Appendix S1: Harbor Seal Ice Dynamics Data Analysis

1 MCAR Model

1.1 Model statement

$$
\begin{aligned}
\mathbf{y}_{ijk} &\sim \text{Normal}(\boldsymbol{\mu}_{ij}, \mathbf{V}), \\
\boldsymbol{\mu}_{ij} &\sim \text{Normal}(\mathbf{x}'_{ij}\boldsymbol{\beta}_{ijl}, \boldsymbol{\Sigma}), \\
\boldsymbol{\Sigma} &\equiv (\text{diag}(\mathbf{W1}) - \rho\mathbf{W})^{-1} \otimes \boldsymbol{\Lambda} \\
\mathbf{V} &\sim \text{Wishart}^{-1}(k_v, \mathbf{I}_{d\times d}) \\
\boldsymbol{\Lambda} &\sim \text{Wishart}^{-1}(k_\lambda, \mathbf{I}_{d\times d}) \\
\boldsymbol{\beta} &\sim \text{Normal}(\mathbf{0}, \sigma^2_\beta \mathbf{I}_{p\times p}) \\
\rho &\sim \text{Uniform}(0, 1)
\end{aligned}
$$

where $\mathbf{y}_{ijk} \equiv (\text{adults}_{ijk}, \text{pups}_{ijk}, \text{ice}_{ijk})'$ are centered and scaled counts of adults, pups, and ice proportion at site $i = 1, \ldots, n$, time $j = 1, \ldots, J$, and photograph $k = 1, \ldots, K$, $\mathbf{V}$ is a covariance matrix characterizing differences in $\mathbf{y}_{ijk}$ within a cell with mean $\boldsymbol{\mu}_{ij}$. The design matrix $\mathbf{X}$ is the same for all responses $l = 1, \ldots, 3$, but parameter estimates $\boldsymbol{\beta}_{ijl}$ may be different among responses. $\boldsymbol{\Sigma}$ is a covariance matrix characterizing the multivariate and spatial covariance, where $(\text{diag}(\mathbf{W1}) - \rho\mathbf{W})^{-1}$ is the CAR prior for the spatial correlation, with adjacecy matrix $\mathbf{W}$, diagonal matrix $\text{diag}(\mathbf{W1})$ consisting of the number of neighbors on the diagonal, and spatial correlation parameter ρ. The matrix $\boldsymbol{\Lambda}$ describes the covariance of the multivariate response.

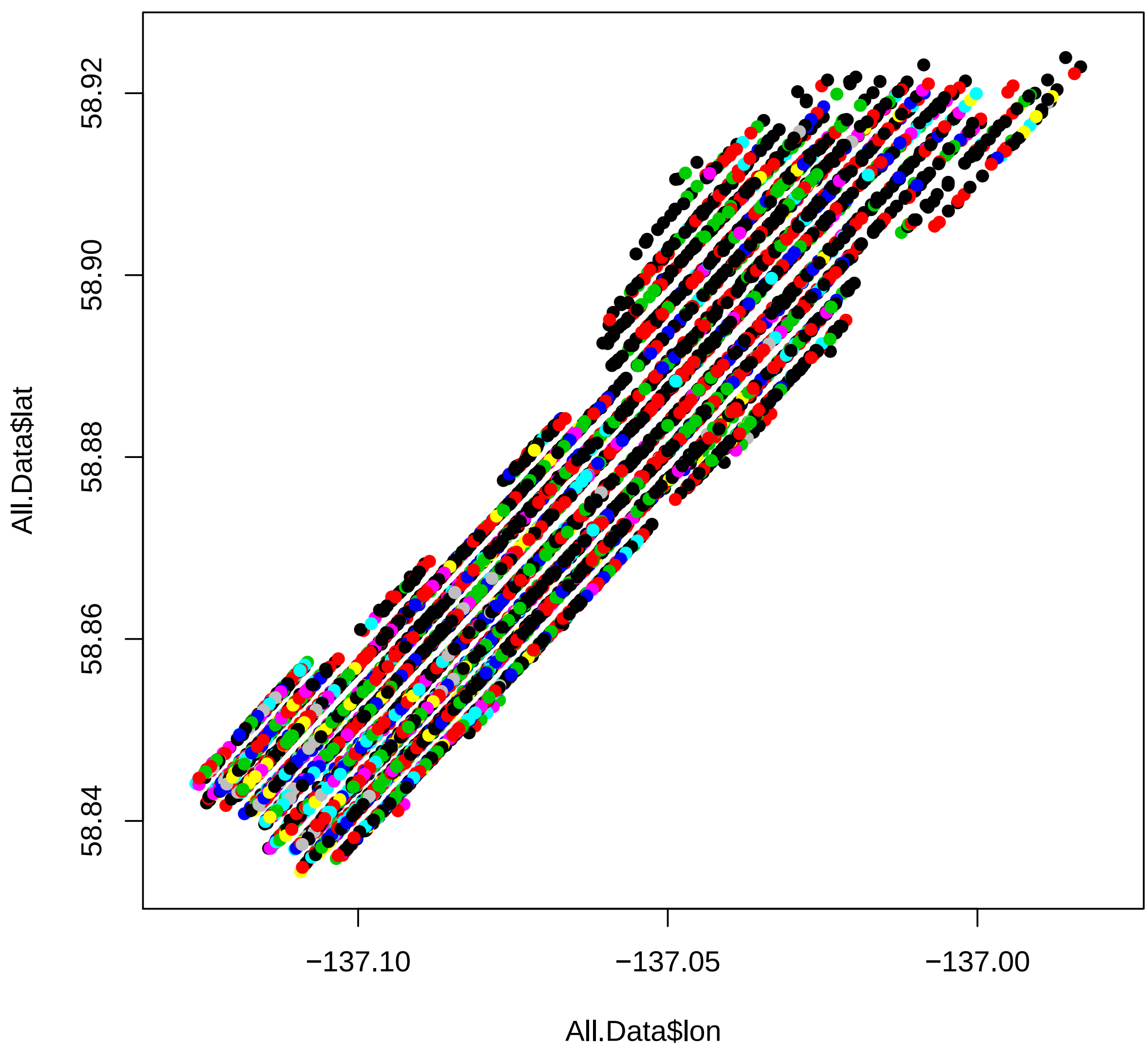
1.2 Fit model in R

```
###
### Libraries and packages
###

required.packages=c("CARBayes",
                    "MASS",
                    "oce",
                    "RColorBrewer",
                    "raster",
                    "spatstat",
                    "spdep",
                    "xtable"
                    )
## install.packages(required.packages)
lapply(required.packages,
       library,
       character.only=TRUE)

## [[1]]
##  [1] "xtable"       "spdep"        "sf"            "spData"
##  [5] "spatstat"     "rpart"        "nlme"          "spatstat.data"
##  [9] "oce"          "gsw"          "testthat"      "CARBayes"
## [13] "Rcpp"         "MASS"         "knitr"         "splines"
## [17] "rgeos"        "rgdal"        "RcppArmadillo" "RCurl"
## [21] "bitops"       "rasterVis"    "latticeExtra"  "RColorBrewer"
## [25] "lattice"      "raster"       "parallel"      "mvtnorm"
## [29] "matrixStats"  "maptools"     "sp"            "itsmr"
## [33] "inline"       "gstat"        "gridExtra"     "ggmap"
## [37] "ggplot2"      "fields"       "maps"          "spam"
## [41] "grid"         "dotCall64"    "fBasics"       "timeSeries"
## [45] "timeDate"     "coda"         "stats"         "graphics"
## [49] "grDevices"    "utils"        "datasets"      "methods"
## [53] "base"
##
## [[2]]
##  [1] "xtable"       "spdep"        "sf"            "spData"
##  [5] "spatstat"     "rpart"        "nlme"          "spatstat.data"
##  [9] "oce"          "gsw"          "testthat"      "CARBayes"
## [13] "Rcpp"         "MASS"         "knitr"         "splines"
## [17] "rgeos"        "rgdal"        "RcppArmadillo" "RCurl"
## [21] "bitops"       "rasterVis"    "latticeExtra"  "RColorBrewer"
## [25] "lattice"      "raster"       "parallel"      "mvtnorm"
## [29] "matrixStats"  "maptools"     "sp"            "itsmr"
## [33] "inline"       "gstat"        "gridExtra"     "ggmap"
## [37] "ggplot2"      "fields"       "maps"          "spam"
```

```
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[3]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[4]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[5]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
```

