Muusoctopus leioderma respiration

Lloyd Trueblood and Kirt Onthank

2022-08-29

Contents

1 Reading in libraries

I am reading in the libraries I use for this analysis. Included among these is the "OTools" package, which was written by Kirt Onthank. This can be install from github using the command:

```
install_github('KirtOnthank\OTools')
```

The install_github command is from the 'remotes' library.

```
library(OTools)
library(xlsx)
library(nlme)
library(car)
library(emmeans)
library(respirometry)
library(knitr)
```

2 Find the relevant files

This code is meant to find the metabolic rate files in the current directory and sort them into their types.

```
files=list.files(recursive=T)
resp.files=grep(".txt",files,value=T)
pcrit.files=grep("pcrit|pcrti",resp.files,value=T,ignore.case=T)
metab.files=setdiff(resp.files,pcrit.files)
blank.files=grep("blank_only",resp.files,value=T,ignore.case=T)
rmr.files=setdiff(metab.files,blank.files)
rmr.files=rmr.files[!grepl("-ch2.txt|-ch3.txt|-ch4.txt|\\(1\\).txt",rmr.files)]
rmr.files=rmr.files[!duplicated(basename(rmr.files))]
```

3 Reading in the data log file

This reads in the data log file, which contains information on octopus mass, flow rate, and other associated information.

```
data.log=read.csv("Muus_Data_Log.csv")
```

4 Running the RMR data analysis

First I am going to make a object to put the RMR data into.

```
routine=data.frame(filename=as.character(),
                   spreadsheet_guess=as.character(),
                   octo=as.character(),
                   mass=as.numeric(),
                   pco2=as.numeric(),
                   day=as.numeric(),
                   rmr=as.numeric()
                   )
```

Then I am running a quick check to make sure that we are matching file names of the metabolic runs with the lines in the Data Log file.

```
file_check=as.character()
score=as.numeric()
for (i in 1:length(rmr.files)){
  filename=rmr.files[i]
  guess=which.min(adist(basename(filename),data.log$File.name))
  file_check[i]=data.log$File.name[guess]
  score[i]=min(adist(basename(filename),data.log$File.name))
}

write.csv(cbind(basename(rmr.files),file_check,score),file = "filecheck.csv")
```

Nest, I run the analysis to calculate RMR from each file. We are discarding the first three hours (using data as resp[resp$times>3600*3,] in the resp.open function) from each run to account for elevated metabolic rate at the beginning of the run.

```
column.count=1
for (i in 1:length(rmr.files)){
  filename=rmr.files[i]
  print(paste("starting file ", basename(filename)," (loop",i,")",sep=""))
  if(length(grep("Group 4|presens|ch\\d\\.txt",basename(filename)))>0){
    resp=read.presens(filename)
  }else{
    resp=read.pyro(filename)
  }
  print("finding closest match in log")

guess=which.min(adist(basename(filename),data.log$File.name))

    flow=as.numeric(data.log$flow.rate..L.min.[guess])
    mass=as.numeric(data.log$Mass..g.[guess])
    if(is.na(flow)){
      flow=0.1
    }
    if(is.na(mass)){
      mass=10
    }
    print("calculating rmr")
    resp.mean=mean(resp.open(resp[resp$times>3600*3,],
                             flow.rate=flow*1000,
                             weight=mass)$resp,
                   na.rm=T)
    print("writing data to object")
    routine[column.count,1]=basename(filename)
    routine[column.count,2]=data.log$File.name[guess]
    routine[column.count,3]=data.log$octo1[guess]
    routine[column.count,4]=mass

    if(length(grep("1800",filename))>0){
     routine[column.count,5]=1800
    }
    if(length(grep("1000",filename))>0){
      routine[column.count,5]=1000
    }
    routine[column.count,6]=data.log$day[guess]
    routine[column.count,7]=resp.mean
    column.count=column.count+1
    if(sum(is.na(resp$O23))<10&!grepl("blank",filename)){
      print("found second respirometer")
      flow=as.numeric(data.log$Flow.rate.2[guess])
      mass=as.numeric(data.log$Mass.2[guess])
      resp.mean=mean(resp.open(resp[resp$times>3600*3,],
                               inflow=3,
                               outflow=4,
                               flow.rate=flow*1000,
                               weight=mass)$resp,na.rm=T)
      print("writing data to object")
      routine[column.count,1]=basename(filename)
      routine[column.count,2]=data.log$File.name[guess]
```

```
      routine[column.count,3]=data.log$octo2[guess]
      routine[column.count,4]=mass

      if(length(grep("1800",filename))>0){
       routine[column.count,5]=1800
      }
      if(length(grep("1000",filename))>0){
        routine[column.count,5]=1000
      }
      routine[column.count,6]=data.log$day[guess]
      routine[column.count,7]=resp.mean
      column.count=column.count+1
    }
  print(paste("end of file ", basename(filename)," (loop",i,")",sep=""))
}
```

