EEB313 group B FINAL CODE

2024-12-10

```
require(tidyverse)
require(lme4)
require(deSolve)
require(vegan)
require(car)
require(rgl)
require(ggfortify)
require(fitdistrplus)

options(rgl.useNULL = TRUE)
rgl::setupKnitr(autoprint = TRUE)
```

```
read_csv("Arthropod_List.csv") -> Arthropod_List
```

```
## Rows: 349 Columns: 79
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): Order, Family, Species, Trophic_Group
## dbl (75): OP1, OP2, OP3, OP4, OP5, OP6, OP7, OP8, OP9, OP10, OP11, OP12, OP1...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
read_csv("Plant_Traits.csv") -> Plant_Traits
```

```
## Rows: 75 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): Site, Leaf_N
## dbl (4): Aboveground_Biomass, Plant_Density, Leaf_C, Leaf_P
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
read_csv("Soil_Traits.csv") -> Soil_Traits
```

```
## Rows: 75 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Site
## dbl (6): Soil_C, Soil_N, Soil_P, Soil_pH, Soil_Salinity, Soil_Water
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

REARRANGE ARTHROPOD_LIST DATA FRAME SO THAT EACH SPECIES IS A COLUMN AND EACH SITE IS A ROW

```
site_cols <- Arthropod_List[,5:79]
species_col <- Arthropod_List[c("Species")]

new_AL <- data.frame(species_col, site_cols)

AL_long <- new_AL %>%
  pivot_longer(
    cols = (starts_with("OP") | starts_with("RP") | starts_with("TP") |
              starts_with("IS") | starts_with("PS")), # Select site columns
    names_to = "Site", # Name of new "Site" column
    values_to = "Count" #  # Name of new "Count" column
  )

AL_wide <- AL_long %>%
  pivot_wider(
    names_from = Species, # Create new columns from Species
    values_from = Count   # Fill values from the "Count" column
  )

Rearranged_Arthropod_List <- AL_wide
```

MAKING SURE THE VEGAN PACKAGE DOES WHAT WE THINK IT DOES - DOING SHANNON DIVERSITY BY "HAND"...

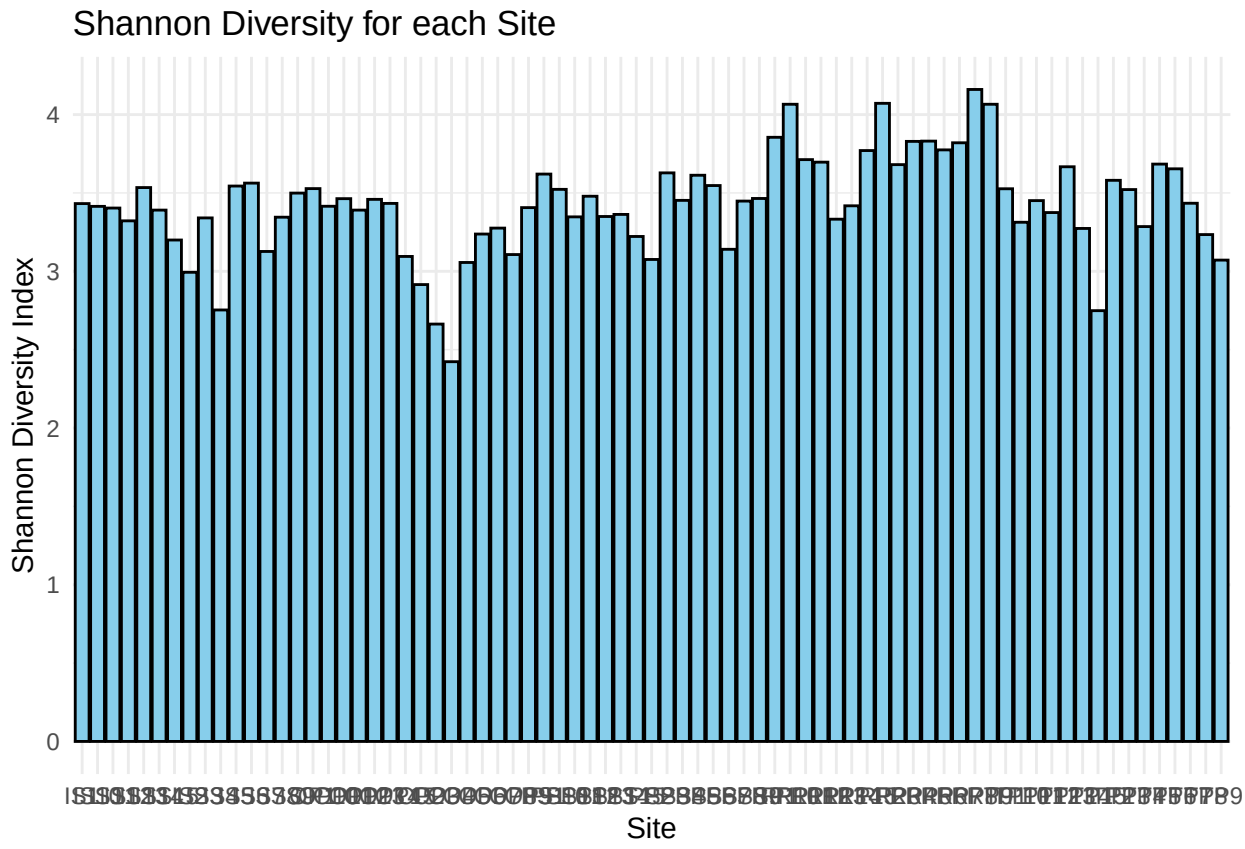
```
Site_Manual <- Rearranged_Arthropod_List$Site
Shannon_Manual <- rep(NA, 75)

Manual_Shannons <- data.frame(Site_Manual, Shannon_Manual)

for (i in 1:75){

  #counts of each species in one site at a time
  species_counts <- Rearranged_Arthropod_List[i,2:350]

  #sum to get total amount of species in that site
  total_individuals <- sum(species_counts)

  # calculate proportions (p_i)
  proportions <- species_counts / total_individuals

  #calculate p_i * ln(p_i) for each species
  pi_ln_pi <- proportions * log(proportions)

  #sum up all the p_i * ln(p_i), na.rm handles any NA values
  shannon_diversity <- -sum(pi_ln_pi, na.rm = TRUE)
```

```
  Manual_Shannons[i,2] <- shannon_diversity
}
```

YAYYYY – WE GOT THE EXACT SAME SHANNON DIVERSITIES!

#Distribution of Shannon Diversity against Site

```
ggplot(Manual_Shannons, aes(x = Site_Manual, y = Shannon_Manual)) +
  geom_bar(stat = "identity", fill = "skyblue", color = "black") +
  theme_minimal() +
  labs(title = "Shannon Diversity for each Site",
       x = "Site", y = "Shannon Diversity Index")
```



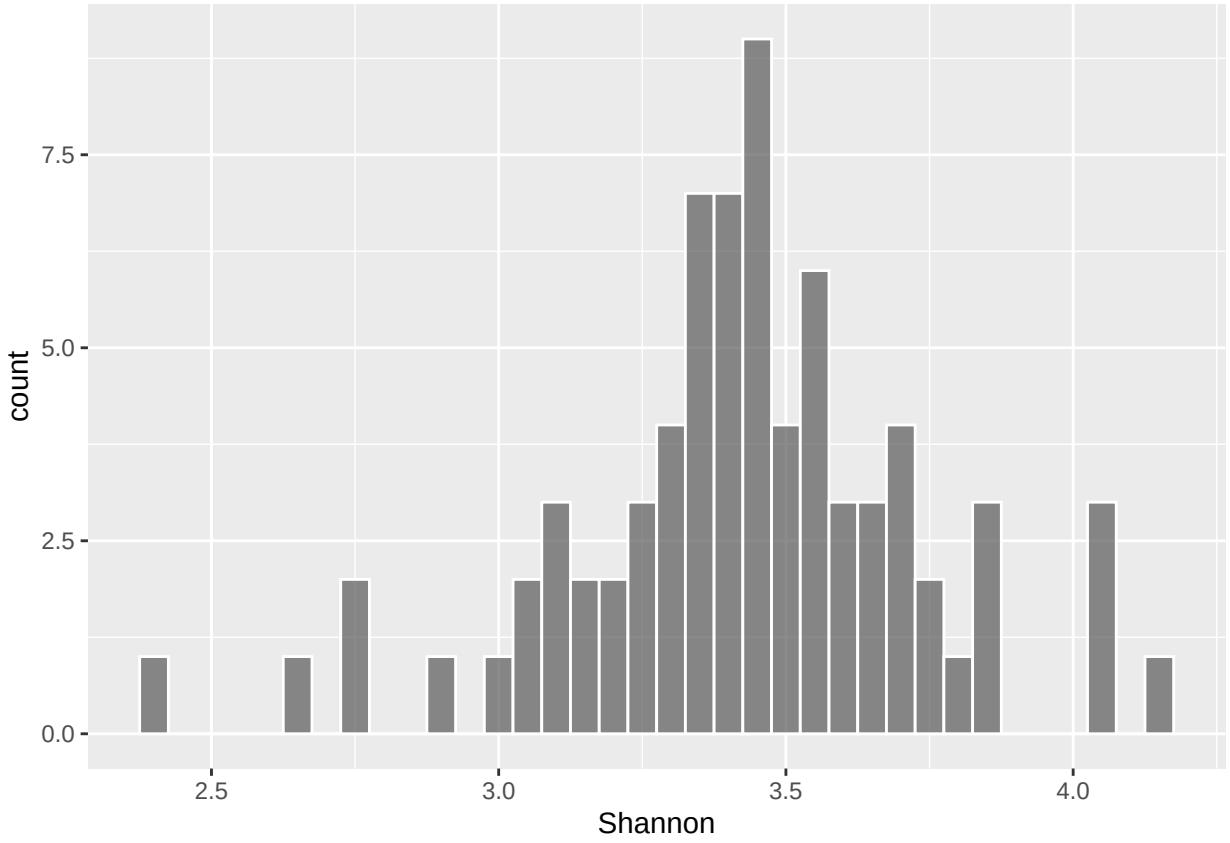
FINAL SHANNON DIVERSITY CALCULATION

