Muusoctopus leioderma respiration

Lloyd Trueblood and Kirt Onthank

2022-09-07

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")
```

```
routine.table=routine[,4:8]
routine.table=routine.table[order(routine.table$octo),]
colnames(routine.table)=c("Octopus ID",
  "Mass (g)",
  "pCO~2~ ($\\mu$atm)",
  "Day",
  "Routine Metabolic Rate ($\\mu$O~2~ g^-1^ hr^-1^)"
  )
kable(routine.table,align="c",row.names = F)
```

Octopus ID	Mass (g)	pCO_2 (μatm)	Day	Routine Metabolic Rate (μO_2 g^{-1} hr^{-1})
1-1	30.8	1800	1	2.402579
1-1	30.8	1800	7	2.595360
1-2	20.6	1000	1	2.382507
1-2	20.6	1000	7	2.054767
1-3	2.5	1800	7	2.006317
1-3	2.5	1800	1	2.594966
2-1	16.8	1800	7	2.865356
2-2	35.0	1000	1	2.406880
2-2	35.0	1000	7	2.067067
2-3	2.6	1800	1	22.318391
2-3	2.6	1800	7	21.854992
3-1	70.0	1800	1	1.739319
3-1	70.0	1800	7	1.975492
3-2	21.6	1000	1	2.683380
3-2	21.6	1000	7	1.661946
3-3	16.9	1800	1	1.883555
3-3	16.9	1800	7	2.431324

Octopus ID	Mass (g)	pCO_2 (μatm)	Day	Routine Metabolic Rate (μO_2 g^{-1} hr^{-1})
4-1	27.5	1800	1	2.008308
4-1	27.5	1800	7	2.589715
4-2	15.5	1000	7	2.270807
4-2	15.5	1000	1	2.755131
4-3	41.3	1800	1	2.027171
4-3	41.3	1800	7	2.209496
5-1	36.3	1000	1	1.424232
5-1	36.3	1000	7	1.605300
5-2	25.1	1000	1	2.980357
5-2	25.1	1000	7	3.044391
5-3	13.0	1000	7	3.911187
5-3	13.0	1000	1	2.508412
5-4	4.4	1000	7	7.658118
5-4	4.4	1000	1	4.642320
5-5	46.1	1800	7	1.371669
5-5	46.1	1800	1	1.996929

5 Running linear effects model

To make the relationship linear between mass and metabolic rate linear, we take 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")
```

```
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	5.84	1	0.01565
pCO2	0.19	1	0.6621
Day	2.18	1	0.13939

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 0.9562967 0.1826338 12 0.5583718 1.354222
## 2 1800   1 2.906025 1.0268016 0.1826337 12 0.6288770 1.424726
## 3 1000   7 2.906025 0.9706947 0.1826338 12 0.5727698 1.368620
## 4 1800   7 2.906025 1.0451365 0.1826337 12 0.6472119 1.443061
```

```
emmeans(routine.lme,~pco2)
```

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

## NOTE: Results may be misleading due to involvement in interactions

##  pco2 emmean    SE df lower.CL upper.CL
##  1000  0.963 0.177 12    0.579     1.35
##  1800  1.036 0.177 12    0.651     1.42
##
## Results are averaged over the levels of: day
## Degrees-of-freedom method: containment
## Confidence level used: 0.95
```

```
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=rmr.df[order(rmr.df[,1]),]
rmr.df
```

```
##     X1 X2   X3          X4
## 1 1000  1 2.60 1.75 - 3.87
## 3 1000  7 2.64 1.77 - 3.93
## 2 1800  1 2.79 1.88 - 4.16
## 4 1800  7 2.84 1.91 - 4.23
```

```
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",row.names = F)
```

pCO_2 (μatm)	day	Routine Metabolic Rate (μO_2 g^{-1} hr^{-1})	RMR 95% CI
1000	1	2.60	1.75 - 3.87
1000	7	2.64	1.77 - 3.93
1800	1	2.79	1.88 - 4.16
1800	7	2.84	1.91 - 4.23

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)
```

