Tutorial for DynamAedes

Daniele Da Re, Diego Montecino-Latorre, Sophie Vanwambeke, Matteo Marcantonio

30/9/2020

This tutorial provides a step-by-step guide to apply the stochastic, time-discrete and spatially-explicit population dynamical model developed in Da Re et. al 2020 (DOI:) for *Aedes aegypti* mosquitoes.

The model is coded in the R function DynamAedes.R.

The model requires three datasets: a numerical matrix with temperatures in Celsius defined in space and time (space in the rows, time in the columns), a two-column numerical matrix reporting the coordinates (in meters) of each space-unit (cell) and a numerical *distance matrix* which reports the distance in meters between the cells connected through a road network. For the purpose of this tutorial, we will use the following simulated datasets:

1. A 10 km lattice grid with 250 m cell size;
2. A 2-year long spatially and temporally correlated temperature time series;
3. A matrix of distances between cells connected through a simulated road network;

```
#Packages for processing
library(raster)
library(sp)
library(gstat)
library(spatstat)
library(maptools)
library(rgeos)
library(parallel)
library(eesim)

#Packages for plotting
library(ggplot2)
library(geosphere)
```

Prepare input data

Create lattice arena

First, we define the physical space where the introduction of our mosquitoes will happen. We define a squared lattice arena having 10 km side and 250 m resolution (40 colums and 40 rows, 1600 total cells).

```
gridDim <- 40 # 10000m/250 m = 40 columns and rows
xy <- expand.grid(x=1:gridDim, y=1:gridDim)
```

We then add a spatial pattern into the lattice area. This spatial pattern will be used later to add spatial correllation (SAC) to the temperature time series. The spatial autocorrelated pattern will be obtained using a semivariogram model with defined sill (value that the semivarion attains at the range) and range (distance of

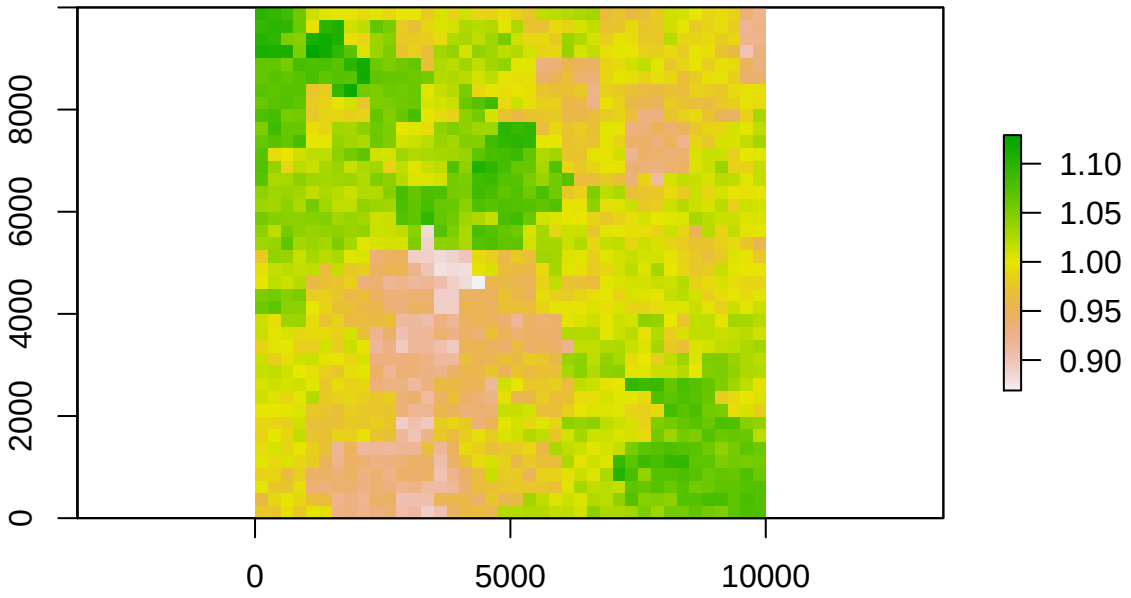
0 spatial correlation) and then predicting the semivariogram model over the lattice grid using unconditional Gaussian simulation.

```
varioMod <- vgm(psill=0.005, range=100, model='Exp') # psill = partial sill = (sill-nugget)
# Set up an additional variable from simple kriging
zDummy <- gstat(formula=z~1,
                locations = ~x+y,
                dummy=TRUE,
                beta=1,
                model=varioMod,
                nmax=1)
# Generate a randomly autocorrelated predictor data field
set.seed(123)
xyz <- predict(zDummy, newdata=xy, nsim=1)
```

```
## [using unconditional Gaussian simulation]
```

We generate a spatially autocorrelated raster adding the SA variable (*xyz$sim1*) to the RasterLayer object. The autocorrelated surface could for example represent the distribution of vegetation cover in a urban landscape.

```
utm32N <- "+proj=utm +zone=32 +ellps=WGS84 +datum=WGS84 +units=m +no_defs"
r <- raster(nrow=40, ncol=40, crs=utm32N, ext=extent(0,10000, 0,10000))
values(r)=xyz$sim1
plot(r)
```




```
df <- data.frame("id"=1:nrow(xyz), coordinates(r))
bbox <- as(extent(r), "SpatialPolygons")

# Store Parameters for autocorrelation
autocorr_factor <- values(r)
```

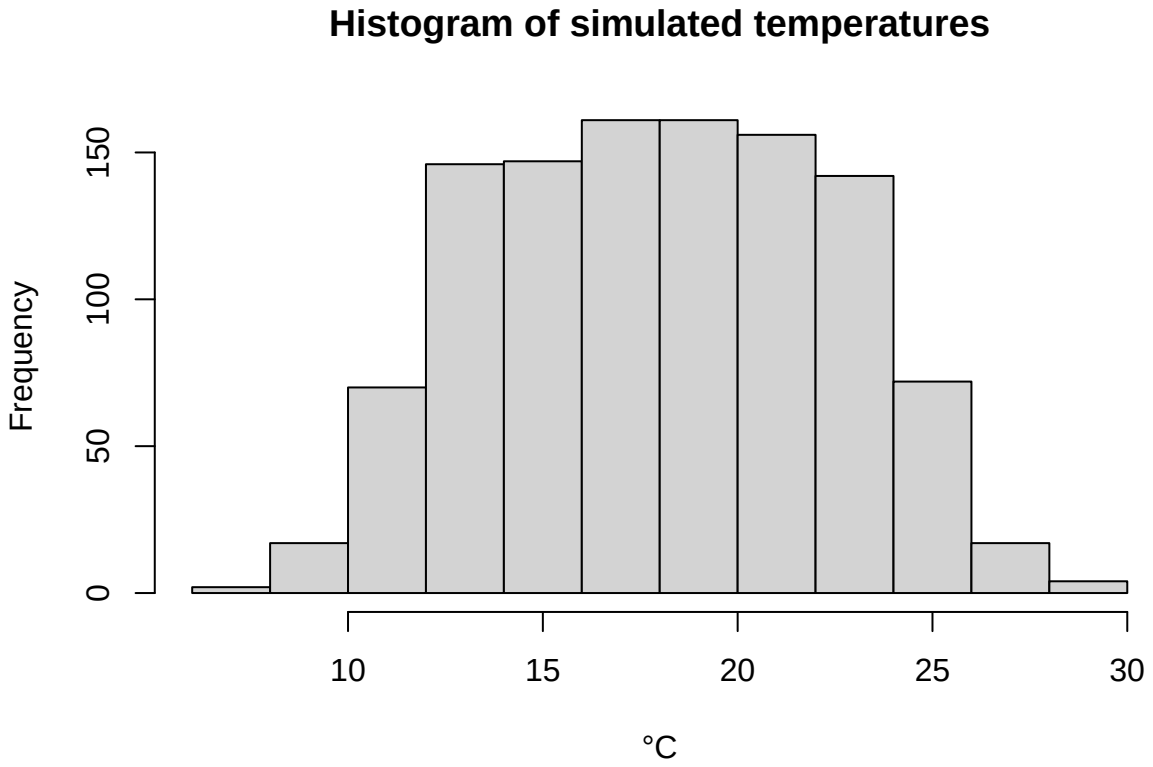
Simulate temperature data with seasonal trend

