

# Libraries

```
In [ ]: library("foodweb") # https://r-forge.r-project.org/projects/foodweb/
library("bipartite")
library("viridis")
library("ape")
library("picante")
library("phytools")
library("pgirmess")
```

# Data

```
In [ ]: # These lists contain host-parasite data by habitat type

# These data are used in "1. Interaction niche of parasites"
ha<-readRDS("ha.RDS") # rows: hosts; columns: parasites
names(ha)<-c("upland", "settlement", "forest", "lowland")

# These data are used in "2. Interaction niche of the hosts"
ha.t<-lapply(ha, t) # rows: parasites; columns: hosts
names(ha.t)<-c("upland", "settlement", "forest", "lowland")
```

## 1. Interaction niche of parasites (hosts are ordered)

### 1.1 Intervality: optimal

Optimal refers to the maximum intervality. We can only be to sure to have found the maximum intervality with the optimisation algorithm when  $G_{min} = 0$ . That's why we refer to optimal intervality in the main text.

```
In [ ]: # Calculate the optimal intervality
Ir.max<-lapply(ha, function(x) Interval(x))
# Extract Gmin
Ir.max.index<-unlist(sapply(Ir.max,function(x) x[1]))
```

### 1.2 Intervality: phylogeny

```
In [ ]: # Load host phylogenetic tree
r.fulltree<-read.nexus("hostphylo.con.nexus")
r.fulltree<-root(r.fulltree, 1, r = TRUE)

# Prune the tree by habitat type
r.phytree<-vector("list", length = 4)
for(i in seq_along(r.fulltree)) {
  r.phytree[[i]]<-prune.sample(samp=t(ha[[i]]),phylo=r.fulltree)
}
names(r.phytree)<-names(ha)

# Constrain rotation by host phylogeny and calculate intervality
```

```

# We edited the function Interval in foodweb package to limit the host
# permutation by phylogeny ordination when minimising the number of gaps.
# Acceptance probability and calculation of number of gaps are identical
# to Interval function in foodweb package.
rr.phy<-vector("list", 4)
for(i in seq_along(r.phytree)){
  rr.phy[[i]]<-I.rotate(ha[[i]],r.phytree[[i]])
}
names(rr.phy)<-names(ha)

# Extract phylogenetically-constrained Gmin
Ir.phy.index.rot<-unlist(lapply(rr.phy, `[`, 1))

```

## 1.3 Intervality: overlap in hosts' parasite community

```

In [ ]: # Load host dendrograms by overlap in parasite communities
r.dends<-readRDS("r.dends.RDS")

# Constrain host rotation by overlap in parasite communities
rr.use<-vector("list", 4) # rr: rodent rotated
for(i in seq_along(r.dends)){
  rr.use[[i]]<-I.rotate(ha[[i]],r.dends[[i]])
}
names(rr.use)<-names(ha)

# Extract overlap in parasite community-constrained Gmin
Ir.use.index.rot<-unlist(lapply(rr.use, `[`, 1))

```

## 1.4 Comparison within habitats: null communities

```

In [ ]: # Create 1,000 null communities for each habitat type
ha.sw<-lapply(ha, function(x) swap.web(1000, x)) # parasites in columns
names(ha.sw)<-c("upland", "settlement", "forest", "lowland")

# Calculate intervality of the null communities
Ir.max.sw<-vector('list', length(ha.sw))
for(i in seq_along(ha.sw)){
  for(j in seq_along(ha.sw[[i]])){
    Ir.max.sw[[i]][j]<-Interval(ha.sw[[i]][j])
  }
}
names(Ir.max.sw)<-c("upland", "settlement", "forest", "lowland")

# Extract Gmin of the null communities
Ir.max.sw.index<-vector("list", length = 4)
for (i in seq_along(Ir.max.sw)){
  Ir.max.sw.index[[i]]<-unlist(lapply(Ir.max.sw[[i]], `[`, 1))
}
names(Ir.max.sw.index)<-c("upland", "settlement", "forest", "lowland")

# Results null communities
Ir.results.rot<-cbind(Ir.max.index,
  Ir.phy.index.rot,
  Ir.use.index.rot)
rownames(Ir.results.rot)<-c("upland", "settlement", "forest", "lowland")

# Plot results by habitat type
for (i in seq_along(Ir.max.sw.index)) {
  hist(Ir.max.sw.index[[i]],
    xlim=c(min(Ir.max.sw.index[[i]])-5, max(Ir.max.sw.index[[i]]+40),

