CpGoe Multiple species

Guillem Ylla

10/1/2019

CpGoe calculated in:~WorkingDir/CpGoe/1-Count_CpGoe.ipynb

Here, I apply a gaussian Mixture model, getb the mean of each curve an caclulate the standard error of each mean(se). The SE is used to calculate the 99% confidence interval.

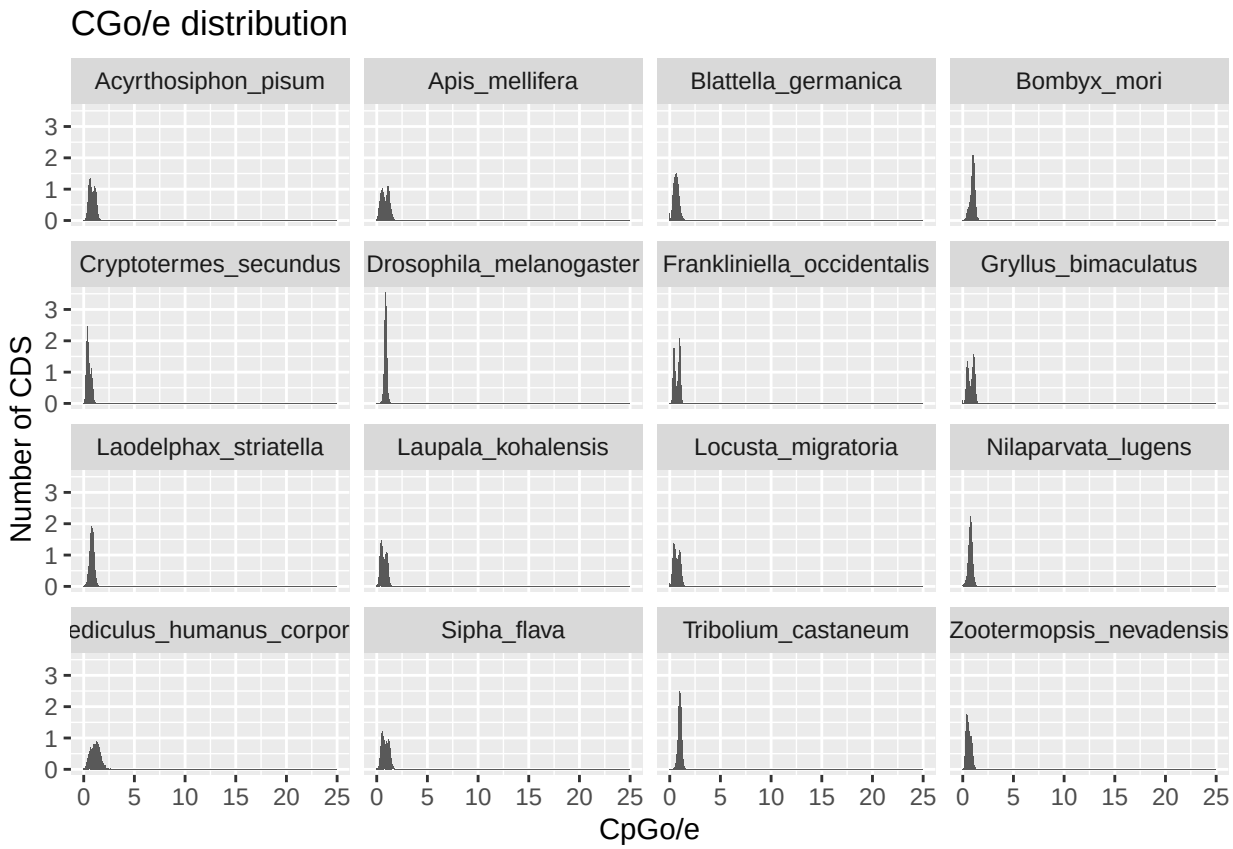
Load CpGoe values

```
CpGoe_insects <- read.delim("/home/guillem/WorkingDir/CpGoe/CpGoe_outputs/All_insects_CpGpe")
CpGoe_insects <- CpGoe_insects[!is.na(CpGoe_insects$CpGoe), ]
```

Overview CpGoe 16 Insects

Plot the CpG distribution for all 16 insects:

```
CGoe_plot1 <- ggplot(CpGoe_insects, aes(x = CpGoe, y = ..density..)) +
    facet_wrap(~Spp) + geom_histogram(bins = 1000) + scale_x_continuous(name = "CpGo/e") +
    ggtitle("CGo/e distribution") + ylab("Number of CDS")
CGoe_plot1
```

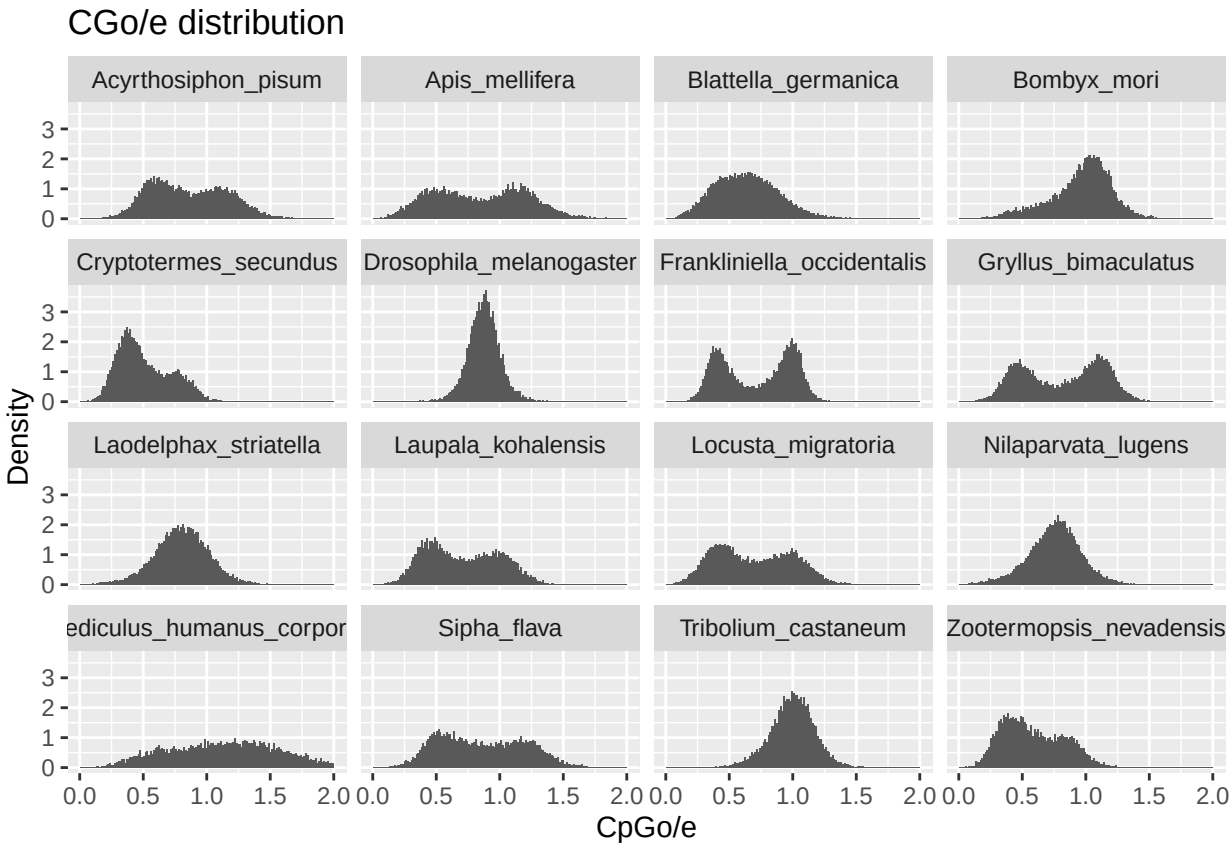


Same, but removing genes CpHoe>2.

```
dim(CpGoe_insects[CpGoe_insects$CpGoe > 2, ])
```

```
## [1] 448   8
```

```
CpGoe_insects_nolarger2 <- CpGoe_insects[CpGoe_insects$CpGoe <
    2, ]

CpGoe_insects_b02 <- CpGoe_insects[CpGoe_insects$CpGoe < 2 &
    CpGoe_insects$CpGoe > 0, ]


CGoe_plot2 <- ggplot(CpGoe_insects_b02, aes(x = CpGoe, y = ..density..)) +
    facet_wrap(~Spp) + geom_histogram(bins = 150) + scale_x_continuous(name = "CpGo/e") +
    ggtitle("CGo/e distribution") + ylab("Density")
CGoe_plot2
```




```
CGoe_plotMiniatures<-ggplot(CpGoe_insects_b02, aes(x=CpGoe,  y = ..density.. )) +
   facet_wrap(~Spp)+
   geom_histogram(bins=150, colour="black")+
   scale_x_continuous(name = "CpGo/e") +
  ggtitle("CGo/e distribution")  +ylab("Density")+
theme(
    panel.background = element_rect(fill = "transparent"), # bg of the panel
    #plot.background = element_rect(fill = "transparent", color = NA), # bg of the plot
    panel.grid.major = element_blank(), # get rid of major grid
    panel.grid.minor = element_blank(), # get rid of minor grid
```

```
    axis.text.x=element_blank(),
    axis.text.y=element_blank(),
    axis.ticks.x=element_blank(),
   axis.ticks.y=element_blank(),
    legend.background = element_rect(fill = "transparent"), # get rid of legend bg
    legend.box.background = element_rect(fill = "transparent") # get rid of legend panel bg
  )


#ggsave("CGoe_plotMiniatures.png",CGoe_plotMiniatures, dpi = 300)
```

CpGoe bimodal model 16 insects

Lets try to fit all of them in a bimofdal distribution:

FOLLOWING CHUNCK WORKS IF COPY/PAST IN THE R TERMINAl

```
for (species in unique(CpGoe_insects$Spp)) {
    print(species)
    # only cpg values >0 & <2
    CpGoe_Spp <- CpGoe_insects_b02[CpGoe_insects_b02$Spp == species,
        ]
    # CpGoe_Spp<-CpGoe_insects[CpGoe_insects$Spp==species,]
    CpGoe_toplot = CpGoe_Spp$CpGoe
    CpGoe_toplot <- CpGoe_toplot[!is.na(CpGoe_toplot)]
    mixmdl = normalmixEM(CpGoe_toplot, maxit = 20000)


    # SE1=mixmdl$sigma[1]/sqrt(length(CpGoe_toplot)*mixmdl$lambda[1])
    # CI_1_right=mixmdl$mu[1]+(SE1*1.96)
    # CI_1_left=mixmdl$mu[1]-(SE1*1.96)
    # SE2=mixmdl$sigma[2]/sqrt(length(CpGoe_toplot)*mixmdl$lambda[1])
    # CI_2_right=mixmdl$mu[2]+(SE2*1.96)
    # CI_2_left=mixmdl$mu[2]-(SE2*1.96)

    # Standard Error function from mixtools bootsratping 100
    se <- boot.se(mixmdl, B = 100)
    # Intervals
    CI_1_right = mixmdl$mu[1] + (se$mu.se[1] * 2.8)
    CI_1_left = mixmdl$mu[1] - (se$mu.se[1] * 2.8)

    CI_2_right = mixmdl$mu[2] + (se$mu.se[2] * 2.8)
    CI_2_left = mixmdl$mu[2] - (se$mu.se[2] * 2.8)


    ## function for ggplots:
    plot_mix_comps <- function(x, mu, sigma, lam) {
        lam * dnorm(x, mu, sigma)
    }


    Biomdal_dist <- data.frame(x = mixmdl$x) %>% ggplot() + geom_histogram(aes(x,
        ..density..), colour = "black", alpha = 0.8, bins = 120) +
```

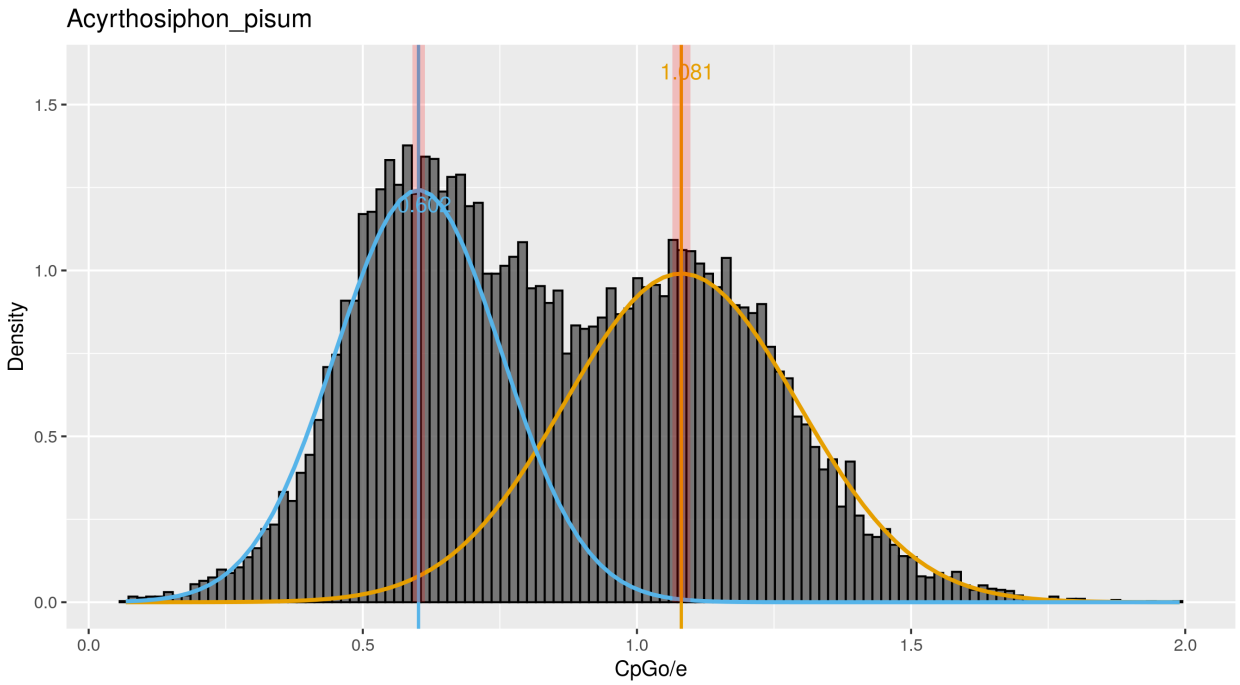
```
        stat_function(geom = "line", fun = plot_mix_comps, args = list(mixmdl$mu[1],
            mixmdl$sigma[1], lam = mixmdl$lambda[1]), colour = "#E69F00",
            lwd = 1) + stat_function(geom = "line", fun = plot_mix_comps,
        args = list(mixmdl$mu[2], mixmdl$sigma[2], lam = mixmdl$lambda[2]),
        colour = "#56B4E9", lwd = 1) + geom_vline(xintercept = mixmdl$mu[1],
        col = "#E69F00", size = 0.8) + annotate("text", label = round(mixmdl$mu[1],
        3), y = 1.6, x = mixmdl$mu[1] + 0.01, col = "#E69F00",
        size = 4) + annotate("rect", xmin = CI_1_left, xmax = CI_1_right,
        ymin = 0, ymax = Inf, alpha = 0.2, fill = "red") + geom_vline(xintercept = mixmdl$mu[2],
        col = "#56B4E9", size = 0.8) + annotate("rect", xmin = CI_2_left,
        xmax = CI_2_right, ymin = 0, ymax = Inf, alpha = 0.2,
        fill = "red") + annotate("text", label = round(mixmdl$mu[2],
        3), y = 1.2, x = mixmdl$mu[2] + 0.01, col = "#56B4E9",
        size = 4) + # annotate('text',label=paste0('LogLik=',round(mixmdl$loglik,
    # 1)),y= 1.5, x=1.5, col='black',size=4)+
    ylab("Density") + xlab("CpGo/e") + ggtitle(species)

    # red line spearateing
    data <- ggplot_build(Biomdal_dist)$data[[1]]
    data_int <- data[data$x > mixmdl$mu[1] & data$x < mixmdl$mu[2],
        ]
    cpg_threshold <- data_int[data_int$y == min(data_int$y),
        ]$x
    Biomdal_dist <- Biomdal_dist + geom_vline(xintercept = cpg_threshold,
        col = "red", size = 0.5) + annotate("text", label = round(cpg_threshold,
        3), y = 1.3, x = cpg_threshold + 0.01, col = "red", size = 4)

    ggsave(paste0("/home/guillem/WorkingDir/CpGoe/plots/", paste0(species,
        ".png")), Biomdal_dist, width = 9, height = 5, units = "in",
        dpi = 300, )

}
```

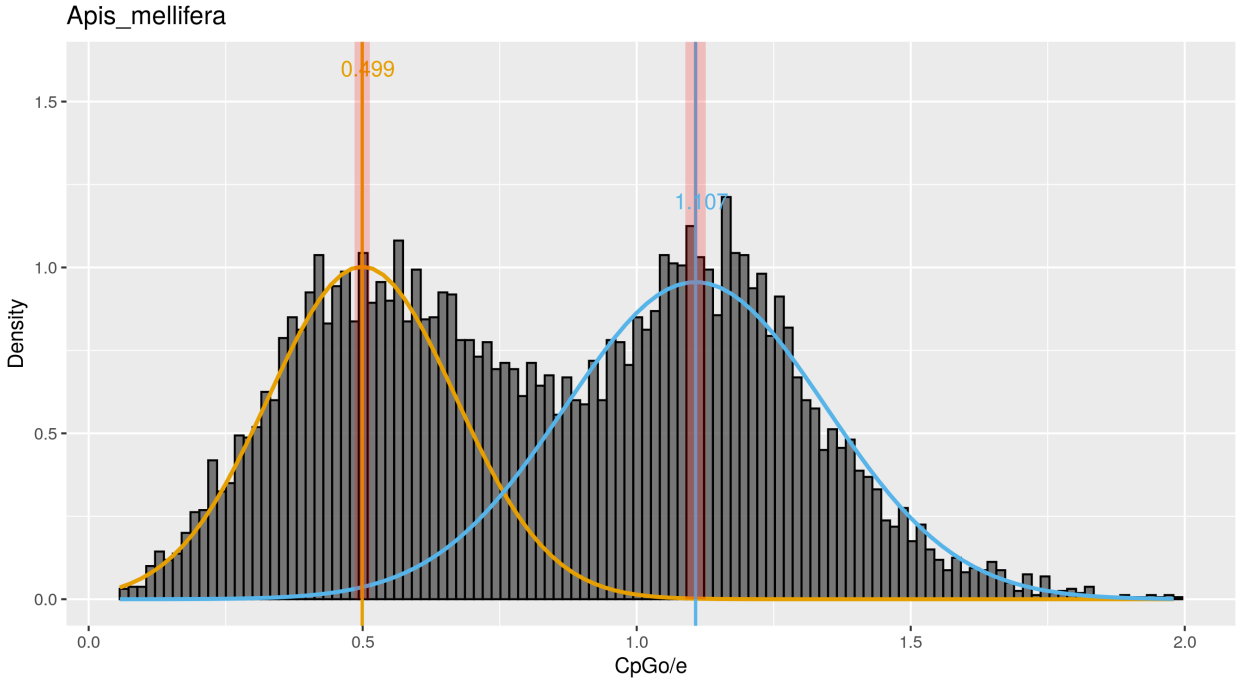
The Gussian Mixture model, fits 2 gaussian ditributions to the CpGoe of each insect.

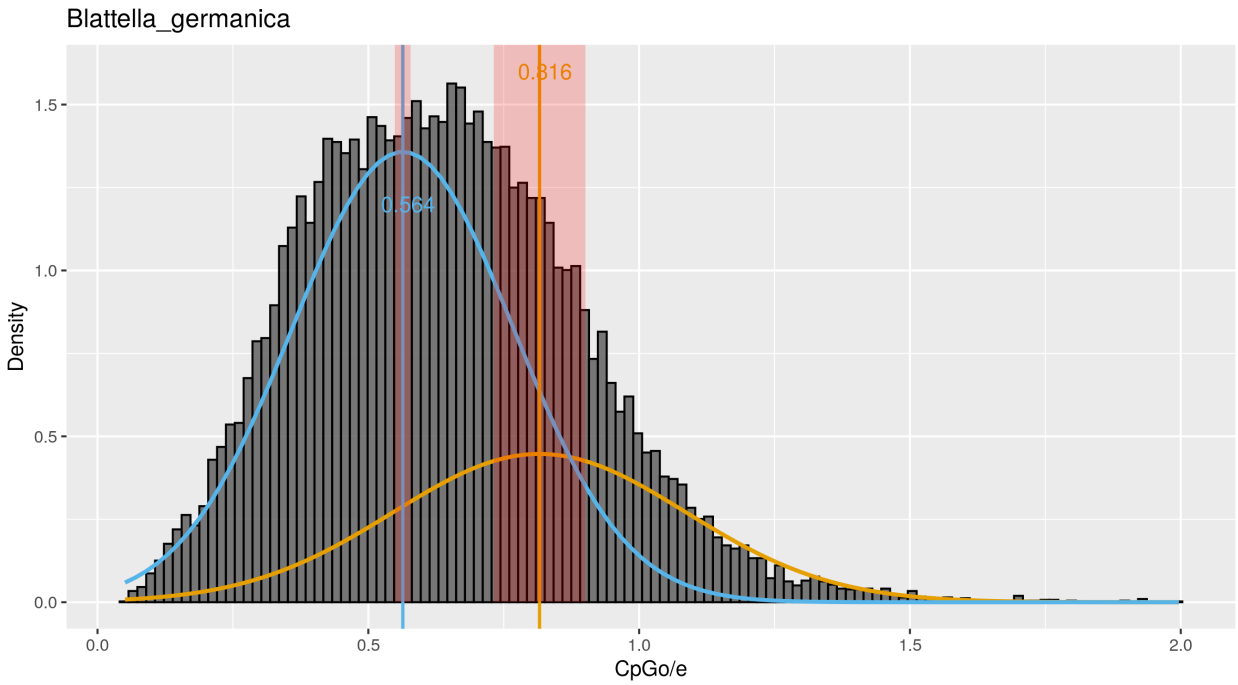
I get the mean of each curve (vertical bars) an calculate the standard error (SE) of each mean with 100 bootstraps The SE is used to calculate the 99% confidence interval around the mean (red shade).

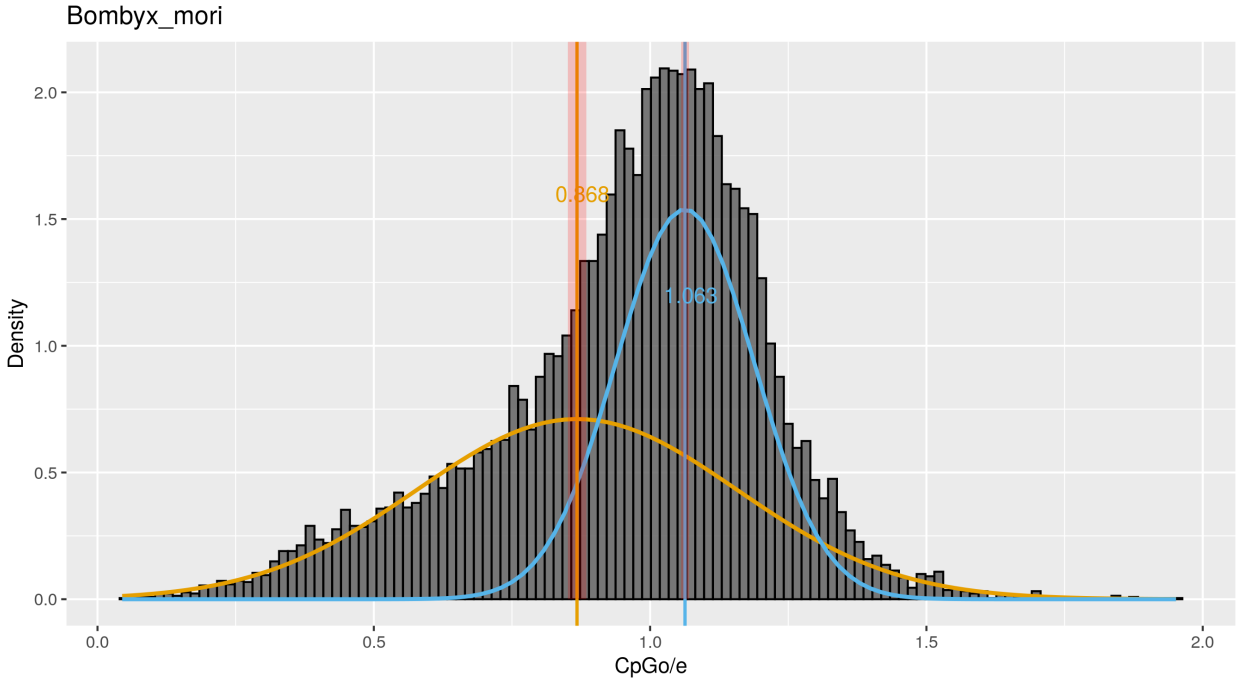
The red line represents the intersection between the 2 curves, and it is used as threshold between genes with **Low and High CpGoe**.

