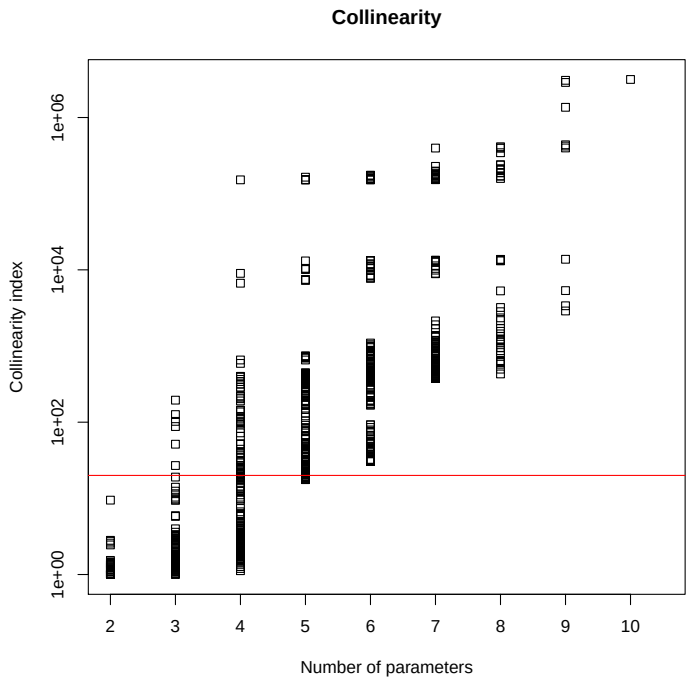


Figure 4: plot of chunk Collinearity_S2

```
# select "identifiable" sets with 4 parameters
CollinC [CollinC[,"collinearity"] < 20 & CollinC[,"N"]==5,]
```

```
##    Ra Af Ar Aw Lf Lr Lw As As12 As21 N collinearity
## 1   1  1  1  1  0  0  0  0    0    1 5           19
## 2   1  1  1  0  0  0  0  0    1    1 5           19
## 3   1  1  0  0  0  1  0  0    1    1 5           19
## 4   1  0  1  1  1  0  0  0    0    1 5           20
## 5   1  0  1  1  0  0  0  1    0    1 5           19
## 6   1  0  1  1  0  0  0  0    1    1 5           18
## 7   1  0  1  0  1  0  0  0    1    1 5           19
## 8   1  0  1  0  0  1  0  0    1    1 5           19
## 9   1  0  1  0  0  0  1  0    1    1 5           19
## 10  1  0  1  0  0  0  0  1    1    1 5           19
## 11  1  0  0  1  0  1  0  0    1    1 5           19
## 12  1  0  0  0  1  1  0  0    1    1 5           19
## 13  1  0  0  0  0  1  1  0    1    1 5           19
## 14  1  0  0  0  0  1  0  1    1    1 5           19
## 15  0  1  1  1  1  0  0  0    0    1 5           19
## 16  0  1  1  1  0  0  0  1    0    1 5           19
## 17  0  1  1  0  1  0  0  0    1    1 5           19
## 18  0  1  1  0  0  0  0  1    1    1 5           19
## 19  0  1  0  0  1  1  0  0    1    1 5           19
## 20  0  1  0  0  0  1  0  1    1    1 5           19
## 21  0  0  1  1  1  0  0  0    1    1 5           18
## 22  0  0  1  1  0  0  0  1    1    1 5           18
## 23  0  0  1  0  1  1  0  0    1    1 5           19
## 24  0  0  1  0  1  0  1  0    1    1 5           18
## 25  0  0  1  0  1  0  0  1    1    1 5           19
## 26  0  0  1  0  0  1  0  1    1    1 5           19
## 27  0  0  1  0  0  0  1  1    1    1 5           18
## 28  0  0  0  1  1  1  0  0    1    1 5           19
## 29  0  0  0  1  0  1  0  1    1    1 5           19
## 30  0  0  0  0  1  1  1  0    1    1 5           19
## 31  0  0  0  0  1  1  0  1    1    1 5           19
## 32  0  0  0  0  0  1  1  1    1    1 5           19
```

4. Classical ('modFit') and Bayesian ('modMCMC') optimizations

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)

niter = 1.1*1e3
bl = 1e2 #burninlength
bestA=modFit(f=VEGa_cost,p=parsA,method="Port",lower=pA_l,upper=pA_u)
saveRDS(bestA,file = "bestA.Rda")
#bestA <- readRDS(file = "bestA.Rda")
fmA=modMCMC(f=VEGa_cost,p=bestA$par,niter=niter,burninlength=bl,lower=pA_l,upper=pA_u)
```

```
## number of accepted runs: 246 out of 1100 (22.36364%)
```

```
saveRDS(fmA,file = "fmA.Rda")
#fmA <- readRDS(file = "fmA.Rda")

pairs(fmA, pch=".")
```

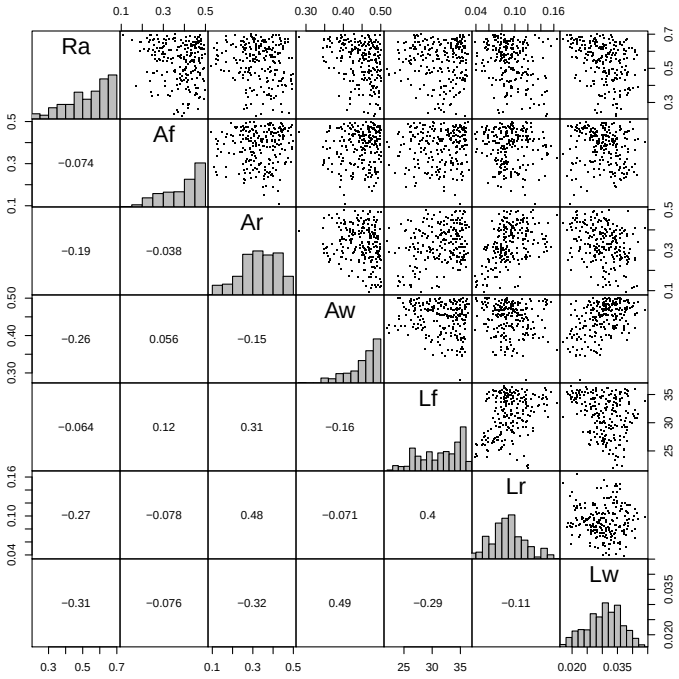


Figure 5: plot of chunk Optimization_S0

```
summary(fmA)
```

```
##              Ra          Af          Ar          Aw        Lf          Lr
## mean 0.5377394 0.38969848 0.32990074 0.45359408 31.495592 0.08987119
## sd   0.1204403 0.08553033 0.09207687 0.04074248  3.612683 0.02593515
## min  0.2224292 0.10976952 0.09489612 0.28201689 22.036736 0.03892560
## max  0.6996265 0.49670183 0.49703785 0.49989810 36.477193 0.16441160
## q025 0.4545934 0.31626382 0.26422494 0.42580837 28.471380 0.07396357
## q050 0.5659175 0.41766283 0.33144985 0.46735614 32.118828 0.08854266
## q075 0.6406950 0.46347849 0.40094597 0.48572462 34.917494 0.10357295
##                Lw
## mean 0.030546950
## sd   0.005697464
## min  0.017069934
## max  0.043998385
## q025 0.026805889
## q050 0.030784974
## q075 0.035706787
```

```
plot(fmA, pch=".", Full=FALSE)
```

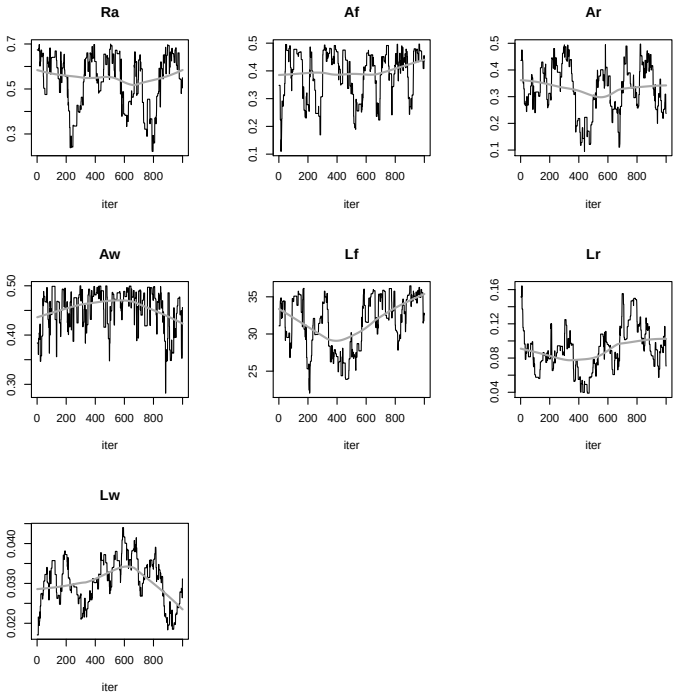


Figure 6: plot of chunk Optimization_S0

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)

bestB=modFit(f=VEGb_cost,p=parsB,method="Port",lower=pB_l,upper=pB_u)
saveRDS(bestB,file = "bestB.Rda")
#bestB <- readRDS(file = "bestB.Rda")
fmB=modMCMC(f=VEGb_cost,p=bestB$par,niter=niter,burninlength=bl,lower=pB_l,upper=pB_u)

## number of accepted runs: 562 out of 1100 (51.09091%)

saveRDS(fmB,file = "fmB.Rda")
#fmB <- readRDS(file = "fmB.Rda")

pairs(fmB, pch=".")

summary(fmB)

##              Ra          Af          Ar          Aw          Lf          Lr
## mean 0.4852945 0.20444087 0.36275555 0.37824973 10.2838051 0.17183288
## sd   0.1509273 0.10271544 0.07972733 0.07188536  4.8226830 0.04375194
## min  0.2066780 0.01343316 0.16807487 0.22690444  0.7353323 0.08271257
## max  0.6993101 0.45056846 0.49797684 0.49917740 20.7308706 0.33201781
## q025 0.3504351 0.13188283 0.30226199 0.31953798  6.4876970 0.14202645
## q050 0.5163725 0.18656159 0.36399198 0.36743049 10.8242871 0.16703322
## q075 0.6129705 0.30429350 0.43211896 0.44358383 14.1806690 0.19496801
##               Lw        As
## mean 0.04168362 3.3845617
## sd   0.01118481 1.2815884
## min  0.01416703 0.9056213
## max  0.06994099 6.5317302
## q025 0.03317680 2.2594473
## q050 0.04086602 3.5485872
## q075 0.04975787 4.2871813

plot(fmB, pch=".", Full=FALSE)

knitr::opts_chunk$set(dev = 'pdf', dpi = 300)

bestC=modFit(f=VEGc_cost,p=parsC,method="Port",lower=pC_l,upper=pC_u)
```

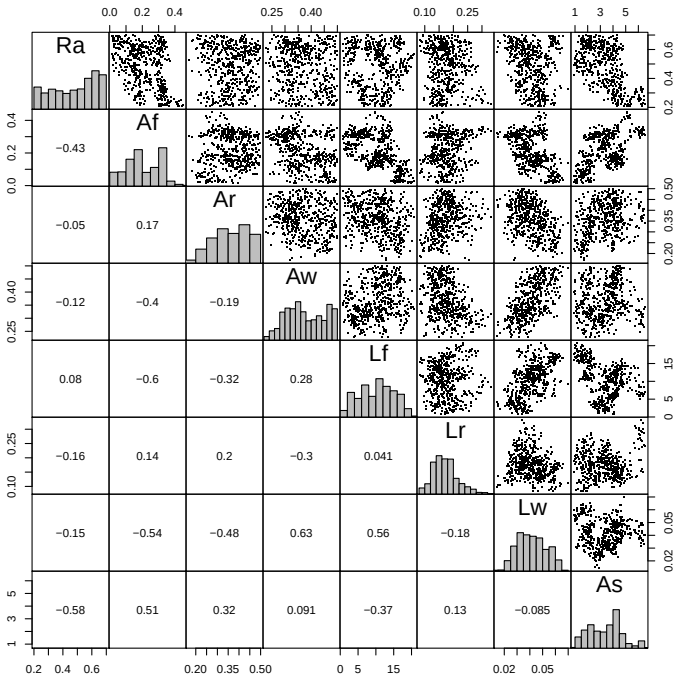


Figure 7: plot of chunk Optimization_S1

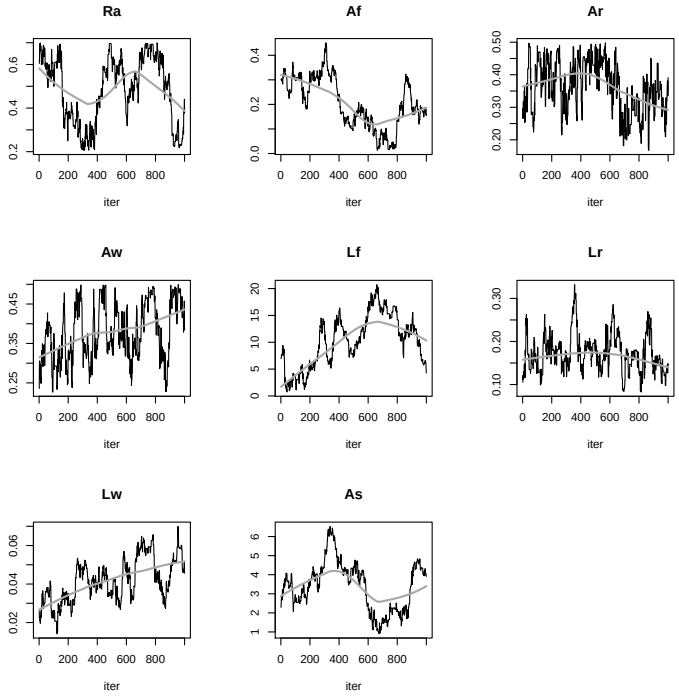


Figure 8: plot of chunk Optimization_S1

```
saveRDS(bestC,file = "bestC.Rda")
#bestC <- readRDS(file = "bestC.Rda")
fmC=modMCMC(f=VEGc_cost,p=bestC$par,niter=niter,burninlength=bl,lower=pC_l,upper=pC_u)
```

```
## number of accepted runs: 424 out of 1100 (38.54545%)
```

```
saveRDS(fmC,file = "fmC.Rda")
#fmC <- readRDS(file = "fmC.Rda")
```

```
pairs(fmC, pch=".")
```

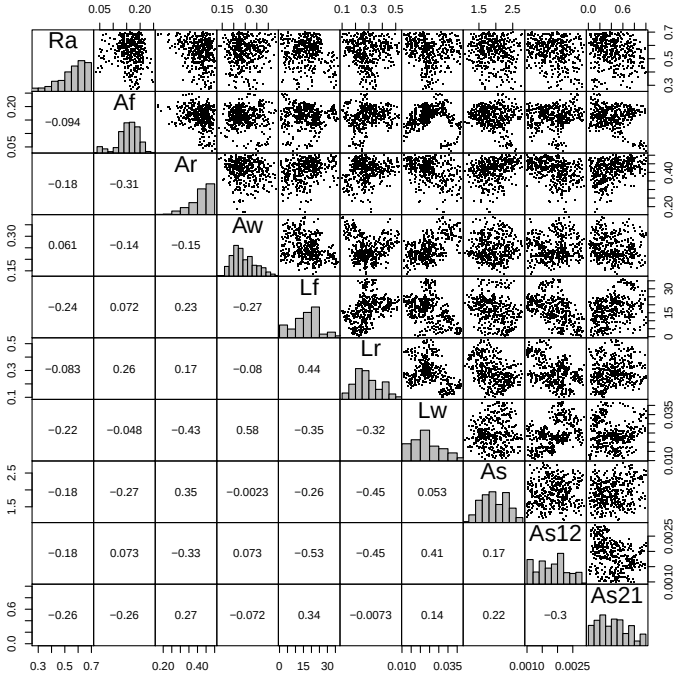
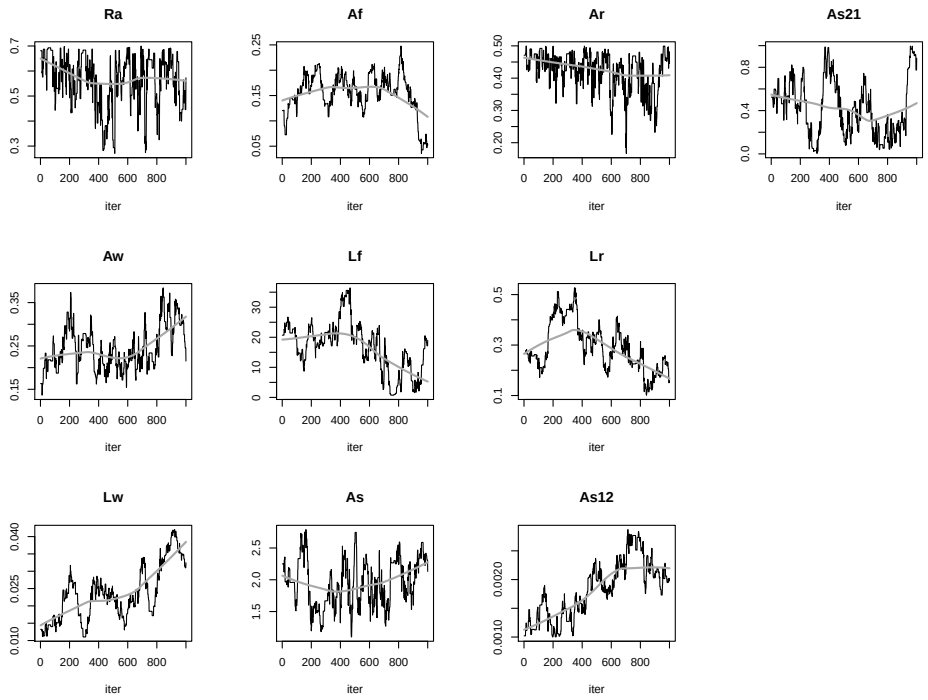