```
for_shannon = Rearranged_Arthropod_List[,2:350]

shannon_indices = diversity(for_shannon, index = "shannon")
view(shannon_indices)
```

```
final_shannon = data.frame(Site = Rearranged_Arthropod_List$Site,
                           Shannon = shannon_indices)
```

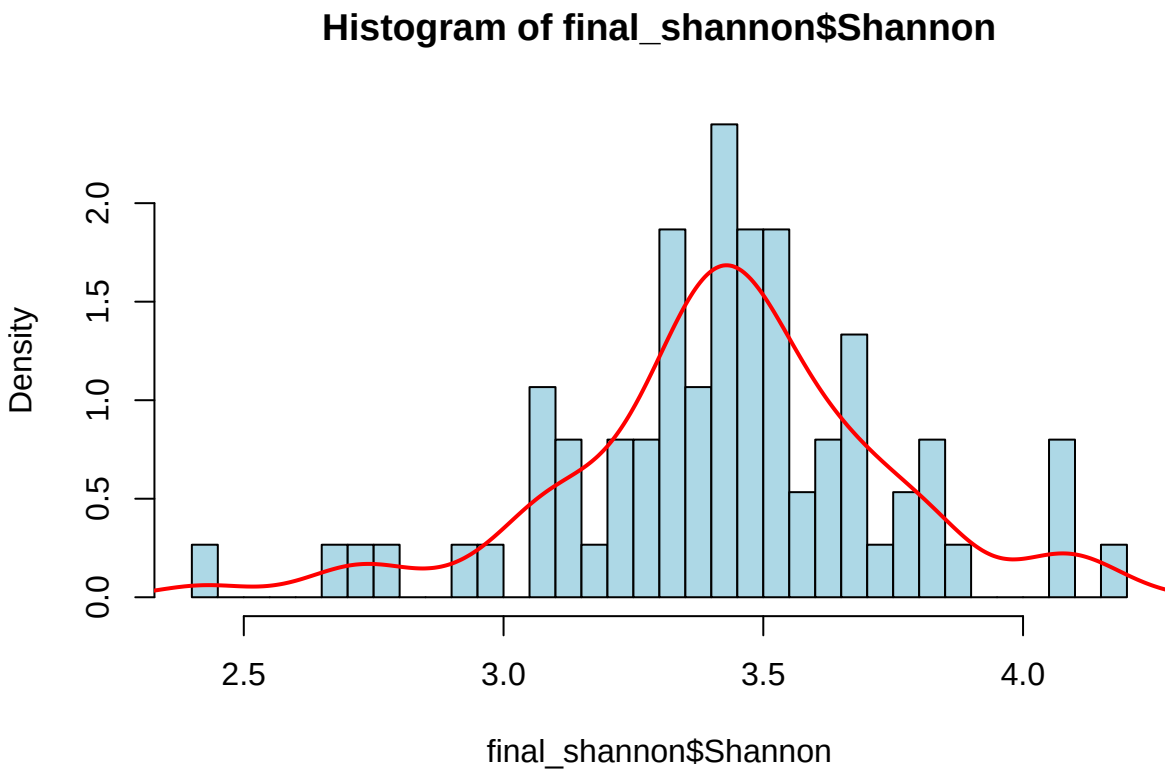
#Histogram of Shannon Diversity – kinda normal lookin' ?!

```
ggplot(final_shannon, aes(Shannon)) +
  geom_histogram(binwidth = 0.05, alpha = 0.7, colour="white")
```



#Another way to histogram... still kinda normal looking?!?!

```
hist(final_shannon$Shannon, probability = TRUE, col = "lightblue",
     border = "black", breaks = 25)
# Adds a density curve
lines(density(final_shannon$Shannon), col = "red", lwd = 2)
```



Group Sites into Communities

```
final_shannon$Community <-
  ifelse(startsWith(final_shannon$Site, "OP"), "OP",
         ifelse(startsWith(final_shannon$Site, "RP"), "RP",
                ifelse(startsWith(final_shannon$Site, "TP"), "TP",
                       ifelse(startsWith(final_shannon$Site, "IS"), "IS",
                              ifelse(startsWith(final_shannon$Site, "PS"), "PS",
                                "Other")))))
```

Move Communities column to the left to be beside the Site column...

```
final_shannon <- final_shannon[, c("Site", "Community", "Shannon")]
```

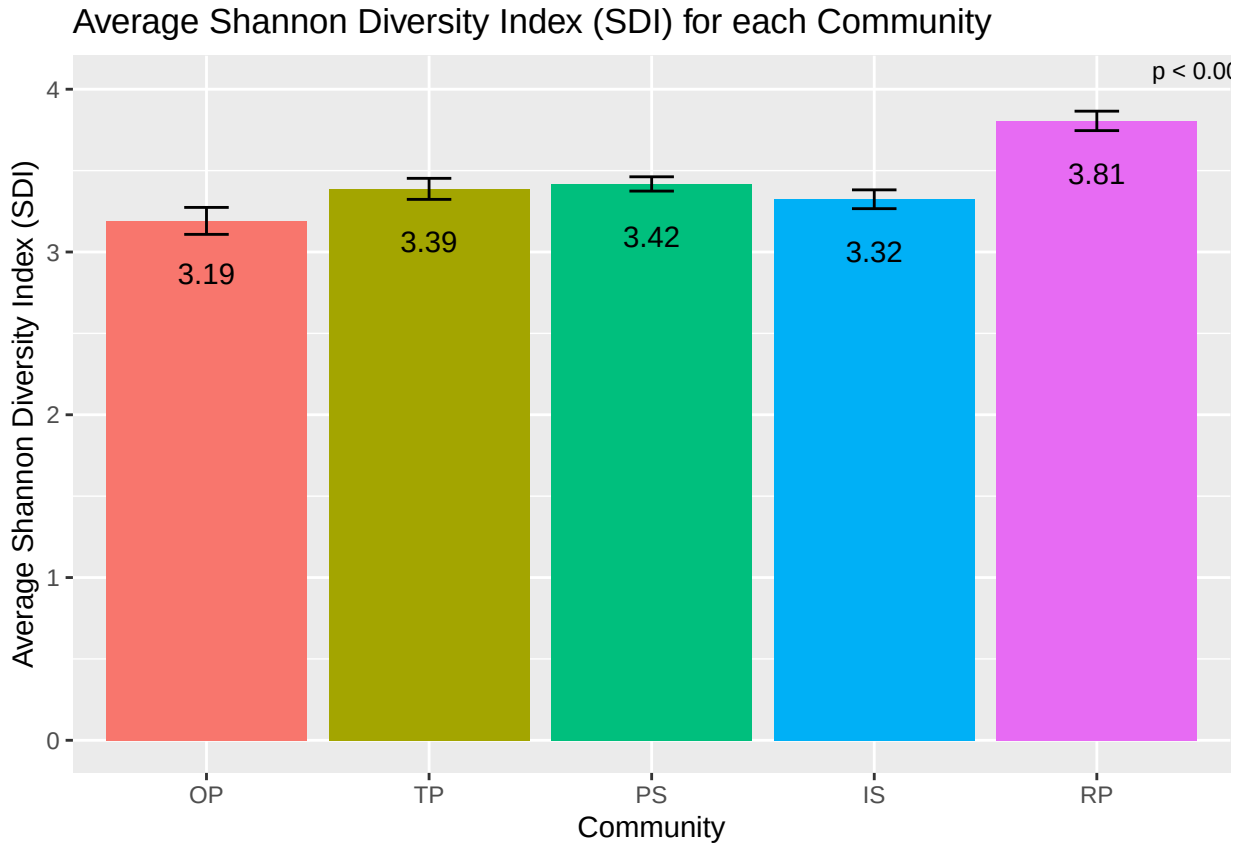
Find the mean of the Shannon diversity indices for each of the five communities

And also find the standard error for each community (used later to make error bars)

```
bar_plot_shannon <- final_shannon %>%
  group_by(Community) %>%
  summarise(Avg_Shannon = mean(Shannon),
            SE_Shannon = sd(Shannon)/sqrt(length(Shannon)))

# Reorder the Community factor levels to represent the "gradient" of change
# from fully native to fully invaded to restored back to native...
# (This is just for a better visuals -- also aligns with OG paper)
bar_plot_shannon$Community <- factor(bar_plot_shannon$Community,
                                     levels = c("OP", "TP", "PS", "IS", "RP"))
```

#Average SDI of each community with error bars!

```
ggplot(bar_plot_shannon, aes(x = Community, y = Avg_Shannon, fill = Community)) +
  geom_bar(stat = "identity") +
  geom_errorbar(aes(ymax = Avg_Shannon + SE_Shannon,
                    ymin = Avg_Shannon - SE_Shannon), width = 0.2) +
  # Display rounded Avg_Shannon values
  geom_text(aes(label = round(Avg_Shannon, 2)),
            vjust = 3, color = "black") +
  # Add "p < 0.001" annotation
  annotate("text", x = 5, y = max(bar_plot_shannon$Avg_Shannon) + 0.2,
           label = "p < 0.001", size = 3, hjust = -0.55, vjust = -0.55) +
  labs(title = "Average Shannon Diversity Index (SDI) for each Community",
       x = "Community",
       y = "Average Shannon Diversity Index (SDI)") +
  theme(legend.position = "none") # Remove that redundant legend!
```



Find the significance using ANOVA (what's the p-value?)