```

```

    main=rownames(Ir.results.rot)[[i]],
    las=1, col="grey95", breaks = 10)
abline(v=Ir.results.rot[i,], col = viridis(3, alpha=1, option = "D"),
       lwd=5)
box()

}

# Calculate the z-score
z<-vector("list", length = 4)
for (i in seq_along(Ir.max.sw.index)) {
  for(j in seq_along(Ir.max.sw.index[[i]])){
    z[[i]][[j]]<-(Ir.results.rot[i,j]-mean(Ir.max.sw.index[[i]]))/sd(Ir.max.sw.index[[i]]
  }
}

```

## 1.5 Comparison between habitats

```

In [ ]: # Load the 50 replicates of each dataset
h5.rt98<-readRDS("h5.rt98.RDS")

# Size of the replicates
h5.rt98.size<-vector("list", length = length(ha))
h5.rt98.size<-lapply(h5.rt98.size, function(x) vector("list", length = 50))
for(i in seq_along(h5.rt98)){
  for(j in seq_along(h5.rt98[[i]])) {
    h5.rt98.size[[i]][[j]]<-dim(h5.rt98[[i]][[j]])[1]*dim(h5.rt98[[i]][[j]])[2]
  }
}
h5.rt98.size<-lapply(h5.rt98.size, unlist)
names(h5.rt98.size)<-c("upland", "settlement", "forest", "lowland")

# Replicates by habitat type
rep98.land<-c(rep("up",50),rep("se",50),rep("fo",50),rep("lo",50))

```

### 1.5.1 Intervality: optimal

```

In [ ]: # Calculate the optimal intervality
rep98.I<-vector("list", length = length(ha))
for (i in seq_along(h5.rt98)) {
  for (j in seq_along(h5.rt98[[i]])) {
    rep98.I[[i]][[j]]<-Interval(h5.rt98[[i]][[j]])
  }
}
names(rep98.I)<-c("upland", "settlement", "forest", "lowland")

# Extract Gmin
rep98I.max.index<-vector("list", length = length(ha))
for(i in seq_along(rep98.I)){
  rep98I.max.index[[i]]<-unlist(sapply(rep98.I[[i]],function(x) x[1]))
}
names(rep98I.max.index)<-c("upland", "settlement", "forest", "lowland")

# Calculate unweighted connectance
rep98I.mat<-vector("list", length = length(ha))
for(i in seq_along(rep98.I)){
  rep98I.mat[[i]]<-sapply(rep98.I[[i]],function(x) x[2])
}
names(rep98I.mat)<-c("upland", "settlement", "forest", "lowland")

rep98.uc<-vector("list", length = length(ha)) # uc: unweighted connectance

```

```

for(i in seq_along(h5.rt98)){
  for(j in seq_along(h5.rt98[[i]])){
    rep98.uc[[i]][[j]]<-networklevel(rep98I.mat[[i]][[j]], index = "connectance")
  }
}
names(rep98.uc)<-c("upland", "settlement", "forest", "lowland")

# Statistics
mod1<-lm(unlist(rep98I.max.index)~unlist(h5.rt98.size)+unlist(rep98.uc))
TukeyHSD(aov(lm(mod1$residuals~rep98.land)))

Ir.max.mod1.uc<-data.frame(mod1$residuals,rep98.land)
Ir.max.mod1.uc$or<-factor(Ir.max.mod1.uc$rep98.land, levels=c("fo","up","lo","se"))
Ir.max.mod1.uc<-Ir.max.mod1.uc[order(Ir.max.mod1.uc$or),]

