oneFill_Bottleneck

December 20, 2017

0.1 Effects of migration and bottleneck on the strength and formation of cyanogenesis clines

In this notebook, I look at the effects of migration rate and population bottlenecks on the strength and proportion of significant positive and negative clines in cyanogenesis.

Positive clines Less HCN in urban environment.

Negative clines More HCN in urban environment.

```
In [1]: library(plyr)
        library(ggplot2)
        library(data.table, lib="~/Rpackages")
        library(Rmisc, lib = "~/Rpackages")
        library(dplyr)
        library(broom)
```

```
Loading required package: lattice

Attaching package: dplyr

The following objects are masked from package:data.table:

    between, first, last

The following objects are masked from package:plyr:

    arrange, count, desc, failwith, id, mutate, rename, summarise,
    summarize

The following objects are masked from package:stats:

    filter, lag

The following objects are masked from package:base:

    intersect, setdiff, setequal, union
```

```
In [82]: #Working directory for datasets varying migration rate and bottleneck proportion
         setwd('/scratch/research/projects/trifolium/SEC_Simulation.Evolutionary.Clines/SEC_Data
         /drift-migration/1D/Mig_Bot_Vary')

         #Load datasets that will be used for analyses
         datSlopes <- fread('20171106_SlopeSum_Gen_BotMig-Merged_distRev.csv', header = T)
         datFreqFirst <- fread('20171106_FreqFirstGen_BotMig-Merged_distRev.csv', header = T)
         datSlopes$bot <- as.factor(as.character(datSlopes$bot))
         datSlopes$Mig_rate <- as.factor(as.character(datSlopes$Mig_rate))
```

```
datFreqFirst$bot <- as.factor(as.character(datFreqFirst$bot))
datFreqFirst$Mig_rate <- as.factor(as.character(datFreqFirst$Mig_rate))

#Data subsets
datSlopes_GenOne <- subset(datSlopes, seq == "1")
datSlopes_NoMig <- subset(datSlopes, Mig_rate == "0")
datSlopes_GenOne_NoMig <- subset(datSlopes, seq == "1" & Mig_rate == "0")
datFreqFirst_StrongBot_NoMig <- subset(datFreqFirst, bot == "0.01" & Mig_rate == "0")
datFreqFirst_InterBot_NoMig <- subset(datFreqFirst, bot == "0.2" & Mig_rate == "0")
datFreqFirst_NoBot_NoMig <- subset(datFreqFirst, bot == "1" & Mig_rate == "0")


#Proportion of simulations with Cyan lost by distance under strong bottlenecks
datPropLost_StrongBot_NoMig <- datFreqFirst_StrongBot_NoMig %>%
    group_by(Distance) %>%
    summarize(n = n(),
             Lost = sum(Cyan == 0) / n,
             Fixed = sum(Cyan == 1) / n) %>%
    mutate(Founder = "Strong")
datPropLost_StrongBot_NoMig <- dplyr::select(datPropLost_StrongBot_NoMig, Distance,
                                          Lost, Founder)


#Proportion of simulations with Cyan lost by distance under intermediate bottlenecks
datPropLost_InterBot_NoMig <- datFreqFirst_InterBot_NoMig %>%
    group_by(Distance) %>%
    summarize(n = n(),
             Lost = sum(Cyan == 0) / n,
             Fixed = sum(Cyan == 1) / n) %>%
    mutate(Founder = "Intermediate")
datPropLost_InterBot_NoMig <- dplyr::select(datPropLost_InterBot_NoMig, Distance,
                                          Lost, Founder)

#Proportion of simulations with Cyan lost by distance under no bottlenecks
datPropLost_NoBot_NoMig <- datFreqFirst_NoBot_NoMig %>%
    group_by(Distance) %>%
    summarize(n = n(),
             Lost = sum(Cyan == 0) / n,
             Fixed = sum(Cyan == 1) / n) %>%
    mutate(Founder = "None")
datPropLost_NoBot_NoMig <- dplyr::select(datPropLost_NoBot_NoMig, Distance,
                                          Lost, Founder)

# Merge datasets from strong and intermediate bottlenecks above
datPropLost_merged <- rbind(datPropLost_StrongBot_NoMig, datPropLost_InterBot_NoMig,
datPropLost_NoBot_NoMig)

# Calculate mean frequency in first generation across simulations for each distance
MeanFreqFirstGen_Distance <- datFreqFirst %>%
    group_by(Distance, bot, Mig_rate) %>%
    summarize(Freq = mean(Cyan))

MeanFreqFirstGen_Distance_StrongBot_NoMig <- subset(MeanFreqFirstGen_Distance, bot ==
"0.01" & Mig_rate == "0")
MeanFreqFirstGen_Distance_InterBot_NoMig <- subset(MeanFreqFirstGen_Distance, bot ==
"0.2" & Mig_rate == "0")
MeanFreqFirstGen_Distance_NoBot_NoMig <- subset(MeanFreqFirstGen_Distance, bot == "1" &
Mig_rate == "0")
```

```
Read 1200000 rows and 12 (of 12) columns from 0.046 GB file in 00:00:03
```

```
In [16]: str(datSlopes)
```

```
Classes data.table and 'data.frame':  15000 obs. of  20 variables:
 $ bot         : Factor w/ 10 levels "0.01","0.025",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ Mig_rate    : Factor w/ 3 levels "0","0.01","0.05": 1 1 1 1 1 1 1 1 1 1 ...
 $ seq        : int  1 2 3 4 5 6 7 8 9 10 ...
 $ mean       : num  0.00151 0.0015 0.0015 0.0015 0.0015 ...
 $ sd         : num  0.00381 0.00381 0.00381 0.0038 0.0038 ...
 $ n          : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
 $ se         : num  0.000121 0.00012 0.00012 0.00012 0.00012 ...
 $ ci_mean    : num  0.000236 0.000236 0.000236 0.000236 0.000236 ...
 $ prop_sigPos: num  0.421 0.422 0.42 0.422 0.42 0.42 0.415 0.418 0.419 0.416 ...
 $ prop_pos   : num  0.74 0.74 0.739 0.739 0.739 0.739 0.739 0.739 0.739 0.739 ...
 $ se_pos     : num  0.0139 0.0139 0.0139 0.0139 0.0139 ...
 $ ci_pos     : num  0.0272 0.0272 0.0272 0.0272 0.0272 ...
 $ se_sigPos  : num  0.0156 0.0156 0.0156 0.0156 0.0156 ...
 $ ci_sigPos  : num  0.0306 0.0306 0.0306 0.0306 0.0306 ...
 $ prop_sigNeg: num  0.176 0.177 0.176 0.175 0.175 0.177 0.173 0.17 0.175 0.174 ...
 $ prop_neg   : num  0.241 0.241 0.241 0.241 0.241 0.241 0.241 0.241 0.241 0.241 ...
 $ se_neg     : num  0.0135 0.0135 0.0135 0.0135 0.0135 ...
 $ ci_neg     : num  0.0265 0.0265 0.0265 0.0265 0.0265 ...
 $ se_sigNeg  : num  0.012 0.0121 0.012 0.012 0.012 ...
 $ ci_sigNeg  : num  0.0236 0.0237 0.0236 0.0236 0.0236 ...
 - attr(*, ".internal.selfref")=<externalptr>
```

```
In [4]: str(datFreqFirst)
```

```
Classes data.table and 'data.frame':  1200000 obs. of  12 variables:
 $ x         : int  0 0 0 0 0 0 0 0 0 0 ...
 $ y         : int  0 0 0 0 0 0 0 0 0 0 ...
 $ bot       : num  0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 ...
 $ Sim       : int  0 0 0 1 1 1 2 2 2 3 ...
 $ Generation: int  1 1 1 1 1 1 1 1 1 1 ...
 $ Cyan      : num  0.563 0.563 0.563 0.563 0.563 0.563 0.563 0.563 0.563 0.563 ...
 $ Mat_full  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Pop_size  : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
 $ Mig_rate  : num  0 0.01 0.05 0 0.01 0.05 0 0.01 0.05 0 ...
 $ pA        : num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
 $ pB        : num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
 $ Distance  : int  40 40 40 40 40 40 40 40 40 40 ...
 - attr(*, ".internal.selfref")=<externalptr>
```