```
# Perform an ANOVA
anova_result <- aov(Shannon ~ Community, data = final_shannon)

# Summary of the ANOVA
summary(anova_result)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## Community    4  3.166  0.7916   13.28 4.23e-08 ***
## Residuals   70  4.174  0.0596
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

So the p-value for Community is 4.23e-08 or 0.0000000423. This extremely small p-value indicates a highly significant effect of the Community factor on the response variable.

Standard Error for Entire Dataset

```
SE_total <- sd(final_shannon$Shannon) / sqrt(nrow(final_shannon))
SE_total
```

```
## [1] 0.03636722
```

Add all the predictor variable columns (soil and plant traits) onto final_shannon...

```
final_shannon$soil_C <- Soil_Traits$Soil_C
final_shannon$soil_N <- Soil_Traits$Soil_N
final_shannon$soil_P <- Soil_Traits$Soil_P
final_shannon$soil_pH <- Soil_Traits$Soil_pH
final_shannon$soil_sal <- Soil_Traits$Soil_Salinity
final_shannon$soil_wat <- Soil_Traits$Soil_Water

final_shannon$biomass_above <- Plant_Traits$Aboveground_Biomass
final_shannon$plant_dens <- Plant_Traits$Plant_Density
final_shannon$leaf_N <- Plant_Traits$Leaf_N
final_shannon$leaf_C <- Plant_Traits$Leaf_C
final_shannon$leaf_P <- Plant_Traits$Leaf_P
```

Make sure all the number columns are actually numeric!!

Note that this also converts 'n/a' cells into NULL – we will have to replace this later with an estimate!

```
final_shannon[, 3:14] <- lapply(final_shannon[, 3:14], as.numeric)
```

```
## Warning in lapply(final_shannon[, 3:14], as.numeric): NAs introduced by
## coercion
```

First filter out the data to involve only the OP community and then exclude the 2 rows with "n/a" leaf_N values:

```
final_shannon %>%
  filter(final_shannon$Community == "OP",
         !is.na(final_shannon$leaf_N)) -> OPonly_noNA_FS
```

Then replace the 2 cells with the mean of the OP'S leaf_N column!...

```
replaced_FS <- final_shannon

replaced_FS[replaced_FS$Site == "OP14", "leaf_N"] <-
  mean(OPonly_noNA_FS$leaf_N)
replaced_FS[replaced_FS$Site == "OP15", "leaf_N"] <-
  mean(OPonly_noNA_FS$leaf_N)
```

Renaming master data frame to a shorter name bc I will have to use it over and over again in log-likelihood evalutors!...

```
data <- replaced_FS
```

OPTIONAL: Save the data dataframe as a csv file for sharing and future edits...

```
write.csv(data, "GROUP_B_MASTER_DATAFRAME.csv")
```

MAKING RANGES OF PARAMETER COMBONINATIONS FOR EACH DISTRIBUTION

dgamma parameters:

- shape: Controls the shape of the distribution. Low values -> Highly right-skewed. High values -> Symmetric or normal-like.
- scale: Scales the spread of the data (mean and variance). Larger values result in broader distributions.

inverse gaussian parameters:

- mu (mean): Determines the central tendency of the distribution. It should align with the average value of your data.
- lambda (shape): Controls the variance and skewness. Larger implies less skewness and lower variance.

```
# MLE: LogLik = -19.28128, mean = 3.43, sd = 0.31

norm_combos <- expand.grid(mean = seq(2.4, 4.2, 0.01),
                           sd = seq(0.15, 0.7, 0.01))
```

```
# MLE: LogLik = -21.03414, shape = 98, scale = 0.035

gamma_combos <- expand.grid(shape = seq(98, 100, 0.001),
                            scale = seq(0.03, 0.04, 0.001))
```

```
# MLE: LogLik = -21.50412, mu = 3.43, lambda = 381.81

invgaus_combos <- expand.grid(mu = seq(2.4, 4.2, 0.01),
                              lambda = seq(375, 385, 0.01))
```

UPDATED METE'S RANGES

```
# MLE: -19.28128, mean = 3.43, sd = 0.31
norm_combos <- expand.grid(mean = seq(2.4, 4.2, 0.01),
                           sd = seq(0.15, 0.7, 0.01))
```

```r
# MLE: LogLik = -20.56567, shape = 116.1, scale = 0.0295
gamma_combos <- expand.grid(shape = seq(100, 150, 0.1),
                            scale = seq(0.02, 0.04, 0.0005))

# MLE: LogLik = -21.72646, mu = 3.4, lambda = 379.6
invgaus_combos <- expand.grid(mu = seq(2, 6, 0.1),
                              lambda = seq(200, 600, 0.1))
```

LOG-LIKELIHOOD EVALUATORS

```r
norm_loglik <- function(x = data$Shannon, mean, sd){
  return (sum(log(dnorm(x = x, mean = mean, sd = sd))))
}
```

```r
gamma_loglik <- function(x = data$Shannon, shape, scale){
  return (sum(log(dgamma(x = x, shape = shape, scale = scale))))
}
```

```r
invgaus_loglik <- function(x = data$Shannon, mu, lambda){
  return (sum(log(
    sqrt(lambda/(2*pi*x^3)) * exp((-lambda*(x-mu)^2)/(2*x*mu^2))
    )))
}
```

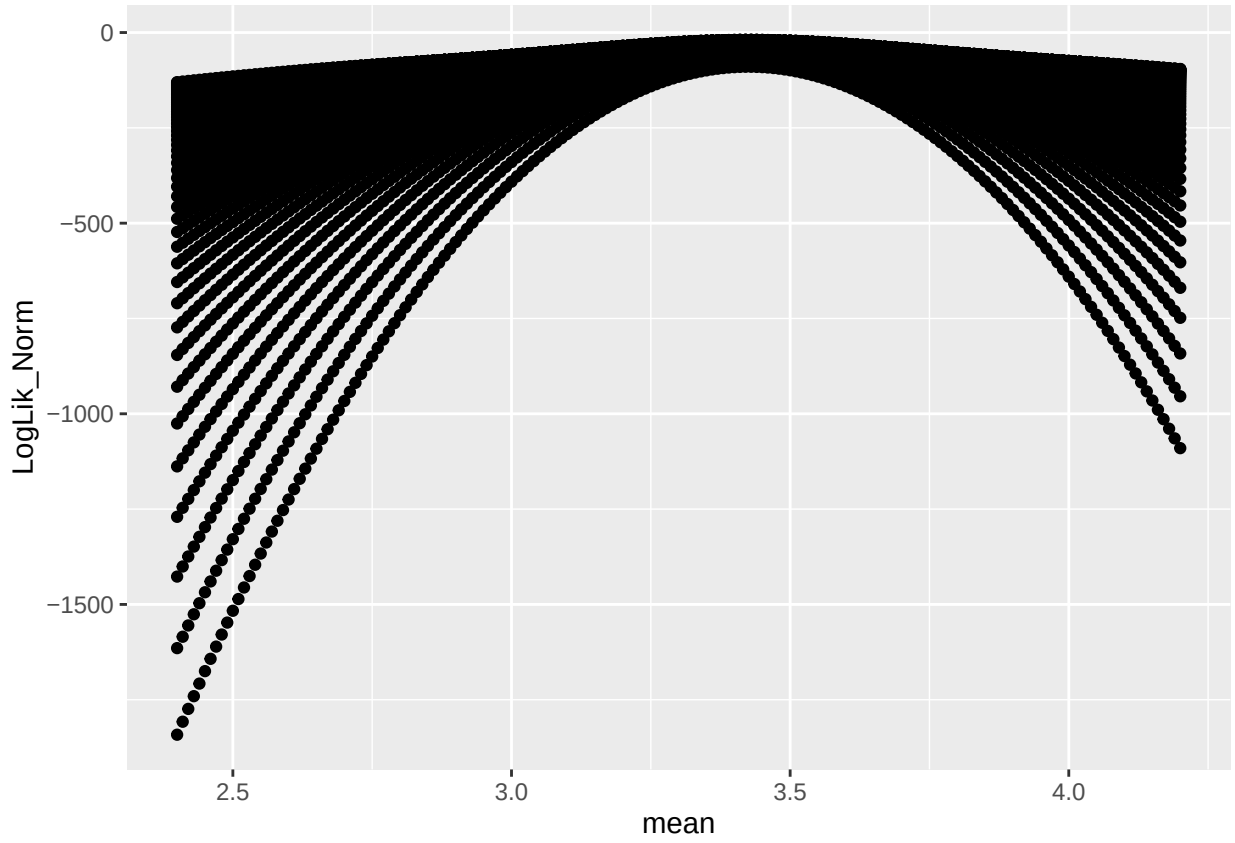
```r
LogLik_norm <- c() # empty log-likelihoods vector for NORMAL

for (i in 1:nrow(norm_combos)){
  LogLik_norm[i] <- norm_loglik(mean = norm_combos$mean[i],
                                sd = norm_combos$sd[i])
}

LLs_norm <- data.frame(LogLik_Norm = LogLik_norm,
                       mean = norm_combos$mean,
                       sd = norm_combos$sd)
```

```r
LogLik_gamma <- c() # empty log-likelihoods vector for GAMMA

for (i in 1:nrow(gamma_combos)){
  LogLik_gamma[i] <- gamma_loglik(shape = gamma_combos$shape[i],
                                  scale = gamma_combos$scale[i])
}

LLs_gamma <- data.frame(LogLik_Gamma = LogLik_gamma,
                       shape = gamma_combos$shape,
                       scale = gamma_combos$scale)
```