```

## 1.5.2 Intervality: phylogeny

```

In [ ]: # Prune the dendrograms by replicate
rep98.phy<-vector("list", length = 4)
rep98.phy<-lapply(rep98.phy, function(x) vector("list", length = 50))

for(i in seq_along(h5.rt98)) {
  for(j in seq_along(h5.rt98[[i]])){
    rep98.phy[[i]][[j]]<-prune.sample(samp=t(h5.rt98[[i]][[j]]),phylo=r.fulltree)
  }
}
names(rep98.phy)<-c("upland", "settlement", "forest", "lowland")

# Constraint rotation by host phylogeny and calculate intervality.
# We edited the function Interval in foodweb package to limit the row
# rotation by dendrogram ordination to minimise the number of gaps.
rep98.I.phy<-vector("list", 4)
rep98.I.phy<-lapply(rep98.I.phy, function(x) vector("list", length = 50))
for(i in seq_along(rep98.phy)){
  for(j in seq_along(rep98.phy[[i]])){
    rep98.I.phy[[i]][[j]]<-I.rotate(h5.rt98[[i]][[j]],rep98.phy[[i]][[j]])
  }
}
names(rep98.I.phy)<-c("upland", "settlement", "forest", "lowland")

# Extract taxonomically-constrained Gmin
rep98I.phy.index<-vector("list", length = length(ha))
for(i in seq_along(rep98.I.phy)){
  rep98I.phy.index[[i]]<-unlist(sapply(rep98.I.phy[[i]],function(x) x[1]))
}
names(rep98I.mat.phy)<-c("upland", "settlement", "forest", "lowland")

# Calculate unweighted connectance
rep98I.mat.phy<-vector("list", length = length(ha))
for(i in seq_along(rep98.I.phy)){
  rep98I.mat.phy[[i]]<-sapply(rep98.I.phy[[i]],function(x) x[2])
}
names(rep98I.phy.index)<-c("upland", "settlement", "forest", "lowland")

rep98.uc.phy<-vector("list", length = length(ha)) # uc: unweighted connectance
for(i in seq_along(rep98I.mat.phy)){
  for(j in seq_along(rep98I.mat.phy[[i]])){
    rep98.uc.phy[[i]][[j]]<-networklevel(rep98I.mat.phy[[i]][[j]], index = "connect
  }
}
names(rep98.uc.phy)<-c("upland", "settlement", "forest", "lowland")

# Statistics

```

```
mod1<-lm(unlist(rep98I.phy.index)~unlist(h5.rt98.size)+unlist(rep98.uc.phy))
TukeyHSD(aov(lm(mod1$residuals~rep98.land)))

Ir.phy.mod1.uc<-data.frame(mod1$residuals,rep98.land)
Ir.phy.mod1.uc$or<-factor(Ir.phy.mod1.uc$rep98.land, levels=c("fo","up","lo","se"))
Ir.phy.mod1.uc<-Ir.phy.mod1.uc[order(Ir.phy.mod1.uc$or),]
```

### 1.5.3 Intervality: overlap in hosts' parasite community

```
In [ ]: # Prune the dendrograms by replicate
rep98.use<-vector("list", length = 4)
rep98.use<-lapply(rep98.use, function(x) vector("list", length = 50))
for(i in seq_along(h5.rt98)) {
  for(j in seq_along(h5.rt98[[i]])){
    rep98.use[[i]][[j]]<-prune.sample(samp=t(h5.rt98[[i]][[j]]),phylo=r.dends[[i]])
  }
}
names(rep98.use)<-c("upland", "settlement", "forest", "lowland")

# Constrain rotation by overlap in parasite community between host use and calculate
rep98.I.use<-vector("list", 4)
rep98.I.use<-lapply(rep98.I.use, function(x) vector("list", length = 50))
for(i in seq_along(rep98.use)){
  for(j in seq_along(rep98.use[[i]])){
    rep98.I.use[[i]][[j]]<-I.rotate(rep98.use[[i]][[j]],h5.rt98[[i]][[j]])
  }
}
names(rep98.I.use)<-c("upland", "settlement", "forest", "lowland")

# Extract parasite community-constrained Gmin
rep98I.use.index<-vector("list", length = length(ha))
for(i in seq_along(rep98.I.use)){
  rep98I.use.index[[i]]<-unlist(sapply(rep98.I.use[[i]],function(x) x[1]))
}
names(rep98I.use.index)<-c("upland", "settlement", "forest", "lowland")

# Calculate unweighted connectance
rep98I.mat.use<-vector("list", length = length(ha))
for(i in seq_along(rep98.I.use)){
  rep98I.mat.use[[i]]<-sapply(rep98.I.use[[i]],function(x) x[2])
}
names(rep98I.mat.use)<-c("upland", "settlement", "forest", "lowland")

rep98.uc.use<-vector("list", length = length(ha)) # uc: unweighted connectance
for(i in seq_along(rep98I.mat.use)){
  for(j in seq_along(rep98I.mat.use[[i]])){
    rep98.uc.use[[i]][[j]]<-networklevel(rep98I.mat.use[[i]][[j]], index = "connectance")
  }
}
names(rep98.uc.use)<-c("upland", "settlement", "forest", "lowland")

# Statistics
mod1<-glm(unlist(rep98I.use.index)~unlist(h5.rt98.size)+unlist(rep98.uc.use), family="binomial")
kruskal.test(mod1$residuals~rep98.land)
kruskalmc(mod1$residuals, rep98.land, alpha=0.05/4)

Ir.use.mod1.uc<-data.frame(mod1$residuals,rep98.land)
Ir.use.mod1.uc$or<-factor(Ir.use.mod1.uc$rep98.land, levels=c("fo","up","lo","se"))
Ir.use.mod1.uc<-Ir.use.mod1.uc[order(Ir.use.mod1.uc$or),]
```