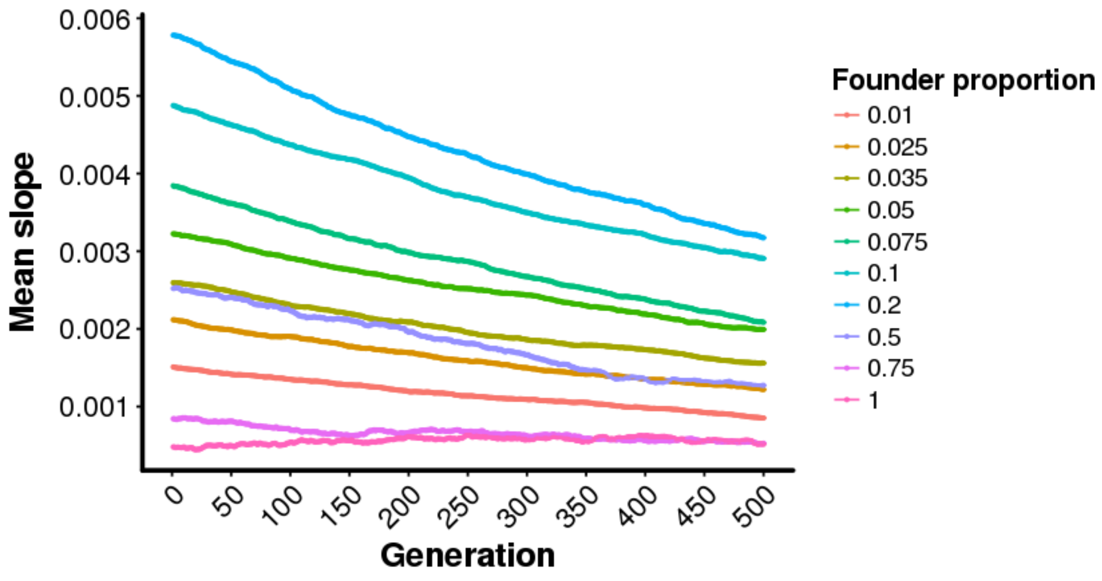
```
In [5]: #Themes used for plotting
        ng1=theme(aspect.ratio=0.7,panel.background = element_blank(),
                  panel.grid.major = element_blank(),
                  panel.grid.minor = element_blank(),
                  panel.border=element_blank(),
                  axis.line.x = element_line(color="black",size=1),
                  axis.line.y = element_line(color="black",size=1),
                  axis.ticks=element_line(color="black"),
                  axis.text=element_text(color="black",size=15),
                  axis.title=element_text(color="black",size=1),
                  axis.title.y=element_text(vjust=2,face="bold",size=15),
                  axis.title.x=element_text(vjust=0.1,face="bold",size=15),
                  axis.text.x=element_text(size=13),
                  axis.text.y=element_text(size=13),
                  legend.position = "right", legend.direction="vertical",
                  legend.text=element_text(size=11), legend.key = element_rect(fill = "white"),
                  legend.title = element_text(size=13,face="bold"),legend.key.size = unit(0.5,
        "cm"))

        ng1.45=theme(aspect.ratio=0.7,panel.background = element_blank(),
                  panel.grid.major = element_blank(),
                  panel.grid.minor = element_blank(),
                  panel.border=element_blank(),
                  axis.line.x = element_line(color="black",size=1),
                  axis.line.y = element_line(color="black",size=1),
                  axis.ticks=element_line(color="black"),
                  axis.text=element_text(color="black",size=15),
                  axis.title=element_text(color="black",size=1),
                  axis.title.y=element_text(vjust=2,face="bold",size=15),
                  axis.title.x=element_text(vjust=0.1,face="bold",size=15),
                  axis.text.x=element_text(size=13,angle=45,hjust=1),
                  axis.text.y=element_text(size=13),
                  legend.position = "right", legend.direction="vertical",
                  legend.text=element_text(size=11), legend.key = element_rect(fill = "white"),
                  legend.title = element_text(size=13,face="bold"),legend.key.size = unit(0.5,
        "cm"))
```

```
In [10]: path = "/scratch/research/projects/trifolium/SEC_Simulation.Evolutionary.Clines/SEC_Sync
         /SEC_Figures/Drift.Migration/Mig_Bot_Vary"

         # ggsave("Mean-slope_BotMig.pdf", plot = MeanSlope_BotMig, device = "pdf", width = 6.0,
         height = 6.0, path = path, dpi = 600)
         # ggsave("PropSigPos_BotMig.pdf", plot = PropSigPos_BotMig, device = "pdf", width = 6.0,
         height = 6.0, path = path, dpi = 600)
         # ggsave("propLost_bot_NoMig.pdf", plot = propLost_bot_NoMig, device = "pdf", width =
         6.0, height = 6.0, path = path, dpi = 600)
         # ggsave("Regression_LostAtGen2_Bot001_NpMig.pdf", plot = LostAt2_bot001_NoMig, device =
         "pdf", width = 6.0, height = 6.0, path = path, dpi = 600)
         # ggsave("Regression_NotLost_Bot02_NoMig.pdf", plot = NotLost_bot02_NoMig, device =
         "pdf", width = 6.0, height = 6.0, path = path, dpi = 600)
         # ggsave("PropNeg_BotMig.pdf", plot = PropNeg_BotMig, device = "pdf", width = 6.0,
         height = 6.0, path = path, dpi = 600)
         ggsave("Mean-slope_Bot_HighMig.pdf", plot = MeanSlope_Bot_HighMig, device = "pdf", width
         = 6.0, height = 6.0, path = path, dpi = 600)
         ggsave("PropSigPos_Bot_HighMig.pdf", plot = PropPos_Bot_HighMig, device = "pdf", width =
         6.0, height = 6.0, path = path, dpi = 600)
```