In order to not need to re-run the analysis each time, I am writing the results out to a .csv, then reading it back in. I will set both the evaluation of this chunk and the last to FALSE.

```
write.csv(routine,"RMR_Results.csv")
```

```
routine=read.csv("RMR_Results.csv")
```

5 Running linear effects model

To make the relationship linear between mass and metabolic rate linear, I took the log of both.

```
routine$mass.log=log(routine$mass)
routine$rmr.log=log(routine$rmr)
```

Setting pCO_2 to factor class:

```
routine$pco2=as.factor(routine$pco2)
```

Next I set orthogonal contrasts:

```
contrasts(routine$pco2)=contr.poly(2)
```

Running the linear mixed effects model and ANOVA using type III sum of squares:

```
routine.lme=lme(rmr.log~mass.log+pco2+day,random=~1|octo,
             correlation=corAR1(form=~day|octo),
              data=routine[routine$octo!="2-1",])
routine.anova=Anova(routine.lme,type="III")
routine.anova
```

```
## Analysis of Deviance Table (Type III tests)
##
## Response: rmr.log
```

```
##               Chisq Df Pr(>Chisq)
## (Intercept) 35.8232  1  2.161e-09 ***
## mass.log    13.2309  1  0.0002754 ***
## pco2         0.0906  1  0.7634316
## day          0.0555  1  0.8138299
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
rmr.lme.table=cbind(
  c("Mass","pCO2","Day"),
  round(routine.anova$Chisq[2:4],2),
  routine.anova$Df[2:4],
  round(routine.anova$`Pr(>Chisq)`[2:4],5)
)
colnames(rmr.lme.table)=c("Factor","Chi-square", "DF", "p-value")

kable(rmr.lme.table)
```

Factor	Chi-square	DF	p-value
Mass	13.23	1	0.00028
pCO2	0.09	1	0.76343
Day	0.06	1	0.81383

6 Summary of LME

```
routine.em=data.frame(emmeans(routine.lme,~pco2+day+mass.log))
```

```
## Warning: contrasts dropped from factor pco2
```

```
routine.em
```

```
##   pco2 day mass.log    emmean        SE df  lower.CL upper.CL
## 1 1000   1 2.906025 0.9553184 0.1738296 13 0.5797823 1.330855
## 2 1800   1 2.906025 1.0278433 0.1738296 13 0.6523072 1.403379
## 3 1000   7 2.906025 0.9715304 0.1738296 13 0.5959943 1.347067
## 4 1800   7 2.906025 1.0440553 0.1738296 13 0.6685192 1.419591
```

```
rmr.df=
data.frame(cbind(
  as.numeric(as.character(routine.em$pco2)),
  routine.em$day,
  sprintf("%.2f",signif(exp(routine.em$emmean),3)),
  paste(sprintf("%.2f",signif(exp(data.frame(routine.em)$lower.CL),3)),
        "-",
        sprintf("%.2f",signif(exp(data.frame(routine.em)$upper.CL),3)))
))
rmr.df
```

```
##     X1 X2   X3          X4
## 1 1000  1 2.60 1.79 - 3.78
## 2 1800  1 2.80 1.92 - 4.07
## 3 1000  7 2.64 1.81 - 3.85
## 4 1800  7 2.84 1.95 - 4.14
```

```
colnames(rmr.df)=c("pCO~2~ ($\\mu$atm)",
                   "day",
                   "Routine Metabolic Rate ($\\mu$O~2~ g^-1^ hr^-1^)",
                   "RMR 95% CI")
kable(rmr.df,align="c")
```

pCO_2 (μatm)	day	Routine Metabolic Rate (μO_2 g^{-1} hr^{-1})	RMR 95% CI
1000	1	2.60	1.79 - 3.78
1800	1	2.80	1.92 - 4.07
1000	7	2.64	1.81 - 3.85
1800	7	2.84	1.95 - 4.14