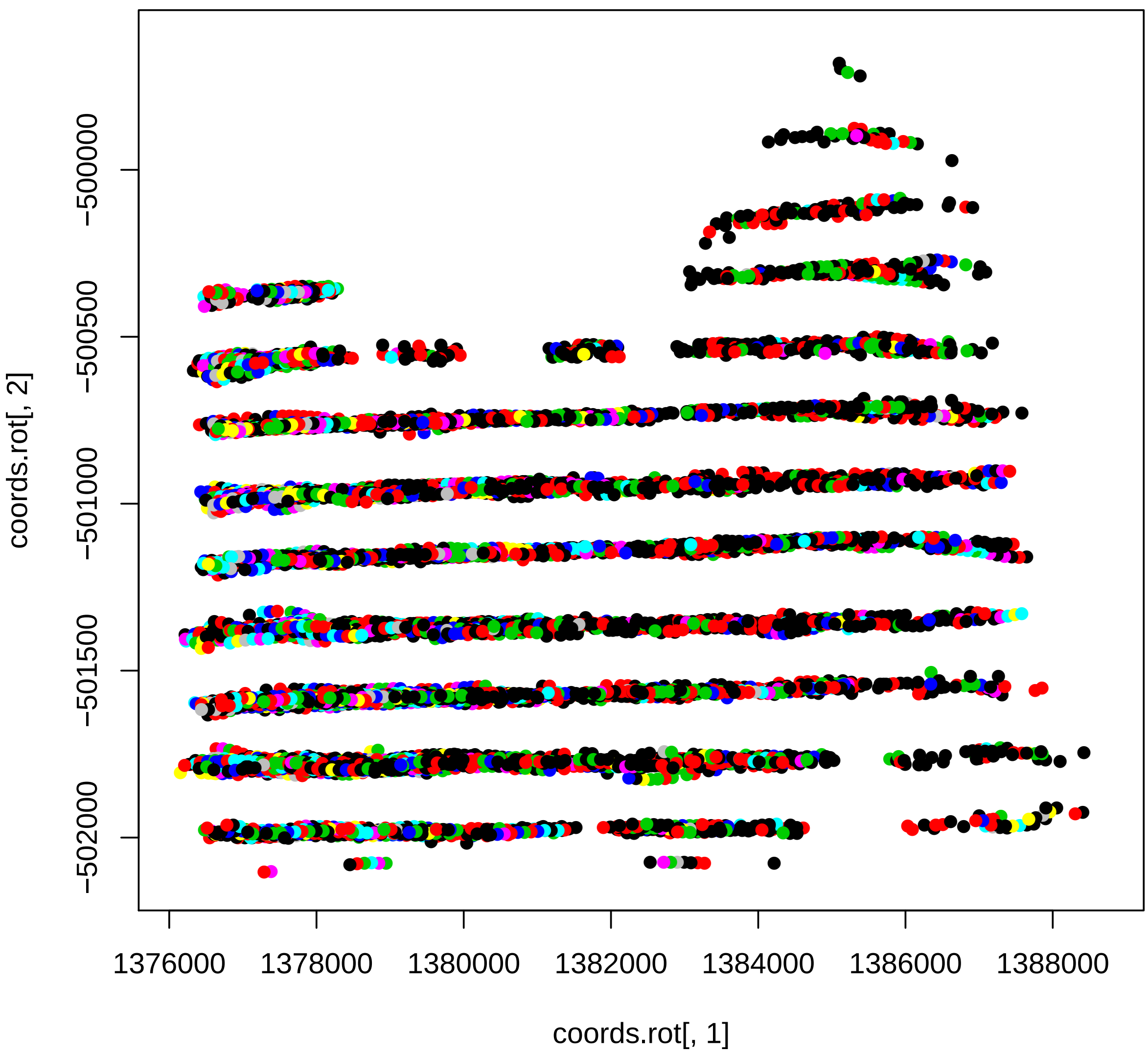
```
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[6]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[7]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[8]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
```

```
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"

depth3d=function(x,y,z, pmat, minsize=0.2, maxsize=2) {

    ## determine depth of each point from
    ## xyz and transformation matrix pmat
    tr=as.matrix(cbind(x, y, z, 1)) %*% pmat
    tr=tr[,3]/tr[,4]

    ## scale depth to point sizes between minsize and maxsize
    psize=((tr-min(tr) ) * (maxsize-minsize)) / (max(tr)-min(tr)) + minsize
    return(psize)
}

###
### Load all data
###

All.Data=read.csv(paste0("~/HarborSealData.csv"))
All.Data$month[All.Data$month==7]=6
plot(All.Data$lon,All.Data$lat,pch=16,col=All.Data$binSealsAdult)
```




```
###
### Rotate data for computation
###

coords.rot=Rotation(cbind(All.Data$x,All.Data$y),-69*pi/180)
ymin=min(coords.rot[,2])
ymax=max(coords.rot[,2])
xmin=min(coords.rot[,1])
xmax=max(coords.rot[,1])
head(coords.rot)

##          [,1]       [,2]
```

```
## [1,] 1385029 -501758.5
## [2,] 1384930 -501758.3
## [3,] 1384832 -501758.1
## [4,] 1384733 -501758.8
## [5,] 1384633 -501759.2
## [6,] 1384535 -501760.1

plot(coords.rot[,1],coords.rot[,2],pch=16,col=All.Data$binSealsAdult)
```




```
###
### XYZ dataframe
###
```

```
y_1=scale(All.Data$binSealsAdult)
y_2=scale(All.Data$binSealsPup)
y_3=scale(All.Data$binIcePercent)
time=All.Data$j_date-min(All.Data$j_date)+1
season=factor(All.Data$month)
year=as.factor(All.Data$year)
period=All.Data$period
dist=scale(All.Data$dist2calve)
data=data.frame(coords.rot[,1],
                coords.rot[,2],
                y_1,
                y_2,
                y_3,
                dist,
                season,
                year)

########################
### June Data Analysis
########################

###
### Subset data
###

Y.save=list()
model.output=list()
June.ind=seq(1,16,2)
for(k in June.ind){
    d1=subset(data, period==k)
    names(d1)=c("x","y","y_1","y_2","y_3","dist","season","year")

    ##
    ## Bin data
    ##

    xbins=round((xmax-xmin)/200)
    ybins=round((ymax-ymin)/300)
    bin.list=list(x=cut(d1$x,breaks=seq(min(d1$x),
                                        max(d1$x),
                                        length.out=xbins),
                        include.lowest=TRUE),
                  y=cut(d1$y,breaks=seq(min(d1$y),
                                        max(d1$y),
                                        length.out=ybins),
                        include.lowest=TRUE)
```

```
                )
y_1.bin=with(d1,tapply(y_1,bin.list,mean))
y_2.bin=with(d1,tapply(y_2,bin.list,mean))
y_3.bin=with(d1,tapply(y_3,bin.list,mean))
d.bin=with(d1,tapply(dist,bin.list,mean))

##
## Settings
##

nc=xbins-1
nr=ybins-1
n=nr*nc
d=3
Y=matrix(c(y_1.bin,y_2.bin,y_3.bin),n,d)
Y.save[[k]]=Y
x.vals=1:(xbins-1)
y.vals=1:(ybins-1)
Grid=expand.grid(x.vals, y.vals)

##
## Distance and neighborhood structure
##

distance=as.matrix(dist(Grid))
W=array(0,c(n,n))
W[distance==1]=1
D=diag(rowSums(W))

##
## Covariate model
##

X=matrix(d.bin,n,1)
d.m=matrix(d.bin,nr,nc,byrow=TRUE)
which(d.m==min(d.m,na.rm=TRUE))
epi=c(5,1)
dnew.m=matrix(,nr,nc)
for(i in 1:nr){
    for(j in 1:nc){
        dnew.m[i,j]=sqrt((i-epi[1])^2+(j-epi[2])^2)
    }
}
X=scale(c(dnew.m))

##
## Fit the MCAR model
```

```
    ##

    n.iter=1050000
    burn=50000
    thin=100
    formula=Y~X

    ## Uncomment to refit the model
    ## model.output=MVS.CARleroux(formula=formula,
    ##                            family="gaussian",
    ##                            W=W,
    ##                            thin=thin,
    ##                            burnin=burn,
    ##                            n.sample=n.iter
    ##                            )
    ## save(model.output,file=paste0("~/MCARFit",
    ##                               k,".RData"))

}

########################
### August Data Analysis
########################

###
### Subset data
###

August.ind=seq(2,16,2)
for(k in August.ind){
    d1=subset(data, period==k)
    names(d1)=c("x","y","y_1","y_2","y_3","dist","season","year")

    ##
    ## Bin data
    ##

    xbins=round((xmax-xmin)/200)
    ybins=round((ymax-ymin)/300)
    bin.list=list(x=cut(d1$x,breaks=seq(min(d1$x),
                                          max(d1$x),
                                          length.out=xbins),
                        include.lowest=TRUE),
                  y=cut(d1$y,breaks=seq(min(d1$y),
                                          max(d1$y),
                                          length.out=ybins),
                        include.lowest=TRUE)
```

```
                )
y_1.bin=with(d1,tapply(y_1,bin.list,mean))
y_3.bin=with(d1,tapply(y_3,bin.list,mean))
d.bin=with(d1,tapply(dist,bin.list,mean))

##
## Settings
##

nc=xbins-1
nr=ybins-1
n=nr*nc
d=2
Y=matrix(c(y_1.bin,y_3.bin),n,d)
Y.save[[k]]=Y
x.vals=1:(xbins-1)
y.vals=1:(ybins-1)
Grid=expand.grid(x.vals, y.vals)

##
## Distance and neighborhood structure
##

distance=as.matrix(dist(Grid))
W=array(0,c(n,n))
W[distance==1]=1
D=diag(rowSums(W))

##
## Covariate model
##

X=matrix(d.bin,n,1)
d.m=matrix(d.bin,nr,nc,byrow=TRUE)
which(d.m==min(d.m,na.rm=TRUE))
epi=c(5,1)
dnew.m=matrix(,nr,nc)
for(i in 1:nr){
    for(j in 1:nc){
        dnew.m[i,j]=sqrt((i-epi[1])^2+(j-epi[2])^2)
    }
}
X=scale(c(dnew.m))

##
## Fit the MCAR model
##
```

```
    formula=Y~X

    ## Uncomment to refit the model
    ## model.output=MVS.CARleroux(formula=formula,
    ##                               family="gaussian",
    ##                                 W=W,
    ##                                 thin=thin,
    ##                                 burnin=burn,
    ##                                 n.sample=n.iter
    ##                            )
    ## save(model.output,file=paste0("~/MCARFit",
    ##                               k,".RData"))
    ## print(model.output$summary.results)
}

save(Y.save,file=paste0("~/Y.save.RData"))
```

1.3 Results

```
###
### Load required packages
###

required.packages=c("CARBayes",
                    "MASS",
                    "oce",
                    "RColorBrewer",
                    "raster",
                    "spatstat",
                    "spdep"
                    )
## install.packages(required.packages)
lapply(required.packages,
       library,
       character.only=TRUE)

## [[1]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[2]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
```

```
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[3]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[4]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
## [41] "grid"          "dotCall64"     "fBasics"       "timeSeries"
## [45] "timeDate"      "coda"          "stats"         "graphics"
## [49] "grDevices"     "utils"         "datasets"      "methods"
## [53] "base"
##
## [[5]]
##  [1] "xtable"        "spdep"         "sf"            "spData"
##  [5] "spatstat"      "rpart"         "nlme"          "spatstat.data"
##  [9] "oce"           "gsw"           "testthat"      "CARBayes"
## [13] "Rcpp"          "MASS"          "knitr"         "splines"
## [17] "rgeos"         "rgdal"         "RcppArmadillo" "RCurl"
## [21] "bitops"        "rasterVis"     "latticeExtra"  "RColorBrewer"
## [25] "lattice"       "raster"        "parallel"      "mvtnorm"
## [29] "matrixStats"   "maptools"      "sp"            "itsmr"
## [33] "inline"        "gstat"         "gridExtra"     "ggmap"
## [37] "ggplot2"       "fields"        "maps"          "spam"
```

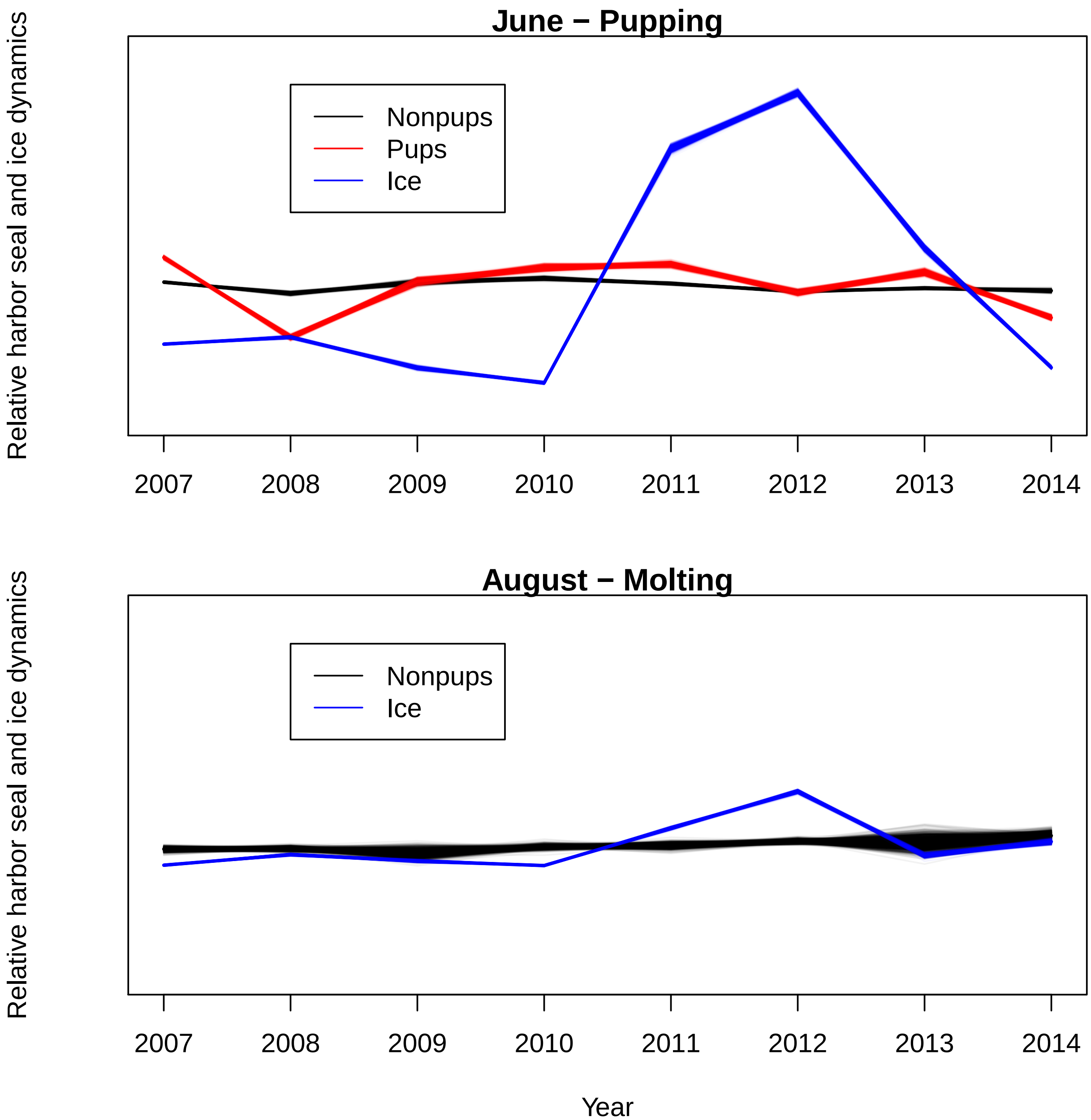
```
## [41] "grid"           "dotCall64"      "fBasics"        "timeSeries"
## [45] "timeDate"       "coda"           "stats"          "graphics"
## [49] "grDevices"      "utils"          "datasets"       "methods"
## [53] "base"
##
## [[6]]
##  [1] "xtable"         "spdep"          "sf"             "spData"
##  [5] "spatstat"       "rpart"          "nlme"           "spatstat.data"
##  [9] "oce"            "gsw"            "testthat"       "CARBayes"
## [13] "Rcpp"           "MASS"           "knitr"          "splines"
## [17] "rgeos"          "rgdal"          "RcppArmadillo" "RCurl"
## [21] "bitops"         "rasterVis"      "latticeExtra"   "RColorBrewer"
## [25] "lattice"        "raster"         "parallel"       "mvtnorm"
## [29] "matrixStats"    "maptools"       "sp"             "itsmr"
## [33] "inline"         "gstat"          "gridExtra"      "ggmap"
## [37] "ggplot2"        "fields"         "maps"           "spam"
## [41] "grid"           "dotCall64"      "fBasics"        "timeSeries"
## [45] "timeDate"       "coda"           "stats"          "graphics"
## [49] "grDevices"      "utils"          "datasets"       "methods"
## [53] "base"
##
## [[7]]
##  [1] "xtable"         "spdep"          "sf"             "spData"
##  [5] "spatstat"       "rpart"          "nlme"           "spatstat.data"
##  [9] "oce"            "gsw"            "testthat"       "CARBayes"
## [13] "Rcpp"           "MASS"           "knitr"          "splines"
## [17] "rgeos"          "rgdal"          "RcppArmadillo" "RCurl"
## [21] "bitops"         "rasterVis"      "latticeExtra"   "RColorBrewer"
## [25] "lattice"        "raster"         "parallel"       "mvtnorm"
## [29] "matrixStats"    "maptools"       "sp"             "itsmr"
## [33] "inline"         "gstat"          "gridExtra"      "ggmap"
## [37] "ggplot2"        "fields"         "maps"           "spam"
## [41] "grid"           "dotCall64"      "fBasics"        "timeSeries"
## [45] "timeDate"       "coda"           "stats"          "graphics"
## [49] "grDevices"      "utils"          "datasets"       "methods"
## [53] "base"

###
### Load model fit
###

model.output.l=list()
for(i in 1:16){
    load(paste0("~/ModelFit/MCARFit",i,
             ".RData"))
    model.output.l[[i]]=model.output
}
```

```r
load("~/Y.save.RData")

nc=62
nr=7
n=nr*nc
Boundary=rep(NA,n)
Boundary[!is.na(Y.save[[1]][,1])]=1

#########################################################################
### Figure 2 in manuscript
#########################################################################

###
### June Nonpups
###

d=3
lines=1000
adult.June.marg=matrix(NA,lines,8)
for(i in 1:8){
    y.pred=model.output.l[[June.ind[i]]]$samples$fitted
    for(k in 1:lines){
        y.pred.ad=y.pred[k,seq(1,nr*nc*d,d)]
        adult.June.marg[k,i]=sum(y.pred.ad*Boundary,na.rm=TRUE)
    }
}
adult.June.marg.c=adult.June.marg-mean(adult.June.marg)

par(mfrow=c(2,1),mar=c(4,4,1,1))
plot(2007:2014,apply(adult.June.marg.c,2,quantile,0.5,na.rm=TRUE),
     type='l',

     xlab="",
     ylab="Relative harbor seal and ice dynamics",
     yaxt='n',
     ylim=c(-200,350),
     lwd=2,
     main="June - Pupping")

for(k in 1:lines){
    lines(2007:2014,adult.June.marg.c[k,],
          col=rgb(0,0,0,0.03))
}

###
### June Pups
###
```

```r
pup.June.marg=matrix(NA,10000,8)
for(i in 1:8){
    y.pred=model.output.l[[June.ind[i]]]$samples$fitted
    for(k in 1:10000){
        y.pred.ad=y.pred[k,seq(2,nr*nc*d,d)]
        pup.June.marg[k,i]=sum(y.pred.ad*Boundary,na.rm=TRUE)
    }
}
pup.June.marg.c=pup.June.marg-mean(pup.June.marg)
lines(2007:2014,apply(pup.June.marg.c,2,quantile,0.5),
      lwd=2,
      col=2)
for(k in 1:lines){
    lines(2007:2014,pup.June.marg.c[k,],
          col=rgb(1,0,0,0.05))
}

###
### June ice
###

ice.June.marg=matrix(NA,10000,8)
for(i in 1:8){
    y.pred=model.output.l[[June.ind[i]]]$samples$fitted
    for(k in 1:10000){
        y.pred.ad=y.pred[k,seq(3,nr*nc*d,d)]
        ice.June.marg[k,i]=sum(y.pred.ad*Boundary,na.rm=TRUE)
    }
}
ice.June.marg.c=ice.June.marg-mean(ice.June.marg)
lines(2007:2014,apply(ice.June.marg.c,2,quantile,0.5),
      lwd=2,
      col=4)
for(k in 1:lines){
    lines(2007:2014,ice.June.marg.c[k,],
          col=rgb(0,0,1,0.03))
}
legend(2008,300,
       legend=c("Nonpups","Pups","Ice"),
       lty=1,
       col=c(1,2,4)
       )

###
### August Nonpups
###
```