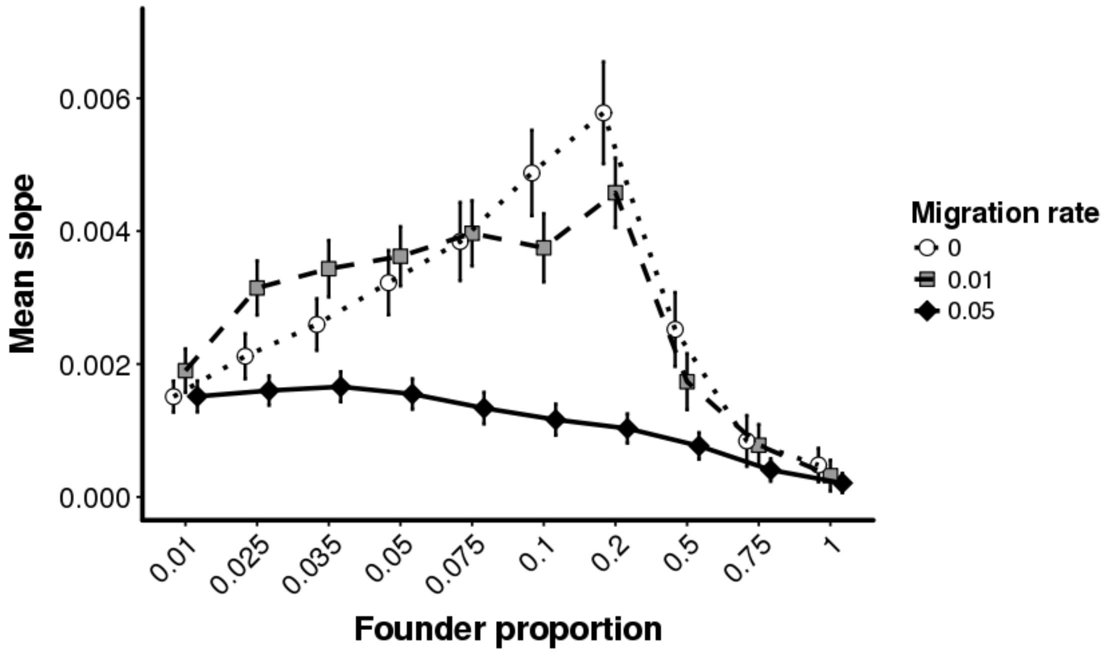
0.2 Mean slope with founder proportion under varying migration rates

