

Fig_all_NC

2022-11-24

```
# script that creates all Figures for the publication
# Wood structure explained by complex spatial source-sink interactions
# Friend, Eckes-Shephard and Tupker 2022,

#Script - Authors: Andrew Friend, Annemarie Eckes-Shephard

#For guidance, Nature's standard figure sizes are 90 mm (single column) and
#180 mm (double column) and the full depth of the page is 170 mm.
res= 1000 #changed for high-quality images after acceptance
cex.legend = 0.5
cex.lab = 0.7
file_path <- ""
obs_path <- paste0(file_path,"CF_FIGS_final/Observations/")
#specs similar across plots:
runs_col <- safe_colorblind_palette <- c("#E6AB02", "#1B9E77", "#D95F02",
                                           "#7570B3", "#666666", "#E7298A",
                                           "#6A61E", "#E6AB02", "#D55E00",
                                           "#56B4E9")

#
ttl = 50.1
tal = 2680.
#####
#Functions used in data aggregation and plotting

#####
#bin_and_running_mean
#input: model_config_output numeric vector,
#input: bin_size numeric, desired size used for binning across the radial file
#input: maxlenlength_radial_file numeric, maximal radial file length that has come
# out of the simulations, it is used together with the bin size to derive the
# total number of bins for the aggregation.
#input: varname, string, whether binning is done over cell mass density ("den")
# or cell lengths ("L")
#output: a list of three numeric vectors with either cell length ('L') or
# cell mass density ('den'), together with cell distance from the cambium ('Di')
# and number of datapoints in the bin ('n')
bin_and_running_mean <- function(model_config_output, bin_size,
                                maxlenlength_radial_file, varname){
  # author of original code: Andrew F, translated to R, Annemarie ES
#binning:
  bs = bin_size
  mlen = maxlenlength_radial_file
```

```

den_a = vector()
pos_a = vector()
ns_a   = vector()
a = .0
b = a + bs
while (b < mlen+bs){
  n = length(t(model_config_output[varname]))
  i = 1
  j = .0
  dena_sum = .0
  while (i <=n){
    d = model_config_output$Di[i]
    # print(i)
    # print(d)
    if (d >= a & d < b){
      dena_sum = dena_sum + model_config_output[i,varname]
      j = j + 1.
    }
    i = i + 1
  }
  if (j > 0){
    den_a <- append(den_a,dena_sum/j)
    ns_a   <- append(ns_a,j) # record how many data points were actually
    #used to derive the number in this bin
    pos_a <- append(pos_a,a+(b-a)/2.)
  }

  a = a + bs
  b = b + bs
}

n = length(den_a)
run = 3
den_ar = vector()
den_ar = append(den_ar,den_a[1])
# running means across bins:
for (i in 2:(n-1)){
  sden = .0
  for (j in (i-1):(i+1)){
    sden = sden + den_a [j]
  }
  den_ar = append(den_ar,(sden/run))
}
den_ar = append(den_ar,den_a[n])

#prep output:
if(varname=="L"){
  out <- list(den_ar,pos_a,ns_a)
  names(out)[1:3] <- c("L","Di","n")
  return(out)
}
if(varname=="den"){
  out <- list(den_ar,pos_a,ns_a)

```

```

names(out)[1:3] <- c("den","Di","n")
return(out)
}

}

#####makeTransparent
#To show where overlap between predicted cells is greatest
#note: always pass alpha on the 0-255 scale
makeTransparent<-function(someColor, alpha=100){
  newColor<-col2rgb(someColor)
  apply(newColor, 2, function(curcoldata){rgb(red=curcoldata[1],
                                              green=curcoldata[2],
                                              blue=curcoldata[3],
                                              alpha=alpha, maxColorValue=255)})})
}

##### add_legend
add_legend <- function(...) {
  opar <- par(fig=c(0, 1, 0, 1), oma=c(0, 0, 0, 0),
             mar=c(0, 0, 0, 0), new=TRUE)
  on.exit(par(opar))
  plot(0, 0, type='n', bty='n', xaxt='n', yaxt='n')
  legend(...)
}

#####add_cellNumberColumn
# quick and dirty addition of column to highlight each radial file for later
# year-specific binning and plotting
# input: model_output RINGS_v2 model output from fort.97 table read in.
# output model output, with additional column
add_cellNumberColumn <- function(model_output){
  pos=1
  rfile = 1
  model_output$celln[1] <- pos
  model_output$row[1] <- rfile
  for(i in 2:dim(model_output)[1]){
    pos=pos+1
    if(model_output$Di[i-1] < model_output$Di[i]){
      model_output$celln[i] <- pos
      model_output$row[i] <- rfile
    }else{
      pos = 1
      rfile = rfile+1
      model_output$celln[i] <- pos
      model_output$row[i] <- rfile
    }
  }
  return(model_output)
}

#####FIGURE1:#####
#FIGURE1:
# we normalis every simulated set of radial files for each model configuration run