7 Plotting the data

7.1 Predicted values

First I am getting the model predicted values for each treatment between the max an minimum mass values.

```
seq1.1800=seq(from=min(routine$mass.log[routine$pco2==1800]),
              to=max(routine$mass.log[routine$pco2==1800]),
              length.out=100)

df1.1800=data.frame(
  day=rep(1,100),
  mass.log=seq1.1800,
  pco2=as.factor(rep(1800,100))
)
pred1.1800= predict(routine.lme,newdata = df1.1800,level=0)

seq1.1000=seq(from=min(routine$mass.log[routine$pco2==1000]),
              to=max(routine$mass.log[routine$pco2==1000]),
              length.out=100)

df1.1000=data.frame(
  day=rep(1,100),
  mass.log=seq1.1000,
  pco2=as.factor(rep(1000,100))
)

pred1.1000=predict(routine.lme,newdata = df1.1000,level=0)

seq7.1800=seq(from=min(routine$mass.log[routine$pco2==1800]),
              to=max(routine$mass.log[routine$pco2==1800]),
              length.out=100)
```

```
df7.1800=data.frame(
  day=rep(7,100),
  mass.log=seq7.1800,
  pco2=as.factor(rep(1800,100))
)

pred7.1800=predict(routine.lme,newdata = df7.1800,level=0)

seq7.1000=seq(from=min(routine$mass.log[routine$pco2==1000]),
              to=max(routine$mass.log[routine$pco2==1000]),
              length.out=100)

df7.1000=data.frame(
  day=rep(7,100),
  mass.log=seq7.1000,
  pco2=as.factor(rep(1000,100))
)

pred7.1000=predict(routine.lme,newdata = df7.1000,level=0)
```

Next, I am actually plotting it.

```
svg(filename="Figure_1.svg",height=3.5,width=3.5,pointsize=6)
par(fig=c(0.04,1,0,1))
plot(rmr~mass,data=routine[routine$octo!="2-1",],log="xy",axes=F,ylab="",xlab="",type="n")
box(lwd=2)
axis(1,lwd=2,cex.axis=1.5)
axis(2,lwd=2,cex.axis=1.5)
mtext(expression("Routine Metabolic Rate ("*mu*"molO"[2]*" g"^-1*"hr"^-1*")"),
      side=2,cex=1.8,line=2.5)
mtext("Mass (g)",side=1,cex=1.8,line=2.5)

points(rmr~mass,data=routine[routine$pco2==1000&routine$day==1&routine$octo!="2-1",],
       pch=21,bg="white",col="blue",cex=2)
points(rmr~mass,data=routine[routine$pco2==1000&routine$day==7&routine$octo!="2-1",],
       pch=21,bg="blue",cex=2)
points(rmr~mass,data=routine[routine$pco2==1800&routine$day==7&routine$octo!="2-1",],
       pch=21,bg="red",cex=2)
points(rmr~mass,data=routine[routine$pco2==1800&routine$day==1&routine$octo!="2-1",],
       pch=21,bg="white",col="red",cex=2)
lines(exp(seq1.1800),exp(pred1.1800),col="red",lwd=2,lty=2)
lines(exp(seq1.1000),exp(pred1.1000),col="blue",lwd=2,lty=2)
lines(exp(seq7.1800),exp(pred7.1800),col="red",lwd=2,lty=1)
lines(exp(seq7.1000),exp(pred7.1000),col="blue",lwd=2,lty=1)
legend("topright", c(expression("1000 "*mu*"atm pCO"["2"]*", day 1"),
                     expression("1000 "*mu*"atm pCO"["2"]*", day 7"),
                     expression("1800 "*mu*"atm pCO"["2"]*", day 1"),
                     expression("1800 "*mu*"atm pCO"["2"]*", day 7")),
       pch = 21,bty="n",title = expression("Treatment pCO"["2"]),
       pt.bg=c("white","blue","white","red"),col=c("blue","black","red","black"),
       inset = .02,cex=1.3,box.lwd=2,pt.lwd=1,pt.cex=2)

dev.off()
```

```
## pdf
##   2
```

Converting the image to a png to be displayed in the RMarkdown.

```
cairosvg Figure_1.svg -o Figure_1.png -d 300
```

Converting to eps for submission.

```
inkscape Figure_1.svg -o Figure_1.eps --export-ignore-filters --export-ps-level=3
```