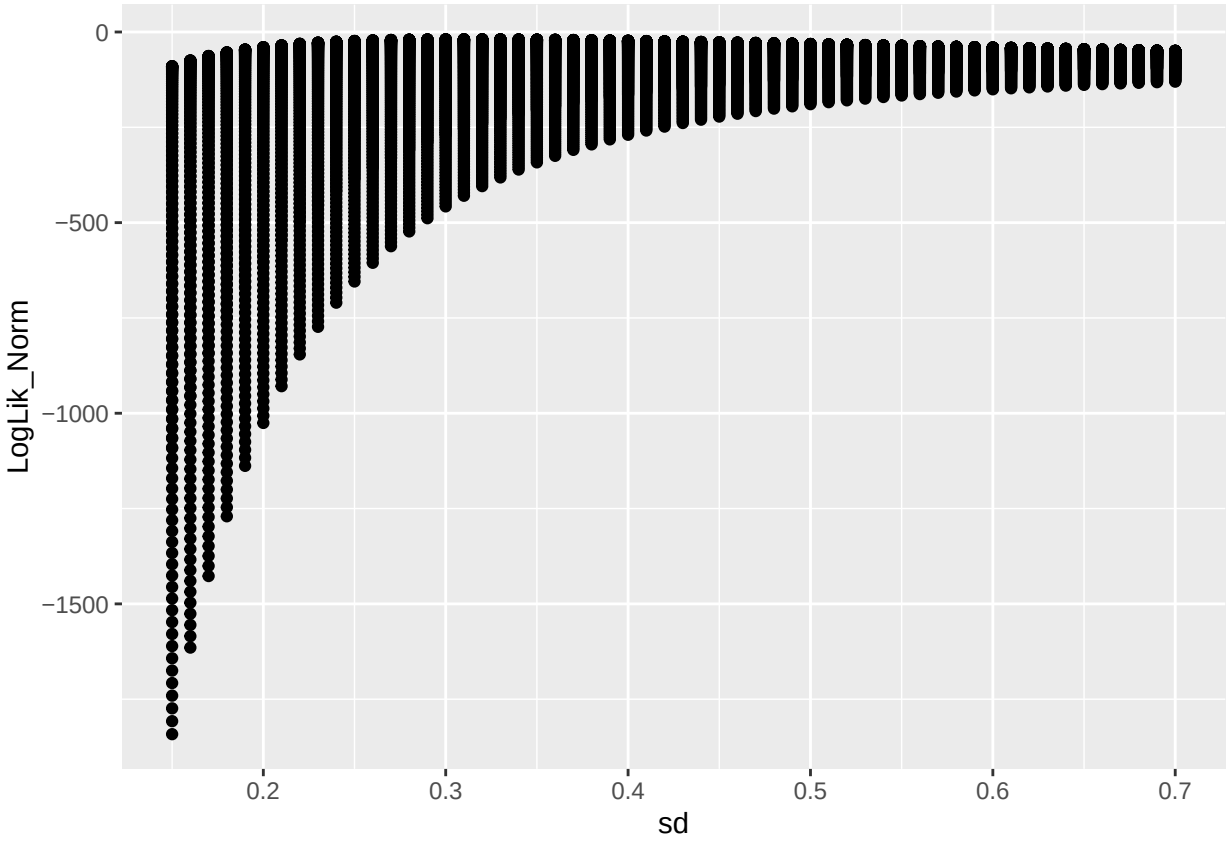
```r
LogLik_invgaus <- c() # empty log-likelihoods vector for INVERSE GAUSSIAN

for (i in 1:nrow(invgaus_combos)){
```

```
  LogLik_invgaus[i] <- invgaus_loglik(mu = invgaus_combos$mu[i],
                                      lambda = invgaus_combos$lambda[i])
}

LLs_invgaus <- data.frame(LogLik_InvGaus = LogLik_invgaus,
                          mu = invgaus_combos$mu,
                          lambda = invgaus_combos$lambda)
```

```
ggplot(LLs_norm, aes(x = mean, y = LogLik_Norm)) + geom_point()
```



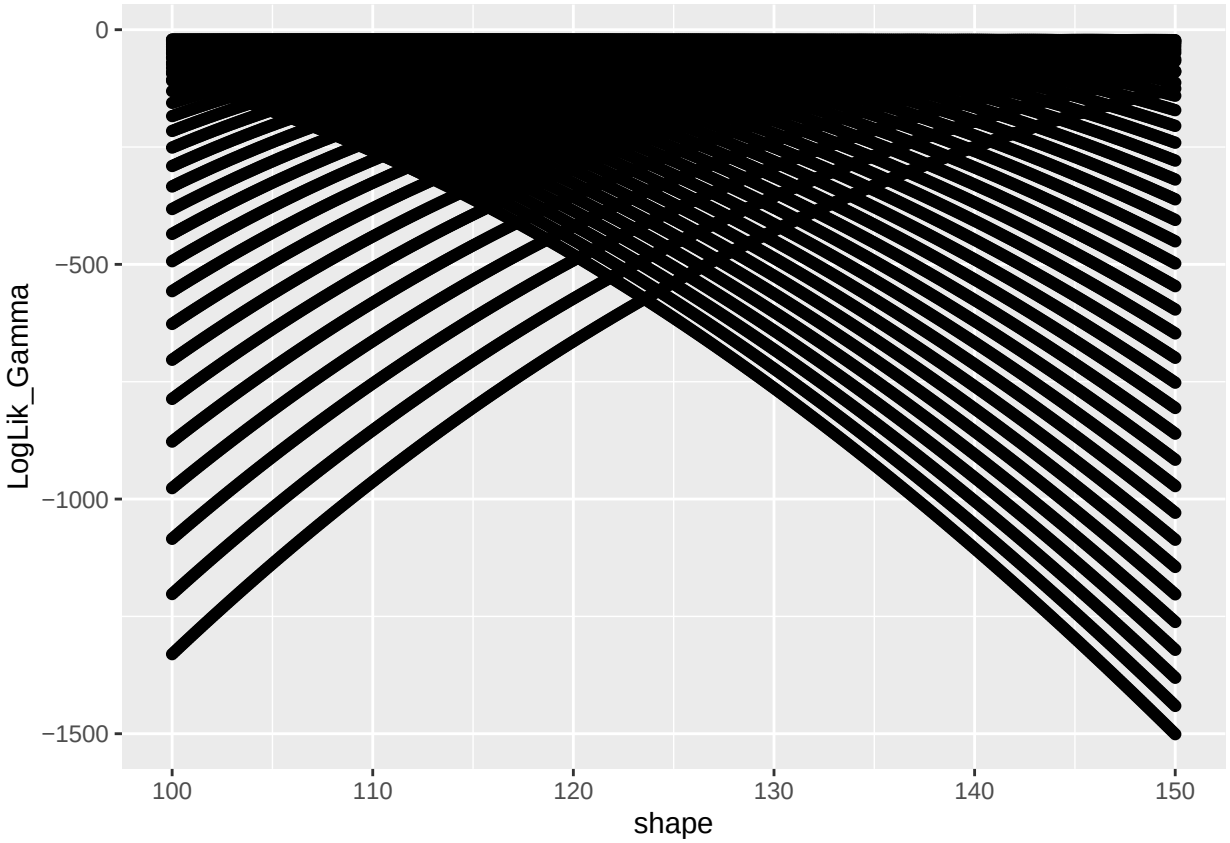

```
ggplot(LLs_norm, aes(x = sd, y = LogLik_Norm)) + geom_point()
```



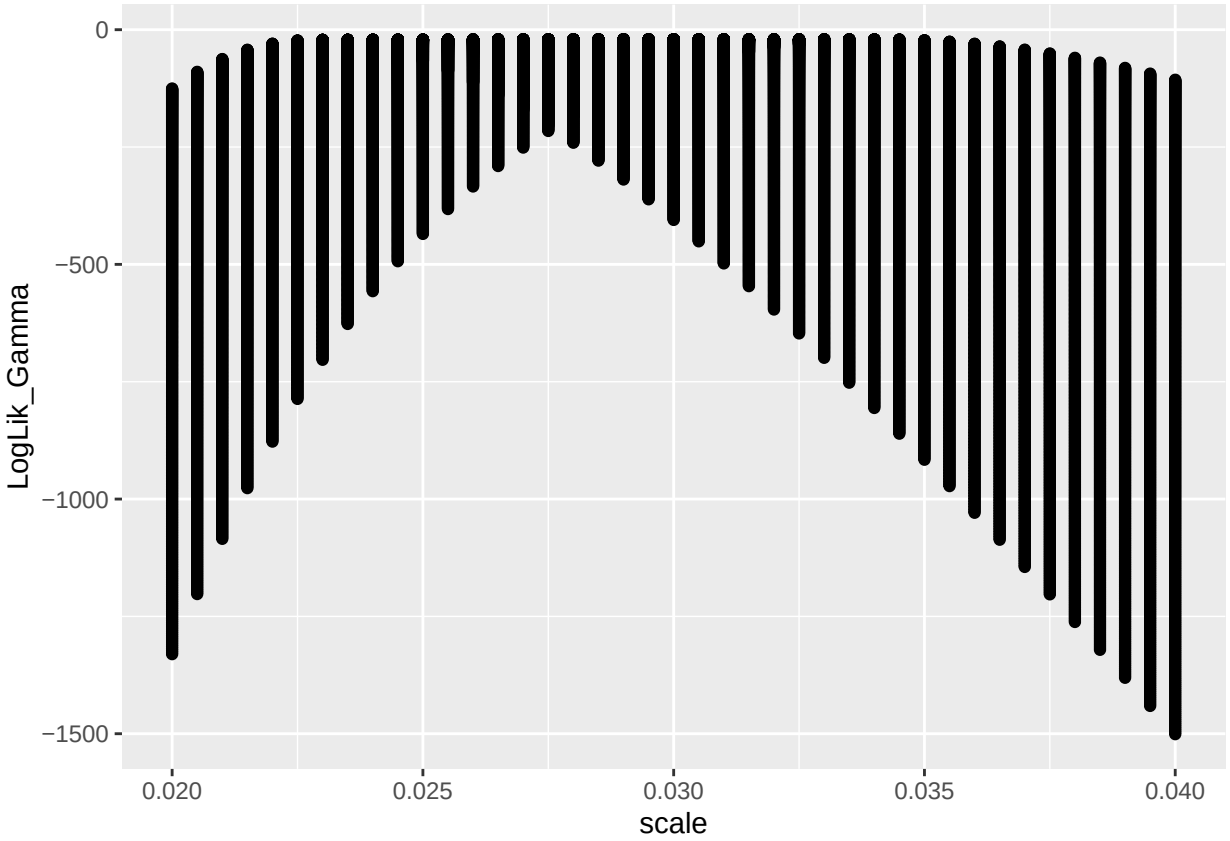

```
MLE_norm <- LLs_norm %>% subset(LogLik_Norm == max(LogLik_Norm))
MLE_norm
```

```
##      LogLik_Norm mean   sd
## 3000   -19.28128 3.43 0.31
```

```
ggplot(LLs_gamma, aes(x = shape, y = LogLik_Gamma)) + geom_point()
```