```

```

bs = 0.02 # bin size (frac)
mlen = 1.0 # length to process (frac)

#load data for each model configuration run:

full_model <- read.table(paste0(file_path,'fort.97_20'))
names(full_model) <- c("Di", "L ", "M")
# add another cellnumber column, for analysis with tgram
full_model <- add_cellNumberColumn(full_model)

fixed_temp <- read.table(paste0(file_path,'fort.97_20_nT') )
names(fixed_temp) <- c("Di", "L ", "M")
# add another cellnumber column, for analysis with tgram
fixed_temp <- add_cellNumberColumn(fixed_temp)

fixed_zonewidth <- read.table(paste0(file_path,'fort.97_20_fzw'))
names(fixed_zonewidth)<- c("Di", "L ", "M")
# add another cellnumber column, for analysis with tgram
fixed_zonewidth <- add_cellNumberColumn(fixed_zonewidth)

fixed_zonewidth_nodormancy <- read.table(paste0(file_path,'fort.97_20_fzw_nd'))
names(fixed_zonewidth_nodormancy) <- c("Di", "L ", "M")
# add another cellnumber column, for analysis with tgram
fixed_zonewidth_nodormancy <- add_cellNumberColumn(fixed_zonewidth_nodormancy)

sugar_saturation <- read.table(paste0(file_path,'fort.97_20_sS'))
names(sugar_saturation) <- c("Di", "L ", "M")
# add another cellnumber column, for analysis with tgram
sugar_saturation <- add_cellNumberColumn(sugar_saturation)

# some data manipulation and aggregation:

#turn radial files around
full_model$Di <- 1- full_model$Di
fixed_temp$Di <- 1- fixed_temp$Di
fixed_zonewidth$Di <- 1- fixed_zonewidth$Di
fixed_zonewidth_nodormancy$Di <- 1- fixed_zonewidth_nodormancy$Di
sugar_saturation$Di <- 1- sugar_saturation$Di

# list model output objects for easy looping access:
runs <- list(full_model,fixed_temp,fixed_zonewidth,fixed_zonewidth_nodormancy,
            sugar_saturation)
names(runs) <- c("full_model","fixed_temp","fixed_zonewidth",
               "fixed_zonewidth_nodormancy","sugar_saturation")

# loop through all model configurations and:
# add new variable-columns, that will hold derived variables Vc and den,
from output variables:
# create binned tracheidograms for plotting

#initialise list of lists for data storage and easy retrieval in loops for plotting:

```

```

collect_stdTrach_single <- list(list(),list(),list(),list(),list())
names(collect_stdTrach_single) <- c("full_model","fixed_temp","fixed_zonewidth",
                                   "fixed_zonewidth_nodormancy",
                                   "sugar_saturation")

for (e in names(runs)){
  # add new vars:
  print(e)
  runs[[e]]$Vc <- runs[[e]]$L * ttl * tal / 1.E12 # ml      eq 11 in manuscript
  runs[[e]]$den <- 1.E-6 * runs[[e]]$M / runs[[e]]$Vc # g/cm3
  # create binned trachs:
  collect_stdTrach_single[[e]] <- bin_and_running_mean(model_config_output = runs[[e]],
                                                         bin_size = bs,
                                                         maxlength_radial_file = mlen,
                                                         varname="den")
}

# plot:

jpeg(filename='Figures_update/Fig1.jpeg',units="mm",width=90, height=60, res=res)

par(mfrow=c(2,1),mar=c(0,2,0,0),oma=c(1,0,0.02,4),cex.axis=0.7, cex.lab=0.7,
    cex.main=1, cex.sub=1,ps=8,xpd=TRUE)

#Fig 1a)
e="full_model"

# start with all simulated density values that were simulated across all 20 years:
plot(runs[[e]]$Di,runs[[e]]$den,pch=16,cex=0.12,ylab="",xlab="",yaxt="n",
     col=makeTransparent('grey',alpha=50),
     ylim=c(0,1.2),xaxt="n")

# add all the tracheidograms across all 20 years to plot
# grey lines are the (binned) dens - tracheidograms of the different years at that site.
# 100 radial files were modelled for each year between the simulation years 1976 to 1995.
# (note "row" is equivalent to an individually simulated radial file)
for(row in 1:20){
  n=100
  tmp <- bin_and_running_mean(model_config_output = runs[[e]][which(runs[[e]]$row > row*n-99 & runs[[e]]$row <= row*n)],
                             bin_size = bs,maxlength_radial_file = mlen,varname="den")
  lines(tmp$Di, tmp$den,type="l",col="black",lwd=0.2)
}

# add binned tracheidogram derived from all years' simulations:
lines(collect_stdTrach_single[[e]]$Di,collect_stdTrach_single[[e]]$den,lwd=1)

# add observations:
# Source:
# Cuny, H. E., Rathgeber, C. B. K., Frank, D., Fonti, P. & Fournier, M.
# Kinetics of tracheid development explain conifer tree-ring structure.
# New Phytol. 203, 1231-1241, DOI: 10.1111/nph. 12871 (2014).
e3 = read.table(file = paste0(obs_path,"cuny_3e.dat"),col.names = c("d_3e", "den_3e"),sep = ",")