Next, we simulate a two-year temperature time series with seasonal trend. For the time series we consider a mean value of 18°C and standard deviation of 3°C.

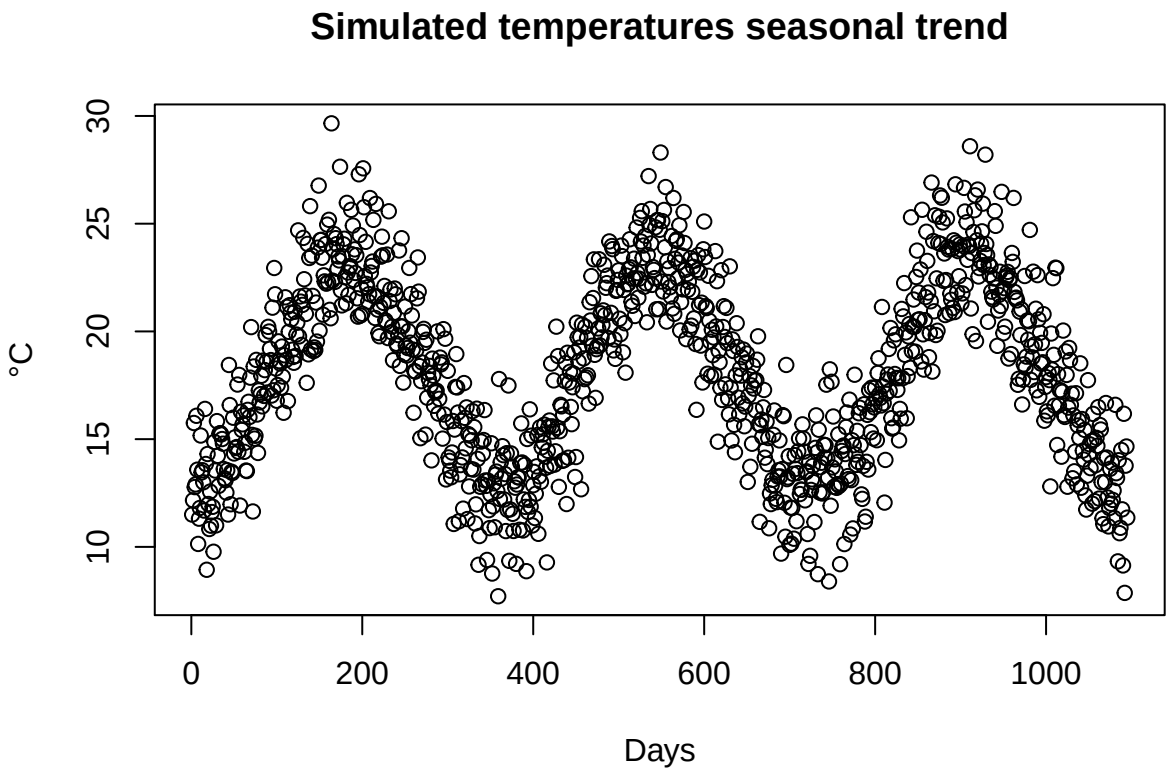
```
ndays = 365*3 #length of the time series in days
set.seed(123)
sim_temp <- create_sims(n_reps = 1,
                        n = ndays,
                        central = 18,
                        sd = 2,
                        exposure_type = "continuous",
                        exposure_trend = "cos1", exposure_amp = -.3,
                        average_outcome = 12,
                        outcome_trend = "cos1",
                        outcome_amp = 0.8,
                        rr = 1.0005)
```

A visualisation of the distribution of temperature values and the "average" temporal trend.

```
hist(sim_temp[[1]]$x,
     xlab="°C",
     main="Histogram of simulated temperatures")
```




```
plot(1:ndays,
     sim_temp[[1]]$x*mean(sapply(autocorr_factor,max)),
     main="Simulated temperatures seasonal trend", xlab="Days", ylab="°C")
```