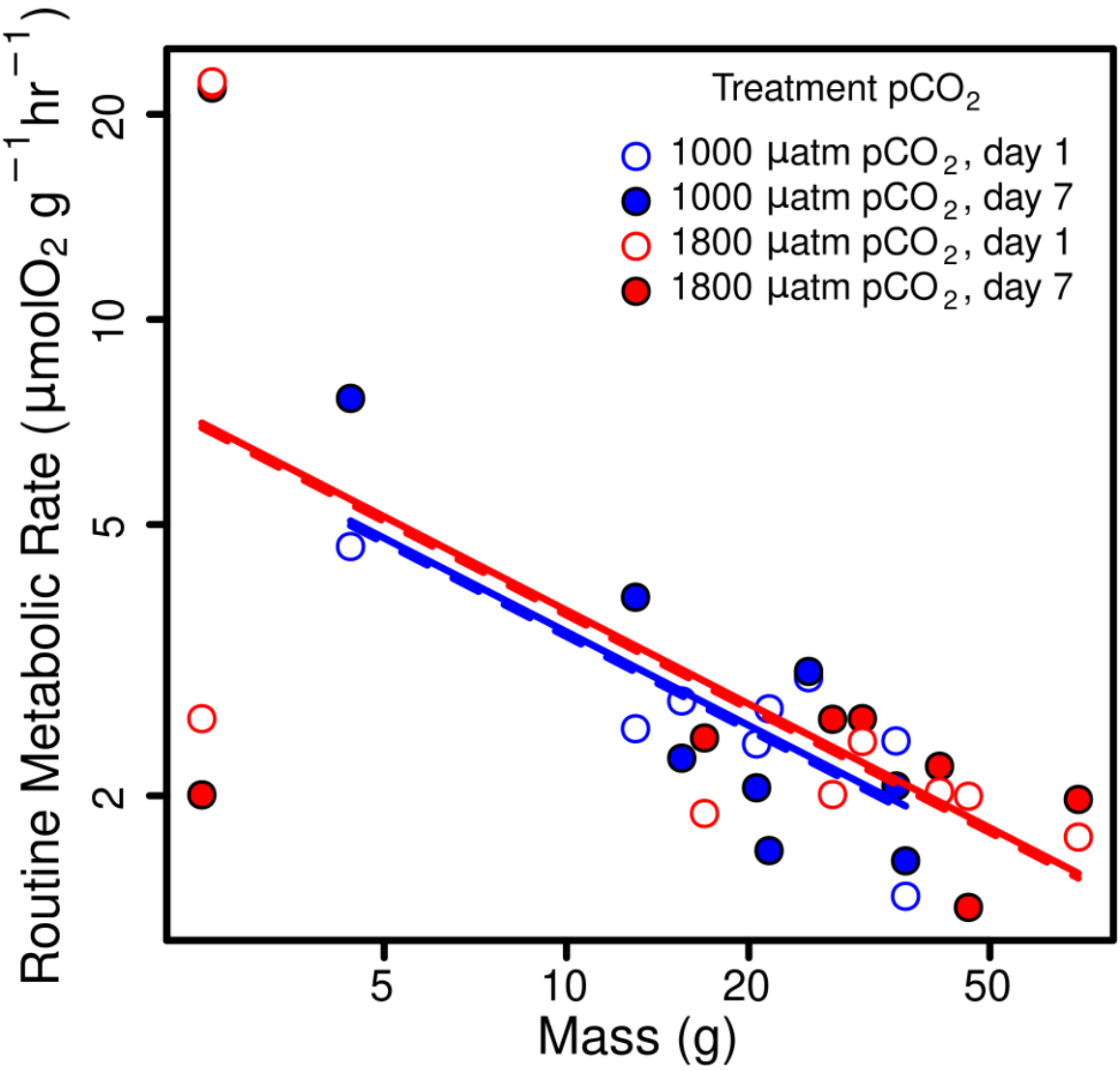

Figure 1: work in progress text

```
#logmass=log(median(routine$mass))
logmass=2.906025
rmr.df=data.frame(day=c(1,1,7,7),
           pCO2=c(1000,1800,1000,1800),
           RMR=round(exp(predict(routine.lme,
                  newdata=data.frame(
                       day=c(1,1,7,7),
                       mass.log=rep(logmass,4),
                       pco2=as.factor(c(1000,1800,1000,1800))),
                     level=0)
                )[1:4],2)
           )

colnames(rmr.df)[2]="pCO~2~ ($\\mu$atm)"
colnames(rmr.df)[3]="Routine Metabolic Rate ($\\mu$O~2~ g^-1^ hr^-1^)"

kable(rmr.df,align="c")
```

day	pCO_2 (μatm)	Routine Metabolic Rate (μO_2 g^{-1} hr^{-1})
1	1000	2.60
1	1800	2.80
7	1000	2.64
7	1800	2.84