```

```

f3 = read.table(file = paste0(obs_path,"cuny_3f.dat"),col.names = c("d_3f", "den_3f"),sep = ",")

#unit conversion to g cm-3
e3$d_3e = e3$d_3e / 1.0E2
f3$d_3f = f3$d_3f / 1.0E2
e3$den_3e = e3$den_3e / 1.0E3
f3$den_3f = f3$den_3f / 1.0E3

#Source: Southern finland (SFinP) and Northern Finland (NFinP):
# Decoux, V., Varcin, E. & Leban, J.-M. Relationships between the intra-ring wood
# density assessed by X-ray densitometry and optical anatomical measurements in conifers.
# Consequences for the cell wall apparent density determination.
# Annals For. Sci. 61, 251-262, DOI: 10.1051/forest:2004018 (2004).

NFinP= c(433,371,352,361,362,354,366,376,375,378,396,425,479,530,628,746,826,
921,959,875)
SFinP= c(508,374,355,346,348,360,383,398,406,427,453,530,706,817,845,859,975,
997,998,862)
NfP = c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20)
x = NfP / 20. - .5 / 20.
#unit conversion to g cm-3
yN = NFinP / 1000.
yS = SFinP / 1000.

#France:
lines(e3$d_3e ,e3$den_3e, col='black', lty=2 ,lwd=1)
lines(f3$d_3f ,f3$den_3f, col='black', lty=3 ,lwd=1)

#Finland:
lines(x,yN, col='black', lty=4 ,lwd=1)
lines(x,yS, col='black', lty=5 ,lwd=1)

## layout and legend:
axis(side = 2,lwd = 0, line = -0.8, las = 2)
axis(side = 2, tck = -.015, labels = NA)
axis(side = 1, tck = .015, labels = NA)
#mtext(side = 3 , "a"),line=-1, adj=0.05 ,outer = FALSE)
mtext(side=1,'a',line=-5.7, adj=0.015,col='black',cex=0.7)

# Fig 1 b)
# start with all simulated density values that were simulated across all 20 years
for the full model:
e="full_model"
plot(runs[[e]]$Di,runs[[e]]$den,pch=16,cex=0.1,ylab="",xlab="",yaxt="n",
col=makeTransparent('grey',alpha=50),ylim=c(0,1.2),xaxt="n")

# add all the binned (annual) dens - tracheidograms to plot
# grey lines are the (binned) dens - tracheidograms of the different years at that site.
for(row in 1:20){
  n=100
  tmp <- bin_and_running_mean(model_config_output = runs[[e]][which(runs[[e]]$row > row*n-99 & runs[[e]]$row < row*n)],
  bin_size = bs,maxlength_radial_file = mlen,

```

```

                                varname="den")
  lines(tmp$Di, tmp$den,type="l",col="black",lwd=0.2)
}

#add other model configurations- but only their binned tracheidogram derived
#from binning across all 20 years:
lines(collect_stdTrach_single[["fixed_temp"]]$Di,
      collect_stdTrach_single[["fixed_temp"]]$den,type="l",col=runs_col[1],
      lwd=1)
points(collect_stdTrach_single[["fixed_zonewidth"]]$Di,
       collect_stdTrach_single[["fixed_zonewidth"]]$den,type="l",col=runs_col[2],
       lwd=1)
points(collect_stdTrach_single[["fixed_zonewidth_nodormancy"]]$Di,
       collect_stdTrach_single[["fixed_zonewidth_nodormancy"]]$den,type="l",
       col=runs_col[3], lwd=1)
points(collect_stdTrach_single[["sugar_saturation"]]$Di,
       collect_stdTrach_single[["sugar_saturation"]]$den,pch=2,type="l",
       col=runs_col[4], lwd=1)

# add full_model binned tracheidogram derived from all years' binned tracheidograms:
# binned tracheidogram derived from all years' simulations:
lines(collect_stdTrach_single[[e]]$Di,collect_stdTrach_single[[e]]$den,lwd=1.4)

axis(side = 2,lwd = 0, line = -0.8, las = 2)
axis(side = 2, tck = -.015, labels = NA)
axis(side = 1, tck = -.015, labels = NA)
axis(side = 1,lwd = 0, line = -1.3,cex=0.7)
mtext(side = 2 , expression(Density ~ (g ~ cm-3)),line=-1 ,outer = TRUE,
      cex = cex.lab)

mtext(outer=FALSE,side=1,expression(Distance~from~cambium~(fraction~of~ring)),
      line=0.2,cex = cex.lab)

#mtext(side = 3 , "b)",line=-1, adj=0.05 ,outer = FALSE,cex=0.7)
mtext(side=1,'b)',line=-5.7, adj=0.015,col='black',cex = cex.lab)

# Add legend between a&b, outside plot region
add_legend(0.62,0.25,xpd=TRUE, legend = c("density/cell","tracheidograms"),
          title = "Model output",
          pch = c(16,NA), lty = c(NA,1), lwd= c(NA,1.5), col='grey',box.lwd = 0,
          cex = cex.legend, bg = "transparent")

# Add legend to the side of Fig 1 b), outside plot region
add_legend(0.58,-0.25, xpd=TRUE, legend = c("full model","fixed temp","fixed zw",
                                             "fixed zw, no dorm. ", "sugar saturation"),
          cex = cex.legend,
          lty = c(1,1,1,1,1), col = c(1,runs_col[1:4]), lwd = 1.5,
          box.lwd = 0, title = "Model configurations", bg = "transparent")

# Add legend to the side of fig 1 a), outside plot region
add_legend(0.58,0.95, xpd=TRUE,