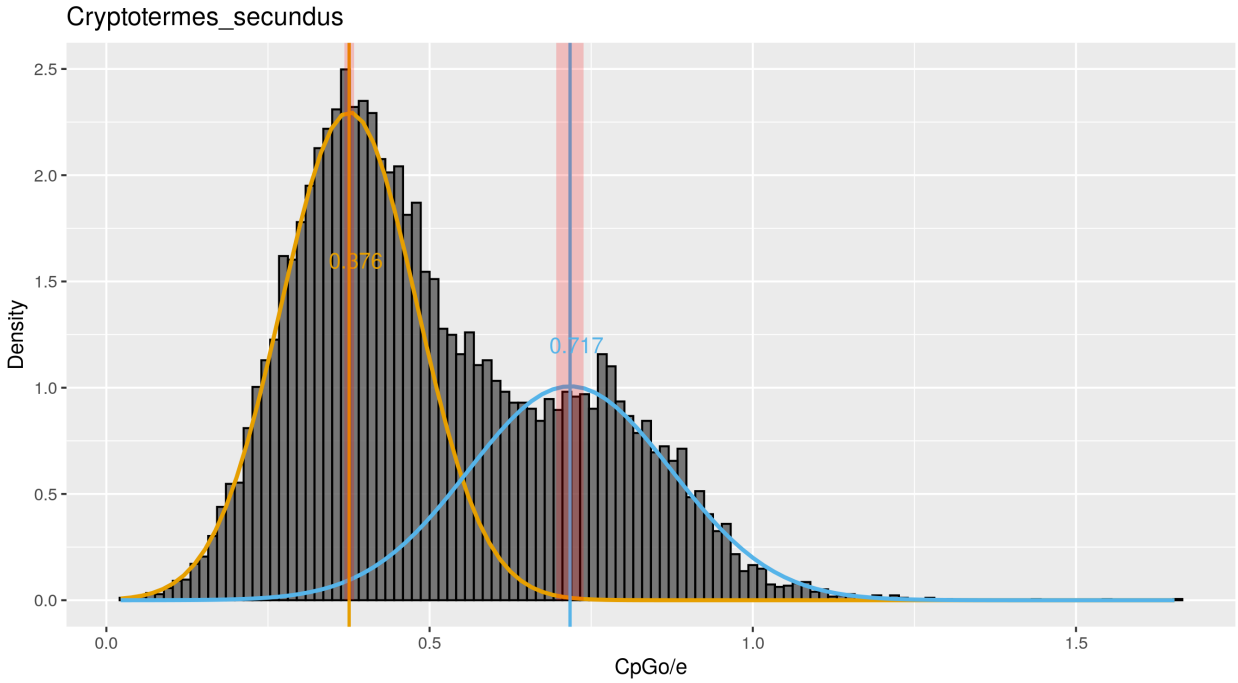
```
plots <- list.files("/home/guillem/WorkingDir/CpGoe/plots/")

for (i in plots) {
    filename <- file.path("/home/guillem/WorkingDir/CpGoe/plots",
        i)
    cat("![text](", filename, ")\n")
}
```

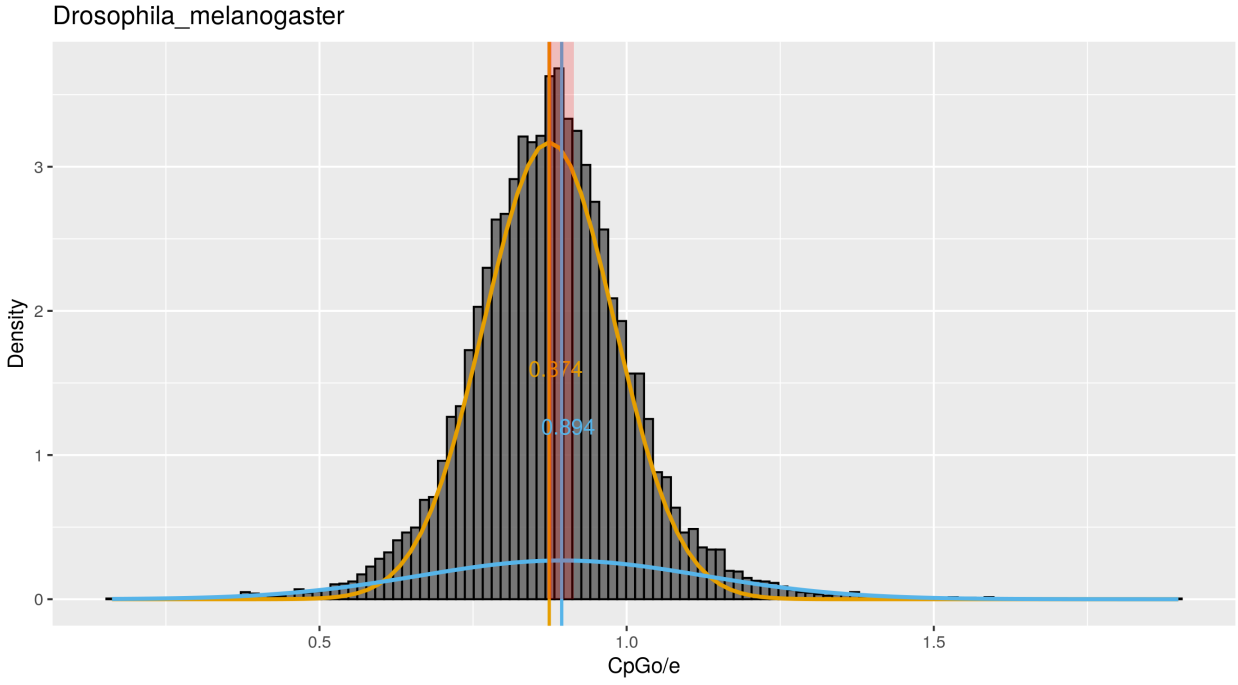

Acyrthosiphon_pisum
1.081
0.608
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0

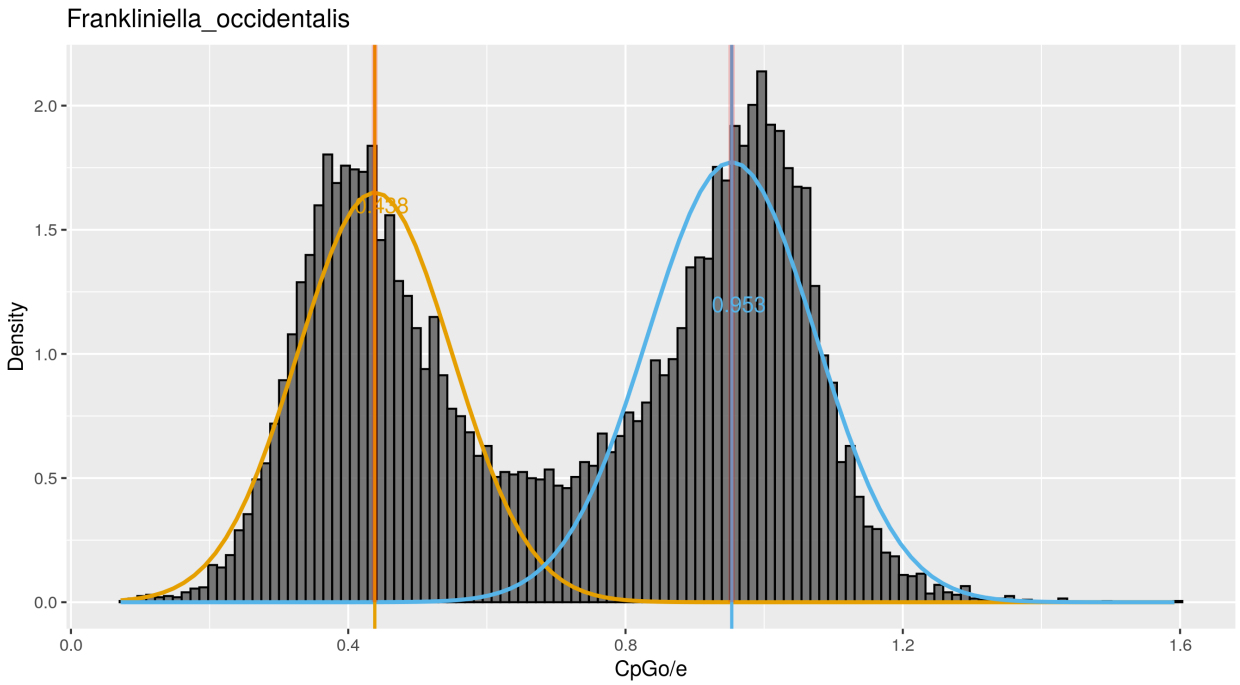

Apis_mellifera
0.499
1.107
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0

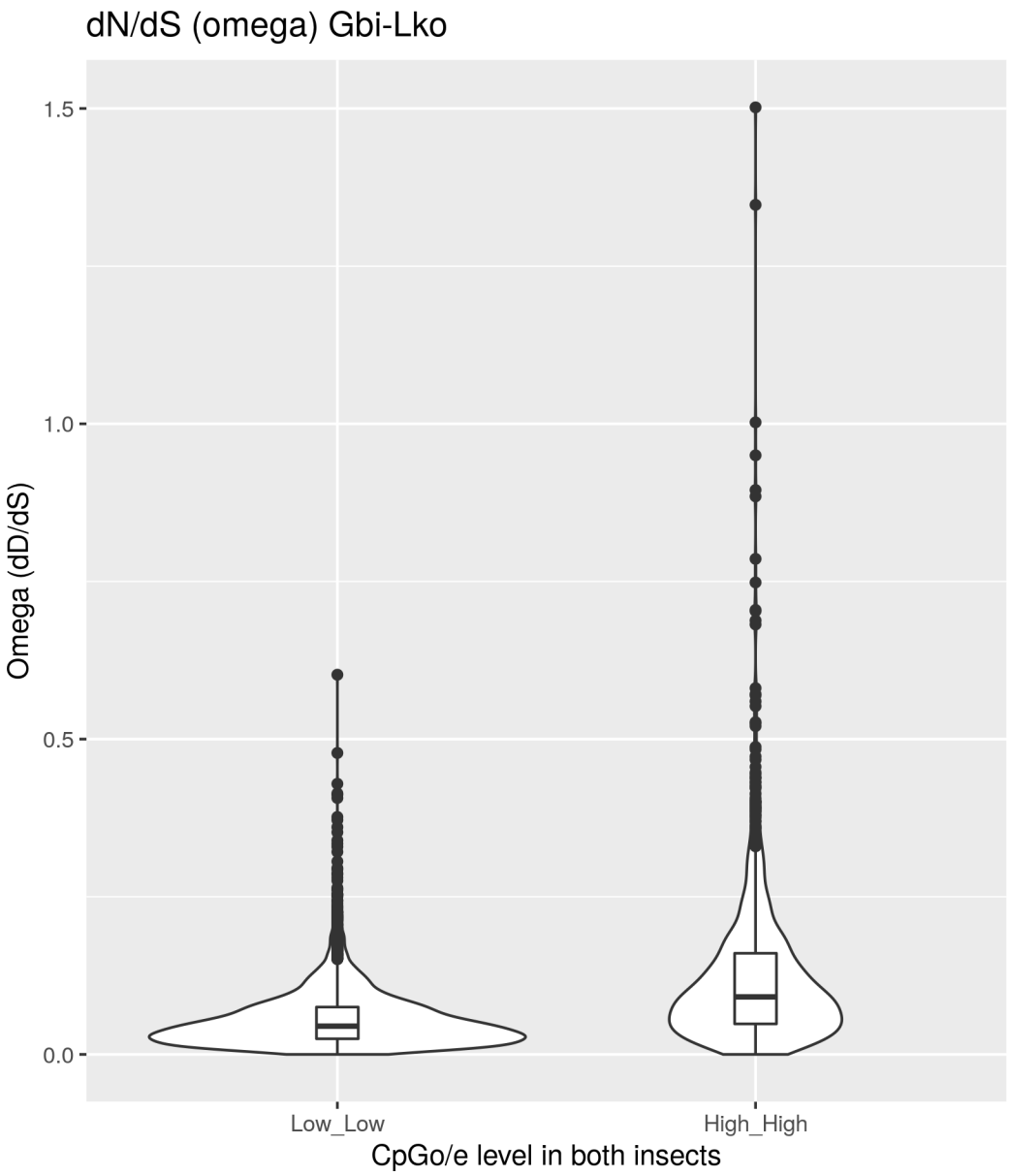

Blattella_germanica
0.816
0.564
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0

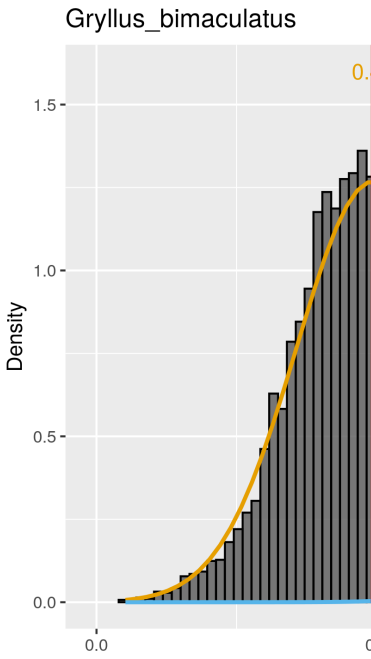

Bombyx_mori
0.868
1.063
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0

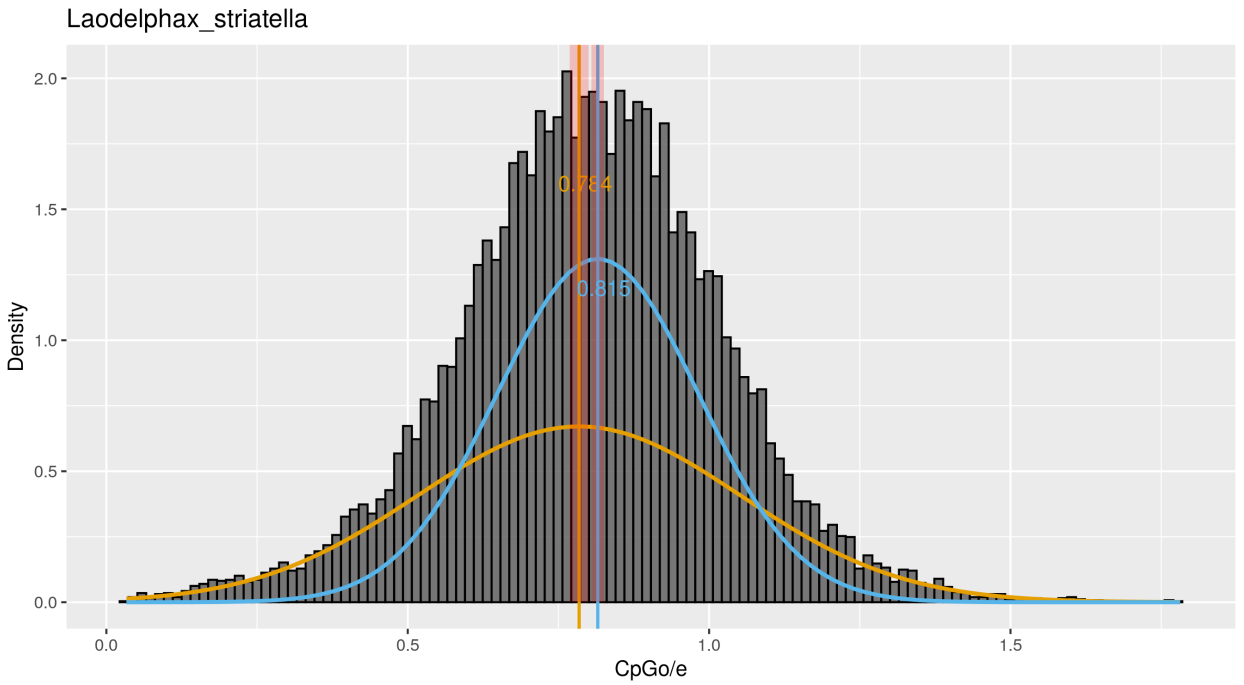

Cryptotermes_secundus
Density
CpGo/e
0.376
0.717
0.0
0.5
1.0
1.5
2.0
2.5

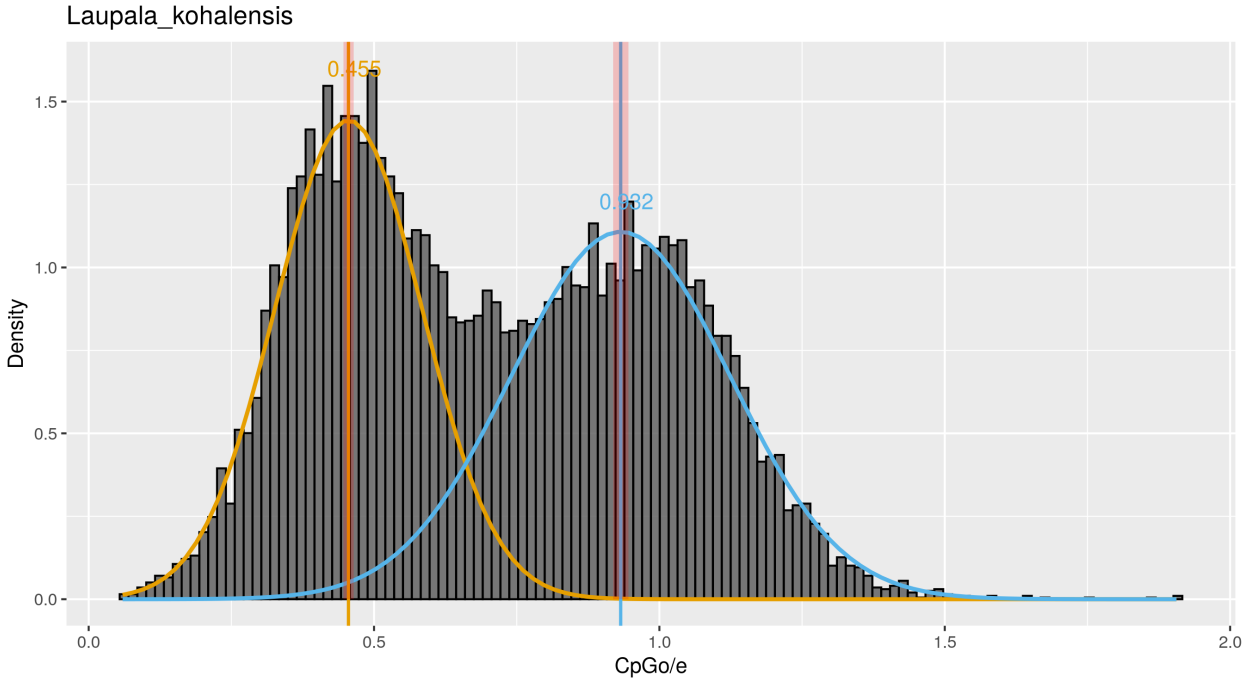

Drosophila_melanogaster
Density
CpGo/e
0.374
0.894
0
1
2
3
0.5
1.0
1.5

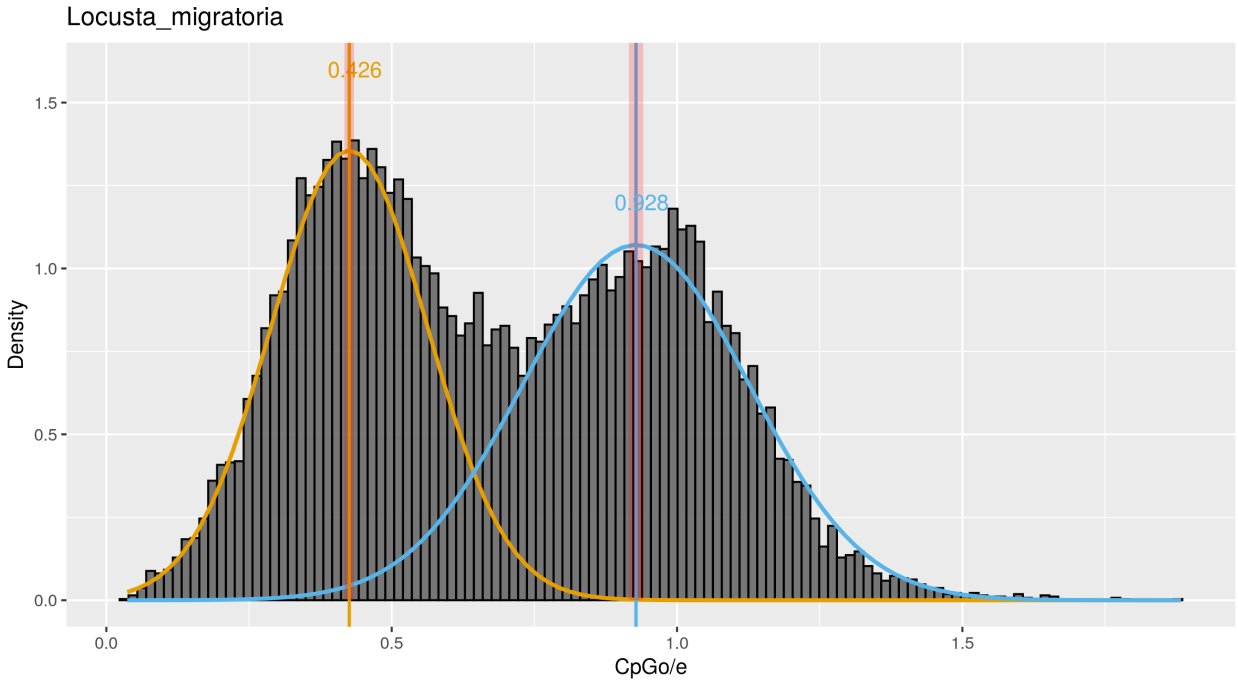

Frankliniella_occidentalis
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0
0.0
0.4
0.8
1.2
1.6
0.438
0.953

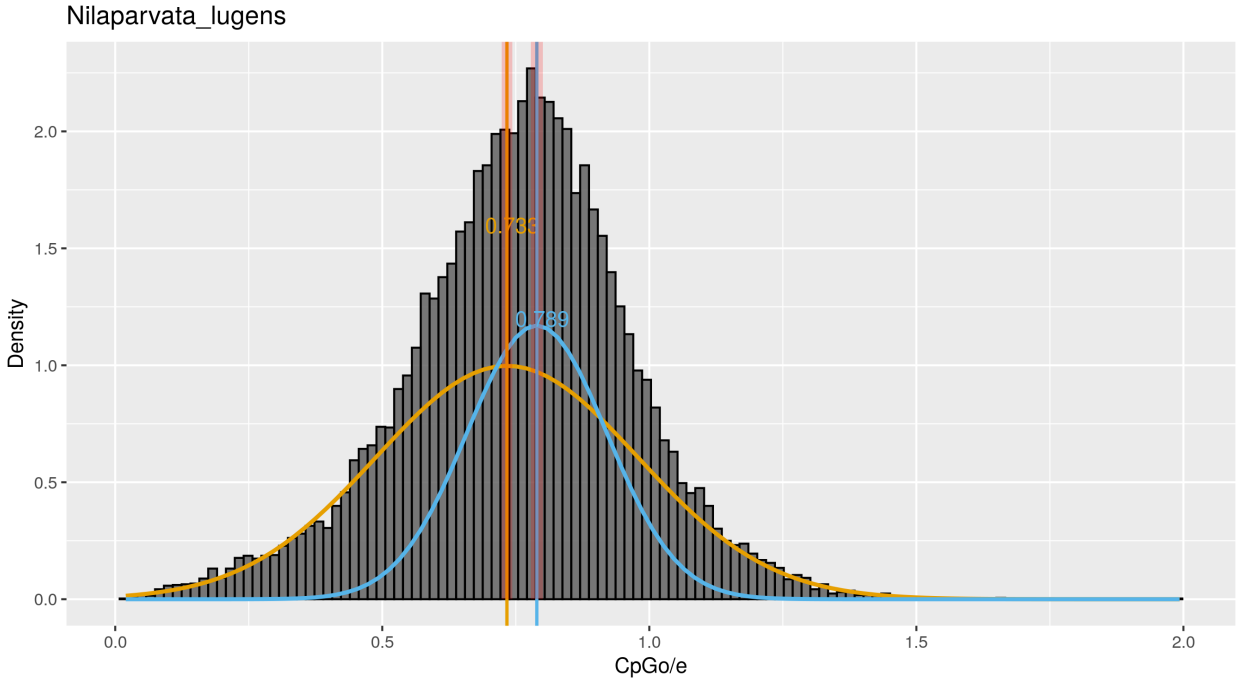

dN/dS (omega) Gbi-Lko
Omega (dD/dS)
1.5
1.0
0.5
0.0
Low_Low
High_High
CpGo/e level in both insects