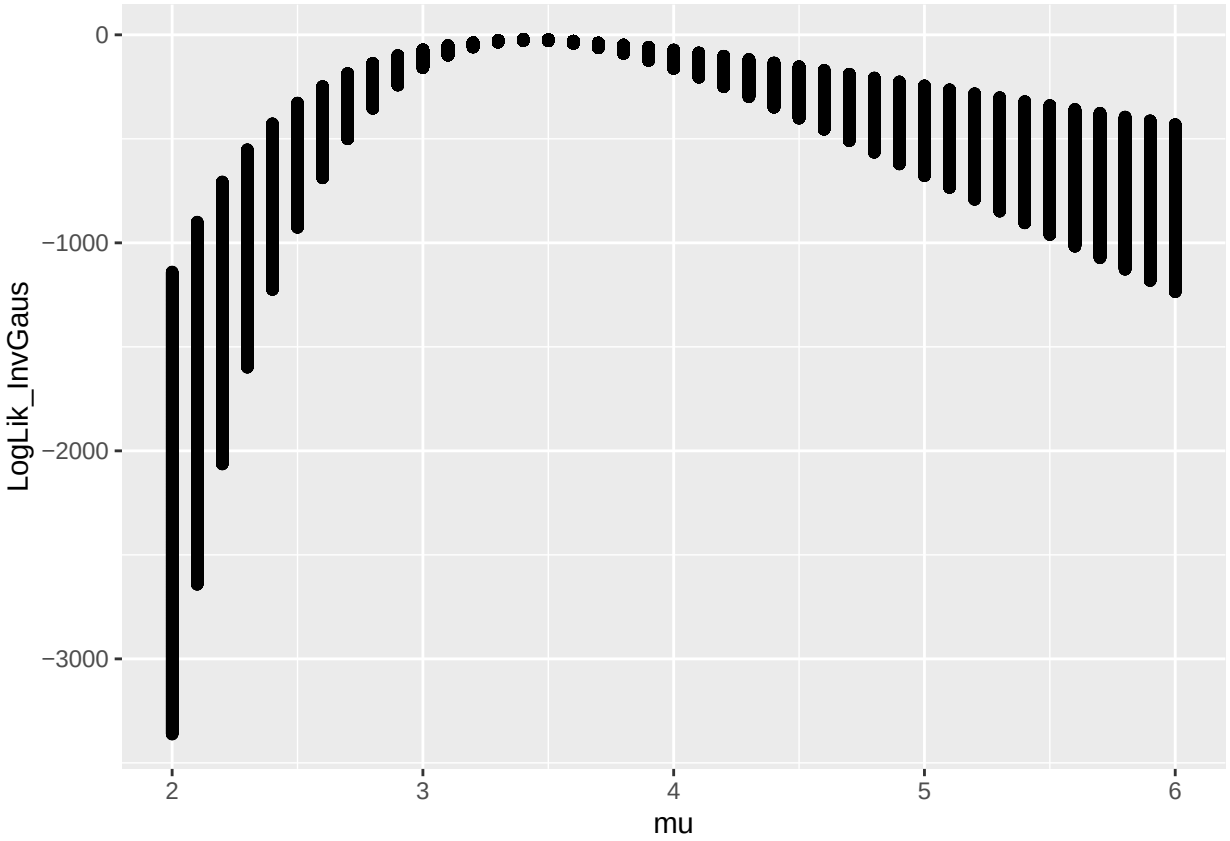

```
ggplot(LLs_gamma, aes(x = scale, y = LogLik_Gamma)) + geom_point()
```



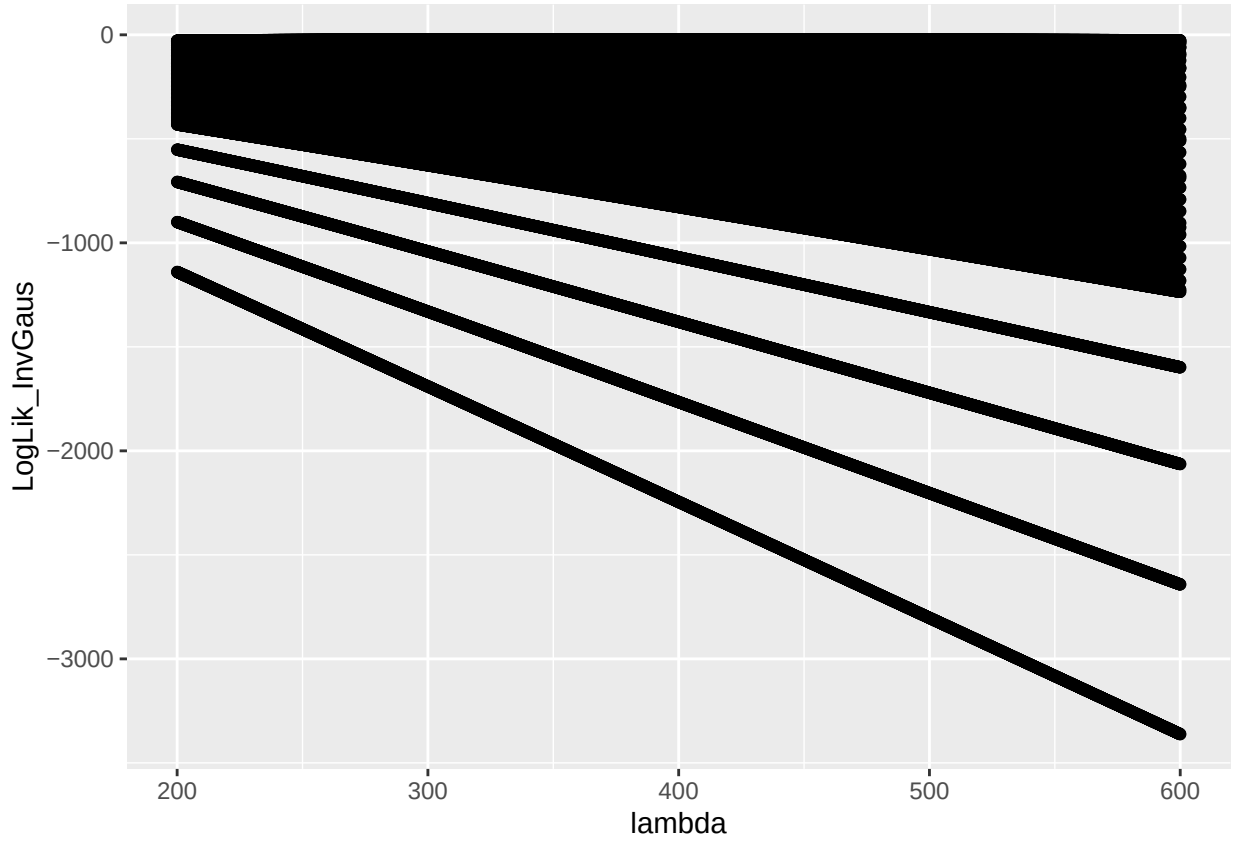

```
MLE_gamma <- LLs_gamma %>% subset(LogLik_Gamma == max(LogLik_Gamma))
MLE_gamma
```

```
##      LogLik_Gamma shape  scale
## 9681    -20.56567 116.1 0.0295
```

```
ggplot(LLs_invgaus, aes(x = mu, y = LogLik_InvGaus)) + geom_point()
```




```
ggplot(LLs_invgaus, aes(x = lambda, y = LogLik_InvGaus)) + geom_point()
```