Here I look at the mean slope across simulations under varying founding proportions and migration rates. Mean slope is calculated in the first generation following the filling of the landscape matrix. I also plot the change in the mean slope with increasing generations.

```
In [18]: MeanSlope_Cyan_byGen <- ggplot(datSlopes_NoMig, aes(x = seq, y = mean, group = bot,
         color = bot)) +
             ylab("Mean slope") + xlab("Generation") + geom_point(size = 0.5, aes(color = bot)) +
             geom_line(size = 0.5, aes(color = bot)) +
             coord_cartesian(xlim = c(0, 500)) + scale_x_continuous(breaks = seq(from = 0, to =
         500, by = 50)) +
             labs(color = "Founder proportion") + ng1.45
         MeanSlope_Cyan_byGen
```



```
In [9]: MeanSlope_BotMig <- ggplot(datSlopes_GenOne, aes(x = bot, y = mean, group = Mig_rate)) +
            geom_errorbar(aes(ymin = mean - ci_mean, ymax = mean + ci_mean), width=0.15,
        size=0.7,
            position = position_dodge(width = 0.5)) +
            geom_point(size = 3, aes(fill = Mig_rate, shape = Mig_rate), position =
        position_dodge(width = 0.5)) +
            geom_line(size = 1, aes(linetype = Mig_rate), position = position_dodge(width =
        0.5)) +
            scale_shape_manual(labels = c("0", "0.01", "0.05"), values = c(21, 22, 23)) +
            scale_fill_manual(labels = c("0", "0.01", "0.05"), values = c("white", "grey60",
        "black")) +
            scale_linetype_manual(labels = c("0", "0.01", "0.05"), values = c("dotted",
        "dashed", "solid")) +
            coord_cartesian(ylim = c(0, 0.007)) + scale_y_continuous(breaks = seq(from = 0, to =
        0.006, by = 0.002)) +
            ylab("Mean slope") + xlab("Founder proportion") +
            labs(fill = "Migration rate", shape = "Migration rate", linetype = "Migration rate")
        + ng1.45
        MeanSlope_BotMig
```

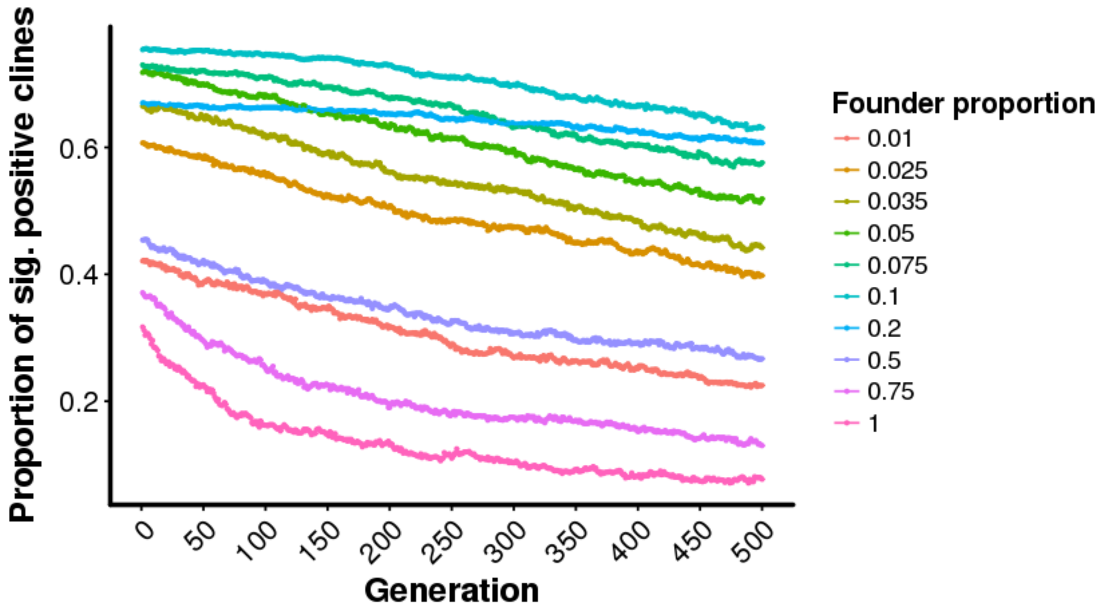


0.3 Effects of founder events and migration on proportion of negative and positive clines

Here I look at how founder effects and migration affect the proportion of significantly negative and positive clines in the *Ac* and *Li* alleles and in HCN clines.