### 1.5.4 Plots

```
In [ ]: rep98.mod.uc<-data.frame(I=c(Ir.max.mod1.uc$mod1.residuals,Ir.phy.mod1.uc$mod1.residuals),
                                order=c(rep("opt",12),rep("phy",12),rep("use",12)),
                                land=c(rep(c(rep("fo", 3), rep("up", 3),rep("lo", 3),rep("se", 3)),12)))

par(mfrow=c(2,3), cex.axis=0.5, cex.lab=0.5)
boxplot(I~land+order, data = rep98.mod.uc[rep98.mod.uc$order=="opt",],
        col=c(rep(viridis(3, alpha=1, option = "D")[1],4)),
        xlab="Land",
        ylab=expression("Residuals G(k" ["min"] * ") ~ size + connectance"),
        yaxt="n",
        names=rep(c("Forest", "Upland", "Lowland", "Settlement"),1))
axis(side=2, las=2)

boxplot(I~land+order, data = rep98.mod.uc[rep98.mod.uc$order=="phy",],
        col=c(rep(viridis(3, alpha=1, option = "D")[2],4)),
        xlab="Land",
        ylab=expression("Residuals G(k" ["min"] * ") ~ size + connectance"),
        yaxt="n",
        names=rep(c("Forest", "Upland", "Lowland", "Settlement"),1))
axis(side=2, las=2)

boxplot(I~land+order, data = rep98.mod.uc[rep98.mod.uc$order=="use",],
        col=c(rep(viridis(3, alpha=1, option = "D")[3],4)),
        xlab="Land",
        ylab=expression("Residuals G(k" ["min"] * ") ~ size + connectance"),
        yaxt="n",
        names=rep(c("Forest", "Upland", "Lowland", "Settlement"),1))
axis(side=2, las=2)
```

## 2. Interaction niche of hosts

### 2.1 Intervality: optimal

```
In [ ]: # Calculate the optimal intervality
I.max<-lapply(ha.t, function(x) Interval(x))

# Extract Gmin
I.max.index<-unlist(sapply(I.max,function(x) x[1]))
```

### 2.2 Intervality: taxonomy

```
In [ ]: # Load parasite taxonomic trees
trees<-readRDS("trees.RDS")
trees.tiplabel<-lapply(trees, `[`, 4)

# Constrain rotation by parasite taxonomy and calculate intervality
# We edited the function Interval in foodweb package to limit the row
# rotation by dendrogram ordination to minimise the number of gaps.
# Acceptance probability and calculation of number of gaps are identical
# to Interval function in foodweb package.
rr.2<-vector("list", 4)
for(i in seq_along(trees)){
  rr.2[[i]]<-I.rotate(ha.t[[i]],trees[[i]])
}
```

```
# Extract taxonomically-constrained Gmin
I.tax.index.rot<-unlist(lapply(rr.2, `[`, 1))
```