```

```

        legend = c("full model","Obs1 (France)","Obs2 (France)",
                    "Obs3 (Finland)","Obs4 (Finland)"),
        lty = c(1,2,3,4,5), box.lwd = 0, cex = cex.legend , title = "Model vs. observation", bg = "t

dev.off()
#####

#####APPENDIX FIGURE 1#####
#APPENDIX FIGURE 1 - plots all binned dens - radial files and the simulated densities
# from Figure 1b) from the model configurations that were not included in the graph
# plot all binned tracheidograms for the appendix, this time with a graph each
# year containing all model output

pdf(file='Figures_update/appdx_1b.pdf',width=10,height =6)

#par(mfrow=c(5,4),mar=c(0,0,0,0),oma=c(4,4,0.5,0.5))
par(mfrow=c(4,6),mar=c(0,0,0,0),oma=c(4,4,0.5,0.5))

years <- seq(1976,1995,1)

for(row in 1:20){
  c=1
  for (e in names(runs)){
    #print(e)
    if(e == "full_model"){
      tmp_save <- bin_and_running_mean(model_config_output = runs[[e]][which(runs[[e]]$row > row*n-99 &

      plot(runs[[e]][which(runs[[e]]$row > row*n-99 & runs[[e]]$row <= row*n),]$Di,runs[[e]][which(runs
        ylim=c(0,1.2),xlim=c(0,1),axes = FALSE)
      lines(tmp_save$Di, tmp_save$den,type="l",col="black",lwd=1.2)

    }else{
      tmp <- bin_and_running_mean(model_config_output = runs[[e]][which(runs[[e]]$row > row*n-99 & runs[[

      points(runs[[e]][which(runs[[e]]$row > row*n-99 & runs[[e]]$row <= row*n),]$Di,runs[[e]][which(runs
        ylim=c(0,1.2),xlim=c(0,1),axes = FALSE)
      lines(tmp$Di, tmp$den,type="l",col=runs_col[c],lwd=1.2)
      c = c + 1 # switch to next colour (= model assumption)
    }
    # redraw full_model line again, as it is covered quite a bit:
    lines(tmp_save$Di, tmp_save$den,type="l",col="black",lwd=1.2)

  }
  #axes on the outer edges only

  #if(row==1|| row==5 || row==9 ||row==13||row==17){
if(row==1|row==7|row==13|row==19){
  axis(side=2,las=2)
}
}

```

```

    #if(row>=17){
    if(row>=15){
      axis(side=1,las=2)
    }

    box()
    par(new=TRUE)
    mtext(paste(years[row]),side=1,line=-1.7,adj=0.4)

  }
  # add legend
  plot.new()
  legend("center",c("Single cells","binned tracheidograms:"),
        lty=c(NA,1),pch=c(16,16),pt.cex=c(2,0),lwd=c(0,1),col=c("grey","black"),
        text.font=c(1,0.5),bty="n")
  legend("bottomleft",legend = names(runs), col = c(1,runs_col),
        lty=rep(1,length(runs_col)))

  mtext(outer=TRUE,side=1,"Cell position (#)",line=2.4)
  mtext(outer=TRUE,side=2,expression(Density ~ (g ~ cm^-3)),line=2.1)

  dev.off()
#####

#####FIGURE2:#####

#load data for each model configuration run:

full_model <- read.table(paste0(file_path,'fort.94.def'))
names(full_model) <- c("Di", "L", "M")
#full_model <- add_cellNumberColumn(full_model)

full_temp_incr<- read.table(paste0(file_path,'fort.94.two'))
names(full_temp_incr) <- c("Di", "L", "M")
#full_temp_incr <- add_cellNumberColumn(full_temp_incr)

THKOnly_temp_incr <- read.table(paste0(file_path,'fort.94.twob'))
names(THKOnly_temp_incr) <- c("Di", "L", "M")
#THKOnly_temp_incr <- add_cellNumberColumn(THKOnly_temp_incr)

SucSat_model <- read.table(paste0(file_path,'fort.94.defC'))
names(SucSat_model) <- c("Di", "L", "M")
#SucSat_model <- add_cellNumberColumn(SucSat_model)

SucSat_full_temp_incr<- read.table(paste0(file_path,'fort.94.twoC'))
names(SucSat_full_temp_incr) <- c("Di", "L", "M")
#SucSat_full_temp_incr <- add_cellNumberColumn(SucSat_full_temp_incr)

```

```

# some data manipulation and aggregation:

bs = 40. # bin size (um)
mlen = 1850. # length to process (um)
maxlen=mlen
runs <- list(full_model,full_temp_incr,THKOnly_temp_incr,SucSat_full_temp_incr,
             SucSat_model)
names(runs) <- c("full_model","full_temp_incr","THKOnly_temp_incr",
                "SucSat_full_temp_incr","SucSat_model")

# loop through all model configurations and:
# add new variable-columns, that will hold derived variables Vc and den, from output variables:
# create binned tracheidograms for plotting
#initialise list of lists for data storage and easy retrieval in loops for plotting:
collect_stdTrach_single <- list(list(),list(),list(),list(),list())
names(collect_stdTrach_single) <- c("full_model","full_temp_incr","THKOnly_temp_incr","SucSat_full_temp_incr")

for (e in names(runs)){
  # add new vars:
  print(e)
  runs[[e]]$Vc <- runs[[e]]$L * ttl * tal / 1.E12 # ml      eq 11 in manuscript
  runs[[e]]$den <- 1.E-6 * runs[[e]]$M / runs[[e]]$Vc # g/cm3
  # create binned trachs:
  collect_stdTrach_single[[e]] <- bin_and_running_mean(model_config_output = runs[[e]],
                                                         bin_size = bs,
                                                         maxlength_radial_file = mlen,
                                                         varname="den")
}

#plot:

jpeg(filename='Figures_update/Fig2.jpeg',units="mm",width=90, height=60, res=res)

par(mfrow=c(2,1),mar=c(0,2,0,0),oma=c(1,0,0.02,4),cex.axis=0.7, cex.lab=0.7, cex.main=1, cex.sub=1,ps=8)

# start with all simulated density positions:
plot(runs[["full_model"]]$Di,runs[["full_model"]]$den,
     pch=16,cex=0.5,col=makeTransparent('grey',alpha=50),
     xlim=c(maxlen,0),ylim=c(0,1.5),ylab='',xlab='',xaxt='n',yaxt='n')
points(runs[["full_temp_incr"]]$Di,runs[["full_temp_incr"]]$den,
       pch=16,cex=0.5,col=makeTransparent(runs_col[7],alpha=50))
points(runs[["THKOnly_temp_incr"]]$Di,runs[["THKOnly_temp_incr"]]$den,
       pch=16,cex=0.5,col=makeTransparent(runs_col[6],alpha=50))
points(runs[["SucSat_full_temp_incr"]]$Di,runs[["SucSat_full_temp_incr"]]$den,
       pch=16,cex=0.5,col=makeTransparent(runs_col[8],alpha=50))

#add binned tracheidograms
lines( collect_stdTrach_single[["full_model"]]$Di,
       collect_stdTrach_single[["full_model"]]$den,type="l",col="black", lwd=2)
points( collect_stdTrach_single[["full_temp_incr"]]$Di,
       collect_stdTrach_single[["full_temp_incr"]]$den,type="l",col=runs_col[7],
       lwd=2)
points( collect_stdTrach_single[["THKOnly_temp_incr"]]$Di,