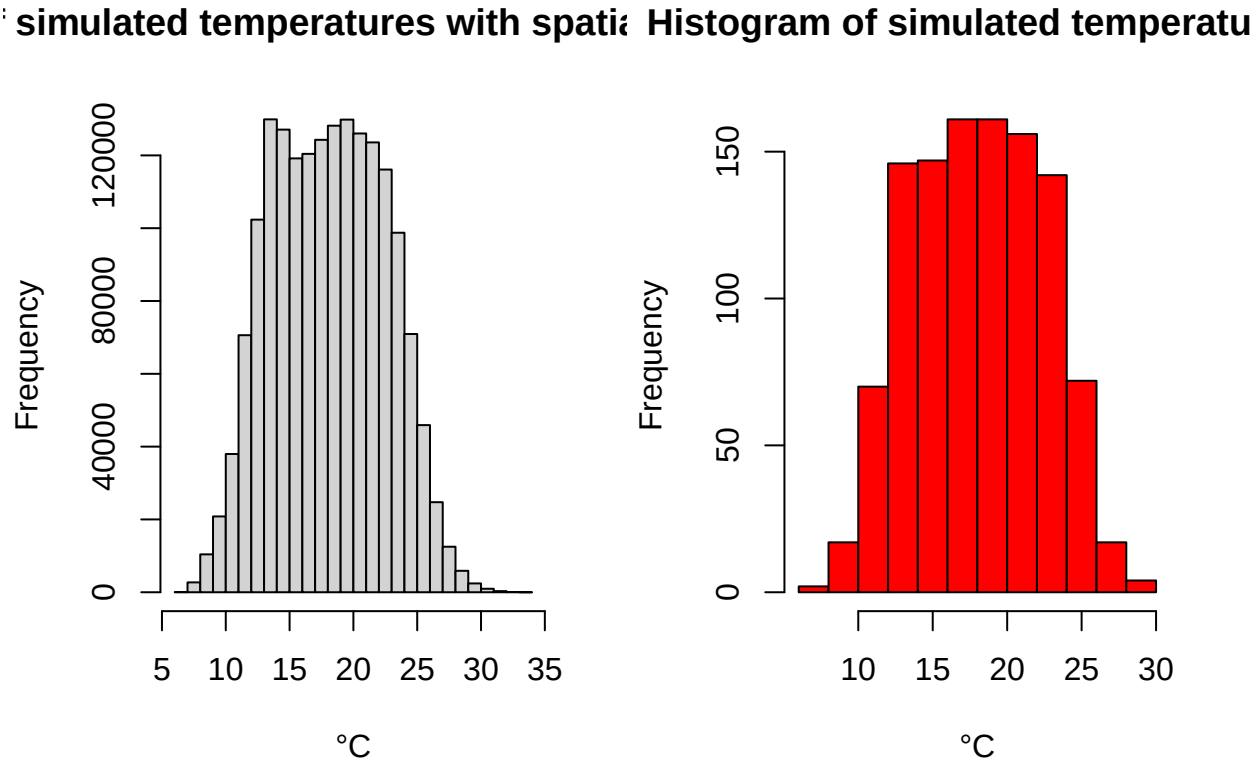
We can now "expand in space" the temperature time series by multiplying it with the autocorrelated surface simulated above.

```
mat <- mclapply(1:ncell(r), function(x) {
    d_t <- sim_temp[[1]]$x*autocorr_factor[[x]]
    return(d_t)
},mc.cores=1) #mc.cores=1 in Windows OS, which does not support mclapply function

mat <- do.call(rbind,mat)
```

A comparison betweem the distribution of the initial temperature time series and autocorrelated temperature surface

```
par(mfrow=c(1,2))
hist(mat, xlab="°C", main="Histogram of simulated temperatures with spatial autocorrelation")
hist(sim_temp[[1]]$x, xlab="°C", main="Histogram of simulated temperatures", col="red")
```




```
dev.off()
```

```
## null device
##           1
```

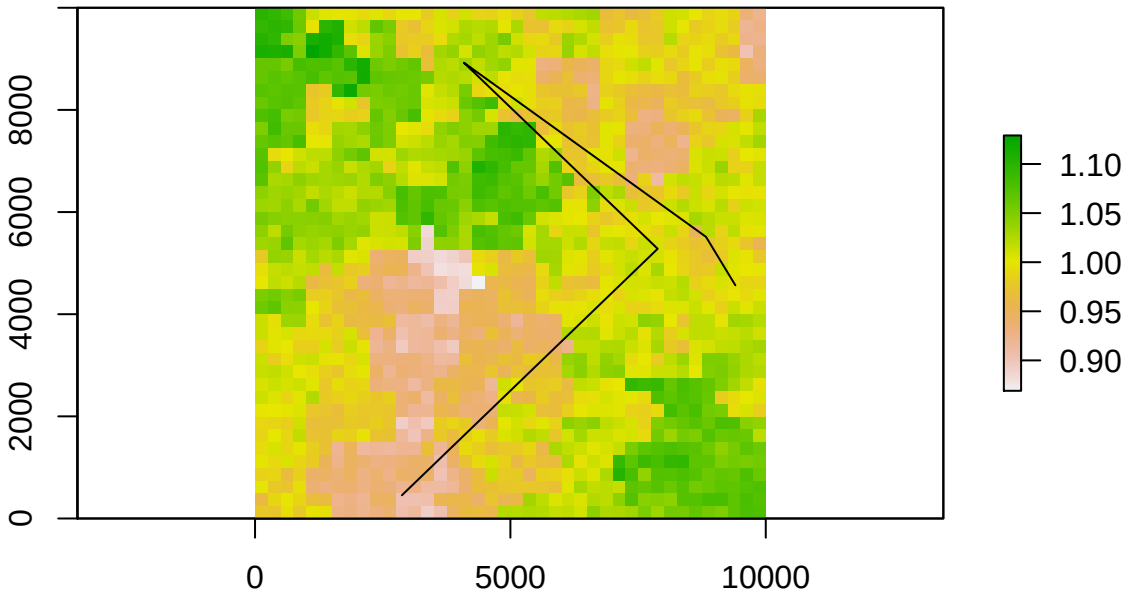
```
# Format temperature data
names(mat) <- paste0("d_", 1:ndays)
df_temp <- cbind(df, mat)
```

Simulate an arbitrary road segment for medium-range dispersal

In the model we have considered the possiblity of medium-range passive dispersal. Thus, we will simulate an arbitrary road segment along which adult mosquitoes can disperse passively (i.e., thorugh car traffic).

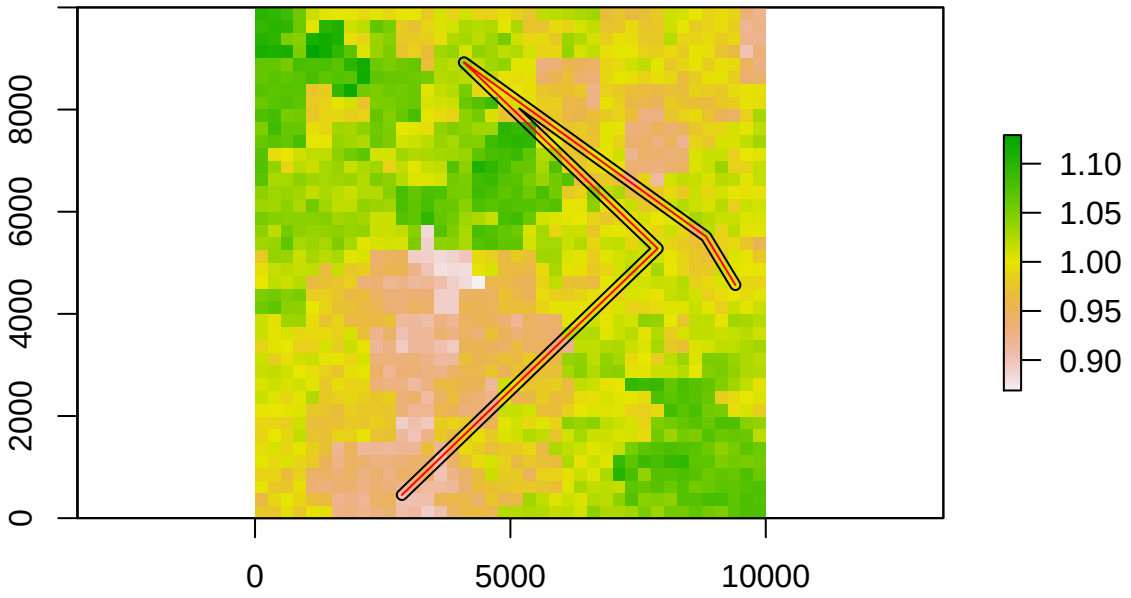
```
set.seed(123)
pts <- spsample(bbox, 5, type="random")
roads <- spLines(pts)

# Check simulated segment
raster::plot(r)
raster::plot(roads, add=T)
```