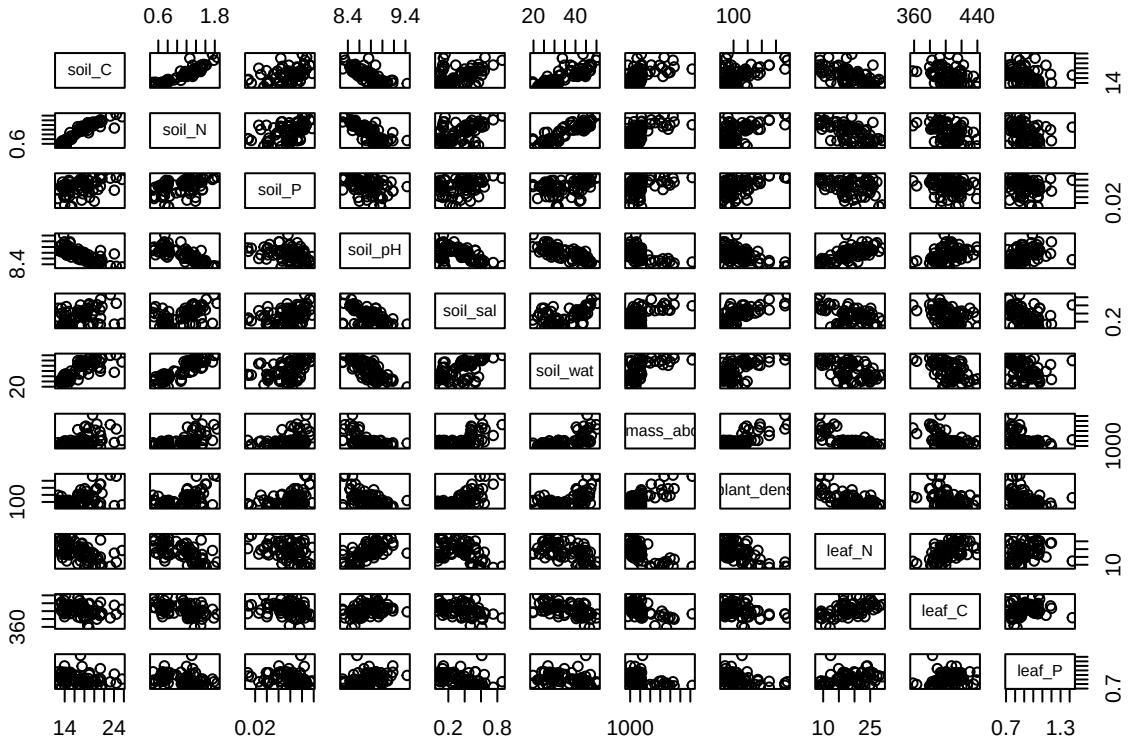



```
MLE_invgaus <- LLs_invgaus %>% subset(LogLik_InvGaus == max(LogLik_InvGaus))
MLE_invgaus
```

```
##       LogLik_InvGaus  mu lambda
## 73651      -21.72646 3.4  379.6
```

Find which varaiables are linearly and positively correlated...

```
par(mfrow=c(1,1))

pairs(data[, c("soil_C", "soil_N", "soil_P", "soil_pH", "soil_sal", "soil_wat",
               "biomass_above", "plant_dens", "leaf_N", "leaf_C", "leaf_P")])
```



#Test for co-linearality?

```
cor_matrix <- cor(data[, c("soil_C", "soil_N", "soil_P", "soil_pH", "soil_sal",
                           "soil_wat", "biomass_above", "plant_dens", "leaf_N",
                           "leaf_C", "leaf_P")],
                  use = "pairwise.complete.obs")
print(cor_matrix)
```

```
##                  soil_C     soil_N     soil_P    soil_pH   soil_sal   soil_wat
## soil_C         1.0000000  0.9269588  0.4079972 -0.7594546  0.3689419  0.7843768
## soil_N         0.9269588  1.0000000  0.4612699 -0.7713667  0.4504365  0.8769814
## soil_P         0.4079972  0.4612699  1.0000000 -0.3512759  0.3214240  0.3975461
## soil_pH       -0.7594546 -0.7713667 -0.3512759  1.0000000 -0.5082347 -0.7704241
## soil_sal       0.3689419  0.4504365  0.3214240 -0.5082347  1.0000000  0.5721562
## soil_wat       0.7843768  0.8769814  0.3975461 -0.7704241  0.5721562  1.0000000
## biomass_above  0.4958170  0.5719650  0.3556092 -0.5908822  0.5816223  0.5926635
## plant_dens     0.4596174  0.5237192  0.4443572 -0.5009611  0.7244117  0.5890475
## leaf_N        -0.5434564 -0.6102046 -0.4077548  0.6527521 -0.4116900 -0.5335999
## leaf_C        -0.3695966 -0.4709862 -0.2610081  0.3741878 -0.4108897 -0.5415672
## leaf_P        -0.2657054 -0.2912269 -0.1731799  0.3522477 -0.3349665 -0.2966488
##                biomass_above plant_dens     leaf_N     leaf_C     leaf_P
## soil_C             0.4958170  0.4596174 -0.5434564 -0.3695966 -0.2657054
## soil_N             0.5719650  0.5237192 -0.6102046 -0.4709862 -0.2912269
## soil_P             0.3556092  0.4443572 -0.4077548 -0.2610081 -0.1731799
## soil_pH           -0.5908822 -0.5009611  0.6527521  0.3741878  0.3522477
## soil_sal           0.5816223  0.7244117 -0.4116900 -0.4108897 -0.3349665
```

```
## soil_wat          0.5926635  0.5890475 -0.5335999 -0.5415672 -0.2966488
## biomass_above     1.0000000  0.7600965 -0.6732280 -0.5668411 -0.3750552
## plant_dens        0.7600965  1.0000000 -0.6076623 -0.3558787 -0.3951235
## leaf_N           -0.6732280 -0.6076623  1.0000000  0.5349996  0.3001950
## leaf_C           -0.5668411 -0.3558787  0.5349996  1.0000000  0.1715634
## leaf_P           -0.3750552 -0.3951235  0.3001950  0.1715634  1.0000000
```

```
read_csv("GROUP_B_MASTER_DATAFRAME.csv") -> data
```

```
## New names:
## Rows: 75 Columns: 15
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (2): Site, Community dbl (13): ...1, Shannon, soil_C, soil_N, soil_P, soil_pH,
## soil_sal, soil_wat...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

PCA TIME!

```
pca.out <- prcomp(data[, c("soil_C", "soil_N", "soil_P", "soil_pH", "soil_sal",
                           "soil_wat", "biomass_above", "plant_dens", "leaf_N",
                           "leaf_C", "leaf_P")], scale.=T)
```

PC1: (1) soil_N (-0.3543157) (2) soil_wat (-0.3536549) (3) soil_pH (0.3370312) (4) biomass_above (0.3269548) (5) soil_C (-0.3267151)

PC2: (1) soil_C (0.44891398) (2) plant_dens (-0.40295140) (3) soil_sal (0.39164170) (4) soil_N (0.37709659)

```
pca.out
```

```
## Standard deviations (1, .., p=11):
##  [1] 2.4739400 1.0748490 0.9303355 0.8883592 0.8115063 0.7306543 0.5751151
##  [8] 0.4855427 0.4101910 0.3178748 0.2034850
##
## Rotation (n x k) = (11 x 11):
##                      PC1         PC2          PC3          PC4         PC5
## soil_C        -0.3270867  0.44789885  0.166535148 -0.008676304  0.03928796
## soil_N        -0.3547747  0.37618972  0.080336514 -0.012495465  0.03310391
## soil_P        -0.2202713  0.02402190 -0.102260684  0.868617978 -0.28746538
## soil_pH        0.3376054 -0.22697065 -0.188488951  0.129101849 -0.05900525
## soil_sal      -0.2806422 -0.39256180 -0.001680892  0.031298086  0.57534475
## soil_wat      -0.3539508  0.23179513  0.023637151 -0.106935165  0.22817143
```

```
## biomass_above -0.3274486 -0.30076393 -0.166191853 -0.113060140 -0.05552193
## plant_dens    -0.3154603 -0.40412756  0.019900979  0.196156607  0.24912045
## leaf_N         0.3142092  0.06465172  0.204688059  0.042696425  0.39640405
## leaf_C         0.2491950  0.08525667  0.592994804  0.361212419  0.30039910
## leaf_P         0.1817902  0.36430053 -0.705485220  0.180643295  0.46436986
##                       PC6          PC7         PC8           PC9        PC10
## soil_C        -0.01522495  0.233006840  0.19895056  0.5003731605 -0.30717955
## soil_N         0.07140457  0.200468481  0.19590211  0.1164541941  0.32749935
## soil_P         0.24960073 -0.095511191 -0.17363341 -0.0004085302  0.01494356
## soil_pH        0.22688892  0.518724636  0.51089232  0.1140463425  0.38564887
## soil_sal       0.24372175 -0.422126461  0.12036129  0.3616190353  0.22135118
## soil_wat       0.25065950  0.094120337 -0.02413831 -0.6709013573  0.29839233
## biomass_above -0.26882044  0.445515232 -0.58930686  0.2239258302  0.28764553
## plant_dens    -0.23672949  0.340168319  0.29150445 -0.2968152659 -0.51071165
## leaf_N         0.54339602  0.353140394 -0.42365379  0.0539462712 -0.26428002
## leaf_C        -0.50049505  0.036650528 -0.03818856 -0.0315869034  0.31180818
## leaf_P        -0.29082673 -0.004844742 -0.05112435  0.0056049776 -0.02770573
##                      PC11
## soil_C         0.47350843
## soil_N        -0.72140762
## soil_P         0.03895873
## soil_pH        0.18850329
## soil_sal       0.05734670
## soil_wat       0.38146521
## biomass_above  0.08899378
## plant_dens    -0.17485820
## leaf_N        -0.15495914
## leaf_C         0.08220256
## leaf_P        -0.01550201
```