```

```

collect_stdTrach_single[["THKOnly_temp_incr"]]]$den,type="l",col=runs_col[6],
lwd=2)
points( collect_stdTrach_single[["SucSat_full_temp_incr"]]]$Di,
collect_stdTrach_single[["SucSat_full_temp_incr"]]]$den,pch=2,type="l",
col=runs_col[8], lwd=2)

axis(side = 2,lwd = 0, line = -0.8, las = 2)
axis(side = 2, tck = -.015, labels = NA)
axis(1, at = seq(0,maxlen,by=200), labels = NA,tck= .015)

mtext(outer=FALSE,side=2,expression(Cell~mass~density),line=1.3,cex= cex.lab)
mtext(outer=FALSE,side=2,expression((g ~cm^-3)),line=0.8,cex= cex.lab,adj=0.5)

mtext(side=1,'a',line=-5.7, adj=0.015,col='black',cex=0.7)

```

#Fig 2b)

*# We need to compare the difference in cell mass density effect across the radial
files between model configurations under a temperature increase
It happens that due to the stochastic properties of RINGS, radial files have
slightly different cell mass densities at different locations, which means that
when performing the binning especially for density-related variables,
towards the end of the radial files, fewer datapoints can be collected
in each bin, to the extent that in some cases below,
the last bin can e.g. only be filled with 3 datapoints.*

*# As the model configurations produce radial files of different lengths (max Di)
#, in order to calculate the difference between the radial files of different
model configurations and the default (full) model, we need to select the
minimum vector length for which all binned radial files have a reasonable
amount of datapoints. We determine that a reasonable minimum number of
datapoints in a bin is 10% of the maximum number of datapoints found in any
bin (renders a minimum vector length of 38)
going with this makes sense because this is equivalent to assuming that the
maximum number of datapoints found in any bin is the population size
and in statistics, 10% of the population size is a good sample size to get
a reasonable estimate of the population.*

```

#find maximum datapoints in any bins
maxpoints=max(collect_stdTrach_single[["full_model"]]]$n,
collect_stdTrach_single[["SucSat_full_temp_incr"]]]$n,
collect_stdTrach_single[["SucSat_model"]]]$n,
collect_stdTrach_single[["full_temp_incr"]]]$n,
collect_stdTrach_single[["THKOnly_temp_incr"]]]$n)

```

```

# get 10% of population size
npoints = floor(maxpoints-(maxpoints*0.9))
# determine minimum number of bins which all contain more or equal to a sample
# size that is 10% of the population size
n=min(length(which(collect_stdTrach_single[["full_model"]]]$n>=npoints)),

```

```

length(which(collect_stdTrach_single[["SucSat_full_temp_incr"]] $\$$ n>=npoints)),
length(which(collect_stdTrach_single[["SucSat_model"]] $\$$ n>=npoints)),
length(which(collect_stdTrach_single[["full_temp_incr"]] $\$$ n>=npoints)),
length(which(collect_stdTrach_single[["THKOnly_temp_incr"]] $\$$ n>=npoints)) )
# so, only select [1:n] of the vector lengths for plotting

#plot:
lwd=1.4
plot(NULL,ylim=c(-0.10,0.15),xlim=c(maxlen,0),ylab='',yaxt='n',xlab='',xaxt='n',
      axes=FALSE)
par(xpd=FALSE,cex.axis=0.7, cex.lab=0.7, cex.main=1, cex.sub=1,ps=8)
abline(h=0,col='grey',lwd=1.4) # signifying the 'full model'
mtext(side=1,'neutral \neffect',line=-2.7, adj=0.01,cex=0.6,col='grey')
lines(collect_stdTrach_single[["SucSat_full_temp_incr"]] $\$$ Di[1:n],
      (collect_stdTrach_single[["SucSat_full_temp_incr"]] $\$$ den[1:n] -
       collect_stdTrach_single[["SucSat_model"]] $\$$ den[1:n])/2,xlim=c(2500,0),
      col= "black",lty=2,lwd=lwd)
lines(collect_stdTrach_single[["full_temp_incr"]] $\$$ Di[1:n],
      (collect_stdTrach_single[["full_temp_incr"]] $\$$ den[1:n] -
       collect_stdTrach_single[["full_model"]] $\$$ den[1:n])/2,xlim=c(2500,0),
      col= "black",lty=1,lwd=lwd)
lines(collect_stdTrach_single[["THKOnly_temp_incr"]] $\$$ Di[1:n],
      (collect_stdTrach_single[["THKOnly_temp_incr"]] $\$$ den[1:n] -
       collect_stdTrach_single[["full_model"]] $\$$ den[1:n])/2,xlim=c(2500,0),
      col= "black",lty=4,lwd=lwd)

####select the area of more carbon/temp (positive values), to colour in red:
idx <- collect_stdTrach_single[["full_temp_incr"]] $\$$ den[1:n]-
       collect_stdTrach_single[["full_model"]] $\$$ den[1:n] > 0

# select for positive values:
x <- collect_stdTrach_single[["full_temp_incr"]] $\$$ Di[1:n][idx]
y <- (collect_stdTrach_single[["full_temp_incr"]] $\$$ den[1:n]-
      collect_stdTrach_single[["full_model"]] $\$$ den[1:n])[idx]/2

x.poly = c(x,tail(x, n=1),x[1])
y.poly = c(y,0,0)
polygon(x.poly, y.poly, col=makeTransparent("red",alpha=60), border=NA)

# select for negative values:

x <- collect_stdTrach_single[["full_temp_incr"]] $\$$ Di[1:n][!idx]
y <- (collect_stdTrach_single[["full_temp_incr"]] $\$$ den[1:n]-
      collect_stdTrach_single[["full_model"]] $\$$ den[1:n])[!idx]/2

x.poly = c(x,rev(x))
y.poly = c(y,rep(0,length(y)))
polygon(x.poly, y.poly, col=gray(0.9), border=NA)#gray(0.7)

#redraw black line
lines(collect_stdTrach_single[["full_temp_incr"]] $\$$ Di[1:n],
      (collect_stdTrach_single[["full_temp_incr"]] $\$$ den[1:n] -
       collect_stdTrach_single[["full_model"]] $\$$ den[1:n])/2,xlim=c(2500,0),