After defining the road segment we will add a "buffer" of 100 m around the road segment. Adult mosquitoes that reach or develop into cells comprised in the 100 m buffer around roads will be able to undergo passive dispersal.

```
buff <- buffer(roads, width=100)
crs(buff) <- crs(r)
# Check grid, road segment and buffer
raster::plot(r)
raster::plot(buff, add=T)
raster::plot(roads, add=T, col="red")
```

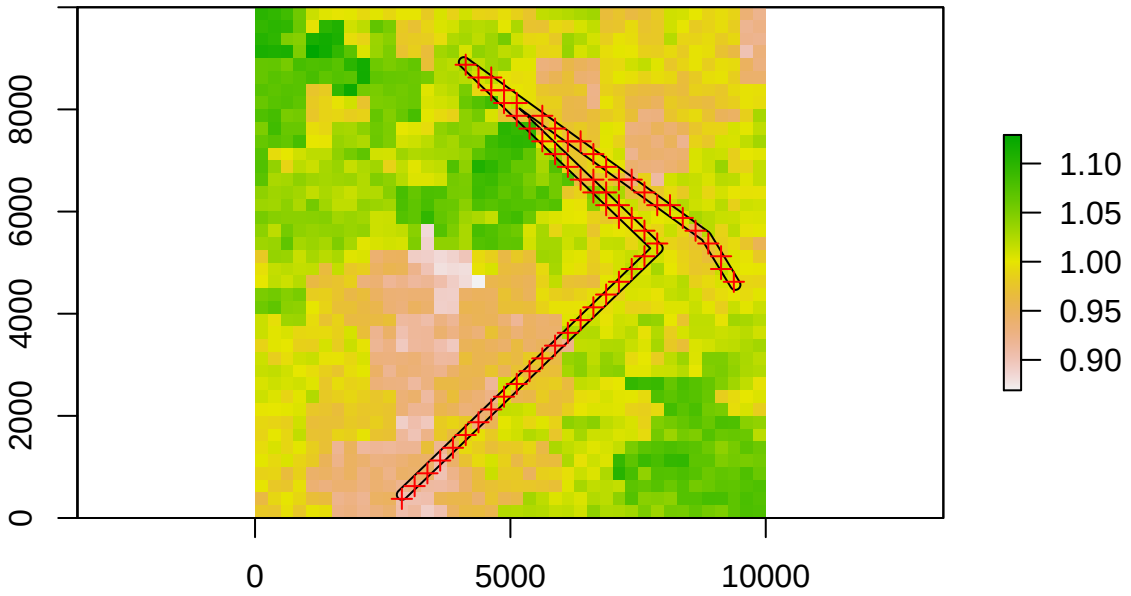


Next, we derive a distance matrix between cells comprised in the spatial buffer along the road network. First, we select the cells.

```
df_sp=df
coordinates(df_sp)=~x+y
df_sp=intersect(df_sp,buff)
```

```
## Warning in intersect(df_sp, buff): non identical CRS
```

```
# Check selected cells
raster::plot(r)
raster::plot(buff, add=T)
raster::plot(df_sp, add=T, col="red")
```



Then, we compute the Euclidean distance between each selected cell.

```
dist_matrix <- as.matrix(dist(coordinates(df_sp)))
```

Format the simulated input datasets and run the model

Model settings

Float numbers in the temperature matrix would slow the computational speed, thus we first multiply them by 1000 and then transform them in integer numbers.

```
w <- sapply(df_temp[,-c(1:3)], function(x) as.integer(x*1000))
```

We can now define a two-column matrix of coordinates to identify each cell in the lattice grid.

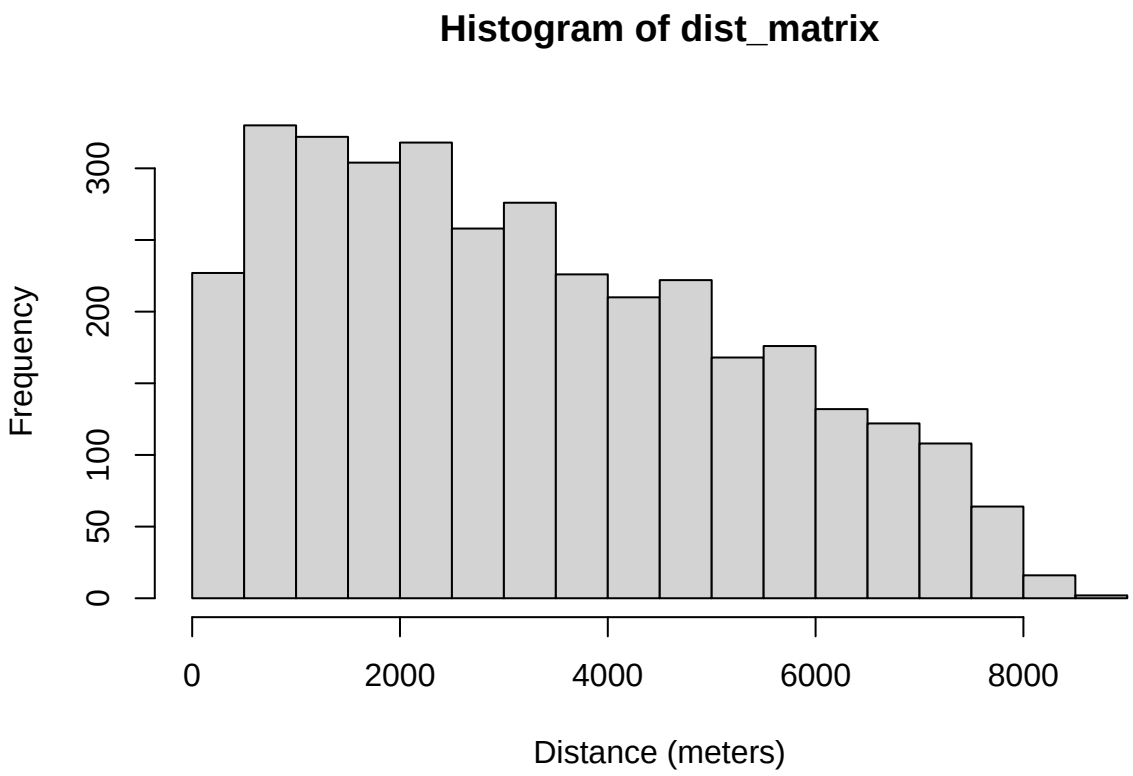
```
cc <- df_temp[,c("x","y")]
```

As for model requirement, the distance matrix must have column names equal to row names.

```
colnames(dist_matrix) <- row.names(dist_matrix)
```