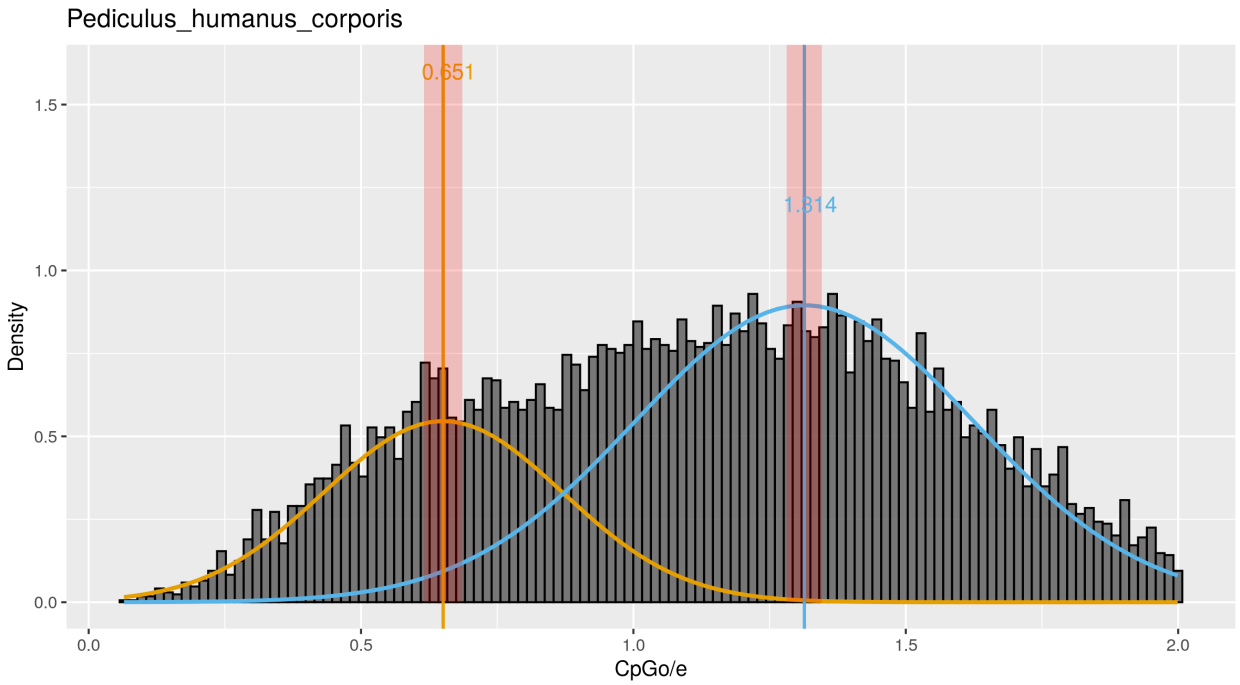

Gryllus_bimaculatus
Density
1.5
1.0
0.5
0.0
0.0

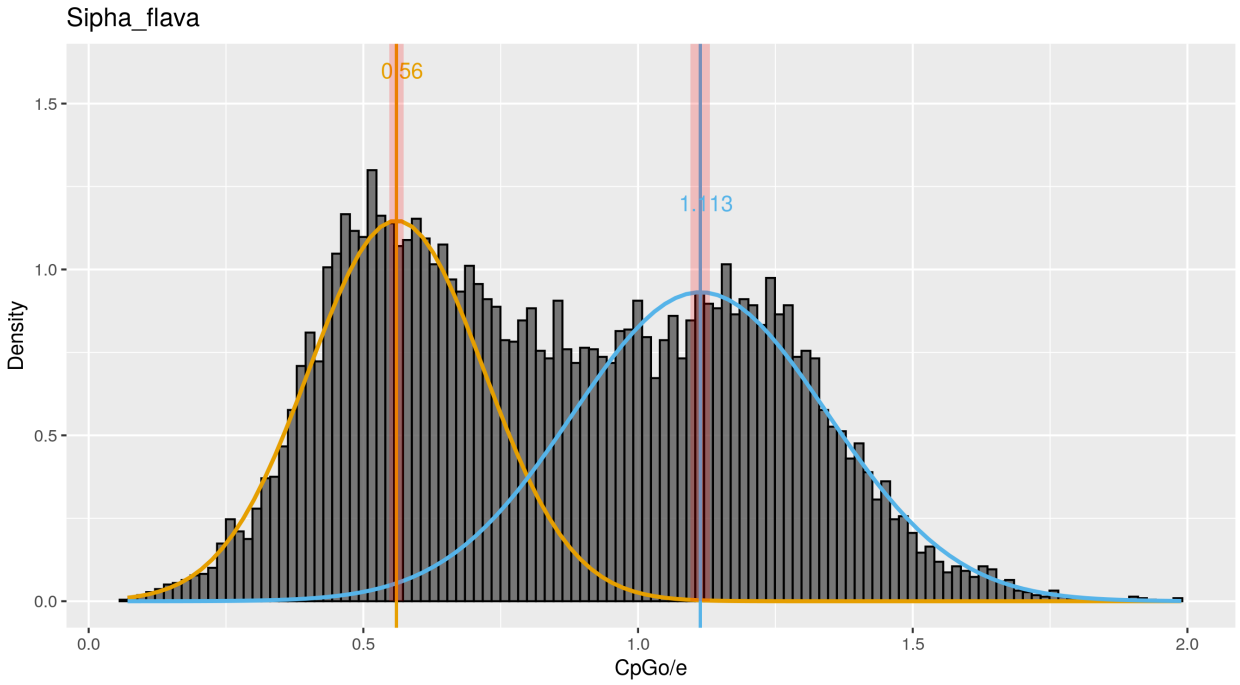

Laodelphax_striatella
0.784
0.815
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0

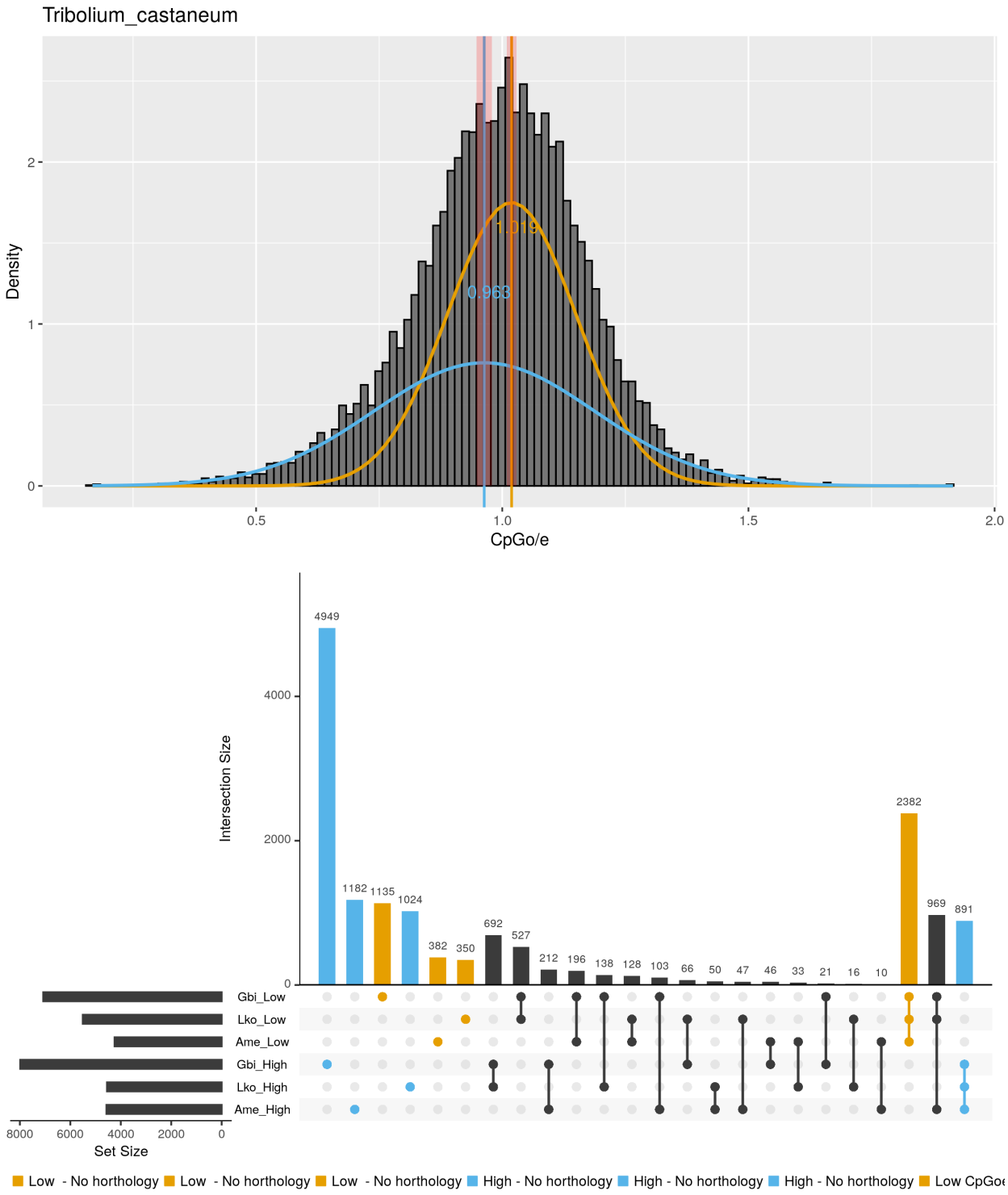

Laupala_kohalensis
0.455
0.932
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0

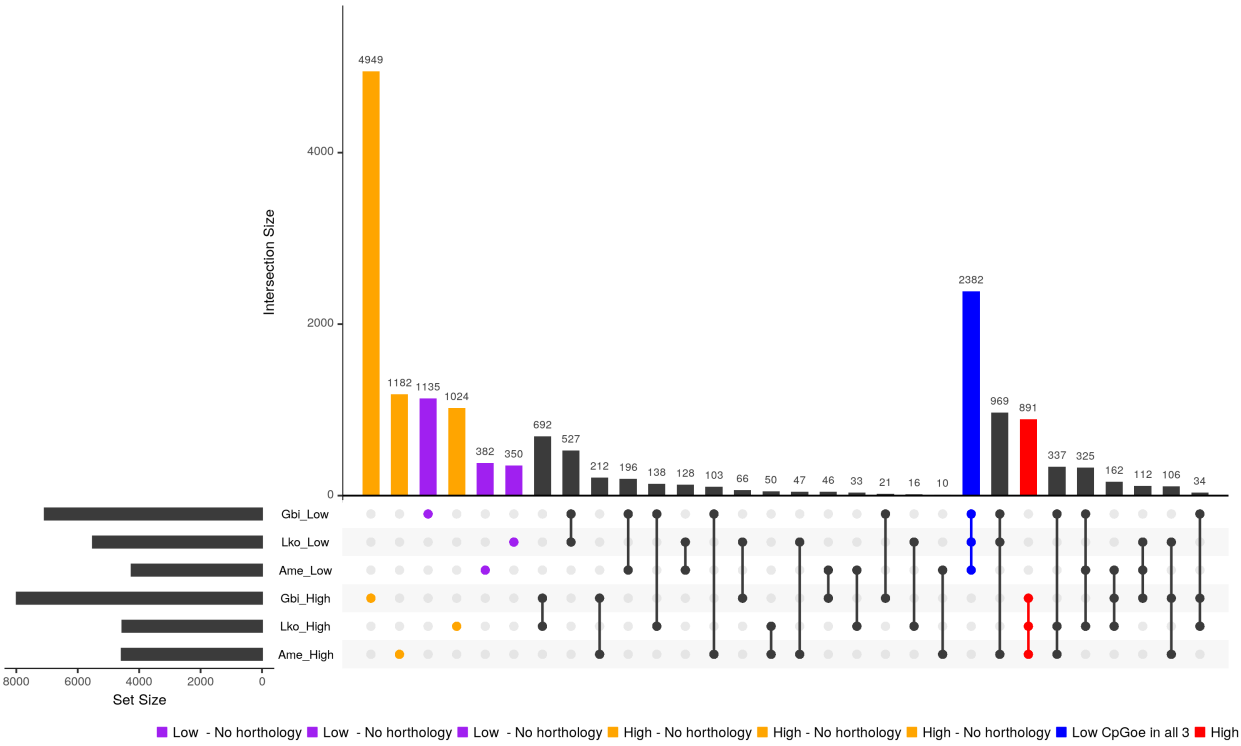

Locusta_migratoria
0.426
0.928
Density
CpGo/e
0.0
0.5
1.0
1.5


Nilaparvata_lugens
0.736
0.789
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0

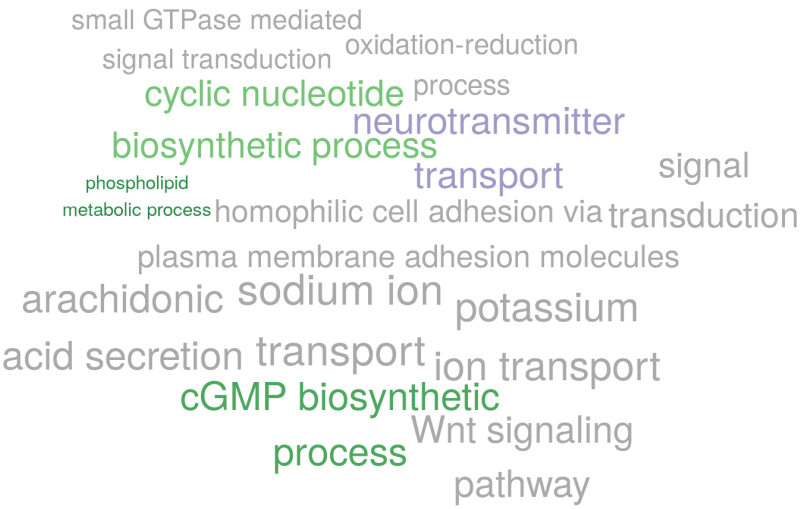

Pediculus_humanus_corporis
0.651
1.314
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0


Sipha_flava
0.56
1.113
Density
CpGo/e
0.0
0.5
1.0
1.5
2.0

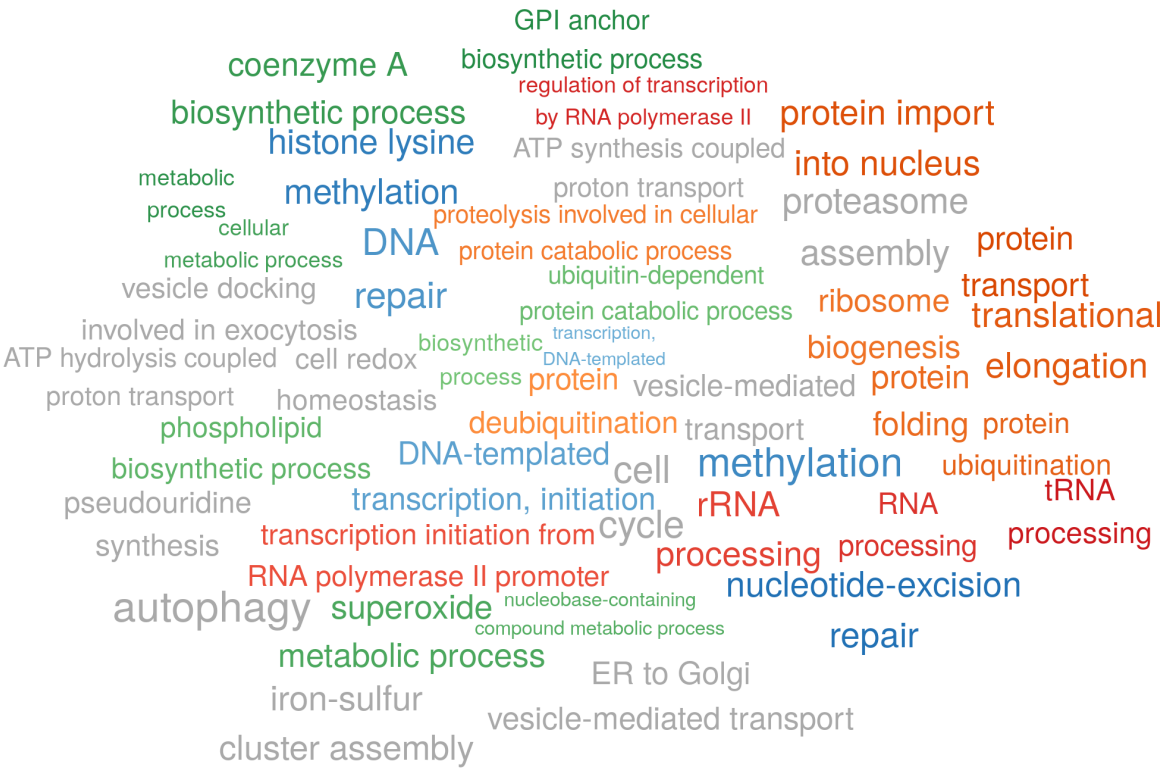

Tribolium_castaneum
Density
CpGo/e
1.019
0.963
Intersection Size
Set Size
Gbi_Low
Lko_Low
Ame_Low
Gbi_High
Lko_High
Ame_High
4949
1182
1135
1024
382
350
692
527
212
196
138
128
103
66
50
47
46
33
21
16
10
2382
969
891
Low - No horthology
Low - No horthology
Low - No horthology
High - No horthology
High - No horthology
High - No horthology
Low CpGo


Intersection Size
4000
2000
0
4949
1182
1135
1024
382
350
692
527
212
196
138
128
103
66
50
47
46
33
21
16
10
2382
969
891
337
325
162
112
106
34
Gbi_Low
Lko_Low
Ame_Low
Gbi_High
Lko_High
Ame_High
8000
6000
4000
2000
0
Set Size
Low - No horthology
Low - No horthology
Low - No horthology
High - No horthology
High - No horthology
High - No horthology
Low CpGoe in all 3
High