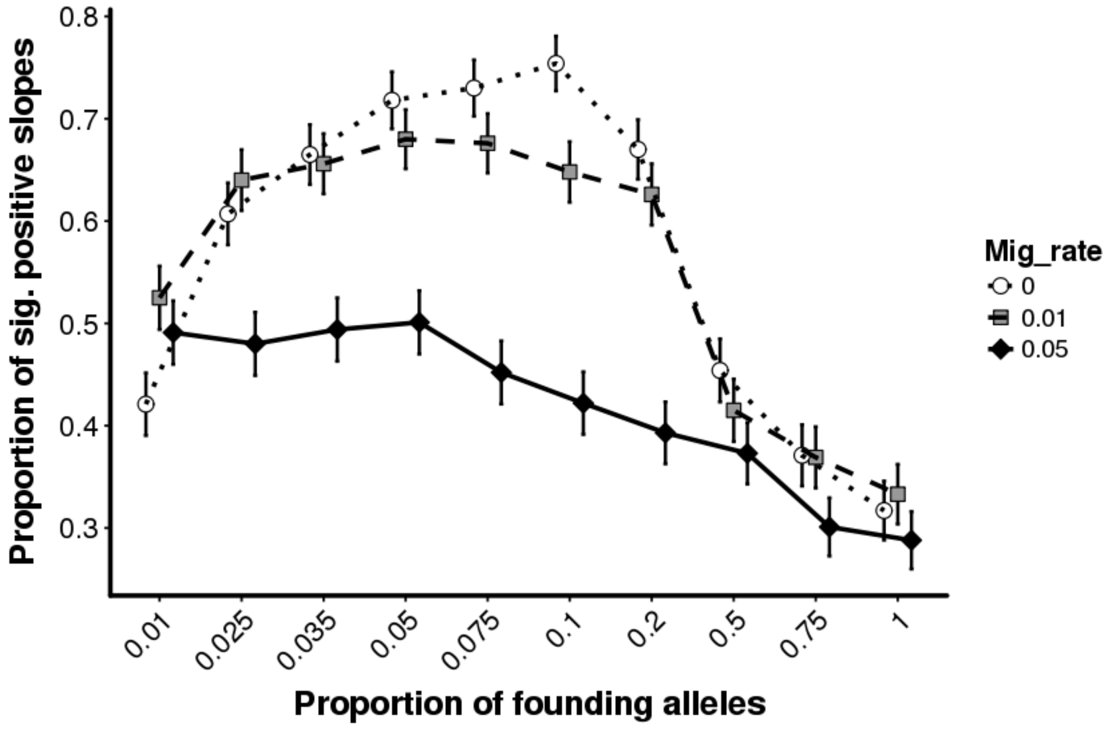
Positive clines: Less HCN in urban environment (i.e. loss of HCN across space)
Negative clines: More HCN in urban environment (i.e. gain in HCN across space)

```
In [21]: PropSigPos_Cyan_byGen <- ggplot(datSlopes_NoMig, aes(x = seq, y = prop_sigPos, group =
         bot, color = bot)) +
             ylab("Proportion of sig. positive clines") + xlab("Generation") + geom_point(size =
         0.5, aes(color = bot)) +
             geom_line(size = 0.5, aes(color = bot)) +
             coord_cartesian(xlim = c(0, 500)) + scale_x_continuous(breaks = seq(from = 0, to =
         500, by = 50)) +
             labs(color = "Founder proportion") + ng1.45
         PropSigPos_Cyan_byGen
```



```
In [91]: PropSigPos_BotMig <- ggplot(datSlopes_GenOne, aes(x = bot, y = prop_sigPos, group =
         Mig_rate)) +
             geom_errorbar(aes(ymin = prop_sigPos - ci_sigPos, ymax = prop_sigPos + ci_sigPos),
         width=0.15, size=0.7,
             position = position_dodge(width = 0.5)) +
             geom_point(size = 3, aes(fill = Mig_rate, shape = Mig_rate), position =
         position_dodge(width = 0.5)) +
             geom_line(size = 1, aes(linetype = Mig_rate), position = position_dodge(width =
         0.5)) +
             scale_shape_manual(labels = c("0", "0.01", "0.05"), values = c(21, 22, 23)) +
             scale_fill_manual(labels = c("0", "0.01", "0.05"), values = c("white", "grey60",
         "black")) +
             scale_linetype_manual(labels = c("0", "0.01", "0.05"), values = c("dotted",
         "dashed", "solid")) +
             ylab("Proportion of sig. positive slopes") + xlab("Proportion of founding alleles")
```