```

```

col= "black",lty=1,lwd=lwd)

axis(side = 2, lwd = 0, line = -0.8, las = 2)
axis(side = 2, tck = -.015, labels = NA)
#axis(side = 1, tck = -.015, labels = NA)
axis(1, at = seq(0,maxlen,by=200), labels = NA,tck= -.015, tick = TRUE)
axis(1, at = seq(0,maxlen,by=200),line = -1.3,lwd=0)

box()

mtext(outer=FALSE,side=2,expression(Delta*Carbon~deposition),line=1.3,cex= cex.lab,
      adj=0.1)
mtext(outer=FALSE,side=2,expression((g ~cm^-3) /Delta*K),line=0.8,cex= cex.lab,
      adj=0.33)

mtext(side=1,'b',line=-5.7, adj=0.015,col='black',cex=0.7)
mtext(outer=FALSE,side=1,expression(Distance~from~cambium~(~mu*m)),line=0.2,
      cex= cex.lab)

par(fig=c(7,10,1,6)/10)
par(new=T)
add_legend(0.58,-0.30, xpd=TRUE,legend=c('full model','sugar saturation',
                                         'wall thickening only'),
           lty=c(1,2,4),col="black", bty='n',cex = cex.legend)

add_legend(0.58,0.8, xpd=TRUE,legend=c('full model','full model +2K',
                                         'sugar saturation +2K',
                                         'wall thickening only +2K'),
           fill=c('black',runs_col[7],runs_col[8],runs_col[6]), bty='n',
           cex = cex.legend,title="Model configurations")

dev.off()
#####

#####FIGURE3#####
#maximum distance has changed, as these model configurations lead to
#slightly longer files
maxlen = mlen = 1910 #file length to process(um)
bs =100 #bin size (um)

#load data for each model configuration run:

full_model <- read.table(paste0(file_path,'fort.94.def'))
names(full_model) <- c("Di", "L", "M")
#full_model <- add_cellNumberColumn(full_model)

full_temp_incr<- read.table(paste0(file_path,'fort.94.two'))
names(full_temp_incr) <- c("Di", "L", "M")
#full_temp_incr <- add_cellNumberColumn(full_temp_incr)

Prolif_temp_incr<- read.table(paste0(file_path,'fort.94.twoP'))

```

```

names(Prolif_temp_incr) <- c("Di", "L", "M")
#Prolif_temp_incr <- add_cellNumberColumn(Prolif_temp_incr)

SucSat_model <- read.table(paste0(file_path,'fort.94.defC'))
names(SucSat_model) <- c("Di", "L", "M")
#SucSat_model <- add_cellNumberColumn(SucSat_model)

Enl_temp_incr<- read.table(paste0(file_path,'fort.94.twoE'))
names(Enl_temp_incr) <- c("Di", "L", "M")
#Enl_temp_incr <- add_cellNumberColumn(Enl_temp_incr)

# some data manipulation and aggregation:

runs <- list(full_model,full_temp_incr,Enl_temp_incr,Prolif_temp_incr)
names(runs) <- c("full_model","full_temp_incr","Enl_temp_incr","Prolif_temp_incr")

# loop through all model configurations and:
# add new variable-columns, that will hold derived variables Vc and den,
#from output variables:
# create binned tracheidograms for plotting
#initialise list of lists for data storage and easy retrieval in loops for plotting:
collect_stdTrach_single <- list(list(),list(),list(),list(),list())
names(collect_stdTrach_single) <- c("full_model","full_temp_incr","Enl_temp_incr",
                                   "Prolif_temp_incr")

for (e in names(runs)){
  # add new vars:
  print(e)
  runs[[e]]$Vc <- runs[[e]]$L * ttl * tal / 1.E12 # ml      eq 11 in manuscript
  runs[[e]]$den <- 1.E-6 * runs[[e]]$M / runs[[e]]$Vc # g/cm3
  # create binned trachs:
  collect_stdTrach_single[[e]] <- bin_and_running_mean(model_config_output = runs[[e]],
                                                         bin_size = bs,
                                                         maxlength_radial_file = mlen,
                                                         varname="L")
}

#plot:
jpeg(filename='Figures_update/Fig3.jpeg',units="mm",width=90, height=60, res=res)

par(mfrow=c(2,1),mar=c(0,2,0,0),oma=c(1,0,0.02,4),cex.axis=0.7, cex.lab=0.7,
    cex.main=1, cex.sub=1,ps=8,xpd=TRUE)

# add all simulated length positions:
plot(runs[["full_model"]]$Di,runs[["full_model"]]$L,pch=16,cex=0.5,
     col=makeTransparent('grey',alpha=50),xlim=c(maxlen,0),ylim=c(0,160),
     ylab='',xlab='',xaxt='n',yaxt='n')
points(runs[["full_temp_incr"]]$Di,runs[["full_temp_incr"]]$L,pch=16,
       cex=0.5,col=makeTransparent(runs_col[7],alpha=50))
points(runs[["Prolif_temp_incr"]]$Di,runs[["Prolif_temp_incr"]]$L,