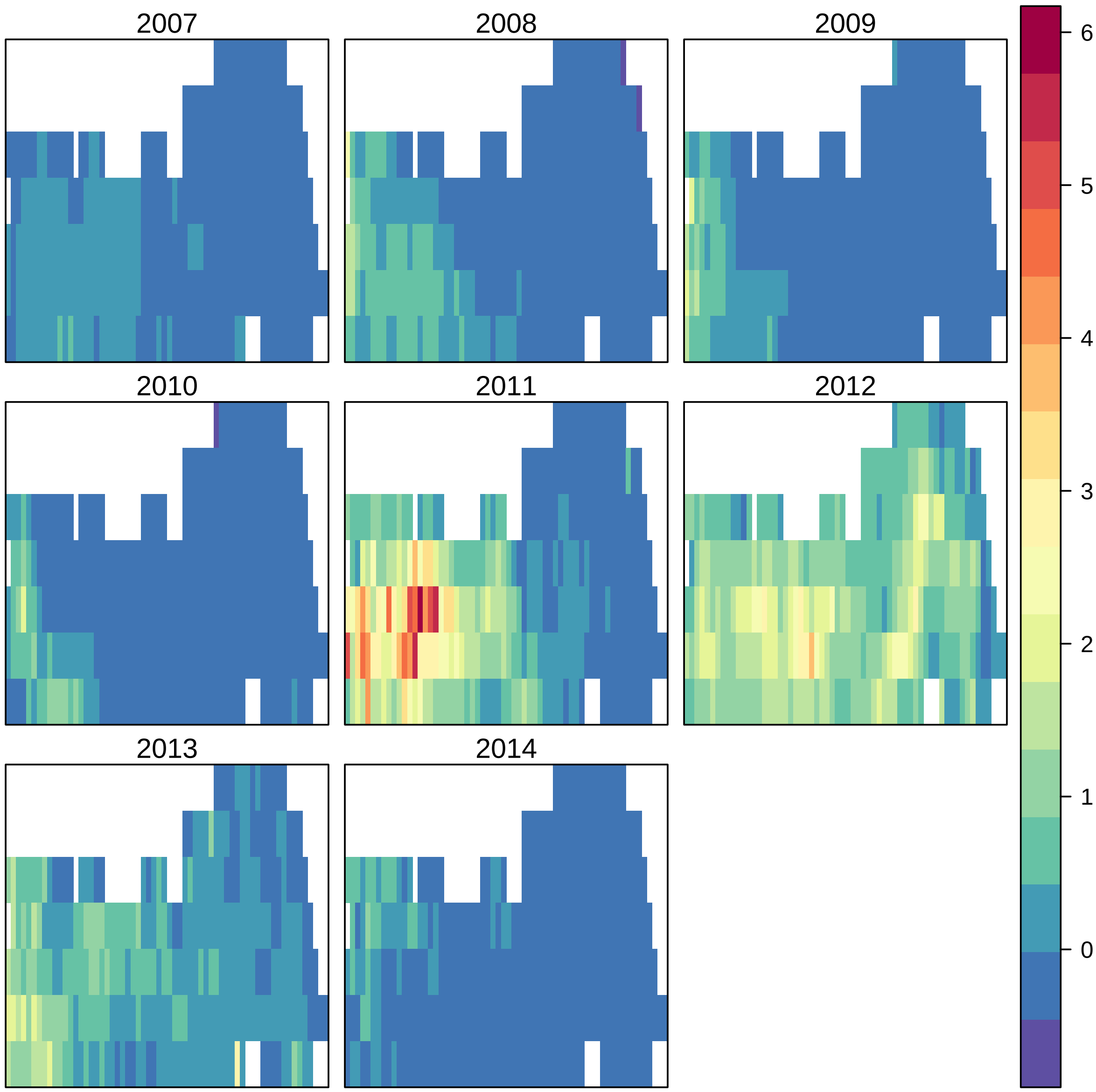
```
d=2
adult.August.marg=matrix(NA,10000,8)
for(i in 1:8){
    y.pred=model.output.l[[August.ind[i]]]$samples$fitted
    for(k in 1:10000){
        y.pred.ad=y.pred[k,seq(1,nc*nr*d,d)]
        adult.August.marg[k,i]=sum(y.pred.ad*Boundary,na.rm=TRUE)
    }
}
adult.August.marg.c=adult.August.marg-mean(adult.August.marg)

plot(2007:2014,apply(adult.August.marg.c,2,quantile,0.5),
     type='l',
     xlab="Year",
     ylab="Relative harbor seal and ice dynamics",
     yaxt='n',
     ylim=c(-200,350),
     lwd=2,
     main="August - Molting")

for(k in 1:lines){
    lines(2007:2014,adult.August.marg.c[k,],
          col=rgb(0,0,0,0.05))
}

###
### August ice
###

ice.August.marg=matrix(NA,10000,8)
for(i in 1:8){
    y.pred=model.output.l[[August.ind[i]]]$samples$fitted
    for(k in 1:10000){
        y.pred.ad=y.pred[k,seq(2,nc*nr*d,d)]
        ice.August.marg[k,i]=sum(y.pred.ad*Boundary,na.rm=TRUE)
    }
}
ice.August.marg.c=ice.August.marg-mean(ice.August.marg)
lines(2007:2014,apply(ice.August.marg.c,2,quantile,0.5),
      lwd=2,
      col=4)
for(k in 1:lines){
    lines(2007:2014,ice.August.marg.c[k,],
          col=rgb(0,0,1,0.05))
}
legend(2008,300,
```

```
      legend=c("Nonpups","Ice"),
      lty=1,
      col=c(1,4)
      )
```



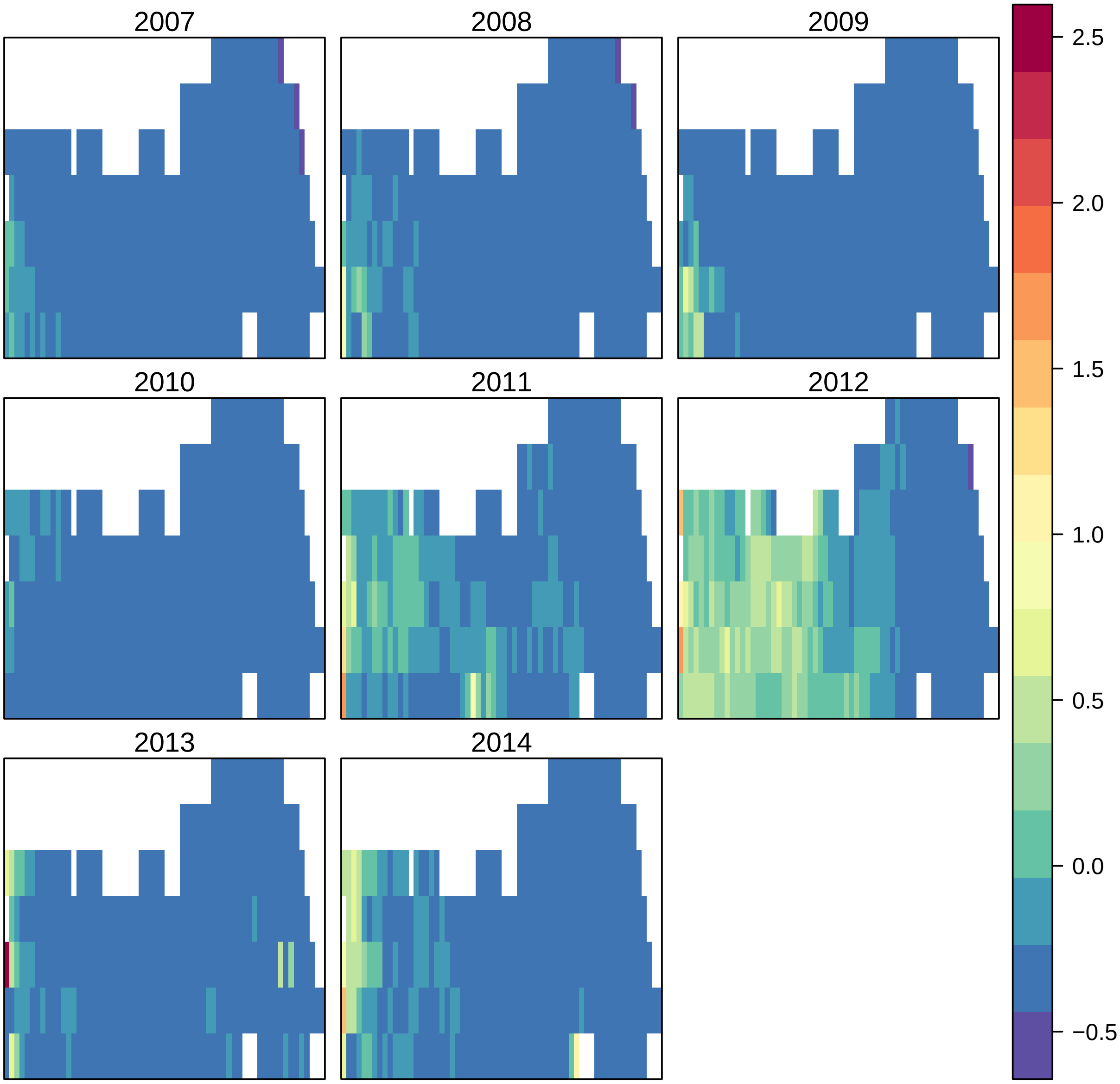

```
######################################################################
### Figure 3 in manuscript
######################################################################

June.Ice=list()
June.Pups=list()
June.Adults=list()
```

```r
August.Ice=list()
August.Nonpups=list()

###
### June annual posterior median in Ice
###

for(i in 1:8){
    med.m=apply(model.output.l[[June.ind[i]]]$samples$fitted,2,quantile,0.5)
    med.m.Ice=med.m[seq(3,nr*nc*3,3)]
    June.Ice[[i]]=matrix(med.m.Ice*Boundary,nc,nr)
    med.m=apply(model.output.l[[August.ind[i]]]$samples$fitted,2,quantile,0.5)
    med.m.Ice=med.m[seq(2,nr*nc*2,2)]
    August.Ice[[i]]=matrix(med.m.Ice*Boundary,nc,nr)
}

June.Ice.med.rs=stack(raster(t(June.Ice[[1]])[nr:1,]),
                  raster(t(June.Ice[[2]])[nr:1,]),
                  raster(t(June.Ice[[3]])[nr:1,]),
                  raster(t(June.Ice[[4]])[nr:1,]),
                  raster(t(June.Ice[[5]])[nr:1,]),
                  raster(t(June.Ice[[6]])[nr:1,]),
                  raster(t(June.Ice[[7]])[nr:1,]),
                  raster(t(June.Ice[[8]])[nr:1,])
                  )

rasterVis::levelplot(June.Ice.med.rs,
          col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                         bias=1),
          names.attr=as.character(2007:2014), scales=list(draw=FALSE)
          )
```