Figure 9: plot of chunk Optimization_S2

```
summary(fmC)
```

```
##              Ra          Af          Ar          Aw          Lf          Lr
## mean 0.5627097 0.15551934 0.42111968 0.24452663 16.7965711 0.2911243
## sd   0.1024067 0.03920269 0.05938581 0.04956248  8.2547012 0.0950537
## min  0.2706697 0.03574996 0.16678426 0.13643688  0.7371519 0.1013241
## max  0.6996823 0.24766430 0.49937912 0.38403629 36.4438919 0.5267251
## q025 0.5030016 0.13776851 0.38940898 0.20848175 12.0336026 0.2253323
## q050 0.5790685 0.16075213 0.43574102 0.23387609 17.8877389 0.2696329
## q075 0.6451730 0.18294743 0.46664409 0.27787867 21.5997087 0.3508147
##                Lw        As          As12        As21
## mean 0.023605872 1.9555911 0.0018224667 0.425682265
## sd   0.007823027 0.3707416 0.0004993102 0.257974859
## min  0.011031906 1.1004875 0.0010013787 0.004439109
## max  0.042016901 2.7889509 0.0028693707 0.996000251
## q025 0.017171291 1.6684939 0.0014331269 0.216327207
## q050 0.023173902 1.9576630 0.0018331637 0.414335918
## q075 0.028310269 2.2633133 0.0021590042 0.598712991
```

```
plot(fmC, pch=".", Full=FALSE)
```



Parameter sets

```
# Select the parameter set with the highest frequency
freqA <- plyr::count(fmA$pars)
mfpA <- freqA[freqA$freq == max(freqA$freq),] # Parameter set with highest frequency

freqB <- plyr::count(fmB$pars)
mfpB <- freqB[freqB$freq == max(freqB$freq),]

freqC <- plyr::count(fmC$pars)
mfpC <- freqC[freqC$freq == max(freqC$freq),]

finpara <- as.double(mfpA[,1:7])
names(finpara) = c("Ra","Af","Ar","Aw","Lf","Lr","Lw")

finparb <- as.double(mfpB[,1:8])
names(finparb) = c("Ra","Af","Ar","Aw","Lf","Lr","Lw","As")

finparc <- as.double(mfpC[,1:10])
names(finparc) = c("Ra","Af","Ar","Aw","Lf","Lr","Lw","As","As12","As21")

# Parameter values of each model
parsA <- data.frame("Final"=finpara,"Best1"=bestA$par, "Best2"=fmA$bestpar, "Median"=unlist(summary(fmA)["q050",]),
"q_25"=unlist(summary(fmA)["q025",]), "q_75"=unlist(summary(fmA)["q075",]))

parsB <- data.frame("Final"=finparb,"Best1"=bestB$par, "Best2"=fmB$bestpar, "Median"=unlist(summary(fmB)["q050",]),
"q_25"=unlist(summary(fmB)["q025",]), "q_75"=unlist(summary(fmB)["q075",]))

parsC <- data.frame("Final"=finparc,"Best1"=bestC$par, "Best2"=fmC$bestpar, "Median"=unlist(summary(fmC)["q050",]),
"q_25"=unlist(summary(fmC)["q025",]), "q_75"=unlist(summary(fmC)["q075",]))

# Parameter values of all the models:
parsT <- rbind(parsA,parsB,parsC)
is.num <- sapply(parsT, is.numeric)
parsT[is.num] <- lapply(parsT[is.num], round, 2)
parsT
```

```
##      Final Best1 Best2 Median  q_25  q_75
## Ra    0.64  0.70  0.70   0.57  0.45  0.64
```

```
## Af    0.48  0.50  0.50   0.42  0.32  0.46
## Ar    0.32  0.50  0.50   0.33  0.26  0.40
## Aw    0.50  0.49  0.49   0.47  0.43  0.49
## Lf   35.09 17.84 17.84  32.12 28.47 34.92
## Lr    0.06  0.12  0.12   0.09  0.07  0.10
## Lw    0.04  0.02  0.02   0.03  0.03  0.04
## Ra1   0.61  0.32  0.32   0.52  0.35  0.61
## Af1   0.31  0.17  0.17   0.19  0.13  0.30
## Ar1   0.47  0.50  0.50   0.36  0.30  0.43
## Aw1   0.29  0.30  0.30   0.37  0.32  0.44
## Lf1   3.04  7.90  7.90  10.82  6.49 14.18
## Lr1   0.13  0.21  0.21   0.17  0.14  0.19
## Lw1   0.02  0.03  0.03   0.04  0.03  0.05
## As    2.55  2.14  2.14   3.55  2.26  4.29
## Ra2   0.65  0.70  0.70   0.58  0.50  0.65
## Af2   0.15  0.10  0.10   0.16  0.14  0.18
## Ar2   0.47  0.50  0.50   0.44  0.39  0.47
## Aw2   0.23  0.19  0.19   0.23  0.21  0.28
## Lf2   0.74 17.52 17.52  17.89 12.03 21.60
## Lr2   0.23  0.21  0.21   0.27  0.23  0.35
## Lw2   0.02  0.01  0.01   0.02  0.02  0.03
## As1   2.27  1.69  1.69   1.96  1.67  2.26
## As12  0.00  0.00  0.00   0.00  0.00  0.00
## As21  0.09  0.82  0.82   0.41  0.22  0.60
```

Run models with estimated parameter sets and prepare data frames for plotting C stocks and releases

```
fVEGa = modelA(finpara)
fCta=getC(fVEGa)
fRta=getReleaseFlux(fVEGa)
fC14ta=getF14(fVEGa)

Rp <- data.frame("time"=sequence,"stock"=fCta[,1],"rel"=fRta[,1])
Rp$col <- "Photoassimilates"
Rf <- data.frame("time"=sequence,"stock"=fCta[,2],"rel"=fRta[,2])
Rf$col <- "Str. Foliage"
Rw <- data.frame("time"=sequence,"stock"=fCta[,3],"rel"=fRta[,3])
Rw$col <- "Wood"
Rr <- data.frame("time"=sequence,"stock"=fCta[,4],"rel"=fRta[,4])
Rr$col <- "Roots"
df_a <- do.call(rbind, list(Rp,Rf,Rw,Rr))

fVEGb = modelB(finparb)
fCtb=getC(fVEGb)
fRtb=getReleaseFlux(fVEGb)
fC14tb=getF14(fVEGb)

Rp <- data.frame("time"=sequence,"stock"=fCtb[,1],"rel"=fRtb[,1])
Rp$col <- "Photoassimilates"
Rs <- data.frame("time"=sequence,"stock"=fCtb[,2],"rel"=fRtb[,2])
Rs$col <- "Storage"
Rf <- data.frame("time"=sequence,"stock"=fCtb[,3],"rel"=fRtb[,3])
Rf$col <- "Str. Foliage"
Rw <- data.frame("time"=sequence,"stock"=fCtb[,4],"rel"=fRtb[,4])
Rw$col <- "Wood"
Rr <- data.frame("time"=sequence,"stock"=fCtb[,5],"rel"=fRtb[,5])
Rr$col <- "Roots"
df_b <- do.call(rbind, list(Rp,Rs,Rf,Rw,Rr))

fVEGc = modelC(finparc)
fCtc=getC(fVEGc)
fRtc=getReleaseFlux(fVEGc)
fC14tc=getF14(fVEGc)
```

```
Rp <- data.frame("time"=sequence,"stock"=fCtc[,1],"rel"=fRtc[,1])
Rp$col <- "Photoassimilates"
Rs1 <- data.frame("time"=sequence,"stock"=fCtc[,2],"rel"=fRtc[,2])
Rs1$col <- "Storage (fast)"
Rs2 <- data.frame("time"=sequence,"stock"=fCtc[,3],"rel"=fRtc[,3])
Rs2$col <- "Storage (slow)"
Rf <- data.frame("time"=sequence,"stock"=fCtc[,4],"rel"=fRtc[,4])
Rf$col <- "Str. Foliage"
Rw <- data.frame("time"=sequence,"stock"=fCtc[,5],"rel"=fRtc[,5])
Rw$col <- "Wood"
Rr <- data.frame("time"=sequence,"stock"=fCtc[,6],"rel"=fRtc[,6])
Rr$col <- "Roots"
df_c <- do.call(rbind, list(Rp,Rs1,Rs2,Rf,Rw,Rr))
```

Plots of the results for C stocks, comparing the observed data and the fitted models

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot(df_a, aes(x=time, y=stock))+
  geom_point(data = dR, aes(time, Cr), colour="red") +
  scale_shape_manual(values = 15) +
  geom_errorbar(
      data = dR,
      aes(time, Cr, ymin = Cr-sd, ymax = Cr+sd),
      width = 0.05,alpha=0.5) +
  geom_point(data = d_W, aes(time, mean), colour="brown") +
  geom_errorbar(
    data = d_W,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_point(data = d_F, aes(time, mean), colour="green") +
  geom_errorbar(
    data = d_F,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_line(aes(colour=col)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,0.5), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  xlim(1995, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("green","red","orange","brown")) +
  guides(color=guide_legend(
    keywidth=0.3,
    keyheight=0.3,
    default.unit="inch"))

knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot(df_b, aes(x=time, y=stock))+
  geom_point(data = dS, aes(time, Cs), colour="Purple") +
  geom_errorbar(
      data = dS,
      aes(time, Cs, ymin = Cs-sd, ymax = Cs-sd),
      width = 0.05,alpha=0.5) +
  geom_point(data = dR, aes(time, Cr), colour="red") +
  scale_shape_manual(values = 15) +
  geom_errorbar(
      data = dR,
      aes(time, Cr, ymin = Cr-sd, ymax = Cr+sd),
      width = 0.05,alpha=0.5) +
  geom_point(data = d_W, aes(time, mean), colour="brown") +
```

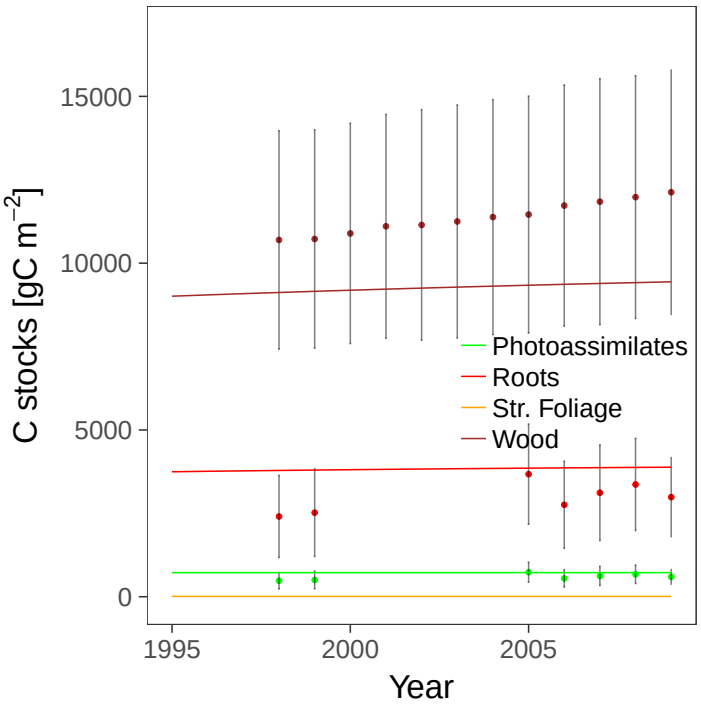


Figure 10: plot of chunk C_stocks_S0

```
  geom_errorbar(
    data = d_W,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_point(data = d_F, aes(time, mean), colour="green") +
  geom_errorbar(
    data = d_F,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_line(aes(colour=col)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,0.5), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  xlim(1995, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("green","red","purple","orange","brown")) +
  guides(color=guide_legend(
    keywidth=0.3,
    keyheight=0.3,
    default.unit="inch"))

knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot(df_c, aes(x=time, y=stock))+
  geom_point(data = dS, aes(time, Cs), colour="Purple") +
  geom_errorbar(
      data = dS,
      aes(time, Cs, ymin = Cs-sd, ymax = Cs-sd),
      width = 0.05,alpha=0.5) +
  geom_point(data = dR, aes(time, Cr), colour="red") +
  scale_shape_manual(values = 15) +
  geom_errorbar(
      data = dR,
      aes(time, Cr, ymin = Cr-sd, ymax = Cr+sd),
```

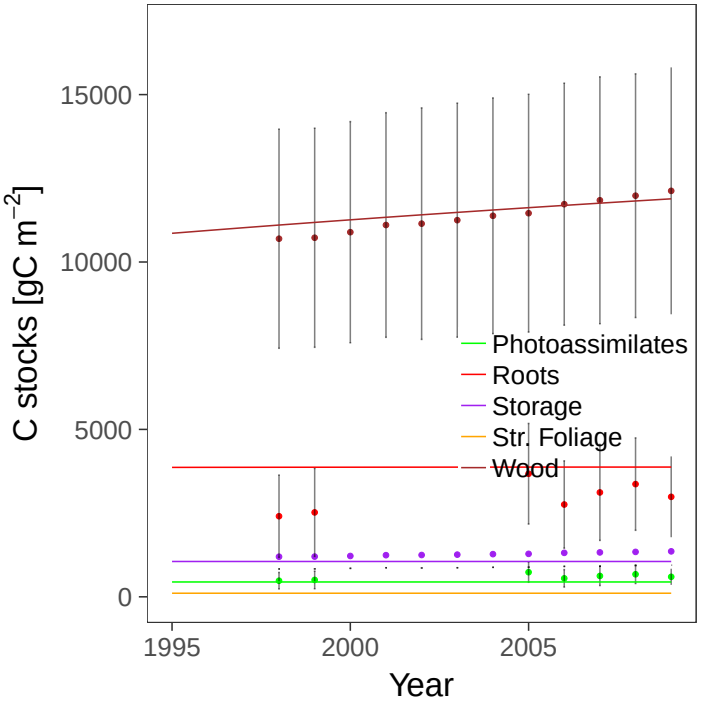


Figure 11: plot of chunk C_stocks_S1

```
    width = 0.05,alpha=0.5) +
  geom_point(data = d_W, aes(time, mean), colour="brown") +
  geom_errorbar(
    data = d_W,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_point(data = d_F, aes(time, mean), colour="green") +
  geom_errorbar(
    data = d_F,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_line(aes(colour=col)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,0.5), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  xlim(1995, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("green","red","purple","blue","orange","brown")) +
  guides(color=guide_legend(
    keywidth=0.3,
    keyheight=0.3,
    default.unit="inch"))
```

Comparison of the C stocks among all models

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
S0w <- data.frame("time"=sequence,"stock"=fCta[,3])
S0w$col <- "Storage: 0"
S1w <- data.frame("time"=sequence,"stock"=fCtb[,4])
S1w$col <- "Storage: 1"
S2w <- data.frame("time"=sequence,"stock"=fCtc[,5])
S2w$col <- "Storage: 2"
```

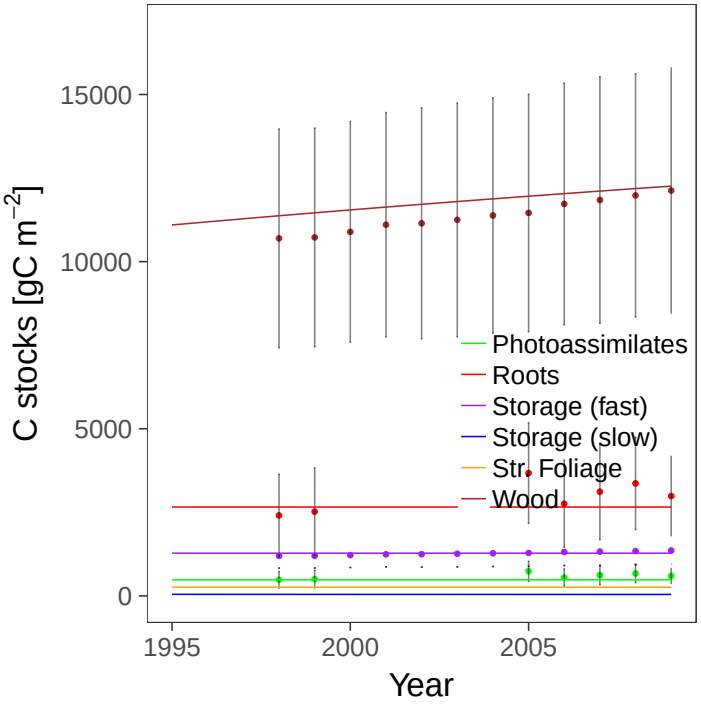


Figure 12: plot of chunk C_stocks_S2

```
DF_w <- do.call(rbind, list(S0w,S1w,S2w))

ggplot(DF_w, aes(x=time, y=stock))+
  geom_point(data = d_W, aes(time, mean)) +
  xlim(1951, 2009) +
  geom_errorbar(
    data = d_W,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_line(aes(colour=col)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.5,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.text = element_text(size = 22),
        legend.title = element_blank()) +
  xlab("Year") +
  ylab(bquote('C stock in Wood ['*gC~m^-2*']')) +
  guides(color=guide_legend(
    keywidth=0.3,
    keyheight=0.3,
    default.unit="inch"))

S0f <- data.frame("time"=sequence,"stock"=fCta[,2]+fCta[,1])
S0f$col <- "Storage: 0"
S1f <- data.frame("time"=sequence,"stock"=fCtb[,3]+fCtb[,1])
S1f$col <- "Storage: 1"
S2f <- data.frame("time"=sequence,"stock"=fCtc[,4]+fCtc[,1])
S2f$col <- "Storage: 2"
DF_f <- do.call(rbind, list(S0f,S1f,S2f))

ggplot(DF_f, aes(x=time, y=stock))+
  xlim(1951, 2009) +
  geom_point(data = d_F, aes(time, mean)) +
  scale_shape_manual(values = 15) +
  geom_errorbar(
```