## 2.3 Intervality: overlap in host use by parasites

```
In [ ]: # Load parasite dendrograms by host use
dends<-readRDS("dends.RDS")

# Constrain rotation by parasites' host use and calculate intervality
r.use<-vector("list", 4)
for(i in seq_along(dends)){
  r.use[[i]]<-I.rotate(ha.t.use.ord[[i]],dends[[i]])
}

# Extract host use-constrained Gmin
I.use.index.rot<-unlist(lapply(r.use, `[`, 1))
```

## 2.4 Comparison within habitats: null communities

```
In [ ]: # Create 1,000 null communities for each habitat type
ha.t.sw<-lapply(ha.t, function(x) swap.web(1000, x)) # parasites in rows
names(ha.t.sw)<-c("upland", "settlement", "forest", "lowland")

# Calculate intervality of the null communities
I.max.sw<-vector('list', length(ha.t.sw))
for(i in seq_along(ha.t.sw)){
  for(j in seq_along(ha.t.sw[[i]])){
    I.max.sw[[i]][[j]]<-Interval(ha.t.sw[[i]][[j]])
  }
}
names(I.max.sw)<-c("upland", "settlement", "forest", "lowland")

# Extract Gmin of the null communities
I.max.sw.index<-vector("list", length = 4)
for (i in seq_along(I.max.sw)){
  I.max.sw.index[[i]]<-unlist(lapply(I.max.sw[[i]], `[`, 1))
}
names(I.max.sw.index)<-c("upland", "settlement", "forest", "lowland")

# Results null communities
I.results.rot<-cbind(I.max.index,
                    I.tax.index.rot,
                    I.use.index.rot)
rownames(I.results.rot)<-c("upland", "settlement", "forest", "lowland")

# Plot results by habitat type
par(mfrow=c(2,2))
for (i in seq_along(I.max.sw.index)) {
  hist(I.max.sw.index[[i]],
       xlim=c(min(I.max.sw.index[[i]]-10, max(I.max.sw.index[[i]]+100),
               main=rownames(I.results.rot)[[i]],
               las=1, col="grey95", breaks = 10)
  abline(v=I.results.rot[i,], col = viridis(3, alpha=1, option = "D"),
         lwd=5)
  box()
}

# Calculate the z-score
z<-vector("list", length = 4)
```

```

for (i in seq_along(I.max.sw.index)) {
  for(j in seq(I.max.sw.index[[i]])){
    z[[i]][[j]]<-(I.results.rot[i,j]-mean(I.max.sw.index[[i]]))/sd(I.max.sw.index[[i]])
  }
}

```

## 2.5 Comparison between communities

### 2.5.1 Intervality: optimal

```

In [ ]: # Calculate the optimal intervality
h.rep98.I<-vector("list", length = length(ha.t))
for (i in seq_along(h5.rt98)) {
  for (j in seq_along(h5.rt98[[i]])) {
    h.rep98.I[[i]][[j]]<-Interval(t(h5.rt98[[i]][[j]]))
  }
}
names(h.rep98.I)<-c("upland", "settlement", "forest", "lowland")

# Extract Gmin
h.rep98I.max.index<-vector("list", length = length(ha))
for(i in seq_along(h.rep98.I)){
  h.rep98I.max.index[[i]]<-unlist(sapply(h.rep98.I[[i]],function(x) x[1]))
}
names(h.rep98I.max.index)<-c("upland", "settlement", "forest", "lowland")

# Calculate unweighted connectance
h.rep98.uc<-vector("list", length = length(ha)) # uc: unweighted connectance
for(i in seq_along(h.rep98.mat)){
  for(j in seq_along(h.rep98.mat[[i]])){
    h.rep98.uc[[i]][[j]]<-networklevel(h.rep98.mat[[i]][[j]], index = "connectance")
  }
}

names(h.rep98.uc)<-c("upland", "settlement", "forest", "lowland")

# Statistics
mod1<-glm(unlist(h.rep98I.max.index)~unlist(h5.rt98.size)+unlist(h.rep98.uc), family="binomial")
kruskal.test(mod1$residuals~rep98.land)
kruskalmc(mod1$residuals, rep98.land, alpha=0.05/4)

I.max.mod1.uc<-data.frame(mod1$residuals,rep98.land)
I.max.mod1.uc$or<-factor(I.max.mod1.uc$rep98.land, levels=c("fo","up","lo","se"))
I.max.mod1.uc<-I.max.mod1.uc[order(I.max.mod1.uc$or),]