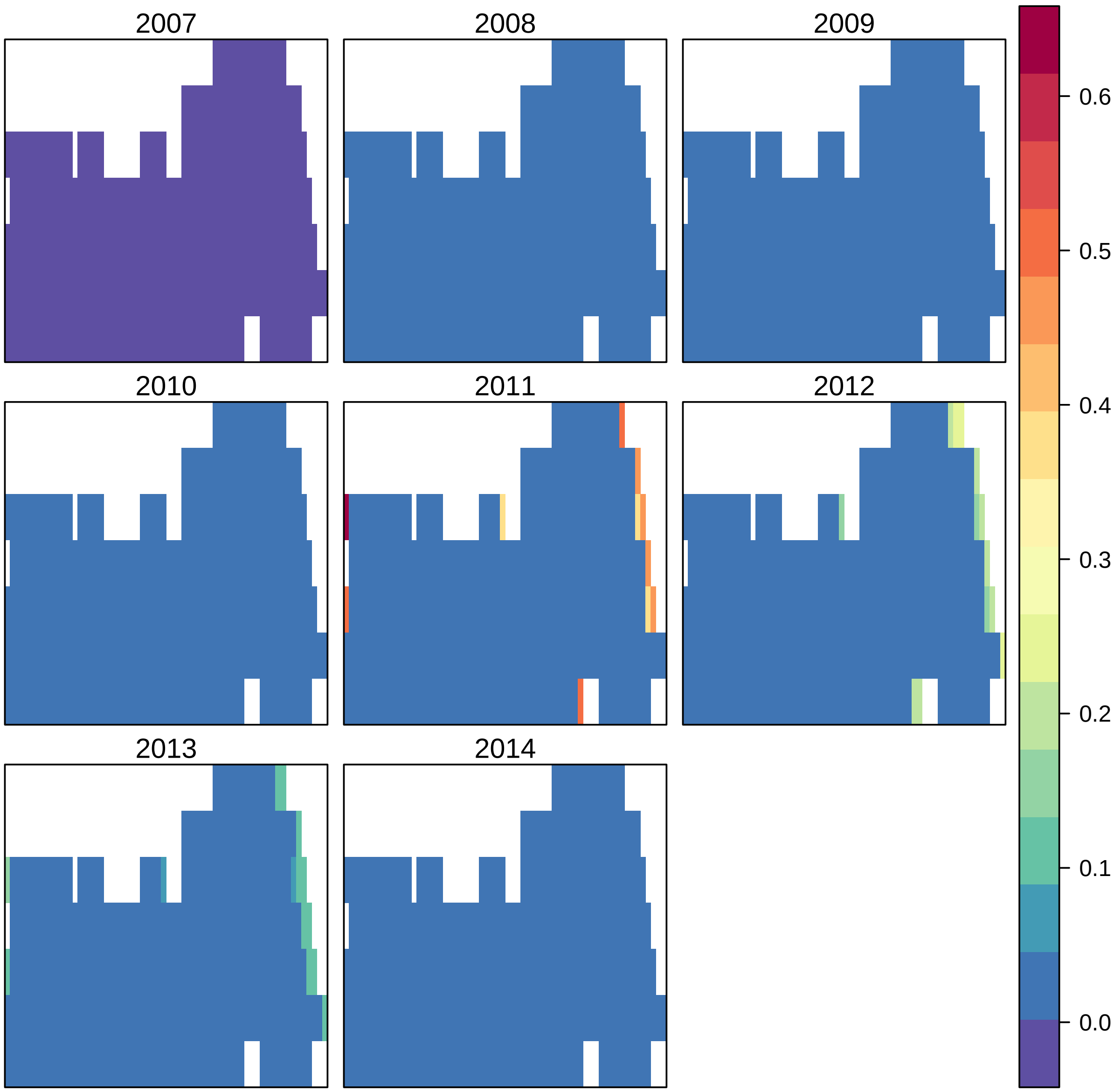

```
August.Ice.rs=stack(raster(t(August.Ice[[1]])[nr:1,]),
                    raster(t(August.Ice[[2]])[nr:1,]),
                    raster(t(August.Ice[[3]])[nr:1,]),
                    raster(t(August.Ice[[4]])[nr:1,]),
                    raster(t(August.Ice[[5]])[nr:1,]),
                    raster(t(August.Ice[[6]])[nr:1,]),
                    raster(t(August.Ice[[7]])[nr:1,]),
                    raster(t(August.Ice[[8]])[nr:1,])
                    )

rasterVis::levelplot(August.Ice.rs,
           col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
```

```
                          bias=1),
     names.attr=as.character(2007:2014), scales=list(draw=FALSE)
     )
```



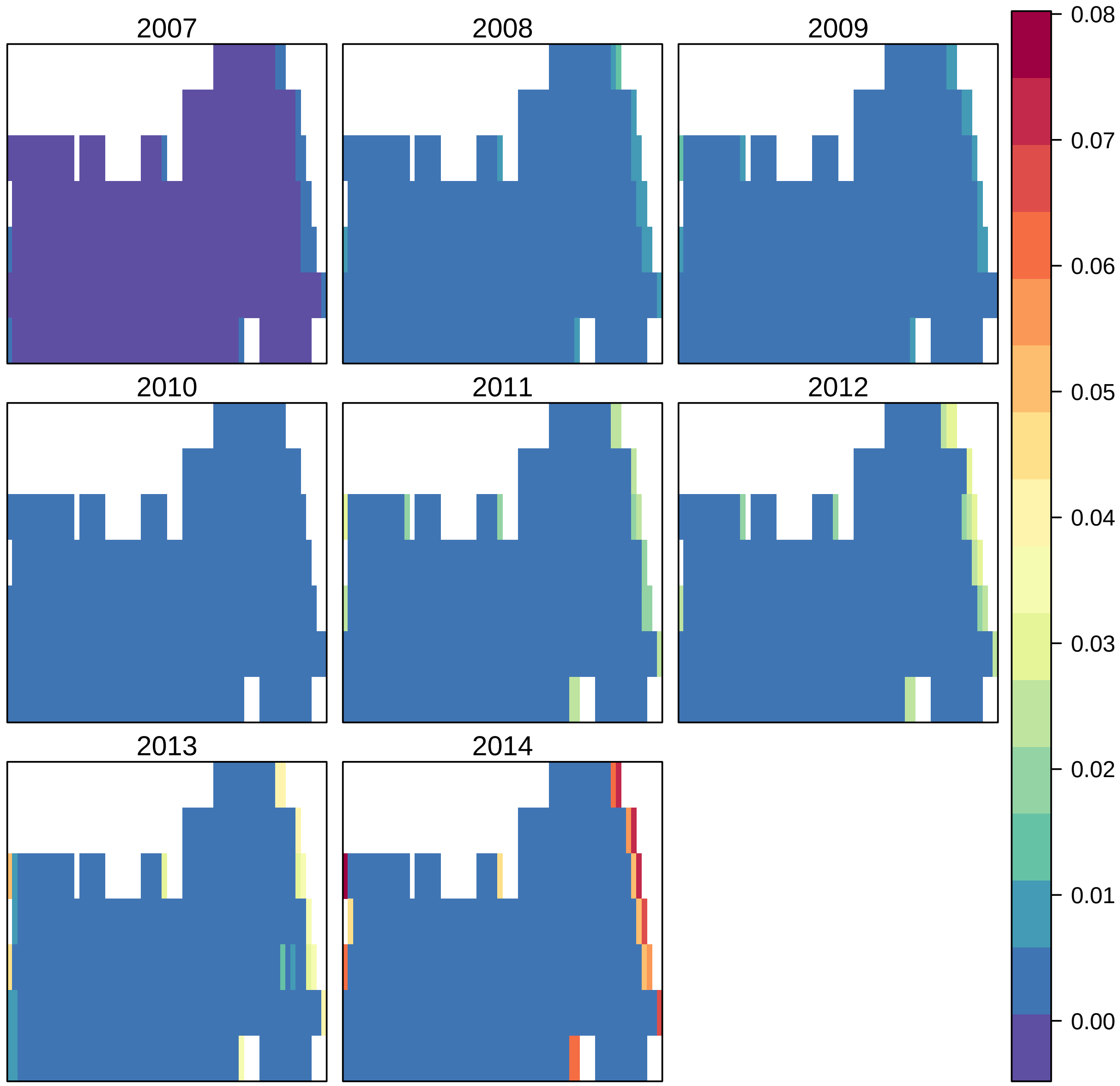

```
#########################################################################
### Figure 4 (model based)
#########################################################################

June.Ice.var=list()
August.Ice.var=list()

###
```

```
### June annual variance in Ice
###

for(i in 1:8){
    var.m=apply(model.output.l[[June.ind[i]]]$samples$fitted,2,var)
    var.m.Ice=var.m[seq(3,nr*nc*3,3)]
    June.Ice.var[[i]]=matrix(var.m.Ice*Boundary,nc,nr)
}
June.Ice.var.rs=stack(raster(t(June.Ice.var[[1]])[nr:1,]),
                  raster(t(June.Ice.var[[2]])[nr:1,]),
                  raster(t(June.Ice.var[[3]])[nr:1,]),
                  raster(t(June.Ice.var[[4]])[nr:1,]),
                  raster(t(June.Ice.var[[5]])[nr:1,]),
                  raster(t(June.Ice.var[[6]])[nr:1,]),
                  raster(t(June.Ice.var[[7]])[nr:1,]),
                  raster(t(June.Ice.var[[8]])[nr:1,])
                  )

rasterVis::levelplot(June.Ice.var.rs,
          col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                         bias=1),
          names.attr=as.character(2007:2014), scales=list(draw=FALSE)
          )
```

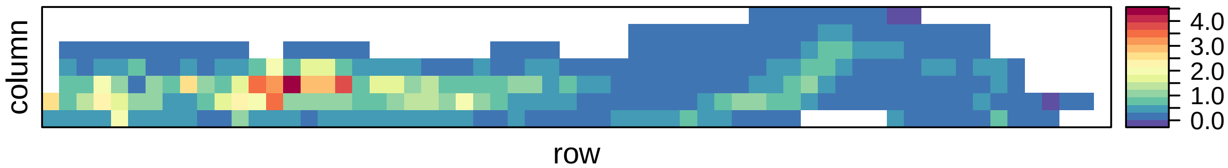



```
###
### August annual variance in Ice
###

for(i in 1:8){
    var.m=apply(model.output.l[[August.ind[i]]]$samples$fitted,2,var)
    var.m.Ice=var.m[seq(2,nr*nc*2,2)]
    August.Ice.var[[i]]=matrix(var.m.Ice*Boundary,nc,nr)
}
August.Ice.var.rs=stack(raster(t(August.Ice.var[[1]])[nr:1,]),
                        raster(t(August.Ice.var[[2]])[nr:1,]),
                        raster(t(August.Ice.var[[3]])[nr:1,]),
```

```
                    raster(t(August.Ice.var[[4]])[nr:1,]),
                    raster(t(August.Ice.var[[5]])[nr:1,]),
                    raster(t(August.Ice.var[[6]])[nr:1,]),
                    raster(t(August.Ice.var[[7]])[nr:1,]),
                    raster(t(August.Ice.var[[8]])[nr:1,])
                    )

rasterVis::levelplot(August.Ice.var.rs,
           col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                         bias=1),
           names.attr=as.character(2007:2014), scales=list(draw=FALSE)
           )
```