```
summary(pca.out)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.4739 1.0748 0.93034 0.88836 0.81151 0.73065 0.57512
## Proportion of Variance 0.5564 0.1050 0.07868 0.07174 0.05987 0.04853 0.03007
## Cumulative Proportion  0.5564 0.6614 0.74011 0.81185 0.87172 0.92025 0.95032
##                           PC8    PC9    PC10    PC11
## Standard deviation     0.48554 0.4102 0.31787 0.20349
## Proportion of Variance 0.02143 0.0153 0.00919 0.00376
## Cumulative Proportion  0.97175 0.9870 0.99624 1.00000
```

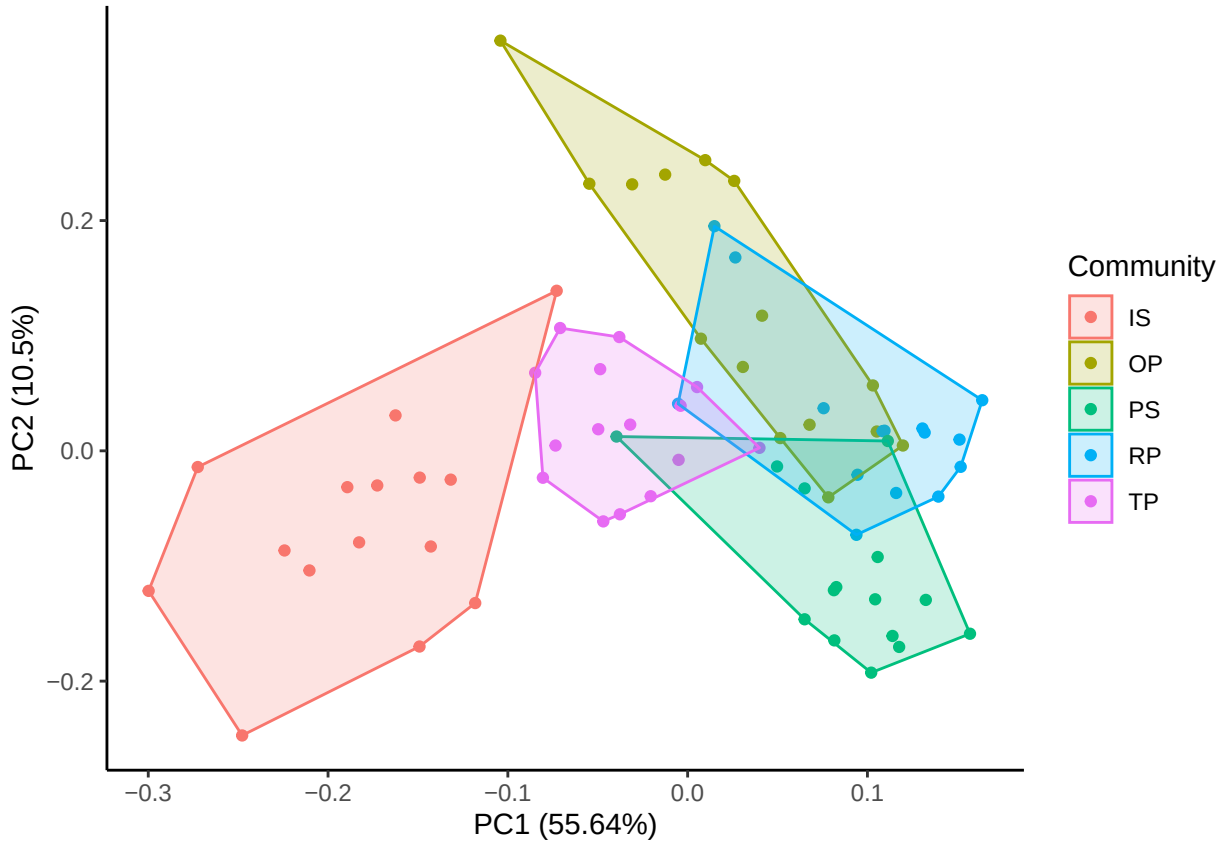
```
PC1 <- pca.out$x[, 1]
PC2 <- pca.out$x[, 2]
```

```
data_with_PCs <- data
data_with_PCs$PC1 <- PC1
data_with_PCs$PC2 <- PC2
```

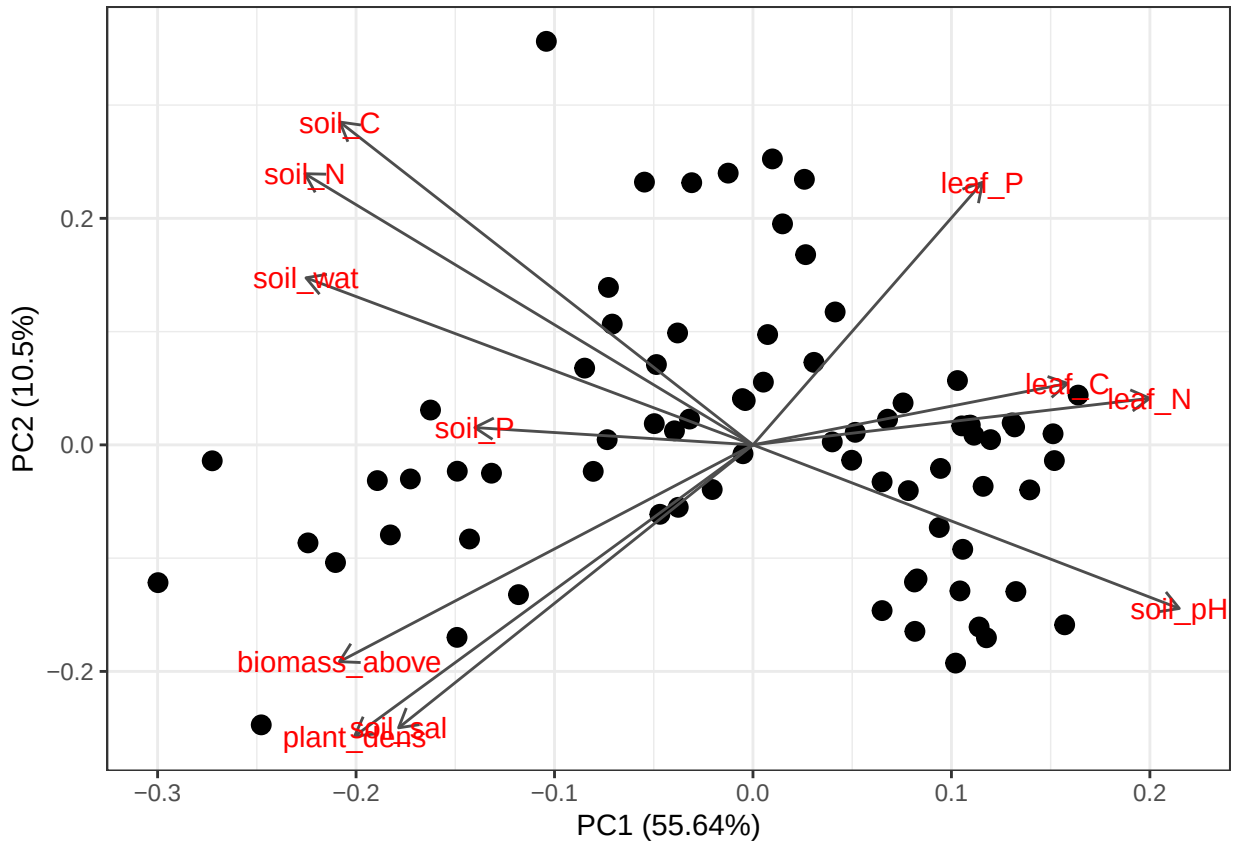
```
autoplot(pca.out, x=1,y=2,
         data=data,
         colour="Community",
```

```
        frame=TRUE) +
  theme_classic()
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## * `...1` -> `...3`
```




```
autoplot(pca.out, x=1,y=2,loadings=TRUE, loadings.label=TRUE,
         loadings.colour="grey30",
          size=3) +
  theme_bw()
```




```
model1 <- lmer(Shannon ~ PC1 + (1 | Community), data = data_with_PCs)
model2 <- lmer(Shannon ~ PC1 * PC2 + (1 | Community), data = data_with_PCs)
model3 <- lm(Shannon ~ PC1 * PC2, data = data_with_PCs)
model4 <- lm(Shannon ~ PC1, data = data_with_PCs)
model5 <- lm(Shannon ~ PC2, data = data_with_PCs)
model6 <- lmer(Shannon ~ PC2 + (1 | Community), data = data_with_PCs)
```

```
AIC(model1)
```

```
## [1] 23.03914
```

```
AIC(model2)
```

```
## [1] 36.55846
```

```
AIC(model3)
```

```
## [1] 29.98565
```

```
AIC(model4)
```

```
## [1] 33.16352
```

```
AIC(model5)
```

```
## [1] 41.92991
```

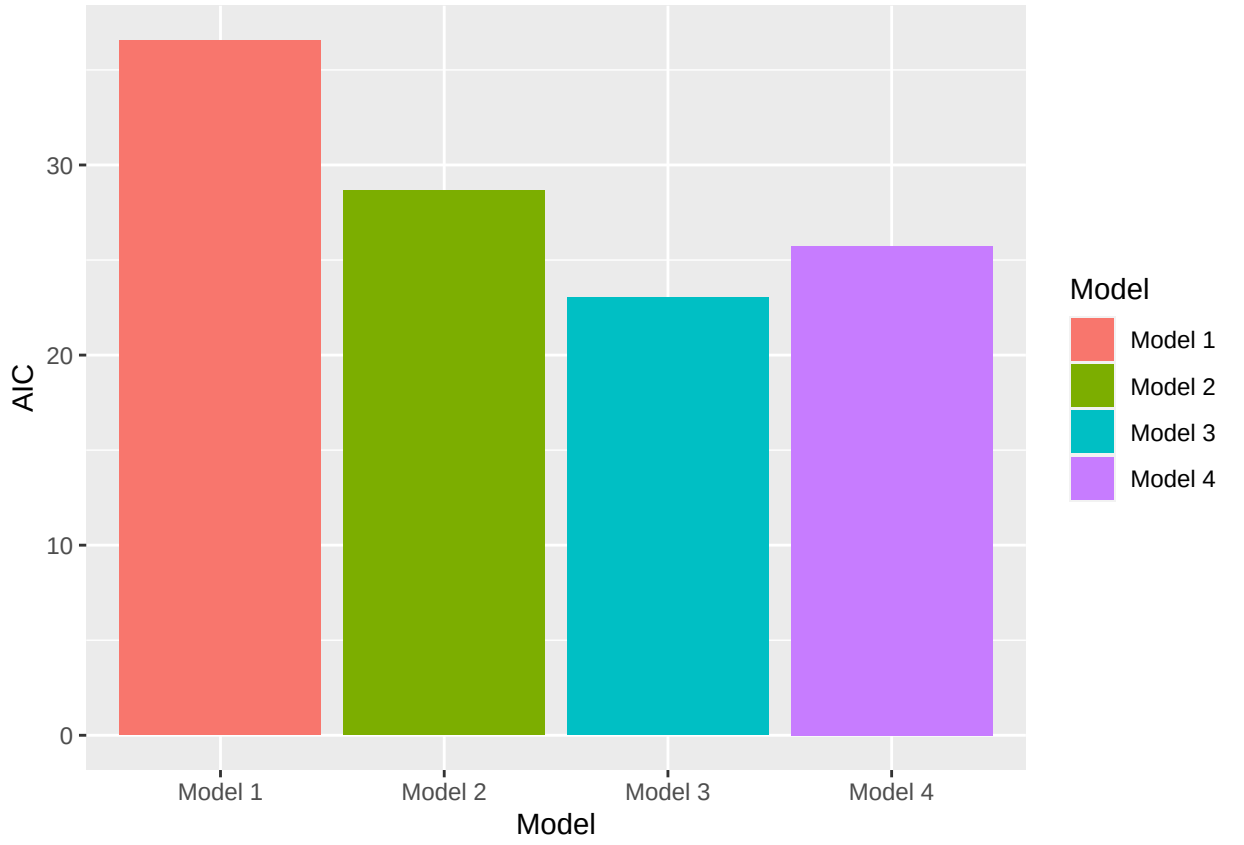
```
AIC(model6)
```