```

### 2.5.2 Intervality: taxonomy

```

In [ ]: # Prune the dendrograms by replicate
h.rep98.phy<-vector("list", length = 4)
h.rep98.phy<-lapply(h.rep98.phy, function(x) vector("list", length = 50))
for(i in seq_along(h5.rt98)) {
  for(j in seq_along(h5.rt98[[i]])){
    h.rep98.phy[[i]][[j]]<-prune.sample(samp=h5.rt98[[i]][[j]],phylo=trees[[i]])
  }
}
names(h.rep98.phy)<-c("upland", "settlement", "forest", "lowland")

# Constraint rotation by parasite taxonomy and calculate intervality.
# We edited the function Interval in foodweb package to limit the row

```



```

# rotation by dendrogram ordination to minimise the number of gaps.
h.rep98.I.phy<-vector("list", 4)
h.rep98.I.phy<-lapply(h.rep98.I.phy, function(x) vector("list", length = 50))
for(i in seq_along(h.rep98.phy)){
  for(j in seq_along(h.rep98.phy[[i]])){
    h.rep98.I.phy[[i]][[j]]<-I.rotate(t(h5.rt98[[i]][[j]]),h.rep98.phy[[i]][[j]])
  }
}
names(h.rep98.I.phy)<-c("upland", "settlement", "forest", "lowland")

# Extract taxonomically-constrained Gmin
h.rep98I.phy.index<-vector("list", length = length(ha))
for(i in seq_along(h.rep98.I.phy)){
  h.rep98I.phy.index[[i]]<-unlist(sapply(h.rep98.I.phy[[i]],function(x) x[1]))
}
names(h.rep98I.phy.index)<-c("upland", "settlement", "forest", "lowland")

# Calculate unweighted connectance
h.rep98I.mat.phy<-vector("list", length = length(ha))
for(i in seq_along(h.rep98.I.phy)){
  h.rep98I.mat.phy[[i]]<-sapply(h.rep98.I.phy[[i]],function(x) x[2])
}
names(h.rep98I.mat.phy)<-c("upland", "settlement", "forest", "lowland")

h.rep98.uc.phy<-vector("list", length = length(ha)) # uc: unweighted connectance
for(i in seq_along(h.rep98I.mat.phy)){
  for(j in seq_along(h.rep98I.mat.phy[[i]])){
    h.rep98.uc.phy[[i]][[j]]<-networklevel(h.rep98I.mat.phy[[i]][[j]], index = "cor")
  }
}
names(h.rep98.uc.phy)<-c("upland", "settlement", "forest", "lowland")

# Statistics
mod1<-glm(unlist(h.rep98I.phy.index)~unlist(h5.rt98.size)+unlist(h.rep98.uc.phy), family="binomial")
kruskal.test(mod1$residuals~rep98.land)
kruskalmc(mod1$residuals, rep98.land, alpha=0.05/4)

I.phy.mod1.uc<-data.frame(mod1$residuals,rep98.land)
I.phy.mod1.uc$or<-factor(I.phy.mod1.uc$rep98.land, levels=c("fo","up","lo","se"))
I.phy.mod1.uc<-I.phy.mod1.uc[order(I.phy.mod1.uc$or),]

```

### 2.5.3 Intervality: overlap in host use by parasites

```

In [ ]: # Prune the dendrograms by replicate
h.rep98.use<-vector("list", length = 4)
h.rep98.use<-lapply(h.rep98.use, function(x) vector("list", length = 50))
for(i in seq_along(h5.rt98)) {
  for(j in seq_along(h5.rt98[[i]])){
    h.rep98.use[[i]][[j]]<-prune.sample(samp=h5.rt98[[i]][[j]],phylo=dends[[i]])
  }
}
names(h.rep98.use)<-c("upland", "settlement", "forest", "lowland")

# Constraint rotation by parasites' host use and calculate intervality
h.rep98.I.use<-vector("list", 4)
h.rep98.I.use<-lapply(h.rep98.I.use, function(x) vector("list", length = 50))
for(i in seq_along(h.rep98.use)){
  for(j in seq_along(h.rep98.use[[i]])){
    h.rep98.I.use[[i]][[j]]<-I.rotate(t(h5.rt98[[i]][[j]]),h.rep98.use[[i]][[j]])
  }
}