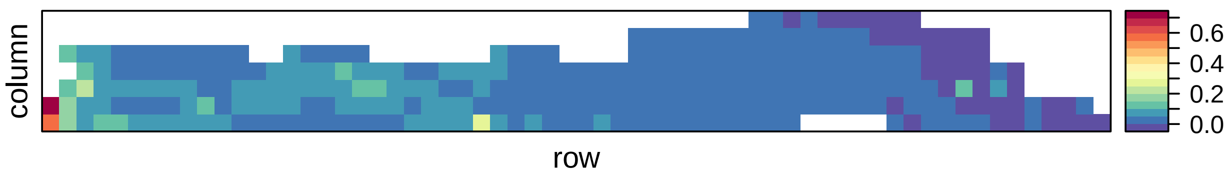


```r
#######################################################################
### Alternative Figure 4 (data based)
#######################################################################

Y.June.Ice.var=matrix(apply(cbind(Y.save[[1]][,3],
            Y.save[[3]][,3],
            Y.save[[5]][,3],
            Y.save[[7]][,3],
            Y.save[[9]][,3],
            Y.save[[11]][,3],
            Y.save[[13]][,3],
            Y.save[[15]][,3]),1,var),nc,nr)
rasterVis::levelplot(Y.June.Ice.var,
          col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                         bias=1),
          names.attr="June Ice Variance", scales=list(draw=FALSE)
          )
```



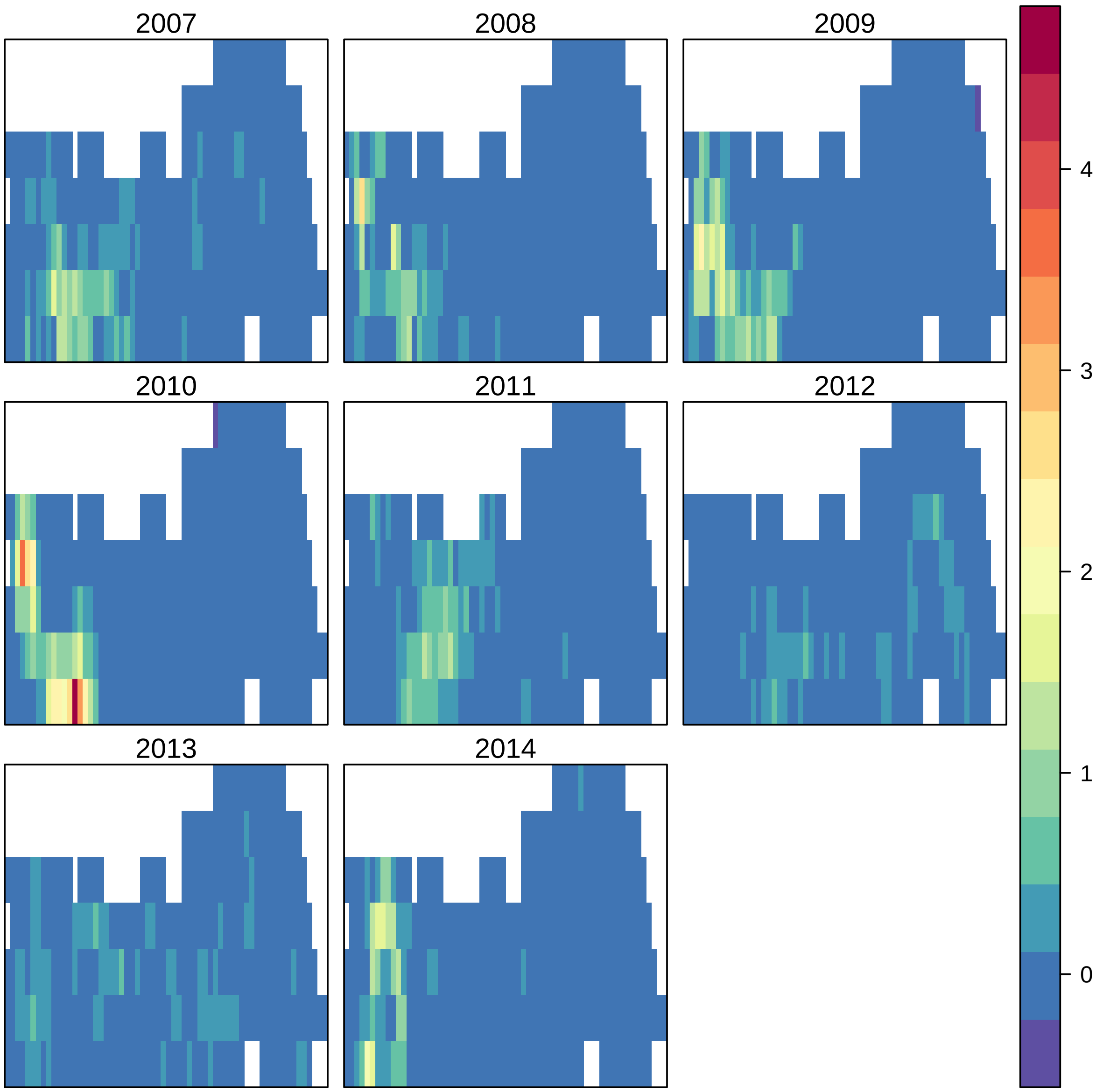

```
Y.August.Ice.var=matrix(apply(cbind(Y.save[[2]][,2],
            Y.save[[4]][,2],
            Y.save[[6]][,2],
            Y.save[[8]][,2],
            Y.save[[10]][,2],
            Y.save[[12]][,2],
            Y.save[[14]][,2],
            Y.save[[16]][,2]),1,var),nc,nr)
rasterVis::levelplot(Y.August.Ice.var,
          col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                         bias=1),
          names.attr="June Ice Variance", scales=list(draw=FALSE)
```

```
        )
```



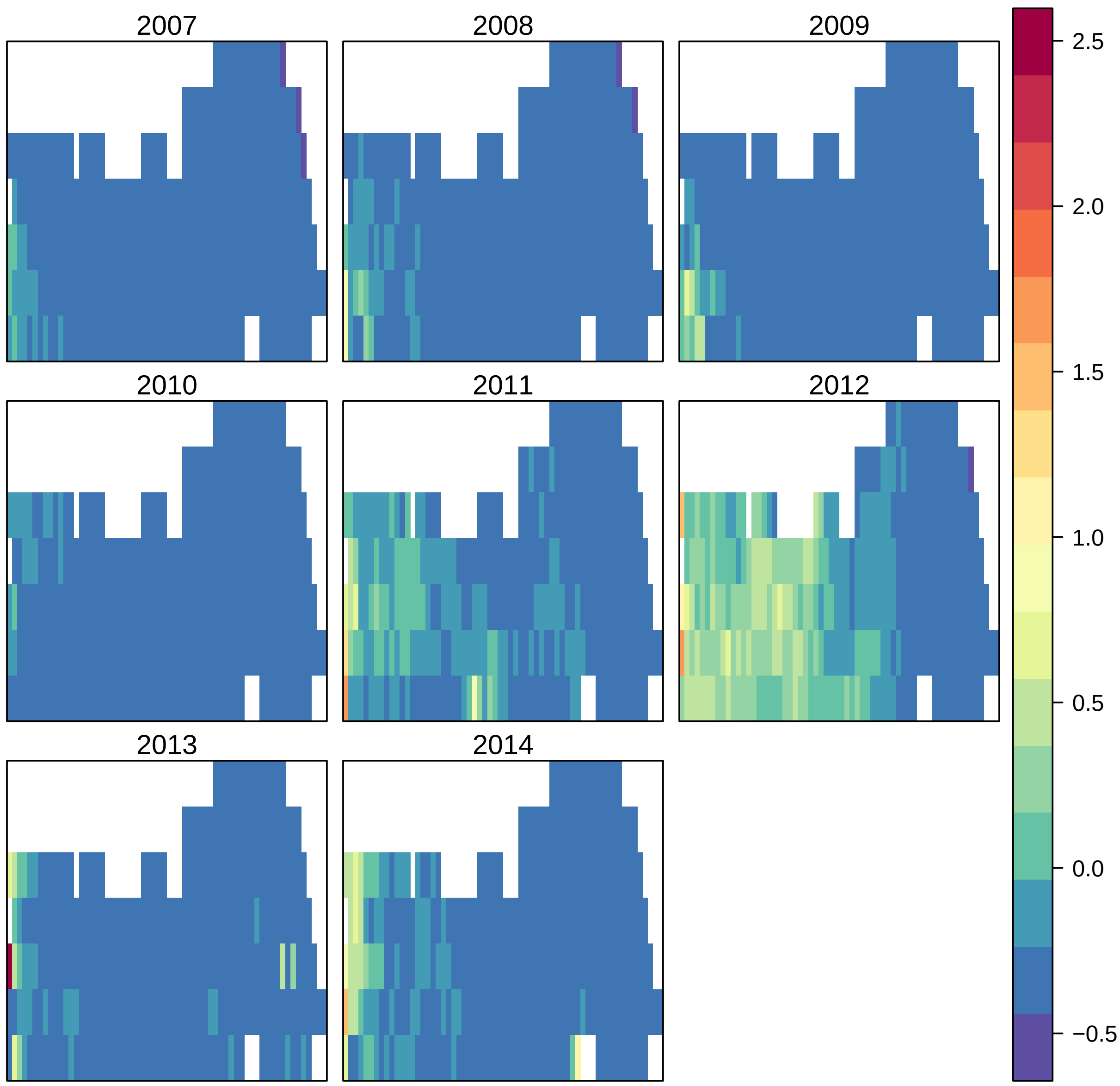

```
########################################################################
### Figure 5a is the same as top row in Figure 3
########################################################################

########################################################################
### Figure 5b
########################################################################

June.Adult=list()
August.Adult=list()
```

```r
###
### June annual Adult median
###

for(i in 1:8){
    med.m=apply(model.output.l[[June.ind[i]]]$samples$fitted,2,quantile,0.5)
    med.m.Adult=med.m[seq(1,nr*nc*3,3)]
    June.Adult[[i]]=matrix(med.m.Adult*Boundary,nc,nr)
}
June.Adult.rs=stack(raster(t(June.Adult[[1]])[nr:1,]),
                  raster(t(June.Adult[[2]])[nr:1,]),
                  raster(t(June.Adult[[3]])[nr:1,]),
                  raster(t(June.Adult[[4]])[nr:1,]),
                  raster(t(June.Adult[[5]])[nr:1,]),
                  raster(t(June.Adult[[6]])[nr:1,]),
                  raster(t(June.Adult[[7]])[nr:1,]),
                  raster(t(June.Adult[[8]])[nr:1,])
                  )

rasterVis::levelplot(June.Adult.rs,
          col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                         bias=1),
          names.attr=as.character(2007:2014), scales=list(draw=FALSE)
          )
```