```

```

    pch=16,cex=0.5,col=makeTransparent(runs_col[9],alpha=50))
points(runs[["Enl_temp_incr"]] $\$$ Di,runs[["Enl_temp_incr"]] $\$$ L,pch=16,
    cex=0.5,col=makeTransparent(runs_col[10],alpha=50))

# add binned tracheidograms
lines(collect_stdTrach_single[["full_model"]] $\$$ Di, collect_stdTrach_single[["full_model"]] $\$$ L,type="l",col=runs_col[9],lwd=2)
points(collect_stdTrach_single[["full_temp_incr"]] $\$$ Di, collect_stdTrach_single[["full_temp_incr"]] $\$$ L,type="l",col=runs_col[9],lwd=2)
points(collect_stdTrach_single[["Prolif_temp_incr"]] $\$$ Di,
    collect_stdTrach_single[["Prolif_temp_incr"]] $\$$ L,pch=2,type="l",
    col=runs_col[9], lwd=2)
points(collect_stdTrach_single[["Enl_temp_incr"]] $\$$ Di, collect_stdTrach_single[["Enl_temp_incr"]] $\$$ L,type="l",col=runs_col[9],lwd=2)

axis(side = 2,lwd = 0, line = -0.8, las = 2)
axis(side = 2, tck = -.015, labels = NA)
axis(1, at = seq(0,maxlen,by=200), labels = NA,tck= .015)

mtext(side=1,'a',line=-5.7, adj=0.015,col='black',cex=0.7)

mtext(outer=FALSE,side=2,expression(Cell~length ~( $\mu$ *m)),line=1,cex = cex.lab)

# We need to compare the difference in cell length effect across the radial files
# between model configurations under a temperature increase
# It happens that due to the stochastic properties of RINGS, radial files have
# different lengths, which means that when performing the binning especially
# for length-related variables, towards the end of the radial files, fewer
# datapoints can be collected in each bin, to the extent that in some cases below,
# the last bin is only filled with 1 datapoint.

# As the model configurations produce radial files of different lengths, in order
# to calculate the difference between the radial files of different model
# configurations and the default (full) model, we need to select the minimum
# vector length for which all binned radial files have a reasonable amount of
# datapoints. We deem appropriate the minimum number of datapoints in a bin is 10% of
# the maximum number of datapoints found in any bin (renders a minimum vector
# length of 16).
# Going with this makes sense because this is equivalent to assuming that the
# maximum number of datapoints found in any bin is the population size
# and in statistics, 10% of the population size is a good sample size to get
# a reasonable estimate of the population.

#find maximum datapoints in any bins
maxpoints=max(collect_stdTrach_single[["full_model"]] $\$$ n,
    collect_stdTrach_single[["Prolif_temp_incr"]] $\$$ n,
    collect_stdTrach_single[["full_temp_incr"]] $\$$ n,
    collect_stdTrach_single[["Enl_temp_incr"]] $\$$ n)

# get 10% of population size
npoints = floor(maxpoints-(maxpoints*0.9))
# determine minimum number of bins which all contain more or equal to a sample
# size that is 10% of the population size
n=min(length(which(collect_stdTrach_single[["full_model"]] $\$$ n>=npoints)),
    length(which(collect_stdTrach_single[["Prolif_temp_incr"]] $\$$ n>=npoints)),
    length(which(collect_stdTrach_single[["full_temp_incr"]] $\$$ n>=npoints)),