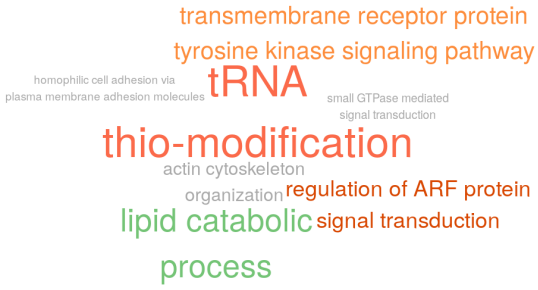
A. mellifera - High CpGoe



A. mellifera - Low CpGoe

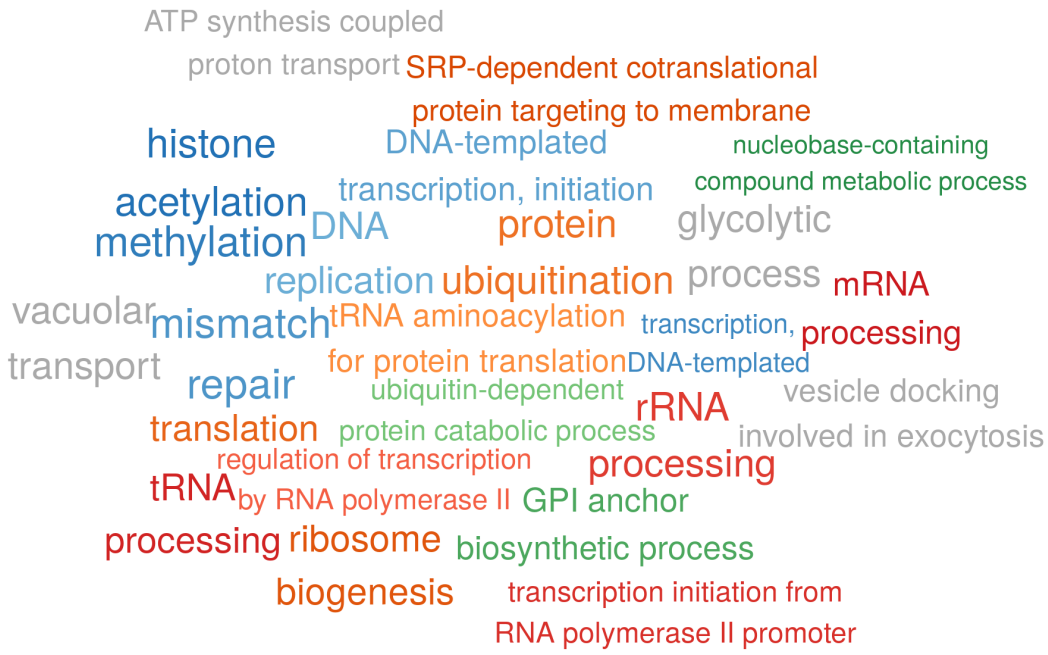


Crickets Low and Apis High CpGoe -- Functions form Gbi genes

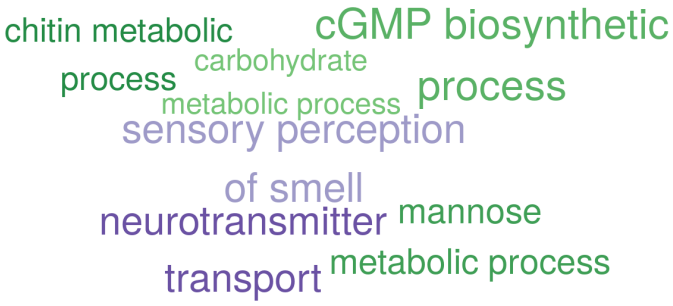


G. bimaculatus - High CpGoe

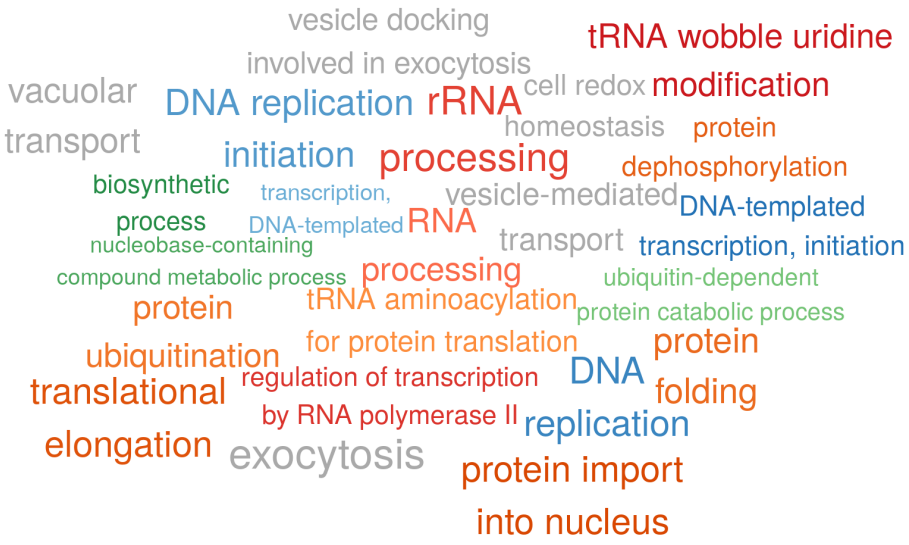
G. bimaculatus - Low CpGoe

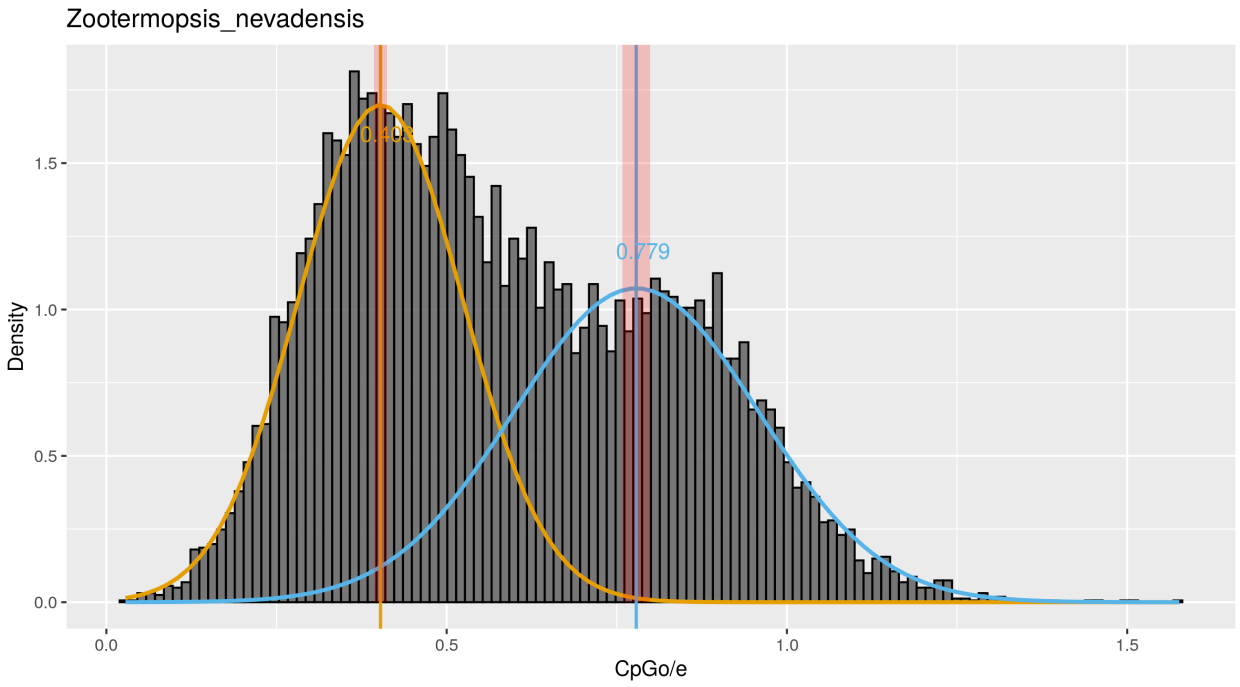


L. kohalensis - High CpGoe



L. kohalensis- Low CpGoe



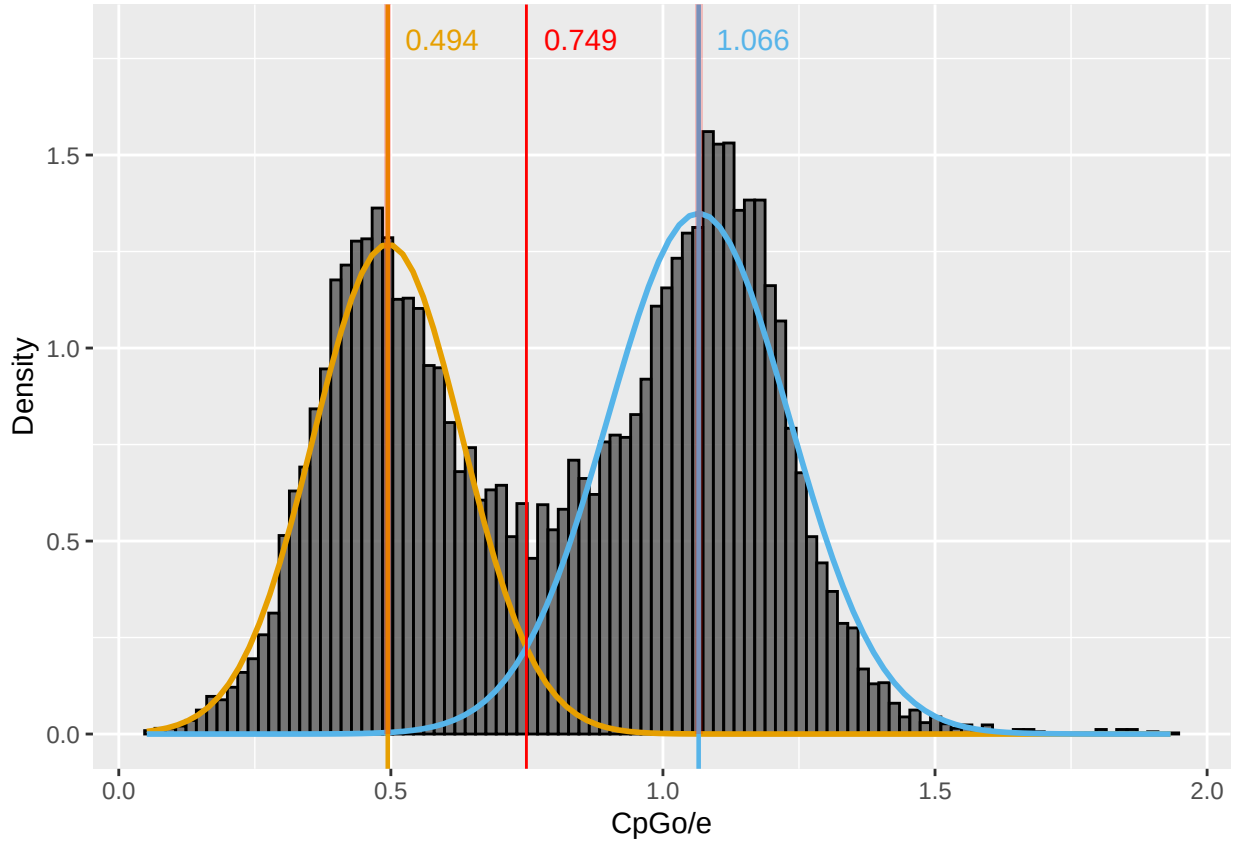


Gryllus bimaculatus CpGoe distribution

```
# Gbi_CpG<-CpGoe_insects[CpGoe_insects$Spp=='Gryllus_bimaculatus',]
Gbi_CpG <- CpGoe_insects_b02[CpGoe_insects_b02$Spp == "Gryllus_bimaculatus",
    ]

CpGoe_toplotGbi = Gbi_CpG$CpGoe
CpGoe_toplotGbi <- CpGoe_toplotGbi[!is.na(CpGoe_toplotGbi)]
mixmdlGbi = normalmixEM(CpGoe_toplotGbi, maxit = 20000)


## 1.96?? wikpedia: 95% confidence limits, where x ¯
## {\displaystyle {\bar {x}}} {\bar {x}} is equal to the
## sample mean, S E {\displaystyle SE} SE is equal to the
## standard error for the sample mean, and 1.96 is the
## approximate value of the 97.5 percentile point of the
## normal distribution: Lambda is the poportion of
## observations belonging to each distribution
## SE1=mixmdlGbi$sigma[1]/sqrt(length(CpGoe_toplotGbi)*mixmdlGbi$lambda[1])
## CI_1_right=mixmdlGbi$mu[1]+(SE1*1.96)
## CI_1_left=mixmdlGbi$mu[1]-(SE1*1.96)
## SE2=mixmdlGbi$sigma[2]/sqrt(length(CpGoe_toplotGbi)*mixmdlGbi$lambda[2])
## CI_2_right=mixmdlGbi$mu[2]+(SE2*1.96)
## CI_2_left=mixmdlGbi$mu[2]-(SE2*1.96)


# 95% 1.960 99% 2.57 99.5% 2.807

## Calculating SE using mixtools bootystrap function
se <- boot.se(mixmdlGbi, B = 100)
```

```
# confidence intervals
CI_1_right = mixmdlGbi$mu[1] + (se$mu.se[1] * 2.8)
CI_1_left = mixmdlGbi$mu[1] - (se$mu.se[1] * 2.8)

CI_2_right = mixmdlGbi$mu[2] + (se$mu.se[2] * 2.8)
CI_2_left = mixmdlGbi$mu[2] - (se$mu.se[2] * 2.8)


# plot(mixmdlGbi, breaks=100 ,which=2)
# lines(density(CpGoe_toplot), lty=2, lwd=2)


## In ggplots:
plot_mix_comps <- function(x, mu, sigma, lam) {
    lam * dnorm(x, mu, sigma)
}


Bimodal_Gbi <- data.frame(x = mixmdlGbi$x) %>% ggplot() + geom_histogram(aes(x,
    ..density..), colour = "black", alpha = 0.8, bins = 100) +
    stat_function(geom = "line", fun = plot_mix_comps, args = list(mixmdlGbi$mu[1],
        mixmdlGbi$sigma[1], lam = mixmdlGbi$lambda[1]), colour = "#E69F00",
        lwd = 1) + stat_function(geom = "line", fun = plot_mix_comps,
    args = list(mixmdlGbi$mu[2], mixmdlGbi$sigma[2], lam = mixmdlGbi$lambda[2]),
    colour = "#56B4E9", lwd = 1) + geom_vline(xintercept = mixmdlGbi$mu[1],
    col = "#E69F00", size = 0.8) + annotate("rect", xmin = CI_1_left,
    xmax = CI_1_right, ymin = 0, ymax = Inf, alpha = 0.2, fill = "red") +
    annotate("text", label = round(mixmdlGbi$mu[1], 3), y = 1.8,
        x = mixmdlGbi$mu[1] + 0.1, col = "#E69F00", size = 4) +
    geom_vline(xintercept = mixmdlGbi$mu[2], col = "#56B4E9",
        size = 0.8) + annotate("rect", xmin = CI_2_left, xmax = CI_2_right,
    ymin = 0, ymax = Inf, alpha = 0.2, fill = "red") + annotate("text",
    label = round(mixmdlGbi$mu[2], 3), y = 1.8, x = mixmdlGbi$mu[2] +
        0.1, col = "#56B4E9", size = 4) + # annotate('text',label=paste0('LogLik=',round(mixmdlGbi$logl
# 1)),y= 1.5, x=1.5, col='black',size=4)+
ylab("Density") + xlab("CpGo/e")


### I separate the 2 distributions by the lowest density
### between means gbidata<-ggplot_build(Bimodal_Gbi)$data[[1]]
### gbidata_int<-gbidata[gbidata$x>mixmdlGbi$mu[1] &
### gbidata$x<mixmdlGbi$mu[2] ,]
### Gbi_cpg_threshold<-gbidata_int[gbidata_int$y==min(gbidata_int$y),]$x
### red line spearateing
curve1 <- function(x) mixmdlGbi$lambda[1] * dnorm(x, mixmdlGbi$mu[1],
    mixmdlGbi$sigma[1])
curve2 <- function(x) mixmdlGbi$lambda[2] * dnorm(x, mixmdlGbi$mu[2],
    mixmdlGbi$sigma[2])

curveintersectGbi <- curve_intersect(curve1, curve2, empirical = FALSE,
    domain = c(0, 5))

Gbi_cpg_threshold = curveintersectGbi$x
```

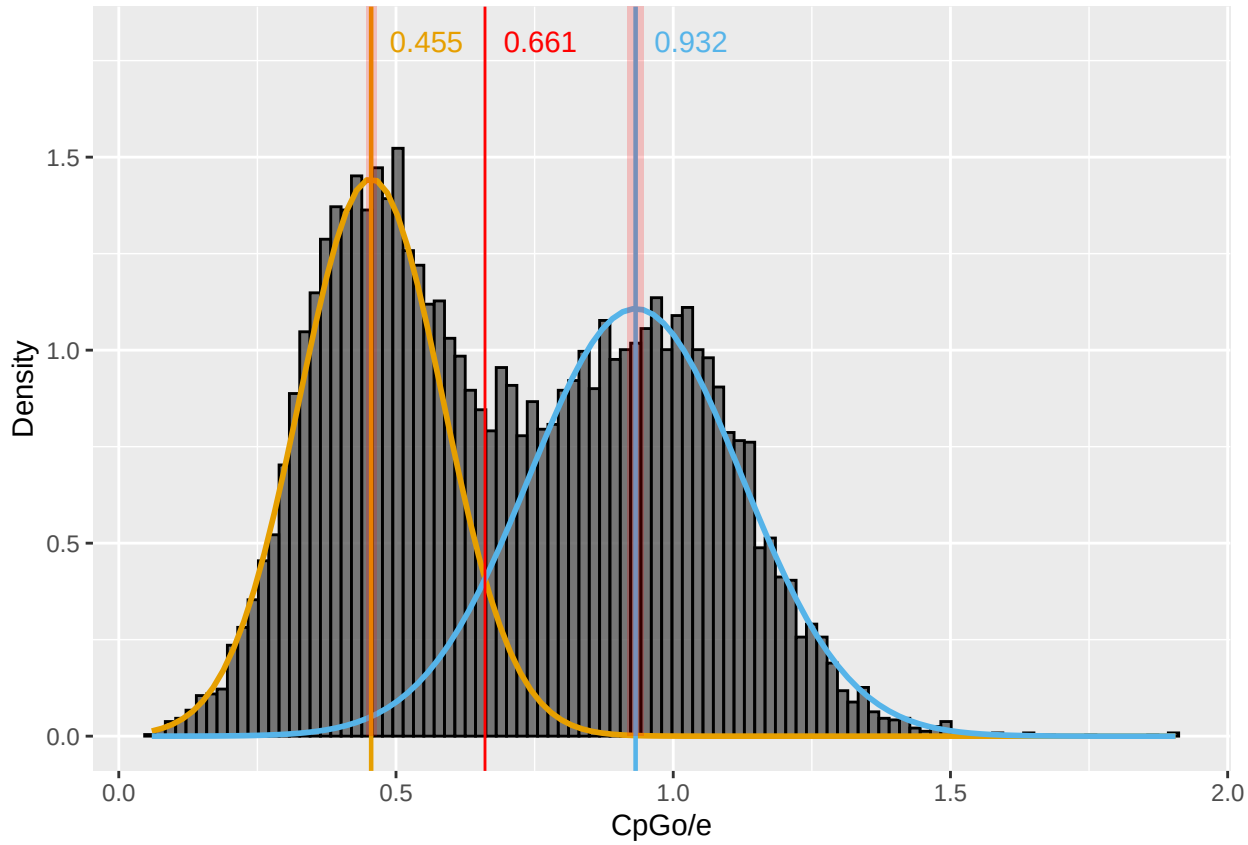
```
Bimodal_Gbi <- Bimodal_Gbi + geom_vline(xintercept = Gbi_cpg_threshold,
    col = "red", size = 0.5) + annotate("text", label = round(Gbi_cpg_threshold,
    3), y = 1.8, x = Gbi_cpg_threshold + 0.1, col = "red", size = 4)

Bimodal_Gbi
```



Laupala kohalensis CpGoe distribution

```
# Lko_CpG<-CpGoe_insects[CpGoe_insects$Spp=='Laupala_kohalensis',]
Lko_CpG <- CpGoe_insects_b02[CpGoe_insects_b02$Spp == "Laupala_kohalensis",
    ]

CpGoe_toplotLko = Lko_CpG$CpGoe
CpGoe_toplotLko <- CpGoe_toplotLko[!is.na(CpGoe_toplotLko)]
mixmdlLko = normalmixEM(CpGoe_toplotLko, maxit = 20000)

## Calculating SE using mixtools bootystrap function
se <- boot.se(mixmdlLko, B = 100)
# confidence intervals
CI_1_right = mixmdlLko$mu[1] + (se$mu.se[1] * 2.8)
CI_1_left = mixmdlLko$mu[1] - (se$mu.se[1] * 2.8)

CI_2_right = mixmdlLko$mu[2] + (se$mu.se[2] * 2.8)
```

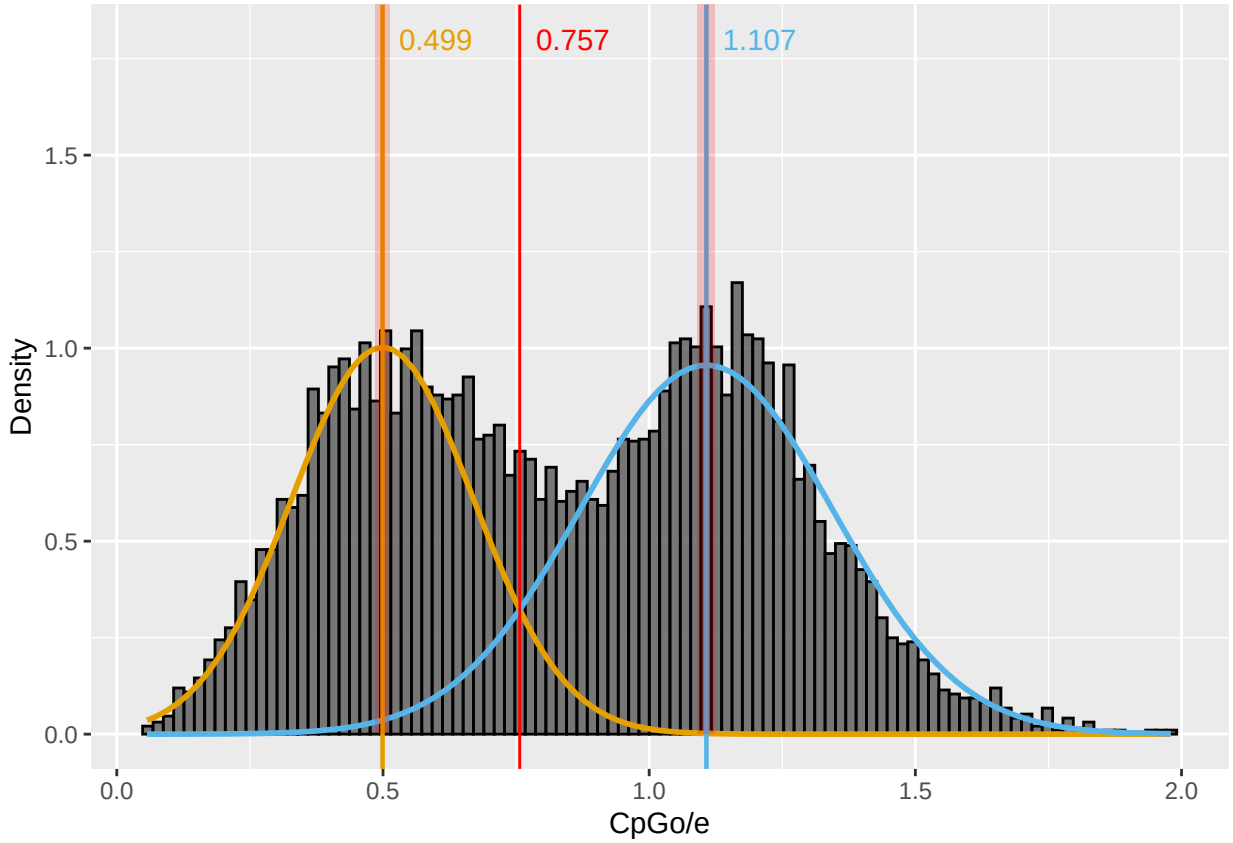
```
CI_2_left = mixmdlLko$mu[2] - (se$mu.se[2] * 2.8)

## In ggplots:
plot_mix_comps <- function(x, mu, sigma, lam) {
    lam * dnorm(x, mu, sigma)
}


Bimodal_Lko <- data.frame(x = mixmdlLko$x) %>% ggplot() + geom_histogram(aes(x,
    ..density..), colour = "black", alpha = 0.8, bins = 100) +
    stat_function(geom = "line", fun = plot_mix_comps, args = list(mixmdlLko$mu[1],
        mixmdlLko$sigma[1], lam = mixmdlLko$lambda[1]), colour = "#E69F00",
        lwd = 1) + stat_function(geom = "line", fun = plot_mix_comps,
    args = list(mixmdlLko$mu[2], mixmdlLko$sigma[2], lam = mixmdlLko$lambda[2]),
    colour = "#56B4E9", lwd = 1) + geom_vline(xintercept = mixmdlLko$mu[1],
    col = "#E69F00", size = 0.8) + annotate("rect", xmin = CI_1_left,
    xmax = CI_1_right, ymin = 0, ymax = Inf, alpha = 0.2, fill = "red") +
    annotate("text", label = round(mixmdlLko$mu[1], 3), y = 1.8,
        x = mixmdlLko$mu[1] + 0.1, col = "#E69F00", size = 4) +
    geom_vline(xintercept = mixmdlLko$mu[2], col = "#56B4E9",
        size = 0.8) + annotate("rect", xmin = CI_2_left, xmax = CI_2_right,
    ymin = 0, ymax = Inf, alpha = 0.2, fill = "red") + annotate("text",
    label = round(mixmdlLko$mu[2], 3), y = 1.8, x = mixmdlLko$mu[2] +
        0.1, col = "#56B4E9", size = 4) + # annotate('text',label=paste0('LogLik=',round(mixmdlLko$logl
# 1)),y= 1.5, x=1.5, col='black',size=4)+
ylab("Density") + xlab("CpGo/e")


### I separate the 2 distributions by the lowest density
### between means lkodata<-ggplot_build(Bimodal_Lko)$data[[1]]
### lkodata_int<-lkodata[lkodata$x>mixmdlLko$mu[1] &
### lkodata$x<mixmdlLko$mu[2] ,]
### Lko_cpg_threshold<-lkodata_int[lkodata_int$y==min(lkodata_int$y),]$x
### red line spearateing

curve1 <- function(x) mixmdlLko$lambda[1] * dnorm(x, mixmdlLko$mu[1],
    mixmdlLko$sigma[1])
curve2 <- function(x) mixmdlLko$lambda[2] * dnorm(x, mixmdlLko$mu[2],
    mixmdlLko$sigma[2])

curveintersectLko <- curve_intersect(curve1, curve2, empirical = FALSE,
    domain = c(0, 5))

Lko_cpg_threshold = curveintersectLko$x


Bimodal_Lko <- Bimodal_Lko + geom_vline(xintercept = Lko_cpg_threshold,
    col = "red", size = 0.5) + annotate("text", label = round(Lko_cpg_threshold,
    3), y = 1.8, x = Lko_cpg_threshold + 0.1, col = "red", size = 4)

Bimodal_Lko
```



Apis mellifera CpGoe distribution

```
# Ame_CpG<-CpGoe_insects[CpGoe_insects$Spp=='Apis_mellifera',]
Ame_CpG <- CpGoe_insects_b02[CpGoe_insects_b02$Spp == "Apis_mellifera",
    ]

CpGoe_toplotAme = Ame_CpG$CpGoe
CpGoe_toplotAme <- CpGoe_toplotAme[!is.na(CpGoe_toplotAme)]
mixmdlAme = normalmixEM(CpGoe_toplotAme, maxit = 20000)

## Calculating SE using mixtools bootystrap function
se <- boot.se(mixmdlAme, B = 100)
# confidence intervals
CI_1_right = mixmdlAme$mu[1] + (se$mu.se[1] * 2.8)
CI_1_left = mixmdlAme$mu[1] - (se$mu.se[1] * 2.8)

CI_2_right = mixmdlAme$mu[2] + (se$mu.se[2] * 2.8)
CI_2_left = mixmdlAme$mu[2] - (se$mu.se[2] * 2.8)

## In ggplots:
plot_mix_comps <- function(x, mu, sigma, lam) {
    lam * dnorm(x, mu, sigma)
}
```

```
Bimodal_Ame <- data.frame(x = mixmdlAme$x) %>% ggplot() + geom_histogram(aes(x,
    ..density..), colour = "black", alpha = 0.8, bins = 100) +
    stat_function(geom = "line", fun = plot_mix_comps, args = list(mixmdlAme$mu[1],
        mixmdlAme$sigma[1], lam = mixmdlAme$lambda[1]), colour = "#E69F00",
        lwd = 1) + stat_function(geom = "line", fun = plot_mix_comps,
    args = list(mixmdlAme$mu[2], mixmdlAme$sigma[2], lam = mixmdlAme$lambda[2]),
    colour = "#56B4E9", lwd = 1) + geom_vline(xintercept = mixmdlAme$mu[1],
    col = "#E69F00", size = 0.8) + annotate("rect", xmin = CI_1_left,
    xmax = CI_1_right, ymin = 0, ymax = Inf, alpha = 0.2, fill = "red") +
    annotate("text", label = round(mixmdlAme$mu[1], 3), y = 1.8,
        x = mixmdlAme$mu[1] + 0.1, col = "#E69F00", size = 4) +
    geom_vline(xintercept = mixmdlAme$mu[2], col = "#56B4E9",
        size = 0.8) + annotate("rect", xmin = CI_2_left, xmax = CI_2_right,
    ymin = 0, ymax = Inf, alpha = 0.2, fill = "red") + annotate("text",
    label = round(mixmdlAme$mu[2], 3), y = 1.8, x = mixmdlAme$mu[2] +
        0.1, col = "#56B4E9", size = 4) + # annotate('text',label=paste0('LogLik=',round(mixmdlAme$logl
# 1)),y= 1.5, x=1.5, col='black',size=4)+
ylab("Density") + xlab("CpGo/e")


### I separate the 2 distributions by the lowest density
### between means amedata<-ggplot_build(Bimodal_Ame)$data[[1]]
### amedata_int<-amedata[amedata$x>mixmdlAme$mu[1] &
### amedata$x<mixmdlAme$mu[2] ,]
### Ame_cpg_threshold<-amedata_int[amedata_int$y==min(amedata_int$y),]$x


# red line spearateing

curve1 <- function(x) mixmdlAme$lambda[1] * dnorm(x, mixmdlAme$mu[1],
    mixmdlAme$sigma[1])
curve2 <- function(x) mixmdlAme$lambda[2] * dnorm(x, mixmdlAme$mu[2],
    mixmdlAme$sigma[2])

curveintersectAme <- curve_intersect(curve1, curve2, empirical = FALSE,
    domain = c(0, 5))

Ame_cpg_threshold = curveintersectAme$x
Bimodal_Ame <- Bimodal_Ame + geom_vline(xintercept = Ame_cpg_threshold,
    col = "red", size = 0.5) + annotate("text", label = round(Ame_cpg_threshold,
    3), y = 1.8, x = Ame_cpg_threshold + 0.1, col = "red", size = 4)

Bimodal_Ame
```



Crickets CpGoe high vs low

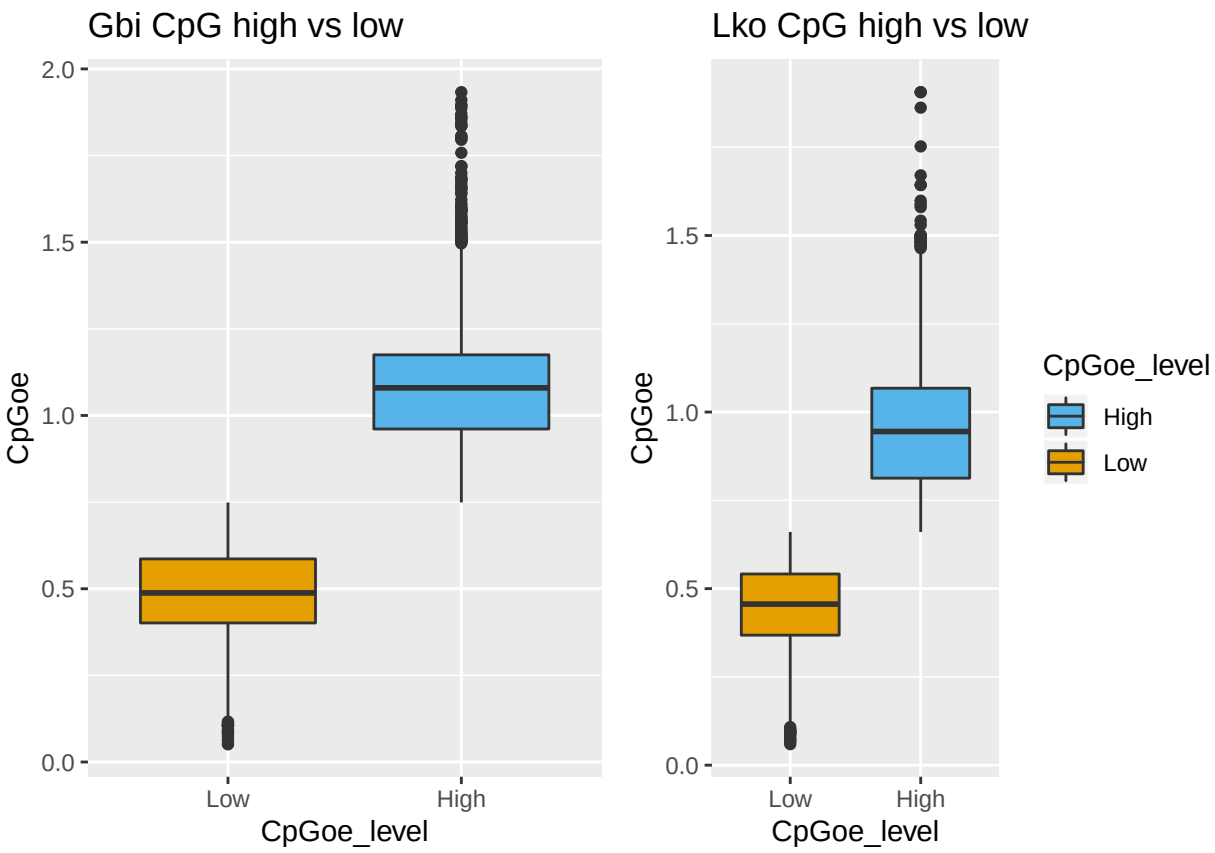
```
# Gbi_CpG<-CpGoe_insects[CpGoe_insects$Spp=='Gryllus_bimaculatus',]
Gbi_CpG <- CpGoe_insects_b02[CpGoe_insects_b02$Spp == "Gryllus_bimaculatus",
    ]

Gbi_CpG$CpGoe_level <- "NA"
Gbi_CpG$CpGoe_level[Gbi_CpG$CpGoe > Gbi_cpg_threshold] <- "High"
Gbi_CpG$CpGoe_level[Gbi_CpG$CpGoe <= Gbi_cpg_threshold] <- "Low"

Lko_CpG <- CpGoe_insects_b02[CpGoe_insects_b02$Spp == "Laupala_kohalensis",
    ]

Lko_CpG$CpGoe_level <- "NA"
Lko_CpG$CpGoe_level[Lko_CpG$CpGoe > Lko_cpg_threshold] <- "High"
Lko_CpG$CpGoe_level[Lko_CpG$CpGoe <= Lko_cpg_threshold] <- "Low"
```

```
## sort levels: 1st Low, then High
Lko_CpG$CpGoe_level <- factor(Lko_CpG$CpGoe_level, levels = c("Low",
    "High"))
Gbi_CpG$CpGoe_level <- factor(Gbi_CpG$CpGoe_level, levels = c("Low",
    "High"))


BoxplotGbi <- ggplot(Gbi_CpG, aes(x = CpGoe_level, y = CpGoe,
```

```
    fill = CpGoe_level)) + geom_boxplot() + ggtitle("Gbi CpG high vs low") +
    scale_fill_manual(breaks = c("High", "Low"), values = c("#E69F00",
        "#56B4E9"))


BoxplotLko <- ggplot(Lko_CpG, aes(x = CpGoe_level, y = CpGoe,
    fill = CpGoe_level)) + geom_boxplot() + ggtitle("Lko CpG high vs low") +
    scale_fill_manual(breaks = c("High", "Low"), values = c("#E69F00",
        "#56B4E9"))

prop.table(table(Gbi_CpG$CpGoe_level)) * 100
```

```
##
##      Low     High
## 43.79599 56.20401
```

```
# summary(Gbi_CpG[Gbi_CpG$CpGoe_level=='High', 'CpGoe'])
# summary(Gbi_CpG[Gbi_CpG$CpGoe_level=='Low', 'CpGoe'])
# t.test(Gbi_CpG[Gbi_CpG$CpGoe_level=='High',
# 'CpGoe'],Gbi_CpG[Gbi_CpG$CpGoe_level=='Low', 'CpGoe'])
grid.arrange(BoxplotGbi + theme(legend.position = "none"), BoxplotLko,
    ncol = 2)
```