Moreover, distances in the distance matrix must be rounded to the thousands.

```
dist_matrix <- apply(dist_matrix,2,function(x) round(x/1000,1)*1000)
# An histogram showing the distribution of distances of cells along the road network
hist(dist_matrix, xlab="Distance (meters)")
```



We are now left with a few model variables which need to be defined.

```
## Define cells into which introduce propagules on day 1
intro.vector <- as.numeric(row.names(dist_matrix))
## Define the day of introduction (day 1)
str = 121
## Define the end-day of life cycle
endr = 365*3;
## Define the number of eggs to be introduced
ie = 100;
## Define the number of model iterations
it = 10 # The higher the number of simulation the better
## Define the number of parallel processes (for sequential itarations set nc=1)
nc = 5
## Set folder where the *.RDS output will be saved
outfolder <- ('./output')
## Set working directory:
# IMPORTANT: This must be the directory where `DynamAedes.R` is placed
setwd('./')
## Source the function
source('DynamAedes.R')
```

Run the model

Running the model takes around 10 minutes with the settings specified in this example.

```
simout <- DynamAedes(temps.matrix=w,
                     cells.coords=cc,
                     road.dist.matrix=dist_matrix,
```

```
                    startd=str,
                    endd=endr,
                    n.clusters=nc,
                    cluster.type="SOCK", #in Windows OS
                    iter=it,
                    intro.cells=intro.vector,
                    intro.eggs=ie,
                    compressed.output=TRUE,
                    country="it", #other options are "es" or "nl"
                    suffix=paste(outfolder,"/DynamAedes_testrun_dayintro_",str,"_end",endr,"_niters",it
```

```
##
##         Loading required packages...
## Iterations concluded. Saving the output to: ./output/DynamAedes_testrun_dayintro_121_end1095_niters1
```

Analyze the results

To analyse and visualise model outputs we have first to define some variables.

```
## Derive maxium simulation length in days from run_DynamAedes.R output
days <- max(sapply(simout, function(x) length(x)))
## Define the day of introduction
str = 121 # In this case the 1 of April of the year of introduction
## Define the end-day of life cycle
endr = 365*3 # 31 December of the year following introduction
## Define temperature dates, assuming introduction was in 2019
tdates <- seq(as.Date("2019-01-01"),as.Date("2019-01-01")+ndays,by="day")
```

Derive probability of a successfull introduction at the end of the simulated period

```
pofe <- sum(unlist(lapply(simout, function(x) {
    days <- length(x)
    pe <- length(which(sum(x[days][[1]])>0))
    gc()
    return(pe)
})) / length(simout)

### Print the probability
cat("\n### p of successfull introduction is:",pofe," ###\n")
```

```
##
## ### p of successfull introduction is: 0.4  ###
```

Derive abundance 95% CI for each life stage and in each day

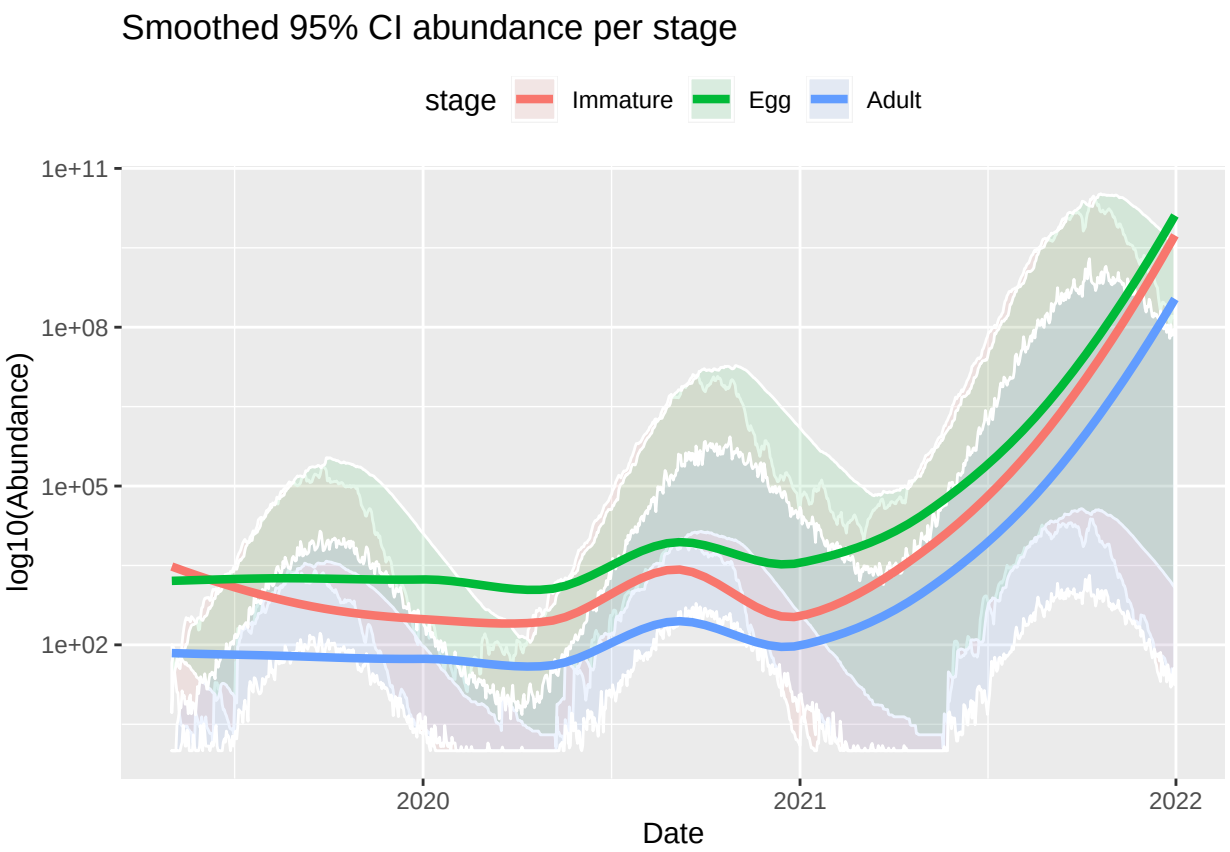
```
# Define function
dabu95ci <- function(outl=NA,st=1,cores=1,days=0){
    out <- apply(do.call(rbind.data.frame,mclapply(outl, function(x) {
        lapply(1:days, function(y) {
            if(y<=length(x)) {sum(x[[y]][st,],na.rm=T)} else {NA}})},mc.cores=cores)),2,quantile,probs=
```

```
    colnames(out)<-NULL
    outo <- rbind.data.frame(out,
        stage=rep(st,nrow(out)),
        day=as.factor(1:ncol(out)))
    return(t(outo))
}

# Apply function and format dataset for ggplot
dabu_df <- rbind.data.frame(dabu95ci(simout,1,days=days),
                            dabu95ci(simout,2,days=days),
                            dabu95ci(simout,3,days=days))
dabu_df$stage <- as.factor(dabu_df$stage)
levels(dabu_df$stage) <- c("Immature","Egg","Adult")
dabu_df$date <- rep(tdates[(str+1):(days+str)],3)

# Plot simple abundance time trendicel
ggplot(dabu_df, aes(y=`50%`+1,x=date,group=stage,col=stage)) +
    geom_ribbon(aes(ymin=`2.5%`+1,ymax=`97.5%`+1,fill=stage),
                col="white",
                alpha=0.1,
                outline.type="full") +
    geom_smooth(linetype=1,size=1.5,se=FALSE) +
    labs(y="log10(Abundance)") +
    scale_y_continuous(trans="log10") +
    xlab("Date") +
    theme(legend.pos="top") +
    ggtitle("Smoothed 95% CI abundance per stage")
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Plot female abundance per day, number of invaded cells and simulated temperature all together

Derive median and inter-quartile abundance per each cell and day

st defines the stage; 1=adult, 2=immature, 3=egg