```

```

length(which(collect_stdTrach_single[["Enl_temp_incr"]][n>=npoints])) )
# so, only select [1:n] of the vector lengths for plotting

#plot;
lwd=1.4
par(xpd=FALSE)
plot(NULL,ylim=c(-4,8),xlim=c(maxlen,0),ylab='',yaxt='n',xlab='',xaxt='n')
abline(h=0,col='grey',lwd=1) # signifying the 'full model'
mtext(side=1,'neutral \neffect',line=-2.3, adj=0.05,cex=0.7,col='grey')
lines(collect_stdTrach_single[["Prolif_temp_incr"]][Di[1:n],
      (collect_stdTrach_single[["Prolif_temp_incr"]][L[1:n] -
        collect_stdTrach_single[["full_model"]][L[1:n]])/2,xlim=c(2500,0),
      col= "black",lty=2,lwd=lwd)
lines(collect_stdTrach_single[["Enl_temp_incr"]][Di[1:n],
      (collect_stdTrach_single[["Enl_temp_incr"]][L[1:n] -
        collect_stdTrach_single[["full_model"]][L[1:n]])/2,xlim=c(2500,0),
      col= "black",lty=4,lwd=lwd)
lines(collect_stdTrach_single[["full_temp_incr"]][Di[1:n],
      (collect_stdTrach_single[["full_temp_incr"]][L[1:n] -
        collect_stdTrach_single[["full_model"]][L[1:n]])/2,xlim=c(2500,0),
      col= "black",lty=1,lwd=lwd)

####select the area of more length/temp incr. (positive values), to colour in red:
idx <- collect_stdTrach_single[["full_temp_incr"]][L[1:n] - collect_stdTrach_single[["full_model"]][L[1:n]]
idx[16:17] <- FALSE # set these false here, as they will be plotted as their own polygon further below.

# select for the first round of positive values:
x <- seq(min(collect_stdTrach_single[["full_temp_incr"]][Di[1:n]],
      max(collect_stdTrach_single[["full_temp_incr"]][Di[1:n]]),length.out = length(collect_stdTrach_single[["full_temp_incr"]][L[1:n]]))
y <- (collect_stdTrach_single[["full_temp_incr"]][L[1:n]] -
      collect_stdTrach_single[["full_model"]][L[1:n]])[idx]/2

x.poly = c(x,tail(x, n=1),x[1])
y.poly = c(y,0,0)
polygon(x.poly, y.poly, col=makeTransparent("red",alpha=60), border=NA)

# select for negative values:
idx <- collect_stdTrach_single[["full_temp_incr"]][L[1:n] - collect_stdTrach_single[["full_model"]][L[1:n]]
# the polygon has to start from the last positive value, for the correct plotting,
#so selecting one additional index
idx[4] <- TRUE
x <- seq(min(collect_stdTrach_single[["full_temp_incr"]][Di[1:n]],
      max(collect_stdTrach_single[["full_temp_incr"]][Di[1:n]]),length.out = length(collect_stdTrach_single[["full_temp_incr"]][L[1:n]]))
y <- (collect_stdTrach_single[["full_temp_incr"]][L[1:n]] -
      collect_stdTrach_single[["full_model"]][L[1:n]])[idx]/2

x.poly = c(x,1500,x[1])#tail(x, n=1) manually set to 1500, because line crosses there in graph.
y.poly = c(y,0,0)
polygon(x.poly, y.poly, col=gray(0.9), border=NA)

####select the area of length/temp(positive values), to colour in red:

```

```

idx <- collect_stdTrach_single[["full_temp_incr"]] $\$$ L[1:n] - collect_stdTrach_single[["full_model"]] $\$$ L[
idx[1:4] <- FALSE # to make the polygon function work properly, define its own polygon for this second
idx[15] <- TRUE # the polygon has to start from the last negative value,
# select for the second round of positive values:
x <- seq(min(collect_stdTrach_single[["full_temp_incr"]] $\$$ Di[1:n]),
        max(collect_stdTrach_single[["full_temp_incr"]] $\$$ Di[1:n]),length.out = length(collect_stdTrach_
y <- (collect_stdTrach_single[["full_temp_incr"]] $\$$ L[1:n]-
      collect_stdTrach_single[["full_model"]] $\$$ L[1:n])[idx]/2

x.poly = c(x,tail(x, n=1),1500)#x[1] manually set to 1500, because line crosses there in graph.
y.poly = c(y,0,0)
polygon(x.poly, y.poly, col=makeTransparent("red",alpha=60), border=NA)

#redraw black line
lines(collect_stdTrach_single[["full_temp_incr"]] $\$$ Di[1:n],
      (collect_stdTrach_single[["full_temp_incr"]] $\$$ L[1:n] -
       collect_stdTrach_single[["full_model"]] $\$$ L[1:n])/2,xlim=c(2500,0),
      col= "black",lty=1,lwd=lwd)

mtext(side=1,'b',line=-5.7, adj=0.015,col='black',cex=0.7)
axis(side = 2, lwd = 0, line = -0.8, las = 2)
axis(side = 2, tck = -.015, labels = NA)
axis(1, at = seq(0,maxlen,by=200), labels = NA,tck= -.015, tick = TRUE)
axis(1, at = seq(0,maxlen,by=200),line = -1.3,lwd=0)

mtext(outer=FALSE,side=1,expression(Distance~from~cambium~( $\sim$ mu*m)),line=0.2,
      cex= cex.lab)
mtext(outer=FALSE,side=2,expression(Delta*Cell~length~change),line=1.2,
      cex= cex.lab,adj=0.3)
mtext(outer=FALSE,side=2,expression( $\sim$ with~temperature~( $\mu$ *m) /Delta*K),
      line=0.7,cex= cex.lab,adj=0.33)

par(fig=c(7,10,0,6)/10)
par(new=T)
add_legend(0.6,0.8, xpd=TRUE,legend=c('full model','full model +2K',
                                       'Prolif. +2K','Enlarg. +2K'),
          fill=c('black',runs_col[7],runs_col[9],runs_col[10]),
          bty='n',cex = cex.legend,title="Model configurations")

add_legend(0.6,-0.30, xpd=TRUE,legend=c('full model','Prolif.','Enlarg.'),
          lty=c(1,2,4),col="black", bty='n',cex = cex.legend)

dev.off()

```