Crickets CpGoe vs Splicing

Is there any relation between CpGoe and Number of isoforms?

```
############################## Gryllus Get all the protein-coding genes from the SQL
############################## database
conGbi <- dbConnect(RSQLite::SQLite(), "/home/guillem/data_disk/Cricket_genome_annotation/GBI_Genome_v3_
transcripts_dbGbi <- dbFetch(dbSendQuery(conGbi, "SELECT TranscriptID, GeneID FROM Transcripts"))
dim(transcripts_dbGbi)
```

```
## [1] 28529     2
```

```
dbDisconnect(conGbi)
```

```
## Warning in connection_release(conn@ptr): There are 1 result in use. The
## connection will be released when they are closed
```

```
# Gbi_Gene_Isonumber<-aggregate(transcripts_dbGbi, by
# =list(transcripts_dbGbi$'GeneID'), length)
Gbi_Gene_Isonumber <- aggregate(. ~ GeneID, data = transcripts_dbGbi,
    length)
colnames(Gbi_Gene_Isonumber) <- c("GeneID", "Transcripts")
# head(Gbi_Gene_Isonumber)


############################## Laupala
conLko <- dbConnect(RSQLite::SQLite(), "/home/guillem/data_disk/Cricket_genome_annotation/Laupala_kohal
transcripts_dbLko <- dbFetch(dbSendQuery(conLko, "SELECT TranscriptID, GeneID FROM Transcripts"))
dim(transcripts_dbLko)
```

```
## [1] 13078     2
```

```
dbDisconnect(conLko)
```

```
## Warning in connection_release(conn@ptr): There are 1 result in use. The
## connection will be released when they are closed
```

```
Lko_Gene_Isonumber <- aggregate(. ~ GeneID, data = transcripts_dbLko,
    length)
colnames(Lko_Gene_Isonumber) <- c("GeneID", "Transcripts")
```

```
# dim(Gbi_Gene_Isonumber)
dim(Gbi_CpG)  # few less bc no eros, nas and >3
```

```
## [1] 17803     9
```

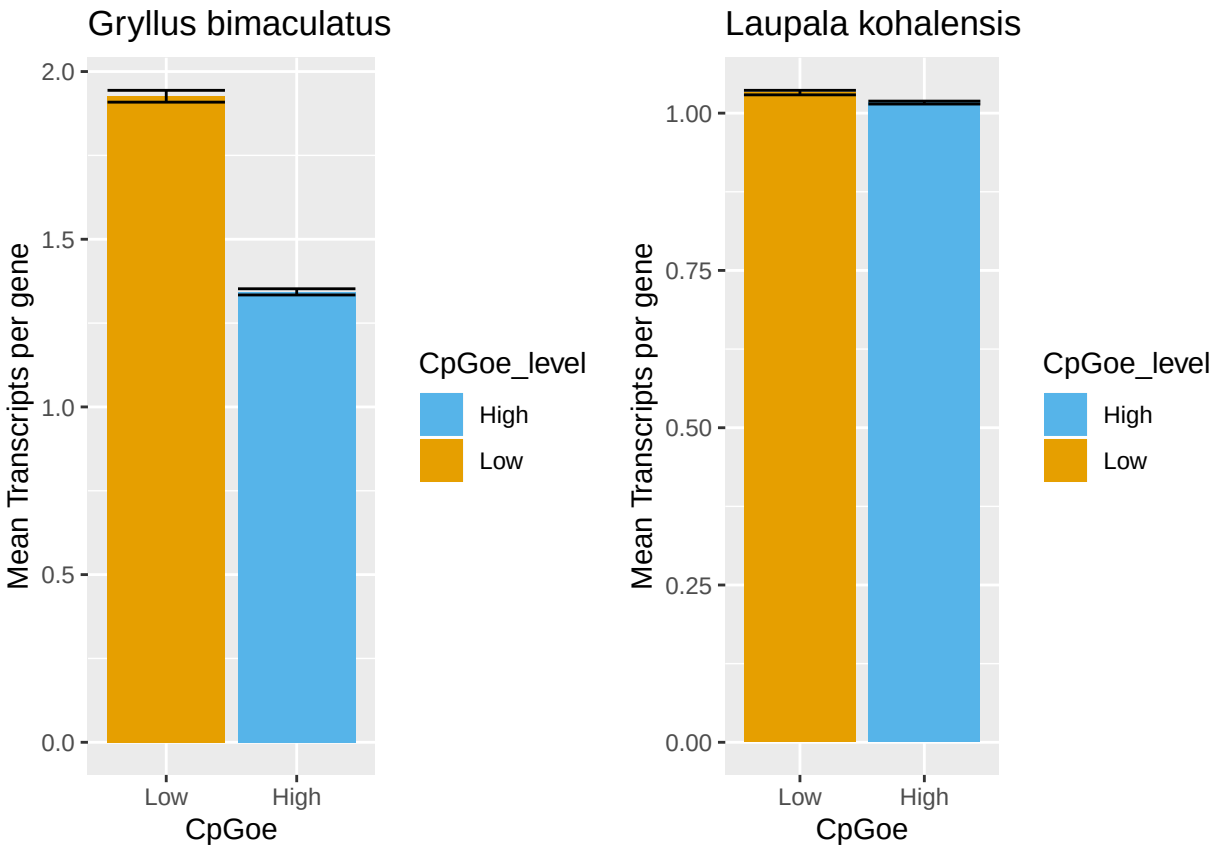
```
Gbi_CpG$Gene <- sapply(strsplit(as.character(Gbi_CpG$ID), "-"),
    `[`, 1)
Gbi_CpG_isoforms <- merge(Gbi_CpG, Gbi_Gene_Isonumber, by.x = "Gene",
    by.y = "GeneID")
# dim(Gbi_CpG_isoforms)


dim(Lko_Gene_Isonumber)
```

```
## [1] 12767     2
```

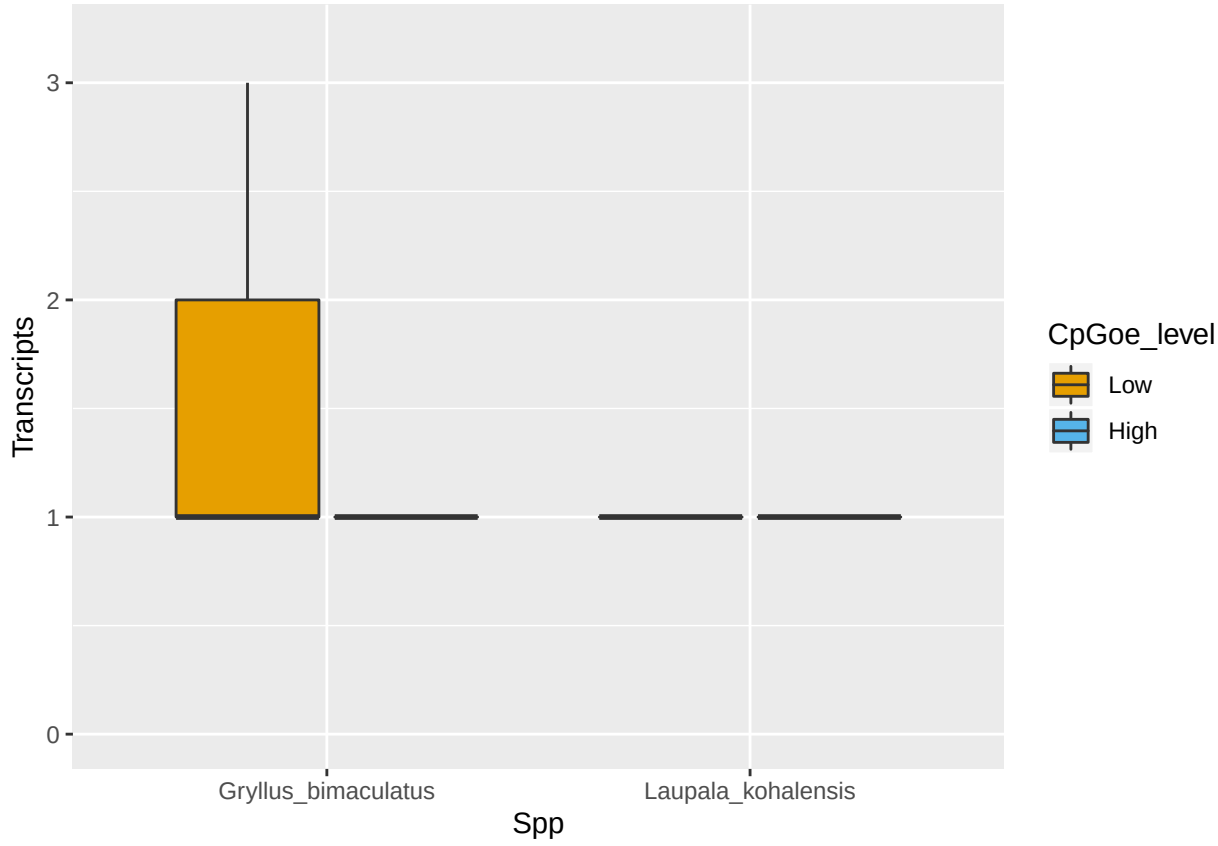
```
dim(Lko_CpG)  # few less bc no eros, nas and >3
```

```
## [1] 12745     9
```

```
Lko_CpG$Gene <- sapply(strsplit(as.character(Lko_CpG$ID), "-"),
    `[`, 1)
Lko_CpG_isoforms <- merge(Lko_CpG, Lko_Gene_Isonumber, by.x = "Gene",
    by.y = "GeneID")



BarplotGbi_CpG_isoform <- ggplot(Gbi_CpG_isoforms, aes(x = CpGoe_level,
    y = Transcripts, fill = CpGoe_level)) + geom_bar(position = "stack",
    stat = "summary", fun.y = "mean") + stat_summary(geom = "errorbar",
    fun.data = mean_se, position = "dodge") + xlab("CpGoe") +
    ylab("Mean Transcripts per gene") + ggtitle("Gryllus bimaculatus") +
    scale_fill_manual(breaks = c("High", "Low"), values = c("#E69F00",
        "#56B4E9"))




# t.test(Gbi_CpG_isoforms[Gbi_CpG_isoforms$CpGoe_level=='High',
# 'Transcripts'],Gbi_CpG_isoforms[Gbi_CpG_isoforms$CpGoe_level=='Low',
# 'Transcripts'])


## Lko
BarplotLko_CpG_isoform <- ggplot(Lko_CpG_isoforms, aes(x = CpGoe_level,
    y = Transcripts, fill = CpGoe_level)) + geom_bar(position = "dodge",
    stat = "summary", fun.y = "mean") + stat_summary(geom = "errorbar",
    fun.data = mean_se, position = "dodge") + xlab("CpGoe") +
    ylab("Mean Transcripts per gene") + ggtitle("Laupala kohalensis") +
    scale_fill_manual(breaks = c("High", "Low"), values = c("#E69F00",
        "#56B4E9"))



# t.test(Gbi_CpG_isoforms[Lko_CpG_isoforms$CpGoe_level=='High',
# 'Transcripts'],Lko_CpG_isoforms[Lko_CpG_isoforms$CpGoe_level=='Low',
# 'Transcripts']) aggregate(Transcripts ~ CpGoe_level , data
# = Gbi_CpG_isoforms, length) aggregate(Transcripts ~
# CpGoe_level , data = Gbi_CpG_isoforms, mean)

grid.arrange(BarplotGbi_CpG_isoform, BarplotLko_CpG_isoform,
    ncol = 2)
```



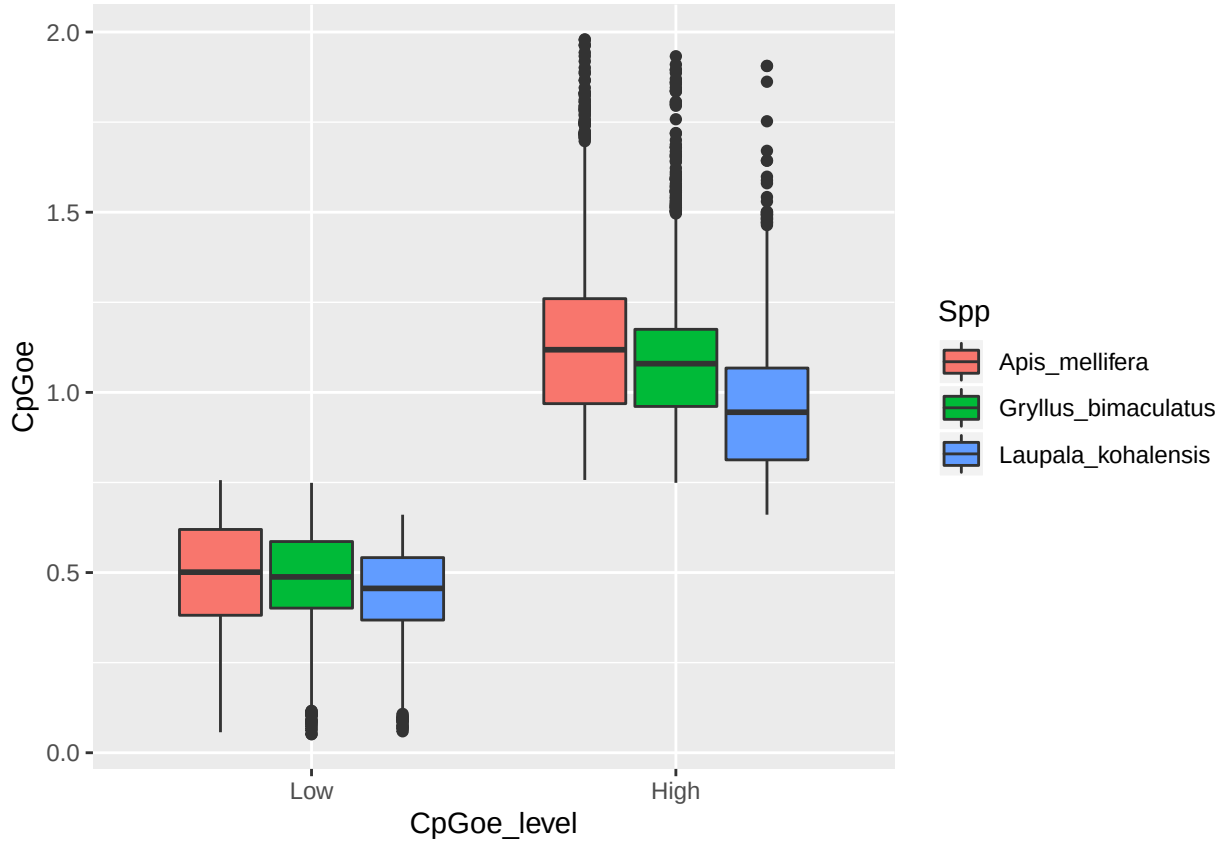

```
Crickets_CpG_isoforms <- rbind(Gbi_CpG_isoforms, Lko_CpG_isoforms)
## sort: 1st Low, then High
Crickets_CpG_isoforms$CpGoe_level <- factor(Crickets_CpG_isoforms$CpGoe_level,
    levels = c("Low", "High"))

BoxPlotCpg <- ggplot(Crickets_CpG_isoforms, aes(x = Spp, y = Transcripts,
    fill = CpGoe_level)) + geom_boxplot(outlier.shape = NA) +
    coord_cartesian(ylim = c(0, 3.2)) + scale_fill_manual(breaks = c("Low",
    "High"), values = c("#E69F00", "#56B4E9"))

BoxPlotCpg
```



CpGoe vs Orthology

Is there any relation between CpGoe and number of orthologs?

```
Ame_CpG <- CpGoe_insects_b02[CpGoe_insects_b02$Spp == "Apis_mellifera",
    ]

Ame_CpG$CpGoe_level <- "NA"
Ame_CpG$CpGoe_level[Ame_CpG$CpGoe > Ame_cpg_threshold] <- "High"
Ame_CpG$CpGoe_level[Ame_CpG$CpGoe <= Ame_cpg_threshold] <- "Low"

Ame_CpG$Gene <- sapply(strsplit(as.character(Ame_CpG$ID), "-"),
    `[`, 2)


AmeGbiLko_insects_CpGeo <- rbind(Ame_CpG, Lko_CpG, Gbi_CpG)

## reorder: 1st low, then high
AmeGbiLko_insects_CpGeo$CpGoe_level <- factor(AmeGbiLko_insects_CpGeo$CpGoe_level,
    levels = c("Low", "High"))



ggplot(AmeGbiLko_insects_CpGeo, aes(x = CpGoe_level, y = CpGoe,
    fill = Spp)) + geom_boxplot()
```



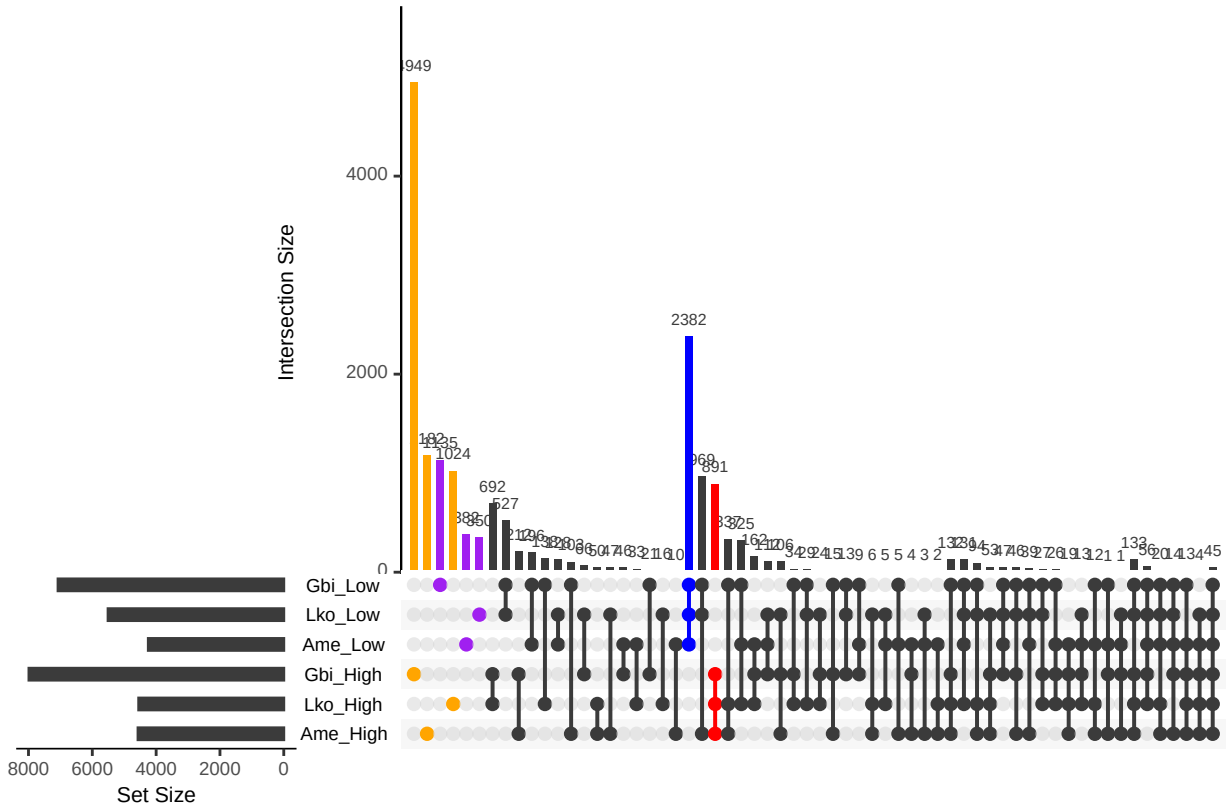

```
##### I need to transform OG table to to 2 coumn OG-gene
## Make a copy of Orthogroups.txt file
# from shutil import copyfile
# copyfile("/home/guillem/data_disk/Cricket_genome_annotation/Comparative_Genomics_V2/Orthofinder/Ortho

Gene_OG_dict = {}
with open("/home/guillem/WorkingDir/CpGoe/CpGoe_outputs/Orthogroups.txt") as f:
    for line in f:
       line=line.rstrip()
       (OG, genes) = line.split(":")
       geneeList=genes.split()
       Gene_OG_dict[OG] = geneeList

fout= open("/home/guillem/WorkingDir/CpGoe/CpGoe_outputs/Orthogroups_list_OG_gene.txt","w+")
for key in Gene_OG_dict:
  for value in Gene_OG_dict[key]:
    fout.write(key+"\t"+value+"\n")

fout.close()
```

```
OG_Gene <- read.delim("/home/guillem/WorkingDir/CpGoe/CpGoe_outputs/Orthogroups_list_OG_gene.txt",
    header = FALSE)
length(unique(OG_Gene$V1))  # should be 59516 (numbe rof OGs)
```

```
## [1] 59516
```

```
OG_Gene <- read.delim("/home/guillem/WorkingDir/CpGoe/CpGoe_outputs/Orthogroups_list_OG_gene.txt",
    header = FALSE, stringsAsFactors = FALSE)
colnames(OG_Gene) <- c("OG", "Transc")
length(unique(OG_Gene$V1))  # should be 59516 (numbe rof OGs)

# rename GBI transcripts
OG_Gene$Transcript <- OG_Gene$Transc
OG_Gene[OG_Gene$Transcript %like% "Gbi", ]$Transcript <- sapply(strsplit(as.character(OG_Gene[OG_Gene$T
    "Gbi", "Transcript"]), "Gbi_"), `[`, 2)
head(OG_Gene)
OG_Gene[OG_Gene$Transc %like% "GBI", ]
# rename Lko transcripts
OG_Gene[OG_Gene$Transcript %like% "Lko", ]$Transcript <- paste0("Lko_",
    sapply(strsplit(as.character(OG_Gene[OG_Gene$Transc %like%
        "Lko", "Transcript"]), "Lko_Lko_"), `[`, 2))
head(OG_Gene)
OG_Gene[OG_Gene$Transc %like% "Lko", ]

# rename Ame transcipts
Ame_cds_gene_dic <- read.delim("/home/guillem/data_disk/Genomes/Apis_mellifera/Ame_Gene_CDS_dict.txt",
    header = FALSE, sep = " ")
head(Ame_cds_gene_dic)
Ame_cds_gene_dic$LOC <- paste0("Ame_", Ame_cds_gene_dic$V1)

AmeGbiLko_insects_CpGeo_amerenames <- merge(AmeGbiLko_insects_CpGeo,
    Ame_cds_gene_dic, by.x = "Gene", by.y = "V2")
head(AmeGbiLko_insects_CpGeo_amerenames)

OG_Gene[OG_Gene$Transc %like% "Ame", ]

Ame_DF <- AmeGbiLko_insects_CpGeo_amerenames[, c("Spp", "Abb",
    "LOC", "Len", "Gfreq", "Cfreq", "CGfreq", "CpGoe", "CpGoe_level",
    "ID")]
colnames(Ame_DF) <- colnames(AmeGbiLko_insects_CpGeo)
# remove ame
dim(AmeGbiLko_insects_CpGeo)
AmeGbiLko_insects_CpGeo_v2 <- AmeGbiLko_insects_CpGeo[!AmeGbiLko_insects_CpGeo$Abb ==
    "Ame", ]
dim(AmeGbiLko_insects_CpGeo_v2)
# Add Ame with new names
AmeGbiLko_insects_CpGeo_v3 <- rbind(AmeGbiLko_insects_CpGeo_v2,
    Ame_DF)
dim(AmeGbiLko_insects_CpGeo_v3)


AmeGbiLko_CpGeo_OG <- merge(AmeGbiLko_insects_CpGeo_v3, OG_Gene,
    by.x = "ID", by.y = "Transcript")

table(AmeGbiLko_CpGeo_OG$Spp)
dim(Gbi_CpG)  #all matched woth gene and OG
dim(Lko_CpG)  #all matched woth gene and OG
dim(Ame_CpG)  # I lost just 1 gene
```