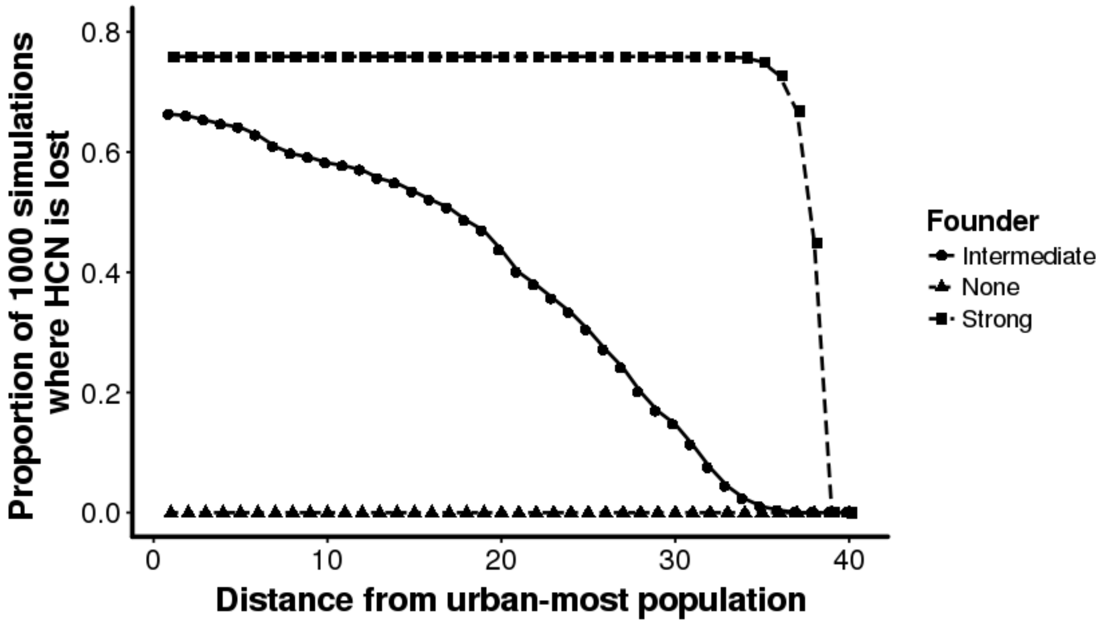
```
+ ng1.45
PropSigPos_BotMig
```



0.4 Dynamics of HCN frequency change during colonization

Here I try to understand the hump-shaped distribution above by exploring how the frequency of HCN changes during the colonization process.

```
In [68]: propLost_bot_NoMig <- ggplot(datPropLost_merged, aes(x = Distance, y = Lost, group =
         Founder)) +
             geom_point(size = 2, aes(fill = Founder, shape = Founder), position =
         position_dodge(width = 0.5)) +
             geom_line(size = 0.75, aes(linetype = Founder)) +
             ylab("Proportion of 1000 simulations
         where HCN is lost") + xlab("Distance from urban-most population") +
             coord_cartesian(ylim = c(0, 0.8)) + scale_y_continuous(breaks = seq(from = 0, to =
         0.8, by = 0.2))  + ng1
         propLost_bot_NoMig
```

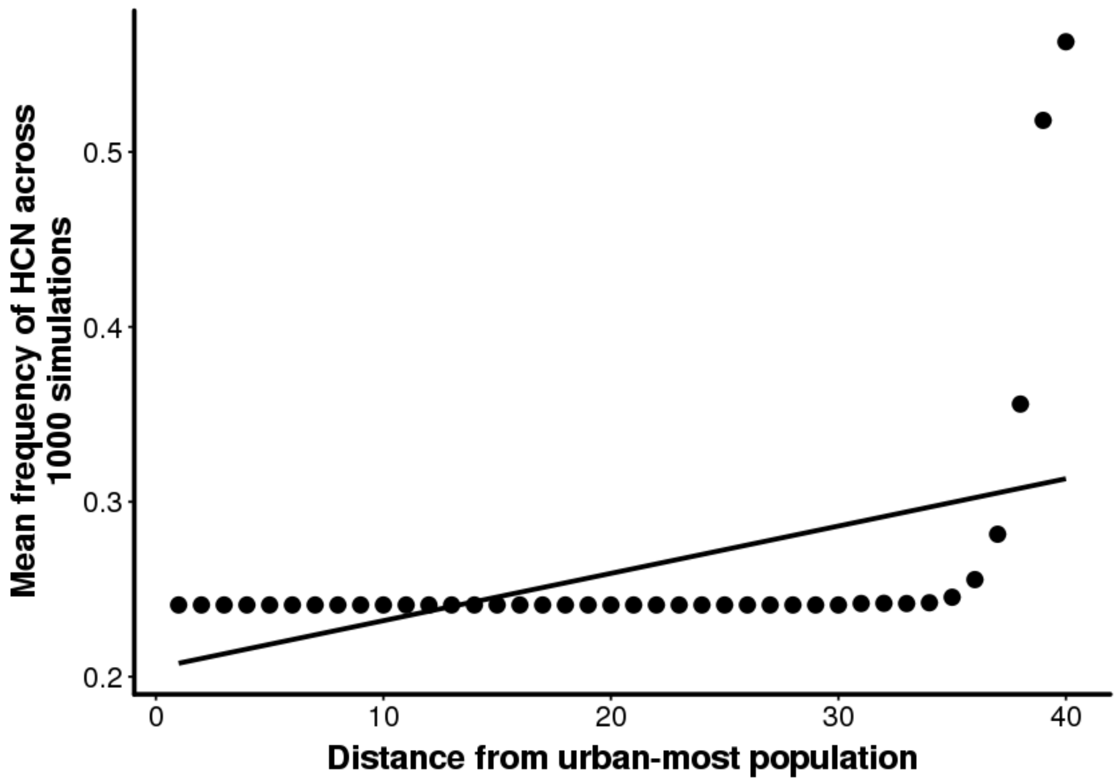


```
In [72]: lm_FreqFirstGen_StrongBot_NoMig <- lm(Freq ~ Distance, data =
         MeanFreqFirstGen_Distance_StrongBot_NoMig)
         summary(lm_FreqFirstGen_StrongBot_NoMig)
```

```
Call:
lm(formula = Freq ~ Distance, data = MeanFreqFirstGen_Distance_StrongBot_NoMig)

Residuals:
     Min       1Q   Median       3Q      Max
-0.05468 -0.03498 -0.01131  0.01506  0.24982
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.2049757  0.0196579  10.427 1.05e-12 ***
Distance    0.0027050  0.0008356   3.237   0.0025 **
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.061 on 38 degrees of freedom
Multiple R-squared:  0.2162,Adjusted R-squared:  0.1956
F-statistic: 10.48 on 1 and 38 DF,  p-value: 0.002504
```

```
In [74]: FreqFirstGen_StrongBot_NoMig <- ggplot(MeanFreqFirstGen_Distance_StrongBot_NoMig, aes(x
         = Distance, y = Freq)) +
             geom_point(size = 3) +
             geom_smooth(method = "lm", se = F, colour = "black", size = 1.05) +
             ylab("Mean frequency of HCN across
         1000 simulations") + xlab("Distance from urban-most population") +
             scale_y_continuous(breaks = seq(from = 0, to = 0.5, by = 0.1)) + ng1
         FreqFirstGen_StrongBot_NoMig
```