```

```

# Extract host use-constrained Gmin
h.rep98I.use.index<-vector("list", length = length(ha))
for(i in seq_along(h.rep98I.use)){
  h.rep98I.use.index[[i]]<-unlist(sapply(h.rep98I.use[[i]],function(x) x[1]))
}
names(h.rep98I.use.index)<-c("upland", "settlement", "forest", "lowland")

# Calculate unweighted connectance
h.rep98I.mat.use<-vector("list", length = length(ha))
for(i in seq_along(h.rep98I.use)){
  h.rep98I.mat.use[[i]]<-sapply(h.rep98I.use[[i]],function(x) x[2])
}
names(h.rep98I.mat.use)<-c("upland", "settlement", "forest", "lowland")

h.rep98.uc.use<-vector("list", length = length(ha)) # uc: unweighted connectance
for(i in seq_along(h.rep98I.mat.use)){
  for(j in seq_along(h.rep98I.mat.use[[i]])){
    h.rep98.uc.use[[i]][[j]]<-networklevel(h.rep98I.mat.use[[i]][[j]], index = "cor")
  }
}
names(h.rep98.uc.use)<-c("upland", "settlement", "forest", "lowland")

# Statistics
mod1<-lm(unlist(h.rep98I.use.index)~unlist(h5.rt98.size)+unlist(h.rep98.uc.use))
TukeyHSD(aov(lm(mod1$residuals~rep98.land)))

I.use.mod1.uc<-data.frame(mod1$residuals, rep98.land)
I.use.mod1.uc$or<-factor(I.use.mod1.uc$rep98.land, levels=c("fo", "up", "lo", "se"))
I.use.mod1.uc<-I.use.mod1.uc[order(I.use.mod1.uc$or),]

```

## 2.5.4 Plot

```

In [ ]: h.rep98.mod.uc<-data.frame(I=c(I.max.mod1.uc$mod1.residuals,I.phy.mod1.uc$mod1.residuals),
                                   order=c(rep("opt",200),rep("phy",200),rep("use",200)),
                                   land=c(rep(c(rep("fo", 50), rep("up", 50),rep("lo", 50),rep("se", 50)),200)))

par(mfrow=c(2,3))
boxplot(I~land+order, data = h.rep98.mod.uc[h.rep98.mod.uc$order=="opt",],
        col=c(rep(viridis(3, alpha=1, option = "D")[1],4)),
        xlab="Habitat",
        ylab=expression("Residuals G(k" ["min"] * ") ~ size + connectance"),
        main="Number of gaps 98%",
        yaxt="n",
        names=rep(c("Forest", "Upland", "Lowland", "Settlement"),1))
axis(side=2, las=2)

boxplot(I~land+order, data = h.rep98.mod.uc[h.rep98.mod.uc$order=="phy",],
        col=c(rep(viridis(3, alpha=1, option = "D")[2],4)),
        xlab="Habitat",
        ylab=expression("Residuals G(k" ["min"] * ") ~ size + connectance"),
        main="Number of gaps 98%",
        yaxt="n",
        names=rep(c("Forest", "Upland", "Lowland", "Settlement"),1))
axis(side=2, las=2)

boxplot(I~land+order, data = h.rep98.mod.uc[h.rep98.mod.uc$order=="use",],
        col=c(rep(viridis(3, alpha=1, option = "D")[3],4)),
        xlab="Habitat",
        ylab=expression("Residuals G(k" ["min"] * ") ~ size + connectance"),
        main="Number of gaps 98%",

```

```
yaxt="n",  
names=rep(c("Forest", "Upland", "Lowland", "Settlement"),1))  
axis(side=2, las=2)
```