Baltic Zooplankton Data Preparation

Peter M.J. Herman and Lisa Sundqvist

Tuesday, November 06, 2018

Introduction: data sources for Baltic zooplankton

At the time of preparation of this product, not all Baltic zooplankton data were yet stored into the EMODNET data base. This will be the case by the finishing date of the project, but in the meantime we drew on diverse data sources to prepare the product. The Swedish SHARK database was incorporated into EMODNET, and was retrieved as two files, one with occurrences and one with 'measurements or facts'. The Finnish data were retrieved from the NOAA Copepod database (https://www.st.nmfs.noaa.gov/copepod/). The large number of files was compiled into a single file and further processed. The German and Polish data were retrieved from the HELCOM DOME database, an extract of which was put at our disposal by HELCOM secretariat.

The following code chunks document the pre-treatment of the different datasets, that were eventually combined into a single file.

Swedish data

```
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 3.2.5
```

```
#
######################
## READ BASIC DATA ##
######################
#
oc      <-   read.table("Swedish-occurrence.txt",header=T,sep="\t",stringsAsFactors=F)
oc$id   <-   as.factor(oc$id)
me      <-   read.table("Swedish-measurementorfact.txt",header=T,sep="\t",
                        stringsAsFactors=F)
me$id   <-   as.factor(me$id)
#
#############################################################
## CORRECT MEASUREMENTS AND LINK WITH OBSERVATION FILE ##
#############################################################
#
# make a pivot table of measurements
mep<-dcast(me,id~measurementType + measurementUnit, value.var='measurementValue',
           function(x) sum(x,na.rm = T))
# recalculate all biomass in g/m3 to biomass in mg/m3
mep[!is.na(mep[,4])&is.na(mep[,5]),5]<-mep[!is.na(mep[,4])&is.na(mep[,5]),4]*1000
# calculate all biomasses in mg/m2 where ratio can be deduced from abundances
ml<-which(!is.na(mep[,2])&!is.na(mep[,3])&is.na(mep[,6])&!is.na(mep[,5]))
mep[ml,6]<-mep[ml,5]*mep[ml,3]/mep[ml,2]
# drop biomasses in g/m2
mep<-mep[,-4]
#merge the two tables
```

```
meoc<-merge(oc,mep,by="id",all.x=T,sort=T)
#
##############################
## PRUNE THE COMBINED TABLE ##
##############################
#
# drop all variables that have no meaningful data (only NA or only one same
# value everywhere)
nvo<-nmo<-vector(length=69)
for (i in 1:69){
  nvo[i]<-length(which(!is.na(meoc[,i])))
  nmo[i]<-length(unique(meoc[,i]))
  }
dl<-which(nvo==0|nmo==1)
meoc<-meoc[,-dl]
# delete records without abundance (not useful for this product - can be very useful
# for other purposes!)
meoc<-meoc[-which(is.na(meoc$'Abundance_ind/m2')),]
# add a single date field to meoc
meoc$date<-as.Date(ISOdate(meoc$year,meoc$month,meoc$day))
# delete records without species information
ll<-which(is.na(meoc$scientificName))
if(length(ll)>0)meoc<-meoc[-ll,]
#
############################
## CORRECT STATION NAMES ##
############################
#
# correct for yes/no space in "? 17" and use normal A
meoc$locality[meoc$locality==meoc$locality[51]]<-"A17"
meoc$locality[meoc$locality==meoc$locality[656]]<-"A17"
#correct for upper/lower case in "Anholt E"
meoc$locality[meoc$locality=="Anholt E"]<-"ANHOLT E"
#similar for Falkenberg
meoc$locality[meoc$locality=="N14 Falkenberg"]<-"N14 FALKENBERG"
# solve problems with double spaces in locality names
meoc$locality<-gsub("  "," ",meoc$locality)
# solve another locality name problem
meoc$locality[meoc$locality=="BY31 LANDSORTSDJ"]<-"BY31"
# and fuse F9 / A13 with A13
meoc$locality[meoc$locality=="F9 / A13"]<-"A13"
#
########################################