Assigning the colors for the treatments.

```
hi.co2.col="#790000ff"
lo.co2.col="#838fd5ff"
```

Next, I am actually plotting it.

```
svg(filename="Figure_3.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=2)
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$octo!="2-1",],
       pch=22,bg="white",col=lo.co2.col,cex=1.5)
points(rmr~mass,data=routine[routine$pco2==1000&routine$day==7&routine$octo!="2-1",],
       pch=22,bg=lo.co2.col,cex=1.5)
points(rmr~mass,data=routine[routine$pco2==1800&routine$day==7&routine$octo!="2-1",],
       pch=21,bg=hi.co2.col,cex=1.5)
points(rmr~mass,data=routine[routine$pco2==1800&routine$day==1&routine$octo!="2-1",],
       pch=21,bg="white",col=hi.co2.col,cex=1.5)
lines(exp(seq1.1800),exp(pred1.1800),col=hi.co2.col,lwd=2,lty=2)
lines(exp(seq1.1000),exp(pred1.1000),col=lo.co2.col,lwd=2,lty=2)
lines(exp(seq7.1800),exp(pred7.1800),col=hi.co2.col,lwd=2,lty=1)
lines(exp(seq7.1000),exp(pred7.1000),col=lo.co2.col,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 = c(22,22,21,21),bty="n",title = expression("Treatment pCO"["2"]),
       pt.bg=c("white",lo.co2.col,"white",hi.co2.col),col=c(lo.co2.col,"black",hi.co2.col,"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_3.svg -o Figure_3.png -d 300
```

Converting to eps for submission.

```
inkscape Figure_3.svg -o Figure_3.eps --export-ignore-filters --export-ps-level=3
```