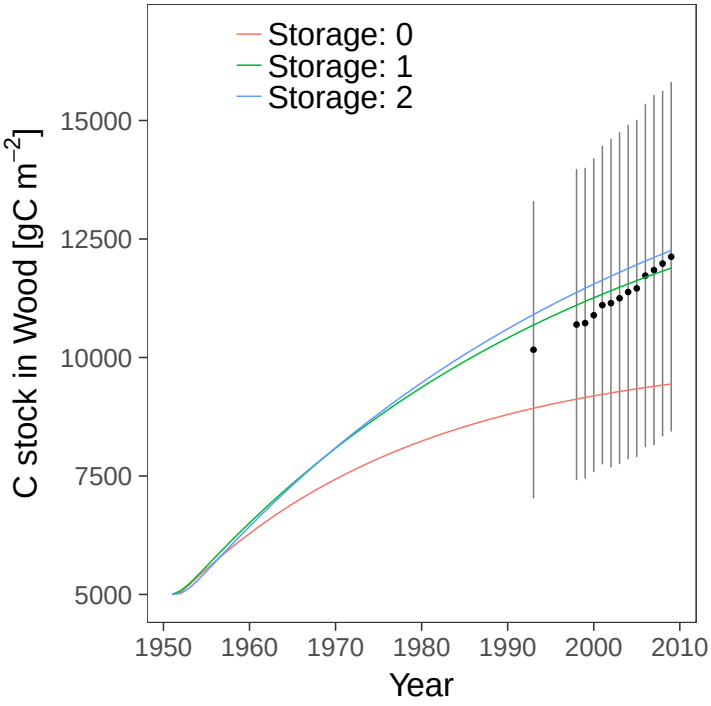


Figure 13: plot of chunk C_stocks_all

```
    data = d_F,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_line(aes(colour=col),lty=1) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.55,0.35), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.text = element_text(size = 22),
        legend.title = element_blank()) +
  xlab("Year") +
  ylab(bquote('C stocks in Foliage ['*gC~m^-2*']')) +
  guides(color=guide_legend(
    keywidth=0.3,
    keyheight=0.3,
    default.unit="inch"))
```

Fluxes of C released from the compartments of each model

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot(df_a, aes(x=time, y=rel))+
  geom_line(aes(colour=col)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  xlab("Year") +
  ylab(bquote('Released C ['*gC~m^-2*']')) +
  scale_color_manual(values=c("green","orange","red","brown")) +
  guides(color=guide_legend(
    keywidth=0.3,
    keyheight=0.3,
    default.unit="inch"))
```

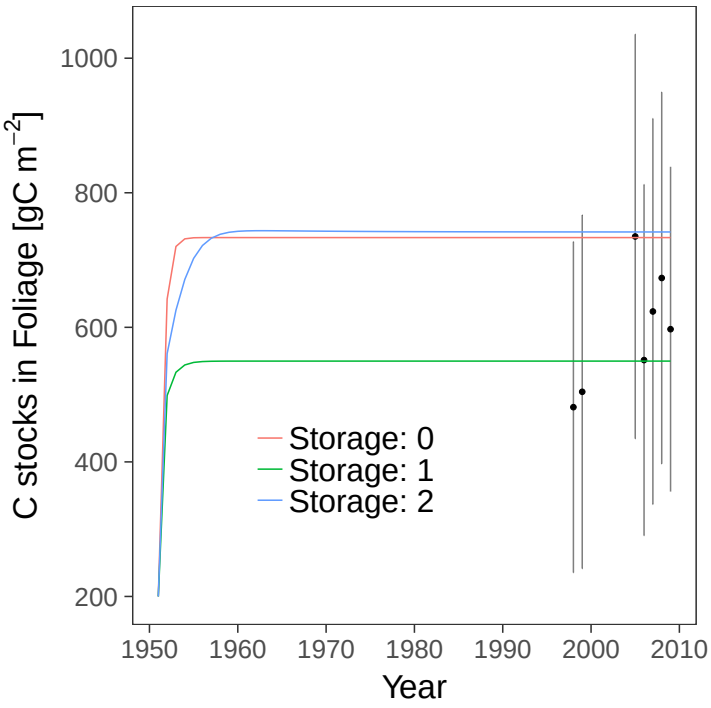


Figure 14: plot of chunk C_stocks_all

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot(df_b, aes(x=time, y=rel))+
  geom_line(aes(colour=col)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  xlab("Year") +
  ylab(bquote('Released C ['*gC~m^-2*']')) +
  scale_color_manual(values=c("green","orange","red","purple","brown")) +
  guides(color=guide_legend(
    keywidth=0.3,
    keyheight=0.3,
    default.unit="inch"))
```

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot(df_c, aes(x=time, y=rel))+
  geom_line(aes(colour=col)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  xlab("Year") +
  ylab(bquote('Released C ['*gC~m^-2*']')) +
  scale_color_manual(values=c("green","orange","red","blue","purple","brown")) +
  guides(color=guide_legend(
    keywidth=0.3,
    keyheight=0.3,
    default.unit="inch"))
```

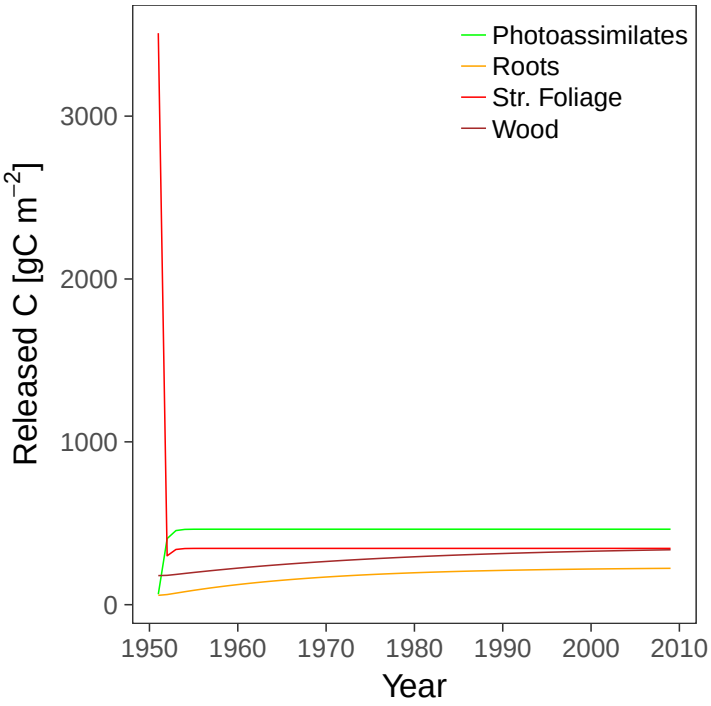


Figure 15: plot of chunk C_released_S0

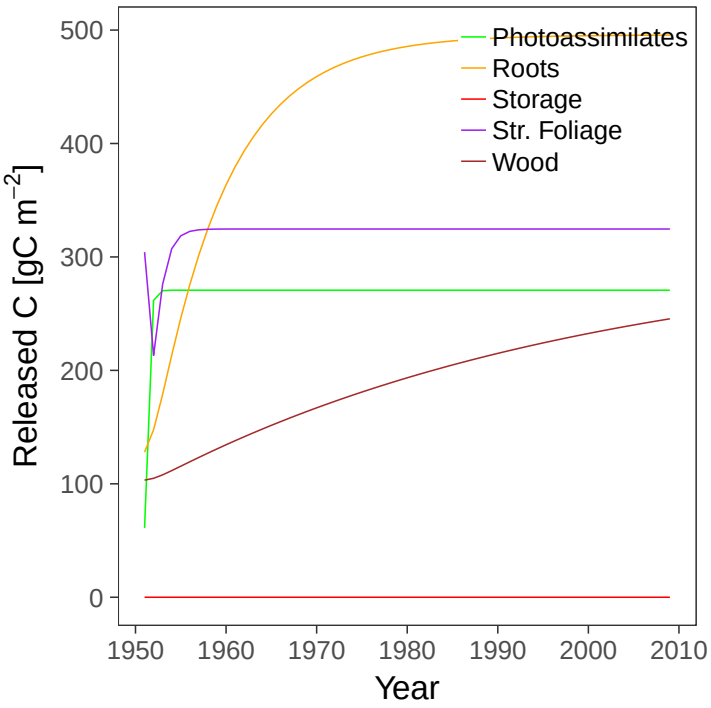


Figure 16: plot of chunk C_released_S1

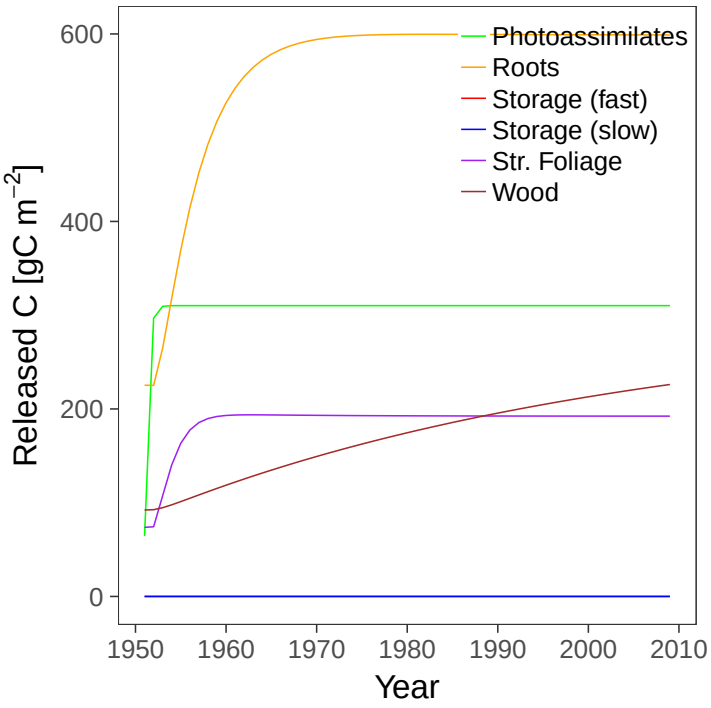


Figure 17: plot of chunk C_released_S2

Radiocarbon content in each compartment

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
plotC14Pool(t=sequence, mat=fC14ta, inputFc=Hua2013$NHZone2[,1:2], xlab="years",
            ylab="Radiocarbon in Delta 14C", col=2:5, cex.axis = 1.5, cex.lab = 1.6)
title(main = "Storage: 0", cex = 5, font = 10, col = 3)
legend("topright", inset=.01, legend=c("Photoassimilates", "Str. Foliage", "Wood", "Roots"),
       lty=1, col=2:5, horiz=FALSE, bty="n", cex = 1.5)

knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
plotC14Pool(t=sequence, mat=fC14tb, inputFc=Hua2013$NHZone2[,1:2], xlab="years",
            ylab="Radiocarbon in Delta 14C", col=c(2,6,3,4,5), cex.axis = 1.5, cex.lab = 1.6)
title(main = "Storage: 1", cex = 5, font = 10, col = 3)
legend("topright", inset=.01, legend=c("Photoassimilates", "Storage", "Str. Foliage", "Wood", "Roots"),
       lty=1, col=c(2,6,3,4,5), horiz=FALSE, bty="n", cex = 1.5)

knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
plotC14Pool(t=sequence, mat=fC14tc, inputFc=Hua2013$NHZone2[,1:2], xlab="years",
            ylab="Radiocarbon in Delta 14C", c(2,6,9,3,4,5), cex.axis = 1.5, cex.lab = 1.6)
title(main = "Storage: 2", cex = 5, font = 10, col = 3)
legend("topright", inset=.01, legend=c("Photoassimilates", "Storage (fast)", "Storage (slow)", "Str. Foliage", "Wood",
       "Roots"), lty=1, col=c(2,6,9,3,4,5), horiz=FALSE, bty="n", cex = 1.5)
```

Ages and transit times

We need the matrix **A** and the input vector **s** for a system in equilibrium (no time-dependencies). First, we calculate system and compartment ages using the function `systemAge` and for the transit times we use the function `transitTime`

```
# Model without storage compartment:
AgeVEGa=systemAge(A=A0(finpara),u=s(finpara), a=Age, q=c(0.05, 0.5, 0.95))
ttVEGa=transitTime(A=A0(finpara),u=s(finpara), a=Age, q=c(0.05, 0.5, 0.95))

# Model with one storage compartment:
AgeVEGb=systemAge(A=Ab0(finparb),u=sb(finpara), a=Age, q=c(0.05, 0.5, 0.95))
ttVEGb=transitTime(A=Ab0(finparb),u=sb(finpara), a=Age, q=c(0.05, 0.5, 0.95))
```

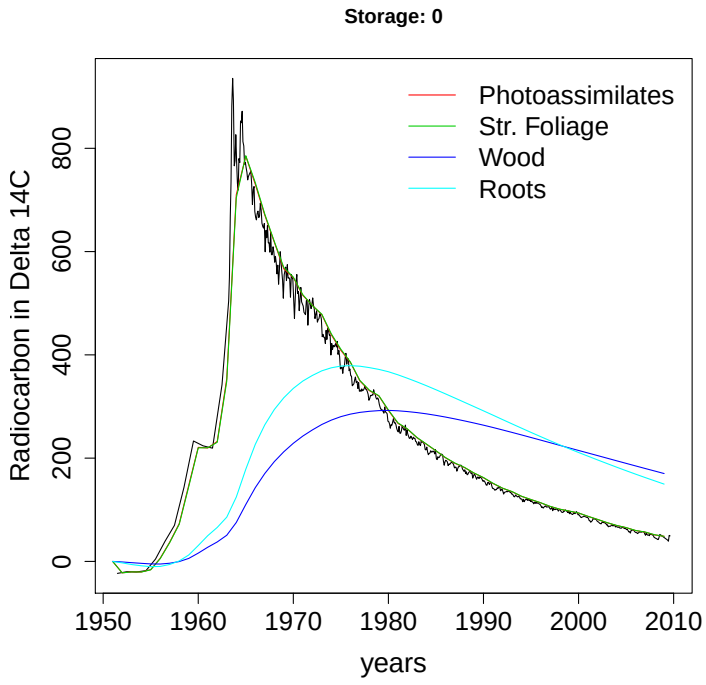


Figure 18: plot of chunk C14_S0

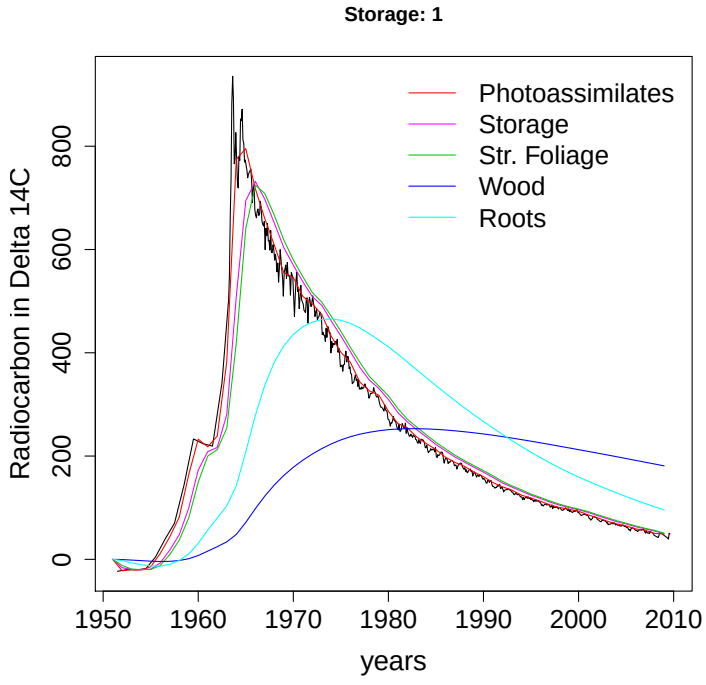


Figure 19: plot of chunk C14_S1

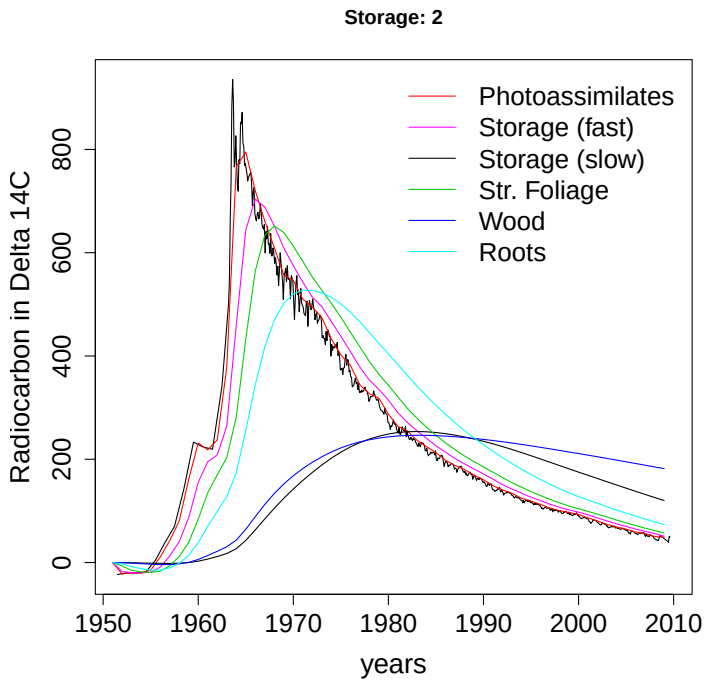


Figure 20: plot of chunk C14_S2

```
# Model with two storage compartments:
AgeVEGc=systemAge(A=Ac0(finparc),u=sc(finpara), a=Age, q=c(0.05, 0.5, 0.95))
ttVEGc=transitTime(A=Ac0(finparc),u=sc(finpara), a=Age, q=c(0.05, 0.5, 0.95))
```