## SOLVE REPLICATE/DEPTH SLICE PROBLEM ##
########################################
#
# make fieldNumber refer to replicates only (drop letters referring to depth slices)
meoc$fieldNumber<-gsub("a","",meoc$fieldNumber)
meoc$fieldNumber<-gsub("b","",meoc$fieldNumber)
meoc$fieldNumber<-gsub("c","",meoc$fieldNumber)
meoc$fieldNumber<-gsub("d","",meoc$fieldNumber)
meoc$fieldNumber<-gsub("e","",meoc$fieldNumber)
meoc$fieldNumber<-gsub(" ","",meoc$fieldNumber)
```

```
meoc$fieldNumber[meoc$fieldNumber==""]<-"1"
# determine the number of replicates per sampling occasion in a station
nrps<-dcast(meoc,formula=locality+date~.,value.var='fieldNumber',fun=max)
```

```
## Warning in .fun(.value[0], ...): no non-missing arguments, returning NA
```

```
names(nrps)[3]<-'nrps'
meoc<-merge(meoc,nrps,by=c('locality','date'))
#
############################################################
## STORE THE BASIC DATA TABLE IN CSV and binary FILEs ##
############################################################
#
save(meoc,file="SwedishData/meoc.Rdata")
write.csv(meoc,file="AllSwedishData.csv")

#############################################################
## PREPARE SWEDISCH DATA IN FORMAT LIKE OTHER COUNTRIES ##
#############################################################

  Swedish<- read.csv("AllSwedishData.csv",header=T,stringsAsFactors = F,sep=",")
  # keep only wanted columns
  houd<-c("locality","individualCount","sex","lifeStage","year","month","day",
          "fieldNumber","minimumDepthInMeters","maximumDepthInMeters",
          "decimalLatitude","decimalLongitude","scientificName","Abundance_ind.m2",
          "lifeClass","nrps")
  Swedish<-Swedish[colnames(Swedish)%in%houd]
  # drop lines with Abundance==0
  Swedish<-Swedish[Swedish$Abundance_ind.m2>0,]
  # many samples have more than one replicate (given in nrps). We will sum up
  # replicates later, now divide
  # abundance by nrps, so that by summing later the average abundance will be given
  Swedish$Abundance_ind.m2<-Swedish$Abundance_ind.m2/Swedish$nrps
  # aggregate over life stages, depth slices and replicates
  Swedish_agg<-aggregate(Abundance_ind.m2~scientificName+year+month+day+locality+
                         decimalLatitude+decimalLongitude,Swedish, FUN=sum)
  # calculate average coordinates of localities
  Swedish_lat<-aggregate(decimalLatitude~locality,Swedish_agg,FUN=mean)
  Swedish_lon<-aggregate(decimalLongitude~locality,Swedish_agg,FUN=mean)
  names(Swedish_lat)<-c("locality","lat")
  names(Swedish_lon)<-c("locality","lon")
  Swedish_agg<-merge(Swedish_agg,Swedish_lat,by="locality")
  Swedish_agg<-merge(Swedish_agg,Swedish_lon,by="locality")
  # and store maximum depth of the sample
  Swedish_mxd<-aggregate(maximumDepthInMeters~year+month+day+locality+
                           decimalLatitude+decimalLongitude,
                           Swedish, FUN=function(x) max(x))
  Swedish_agg_z<-merge(Swedish_agg,Swedish_mxd,by=c("decimalLatitude",
                       "decimalLongitude","day","month","year","locality"))
  # we store the result back in Swedish
  Swedish_2<- data.frame(scientificname=Swedish_agg_z$scientificName,
                       maximumdepth=Swedish_agg_z$maximumDepthInMeters,
                       minimumdepth=rep(0,nrow(Swedish_agg_z)),
```

```
                      latitude=Swedish_agg_z$lat,
                      longitude=Swedish_agg_z$lon,
                      country=rep("Sweden",nrow(Swedish_agg_z)),
                      daycollected=Swedish_agg_z$day,
                      monthcollected=Swedish_agg_z$month,
                      yearcollected=Swedish_agg_z$year,
                      measurementvalue=Swedish_agg_z$Abundance_ind.m2)
```