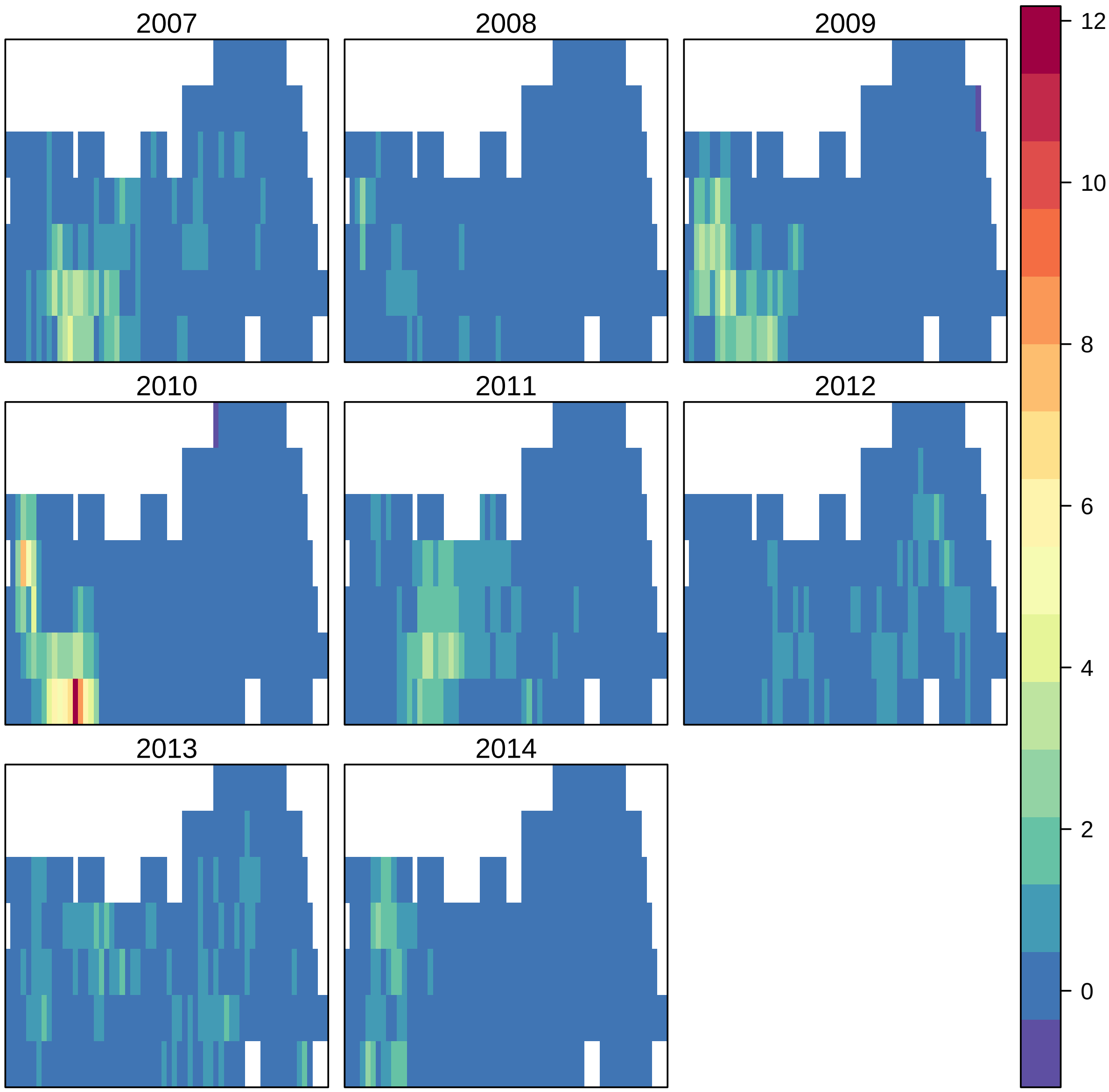



```
###
### August annual variance in Ice
###

for(i in 1:8){
    med.m=apply(model.output.l[[August.ind[i]]]$samples$fitted,2,quantile,0.5)
    med.m.Adult=med.m[seq(2,nr*nc*2,2)]
    August.Adult[[i]]=matrix(med.m.Adult*Boundary,nc,nr)
}
August.Adult.rs=stack(raster(t(August.Adult[[1]])[nr:1,]),
                      raster(t(August.Adult[[2]])[nr:1,]),
                      raster(t(August.Adult[[3]])[nr:1,]),
```

```
                    raster(t(August.Adult[[4]])[nr:1,]),
                    raster(t(August.Adult[[5]])[nr:1,]),
                    raster(t(August.Adult[[6]])[nr:1,]),
                    raster(t(August.Adult[[7]])[nr:1,]),
                    raster(t(August.Adult[[8]])[nr:1,])
                    )

rasterVis::levelplot(August.Adult.rs,
           col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                           bias=1),
           names.attr=as.character(2007:2014), scales=list(draw=FALSE)
           )
```

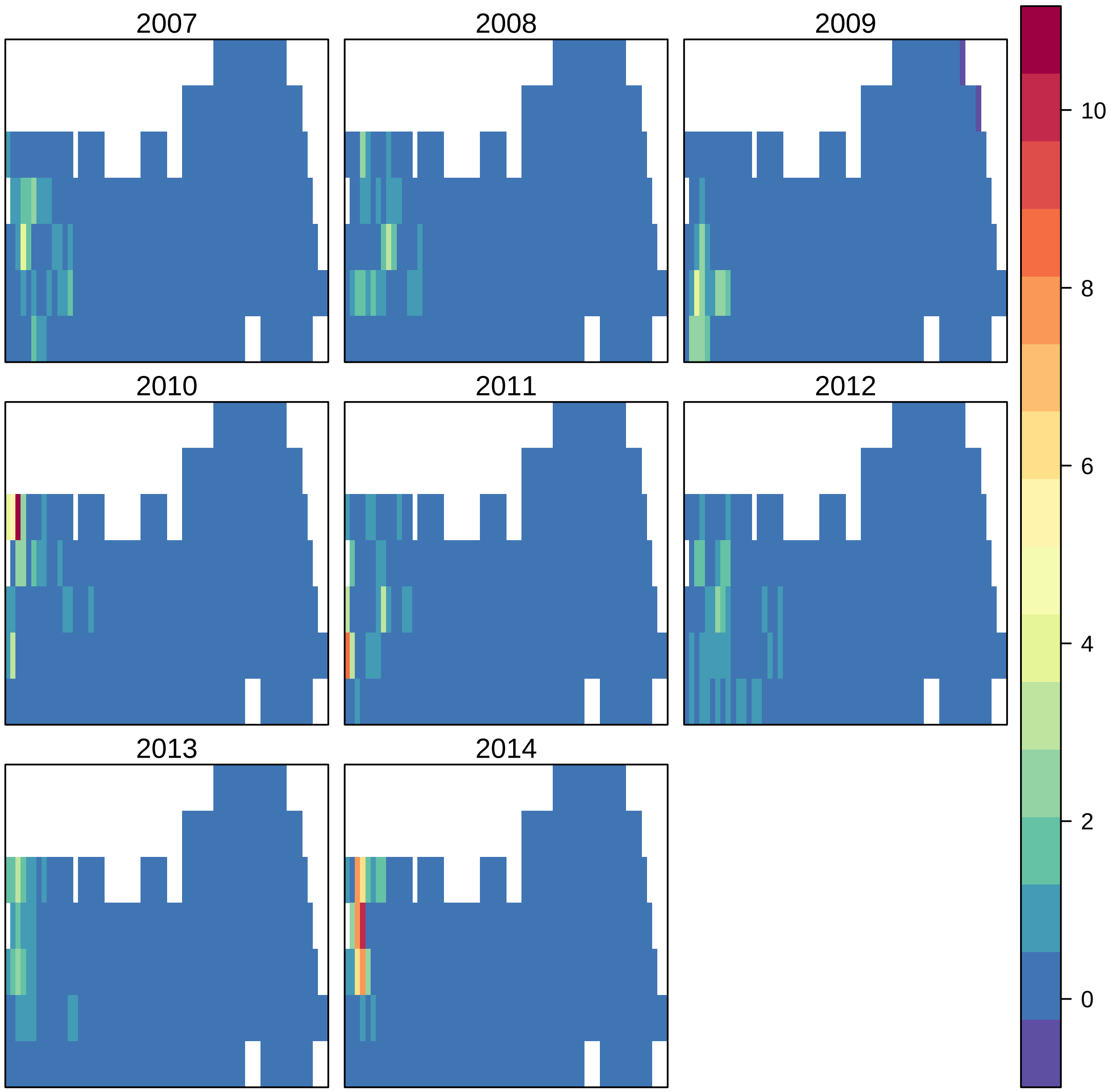


```r
#########################################################################
### Additional Figure 5 June Pup distribution median
#########################################################################

June.Pup=list()

###
### June annual Adult median
###

for(i in 1:8){
    med.m=apply(model.output.l[[June.ind[i]]]$samples$fitted,2,quantile,0.5)
    med.m.Pup=med.m[seq(2,nr*nc*3,3)]
    June.Pup[[i]]=matrix(med.m.Pup*Boundary,nc,nr)
}
June.Pup.rs=stack(raster(t(June.Pup[[1]])[nr:1,]),
                  raster(t(June.Pup[[2]])[nr:1,]),
                  raster(t(June.Pup[[3]])[nr:1,]),
                  raster(t(June.Pup[[4]])[nr:1,]),
                  raster(t(June.Pup[[5]])[nr:1,]),
                  raster(t(June.Pup[[6]])[nr:1,]),
                  raster(t(June.Pup[[7]])[nr:1,]),
                  raster(t(June.Pup[[8]])[nr:1,])
                  )

rasterVis::levelplot(June.Pup.rs,
                     col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                                    bias=1),
                     names.attr=as.character(2007:2014), scales=list(draw=FALSE)
                     )
```




```
#########################################################################
### Figure 6a is the same as bottom row in Figure 3
#########################################################################

#########################################################################
### Figure 6b
#########################################################################

for(i in 1:8){
    med.m=apply(model.output.l[[August.ind[i]]]$samples$fitted,2,quantile,0.5)
    med.m.Adult=med.m[seq(1,nr*nc*2,2)]
    August.Adult[[i]]=matrix(med.m.Adult*Boundary,nc,nr)
```

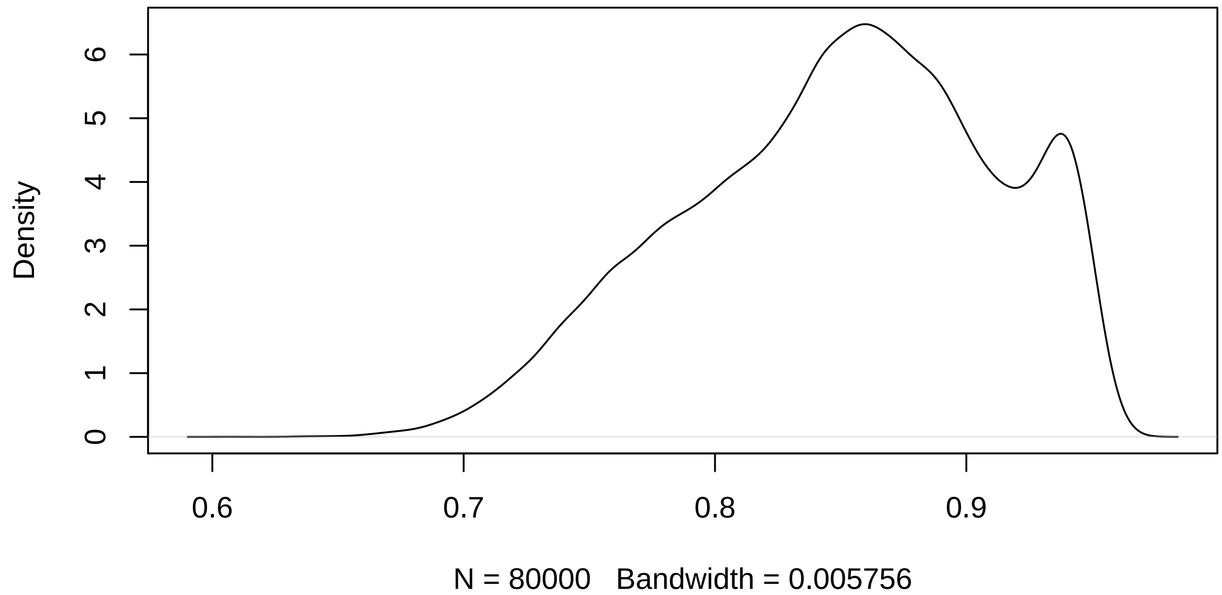
```
}
August.Adult.rs=stack(raster(t(August.Adult[[1]])[nr:1,]),
                      raster(t(August.Adult[[2]])[nr:1,]),
                      raster(t(August.Adult[[3]])[nr:1,]),
                      raster(t(August.Adult[[4]])[nr:1,]),
                      raster(t(August.Adult[[5]])[nr:1,]),
                      raster(t(August.Adult[[6]])[nr:1,]),
                      raster(t(August.Adult[[7]])[nr:1,]),
                      raster(t(August.Adult[[8]])[nr:1,])
                      )

rasterVis::levelplot(August.Adult.rs,
                     col.regions = colorRampPalette(rev(brewer.pal(11,'Spectral')),
                                                    bias=1),
                     names.attr=as.character(2007:2014), scales=list(draw=FALSE)
                     )
```