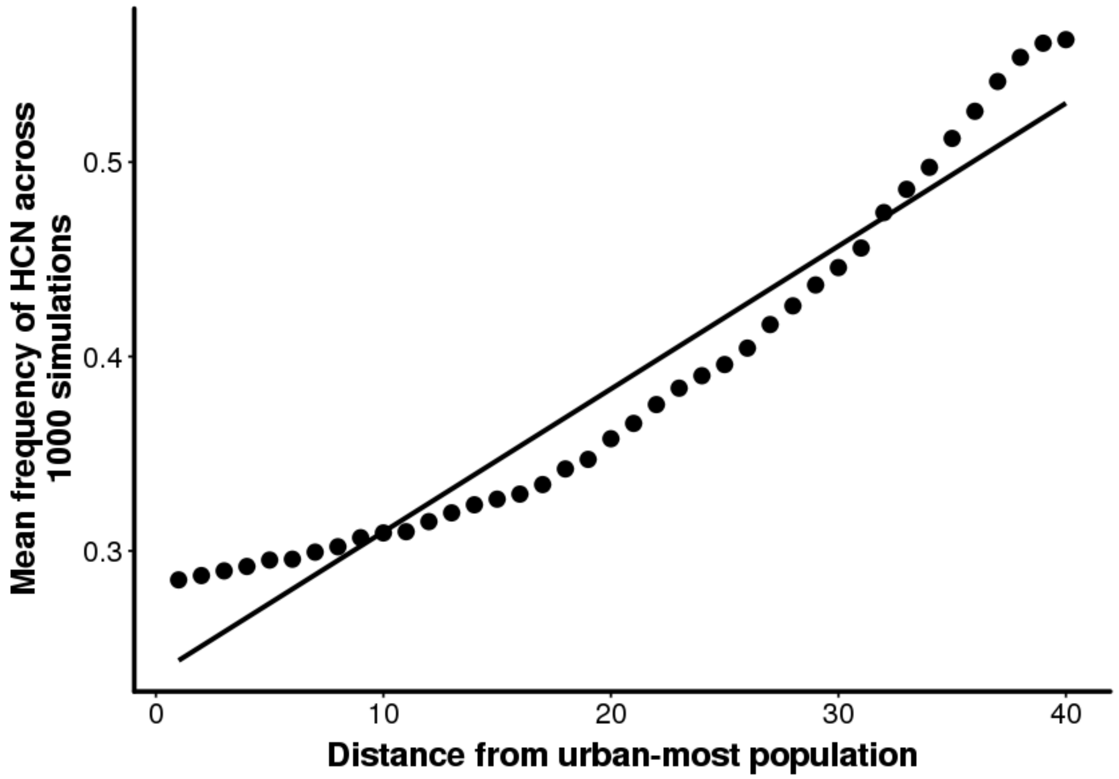


```
In [76]: lm_FreqFirstGen_InterBot_NoMig <- lm(Freq ~ Distance, data =
         MeanFreqFirstGen_Distance_InterBot_NoMig)
         summary(lm_FreqFirstGen_InterBot_NoMig)
```

```
Call:
lm(formula = Freq ~ Distance, data = MeanFreqFirstGen_Distance_InterBot_NoMig)

Residuals:
      Min        1Q    Median        3Q       Max
-0.028870 -0.021871 -0.007804  0.019546  0.041450

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept) 0.2362970  0.0074957   31.52   <2e-16 ***
Distance    0.0073500  0.0003186   23.07   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.02326 on 38 degrees of freedom
Multiple R-squared:  0.9334,Adjusted R-squared:  0.9316
F-statistic: 532.2 on 1 and 38 DF,  p-value: < 2.2e-16
```