For the mean, median, and confidence intervals of system age and transit time of each model we use the attributes `$mean` and `$quantiles`. We also prepare data frames for plotting age and transit time distributions

```
at_a <- data.frame("x"=Age,"age"=AgeVEGa$systemAgeDensity,"tt"=ttVEGa$transitTimeDensity)
at_a$name <- "Storage: 0"
at_b <- data.frame("x"=Age,"age"=AgeVEGb$systemAgeDensity,"tt"=ttVEGb$transitTimeDensity)
at_b$name <- "Storage: 1"
at_c <- data.frame("x"=Age,"age"=AgeVEGc$systemAgeDensity,"tt"=ttVEGc$transitTimeDensity)
at_c$name <- "Storage: 2"
DF <- do.call(rbind, list(at_a,at_b,at_c))

atm_a <- data.frame("mean_age"=AgeVEGa$meanSystemAge, "median_age"=AgeVEGa$quantilesSystemAge[2],
         "age CI.d"=AgeVEGa$quantilesSystemAge[1], "age CI.u"=AgeVEGa$quantilesSystemAge[3],
         "mean_tt"=ttVEGa$meanTransitTime, "median_tt"=ttVEGa$quantiles[2],
         "tt CI.d"=ttVEGa$quantiles[1], "tt CI.u"=ttVEGa$quantiles[3])
is.num <- sapply(atm_a, is.numeric)
atm_a[is.num] <- lapply(atm_a[is.num], round, 2)
# Info needed to limit geom_segment for median and means:
atm_a$mean_a_dens <- as.numeric(subset(DF, name == "Storage: 0" & x==atm_a$mean_age, select = age))
atm_a$median_a_dens <- as.numeric(subset(DF, name == "Storage: 0" & x==atm_a$median_age, select = age))
atm_a$mean_tt_dens <- as.numeric(subset(DF, name == "Storage: 0" & x==atm_a$mean_tt, select = tt))
atm_a$median_tt_dens <- as.numeric(subset(DF, name == "Storage: 0" & x==atm_a$median_tt, select = tt))
atm_a$name <- "Storage: 0"

atm_b <- data.frame("mean_age"=AgeVEGb$meanSystemAge, "median_age"=AgeVEGb$quantilesSystemAge[2],
         "age CI.d"=AgeVEGb$quantilesSystemAge[1], "age CI.u"=AgeVEGb$quantilesSystemAge[3],
         "mean_tt"=ttVEGb$meanTransitTime, "median_tt"=ttVEGb$quantiles[2],
         "tt CI.d"=ttVEGb$quantiles[1], "tt CI.u"=ttVEGb$quantiles[3])
is.num <- sapply(atm_b, is.numeric)
atm_b[is.num] <- lapply(atm_b[is.num], round, 2)
atm_b$mean_a_dens <- as.numeric(subset(DF, name == "Storage: 1" & x==atm_b$mean_age, select = age))
atm_b$median_a_dens <- as.numeric(subset(DF, name == "Storage: 1" & x==atm_b$median_age, select = age))
```

```
atm_b$mean_tt_dens <- as.numeric(subset(DF, name == "Storage: 1" & x==atm_b$mean_tt, select = tt))
atm_b$median_tt_dens <- as.numeric(subset(DF, name == "Storage: 1" & x==atm_b$median_tt, select = tt))
atm_b$name <- "Storage: 1"

atm_c <- data.frame("mean_age"=AgeVEGc$meanSystemAge, "median_age"=AgeVEGc$quantilesSystemAge[2],
         "age CI.d"=AgeVEGc$quantilesSystemAge[1], "age CI.u"=AgeVEGc$quantilesSystemAge[3],
         "mean_tt"=ttVEGc$meanTransitTime, "median_tt"=ttVEGc$quantiles[2],
         "tt CI.d"=ttVEGc$quantiles[1], "tt CI.u"=ttVEGc$quantiles[3])
is.num <- sapply(atm_c, is.numeric)
atm_c[is.num] <- lapply(atm_c[is.num], round, 2)
atm_c$mean_a_dens <- as.numeric(subset(DF, name == "Storage: 2" & x==atm_c$mean_age, select = age))
atm_c$median_a_dens <- as.numeric(subset(DF, name == "Storage: 2" & x==atm_c$median_age, select = age))
atm_c$mean_tt_dens <- as.numeric(subset(DF, name == "Storage: 2" & x==atm_c$mean_tt, select = tt))
atm_c$median_tt_dens <- as.numeric(subset(DF, name == "Storage: 2" & x==atm_c$median_tt, select = tt))
atm_c$name <- "Storage: 2"

DF_m <- do.call(rbind, list(atm_a,atm_b,atm_c))
DF_m
```

```
##   mean_age median_age age.CI.d age.CI.u mean_tt median_tt tt.CI.d tt.CI.u
## 1    24.27      16.12     0.72    75.70   10.59      0.99    0.06   51.73
## 2    38.15      21.18     0.86   131.21   14.60      3.30    0.09   73.18
## 3    44.12      25.40     0.88   150.16   14.94      3.77    0.09   80.26
##   mean_a_dens median_a_dens mean_tt_dens median_tt_dens       name
## 1 0.014371537    0.02025696  0.011718477     0.18425062 Storage: 0
## 2 0.007273221    0.01192867  0.011745318             NA Storage: 1
## 3 0.006524344            NA  0.008029032     0.07629909 Storage: 2
```

Graphically, we can see:

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot(data=DF_m) +
  geom_vline(aes(xintercept=mean_age,linetype = "Mean",colour=name)) +
  geom_vline(aes(xintercept=median_age,linetype = "Median",colour=name)) +
  scale_linetype_manual(values=c("longdash","dotted")) +
  geom_line(data=DF, aes(x=x,y=age,colour=name)) +
  theme_bw(21) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  xlim(0,75) +
  xlab("Age [years]") +
  ylab("Age density") +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  guides(color=guide_legend(
    keywidth=0.33,
    keyheight=0.33,
    default.unit="inch"))
```

Now, we plot age distributions for the compartments of each model

For the model without storage compartment:

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
par(mfrow=c(2,2))
for(i in 1:4){
  poolName <- c("Photoassimilates","Str. Foliage","Wood","Roots")
  plot(Age,AgeVEGa$poolAgeDensity[,i],type="l",  ylab="Probability density",
     xlab="Age [years]", xlim=c(0,30), bty="n", cex.axis = 1.45, cex.lab = 1.56)
  abline(v=AgeVEGa$meanPoolAge[i,1], lty=2)
  legend("topright", inset=.1, legend=poolName[i], horiz=FALSE, bty="n", cex = 1.56)
}
```

For the model with one storage compartment:

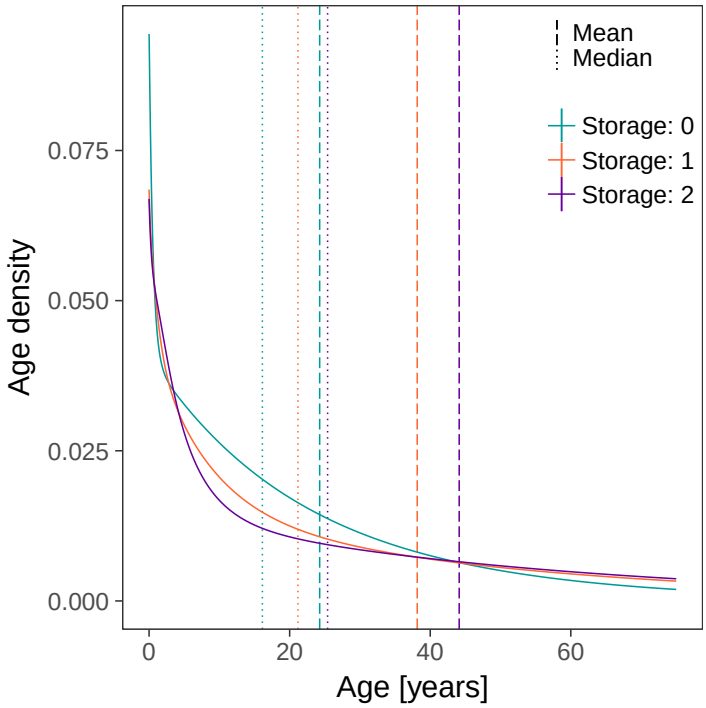


Figure 21: plot of chunk Age_dist_3_models

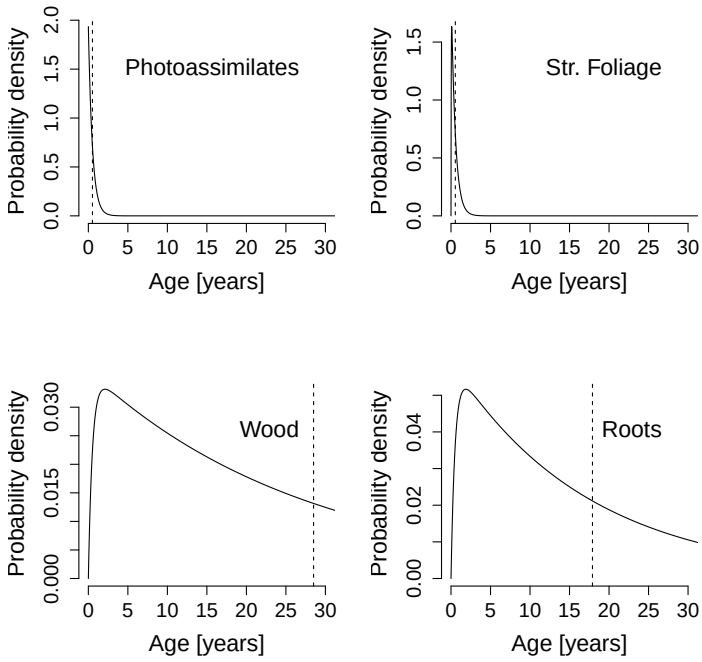


Figure 22: plot of chunk Age_dist_pools_S0

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
par(mfrow=c(2,3))
for(i in 1:5){
  poolName <- c("Photoassimilates","Storage","Str. Foliage","Wood","Roots")
  plot(Age,AgeVEGb$poolAgeDensity[,i],type="l",  ylab="Probability density",
       xlab="Age [years]", bty="n", xlim=c(0,30), cex.axis = 1.45, cex.lab = 1.56)
  abline(v=AgeVEGb$meanPoolAge[i,1], lty=2)
  legend("topright", inset=.1, legend=poolName[i], horiz=FALSE, bty="n", cex = 1.5)
}
```

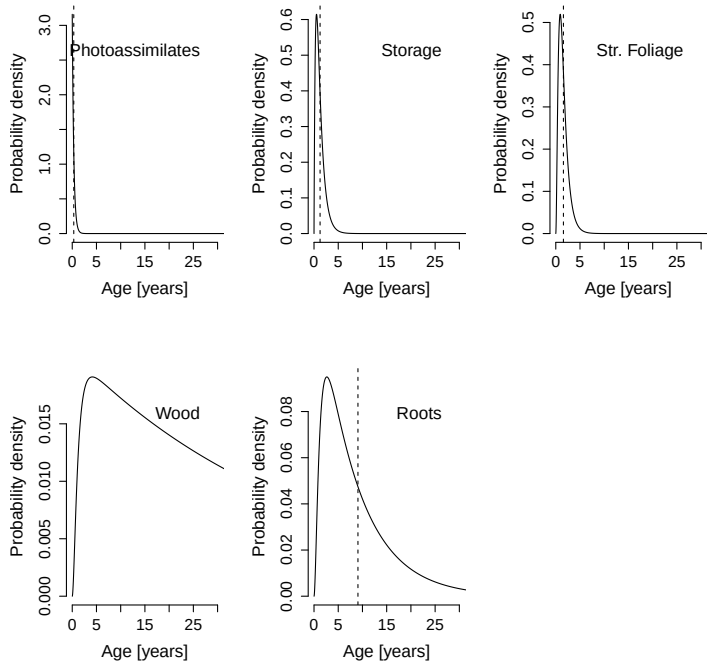


Figure 23: plot of chunk Age_dist_pools_S1

And for the model with two storage compartments:

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
par(mfrow=c(2,3))
for(i in 1:6){
  poolName <- c("Photoassimilates","Storage (fast)","Storage (slow)","Str. Foliage","Wood","Roots")
  plot(Age,AgeVEGc$poolAgeDensity[,i],type="l",  ylab="Probability density",
       xlab="Age [years]", bty="n", xlim=c(0,50), cex.axis = 1.45, cex.lab = 1.56)
  abline(v=AgeVEGc$meanPoolAge[i,1], lty=2)
  legend("topright", inset=.1, legend=poolName[i], horiz=FALSE, bty="n", cex = 1.56)
}
```

Comparing the compartments Photoassimilates, Str. Foliage, Wood, and Roots of the 3 models:

```
ap_a <- data.frame("x"=Age, "Cp"=AgeVEGa$poolAgeDensity[,1], "Cf"=AgeVEGa$poolAgeDensity[,2],
                   "Cw"=AgeVEGa$poolAgeDensity[,3], "Cr"=AgeVEGa$poolAgeDensity[,4])
ap_a$name <- "Storage: 0"
ap_b <- data.frame("x"=Age, "Cp"=AgeVEGb$poolAgeDensity[,1], "Cf"=AgeVEGb$poolAgeDensity[,3],
                   "Cw"=AgeVEGb$poolAgeDensity[,4], "Cr"=AgeVEGb$poolAgeDensity[,5])
ap_b$name <- "Storage: 1"
ap_c <- data.frame("x"=Age, "Cp"=AgeVEGc$poolAgeDensity[,1], "Cf"=AgeVEGc$poolAgeDensity[,4],
                   "Cw"=AgeVEGc$poolAgeDensity[,5], "Cr"=AgeVEGc$poolAgeDensity[,6])
ap_c$name <- "Storage: 2"
DF_p <- do.call(rbind, list(ap_a, ap_b, ap_c))

apm_a <- data.frame("Cp"=AgeVEGa$meanPoolAge[1,1], "Cf"=AgeVEGa$meanPoolAge[2,1],
                    "Cw"=AgeVEGa$meanPoolAge[3,1], "Cr"=AgeVEGa$meanPoolAge[4,1])
is.num <- sapply(apm_a, is.numeric)
```

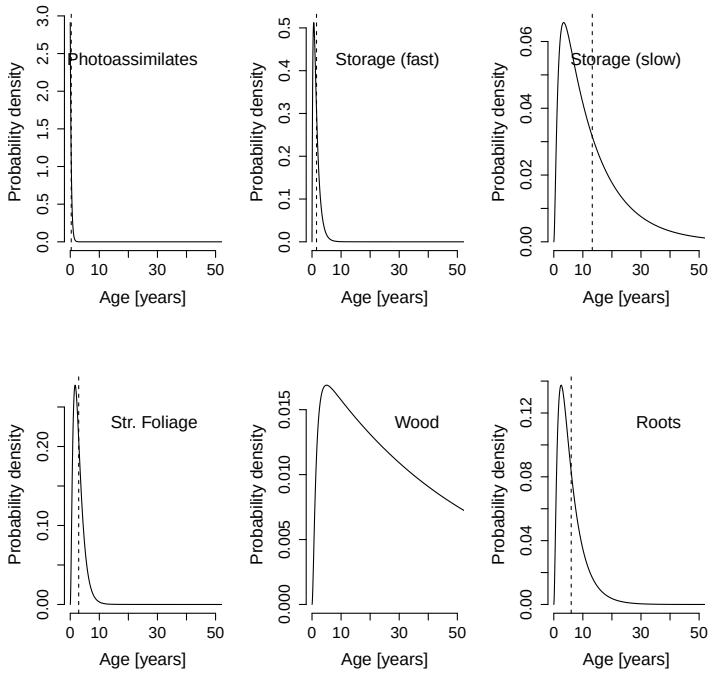


Figure 24: plot of chunk Age_dist_pools_S2

```
apm_a[is.num] <- lapply(apm_a[is.num], round, 2)
apm_a$Cp_dens <- as.numeric(subset(DF_p, name == "Storage: 0" & x==apm_a$Cp, select = Cp))
apm_a$Cf_dens <- as.numeric(subset(DF_p, name == "Storage: 0" & x==apm_a$Cf, select = Cf))
apm_a$Cw_dens <- as.numeric(subset(DF_p, name == "Storage: 0" & x==apm_a$Cw, select = Cw))
apm_a$Cr_dens <- as.numeric(subset(DF_p, name == "Storage: 0" & x==18, select = Cr))
apm_a$name <- "Storage: 0"
apm_b <- data.frame("Cp"=AgeVEGb$meanPoolAge[1,1], "Cf"=AgeVEGb$meanPoolAge[3,1],
                    "Cw"=AgeVEGb$meanPoolAge[4,1], "Cr"=AgeVEGb$meanPoolAge[5,1])
is.num <- sapply(apm_b, is.numeric)
apm_b[is.num] <- lapply(apm_b[is.num], round, 2)
apm_b$Cp_dens <- as.numeric(subset(DF_p, name == "Storage: 1" & x==apm_b$Cp, select = Cp))
apm_b$Cf_dens <- as.numeric(subset(DF_p, name == "Storage: 1" & x==apm_b$Cf, select = Cf))
apm_b$Cw_dens <- as.numeric(subset(DF_p, name == "Storage: 1" & x==apm_b$Cw, select = Cw))
apm_b$Cr_dens <- as.numeric(subset(DF_p, name == "Storage: 1" & x==apm_b$Cr, select = Cr))
apm_b$name <- "Storage: 1"
apm_c <- data.frame("Cp"=AgeVEGc$meanPoolAge[1,1], "Cf"=AgeVEGc$meanPoolAge[4,1],
                    "Cw"=AgeVEGc$meanPoolAge[5,1], "Cr"=AgeVEGc$meanPoolAge[6,1])
is.num <- sapply(apm_c, is.numeric)
apm_c[is.num] <- lapply(apm_c[is.num], round, 2)
apm_c$Cp_dens <- as.numeric(subset(DF_p, name == "Storage: 2" & x==apm_c$Cp, select = Cp))
apm_c$Cf_dens <- as.numeric(subset(DF_p, name == "Storage: 2" & x==apm_c$Cf, select = Cf))
apm_c$Cw_dens <- as.numeric(subset(DF_p, name == "Storage: 2" & x==apm_b$Cw, select = Cw))
apm_c$Cr_dens <- as.numeric(subset(DF_p, name == "Storage: 2" & x==apm_c$Cr, select = Cr))
apm_c$name <- "Storage: 2"
DF_pm <- do.call(rbind, list(apm_a, apm_b, apm_c))
#DF_pm <- as.data.frame(DF_pm)

# The mean age of each compartment:
DF_pm
```

```
##     Cp   Cf    Cw    Cr  Cp_dens   Cf_dens     Cw_dens    Cr_dens
## 1 0.52 0.55 28.51 17.90 0.707447 0.7065043 0.013144015 0.02105105
## 2 0.32 1.58 49.68  9.07 1.149553 0.3818045 0.007597834 0.04744171
## 3 0.34 2.91 55.78  5.99 1.081944 0.2087594 0.007594042 0.08323389
##         name
## 1 Storage: 0
## 2 Storage: 1
```

```
## 3 Storage: 2
```