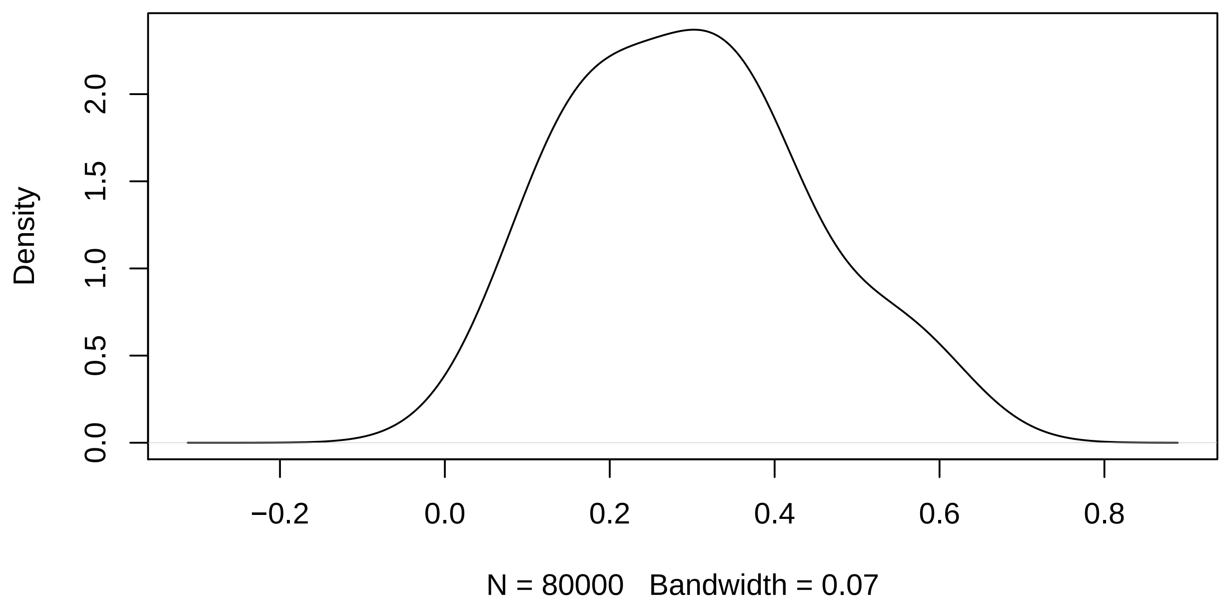



```
############################################################################
### Figure 8
############################################################################

###
### June - pupping
###

##
## Correlation between nonpups and pups
##
```

```
adult.pup.rho=c(model.output.l[[1]]$samples$Sigma[,1,2]/
                (sqrt(model.output.l[[1]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[1]]$samples$Sigma[,2,2])),

                model.output.l[[3]]$samples$Sigma[,1,2]/
                (sqrt(model.output.l[[3]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[3]]$samples$Sigma[,2,2])),

                model.output.l[[5]]$samples$Sigma[,1,2]/
                (sqrt(model.output.l[[5]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[5]]$samples$Sigma[,2,2])),

                model.output.l[[7]]$samples$Sigma[,1,2]/
                (sqrt(model.output.l[[7]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[7]]$samples$Sigma[,2,2])),

                model.output.l[[9]]$samples$Sigma[,1,2]/
                (sqrt(model.output.l[[9]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[9]]$samples$Sigma[,2,2])),

                model.output.l[[11]]$samples$Sigma[,1,2]/
                (sqrt(model.output.l[[11]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[11]]$samples$Sigma[,2,2])),

                model.output.l[[13]]$samples$Sigma[,1,2]/
                (sqrt(model.output.l[[13]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[13]]$samples$Sigma[,2,2])),

                model.output.l[[15]]$samples$Sigma[,1,2]/
                (sqrt(model.output.l[[15]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[15]]$samples$Sigma[,2,2])))

quantile(adult.pup.rho,c(0.025,0.5,.975))

##      2.5%       50%     97.5%
## 0.7251891 0.8546733 0.9472577

plot(density(adult.pup.rho),bw=.15,main="")

##
## Correlation between pups and ice
##

pup.ice.rho=c(model.output.l[[1]]$samples$Sigma[,2,3]/
              (sqrt(model.output.l[[1]]$samples$Sigma[,2,2])*
               sqrt(model.output.l[[1]]$samples$Sigma[,3,3])),

              model.output.l[[3]]$samples$Sigma[,2,3]/
```

```
              (sqrt(model.output.l[[3]]$samples$Sigma[,2,2])*
               sqrt(model.output.l[[3]]$samples$Sigma[,3,3])),

              model.output.l[[5]]$samples$Sigma[,2,3]/
              (sqrt(model.output.l[[5]]$samples$Sigma[,2,2])*
               sqrt(model.output.l[[5]]$samples$Sigma[,3,3])),

              model.output.l[[7]]$samples$Sigma[,2,3]/
              (sqrt(model.output.l[[7]]$samples$Sigma[,2,2])*
               sqrt(model.output.l[[7]]$samples$Sigma[,3,3])),

              model.output.l[[9]]$samples$Sigma[,2,3]/
              (sqrt(model.output.l[[9]]$samples$Sigma[,2,2])*
               sqrt(model.output.l[[9]]$samples$Sigma[,3,3])),

              model.output.l[[11]]$samples$Sigma[,2,3]/
              (sqrt(model.output.l[[11]]$samples$Sigma[,2,2])*
               sqrt(model.output.l[[11]]$samples$Sigma[,3,3])),

              model.output.l[[13]]$samples$Sigma[,2,3]/
              (sqrt(model.output.l[[13]]$samples$Sigma[,2,2])*
               sqrt(model.output.l[[13]]$samples$Sigma[,3,3])),

              model.output.l[[15]]$samples$Sigma[,2,3]/
              (sqrt(model.output.l[[15]]$samples$Sigma[,2,2])*
               sqrt(model.output.l[[15]]$samples$Sigma[,3,3])))

quantile(pup.ice.rho,c(0.025,0.5,.975))

##       2.5%        50%      97.5%
## 0.06726023 0.29431018 0.59006848

plot(density(pup.ice.rho,bw=.07),main="")
```