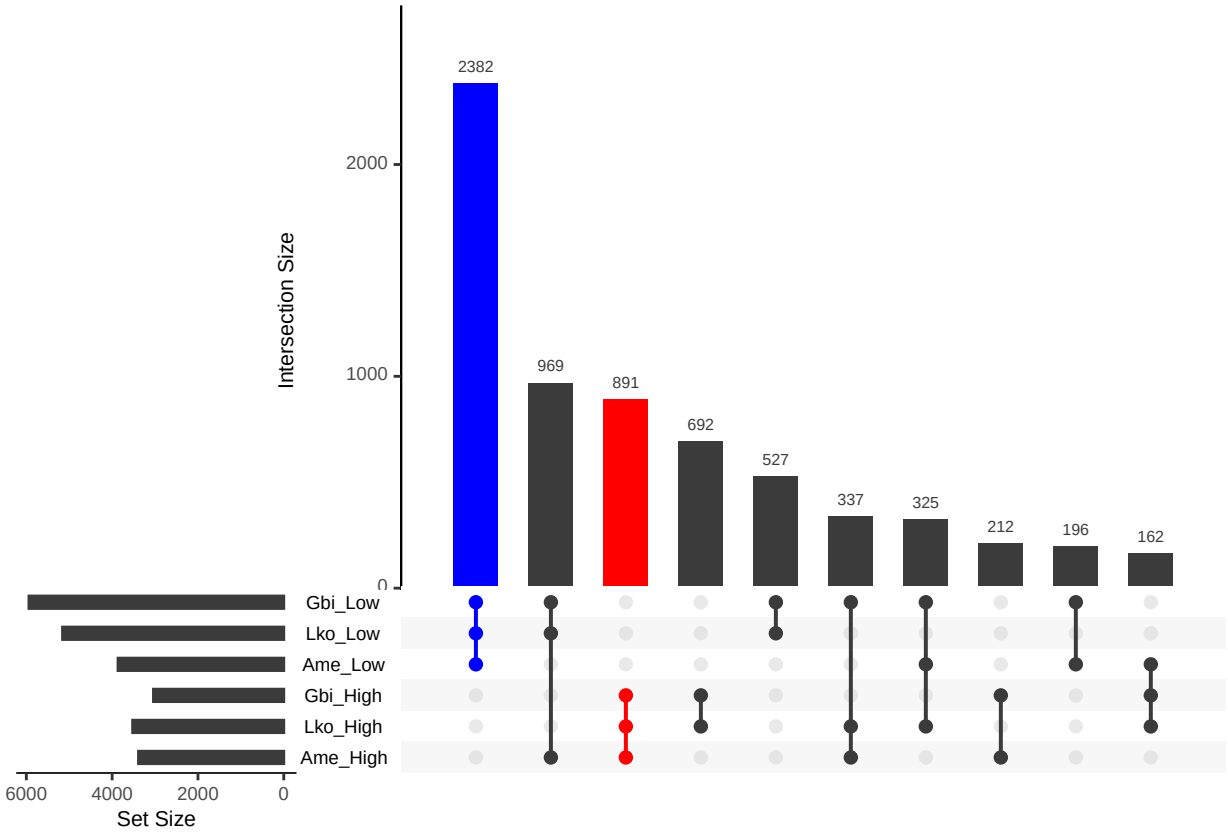
```
library(UpSetR)
Data_for_Upset<-list(Gbi_High=AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level=="High" & AmeGbiLko_CpG
                     Lko_High=AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level=="High" & AmeGbiLko_CpG
                     Ame_High=AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level=="High" & AmeGbiLko_CpG
                     Gbi_Low=AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level=="Low" & AmeGbiLko_CpGeo
                     Lko_Low=AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level=="Low" & AmeGbiLko_CpGeo
                     Ame_Low=AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level=="Low" & AmeGbiLko_CpGeo

Data_for_Upset_list<-fromList(Data_for_Upset)

Upset_raw<-upset(Data_for_Upset_list,  group.by="degree", nintersects = 80,cutoff=100,
      sets = c("Ame_High","Lko_High", "Gbi_High","Ame_Low", "Lko_Low","Gbi_Low"  ),keep.order = TRUE,
       queries = list(list(query = intersects, params = list("Gbi_Low"),  color = "purple", active = T
                  list(query = intersects, params = list("Lko_Low"),  color = "purple", active = T, que
                  list(query = intersects, params = list("Ame_Low"),  color = "purple", active = T, que
                  list(query = intersects, params = list("Gbi_High"),  color = "orange", active = T, qu
                  list(query = intersects, params = list("Lko_High"),  color = "orange", active = T, qu
                  list(query = intersects, params = list("Ame_High"),  color = "orange", active = T, qu
                  list(query = intersects, params = list("Gbi_Low",  "Lko_Low",  "Ame_Low"),  color = "
                  list(query = intersects, params = list("Gbi_High", "Lko_High", "Ame_High"), color = "
      query.legend = "bottom")
Upset_raw
```



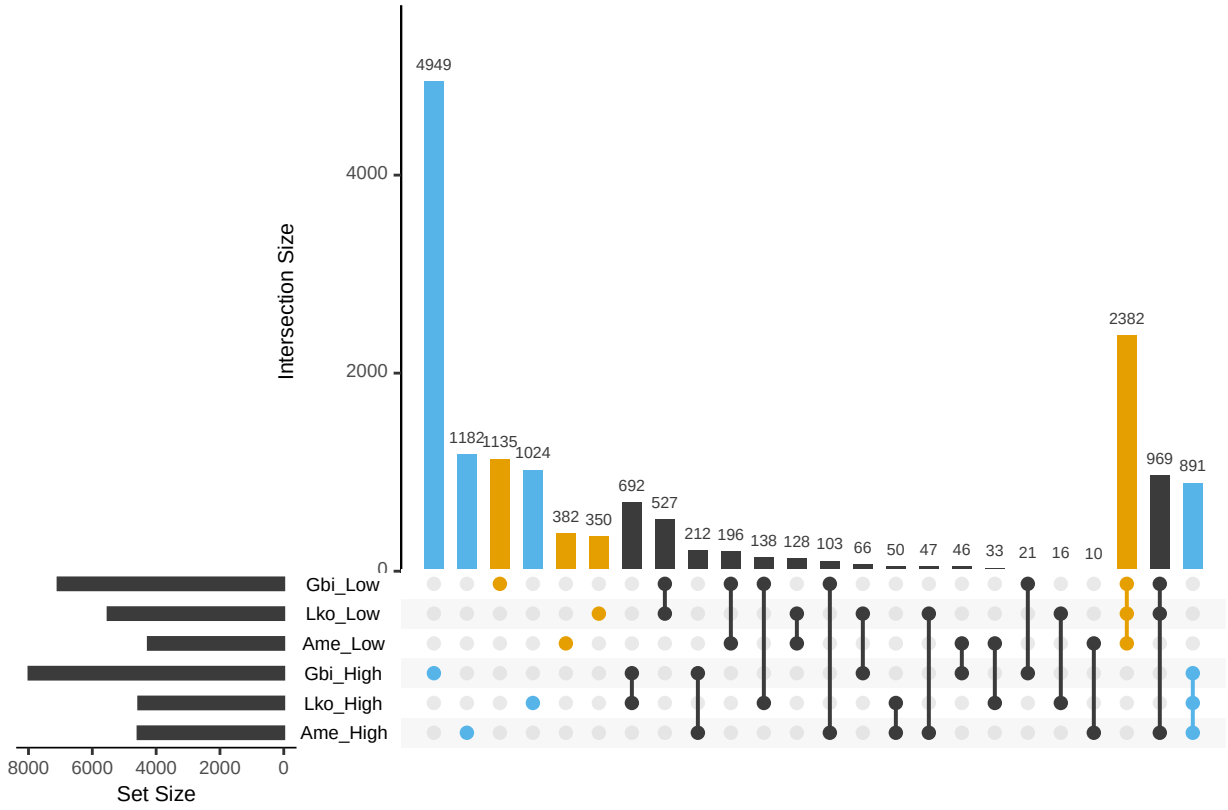
thology ■ Low − No horthology ■ Low − No horthology ■ High − No horthology ■ High − No horthology ■ High − No horthology

```
Data_for_Upset_list_filterzeros<-Data_for_Upset_list[!rowSums(Data_for_Upset_list)==1,]

Upset_filrtered<-upset(Data_for_Upset_list_filterzeros,  order.by = "freq", nintersects = 10,
```

```
      sets = c("Ame_High","Lko_High", "Gbi_High","Ame_Low", "Lko_Low","Gbi_Low"  ), keep.order = TRUE,
      queries = list(list(query = intersects, params = list("Gbi_Low",  "Lko_Low",  "Ame_Low"),  color
                     list(query = intersects, params = list("Gbi_High", "Lko_High", "Ame_High"), color =
Upset_filrtered
```




```
Upset_Publication<-upset(Data_for_Upset_list,  group.by="degree", nintersects = 24,cutoff=100,
      sets = c("Ame_High","Lko_High", "Gbi_High","Ame_Low", "Lko_Low","Gbi_Low"  ),  keep.order = TRUE
      #sets.bar.color = c("#56B4E9","#56B4E9","#56B4E9", "#E69F00", "#E69F00", "#E69F00"),
       queries = list(list(query = intersects, params = list("Gbi_Low"),  color = "#E69F00", active = 
                 list(query = intersects, params = list("Lko_Low"),  color = "#E69F00", active = T, que
                 list(query = intersects, params = list("Ame_Low"),  color = "#E69F00", active = T, que
                 list(query = intersects, params = list("Gbi_High"),  color = "#56B4E9", active = T, qu
                 list(query = intersects, params = list("Lko_High"),  color = "#56B4E9", active = T, qu
                 list(query = intersects, params = list("Ame_High"),  color = "#56B4E9", active = T, qu
                 list(query = intersects, params = list("Gbi_Low",  "Lko_Low",  "Ame_Low"),  color = "
                 list(query = intersects, params = list("Gbi_High", "Lko_High", "Ame_High"), color = "
      query.legend = "bottom")
Upset_Publication
```

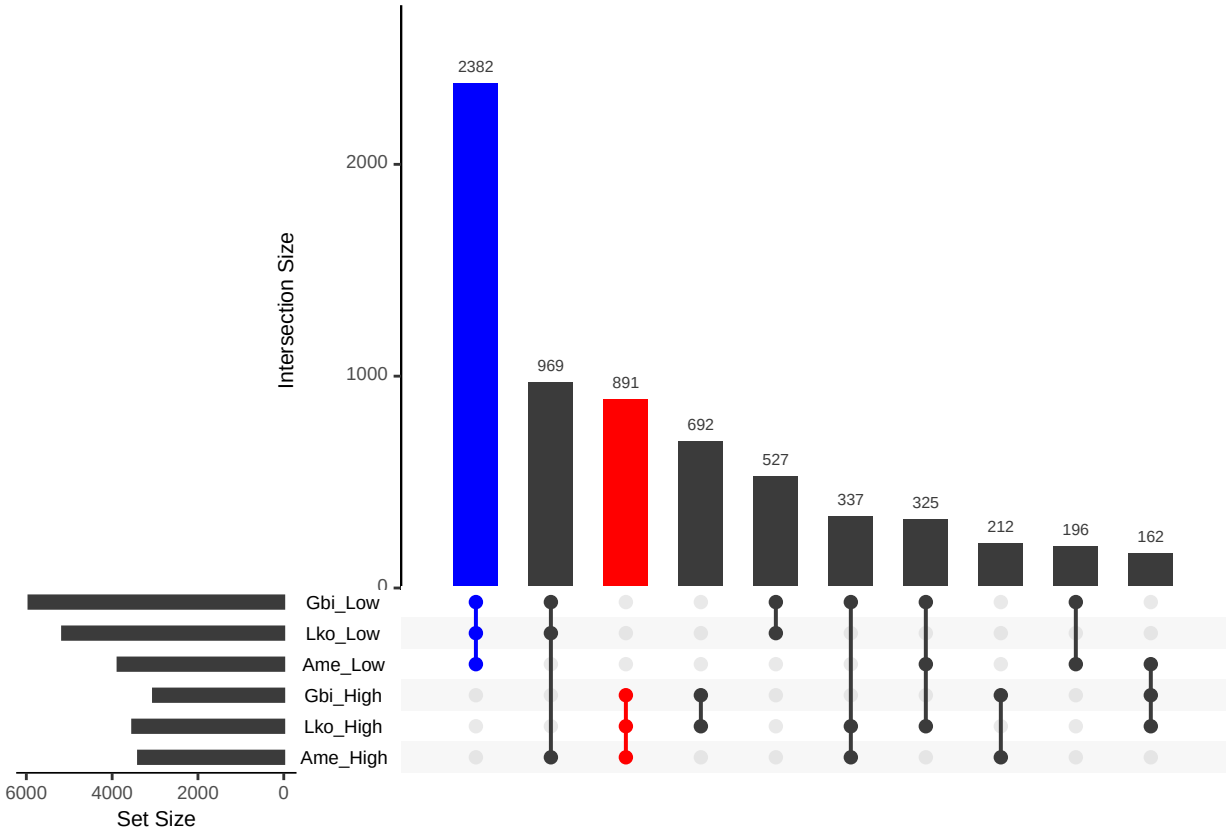


thology ■ Low − No horthology ■ Low − No horthology ■ High − No horthology ■ High − No horthology ■ High − No horthology

```
png( "plots/Upset_Publication.png", width = 8, height = 5, units = "in", res=300)
  Upset_Publication
dev.off()
```

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

```
#E69F00 orange,#56B4E9 blue
Data_for_Upset_list_filterzeros<-Data_for_Upset_list[!rowSums(Data_for_Upset_list)==1,]

Upset_filrtered<-upset(Data_for_Upset_list_filterzeros,  order.by = "freq", nintersects = 10,
       sets = c("Ame_High","Lko_High", "Gbi_High","Ame_Low", "Lko_Low","Gbi_Low"  ), keep.order = TRUE,
       queries = list(list(query = intersects, params = list("Gbi_Low",  "Lko_Low",  "Ame_Low"),  color
                      list(query = intersects, params = list("Gbi_High", "Lko_High", "Ame_High"), color =
Upset_filrtered
```



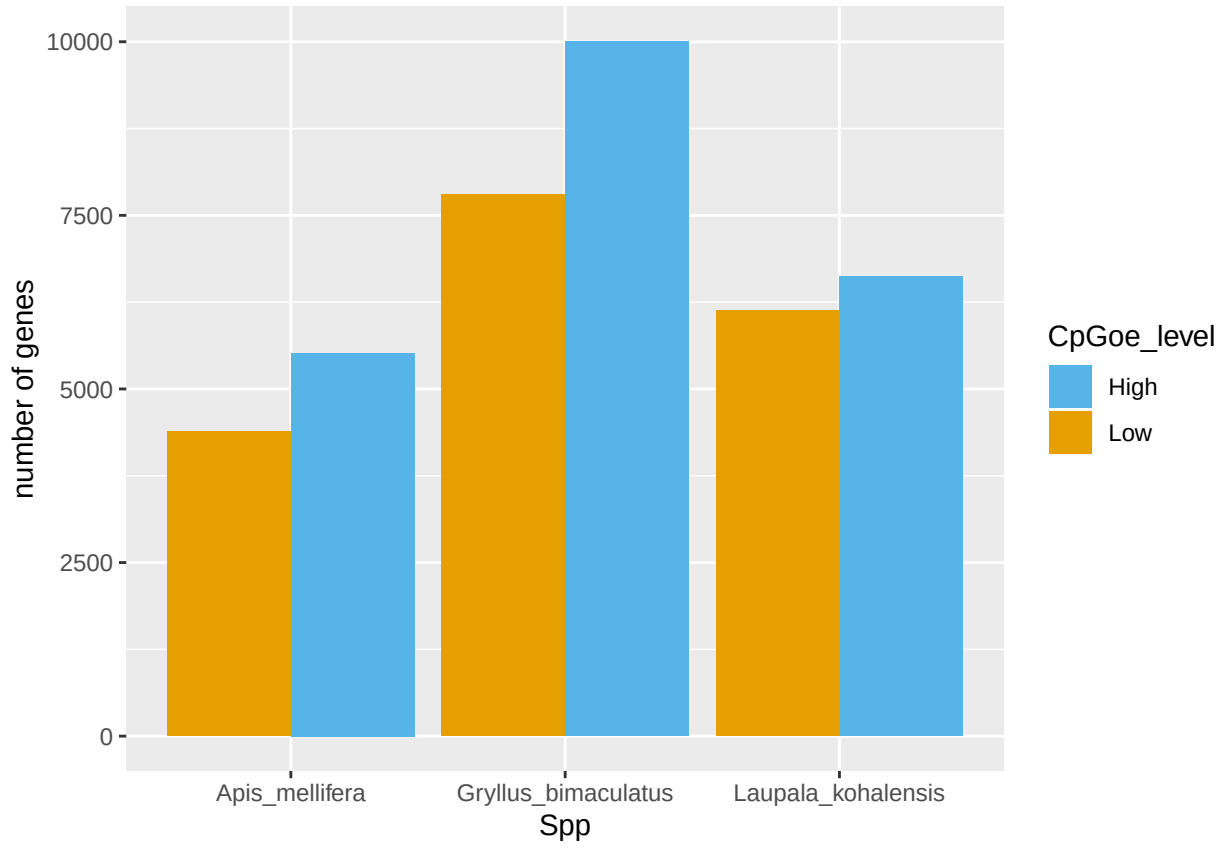

```
AmeGbiLko_CpGeo_OG_Boxplot <- reshape2::melt(AmeGbiLko_CpGeo_OG[,
    c("Spp", "CpGoe_level", "OG")])
```

```
## Using Spp, CpGoe_level, OG as id variables
```

```
head(AmeGbiLko_CpGeo_OG_Boxplot)
```

```
##              Spp CpGoe_level        OG
## 1 Apis_mellifera        High OG0002469
## 2 Apis_mellifera        High OG0000454
## 3 Apis_mellifera        High OG0007134
## 4 Apis_mellifera        High OG0004452
## 5 Apis_mellifera        High OG0003345
## 6 Apis_mellifera         Low OG0017247
```

```
## sort: 1st Low, then High
AmeGbiLko_CpGeo_OG_Boxplot$CpGoe_level <- factor(AmeGbiLko_CpGeo_OG_Boxplot$CpGoe_level,
    levels = c("Low", "High"))



p10 <- ggplot(AmeGbiLko_CpGeo_OG_Boxplot, aes(x = Spp, y = ..count..,
    fill = CpGoe_level)) + geom_bar(position = "dodge") + ylab("number of genes") +
    scale_fill_manual(breaks = c("High", "Low"), values = c("#E69F00",
        "#56B4E9"))
p10
```




```
head(AmeGbiLko_CpGeo_OG)
```

```
##               ID            Spp Abb  Len     Gfreq     Cfreq    CGfreq
## 1       Ame_18-w Apis_mellifera Ame 4113 0.2803307 0.2939460 0.1128405
## 2      Ame_5-HT1 Apis_mellifera Ame 1209 0.2762614 0.2762614 0.0927152
## 3 Ame_5-HT2alpha Apis_mellifera Ame 1962 0.2874618 0.2818552 0.0928098
## 4  Ame_5-HT2beta Apis_mellifera Ame 2202 0.3051771 0.2847411 0.0913221
## 5      Ame_5-ht7 Apis_mellifera Ame 1551 0.2295293 0.2366215 0.0574194
## 6         Ame_A4 Apis_mellifera Ame  582 0.1615120 0.2044674 0.0240964
##       CpGoe CpGoe_level               Gene        OG         Transc
## 1 1.3693885        High rna-NM_001013361.1 OG0002469       Ame_18-w
## 2 1.2148167        High rna-XM_006560049.3 OG0000454      Ame_5-HT1
## 3 1.1454801        High rna-XM_006561560.3 OG0007134 Ame_5-HT2alpha
## 4 1.0509301        High rna-NM_001204249.1 OG0004452  Ame_5-HT2beta
## 5 1.0572211        High rna-XM_026441117.1 OG0003345      Ame_5-ht7
## 6 0.7296642         Low rna-NM_001114198.1 OG0017247         Ame_A4
```

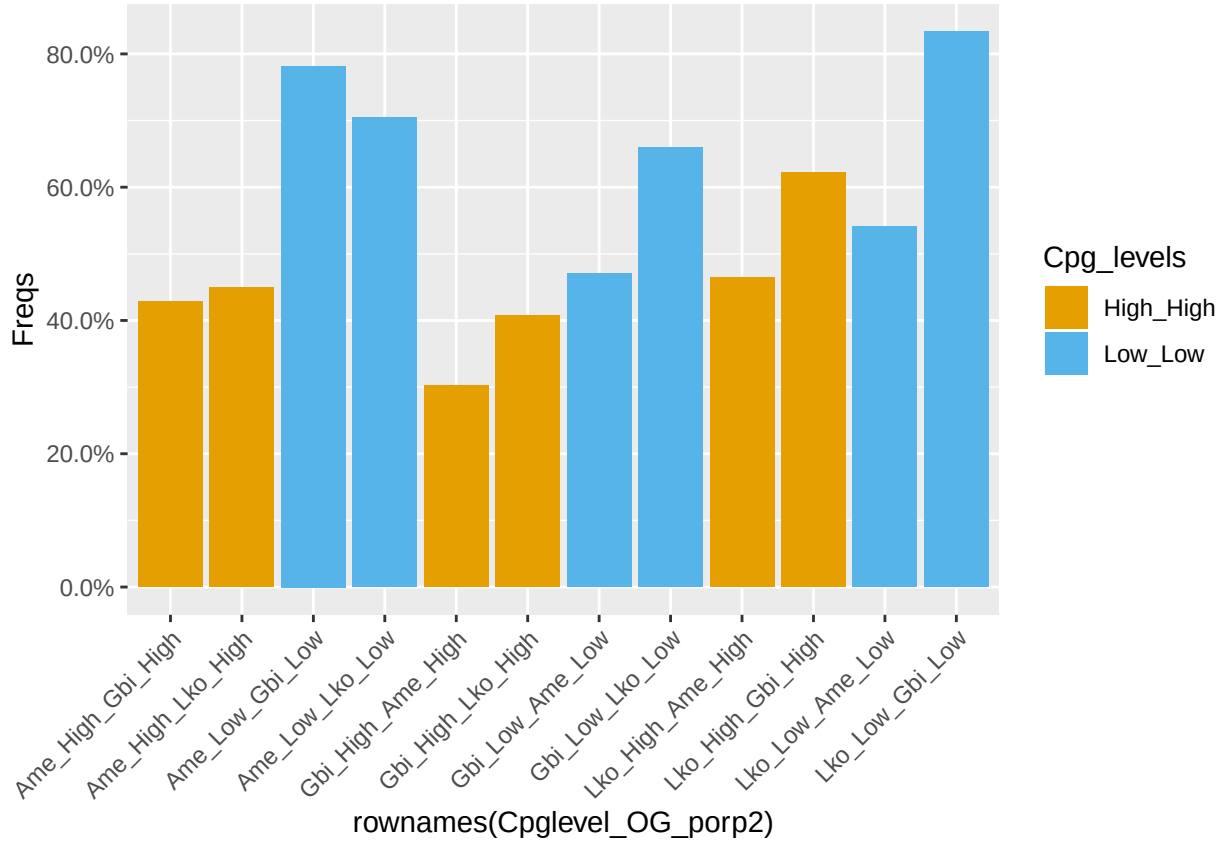
```
Cpglevel_OG_porp <- t(data.frame(Gbi_High_Ame_High = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGe
    "Gbi" & AmeGbiLko_CpGeo_OG$CpGoe_level == "High", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Ame" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "High", "OG"]))[2], Gbi_Low_Ame_Low = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Ab
    "Gbi" & AmeGbiLko_CpGeo_OG$CpGoe_level == "Low", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Ame" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "Low", "OG"]))[2], Gbi_High_Lko_High = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$A
    "Gbi" & AmeGbiLko_CpGeo_OG$CpGoe_level == "High", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Lko" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "High", "OG"]))[2], Gbi_Low_Lko_Low = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Ab
```

```
    "Gbi" & AmeGbiLko_CpGeo_OG$CpGoe_level == "Low", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Lko" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "Low", "OG"]))[2], Lko_High_Ame_High = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Al
    "Lko" & AmeGbiLko_CpGeo_OG$CpGoe_level == "High", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Ame" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "High", "OG"]))[2], Lko_Low_Ame_Low = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abl
    "Lko" & AmeGbiLko_CpGeo_OG$CpGoe_level == "Low", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Ame" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "Low", "OG"]))[2], Lko_High_Gbi_High = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Al
    "Lko" & AmeGbiLko_CpGeo_OG$CpGoe_level == "High", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Gbi" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "High", "OG"]))[2], Lko_Low_Gbi_Low = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abl
    "Lko" & AmeGbiLko_CpGeo_OG$CpGoe_level == "Low", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Gbi" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "Low", "OG"]))[2], Ame_High_Gbi_High = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Al
    "Ame" & AmeGbiLko_CpGeo_OG$CpGoe_level == "High", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Gbi" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "High", "OG"]))[2], Ame_Low_Gbi_Low = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abl
    "Ame" & AmeGbiLko_CpGeo_OG$CpGoe_level == "Low", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Gbi" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "Low", "OG"]))[2], Ame_High_Lko_High = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Al
    "Ame" & AmeGbiLko_CpGeo_OG$CpGoe_level == "High", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Lko" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "High", "OG"]))[2], Ame_Low_Lko_Low = prop.table(table(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abl
    "Ame" & AmeGbiLko_CpGeo_OG$CpGoe_level == "Low", "OG"] %in%
    AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Lko" & AmeGbiLko_CpGeo_OG$CpGoe_level ==
        "Low", "OG"]))[2]))

Cpglevel_OG_porp2 <- data.frame(row.names = rownames(Cpglevel_OG_porp),
    Freqs = as.numeric(Cpglevel_OG_porp), Species1 = sapply(strsplit(rownames(Cpglevel_OG_porp),
        "_"), `[`, 1), Cpg_levels = paste(sapply(strsplit(rownames(Cpglevel_OG_porp),
        "_"), `[`, 2), sapply(strsplit(rownames(Cpglevel_OG_porp),
        "_"), `[`, 4), sep = "_"))

Cpglevel_OG_porp2
```

```
##                       Freqs Species1 Cpg_levels
## Gbi_High_Ame_High 0.3029182      Gbi  High_High
## Gbi_Low_Ame_Low   0.4710786      Gbi    Low_Low
## Gbi_High_Lko_High 0.4072556      Gbi  High_High
## Gbi_Low_Lko_Low   0.6597409      Gbi    Low_Low
## Lko_High_Ame_High 0.4649441      Lko  High_High
## Lko_Low_Ame_Low   0.5413742      Lko    Low_Low
## Lko_High_Gbi_High 0.6219402      Lko  High_High
## Lko_Low_Gbi_Low   0.8333605      Lko    Low_Low
## Ame_High_Gbi_High 0.4278463      Ame  High_High
## Ame_Low_Gbi_Low   0.7818555      Ame    Low_Low
## Ame_High_Lko_High 0.4499637      Ame  High_High
## Ame_Low_Lko_Low   0.7045817      Ame    Low_Low
```

```
ggplot(data = Cpglevel_OG_porp2, aes(x = rownames(Cpglevel_OG_porp2),
    group = Species1, y = Freqs, fill = Cpg_levels)) + geom_bar(stat = "identity") +
    scale_y_continuous(labels = scales::percent) + theme(axis.text.x = element_text(angle = 45,
    hjust = 1)) + scale_fill_manual(breaks = c("High_High", "Low_Low"),
```

```
    values = c("#E69F00", "#56B4E9"))
```



CpGoe Functional Analysis

Gene orthology Enrichment Analysis.

```
library("topGO")
```

```
## Loading required package: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
## Loading required package: GO.db
```

```
## Loading required package: AnnotationDbi
```

```
##
## Attaching package: 'AnnotationDbi'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
##
```

```
## Loading required package: SparseM
##
## Attaching package: 'SparseM'
## The following object is masked from 'package:base':
##
##     backsolve
##
## groupGOTerms:    GOBPTerm, GOMFTerm, GOCCTerm environments built.
##
## Attaching package: 'topGO'
## The following object is masked from 'package:mixtools':
##
##     depth
## The following object is masked from 'package:grid':
##
##     depth
## The following object is masked from 'package:IRanges':
##
##     members
```

```r
## Gbi
conGbi <- dbConnect(RSQLite::SQLite(), "/home/guillem/data_disk/Cricket_genome_annotation/GBI_Genome_v3/
Gbi_Gene_GO <- dbFetch(dbSendQuery(conGbi, "SELECT GeneID,GOterms FROM genes"))
dbDisconnect(conGbi)

dim(Gbi_Gene_GO)
write.table(Gbi_Gene_GO, "/home/guillem/WorkingDir/CpGoe/Gbi_Gene_GO.csv",
    row.names = FALSE, sep = "\t", quote = FALSE)
geneID2GO_gbi <- readMappings("/home/guillem/WorkingDir/CpGoe/Gbi_Gene_GO.csv")
head(geneID2GO_gbi)

## Lko
conLko <- dbConnect(RSQLite::SQLite(), "/home/guillem/data_disk/Cricket_genome_annotation/Laupala_kohal
Lko_Gene_GO <- dbFetch(dbSendQuery(conLko, "SELECT GeneID,GOterms FROM genes"))
dbDisconnect(conLko)

dim(Lko_Gene_GO)
write.table(Lko_Gene_GO, "/home/guillem/WorkingDir/CpGoe/Lko_Gene_GO.csv",
    row.names = FALSE, sep = "\t", quote = FALSE)
geneID2GO_lko <- readMappings("/home/guillem/WorkingDir/CpGoe/Lko_Gene_GO.csv")
head(geneID2GO_lko)

## Ame For Ame, I ran Iprscan to longest prot per geneql
head(Ame_CpG)
geneID2GO_Ame <- readMappings("/home/guillem/data_disk/Genomes/Apis_mellifera/Ame_Gene_GO_dict.txt")
head(geneID2GO_Ame)
```

Gryllus bimaculatus CpGoe GO-ernichment

I select genes belonging to the low CpGoe peak (methylated genes). And eprform a GOI-enrichment vs all genes.