```
st = 1
cabu75ci <- function(outl=NA,st=st,days=0,ppp=c(0.25,0.50,0.75)){
    out <- apply(do.call(rbind.data.frame,lapply(outl, function(x) {
        lapply(1:days, function(y) {
            if(y<=length(x)) {sum(x[[y]][st,],na.rm=T)} else {NA}})})),2,quantile,probs=ppp,na.rm=T);
    colnames(out)<-NULL
    outo <- rbind.data.frame(out,
        stage <- rep(st,nrow(out)),
        day <- as.factor(1:ncol(out)))
    return(t(outo))
}

## Apply the function
cabu <- rbind.data.frame(cabu75ci(simout,st=1,days=days),
                         cabu75ci(simout,st=2,days=days),
                         cabu75ci(simout,st=3,days=days))
## Format columns
names(cabu) <- c("a25%","a50%","a75%","stage", "date")
cabu$stage <- as.factor(cabu$stage)
```

```
levels(cabu$stage) <- c("Egg","Immature","Adult")
cabu$date <- rep(tdates[(str+1):length(tdates)],3)
```

Derive median and interquartile number of invaded cells per day

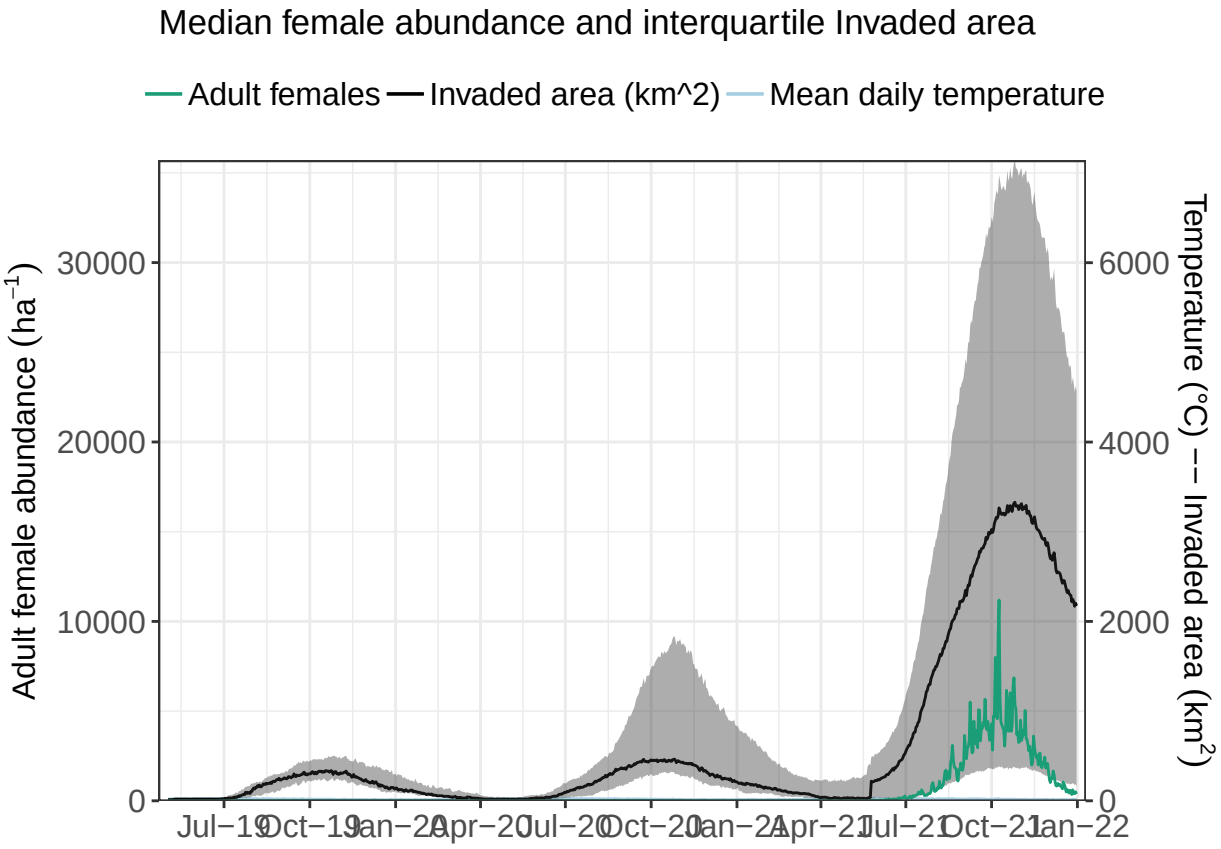
```
icel75ci <- function(outl=NA,days=0,ppp=c(0.25,0.50,0.75)){
            out <- apply(do.call(rbind.data.frame,lapply(outl, function(x) {
                lapply(1:days, function(y) {
                    if( y<=length(x) ) {
                        length(which(x[[y]]>0))
                    } else {NA}})})),2,quantile,probs=ppp,na.rm=T)
            colnames(out)<- NULL
            outo <- rbind.data.frame(out,
                day=as.factor(1:ncol(out)))
            return(t(outo))
}
### Apply function
icel <- rbind.data.frame(icel75ci(simout,days=days))
icel$day <- tail(tdates,-str)
names(icel) <- c("ic25%","ic50%","ic75%","date")
```

Merge female abundance per day with number of invaded cells and add simulated temperature time series