German data, retrieved from HELCOM DOME

```
### Make German file, run only once
# Helcom <- read.csv("HELCOM_DOME_ZP_20171105.csv",header = T,stringsAsFactors = F)
# German <-Helcom[Helcom$Country == 'Germany',]
# Save German data to file
# write.csv(German,file="German.csv",row.names = F,col.names = T)

  German <- read.csv("AllGermanData.csv",header = T,stringsAsFactors = F, sep=',')
  #
  houd<-c("Year","Month","Day","Lat","Lon","Country",
            "MNDEP","MXDEP","Species","Value","PARAM","MUNIT",
            "NPORT","CPORT","SMVOL","SAREA")
  German<-German[colnames(German)%in%houd]
  # restrict to abundance measures only
  German<-German[German$PARAM=="ABUNDNR",]
  # Make two subsets, one with n, the other with n/m3, because they need
  # different treatment
  German_o<-German[German$MUNIT=="nr",]
  German_n<-German[German$MUNIT=="nr/m3",]
  # for the file with abundance per m3, transform to abundance per m2
  German_n$Value<-German_n$Value*(German_n$MXDEP-German_n$MNDEP)
  German_n$MUNIT<-"nr/m2"
  # for the file with abundance nr, complete SAREA, sum per sample,
  # then calculate aerial abundance
  German_o$SAREA[is.na(German_o$SAREA)]<-250
  German_o$SAREA<-German_o$SAREA/100/100  # from cm2 to m2
  German_o<-aggregate(Value~Year+Month+Day+Lat+Lon+Country+MNDEP+MXDEP+
                      Species+PARAM+MUNIT+NPORT+CPORT
                      +SMVOL+SAREA,German_o,FUN=sum)
  German_o$Value<-German_o$Value*German_o$NPORT/German_o$CPORT/German_o$SAREA
  German_o$MUNIT<-"nr/m2"
  # reconstruct German, and sum depth slices
  German<-rbind(German_o,German_n)
  German_agg<-aggregate(Value~Species+Year+Month+Day+Lat+Lon,German,FUN=sum)
  # store maximum depth of the sample
  G_mxd<-aggregate(MXDEP~Lat+Lon+Year+Month+Day,German,FUN=function(x) max(x))
  German_agg_z<-merge(German_agg,G_mxd,by=c("Lat","Lon","Day","Month","Year"))
  # we store the result back in German_2
  German_2<- data.frame(scientificname=German_agg_z$Species,
                      maximumdepth=German_agg_z$MXDEP,
                      minimumdepth=rep(0,nrow(German_agg_z)),
                      latitude=German_agg_z$Lat,
                      longitude=German_agg_z$Lon,
```

```
                      country=rep("Germany",nrow(German_agg_z)),
                      daycollected=German_agg_z$Day,
                      monthcollected=German_agg_z$Month,
                      yearcollected=German_agg_z$Year,
                      measurementvalue=German_agg_z$Value)
```

Polish data

```
### Make Polish file, run only once
# Helcom <- read.csv("HELCOM_DOME_ZP_20171105.csv",header = T,stringsAsFactors = F)
# Polish <-Helcom[Helcom$Country == 'Poland',]
# Save Polish data to file
# write.csv(Polish,file="Polish.csv",row.names = F,col.names = T)
  Polish <- read.csv("AllPolishData.csv",header = T,stringsAsFactors = F)
  # keep only wanted columns
  houd<-c("scientificname_accepted","yearcollected","monthcollected",
          "daycollected","longitude","latitude","minimumdepth","maximumdepth",
          "sex","observedindividualcount","measurementtype",
          "measurementvalue","measurementunit")
  Polish<-Polish[colnames(Polish)%in%houd]
  # restrict to abundance only
  Polish<-Polish[Polish$measurementunit=="#/m3",]
  # recalculate density per m3 to density per m2 by multiplying with (mxdep-mindep)
  Polish$VALUE.per.area<-Polish$measurementvalue*(Polish$maximumdepth-Polish$minimumdepth)
  # aggregate over life stages and depth slices
  Polish_agg<-aggregate(VALUE.per.area~scientificname_accepted+yearcollected
                        +monthcollected+daycollected+
                        latitude+longitude,Polish, FUN=sum)
  # and store maximum depth of the sample
  P_mxd<-aggregate(maximumdepth~latitude+longitude+yearcollected+monthcollected
                   +daycollected,Polish, FUN=function(x) max(x))
  Polish_agg_z<-merge(Polish_agg,P_mxd,by=c("latitude","longitude",
                      "daycollected","monthcollected","yearcollected"))
  # we store the result back in Polish
  Polish_2<- data.frame(scientificname=Polish_agg_z$scientificname_accepted,
                      maximumdepth=Polish_agg_z$maximumdepth,
                      minimumdepth=rep(0,nrow(Polish_agg_z)),
                      latitude=Polish_agg_z$latitude,
                      longitude=Polish_agg_z$longitude,
                      country=rep("Poland",nrow(Polish_agg_z)),
                      daycollected=Polish_agg_z$daycollected,
                      monthcollected=Polish_agg_z$monthcollected,
                      yearcollected=Polish_agg_z$yearcollected,
                      measurementvalue=Polish_agg_z$VALUE.per.area)
```