1. Photoassimilates

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot() +
  ggtitle("Photoassimilates") +
  geom_vline(data=DF_pm,aes(xintercept=Cp,linetype = "Mean",colour=name)) +
  scale_linetype_manual(values="longdash") +
  geom_segment(data = DF_pm, aes(x = Cp, y = Cp_dens, xend = Cp, yend = Inf), colour="white") +
  geom_line(data=DF_p, aes(x=x,y=Cp,colour=name)) +
  geom_text(data = DF_pm, aes(label = Cp, y=Cp_dens+1.2, x=Cp), size=5.5, position=position_jitter(height=0.5)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(0,1) +
  xlab("Age [years]") +
  ylab("Age density") +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  guides(color=guide_legend(
    keywidth=0.33,
    keyheight=0.33,
    default.unit="inch"))
```

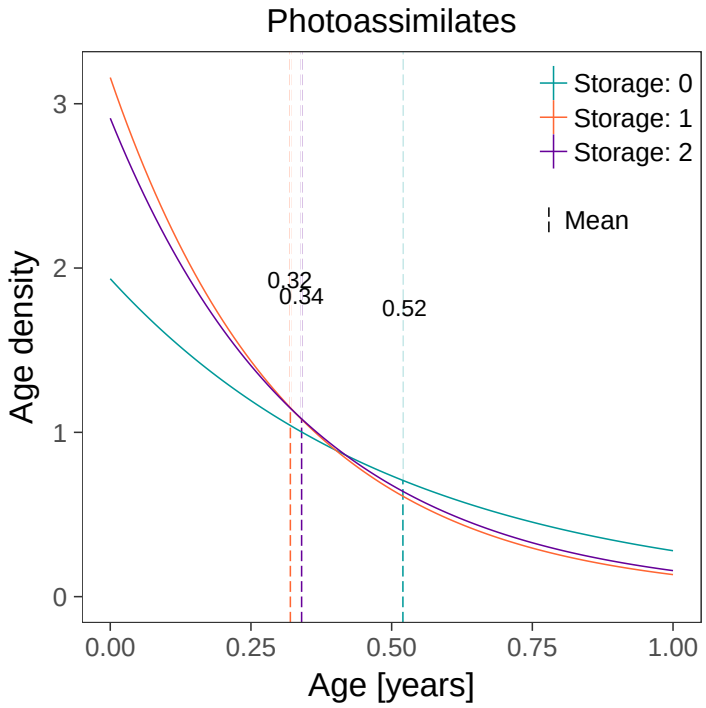


Figure 25: plot of chunk Age_dist_Cp

2. Str. Foliage

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot() +
  ggtitle("Str. Foliage") +
  geom_vline(data=DF_pm,aes(xintercept=Cf,linetype = "Mean",colour=name)) +
  scale_linetype_manual(values="longdash") +
  geom_segment(data = DF_pm, aes(x = Cf, y = Cf_dens, xend = Cf, yend = Inf), colour="white") +
  geom_line(data=DF_p, aes(x=x,y=Cf,colour=name)) +
```

```
  geom_text(data = DF_pm, aes(label = Cf, y=Cf_dens+0.2, x=Cf), size=5.5) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(0,5) +
  xlab("Age [years]") +
  ylab("Age density") +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  guides(color=guide_legend(
    keywidth=0.33,
    keyheight=0.33,
    default.unit="inch"))
```

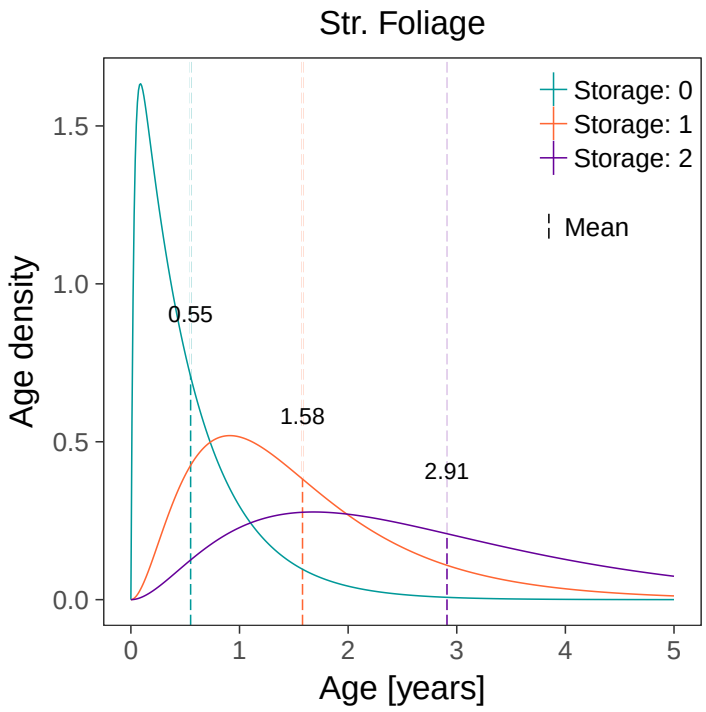


Figure 26: plot of chunk Age_dist_Cf

3. Wood

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot() +
  ggtitle("Wood") +
  geom_vline(data=DF_pm,aes(xintercept=Cw,linetype = "Mean",colour=name)) +
  scale_linetype_manual(values="longdash") +
  geom_segment(data = DF_pm, aes(x = Cw, y = Cw_dens, xend = Cw, yend = Inf), colour="white") +
  geom_line(data=DF_p, aes(x=x,y=Cw,colour=name)) +
  geom_text(data = DF_pm, aes(label = Cw, y=Cw_dens+0.005, x=Cw), size=5.5) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(0,100) +
  xlab("Age [years]") +
  ylab("Age density") +
```

```
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  guides(color=guide_legend(
    keywidth=0.33,
    keyheight=0.33,
    default.unit="inch"))
```

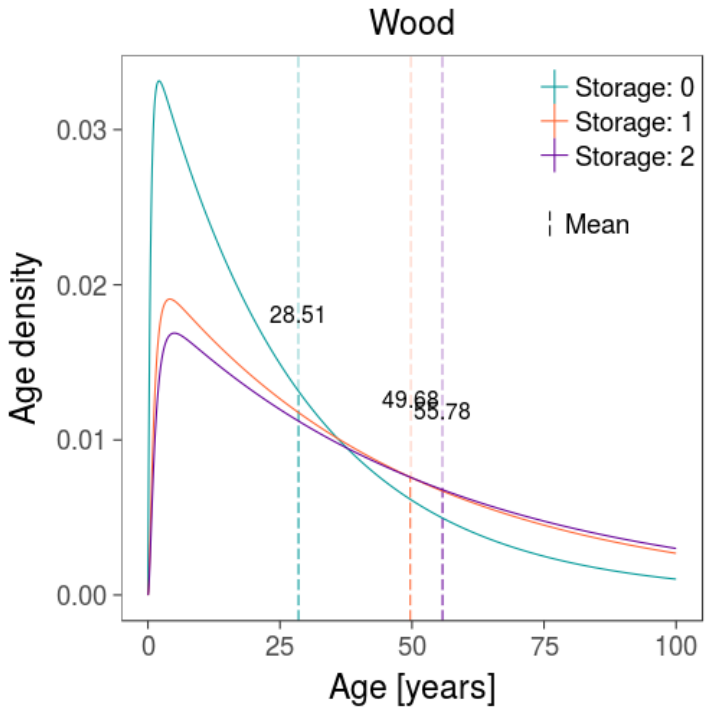


Figure 27: plot of chunk Age_dist_Cw

4. Roots

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot() +
  ggtitle("Roots") +
  geom_vline(data=DF_pm,aes(xintercept=Cr,linetype = "Mean",colour=name)) +
  scale_linetype_manual(values="longdash") +
  geom_segment(data = DF_pm, aes(x = Cr, y = Cr_dens, xend = Cr, yend = Inf), colour="white") +
  geom_line(data=DF_p, aes(x=x,y=Cr,colour=name)) +
  geom_text(data = DF_pm, aes(label = Cr, y=Cr_dens+0.02, x=Cr), size=5.5) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.8,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(0,20) +
  xlab("Age [years]") +
  ylab("Age density") +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  guides(color=guide_legend(
    keywidth=0.33,
    keyheight=0.33,
    default.unit="inch"))
```

Comparing fast cycling storage compartments:

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
as_b <- data.frame("x"=Age, "Cs"=AgeVEGb$poolAgeDensity[,2])
as_b$name <- "Storage: 1"
```

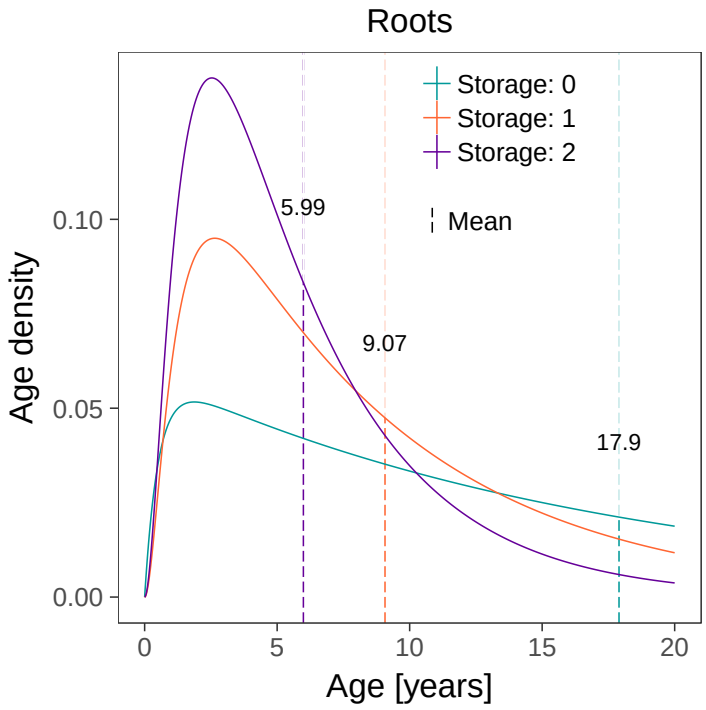


Figure 28: plot of chunk Age_dist_Cr

```
as_c <- data.frame("x"=Age, "Cs"=AgeVEGc$poolAgeDensity[,2])
as_c$name <- "Storage: 2"
DF_s <- do.call(rbind, list(as_b, as_c))

asm_b <- data.frame("Cs"=AgeVEGb$meanPoolAge[2,1])
is.num <- sapply(asm_b, is.numeric)
asm_b[is.num] <- lapply(asm_b[is.num], round, 2)
asm_b$Cs_dens <- as.numeric(subset(DF_s, name == "Storage: 1" & x==asm_b$Cs, select = Cs))
asm_b$name <- "Storage: 1"
asm_c <- data.frame("Cs"=AgeVEGc$meanPoolAge[2,1])
is.num <- sapply(asm_c, is.numeric)
asm_c[is.num] <- lapply(asm_c[is.num], round, 2)
asm_c$Cs_dens <- as.numeric(subset(DF_s, name == "Storage: 2" & x==asm_c$Cs, select = Cs))
asm_c$name <- "Storage: 2"
DF_sm <- do.call(rbind, list(asm_b, asm_c))

ggplot() +
  ggtitle("Fast cycling storage") +
  geom_vline(data=DF_sm,aes(xintercept=Cs,linetype = "Mean",colour=name)) +
  scale_linetype_manual(values="longdash") +
  geom_segment(data = DF_sm, aes(x = Cs, y = Cs_dens, xend = Cs, yend = Inf), colour="white") +
  geom_line(data=DF_s, aes(x=x,y=Cs,colour=name)) +
  theme_bw(23) +
  geom_text(data = DF_sm, aes(label = Cs, y=Cs_dens+0.1, x=Cs), size=6) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.85,1), legend.justification = c(0.5,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(0,5) +
  xlab("Age [years]") +
  ylab("Age density") +
  scale_color_manual(values=c("#FF6633","#660099")) +
  guides(color=guide_legend(
    keywidth=0.33,
    keyheight=0.33,
```

```
    default.unit="inch"))
```

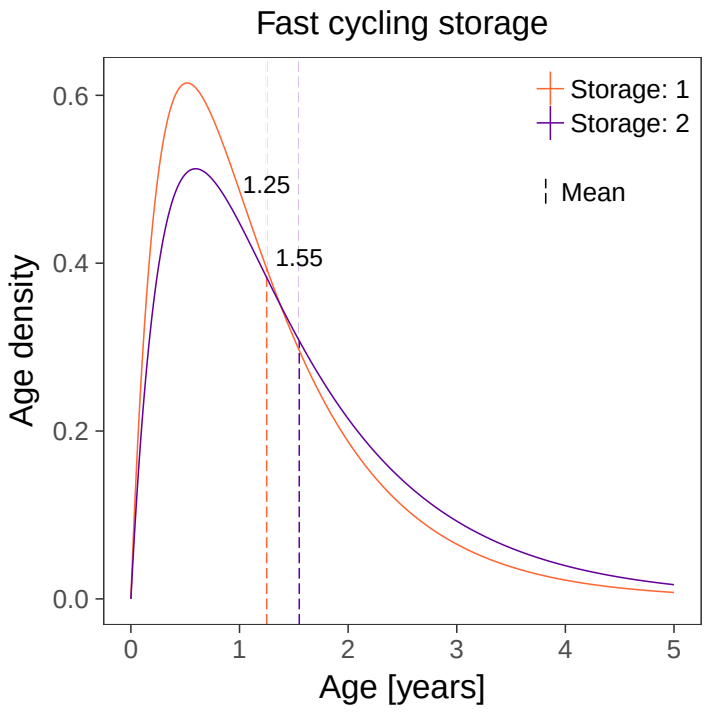


Figure 29: plot of chunk df_storage_age

Plotting only the slow cycling storage compartment:

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
DF_s <- data.frame("x"=Age, "Cs"=AgeVEGc$poolAgeDensity[,3])
DF_s$name <- "Storage: 2"

DF_sm <- data.frame("Cs"=AgeVEGc$meanPoolAge[3,1])
is.num <- sapply(DF_sm, is.numeric)
DF_sm[is.num] <- lapply(DF_sm[is.num], round, 2)
DF_sm$Cs_dens <- as.numeric(subset(DF_s, name == "Storage: 2" & x==DF_sm$Cs, select = Cs))
DF_sm$name <- "Storage: 2"

ggplot() +
  ggtitle("Slow cycling storage") +
  geom_vline(data=DF_sm,aes(xintercept=Cs,linetype = "Mean",colour=name)) +
  scale_linetype_manual(values="longdash") +
  geom_segment(data = DF_sm, aes(x = Cs, y = Cs_dens, xend = Cs, yend = Inf), colour="white") +
  geom_line(data=DF_s, aes(x=x,y=Cs,colour=name)) +
  theme_bw(23) +
  geom_text(data = DF_sm, aes(label = Cs, y=Cs_dens+0.02, x=Cs), size=6) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.85,1), legend.justification = c(0.5,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(0,50) +
  xlab("Age [years]") +
  ylab("Age density") +
  scale_color_manual(values=("#660099")) +
  guides(color=guide_legend(
    keywidth=0.33,
    keyheight=0.33,
    default.unit="inch"))
```

And the transit time distribution

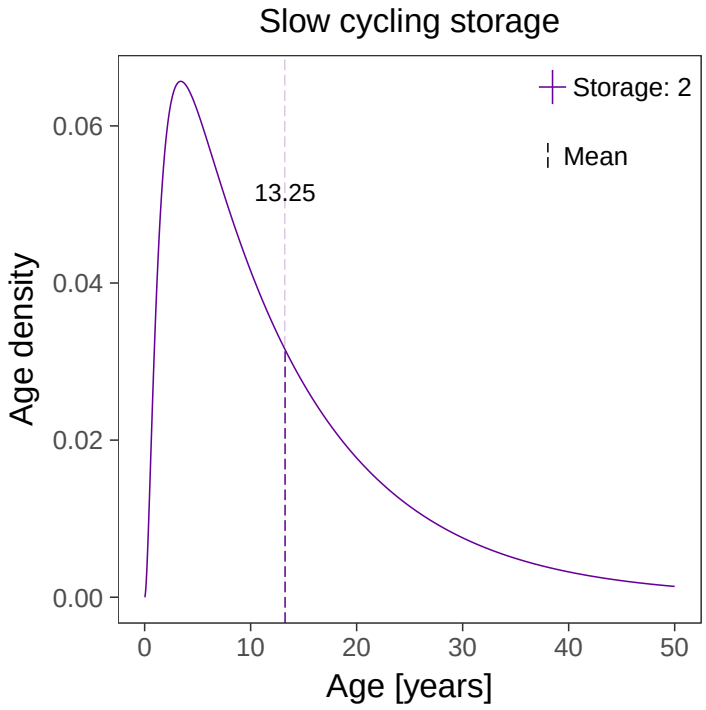


Figure 30: plot of chunk df_slow_storage_age

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
ggplot() +
  geom_vline(data=DF_m,aes(xintercept=mean_tt,linetype = "Mean",colour=name)) +
  geom_vline(data=DF_m,aes(xintercept=median_tt,linetype = "Median",colour=name)) +
  scale_linetype_manual(values=c("longdash","dotted")) +
  geom_line(data = DF, aes(x=x,y=tt,colour=name)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.55,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  xlim(0,18) +
  xlab("Time [years]") +
  ylab("Transit time density") +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  guides(color=guide_legend(
    keywidth=0.33,
    keyheight=0.33,
    default.unit="inch"))
```

Uncertainty analysis

In order to explore model simulations that could result from different parameter sets that are possible and likely, we extract the parameter sets from one of the output `pars` of the function `modMCMC`; then we run again the models using each of these sets, and calculate again the released C, the ages and transit times.

From the new model runs, we calculate the weighted mean and standard deviation of the C release fluxes, as well as mean ages and transit times. The weights are the times in which each of the parameters was repeated in `pars`. Finally, we produce the plots to compare the models in regards to the C released by each compartment.