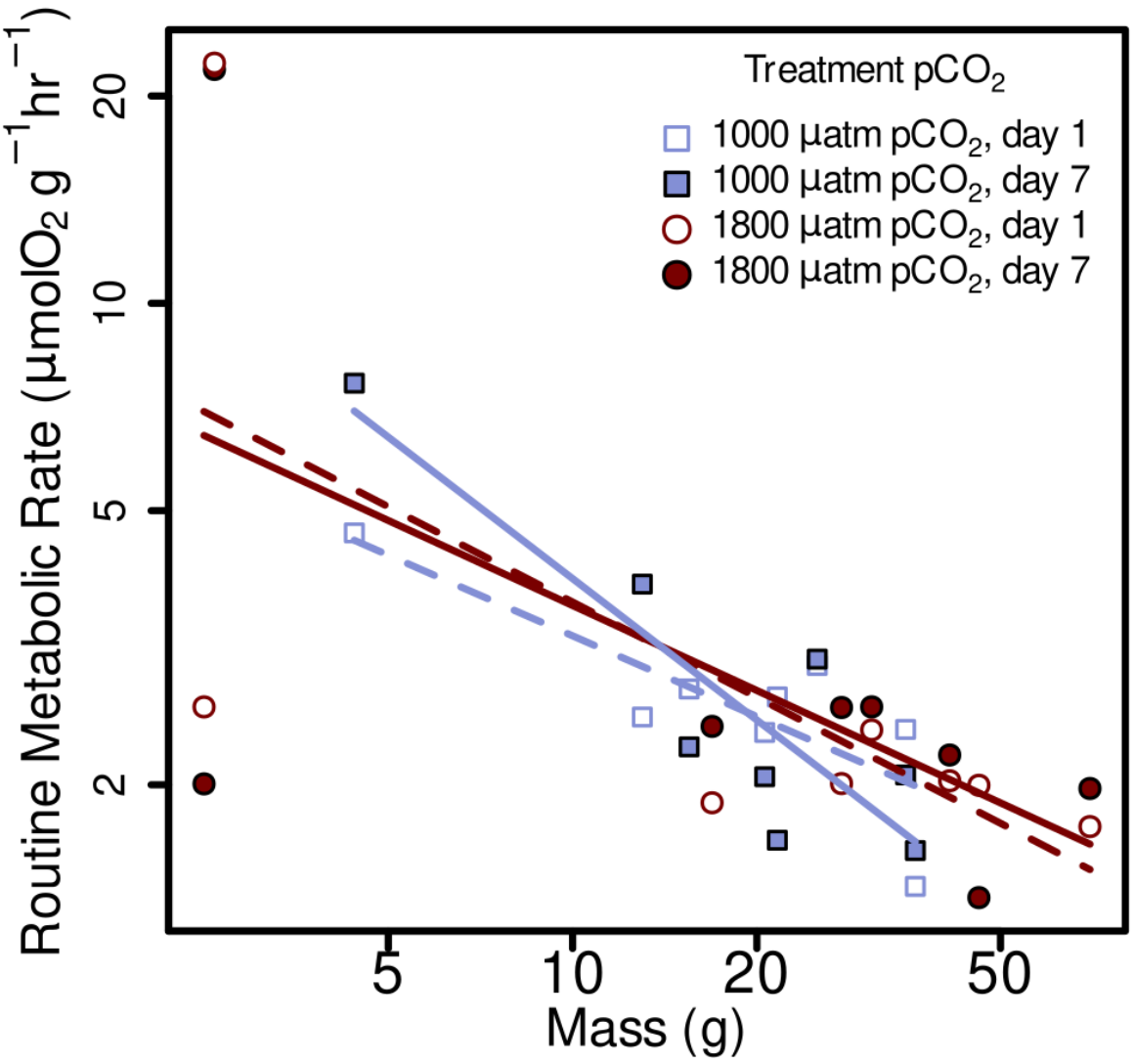

Figure 1: Routine metabolic rates (RMR) from M.leioderma in Burrows Bay, Anacortes Washington

8 Double checking the emmeans output using the predict() function as a reality check

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

colnames(rmr.df.check)[1]="pCO~2~ ($\\mu$atm)"
colnames(rmr.df.check)[3]="Routine Metabolic Rate ($\\mu$O~2~ g^-1^ hr^-1^)"
kable(rmr.df.check,align="c",row.names=F)
```

pCO_2 (μatm)	day	Routine Metabolic Rate (μO_2 g^{-1} hr^{-1})
1000	1	2.60
1000	7	2.64
1800	1	2.79
1800	7	2.84

9 Finding the slopes of the treatments

Here I am producing a table of the slope and intercepts of the linear mixed effects model of routine metabolic rates. These values correspond to the logged mass and logged RMR, and do not translate to the untransformed data. The relationship between the untransformed data is not linear, and therefore has no slope. Also, because the log of 0 is infinite, these intercepts correspond instead to a mass of 1g (exp(0)=1).

```
logmass=0
routine.slope=data.frame(pCO2=c(1000,1000,1800,1800),
           Day=c(1,7,1,7),
           Intercept=round(exp(predict(routine.lme,
                  newdata=data.frame(
                       day=c(1,7,1,7),
                       mass.log=rep(logmass,4),
                       pco2=as.factor(c(1000,1000,1800,1800))),
                     level=0)
                )[1:4],2),
           Slope=c(
            round(-1*diff(range(pred1.1000))/diff(range(seq1.1000)),2),
            round(-1*diff(range(pred7.1000))/diff(range(seq7.1000)),2),
            round(-1*diff(range(pred1.1800))/diff(range(seq1.1800)),2),
            round(-1*diff(range(pred7.1800))/diff(range(seq7.1800)),2)
            )
```

```
            )

colnames(routine.slope)[1]="pCO~2~ ($\\mu$atm)"
kable(routine.slope,align="c",row.names=F)
```

pCO_2 (μatm)	Day	Intercept	Slope
1000	1	8.06	-0.39
1000	7	19.17	-0.68
1800	1	10.61	-0.46
1800	7	9.36	-0.41