Gbi Low CpGoe

```
## Low CpG
dim(Gbi_CpG)  #68 less than the 17871, because NA & 0>0CpGoe<0 removed
```

```
## [1] 17803    10
```

```
GBi_LowCPG <- Gbi_CpG[Gbi_CpG$CpGoe_level == "Low", "Gene"]
# length(GBi_LowCPG) table(Gbi_CpG$CpGoe_level)
# prop.table(table(Gbi_CpG$CpGoe_level))

## selected genes/universe
geneListGbiLow = factor(as.integer(Gbi_CpG$Gene %in% GBi_LowCPG))  ## being or not in the selected ones
names(geneListGbiLow) = Gbi_CpG$Gene
# head(geneListGbiLow) table(geneListGbiLow)

Gbi_Low_GOdataBP <- new("topGOdata", description = "Gbi Low CpGoe",
    ontology = "BP", allGenes = geneListGbiLow, geneSel = Gbi_CpG$Gene,
    nodeSize = 1, annot = annFUN.gene2GO, gene2GO = geneID2GO_gbi)

Gbi_Low_GOdataBP
```

```
##
## ------------------------ topGOdata object -------------------------
##
##  Description:
##    -  Gbi Low CpGoe
##
##  Ontology:
##    -  BP
##
##  17803 available genes (all genes from the array):
##    - symbol:  GBI_21039 GBI_21037 GBI_21038 GBI_05592 GBI_05594  ...
##    - 7797  significant genes.
##
##  3552 feasible genes (genes that can be used in the analysis):
##    - symbol:  GBI_05594 GBI_05590 GBI_05597 GBI_05595 GBI_05599  ...
##    - 1987  significant genes.
##
##  GO graph (nodes with at least  1  genes):
##    - a graph with directed edges
##    - number of nodes = 1502
##    - number of edges = 3152
##
## ------------------------ topGOdata object -------------------------
```

```
Gbi_Low_weight01_fisher_BP = runTest(Gbi_Low_GOdataBP, algorithm = "weight01",
    statistic = "fisher")
# Gbi_Low_weight01_fisher_BP
```

```
allGO = usedGO(Gbi_Low_GOdataBP)

res_Gbi_Low_weight0_fisher_BP = GenTable(Gbi_Low_GOdataBP, weightFisher = Gbi_Low_weight01_fisher_BP,
    orderBy = "weightFisher", topNodes = length(allGO))
# head(res_Gbi_Low_weight0_fisher_BP)

# write.csv(res_Gbi_Low_weight0_fisher_BP,
# 'res_Gbi_Low_weight0_fisher_BP')
```

Gbi High CpGoe

```
# High CpG
GBi_HighCPG <- Gbi_CpG[Gbi_CpG$CpGoe_level == "High", "Gene"]
# length(GBi_HighCPG)


## selected genes/universe
geneListGbiHigh = factor(as.integer(Gbi_CpG$Gene %in% GBi_HighCPG))  ## being or not in the s;leected o
names(geneListGbiHigh) = Gbi_CpG$Gene
# head(geneListGbiHigh) table(geneListGbiHigh)



Gbi_High_GOdataBP <- new("topGOdata", description = "Gbi High CpGoe",
    ontology = "BP", allGenes = geneListGbiHigh, geneSel = Gbi_CpG$Gene,
    nodeSize = 1, annot = annFUN.gene2GO, gene2GO = geneID2GO_gbi)

Gbi_High_GOdataBP
```

```
##
## ------------------------ topGOdata object -------------------------
##
##  Description:
##    -  Gbi High CpGoe
##
##  Ontology:
##    -  BP
##
##  17803 available genes (all genes from the array):
##    - symbol:  GBI_21039 GBI_21037 GBI_21038 GBI_05592 GBI_05594  ...
##    - 10006  significant genes.
##
##  3552 feasible genes (genes that can be used in the analysis):
##    - symbol:  GBI_05594 GBI_05590 GBI_05597 GBI_05595 GBI_05599  ...
##    - 1565  significant genes.
##
##  GO graph (nodes with at least  1  genes):
##    - a graph with directed edges
##    - number of nodes = 1502
##    - number of edges = 3152
##
## ------------------------ topGOdata object -------------------------
```

```
Gbi_High_weight01_fisher_BP = runTest(Gbi_High_GOdataBP, algorithm = "weight01",
    statistic = "fisher")
Gbi_High_weight01_fisher_BP
```

```
##
## Description: Gbi High CpGoe
## Ontology: BP
## 'weight01' algorithm with the 'fisher' test
## 1502 GO terms scored: 14 terms with p < 0.01
## Annotation data:
##     Annotated genes: 3552
##     Significant genes: 1565
##     Min. no. of genes annotated to a GO: 1
##     Nontrivial nodes: 863
```

```
allGO = usedGO(Gbi_High_GOdataBP)


res_Gbi_High_weight0_fisher_BP = GenTable(Gbi_High_GOdataBP,
    weightFisher = Gbi_High_weight01_fisher_BP, orderBy = "weightFisher",
    topNodes = length(allGO))
# head(res_Gbi_High_weight0_fisher_BP)

# write.csv(res_Gbi_High_weight0_fisher_BP,
# 'res_Gbi_High_weight0_fisher_BP')
```

Laupala kohalensis CpGoe GO-ernichment

Lko Low CpGoe

```
## Lko Low CpG
Lko_LowCPG <- Lko_CpG[Lko_CpG$CpGoe_level == "Low", "Gene"]
length(Lko_LowCPG)
```

```
## [1] 6127
```

```
table(Lko_CpG$CpGoe_level)
```

```
##
##  Low High
## 6127 6618
```

```
## selected genes/universe
geneListLkoLow = factor(as.integer(Lko_CpG$Gene %in% Lko_LowCPG))  ## being or not in the s;leected one
names(geneListLkoLow) = Lko_CpG$Gene
# head(geneListLkoLow) table(geneListLkoLow)

Lko_Low_GOdataBP <- new("topGOdata", description = "Lko Low CpGoe",
    ontology = "BP", allGenes = geneListLkoLow, geneSel = Lko_CpG$Gene,
    nodeSize = 1, annot = annFUN.gene2GO, gene2GO = geneID2GO_lko)

Lko_Low_GOdataBP
```

```
##
## ------------------------ topGOdata object -------------------------
##
```

```
##  Description:
##    -  Lko Low CpGoe
##
##  Ontology:
##    -  BP
##
##  12745 available genes (all genes from the array):
##    - symbol:  Lko_35551 Lko_35552 Lko_31320 Lko_31321 Lko_31325  ...
##    - 6127  significant genes.
##
##  3470 feasible genes (genes that can be used in the analysis):
##    - symbol:  Lko_07571 Lko_07572 Lko_33545 Lko_28272 Lko_23652  ...
##    - 1708  significant genes.
##
##  GO graph (nodes with at least  1  genes):
##    - a graph with directed edges
##    - number of nodes = 1376
##    - number of edges = 2878
##
## ------------------------ topGOdata object -------------------------
```

```
Lko_Low_weight01_fisher_BP = runTest(Lko_Low_GOdataBP, algorithm = "weight01",
    statistic = "fisher")
# Lko_Low_weight01_fisher_BP

allGO = usedGO(Lko_Low_GOdataBP)

res_Lko_Low_weight0_fisher_BP = GenTable(Lko_Low_GOdataBP, weightFisher = Lko_Low_weight01_fisher_BP,
    orderBy = "weightFisher", topNodes = length(allGO))
# head(res_Lko_Low_weight0_fisher_BP)

# write.csv(res_Lko_Low_weight0_fisher_BP,
# 'res_Lko_Low_weight0_fisher_BP.csv')
```

Lko High CpGoe

```
# High CpG
Lko_HighCPG <- Lko_CpG[Lko_CpG$CpGoe_level == "High", "Gene"]
# length(Lko_HighCPG)


## selected genes/universe
geneListLkoHigh = factor(as.integer(Lko_CpG$Gene %in% Lko_HighCPG))  ## being or not in the s;leected o
names(geneListLkoHigh) = Lko_CpG$Gene
# head(geneListLkoHigh) table(geneListLkoHigh)



Lko_High_GOdataBP <- new("topGOdata", description = "Lko High CpGoe",
    ontology = "BP", allGenes = geneListLkoHigh, geneSel = Lko_CpG$Gene,
    nodeSize = 1, annot = annFUN.gene2GO, gene2GO = geneID2GO_lko)

Lko_High_GOdataBP
```

```
##
## ------------------------ topGOdata object -------------------------
##
##  Description:
##    -  Lko High CpGoe
##
##  Ontology:
##    -  BP
##
##  12745 available genes (all genes from the array):
##    - symbol:  Lko_35551 Lko_35552 Lko_31320 Lko_31321 Lko_31325  ...
##    - 6618  significant genes.
##
##  3470 feasible genes (genes that can be used in the analysis):
##    - symbol:  Lko_07571 Lko_07572 Lko_33545 Lko_28272 Lko_23652  ...
##    - 1762  significant genes.
##
##  GO graph (nodes with at least  1  genes):
##    - a graph with directed edges
##    - number of nodes = 1376
##    - number of edges = 2878
##
## ------------------------ topGOdata object -------------------------
```

```
Lko_High_weight01_fisher_BP = runTest(Lko_High_GOdataBP, algorithm = "weight01",
    statistic = "fisher")
Lko_High_weight01_fisher_BP
```

```
##
## Description: Lko High CpGoe
## Ontology: BP
## 'weight01' algorithm with the 'fisher' test
## 1376 GO terms scored: 11 terms with p < 0.01
## Annotation data:
##     Annotated genes: 3470
##     Significant genes: 1762
##     Min. no. of genes annotated to a GO: 1
##     Nontrivial nodes: 886
```

```
allGO = usedGO(Lko_High_GOdataBP)


res_Lko_High_weight0_fisher_BP = GenTable(Lko_High_GOdataBP,
    weightFisher = Lko_High_weight01_fisher_BP, orderBy = "weightFisher",
    topNodes = length(allGO))
# head(res_Lko_High_weight0_fisher_BP)


# write.csv(res_Lko_High_weight0_fisher_BP,
# 'res_Lko_High_weight0_fisher_BP')
```

Apis mellifera CpGoe GO-ernichment

Ame Low CpGoe

```r
## I need to add gene name LOCXXX to Ame_CpG, bc LOC isthe one
## in GO table head(Ame_CpG)

Ame_cds_gene_dic <- read.delim("/home/guillem/data_disk/Genomes/Apis_mellifera/Ame_Gene_CDS_dict.txt",
    header = FALSE, sep = " ")
# head(Ame_cds_gene_dic)
colnames(Ame_cds_gene_dic) <- c("LOC", "XM")
Ame_CpG_LOC <- merge(Ame_CpG, Ame_cds_gene_dic, by.x = "Gene",
    by.y = "XM")

# 5274 genes with Goterms
# table(Ame_CpG_LOC$LOC%in%names(geneID2GO_Ame))

## Ame Low CpG
Ame_LowCPG <- Ame_CpG_LOC[Ame_CpG_LOC$CpGoe_level == "Low", "LOC"]
# length(Ame_LowCPG) table(Ame_CpG_LOC$CpGoe_level)


## selected genes/universe
geneListAmeLow = factor(as.integer(Ame_CpG_LOC$LOC %in% Ame_LowCPG))  ## being or not in the s;leected
names(geneListAmeLow) = Ame_CpG_LOC$LOC
# head(geneListAmeLow) table(geneListAmeLow)


Ame_Low_GOdataBP <- new("topGOdata", description = "Ame Low CpGoe",
    ontology = "BP", allGenes = geneListAmeLow, geneSel = Ame_CpG$Gene,
    nodeSize = 1, annot = annFUN.gene2GO, gene2GO = geneID2GO_Ame)

Ame_Low_GOdataBP
```

```
##
## ------------------------ topGOdata object -------------------------
##
##  Description:
##    -  Ame Low CpGoe
##
##  Ontology:
##    -  BP
##
##  9904 available genes (all genes from the array):
##    - symbol:  ATP6 COX1 COX2 COX3 CYTB  ...
##    - 4388  significant genes.
##
##  3016 feasible genes (genes that can be used in the analysis):
##    - symbol:  ATP6 COX1 COX2 ND1 ND3  ...
##    - 1316  significant genes.
##
##  GO graph (nodes with at least  1  genes):
##    - a graph with directed edges
##    - number of nodes = 1548
##    - number of edges = 3241
##
## ------------------------ topGOdata object -------------------------
```

```
Ame_Low_weight01_fisher_BP = runTest(Ame_Low_GOdataBP, algorithm = "weight01",
    statistic = "fisher")
# Ame_Low_weight01_fisher_BP

allGO = usedGO(Ame_Low_GOdataBP)

res_Ame_Low_weight0_fisher_BP = GenTable(Ame_Low_GOdataBP, weightFisher = Ame_Low_weight01_fisher_BP,
    orderBy = "weightFisher", topNodes = length(allGO))
# head(res_Ame_Low_weight0_fisher_BP)

# write.csv(res_Ame_Low_weight0_fisher_BP,
# 'res_Ame_Low_weight0_fisher_BP.csv')
```

Ame High CpGoe

```
## Ame High CpG
Ame_HighCPG <- Ame_CpG_LOC[Ame_CpG_LOC$CpGoe_level == "High",
    "LOC"]
# length(Ame_HighCPG) table(Ame_CpG_LOC$CpGoe_level)


## selected genes/universe
geneListAmeHigh = factor(as.integer(Ame_CpG_LOC$LOC %in% Ame_HighCPG))  ## being or not in the s;leecte
names(geneListAmeHigh) = Ame_CpG_LOC$LOC
# head(geneListAmeHigh) table(geneListAmeHigh)


Ame_High_GOdataBP <- new("topGOdata", description = "Ame High CpGoe",
    ontology = "BP", allGenes = geneListAmeHigh, geneSel = Ame_CpG$Gene,
    nodeSize = 1, annot = annFUN.gene2GO, gene2GO = geneID2GO_Ame)

Ame_High_GOdataBP
```

```
##
## ------------------------ topGOdata object -------------------------
##
##  Description:
##    -  Ame High CpGoe
##
##  Ontology:
##    -  BP
##
##  9904 available genes (all genes from the array):
##    - symbol:  ATP6 COX1 COX2 COX3 CYTB  ...
##    - 5516  significant genes.
##
##  3016 feasible genes (genes that can be used in the analysis):
##    - symbol:  ATP6 COX1 COX2 ND1 ND3  ...
##    - 1700  significant genes.
##
##  GO graph (nodes with at least  1  genes):
##    - a graph with directed edges
##    - number of nodes = 1548
```

```
##     - number of edges = 3241
##
## ------------------------ topGOdata object -------------------------
```

```
Ame_High_weight01_fisher_BP = runTest(Ame_High_GOdataBP, algorithm = "weight01",
    statistic = "fisher")
# Ame_High_weight01_fisher_BP

allGO = usedGO(Ame_High_GOdataBP)

res_Ame_High_weight0_fisher_BP = GenTable(Ame_High_GOdataBP,
    weightFisher = Ame_High_weight01_fisher_BP, orderBy = "weightFisher",
    topNodes = length(allGO))
# head(res_Ame_High_weight0_fisher_BP)

# write.csv(res_Ame_High_weight0_fisher_BP,
# 'res_Ame_High_weight0_fisher_BP.csv')
```

CpGoe Level Change: Crickets low vs Ame High

It is a bit complicated:

1- I get all the Orthorgroups (OGs) from genes that are Low in Ceach cricket and High in Ame 2- The intersection will select those OGs common in the 3 lists *4- There are some OGs that contain genes that are both High and low in same insect. I remove them and I get same number of OGs shown in the UpsetPlot* 5- To do the GO-enrichment, I need to have genes. I use the list of 969 OGs to retrieve its geens in Gbi. *6- Each OG can have multiple genes... So I end p retrieveing 1022 Gbi genes for tyhe GO-nerichment analysis.

Conclusion, tehre are 1022 genes in Gbi that belong to 969 OGs that inCrickets are only low and in Lko are only High.

```
CricketLow_AmeHigh <- Reduce(intersect, list(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "Low" & AmeGbiLko_CpGeo_OG$Abb == "Gbi", "OG"], AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "Low" & AmeGbiLko_CpGeo_OG$Abb == "Lko", "OG"], AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "High" & AmeGbiLko_CpGeo_OG$Abb == "Ame", "OG"]))
length(unique(CricketLow_AmeHigh))
```

```
## [1] 1360
```

```
# same OG can't be in any otehr convination: filter out OGs
# which in the same insect are High and low
Gbihighlow <- Reduce(intersect, list(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "Low" & AmeGbiLko_CpGeo_OG$Abb == "Gbi", "OG"], AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "High" & AmeGbiLko_CpGeo_OG$Abb == "Gbi", "OG"]))

Lkohighlow <- Reduce(intersect, list(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "Low" & AmeGbiLko_CpGeo_OG$Abb == "Lko", "OG"], AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "High" & AmeGbiLko_CpGeo_OG$Abb == "Lko", "OG"]))

Amehighlow <- Reduce(intersect, list(AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "Low" & AmeGbiLko_CpGeo_OG$Abb == "Ame", "OG"], AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$CpGoe_level ==
    "High" & AmeGbiLko_CpGeo_OG$Abb == "Ame", "OG"]))

CricketLow_AmeHigh_2 <- CricketLow_AmeHigh[!CricketLow_AmeHigh %in%
    c(Gbihighlow, Lkohighlow, Amehighlow)]
length(CricketLow_AmeHigh_2)
```

```
## [1] 969
```

```
### GO terms based on GBI genes

Gbisubtable <- AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb == "Gbi",
    ]
CricketLow_AmeHigh_Gbigenes <- Gbisubtable[Gbisubtable$OG %in%
    CricketLow_AmeHigh_2, ]
dim(CricketLow_AmeHigh_Gbigenes)  ## the 1022 OGs contain 3054 cricket genes
```

```
## [1] 1022   12
```

```
table(CricketLow_AmeHigh_Gbigenes$CpGoe_level)
```

```
##
##  Low High
## 1022    0
```

```
## selected genes/universe
geneListCrickLow_AmeHigh = factor(as.integer(Gbi_CpG$Gene %in%
    CricketLow_AmeHigh_Gbigenes$Gene))  ## being or not in the selected ones as Factor
names(geneListCrickLow_AmeHigh) = Gbi_CpG$Gene
table(geneListCrickLow_AmeHigh)
```

```
## geneListCrickLow_AmeHigh
##     0     1
## 16781  1022
```

```
Crick_Low_Ame_High_GOdataBP <- new("topGOdata", description = "Crick Low and Ame High CpGoe (based on G
    ontology = "BP", allGenes = geneListCrickLow_AmeHigh, geneSel = Gbi_CpG$Gene,
    nodeSize = 1, annot = annFUN.gene2GO, gene2GO = geneID2GO_gbi)

Crick_Low_Ame_High_GOdataBP
```

```
##
## ------------------------ topGOdata object -------------------------
##
##  Description:
##    -  Crick Low and Ame High CpGoe (based on Gbi Functons)
##
##  Ontology:
##    -  BP
##
##  17803 available genes (all genes from the array):
##    - symbol:  GBI_21039 GBI_21037 GBI_21038 GBI_05592 GBI_05594  ...
##    - 1022  significant genes.
##
##  3552 feasible genes (genes that can be used in the analysis):
##    - symbol:  GBI_05594 GBI_05590 GBI_05597 GBI_05595 GBI_05599  ...
##    - 316  significant genes.
##
##  GO graph (nodes with at least  1  genes):
##    - a graph with directed edges
##    - number of nodes = 1502
##    - number of edges = 3152
##
## ------------------------ topGOdata object -------------------------
```

```
Crick_Low_Ame_High_weight01_fisher_BP = runTest(Crick_Low_Ame_High_GOdataBP,
    algorithm = "weight01", statistic = "fisher")


allGO = usedGO(Crick_Low_Ame_High_GOdataBP)

Crick_Low_Ame_High_weight0_fisher_BP = GenTable(Crick_Low_Ame_High_GOdataBP,
    weightFisher = Crick_Low_Ame_High_weight01_fisher_BP, orderBy = "weightFisher",
    topNodes = length(allGO))

head(Crick_Low_Ame_High_weight0_fisher_BP)
```

```
##        GO.ID                                         Term Annotated
## 1 GO:0006468                    protein phosphorylation       234
## 2 GO:0006635                  fatty acid beta-oxidation         4
## 3 GO:0034227                     tRNA thio-modification         2
## 4 GO:0030036             actin cytoskeleton organization        12
## 5 GO:0007169 transmembrane receptor protein tyrosine ...         3
## 6 GO:0007264 small GTPase mediated signal transductio...        49
##   Significant Expected weightFisher
## 1          47    20.82      3.0e-08
## 2           4     0.36      6.2e-05
## 3           2     0.18       0.0079
## 4           4     1.07       0.0221
## 5           2     0.27       0.0223
## 6          10     4.36       0.0291
```

```
# write.csv(Crick_Low_Ame_High_weight0_fisher_BP,
# 'Crick_Low_Ame_High_weight0_fisher_BP.csv')
```

CpGoe level Change on Crickets vs Apis

Plot GO-enrichment

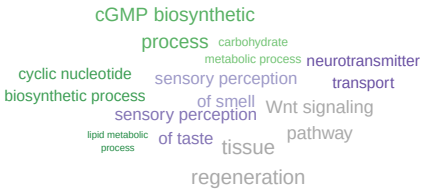
Define function:

```
library(ggwordcloud)
library("RColorBrewer")

WordCloud_GOs <- function(TopGO_result, condition, pvalue) {

    GOs_toPlot <- TopGO_result[TopGO_result$weightFisher < pvalue,
        ]
    GOs_toPlot$weightFisher <- as.numeric(GOs_toPlot$weightFisher)

    rnacolors <- colorRampPalette(brewer.pal(4, "Reds")[3:4])
    rnagenes <- rownames(GOs_toPlot[GOs_toPlot$Term %like% "RNA" |
        GOs_toPlot$Term %like% "transcription", ])
    GOs_toPlot[rnagenes, "colores"] <- rnacolors(length(rnagenes))

    dnacolors <- colorRampPalette(brewer.pal(4, "Blues")[3:4])
    dnagenes <- rownames(GOs_toPlot[GOs_toPlot$Term %like% "DNA" |
        GOs_toPlot$Term %like% "repair" | GOs_toPlot$Term %like%
```