Finnish Data

```
  # For the Finnish data I manually removed unidentified when used after a genus
  # name to be able to match with WORMS
  Finnish <- read.csv("AllFinnishDataFixed.csv",header = T,stringsAsFactors = F, sep=';')
```

```
  # here we can simply select the column called value.per.area and use that as
  # our column n_m2
  # we then proceed to the summation per species and sample.
  # there are no replicates per sample

  # first some cleanup
  names(Finnish)[names(Finnish)=="LONGITDE"]<-"LONGITUDE"
  Finnish$VALUE.per.area<-as.numeric(Finnish$VALUE.per.area)
```

```
## Warning: NAs introduced by coercion
```

```
  Finnish<-Finnish[!is.na(Finnish$VALUE.per.area),]
  # remove unwanted columns
  weg<-c("SHP.CRUISE","TIMEgmt","TIMEloc","T","GEAR","MESH","NMFS_PGC","ITIS_TSN",
       "MOD","LIF","PSC","SEX","V","Water.Strained","Original.VALUE","Orig.UNITS",
       "VALUE.per.volu","F1","F2","F3","F4","F1.1","F2.1","F3.1","F4.1",
       "SCIENTIFIC.NAME....modifiers...","RECORD.ID","DATASET.ID","SHIP",
       "Orig.STATION.ID","Taxa.Name","PROJ","INST","Orig.CRUISE.ID")
  Finnish<-Finnish[! colnames(Finnish)%in%weg]
  # remove lines without species
  Finnish<-Finnish[!is.na(Finnish$Taxa.Name.Fixed),]
  # correct value per area for measurements with units #/ml: multiply by 10^6
  Finnish$VALUE.per.area[Finnish$UNITS==" #/ml"]<-
     Finnish$VALUE.per.area[Finnish$UNITS==" #/ml"]*1000000
  # aggregate over life stages and depth slices
  Finnish_agg<-aggregate(VALUE.per.area~Taxa.Name.Fixed+LATITUDE+LONGITUDE+
                           DAY+MON+YEAR,Finnish, FUN=sum)
  # and store maximum depth of the sample
  F_mxd<-aggregate(LOWER_Z~LATITUDE+LONGITUDE+DAY+MON+YEAR,Finnish,
                   FUN=function(x) max(x))
  Finnish_agg_z<-merge(Finnish_agg,F_mxd,by=c("LATITUDE","LONGITUDE","DAY","MON","YEAR"))
  # We give uniform names and order to the columns
  Finnish_2<-data.frame(scientificname=Finnish_agg_z$Taxa.Name.Fixed,
                        maximumdepth=Finnish_agg_z$LOWER_Z,
                        minimumdepth=rep(0,nrow(Finnish_agg_z)),
                        latitude=Finnish_agg_z$LATITUDE,
                        longitude=Finnish_agg_z$LONGITUDE,
                        country=rep("Finland",nrow(Finnish_agg_z)),
                        daycollected=Finnish_agg_z$DAY,
                        monthcollected=Finnish_agg_z$MON,
                        yearcollected=Finnish_agg_z$YEAR,
                        measurementvalue=Finnish_agg_z$VALUE.per.area)
```

Compilation of all data into a single file

At this stage taxonomic correction by comparison with WORMS is executed. We also solve the problem of stations. In most files only the actually measured coordinates at the time of sampling are stored, not the intended ('station') coordinates. This gives slight modifications of the location, rendering the identification of stations difficult. The problem is solved by grouping coordinates that are close to one another into averaged coordinates of a station.