```
RelSS <- function #Release at steady state
  ### Computes the C release at steady state from a matrix representation of a compartmental model
  (A,  ##<< A compartmental linear square matrix with cycling rates in the diagonal
       ## and transfer rates in the off-diagonal.
   u ##<< A vector with input fluxes
```

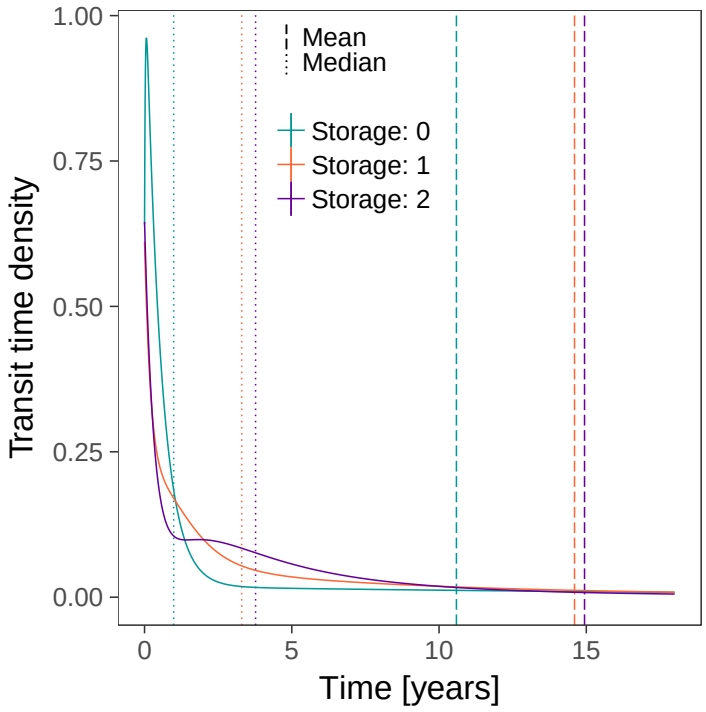


Figure 31: plot of chunk Transit_time_distribution

```
  )
  {
    d=dim(A)[1]
    one=matrix(1,nrow=d,ncol=1)
    zT=-1*t(one)%*%A
    xss=t((-1*solve(A))%*%u)
    z= zT*xss
    return(z)
  }

# For the model without storage compartment:
fmAPars <- fmA$pars # All Pars
ParsSpaceA <- fmAPars[sample(nrow(fmAPars),1000),]
# Calculate number of times in which each parameter set is repeated in 'pars'
weightsA <- plyr::count(ParsSpaceA)
weightsA <- weightsA[,length(weightsA)]
# Select the unique parameter combinations
ParsSpaceA <- unique(ParsSpaceA)

modelsA <- vector("list",dim(ParsSpaceA)[1])
zTA <- as.data.frame(matrix(0, ncol = length(c0), nrow = dim(ParsSpaceA)[1]))
SysAgesA <- numeric(length = dim(ParsSpaceA)[1])
TransitA <- numeric(length = dim(ParsSpaceA)[1])
PoolAgesA <- vector("list",dim(ParsSpaceA)[1])

for(i in 1:dim(ParsSpaceA)[1]){
  pars <- ParsSpaceA[i,]
    A=A0(pars)
    u=s(pars)
    a=Age
    modelsA[[i]] <- modelA(pars)
    zTA[i,]=RelSS(A,u)
    age = systemAge(A,u,a)
    SysAgesA[i] <- age$meanSystemAge
    PoolAgesA[[i]] <- age$meanPoolAge
    Transit = transitTime(A,u,a)
    TransitA[i] <- Transit$meanTransitTime
```

```
}

names(zTA) <- c("Cp","Cf","Cw","Cr")

saveRDS(modelsA,file = "modelsA.Rda")
#modelsA <- readRDS(file = "modelsA.Rda")
saveRDS(zTA,file = "zTA.Rda")
#zTA <- readRDS(file = "zTA.Rda")
saveRDS(SysAgesA,file = "SysAgesA.Rda")
#SysAgesA <- readRDS(file = "SysAgesA.Rda")
saveRDS(TransitA,file = "TransitA.Rda")
#TransitA <- readRDS(file = "TransitA.Rda")

# For the model with one storage compartment:
fmBPars <- fmB$pars # All Pars
ParsSpaceB <- fmBPars[sample(nrow(fmBPars),1000),]
# Calculate number of times in which each parameter set is repeated in 'pars'
weightsB <- plyr::count(ParsSpaceB)
weightsB <- weightsB[,length(weightsB)]
# Select the unique parameter combinations
ParsSpaceB <- unique(ParsSpaceB)

modelsB <- vector("list",dim(ParsSpaceB)[1])
zTB <- as.data.frame(matrix(0, ncol = length(c0b), nrow = dim(ParsSpaceB)[1]))
SysAgesB <- numeric(length = dim(ParsSpaceB)[1])
TransitB <- numeric(length = dim(ParsSpaceB)[1])
PoolAgesB <- vector("list",dim(ParsSpaceB)[1])

for(i in 1:dim(ParsSpaceB)[1]){
  pars <- ParsSpaceB[i,]
    A=Ab0(pars)
    u=sb(pars)
    a=Age
    modelsB[[i]] <- modelB(pars)
    zTB[i,]=RelSS(A,u)
    age = systemAge(A,u,a)
    SysAgesB[i] <- age$meanSystemAge
    PoolAgesB[[i]] <- age$meanPoolAge
    Transit = transitTime(A,u,a)
    TransitB[i] <- Transit$meanTransitTime
}

names(zTB) <- c("Cp","Cs","Cf","Cw","Cr")

saveRDS(modelsB,file = "modelsB.Rda")
#modelsB <- readRDS(file = "modelsB.Rda")
saveRDS(zTB,file = "zTB.Rda")
#zTB <- readRDS(file = "zTB.Rda")
saveRDS(SysAgesB,file = "SysAgesB.Rda")
#SysAgesB <- readRDS(file = "SysAgesB.Rda")
saveRDS(TransitB,file = "TransitB.Rda")
#TransitB <- readRDS(file = "TransitB.Rda")

# For the model with two storage compartments:
fmCPars <- fmC$pars # All Pars
ParsSpaceC <- fmCPars[sample(nrow(fmCPars),1000),]
# Calculate number of times in which each parameter set is repeated in 'pars'
weightsC <- plyr::count(ParsSpaceC)
weightsC <- weightsC[,length(weightsC)]
# Select the unique parameter combinations
ParsSpaceC <- unique(ParsSpaceC)

modelsC <- vector("list",dim(ParsSpaceC)[1])
zTC <- as.data.frame(matrix(0, ncol = length(c0c), nrow = dim(ParsSpaceC)[1]))
SysAgesC <- numeric(length = dim(ParsSpaceC)[1])
TransitC <- numeric(length = dim(ParsSpaceC)[1])
```

```
PoolAgesC <- vector("list",dim(ParsSpaceC)[1])

for(i in 1:dim(ParsSpaceC)[1]){
  pars <- ParsSpaceC[i,]
    A=Ac0(pars)
    u=sc(pars)
    a=Age
    modelsC[[i]] <- modelC(pars)
    zTC[i,]=RelSS(A,u)
    age = systemAge(A,u,a)
    SysAgesC[i] <- age$meanSystemAge
    PoolAgesC[[i]] <- age$meanPoolAge
    Transit = transitTime(A,u,a)
    TransitC[i] <- Transit$meanTransitTime
}

names(zTC) <- c("Cp","Cs1","Cs2","Cf","Cw","Cr")

saveRDS(modelsC,file = "modelsC.Rda")
#modelsC <- readRDS(file = "modelsC.Rda")
saveRDS(zTC,file = "zTC.Rda")
#zTC <- readRDS(file = "zTC.Rda")
saveRDS(SysAgesC,file = "SysAgesC.Rda")
#SysAgesC <- readRDS(file = "SysAgesC.Rda")
saveRDS(TransitC,file = "TransitC.Rda")
#TransitC <- readRDS(file = "TransitC.Rda")

# And for the C stocks
DFafterExploringPS <- function #C stocks after parameter exploration
##Prepares the data frames using the information from the parameter space exploration, to produce plots with means and ribb
  (d, ##<< Data from compartment
   w ##<< Number of repetitions of parameter sets in 'par'
  )
  {
    Data <- data.frame(d[,2:dim(d)[2]])
    mean <- apply(Data, 1, function(x) wtd.mean(x, w))
    median <- rowWeightedMedians(x=Data, w = w)
    #median <- median(rep(Data, times=w))
    sd <- sqrt(apply(Data, 1, function(x) wtd.var(x, w)))
    max <- as.numeric(mean)+as.numeric(sd)
    min <- as.numeric(mean)-as.numeric(sd)
    #max <- apply(Data, 1, max)
    #min <- apply(Data, 1, min)
    df <- data.frame("tm1"=d[,1],mean,min,median,max)
    return(df)
   }

# For the model without storage compartment:

Cp_a <- as.data.frame(sequence)
Cf_a <- as.data.frame(sequence)
Cw_a <- as.data.frame(sequence)
Cr_a <- as.data.frame(sequence)

for(i in 1:length(modelsA)){
  Ct <- getC(modelsA[[i]])
  Cp_a <- cbind(Cp_a,Ct[,1])
  Cf_a <- cbind(Cf_a,Ct[,2])
  Cw_a <- cbind(Cw_a,Ct[,3])
  Cr_a <- cbind(Cr_a,Ct[,4])
}
saveRDS(Cp_a,file = "Cp_a.Rda")
#Cp_a <- readRDS(file = "Cp_a.Rda")
saveRDS(Cf_a,file = "Cf_a.Rda")
#Cf_a <- readRDS(file = "Cf_a.Rda")
saveRDS(Cw_a,file = "Cw_a.Rda")
```

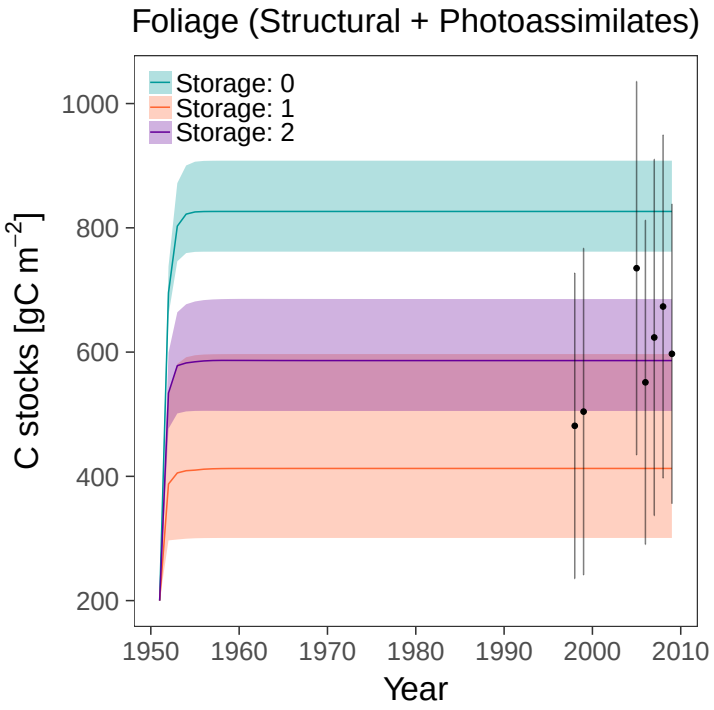
```
#Cw_a <- readRDS(file = "Cw_a.Rda")
saveRDS(Cr_a,file = "Cr_a.Rda")
#Cr_a <- readRDS(file = "Cr_a.Rda")

Cl_a = Cp_a + Cf_a
Cl_a <- DFafterExploringPS(d = Cl_a, w = weightsA)
Cl_a$tm1 <- sequence
Cp_a <- DFafterExploringPS(d = Cp_a, w = weightsA)
Cf_a <- DFafterExploringPS(d = Cf_a, w = weightsA)
Cw_a <- DFafterExploringPS(d = Cw_a, w = weightsA)
Cr_a <- DFafterExploringPS(d = Cr_a, w = weightsA)

# For the model with one storage compartment:

Cp_b <- as.data.frame(sequence)
Cs_b <- as.data.frame(sequence)
Cf_b <- as.data.frame(sequence)
Cw_b <- as.data.frame(sequence)
Cr_b <- as.data.frame(sequence)

for(i in 1:length(modelsB)){
  Ct <- getC(modelsB[[i]])
  Cp_b <- cbind(Cp_b,Ct[,1])
  Cs_b <- cbind(Cs_b,Ct[,2])
  Cf_b <- cbind(Cf_b,Ct[,3])
  Cw_b <- cbind(Cw_b,Ct[,4])
  Cr_b <- cbind(Cr_b,Ct[,5])
}
saveRDS(Cp_b,file = "Cp_b.Rda")
#Cp_b <- readRDS(file = "Cp_b.Rda")
saveRDS(Cs_b,file = "Cs_b.Rda")
#Cs_b <- readRDS(file = "Cs_b.Rda")
saveRDS(Cf_b,file = "Cf_b.Rda")
#Cf_b <- readRDS(file = "Cf_b.Rda")
saveRDS(Cw_b,file = "Cw_b.Rda")
#Cw_b <- readRDS(file = "Cw_b.Rda")
saveRDS(Cr_b,file = "Cr_b.Rda")
#Cr_b <- readRDS(file = "Cr_b.Rda")

Cl_b = Cp_b + Cf_b
Cl_b <- DFafterExploringPS(d = Cl_b, w = weightsB)
Cl_b$tm1 <- sequence
Cp_b <- DFafterExploringPS(d = Cp_b, w = weightsB)
Cs_b <- DFafterExploringPS(d = Cs_b, w = weightsB)
Cf_b <- DFafterExploringPS(d = Cf_b, w = weightsB)
Cw_b <- DFafterExploringPS(d = Cw_b, w = weightsB)
Cr_b <- DFafterExploringPS(d = Cr_b, w = weightsB)

#And for the model with two storage compartments:

Cp_c <- as.data.frame(sequence)
Cs1_c <- as.data.frame(sequence)
Cs2_c <- as.data.frame(sequence)
Cf_c <- as.data.frame(sequence)
Cw_c <- as.data.frame(sequence)
Cr_c <- as.data.frame(sequence)

for(i in 1:length(modelsC)){
  Ct <- getC(modelsC[[i]])
  Cp_c <- cbind(Cp_c,Ct[,1])
  Cs1_c <- cbind(Cs1_c,Ct[,2])
  Cs2_c <- cbind(Cs2_c,Ct[,3])
  Cf_c <- cbind(Cf_c,Ct[,4])
  Cw_c <- cbind(Cw_c,Ct[,5])
  Cr_c <- cbind(Cr_c,Ct[,6])
}
```

```
saveRDS(Cp_c,file = "Cp_c.Rda")
#Cp_c <- readRDS(file = "Cp_c.Rda")
saveRDS(Cs1_c,file = "Cs1_c.Rda")
#Cs1_c <- readRDS(file = "Cs1_c.Rda")
saveRDS(Cs2_c,file = "Cs2_c.Rda")
#Cs2_c <- readRDS(file = "Cs2_c.Rda")
saveRDS(Cf_c,file = "Cf_c.Rda")
#Cf_c <- readRDS(file = "Cf_c.Rda")
saveRDS(Cw_c,file = "Cw_c.Rda")
#Cw_c <- readRDS(file = "Cw_c.Rda")
saveRDS(Cr_c,file = "Cr_c.Rda")
#Cr_c <- readRDS(file = "Cr_c.Rda")

Cl_c = Cp_c + Cf_c
Cl_c <- DFafterExploringPS(d = Cl_c, w = weightsC)
Cl_c$tm1 <- sequence
Cp_c <- DFafterExploringPS(d = Cp_c, w = weightsC)
Cs1_c <- DFafterExploringPS(d = Cs1_c, w = weightsC)
Cs2_c <- DFafterExploringPS(d = Cs2_c, w = weightsC)
Cf_c <- DFafterExploringPS(d = Cf_c, w = weightsC)
Cw_c <- DFafterExploringPS(d = Cw_c, w = weightsC)
Cr_c <- DFafterExploringPS(d = Cr_c, w = weightsC)
```

Graphically we can compare:

0. C stocks of Photoassimilates + Str. Foliage

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)

Cl_a$name <- "Storage: 0"
Cl_b$name <- "Storage: 1"
Cl_c$name <- "Storage: 2"
DF <- do.call(rbind, list(Cl_a, Cl_b, Cl_c))
ggplot(DF, aes(x= tm1, ymin= min, ymax= max, y=median)) +
  ggtitle("Foliage (Structural + Photoassimilates)") +
  geom_ribbon(aes(fill=name), alpha=0.3) +
  geom_line(aes(colour=name)) +
  geom_point(data = d_F, aes(time, mean)) +
  scale_shape_manual(values = 15) +
  geom_errorbar(
    data = d_F,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.3,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(1951, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  scale_fill_manual(values=c("#009999","#FF6633","#660099"))
```



1. C stocks of Photoassimilates

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)

Cp_a$name <- "Storage: 0"
Cp_b$name <- "Storage: 1"
Cp_c$name <- "Storage: 2"
DF <- do.call(rbind, list(Cp_a, Cp_b, Cp_c))
ggplot(DF, aes(x= tm1, ymin= min, ymax= max, y=median)) +
  ggtitle("Photoassimilates") +
  geom_ribbon(aes(fill=name), alpha=0.3) +
  geom_line(aes(colour=name)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(1951, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  scale_fill_manual(values=c("#009999","#FF6633","#660099"))
```

2. C stocks of Str. Foliage

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)

Cf_a$name <- "Storage: 0"
Cf_b$name <- "Storage: 1"
Cf_c$name <- "Storage: 2"
DF <- do.call(rbind, list(Cf_a, Cf_b, Cf_c))
ggplot(DF, aes(x= tm1, ymin= min, ymax= max, y=median)) +
  ggtitle("Str. Foliage") +
  geom_point(data = d_F, aes(time, mean)) +
  scale_shape_manual(values = 15) +
  geom_errorbar(
    data = d_F,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_ribbon(aes(fill=name), alpha=0.3) +
```