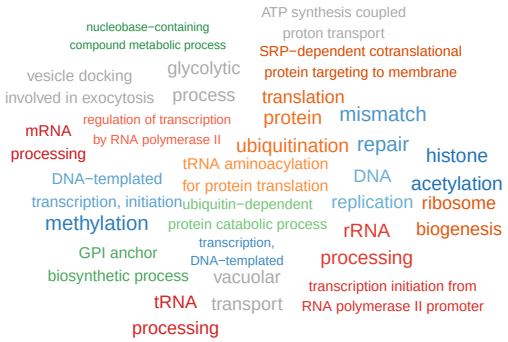
```
        "nucleotide" | GOs_toPlot$Term %like% "histone" | GOs_toPlot$Term %like%
        "methylation", ])
    GOs_toPlot[dnagenes, "colores"] <- dnacolors(length(dnagenes))

    proteincolors <- colorRampPalette(brewer.pal(4, "Oranges")[3:4])
    proteingenes <- rownames(GOs_toPlot[GOs_toPlot$Term %like%
        "protein" | GOs_toPlot$Term %like% "translation" | GOs_toPlot$Term %like%
        "ribosome", ])
    GOs_toPlot[proteingenes, "colores"] <- proteincolors(length(proteingenes))


    metacolors <- colorRampPalette(brewer.pal(4, "Greens")[3:4])
    metagenes <- rownames(GOs_toPlot[GOs_toPlot$Term %like% "catabol" |
        GOs_toPlot$Term %like% "metabol" | GOs_toPlot$Term %like%
        "biosyn", ])
    GOs_toPlot[metagenes, "colores"] <- metacolors(length(metagenes))


    sensorycolors <- colorRampPalette(brewer.pal(4, "Purples")[3:4])
    sensorygenes <- rownames(GOs_toPlot[GOs_toPlot$Term %like%
        "sensory" | GOs_toPlot$Term %like% "perception" | GOs_toPlot$Term %like%
        "neuro", ])
    GOs_toPlot[sensorygenes, "colores"] <- sensorycolors(length(sensorygenes))

    GOs_toPlot[is.na(GOs_toPlot$colores), "colores"] <- "darkgrey"


    ## add whole Term
    library(GO.db)
    GOs_toPlot$LonTerm <- Term(GOs_toPlot$GO.ID)



    set.seed(42)
    Low_Cloud <- ggplot(GOs_toPlot, aes(label = strmultline(LonTerm,
        ratio = 0.4), size = Significant/Annotated)) + geom_text_wordcloud_area(colour = GOs_toPlot$colo
        scale_size_area() + theme_minimal() + ggtitle(condition)

    Low_Cloud
}
```

```
grid.arrange(WordCloud_GOs(res_Gbi_High_weight0_fisher_BP,"G. bimaculatus - High CpGoe" ,0.05),
             WordCloud_GOs(res_Gbi_Low_weight0_fisher_BP,"G. bimaculatus - Low CpGoe", 0.05),

             WordCloud_GOs(res_Lko_High_weight0_fisher_BP,"L. kohalensis - High CpGoe" ,0.05),
             WordCloud_GOs(res_Lko_Low_weight0_fisher_BP,"L. kohalensis- Low CpGoe", 0.05),

            WordCloud_GOs(res_Ame_High_weight0_fisher_BP,"A. mellifera - High CpGoe", 0.05),
            WordCloud_GOs(res_Ame_Low_weight0_fisher_BP,"A. mellifera - Low CpGoe", 0.05),

            WordCloud_GOs(Crick_Low_Ame_High_weight0_fisher_BP,"Crickets Low and Apis High CpGoe -- Fun

             ncol = 2)
```

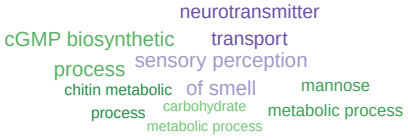
G. bimaculatus − High CpGoe



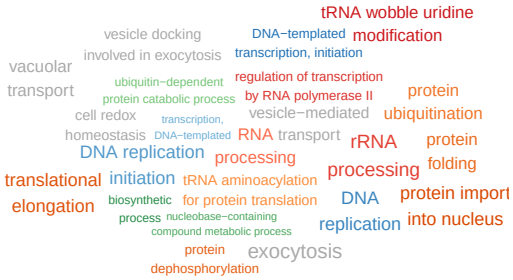
G. bimaculatus − Low CpGoe



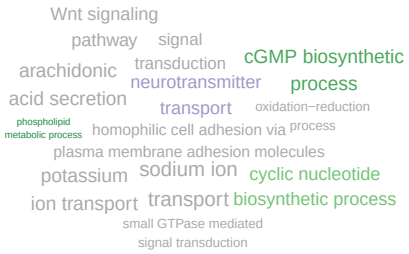
L. kohalensis − High CpGoe



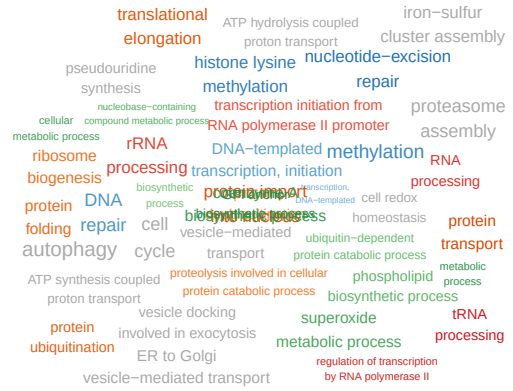
L. kohalensis− Low CpGoe



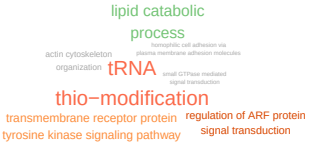
A. mellifera − High CpGoe



A. mellifera − Low CpGoe



Crickets Low and Apis High CpGoe −− Functions form Gbi genes



```
#ggsave("plots/Wordcloud_GbiHigh.png", WordCloud_GOs(res_Gbi_High_weight0_fisher_BP,"G. bimaculatus - H
#ggsave("plots/Wordcloud_Gbilow.png",WordCloud_GOs(res_Gbi_Low_weight0_fisher_BP,"G. bimaculatus - Low

#ggsave("plots/Wordcloud_Lkohigh.png",WordCloud_GOs(res_Lko_High_weight0_fisher_BP,"L. kohalensis - Hig
#ggsave("plots/Wordcloud_LkoLow.png",WordCloud_GOs(res_Lko_Low_weight0_fisher_BP,"L. kohalensis- Low Cp

#ggsave("plots/Wordcloud_AmeHigh.png",WordCloud_GOs(res_Ame_High_weight0_fisher_BP,"A. mellifera - High
#ggsave("plots/Wordcloud_AmeLow.png",WordCloud_GOs(res_Ame_Low_weight0_fisher_BP,"A. mellifera - Low Cp

#ggsave("plots/Wordcloud_CircketsHighAmeLow.png", WordCloud_GOs(Crick_Low_Ame_High_weight0_fisher_BP,"C
```

CpGoe vs dNds

In chunk *'python, preapareOGs'* I had prepared the OG-Gene dictionary. ANd i have all the info in 1 variable: AmeGbiLko_CpGeo_OG

```
# Omega_Gbi_Lko<-read.csv('/home/guillem/WorkingDir/dNdS/Omega_Gbi_Lko.csv',
# header = F, sep = '\t', stringsAsFactors = FALSE)
# Omega_Gbi_Ame<-read.csv('/home/guillem/WorkingDir/dNdS/Omega_Gbi_Ame.csv',
# header = F, sep = '\t', stringsAsFactors = FALSE)
# Omega_Lko_Ame<-read.csv('/home/guillem/WorkingDir/dNdS/Omega_Lko_Ame.csv',
# header = F, sep = '\t', stringsAsFactors = FALSE)
# Omega_Gbi_Lko$OG<-sapply(strsplit(as.character(Omega_Gbi_Lko$V1),'_'),
# `[`, 2) colnames(Omega_Gbi_Lko)<-c('longname','Omega','OG')
# Omega_Gbi_Ame$OG<-sapply(strsplit(as.character(Omega_Gbi_Ame$V1),'_'),
# `[`, 2) colnames(Omega_Gbi_Ame)<-c('longname','Omega','OG')
# Omega_Lko_Ame$OG<-sapply(strsplit(as.character(Omega_Lko_Ame$V1),'_'),
# `[`, 2) colnames(Omega_Lko_Ame)<-c('longname','Omega','OG')
Omega_Gbi_Lko <- read.csv("/home/guillem/WorkingDir/dNdS/Omega_Gbi_Lko_v2.csv",
    header = T, sep = "\t", stringsAsFactors = FALSE)
Omega_Gbi_Ame <- read.csv("/home/guillem/WorkingDir/dNdS/Omega_Gbi_Ame_v2.csv",
    header = T, sep = "\t", stringsAsFactors = FALSE)
Omega_Lko_Ame <- read.csv("/home/guillem/WorkingDir/dNdS/Omega_Lko_Ame_v2.csv",
    header = T, sep = "\t", stringsAsFactors = FALSE)

Omega_Gbi_Lko$OG <- sapply(strsplit(as.character(Omega_Gbi_Lko$key),
    "_"), `[`, 2)
Omega_Gbi_Ame$OG <- sapply(strsplit(as.character(Omega_Gbi_Ame$key),
    "_"), `[`, 2)
Omega_Lko_Ame$OG <- sapply(strsplit(as.character(Omega_Lko_Ame$key),
    "_"), `[`, 2)

dim(Omega_Gbi_Lko)
```

```
## [1] 5728    5
```

```
dim(Omega_Gbi_Ame)
```

```
## [1] 5298    5
```

```
dim(Omega_Lko_Ame)
```

```
## [1] 5009    5
```

```r
dim(Omega_Gbi_Lko)
```

```
## [1] 5728    5
```

```r
Omega_Gbi_Lko_CpG0 <- merge(Omega_Gbi_Lko, AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb ==
    "Gbi", c("OG", "CpGoe", "CpGoe_level")], by.x = "OG", by.y = "OG",
    all.y = FALSE)
Omega_Gbi_Lko_CpG <- merge(Omega_Gbi_Lko_CpG0, AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb ==
    "Lko", c("OG", "CpGoe", "CpGoe_level")], by.x = "OG", by.y = "OG",
    suffixes = c("Gbi", "Lko"), all.y = FALSE)
dim(Omega_Gbi_Lko_CpG)  #3less, probably filter fphe no na, >0 <2...
```

```
## [1] 5725    9
```

```r
Omega_Gbi_Lko_CpG$Both_CpGlevels <- paste(Omega_Gbi_Lko_CpG$CpGoe_levelGbi,
    Omega_Gbi_Lko_CpG$CpGoe_levelLko, sep = "_")
prop.table(table(Omega_Gbi_Lko_CpG$Both_CpGlevels)) * 100
```

```
##
## High_High  High_Low  Low_High   Low_Low
## 20.558952  4.471616 12.663755 62.305677
```

```r
dim(Omega_Gbi_Ame)
```

```
## [1] 5298    5
```

```r
Omega_Gbi_Ame_CpG0 <- merge(Omega_Gbi_Ame, AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb ==
    "Gbi", c("OG", "CpGoe", "CpGoe_level")], by.x = "OG", by.y = "OG",
    all.y = FALSE)
Omega_Gbi_Ame_CpG <- merge(Omega_Gbi_Ame_CpG0, AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb ==
    "Ame", c("OG", "CpGoe", "CpGoe_level")], by.x = "OG", by.y = "OG",
    suffixes = c("Gbi", "Ame"), all.y = FALSE)
dim(Omega_Gbi_Ame_CpG)  # 5 less, probably filter fphe no na, >0 <2...
```

```
## [1] 5293    9
```

```r
Omega_Gbi_Ame_CpG$Both_CpGlevels <- paste(Omega_Gbi_Ame_CpG$CpGoe_levelGbi,
    Omega_Gbi_Ame_CpG$CpGoe_levelAme, sep = "_")
prop.table(table(Omega_Gbi_Ame_CpG$Both_CpGlevels)) * 100
```

```
##
## High_High  High_Low  Low_High   Low_Low
## 15.397695  5.403363 26.034385 53.164557
```

```r
dim(Omega_Lko_Ame)
```

```
## [1] 5009    5
```

```r
Omega_Lko_Ame_CpG0 <- merge(Omega_Lko_Ame, AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb ==
    "Lko", c("OG", "CpGoe", "CpGoe_level")], by.x = "OG", by.y = "OG",
    all.y = FALSE)
Omega_Lko_Ame_CpG <- merge(Omega_Lko_Ame_CpG0, AmeGbiLko_CpGeo_OG[AmeGbiLko_CpGeo_OG$Abb ==
    "Ame", c("OG", "CpGoe", "CpGoe_level")], by.x = "OG", by.y = "OG",
    suffixes = c("Lko", "Ame"), all.y = FALSE)
dim(Omega_Lko_Ame_CpG)  #1less, probably filter fphe no na, >0 <2...
```

```
## [1] 5001    9
```

```
Omega_Lko_Ame_CpG$Both_CpGlevels <- paste(Omega_Lko_Ame_CpG$CpGoe_levelLko,
    Omega_Lko_Ame_CpG$CpGoe_levelAme, sep = "_")
prop.table(table(Omega_Lko_Ame_CpG$Both_CpGlevels)) * 100
```

```
##
## High_High  High_Low  Low_High   Low_Low
## 19.816037  9.418116 21.315737 49.450110
```

```
Omega_Lko_Ame_CpG["Error" %in% Omega_Lko_Ame_CpG$Omega, ]
```

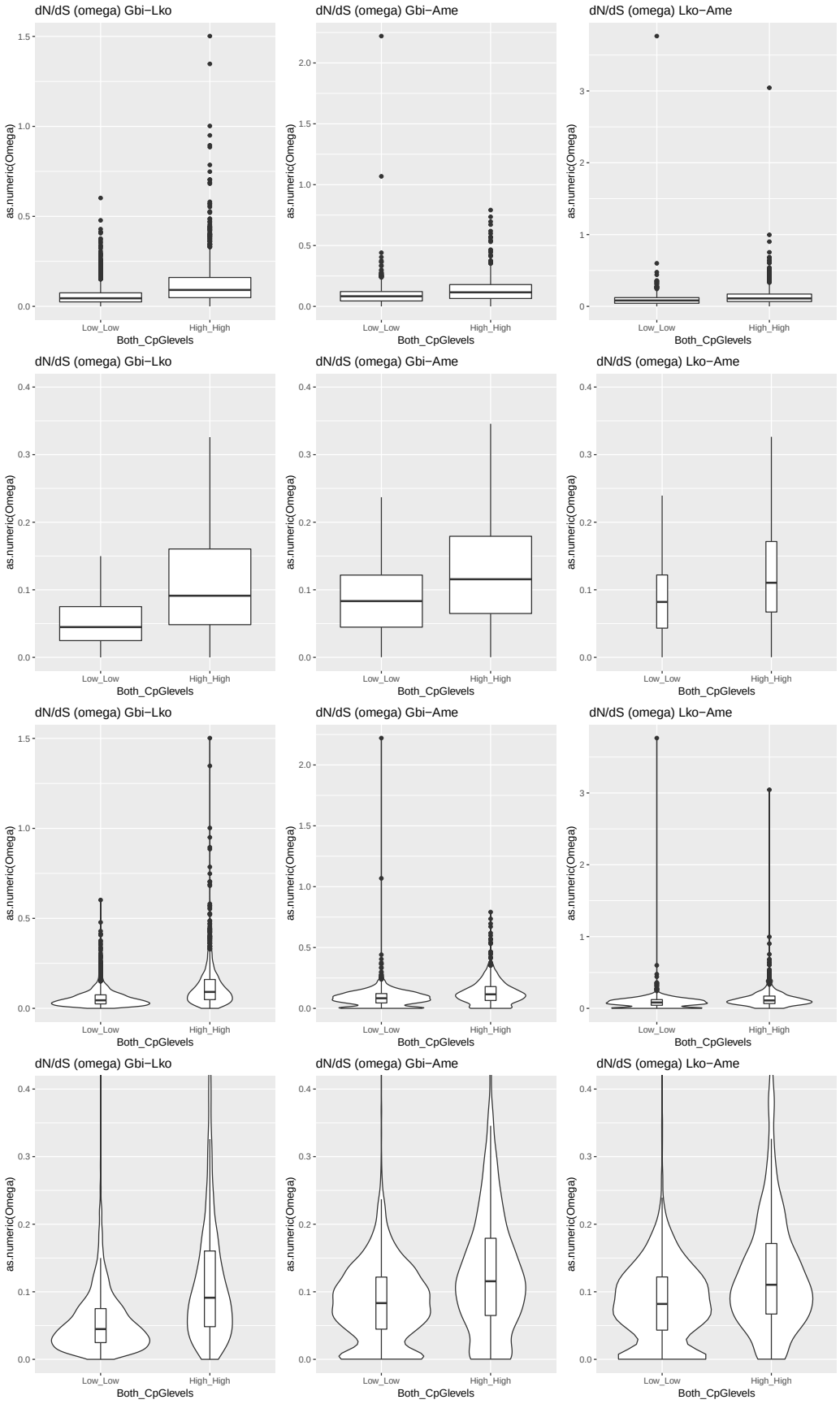
```
##  [1] OG             key            Omega          dN
##  [5] dS             CpGoeLko       CpGoe_levelLko CpGoeAme
##  [9] CpGoe_levelAme Both_CpGlevels
## <0 rows> (or 0-length row.names)
```

```
## remove genes failed dnds
Omega_Gbi_Lko_CpG <- Omega_Gbi_Lko_CpG[!Omega_Gbi_Lko_CpG$Omega %like%
    "None", ]
Omega_Gbi_Ame_CpG <- Omega_Gbi_Ame_CpG[!Omega_Gbi_Ame_CpG$Omega %like%
    "None", ]
Omega_Lko_Ame_CpG <- Omega_Lko_Ame_CpG[!Omega_Lko_Ame_CpG$Omega %like%
    "None", ]


Omega_Gbi_Lko_CpG$Both_CpGlevels <- as.factor(Omega_Gbi_Lko_CpG$Both_CpGlevels)
Omega_Gbi_Ame_CpG$Both_CpGlevels <- as.factor(Omega_Gbi_Ame_CpG$Both_CpGlevels)
Omega_Lko_Ame_CpG$Both_CpGlevels <- as.factor(Omega_Lko_Ame_CpG$Both_CpGlevels)

Omega_Gbi_Lko_CpG$Both_CpGlevels <- factor(Omega_Gbi_Lko_CpG$Both_CpGlevels,
    levels = c("Low_Low", "High_High", "High_Low", "Low_High"))
Omega_Gbi_Ame_CpG$Both_CpGlevels <- factor(Omega_Gbi_Ame_CpG$Both_CpGlevels,
    levels = c("Low_Low", "High_High", "High_Low", "Low_High"))
Omega_Lko_Ame_CpG$Both_CpGlevels <- factor(Omega_Lko_Ame_CpG$Both_CpGlevels,
    levels = c("Low_Low", "High_High", "High_Low", "Low_High"))
```

```
GbiLko_dnDsCpGBocplot <- ggplot(subset(Omega_Gbi_Lko_CpG, Both_CpGlevels ==
    "High_High" | Both_CpGlevels == "Low_Low"), aes(x = Both_CpGlevels,
    y = as.numeric(Omega))) + geom_boxplot() + ggtitle("dN/dS (omega) Gbi-Lko")

GbiAme_dnDsCpGBocplot <- ggplot(subset(Omega_Gbi_Ame_CpG, Both_CpGlevels ==
    "High_High" | Both_CpGlevels == "Low_Low"), aes(x = Both_CpGlevels,
    y = as.numeric(Omega))) + geom_boxplot() + ggtitle("dN/dS (omega) Gbi-Ame")

LkoAme_dnDsCpGBocplot <- ggplot(subset(Omega_Lko_Ame_CpG, Both_CpGlevels ==
    "High_High" | Both_CpGlevels == "Low_Low"), aes(x = Both_CpGlevels,
    y = as.numeric(Omega))) + geom_boxplot() + ggtitle("dN/dS (omega) Lko-Ame")


GbiLko_dnDsCpGBocplot_nout <- ggplot(subset(Omega_Gbi_Lko_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_boxplot(outlier.size = -1) +
    ggtitle("dN/dS (omega) Gbi-Lko") + coord_cartesian(ylim = c(0,
    0.4))

GbiAme_dnDsCpGBocplot_nout <- ggplot(subset(Omega_Gbi_Ame_CpG,
```

```
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_boxplot(outlier.size = -1) +
    ggtitle("dN/dS (omega) Gbi-Ame") + coord_cartesian(ylim = c(0,
    0.4))

LkoAme_dnDsCpGBocplot_nout <- ggplot(subset(Omega_Lko_Ame_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_boxplot(outlier.size = -1,
    width = 0.1) + ggtitle("dN/dS (omega) Lko-Ame") + coord_cartesian(ylim = c(0,
    0.4))



GbiLko_dnDsCpGBviolinplot <- ggplot(subset(Omega_Gbi_Lko_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_violin() +
    geom_boxplot(width = 0.1) + ggtitle("dN/dS (omega) Gbi-Lko")


GbiAme_dnDsCpGBviolinplot <- ggplot(subset(Omega_Gbi_Ame_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_violin() +
    geom_boxplot(width = 0.1) + ggtitle("dN/dS (omega) Gbi-Ame")

LkoAme_dnDsCpGBviolinplot <- ggplot(subset(Omega_Lko_Ame_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_violin() +
    geom_boxplot(width = 0.1) + ggtitle("dN/dS (omega) Lko-Ame")


GbiLko_dnDsCpGBviolin_nout <- ggplot(subset(Omega_Gbi_Lko_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_violin() +
    geom_boxplot(outlier.size = -1, width = 0.1) + ggtitle("dN/dS (omega) Gbi-Lko") +
    coord_cartesian(ylim = c(0, 0.4))

GbiAme_dnDsCpGBviolin_nout <- ggplot(subset(Omega_Gbi_Ame_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_violin() +
    geom_boxplot(outlier.size = -1, width = 0.1) + ggtitle("dN/dS (omega) Gbi-Ame") +
    coord_cartesian(ylim = c(0, 0.4))

LkoAme_dnDsCpGBviolin_nout <- ggplot(subset(Omega_Lko_Ame_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_violin() +
    geom_boxplot(outlier.size = -1, width = 0.1, width = 0.1) +
    ggtitle("dN/dS (omega) Lko-Ame") + coord_cartesian(ylim = c(0,
    0.4))



grid.arrange(GbiLko_dnDsCpGBocplot, GbiAme_dnDsCpGBocplot, LkoAme_dnDsCpGBocplot,
    GbiLko_dnDsCpGBocplot_nout, GbiAme_dnDsCpGBocplot_nout, LkoAme_dnDsCpGBocplot_nout,
```

```
    GbiLko_dnDsCpGBviolinplot, GbiAme_dnDsCpGBviolinplot, LkoAme_dnDsCpGBviolinplot,
    GbiLko_dnDsCpGBviolin_nout, GbiAme_dnDsCpGBviolin_nout, LkoAme_dnDsCpGBviolin_nout,
    ncol = 3)
```


dN/dS (omega) Gbi−Lko
dN/dS (omega) Gbi−Ame
dN/dS (omega) Lko−Ame
as.numeric(Omega)
Low_Low
High_High
Both_CpGlevels

```
GbiLko_dnDsCpGBviolinplot_forpaper <- ggplot(subset(Omega_Gbi_Lko_CpG,
    Both_CpGlevels == "High_High" | Both_CpGlevels == "Low_Low"),
    aes(x = Both_CpGlevels, y = as.numeric(Omega))) + geom_violin() +
    geom_boxplot(width = 0.1) + ggtitle("dN/dS (omega) Gbi-Lko") +
    ylab("Omega (dD/dS)") + xlab("CpGo/e level in both insects")

# ggsave('plots/GbiLko_dnDsCpGBviolinplot.png',GbiLko_dnDsCpGBviolinplot_forpaper)
```

I have observed that thd dN and dS values with *Apis mellifera* are really high! 99% of them dN or dS higher than 2.... are to far away related, and satuirated with mutations.

```
dim(Omega_Gbi_Lko_CpG[Omega_Gbi_Lko_CpG$dN > 2 | Omega_Gbi_Lko_CpG$dS >
    2, ])[1]/dim(Omega_Gbi_Lko_CpG)[1] * 100
```

```
## [1] 77.41485
```

```
dim(Omega_Gbi_Ame_CpG[Omega_Gbi_Ame_CpG$dN > 2 | Omega_Gbi_Ame_CpG$dS >
    2, ])[1]/dim(Omega_Gbi_Ame_CpG)[1] * 100
```

```
## [1] 98.80501
```

```
dim(Omega_Lko_Ame_CpG[Omega_Lko_Ame_CpG$dN > 2 | Omega_Lko_Ame_CpG$dS >
    2, ])[1]/dim(Omega_Lko_Ame_CpG)[1] * 100
```

```
## [1] 98.55451
```

```
# T-test NOT approproate t.test(subset(Omega_Gbi_Lko_CpG,
# Both_CpGlevels=='High_High')$Omega,
# subset(Omega_Gbi_Lko_CpG, Both_CpGlevels=='Low_Low')$Omega)

## It applies Wilcoxon-Mann-Whitney test (Mann-Whitney one
## data is not paired)
wilcox.test(subset(Omega_Gbi_Lko_CpG, Both_CpGlevels == "High_High")$Omega,
    subset(Omega_Gbi_Lko_CpG, Both_CpGlevels == "Low_Low")$Omega,
    paired = F)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  subset(Omega_Gbi_Lko_CpG, Both_CpGlevels == "High_High")$Omega and subset(Omega_Gbi_Lko_CpG, 
## W = 3027728, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```