At the end a single file is written with all data.

```
# combine the four files into a single one
bzp<-rbind(Finnish_2,Polish_2,German_2,Swedish_2)
bzp$scientificname<-as.character(bzp$scientificname)
### Update taxonomy to accepted Scientificnames according to WORMS
# A species list has been made up and submitted to WORMS.
# Output has been edited to produce a translation table
# Import file with species matched to WORMS
#
spml<-read.csv("Allspecies_matched.csv",header = T,stringsAsFactors = F)
# merge with existing file
bzpm<-merge(bzp,spml,by="scientificname",all.x=T)
bzpm<-bzpm[order(bzpm$scientificname_accepted),]
#
bzpm<-bzpm[,-1]
#
# The problem of stations
#
sts<-unique(bzpm[,3:4]) # unique cominations of latitude and longitude in the dataset
sts$nlat<-sts$latitude
sts$nlon<-sts$longitude

# a histogram shows that everything below distance 0.1 is probably just
# replicate samples of the same station
# we look for these pairs and delete them

for(i in 1:(nrow(sts)-1)){
  for (j in ((i+1):nrow(sts))){
    dist<-sqrt((sts$nlat[i]-sts$nlat[j])^2+(sts$nlon[i]-sts$nlon[j])^2)
    if(dist<0.1){
      sts$nlat[j]<-sts$nlat[i]
      sts$nlon[j]<-sts$nlon[i]
    }
  }
}

stun<-unique(sts[,c(3,4)])
stun$station<-paste("STAT",1:nrow(stun),sep="")

bzpm<-merge(bzpm,sts,by=c("latitude","longitude"),all.x=T)
bzpm<-merge(bzpm,stun,by=c("nlat","nlon"),all.x=T)

bzpm<-bzpm[,c(7,13,2,1,6,5,10,9,8,12,11)]
names(bzpm)[3:4]<-c("longitude","latitude")

### Save output files
save(bzpm,file="bzpm.Rdata")
write.csv(bzpm,file="bzpm.csv",row.names = F)
```

Selection of stations and species

We produce maps for a number of years in which many stations across the Baltic were sampled. Only species with sufficiently frequent occurrence (more than 50 occurrences in total) are selected. The result is written into a binary and a .csv file.

```r
# read in data frame with all observations

load("bzpm.Rdata")

############################################
## SELECT YEARS AND MOST FREQUENT SPECIES ##
############################################
#
# select data from the years 2007, 2008, 2010, 2011, 2012, 2013
bz<-subset(bzpm,subset=bzpm$yearcollected %in% c(2007, 2008, 2010, 2011, 2012, 2013)
           & bzpm$monthcollected %in% c(6,7,8,9))
# calculate total numbers caught and frequency of all species
freqsp<-dcast(bz,scientificname_accepted~.,fun.aggregate =
                function(x) length(x[x>0]),value.var = "measurementvalue")
names(freqsp)<-c("scientificname_accepted","frequency")

# select only species with a frequency of over 50 (i.e. occurring in more than 50 samples)
bz<-merge(bz,freqsp,by="scientificname_accepted",all.x=T)
bz<-bz[bz$frequency>50,]
bz<-bz[,-12]

# average coordinates per station
 avglat<-dcast(bz,station~.,fun.aggregate = mean,value.var = "latitude")
 names(avglat)<-c("station","latitude")
 avglon<-dcast(bz,station~.,fun.aggregate = mean,value.var = "longitude")
 names(avglon)<-c("station","longitude")
 locs<-merge(avglat,avglon,by="station")
 nlocs<-nrow(locs)
 bz<-merge(bz[,-c(4,5)],locs,by="station")


specnames<-unique(bz$scientificname_accepted)
specnames<-specnames[order(specnames)]
nspecs<-length(specnames)
#
################################
## STORE DATA IN BINARY FILE ##
################################
#
save(bz,specnames,nspecs,locs,nlocs,file="freq_species_abund.Rdata")
bz2<-data.frame(country=bz$country,
                station=bz$station,
                latitude=bz$latitude,
                longitude=bz$longitude,
                daycollected=bz$daycollected,
                monthcollected=bz$monthcollected,
                yearcollected=bz$yearcollected,
                scientificname_accepted=bz$scientificname_accepted,
                measurementvalue=bz$measurementvalue)
write.csv(bz2,file="balticZooplankton.csv")
#
#
#############################
```

```
## END OF DATA PREPARATION ##
##############################
```