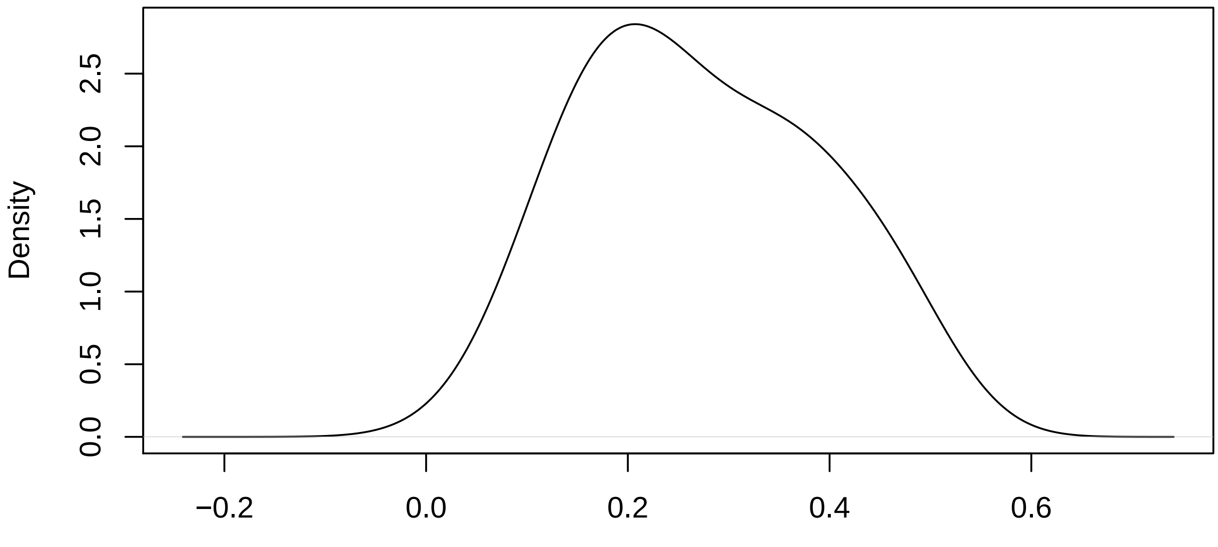


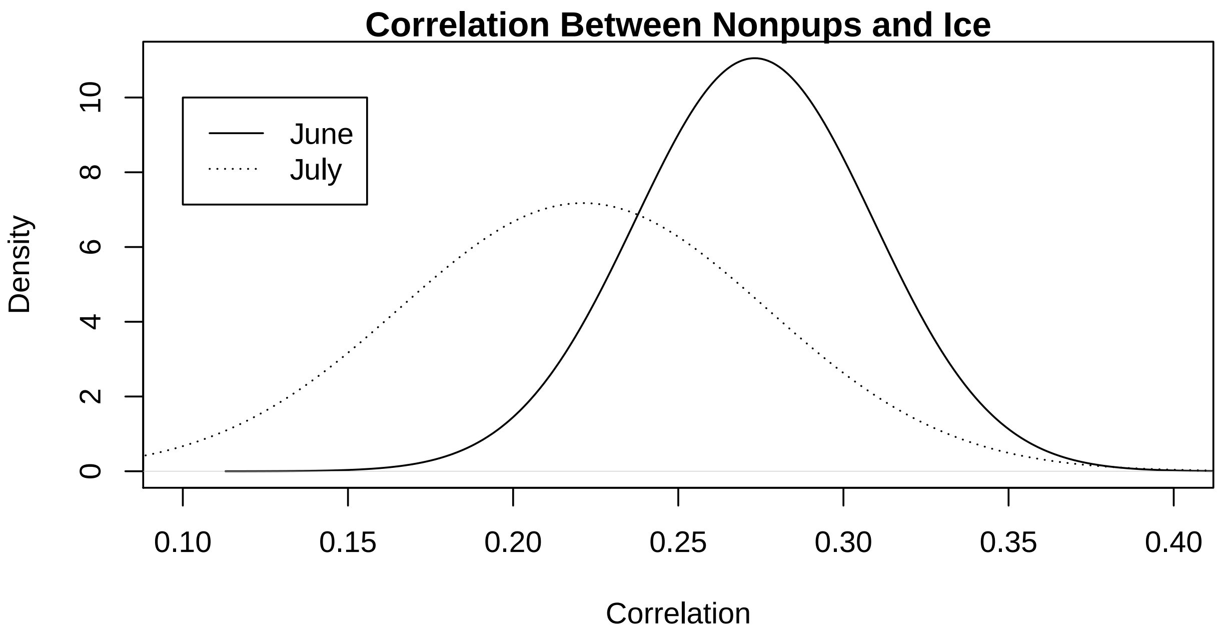



```
##
## Correlation between adults and ice
##

adult.ice.rho=c(model.output.l[[1]]$samples$Sigma[,1,3]/
                (sqrt(model.output.l[[1]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[1]]$samples$Sigma[,3,3])),

                model.output.l[[3]]$samples$Sigma[,1,3]/
                (sqrt(model.output.l[[3]]$samples$Sigma[,1,1])*
                 sqrt(model.output.l[[3]]$samples$Sigma[,3,3])),
```

```
                 model.output.l[[5]]$samples$Sigma[,1,3]/
                 (sqrt(model.output.l[[5]]$samples$Sigma[,1,1])*
                  sqrt(model.output.l[[5]]$samples$Sigma[,3,3])),

                 model.output.l[[7]]$samples$Sigma[,1,3]/
                 (sqrt(model.output.l[[7]]$samples$Sigma[,1,1])*
                  sqrt(model.output.l[[7]]$samples$Sigma[,3,3])),

                 model.output.l[[9]]$samples$Sigma[,1,3]/
                 (sqrt(model.output.l[[9]]$samples$Sigma[,1,1])*
                  sqrt(model.output.l[[9]]$samples$Sigma[,3,3])),

                 model.output.l[[11]]$samples$Sigma[,1,3]/
                 (sqrt(model.output.l[[11]]$samples$Sigma[,1,1])*
                  sqrt(model.output.l[[11]]$samples$Sigma[,3,3])),

                 model.output.l[[13]]$samples$Sigma[,1,3]/
                 (sqrt(model.output.l[[13]]$samples$Sigma[,1,1])*
                  sqrt(model.output.l[[13]]$samples$Sigma[,3,3])),

                 model.output.l[[15]]$samples$Sigma[,1,3]/
                 (sqrt(model.output.l[[15]]$samples$Sigma[,1,1])*
                  sqrt(model.output.l[[15]]$samples$Sigma[,3,3])))

quantile(adult.ice.rho,c(0.025,0.5,.975))

##       2.5%        50%      97.5%
## 0.06047303 0.26182234 0.50845806

plot(density(adult.ice.rho,bw=.04),main="")

########################################################################
### Figure 9
#######################################################################

d=3

##
## Nonpup variance
##

Sigma11=matrix(,10000,length(June.ind))
for(i in 1:length(June.ind)){
    Sigma11[,i]=model.output.l[[June.ind[i]]]$samples$Sigma[,1,1]
}

##
## Ice variance
```

```r
##

Sigma33=matrix(,10000,length(June.ind))
for(i in 1:length(June.ind)){
    Sigma33[,i]=model.output.l[[June.ind[i]]]$samples$Sigma[,3,3]
}

##
## Nonpup-Ice covariance
##

Sigma13=matrix(,10000,length(June.ind))
for(i in 1:length(June.ind)){
    Sigma13[,i]=model.output.l[[June.ind[i]]]$samples$Sigma[,1,3]
}

##
## Nonpup-Ice correlation
##

##June
Corr13S=apply(Sigma13/sqrt(Sigma11*Sigma33),1,mean)
quantile(Corr13S,c(0.025,0.5,0.975))

##      2.5%       50%     97.5%
## 0.2334928 0.2731717 0.3119894

## August
Sigma12A=matrix(,10000,length(June.ind))
for(i in 1:length(August.ind)){
    Sigma12A[,i]=model.output.l[[August.ind[i]]]$samples$Sigma[,1,2]
}
Sigma11A=matrix(,10000,length(August.ind))
Sigma22A=matrix(,10000,length(August.ind))
for(i in 1:length(August.ind)){
    Sigma11A[,i]=model.output.l[[August.ind[i]]]$samples$Sigma[,1,1]
    Sigma22A[,i]=model.output.l[[August.ind[i]]]$samples$Sigma[,2,2]
}

Corr12A=apply(Sigma12A/sqrt(Sigma11A*Sigma22A),1,mean)
quantile(Corr12A,c(0.025,0.5,0.975))

##      2.5%       50%     97.5%
## 0.1737399 0.2210861 0.2689422

plot(density(Corr13S,bw=.03),main="Correlation Between Nonpups and Ice",
     xlim=c(0.1,.4),xlab="Correlation")
lines(density(Corr12A,bw=.05),lty=3)
legend(.1,10,legend=c("June","July"),lty=c(1,3))
```






```
##########################################################################
### Table
##########################################################################

model.output.l[[1]]$summary.results

##                              Median    2.5%   97.5% n.sample % accept
## Category 2 - (Intercept) -0.0386 -0.0644 -0.0085    10000    100.0
## Category 2 - X           -0.1034 -0.1724 -0.0496    10000    100.0
## Category 3 - (Intercept)  0.1852  0.1206  0.2590    10000    100.0
## Category 3 - X           -0.2608 -0.4244 -0.1361    10000    100.0
## Category 4 - (Intercept) -0.1165 -0.1313 -0.1021    10000    100.0
```

```
## Category 4 - X          -0.0738 -0.1070 -0.0400    10000    100.0
## nu2.1                    0.0015  0.0009  0.0025    10000    100.0
## nu2.2                    0.0040  0.0017  0.0089    10000    100.0
## nu2.3                    0.0011  0.0007  0.0017    10000    100.0
## Sigma11                  0.0892  0.0759  0.1058    10000    100.0
## Sigma22                  0.5251  0.4480  0.6198    10000    100.0
## Sigma33                  0.0228  0.0192  0.0272    10000    100.0
## rho                      0.9706  0.9390  0.9899    10000     45.2
##                         n.effective Geweke.diag
## Category 2 - (Intercept)       136.0         1.7
## Category 2 - X                  93.6         2.1
## Category 3 - (Intercept)       107.9         1.5
## Category 3 - X                  81.7         1.9
## Category 4 - (Intercept)       713.8         1.4
## Category 4 - X                 957.9         1.0
## nu2.1                         6792.5         1.3
## nu2.2                         1543.5         0.5
## nu2.3                         8763.6        -0.8
## Sigma11                       3514.0        -0.9
## Sigma22                       2007.9        -1.6
## Sigma33                       8593.8        -0.7
## rho                           7199.4         0.7
```

```
t=1:16
Year=rep(2007:2014,each=2)
Season=rep(c("June","August"),length.out=14)

## Nonpups
beta0=matrix(NA,length(t),3)
beta1=matrix(NA,length(t),3)
for(i in t){
    beta0[i,1]=round(model.output.l[[i]]$summary.results[1,1],2)
    beta0[i,2]=round(model.output.l[[i]]$summary.results[1,2],2)
    beta0[i,3]=round(model.output.l[[i]]$summary.results[1,3],2)
    beta1[i,1]=round(model.output.l[[i]]$summary.results[2,1],2)
    beta1[i,2]=round(model.output.l[[i]]$summary.results[2,2],2)
    beta1[i,3]=round(model.output.l[[i]]$summary.results[2,3],2)
}

## Ice
beta0Ice=matrix(NA,length(t),3)
beta1Ice=matrix(NA,length(t),3)
for(i in t){
    beta0Ice[i,1]=round(model.output.l[[i]]$summary.results[3,1],2)
    beta0Ice[i,2]=round(model.output.l[[i]]$summary.results[3,2],2)
    beta0Ice[i,3]=round(model.output.l[[i]]$summary.results[3,3],2)
    beta1Ice[i,1]=round(model.output.l[[i]]$summary.results[4,1],2)
    beta1Ice[i,2]=round(model.output.l[[i]]$summary.results[4,2],2)
```

```
    beta1Ice[i,3]=round(model.output.l[[i]]$summary.results[4,3],2)
}

## Pups
beta0Pup=matrix(NA,length(t),3)
beta1Pup=matrix(NA,length(t),3)
for(i in seq(1,16,2)){
    beta0Pup[i,1]=round(model.output.l[[i]]$summary.results[5,1],2)
    beta0Pup[i,2]=round(model.output.l[[i]]$summary.results[5,2],2)
    beta0Pup[i,3]=round(model.output.l[[i]]$summary.results[5,3],2)
    beta1Pup[i,1]=round(model.output.l[[i]]$summary.results[6,1],2)
    beta1Pup[i,2]=round(model.output.l[[i]]$summary.results[6,2],2)
    beta1Pup[i,3]=round(model.output.l[[i]]$summary.results[6,3],2)
}

C12=matrix(NA,16,3)
C13=matrix(NA,16,3)
Corr13.June.save=matrix(NA,n.iter,8)
Corr13.August.save=matrix(NA,n.iter,8)
C23=matrix(NA,16,3)
i=1
counter=1
for(i in seq(1,16,2)){
    Sigma12=model.output.l[[i]]$samples$Sigma[,1,2]
    Sigma13=model.output.l[[i]]$samples$Sigma[,1,3]
    Sigma23=model.output.l[[i]]$samples$Sigma[,2,3]
    Sigma11=model.output.l[[i]]$samples$Sigma[,1,1]
    Sigma22=model.output.l[[i]]$samples$Sigma[,2,2]
    Sigma33=model.output.l[[i]]$samples$Sigma[,3,3]
    Corr12=Sigma12/sqrt(Sigma11*Sigma22)
    Corr13=Sigma13/sqrt(Sigma11*Sigma33)
    Corr13.June.save[,counter]=Corr13
    Corr23=Sigma23/sqrt(Sigma22*Sigma33)
    C12[i,]=quantile(Corr12,c(0.5,0.025,0.975))
    C13[i,]=quantile(Corr13,c(0.5,0.025,0.975))
    C23[i,]=quantile(Corr23,c(0.5,0.025,0.975))
    counter=counter+1
}

counter=1
for(i in seq(2,16,2)){
    Sigma13=model.output.l[[i]]$samples$Sigma[,1,2]
    Sigma11=model.output.l[[i]]$samples$Sigma[,1,1]
    Sigma33=model.output.l[[i]]$samples$Sigma[,2,2]
    Corr13=Sigma13/sqrt(Sigma11*Sigma33)
    Corr13.August.save[,counter]=Corr13
    C13[i,]=quantile(Corr13,c(0.5,0.025,0.975))
```

```
    counter=counter+1
}

###
### rho
###

rho=matrix(NA,16,3)

for(i in seq(1,16,2)){
    rho[i,1]=round(model.output.l[[i]]$summary.results[13,1],2)
    rho[i,2]=round(model.output.l[[i]]$summary.results[13,2],2)
    rho[i,3]=round(model.output.l[[i]]$summary.results[13,3],2)
}
for(i in seq(2,16,2)){
    rho[i,1]=round(model.output.l[[i]]$summary.results[9,1],2)
    rho[i,2]=round(model.output.l[[i]]$summary.results[9,2],2)
    rho[i,3]=round(model.output.l[[i]]$summary.results[9,3],2)
}
TableData=data.frame(beta0,beta1,beta0Pup,beta1Pup,beta0Ice,beta1Ice,C12,C13,C23)

TableData

##       X1    X2    X3  X1.1  X2.1  X3.1  X1.2  X2.2  X3.2  X1.3  X2.3  X3.3
## 1  -0.04 -0.06 -0.01 -0.10 -0.17 -0.05 -0.12 -0.13 -0.10 -0.07 -0.11 -0.04
## 2  -0.07 -0.12 -0.02 -0.06 -0.16  0.05    NA    NA    NA    NA    NA    NA
## 3  -0.07 -0.11 -0.02 -0.04 -0.14  0.06 -0.09 -0.11 -0.06 -0.11 -0.18 -0.05
## 4  -0.04 -0.10  0.01 -0.01 -0.14  0.12    NA    NA    NA    NA    NA    NA
## 5  -0.04 -0.08  0.00 -0.09 -0.20  0.00 -0.22 -0.25 -0.20 -0.08 -0.14 -0.01
## 6  -0.10 -0.17 -0.03 -0.11 -0.25  0.03    NA    NA    NA    NA    NA    NA
## 7  -0.03 -0.08  0.02 -0.16 -0.27 -0.04 -0.30 -0.33 -0.28 -0.10 -0.17 -0.04
## 8   0.08  0.00  0.16  0.13 -0.09  0.35    NA    NA    NA    NA    NA    NA
## 9  -0.02 -0.05  0.00 -0.05 -0.11  0.00  0.62  0.52  0.73 -0.33 -0.61 -0.03
## 10 -0.03 -0.10  0.05 -0.03 -0.21  0.15    NA    NA    NA    NA    NA    NA
## 11 -0.07 -0.09 -0.06 -0.07 -0.11 -0.02  0.90  0.84  0.97 -0.26 -0.41 -0.10
## 12 -0.03 -0.07  0.01 -0.07 -0.17  0.02    NA    NA    NA    NA    NA    NA
## 13 -0.05 -0.07 -0.03 -0.05 -0.09  0.00  0.28  0.23  0.33 -0.19 -0.30 -0.08
## 14  0.04 -0.04  0.12  0.08 -0.07  0.22    NA    NA    NA    NA    NA    NA
## 15 -0.06 -0.09 -0.02 -0.05 -0.13  0.04 -0.19 -0.21 -0.16  0.01 -0.05  0.07
## 16  0.11  0.00  0.23  0.16 -0.11  0.44    NA    NA    NA    NA    NA    NA
##     X1.4  X2.4  X3.4  X1.5  X2.5  X3.5      X1.6      X2.6      X3.6
## 1   0.19  0.12  0.26 -0.26 -0.42 -0.14 0.8841746 0.8518149 0.9101001
## 2  -0.40 -0.41 -0.39 -0.03 -0.05 -0.01        NA        NA        NA
## 3  -0.11 -0.15 -0.07 -0.06 -0.16  0.05 0.8506112 0.8003248 0.8902994
## 4  -0.35 -0.37 -0.34 -0.04 -0.07  0.00        NA        NA        NA
## 5   0.06 -0.02  0.14 -0.25 -0.44 -0.08 0.9087655 0.8735323 0.9345168
## 6  -0.38 -0.40 -0.37 -0.04 -0.07  0.00        NA        NA        NA
## 7   0.09 -0.01  0.21 -0.44 -0.68 -0.18 0.9392315 0.9161435 0.9563311
```

```
## 8  -0.38 -0.39 -0.37  0.01 -0.02  0.03        NA        NA        NA
## 9   0.18  0.13  0.25 -0.15 -0.30 -0.02 0.8231593 0.7732297 0.8638064
## 10 -0.23 -0.25 -0.21 -0.04 -0.09  0.01        NA        NA        NA
## 11  0.05  0.01  0.09 -0.12 -0.22 -0.03 0.7534185 0.6890493 0.8079255
## 12 -0.06 -0.08 -0.03 -0.07 -0.13 -0.01        NA        NA        NA
## 13  0.14  0.09  0.20 -0.11 -0.23  0.02 0.8560566 0.8183810 0.8871731
## 14 -0.34 -0.37 -0.31 -0.01 -0.08  0.06        NA        NA        NA
## 15 -0.02 -0.06  0.03 -0.07 -0.17  0.04 0.7905696 0.7340089 0.8398263
## 16 -0.26 -0.30 -0.22 -0.04 -0.13  0.05        NA        NA        NA
##         X1.7         X2.7      X3.7      X1.8       X2.8      X3.8
## 1  0.3486575  0.237006796 0.4510507 0.3663853 0.25686302 0.4666432
## 2  0.2514061  0.132642651 0.3677285        NA         NA        NA
## 3  0.1243324  0.007417758 0.2398723 0.1631213 0.04212764 0.2771393
## 4  0.2833975  0.163387482 0.3900926        NA         NA        NA
## 5  0.2731999  0.153767044 0.3891237 0.2715698 0.15362368 0.3855241
## 6  0.4106932  0.272325378 0.5392633        NA         NA        NA
## 7  0.4672102  0.368714677 0.5579181 0.5560728 0.47069424 0.6322171
## 8  0.2829332  0.164447605 0.3940618        NA         NA        NA
## 9  0.1922728  0.078698583 0.3041736 0.1505008 0.03789414 0.2620512
## 10 0.1825813  0.066002117 0.2976884        NA         NA        NA
## 11 0.3834099  0.277198052 0.4823654 0.3485574 0.23896292 0.4527443
## 12 0.1167641 -0.012949408 0.2370553        NA         NA        NA
## 13 0.2343772  0.115203383 0.3489375 0.1586104 0.03882171 0.2796465
## 14 0.1029735 -0.118057213 0.3245917        NA         NA        NA
## 15 0.1632440  0.045194815 0.2792822 0.3591061 0.25077019 0.4598817
## 16 0.1438696  0.027768626 0.2563290        NA         NA        NA

## xtable(TableData)
```