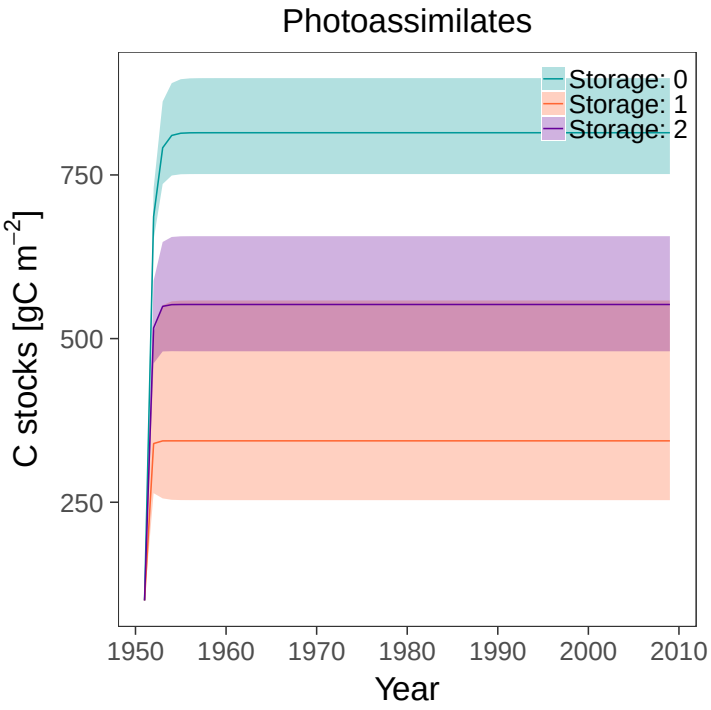


Figure 32: plot of chunk Explore_stocks_Cp

```
  geom_line(aes(colour=name)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.34,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(1951, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  scale_fill_manual(values=c("#009999","#FF6633","#660099"))
```

3. C stocks of Wood

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)

Cw_a$name <- "Storage: 0"
Cw_b$name <- "Storage: 1"
Cw_c$name <- "Storage: 2"
DF <- do.call(rbind, list(Cw_a, Cw_b, Cw_c))
ggplot(DF, aes(x= tm1, ymin= min, ymax= max, y=median)) +
  ggtitle("Wood") +
  geom_point(data = d_W, aes(time, mean)) +
  scale_shape_manual(values = 15) +
  geom_errorbar(
    data = d_W,
    aes(time, mean, ymin = min, ymax = max),
    width = 0.05,alpha=0.5) +
  geom_ribbon(aes(fill=name), alpha=0.3) +
  geom_line(aes(colour=name)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.34,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
```

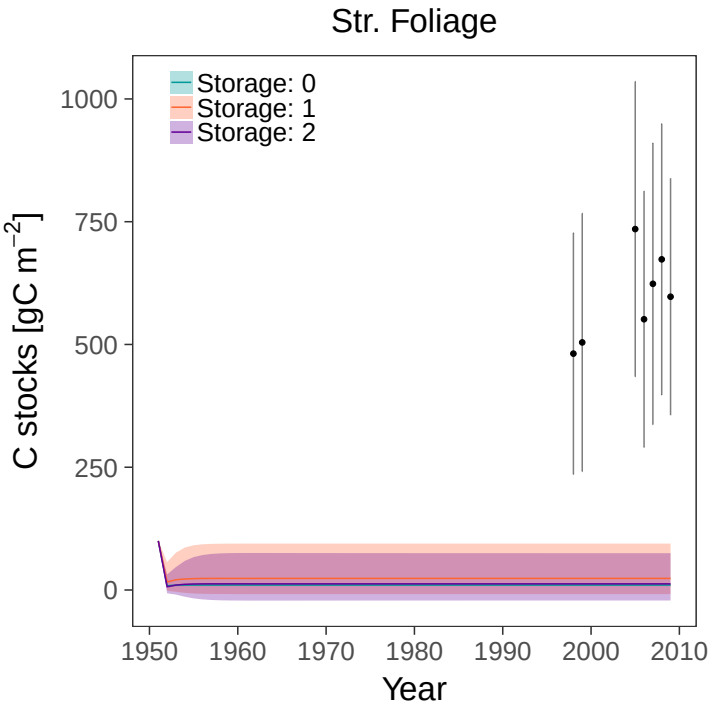


Figure 33: plot of chunk Explore_stocks_Cf

```
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(1951, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  scale_fill_manual(values=c("#009999","#FF6633","#660099"))
```

4. C stocks of Roots

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)

Cr_a$name <- "Storage: 0"
Cr_b$name <- "Storage: 1"
Cr_c$name <- "Storage: 2"
DF <- do.call(rbind, list(Cr_a, Cr_b, Cr_c))
ggplot(DF, aes(x= tm1, ymin= min, ymax= max, y=median)) +
  ggtitle("Roots") +
  geom_ribbon(aes(fill=name), alpha=0.3) +
  geom_line(aes(colour=name)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(0.34,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(1951, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  scale_fill_manual(values=c("#009999","#FF6633","#660099"))
```

5. C stocks of Storage

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
```

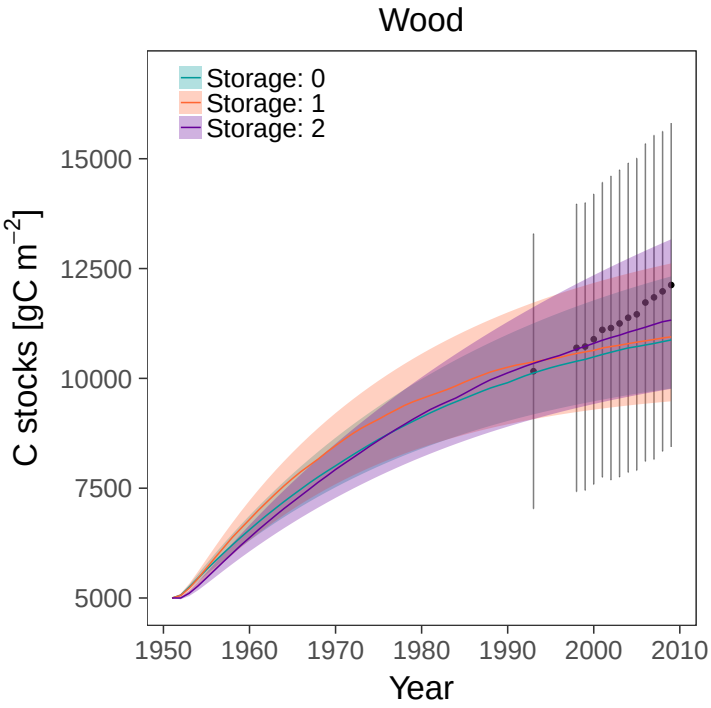


Figure 34: plot of chunk Explore_stocks_Cw

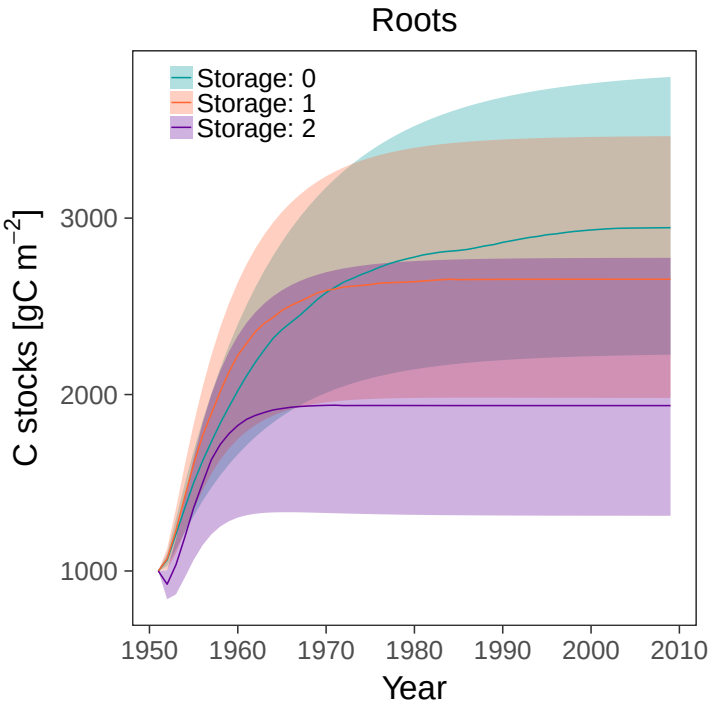


Figure 35: plot of chunk Explore_stocks_Cr

```
Cs_b$name <- "Storage: 1"
Cs1_c$name <- "Storage: 2 fast"
Cs2_c$name <- "Storage: 2 slow"
DF <- do.call(rbind, list(Cs_b, Cs1_c, Cs2_c))
ggplot(DF, aes(x= tm1, ymin= min, ymax= max, y=median)) +
  ggtitle("Storage") +
  geom_ribbon(aes(fill=name), alpha=0.3) +
  geom_line(aes(colour=name)) +
  theme_bw(23) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
       legend.position = c(1,0.5), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        plot.title = element_text(size = rel(1), hjust = 0.5, colour = "black"),
        legend.title = element_blank()) +
  xlim(1951, 2009) +
  xlab("Year") +
  ylab(bquote('C stocks ['*gC~m^-2*']')) +
  scale_color_manual(values=c("#009999","#FF6633","#660099")) +
  scale_fill_manual(values=c("#009999","#FF6633","#660099"))
```

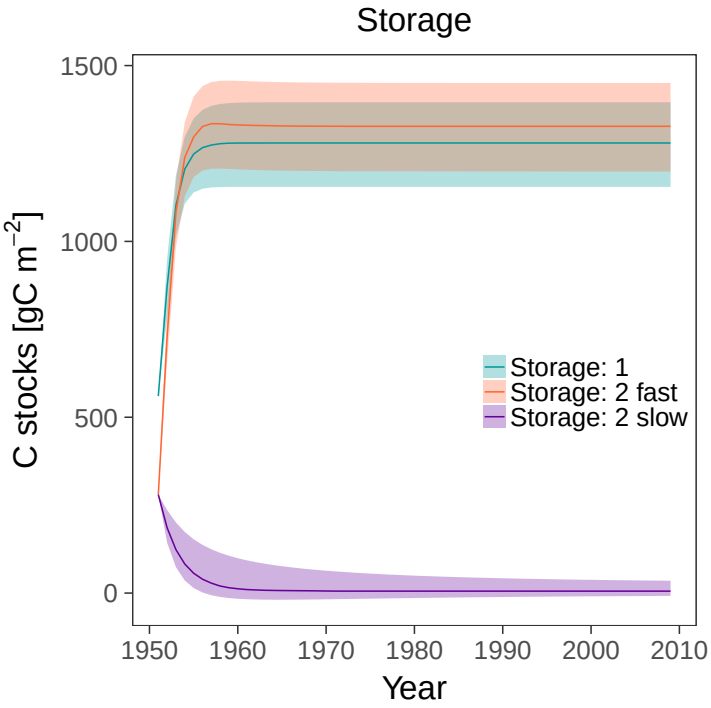


Figure 36: plot of chunk Explore_stocks_Cs

6. The C released from each compartment

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
s0p <- data.frame("zT"=rep(zTA$Cp,weightsA), "pool"="Photoassimilates", "name"="Storage: 0")
s0f <- data.frame("zT"=rep(zTA$Cf,weightsA), "pool"="Str. Foliage", "name"="Storage: 0")
s0w <- data.frame("zT"=rep(zTA$Cw,weightsA), "pool"="Wood", "name"="Storage: 0")
s0r <- data.frame("zT"=rep(zTA$Cr,weightsA), "pool"="Roots", "name"="Storage: 0")
s1p <- data.frame("zT"=rep(zTB$Cp,weightsB), "pool"="Photoassimilates", "name"="Storage: 1")
s1f <- data.frame("zT"=rep(zTB$Cf,weightsB), "pool"="Str. Foliage", "name"="Storage: 1")
s1w <- data.frame("zT"=rep(zTB$Cw,weightsB), "pool"="Wood", "name"="Storage: 1")
s1r <- data.frame("zT"=rep(zTB$Cr,weightsB), "pool"="Roots", "name"="Storage: 1")
s2p <- data.frame("zT"=rep(zTC$Cp,weightsC), "pool"="Photoassimilates", "name"="Storage: 2")
s2f <- data.frame("zT"=rep(zTC$Cf,weightsC), "pool"="Str. Foliage", "name"="Storage: 2")
s2w <- data.frame("zT"=rep(zTC$Cw,weightsC), "pool"="Wood", "name"="Storage: 2")
s2r <- data.frame("zT"=rep(zTC$Cr,weightsC), "pool"="Roots", "name"="Storage: 2")
DF_r <- do.call(rbind, list(s0p,s0f,s0w,s0r,s1p,s1f,s1w,s1r,s2p,s2f,s2w,s2r))
```

```
ggplot(DF_r, aes(x = factor(pool), y = zT, fill = name)) +
  geom_boxplot(alpha=0.7) +
  scale_y_continuous(name = bquote('Released C ['*gC~m^-2*']')) +
  scale_x_discrete(name = "Compartment") +
  theme_bw(19) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = c(1,1), legend.justification = c(1,1),
        legend.background = element_rect(fill = 'transparent'),
        legend.title = element_blank()) +
  guides(color=guide_legend(
    keywidth=0.5,
    keyheight=0.5,
    default.unit="inch")) +
  scale_fill_brewer(palette = "Accent")
```

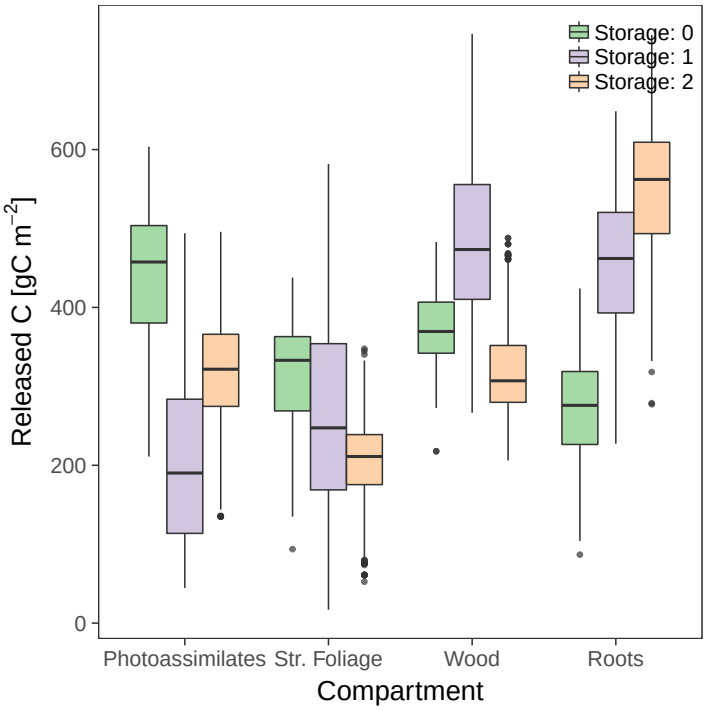


Figure 37: plot of chunk Explore_C_release_SS

7. And the mean ages and transit times of each of the 3 systems

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
# Mean ages
rep_ages_A <- rep(SysAgesA,weightsA)
#boxplot(rep_ages_A, ylab="Mean age Storage: 0", outline=FALSE)
hist(rep_ages_A, xlab="Mean age Storage: 0", cex.axis = 1.5, cex.lab = 1.6)

rep_ages_B <- rep(SysAgesB,weightsB)
#boxplot(rep_ages_B, ylab="Mean age Storage: 1", outline=FALSE)
hist(rep_ages_B, xlab="Mean age Storage: 1", cex.axis = 1.5, cex.lab = 1.6)

rep_ages_C <- rep(SysAgesC,weightsC)
#boxplot(rep_ages_C, ylab="Mean age Storage: 2", outline=FALSE)
hist(rep_ages_C, xlab="Mean age Storage: 2", cex.axis = 1.5, cex.lab = 1.6)