```
res = 250
dfp <- merge(cabu[cabu$stage%in%"Adult",], icel, by="date")
dfp$temp <- colMeans(mat)[(str):endr]
dfp[,c(6:8)] <- dfp[,c(6:8)]*res/40

### Plot the three trends together
ggplot(dfp, aes(y=`a50%`/`ic50%`, x=date, col="Adult females")) +
    geom_line(aes(y=temp*5, col="Mean daily temperature"),linetype=1) +
    geom_line(aes(y=`ic50%`*5, col="Invaded area (km^2)"),linetype=1) +
    geom_ribbon(aes(ymin=`ic25%`*5,ymax=`ic75%`*5),
                col="transparent",alpha=0.4,show.legend=FALSE) +
    geom_line(linetype=1,size=0.5,se=FALSE,alpha=1) +
    scale_color_manual(values = c("#1B9E77","black","#A6CEE3","grey60")) +
    scale_fill_manual(values = c("#1B9E77","black","#A6CEE3","grey60")) +
    labs(x=NULL,y=expression(paste("Adult female abundance ", (ha^-1)))) +
    guides(colour=guide_legend(title="")) +
    scale_y_continuous(sec.axis = sec_axis(name=expression(paste("Temperature (°C) -- Invaded area ", (
    scale_x_date(date_breaks = "3 months", date_labels = "%b-%y",expand=c(0.01,0.01)) +
    theme_bw() +
    theme(axis.text.x = element_text(angle=0,size=12),
          axis.text.y = element_text(size=12),axis.title.y = element_text(size=12),
          legend.position="top",legend.direction="horizontal",
          legend.text = element_text(size=12),
          legend.title = element_text(size=1,face="bold"),
          strip.text.x = element_text(size = 16),
          strip.text.y = element_text(size = 16)) +
    ggtitle("Median female abundance and interquartile Invaded area")
```

```
## Warning: Ignoring unknown parameters: se
```



Derive the Eucliden distance from each cell of introduction for all cells

```
## Define function
eudis <- function(x,x1,y,y1) {
    d <- sqrt((x1-x)^2 + (y1-y)^2)
    return(d)
}
## Select cells of introduction
introcells <- sapply(simout, function(x) which(colSums(x[[1]])>0))
### Apply the function to derive distances
coordd <- lapply(introcells, function(x) {
    ccc <- apply(cc, 1, function(y) {
        ccc <- eudis(x=cc[unlist(x),1],x1=y[1],y=cc[unlist(x),2],y1=y[2])
    })
    cc$distm <- ccc
    return(cc)
})
```

Select only cells with at least 1 mosquito and match cells with distance from cell from introduction

```
ainvc <- lapply(1:length(simout), function(d) {
    sapply(simout[[d]], function(y) {
        z <- data.frame(y[,which(colSums(y)>0)])
        colnames(z) <- which(colSums(y)>0)
        z[4,] <- coordd[[d]]$distm[which(colSums(y)>0)]
        return(z)
    })
})
```

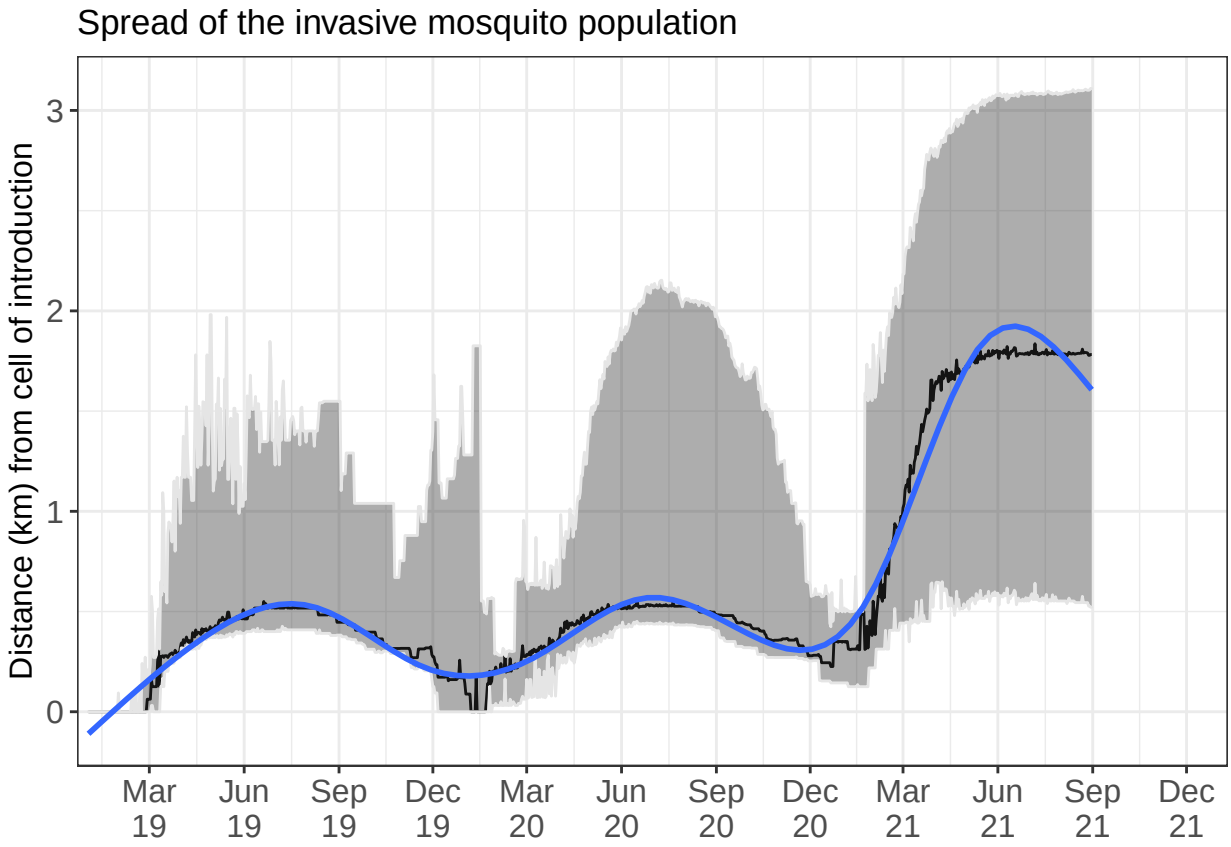
Derive interquartile of distances from cell of introduction for each day

```
interdis <- function(outl=NA,days=0){
    out <- do.call(rbind.data.frame, lapply(1:days,
        function(x) {
            quantile(unlist(sapply(1:length(outl),
                function(y) {
                    mean(if(x<length(outl[[y]])) as.integer(outl[[y]][[x]][4,]) else NA,na.rm=T)
                })), probs=c(0.25,0.50,0.75), na.rm=T)
        }))
    names(out) <- c("25%","50%","75%")
    return(out)
}