```
## [1] 25.74666
```

```
BarData <- read.csv("BarData.csv")
```

```
## Warning in read.table(file = file, header = header, sep = sep, quote = quote, :
## incomplete final line found by readTableHeader on 'BarData.csv'
```

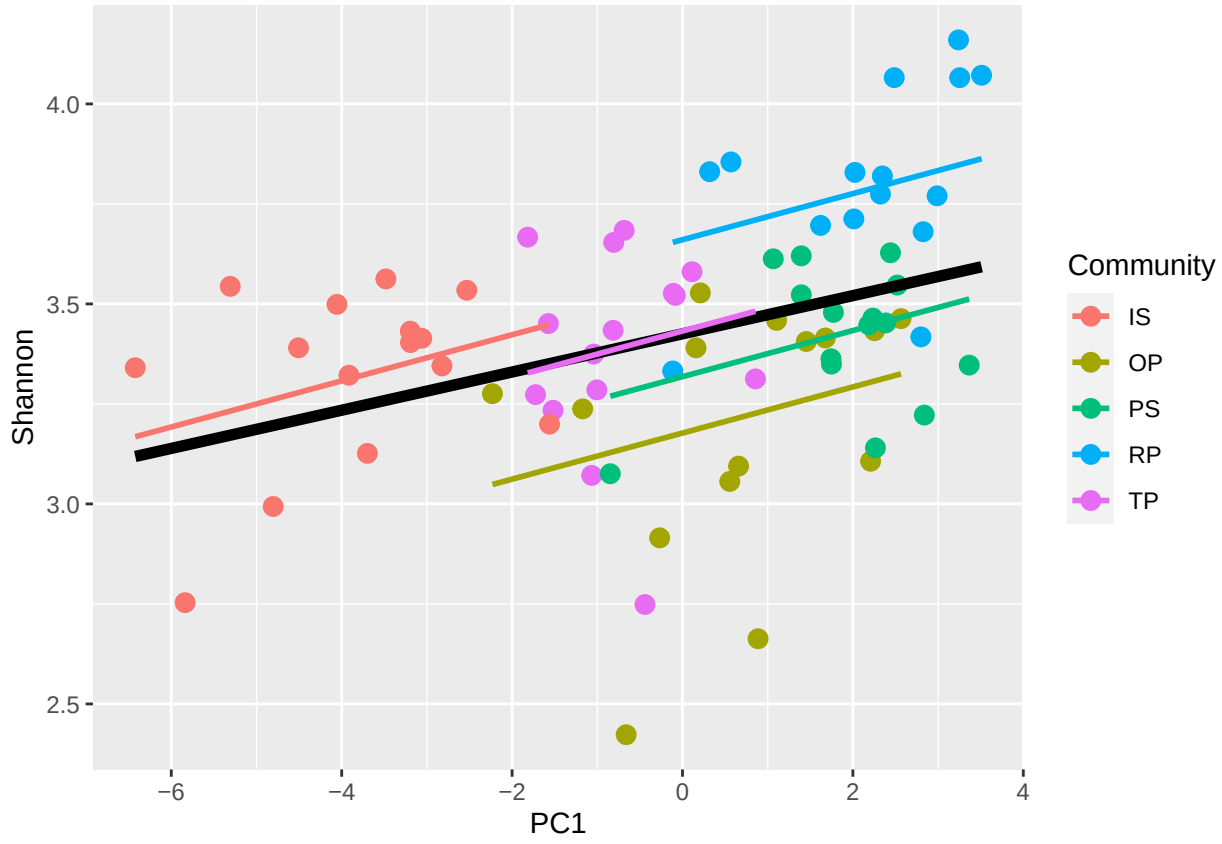
```
ggplot(data=BarData, aes(x=Model, y=AIC, fill=Model)) +
  geom_bar(stat="identity")
```



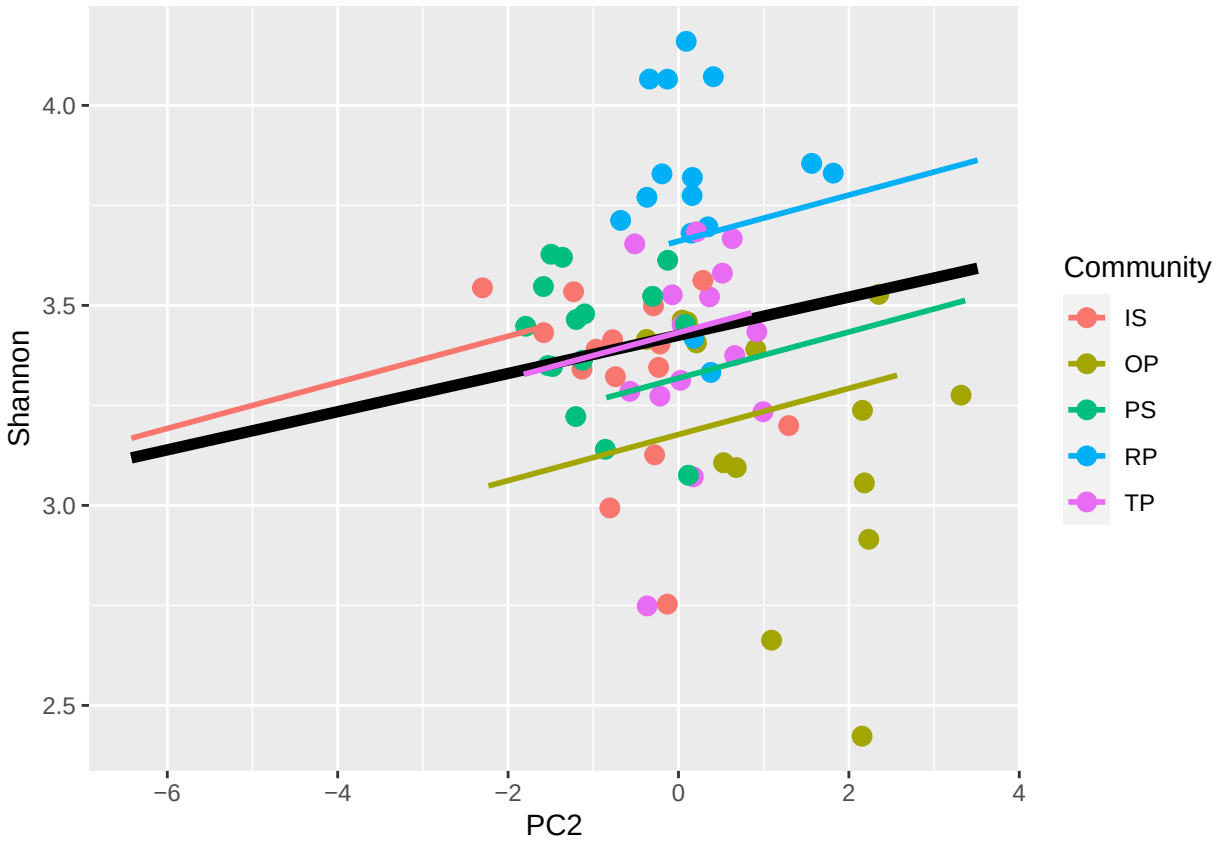

```
data_with_PCs  |>
  mutate(fit.m = predict(model4, re.form = NA), # does not include random effects
         fit.c = predict(model1, re.form = NULL) # includes random effects
         ) ->
  predicted_values

predicted_values |>
```

```
  ggplot(aes(x = PC1, y = Shannon, color = Community)) +
  geom_point(size = 3) +
  geom_line(inherit.aes = F, aes(x = PC1, y = fit.m), color = "black", size = 2) +
  geom_line(inherit.aes = F, aes(x = PC1, y = fit.c, color = Community), size = 1)
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```




```
predicted_values |>
  ggplot(aes(x = PC2, y = Shannon, color = Community)) +
  geom_point(size = 3) +
  geom_line(inherit.aes = F, aes(x = PC1, y = fit.m), color = "black", size = 2) +
  geom_line(inherit.aes = F, aes(x = PC1, y = fit.c, color = Community), size = 1)
```


Shannon
4.0
3.5
3.0
2.5
−6
−4
−2
0
2
4
PC2
Community
IS
OP
PS
RP
TP