boxplot(rep_ages_A, rep_ages_B, rep_ages_C, ylab="Mean age",
       names=c("Storage: 0","Storage: 1","Storage: 2"),
       outline=FALSE, cex.axis = 1.5, cex.lab = 1.6)
```

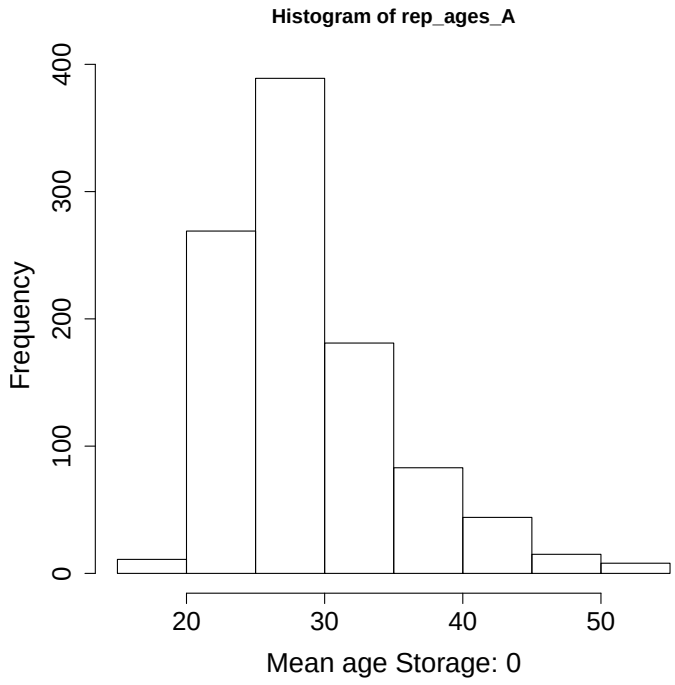


Figure 38: plot of chunk Explore_Age_TTime

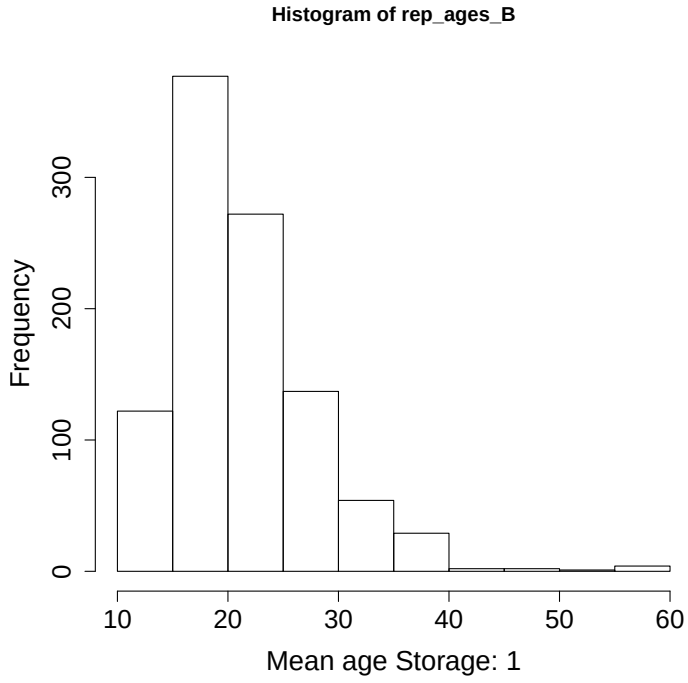


Figure 39: plot of chunk Explore_Age_TTime

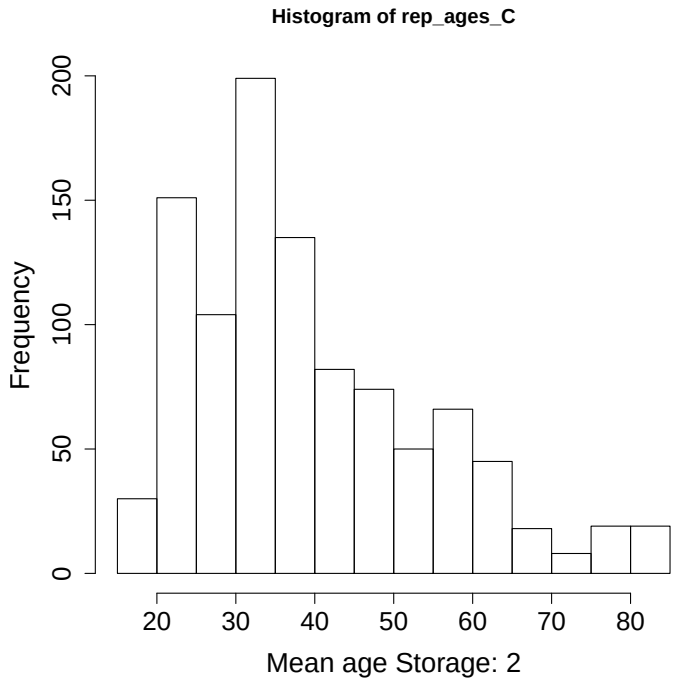


Figure 40: plot of chunk Explore_Age_TTime

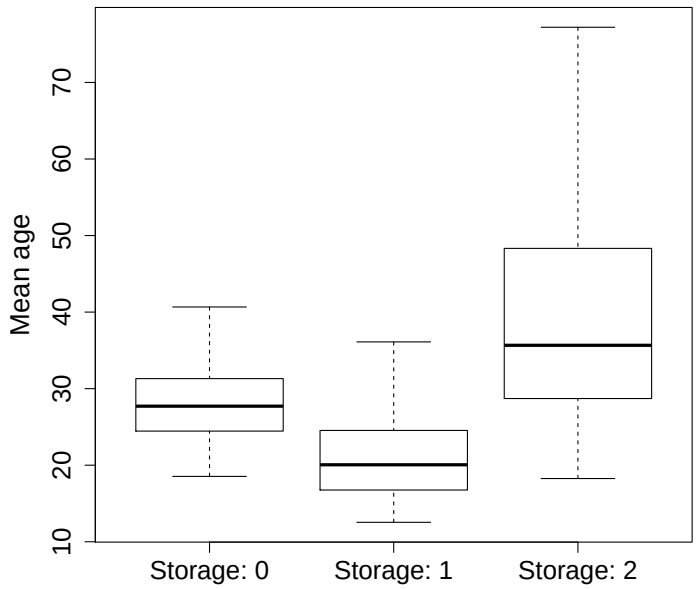


Figure 41: plot of chunk Explore_Age_TTime

```
# Mean transit times

rep_transit_A <- rep(TransitA,weightsA)
#boxplot(rep_transit_A, ylab="Mean transit time Storage: 0", outline=FALSE) #without outliers
hist(rep_transit_A, xlab="Transit time Storage: 0", cex.axis = 1.5, cex.lab = 1.6)
```

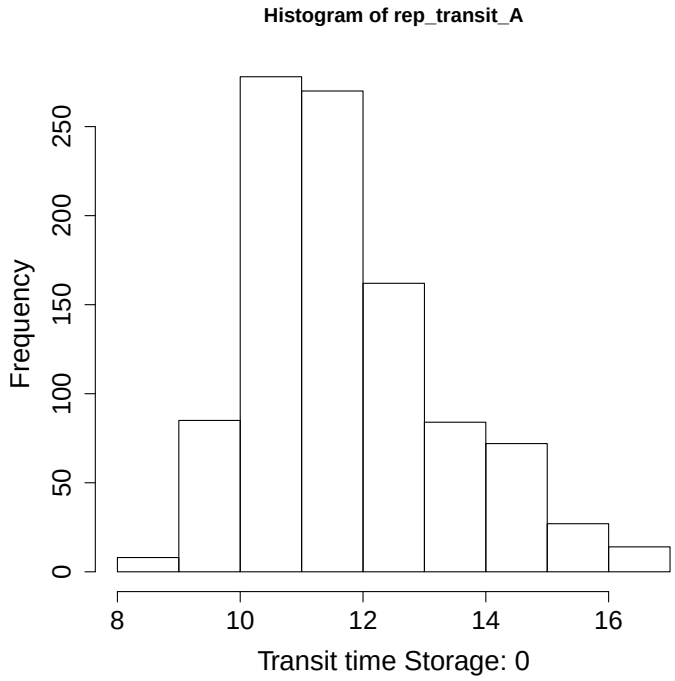


Figure 42: plot of chunk Explore_Age_TTime

```
rep_transit_B <- rep(TransitB,weightsB)
#boxplot(rep_transit_B, ylab="Mean transit time Storage: 1", outline=FALSE)
hist(rep_transit_B, xlab="Transit time Storage: 1", cex.axis = 1.5, cex.lab = 1.6)

rep_transit_C <- rep(TransitC,weightsC)
#boxplot(rep_transit_C, ylab="Mean transit time Storage: 2", outline=FALSE)
hist(rep_transit_C, xlab="Transit time Storage: 2", cex.axis = 1.5, cex.lab = 1.6)

boxplot(rep_transit_A, rep_transit_B, rep_transit_C, ylab="Mean transit time",
       names=c("Storage: 0","Storage: 1","Storage: 2"),
       outline=FALSE, cex.axis = 1.5, cex.lab = 1.6)
```

What is the relation between mean ages and mean transit times?

```
knitr::opts_chunk$set(dev = 'pdf', dpi = 300)
# Exploratory analysis using a scatter plot of the data in log scale:
Ln_a <- data.frame("LnMA" = log(SysAgesA), "LnMTT" = log(TransitA))
Ln_a$name <- "Storage: 0"
Ln_b <- data.frame("LnMA" = log(SysAgesB), "LnMTT" = log(TransitB))
Ln_b$name <- "Storage: 1"
Ln_c <- data.frame("LnMA" = log(SysAgesC), "LnMTT" = log(TransitC))
Ln_c$name <- "Storage: 2"
df_ln <- do.call(rbind, list(Ln_a,Ln_b,Ln_c))

ggplot(df_ln, aes(x=LnMA, y=LnMTT))+
  geom_point(size=1, aes(colour=name)) +
  geom_abline(intercept=0, lty = 2) +
  theme_bw(23) +
```

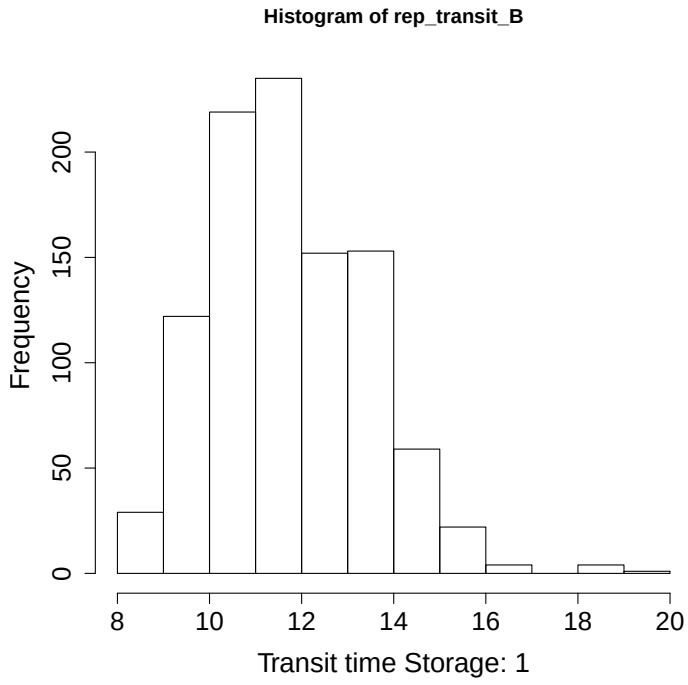


Figure 43: plot of chunk Explore_Age_TTime

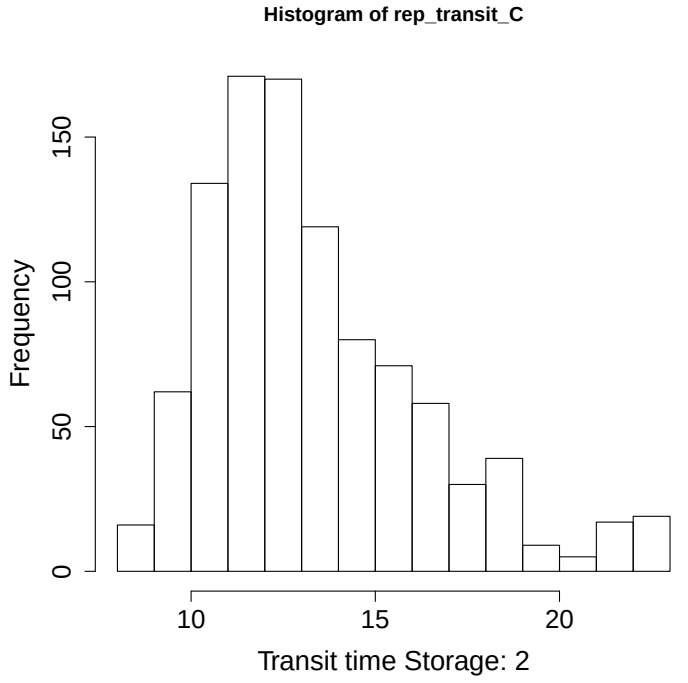


Figure 44: plot of chunk Explore_Age_TTime

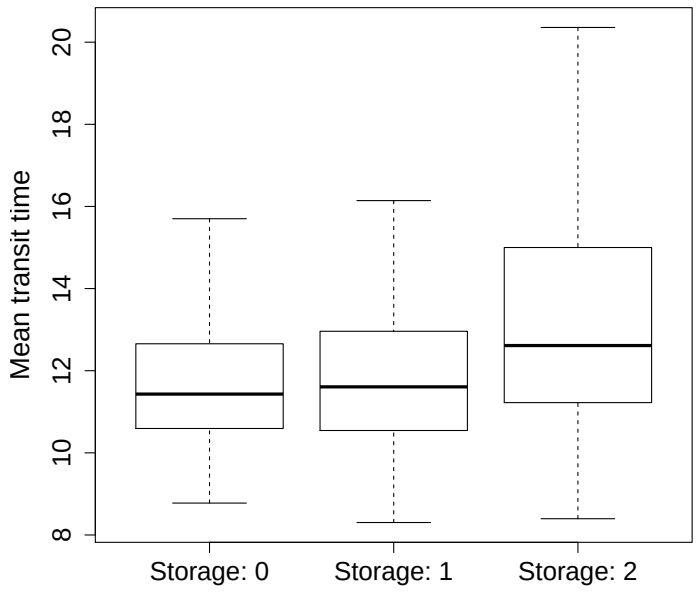


Figure 45: plot of chunk Explore_Age_TTime

```
theme(panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      legend.position = c(1,0.5), legend.justification = c(1,1),
      legend.background = element_rect(fill = 'transparent'),
      legend.title = element_blank()) +
xlab("Ln Mean Age") +
ylab("Ln Mean Transit time") +
scale_color_manual(values=c("#009999","#FF6633","#660099")) +
guides(color=guide_legend(
  keywidth=0.3,
  keyheight=0.3,
  default.unit="inch"))
```

Summary

This workflow shows how to implement three models with different CA schemes and run them in order to obtain important model diagnostics. It was possible to fit the models to the available data, but it is to be taken into account that the collinearity was very high. This problem may be solved by restructuring the models in order to eliminate parameter redundancies or compensations, or by constraining the parameter estimations with more data on C stocks and ages. The difference in the model structures had a lesser impact in some properties, but overall it resulted in different predictions of age and transit time distributions.

References

Bartlett, Megan K., Scott V. Ollinger, David Y. Hollinger, Haley F. Wicklein, and Andrew D. Richardson. 2011. "Canopy-Scale Relationships Between Foliar Nitrogen and Albedo Are Not Observed in Leaf Reflectance and Transmittance Within Temperate Deciduous Tree Species." *Botany* 89 (7): 491–97. doi:10.1139/b11-037.

Canadell, Josep G., Corinne Le Quéré, Michael R. Raupach, Christopher B. Field, Erik T. Buitenhuis, Philippe Ciais, Thomas J. Conway, Nathan P. Gillett, R. A. Houghton, and Gregg Marland. 2007. "Contributions to Accelerating Atmospheric CO_2 Growth from Economic Activity, Carbon Intensity, and Efficiency of Natural Sinks." *Proceedings of the National Academy of Sciences* 104 (47): 18866–70.

Fox, Andrew, Mathew Williams, Andrew D. Richardson, David Cameron, Jeffrey H. Gove, Tristan Quaife, Daniel Ricciuto, et al. 2009. "The REFLEX Project: Comparing Different Algorithms and Implementations for the Inversion of a Terrestrial Ecosystem Model Against

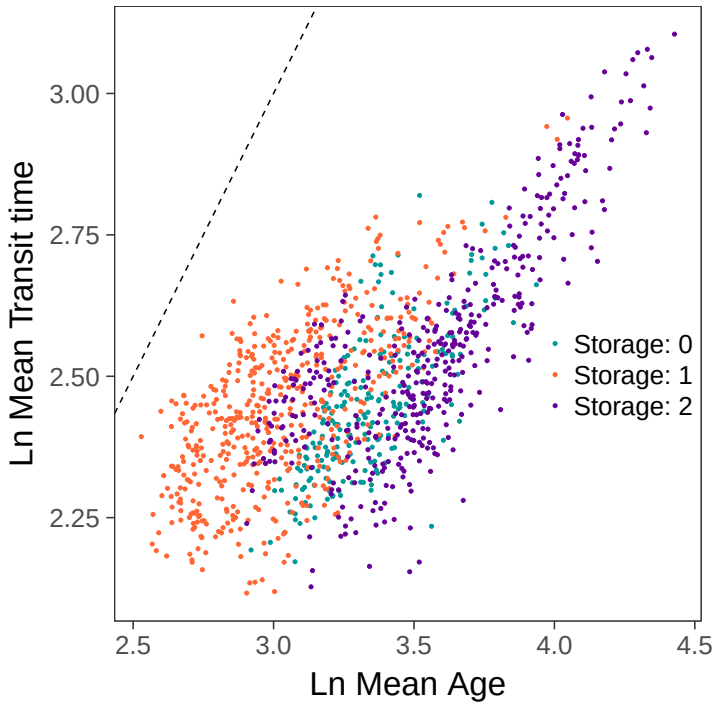


Figure 46: plot of chunk Linear_corr_Age_TT_0

Eddy Covariance Data." *Agricultural and Forest Meteorology* 149 (10): 1597–1615. doi:http://dx.doi.org/10.1016/j.agrformet.2009.05.002.

Lacointe, André. 2000. "Carbon Allocation Among Tree Organs: A Review of Basic Processes and Representation in Functional-Structural Tree Models." *Annals of Forest Science* 57 (5-6): 521–33. doi:10.1051/forest:2000139.

Luo, Yiqi, Luther W. White, Josep G. Canadell, Evan H. DeLucia, David S. Ellsworth, Adrien Finzi, John Lichter, and William H. Schlesinger. 2003. "Sustainability of Terrestrial Carbon Sequestration: A Case Study in Duke Forest with Inversion Approach." *Global Biogeochemical Cycles* 17 (1): n/a–n/a. doi:10.1029/2002GB001923.

Richardson, Andrew D., Mariah S. Carbone, Trevor F. Keenan, Claudia I. Czimczik, David Y. Hollinger, Paula Murakami, Paul G. Schaberg, and Xiaomei Xu. 2013. "Seasonal Dynamics and Age of Stemwood Nonstructural Carbohydrates in Temperate Forest Trees." *New Phytologist* 197 (3): 850–61. doi:10.1111/nph.12042.

Schiestl-Aalto, Pauliina, Liisa Kulmala, Harri Mäkinen, Eero Nikinmaa, and Annikki Mäkelä. 2015. "CASSIA – a Dynamic Model for Predicting Intra-Annual Sink Demand and Interannual Growth Variation in Scots Pine." *New Phytologist* 206 (2): 647–59. doi:10.1111/nph.13275.

Sitch, Stephen, B. Smith, I. C. Prentice, A. Arneth, A. Bondeau, W. Cramer, J. O. Kaplan, et al. 2003. "Evaluation of Ecosystem Dynamics, Plant Geography and Terrestrial Carbon Cycling in the LPJ Dynamic Global Vegetation Model." *Global Change Biology* 9 (2): 161–85. doi:10.1046/j.1365-2486.2003.00569.x.