### Apply the function
## Format date
asp <- interdis(ainvc,days=endr)
asp$date <- head(tdates,-1)

### Plot the interquartile of distances at which mosquitoes are found every day
ggplot(asp, aes(y=`50%`,x=date)) +
  geom_line() +
  geom_ribbon(aes(ymin=`25%`,ymax=`75%`),colour="grey90",alpha=0.4) +
  geom_smooth(se=FALSE) +
  scale_color_brewer(palette = "Dark2") +
  scale_fill_brewer(palette = "Dark2") +
  labs(x=NULL,y="Distance (km) from cell of introduction") +
  scale_y_continuous(labels=function(x) format(x/1000, scientific = FALSE)) +
  scale_x_date(date_breaks = "3 months", date_labels = "%b\n%y",expand=c(0.01,0.01)) +
  theme_bw() +
  theme(axis.text.x = element_text(angle=0,size=12),
        axis.text.y = element_text(size=12),
        axis.title.y = element_text(size=12),
        legend.position="top",
        legend.direction="horizontal",
        legend.text = element_text(size=12),
        legend.title = element_text(size=1,face="bold"),
        strip.text.x = element_text(size = 16),
        strip.text.y = element_text(size = 16))+
  ggtitle("Spread of the invasive mosquito population")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
## Warning: Removed 121 rows containing non-finite values (stat_smooth).
## Warning: Removed 121 row(s) containing missing values (geom_path).
```



Some bonus plots and information

Plot crude average spatial pattern for a single iteration

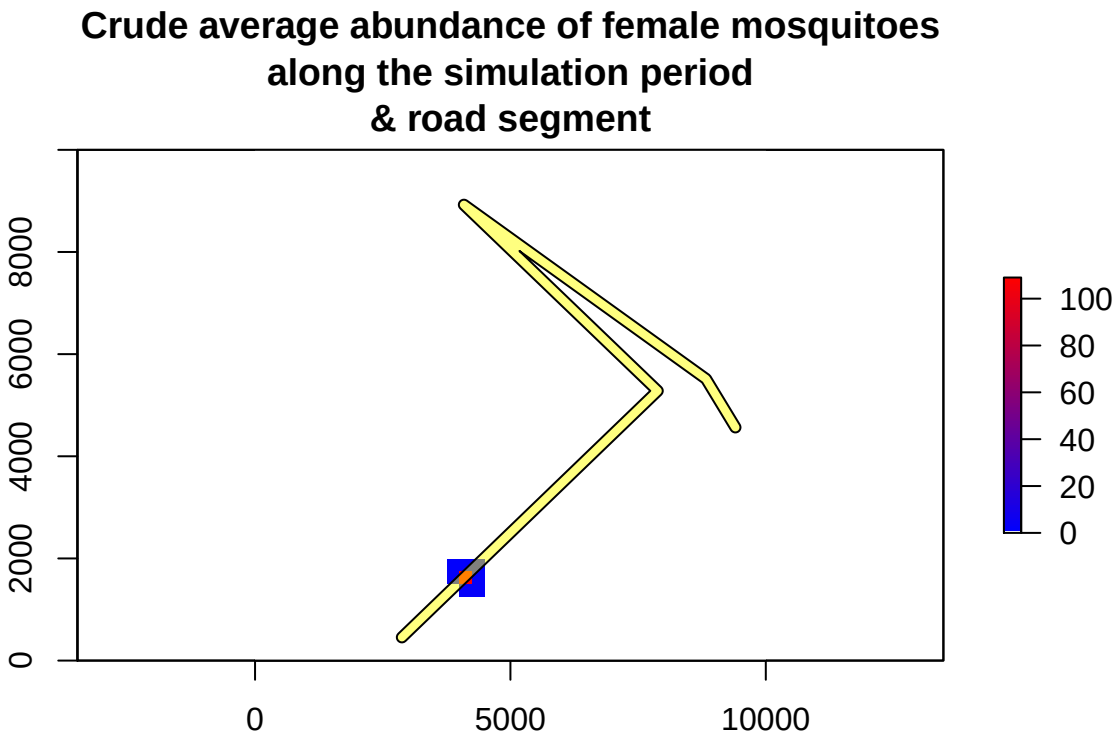
```
# Select simulation to visualise
simu <- 3
# Reduce over time
summ <- Reduce(`+`,simout[[simu]])
# Use `r` raster defined in run_DynamAedes.R as template
rsum <- r

# Plug in crude average abundace per day per cell
values(rsum)<- summ[3,]/days

# Define an adequate color palette
maxColorValue <- max(values(rsum))
palette <- colorRampPalette(c("white","blue","red"),bias=100, alpha = TRUE)(maxColorValue)

# Plot invasion trend in space with road segment
raster::plot(rsum,col = palette)
```

```
raster::plot(buff,col = rgb(red = 1, green = 1, blue = 0, alpha = 0.5),add=T)
title("Crude average abundance of female mosquitoes\nalong the simulation period\n& road segment")
```



Visualise spatial dynamics of adult abudances day by day

```
print("Don't run me")
# Select simulation to visualise
simu <- 3
for (i in 1:ndays){
  message(i)
  # Reduce over time
  summ <- simout[[simu]][i]
  # Use `r` raster defined in run_DynamAedes.R as template
  rsum <- r
  # Plug in crude average abundace per day per cell
  values(rsum)<- ifelse(summ[[1]][3,]==0,NA,summ[[1]][3,])
  # Define an adequate color palette
  #maxColorValue <- ifelse(max(values(rsum),na.rm=TRUE)>30000,30000,max(values(rsum),na.rm=FALSE))
  if(is.na(maxColorValue)){next()}
  palette <- colorRampPalette(c("green","blue","yellow","red"),bias=0.8)(maxColorValue)
  if(length(palette)==0){next()}
  # Plot invasion trend in space with road segment
  raster::plot(rsum,col = palette)
  plot(buff,add=T,col = rgb(red = 1, green = 1, blue = 0, alpha = 0.5))
  text(x=1000,y=100,label=paste("day",i),family="bold")
  title("Crude average abundance of female mosquitoes\nalong the simulation period\n& road segment")
  Sys.sleep(0.05)
}
```

Get date of and median number of adults at population peak

```
dfp[dfp$stage%in%"Adult",][which.max(dfp[dfp$stage%in%"Adult",]$`a50%`),]
```

```
##           date a25%     a50%      a75% stage ic25% ic50% ic75%  temp
## 892 2021-10-09 5514 36546225 585152439 Adult 385.9  3266  6973 22.94
```

Get date of and maximum interquartile distance travelled by mosquitoes

```
asp[which.max(asp$`75%`),]
```

```
##       25%  50%  75%       date
## 973 521.2 1782 3110 2021-08-30
```