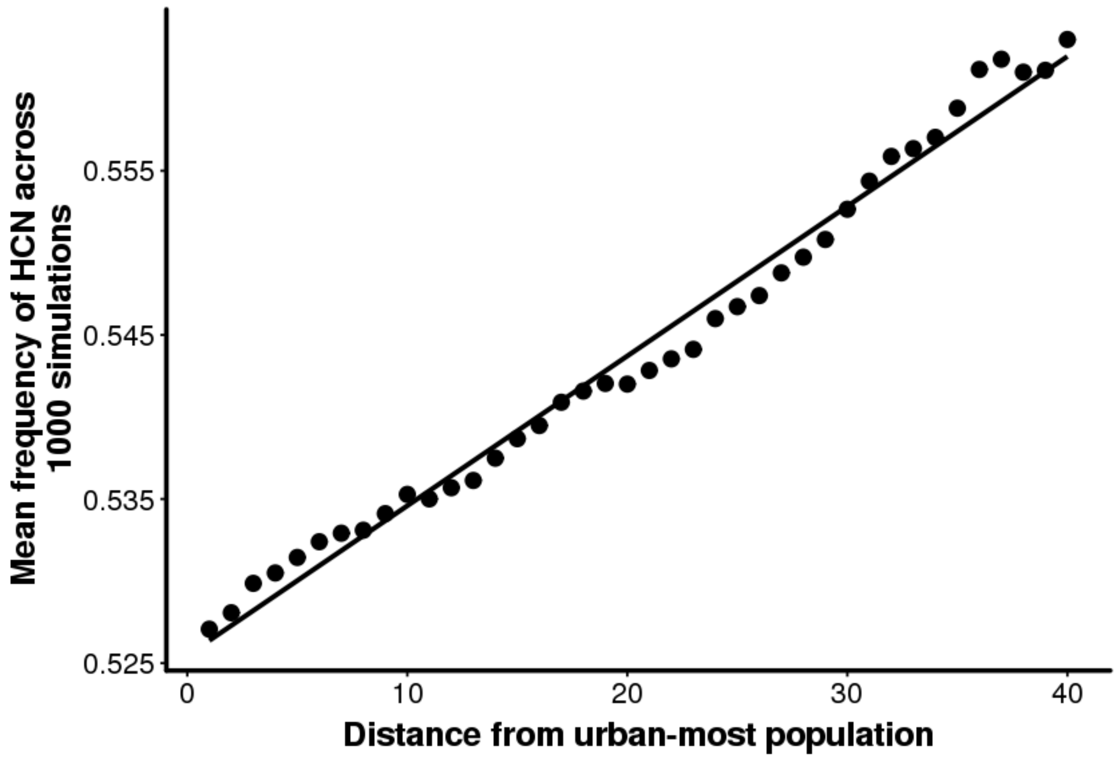
```
In [77]: FreqFirstGen_InterBot_NoMig <- ggplot(MeanFreqFirstGen_Distance_InterBot_NoMig, aes(x =
         Distance, y = Freq)) +
             geom_point(size = 3) +
             geom_smooth(method = "lm", se = F, colour = "black", size = 1.05) +
             ylab("Mean frequency of HCN across
         1000 simulations") + xlab("Distance from urban-most population") + ng1
         FreqFirstGen_InterBot_NoMig + ng1
```



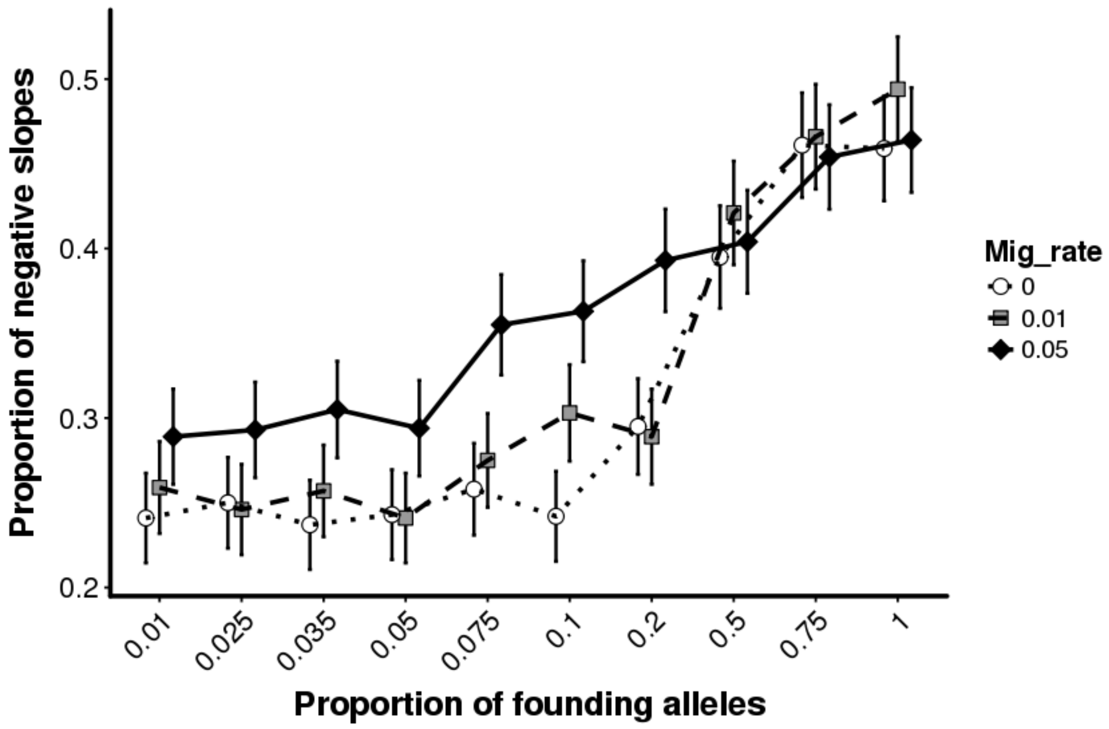
```
In [78]: lm_FreqFirstGen_NoBot_NoMig <- lm(Freq ~ Distance, data =
         MeanFreqFirstGen_Distance_NoBot_NoMig)
         summary(lm_FreqFirstGen_NoBot_NoMig)
```

```
Call:
lm(formula = Freq ~ Distance, data = MeanFreqFirstGen_Distance_NoBot_NoMig)

Residuals:
       Min         1Q     Median         3Q        Max
-2.333e-03 -1.134e-03 -6.130e-06  9.098e-04  2.859e-03

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.5254653  0.0004212 1247.42   <2e-16 ***
Distance    0.0009125  0.0000179   50.96   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.001307 on 38 degrees of freedom
Multiple R-squared:  0.9856,Adjusted R-squared:  0.9852
F-statistic:  2597 on 1 and 38 DF,  p-value: < 2.2e-16
```

```
In [79]: FreqFirstGen_NoBot_NoMig <- ggplot(MeanFreqFirstGen_Distance_NoBot_NoMig, aes(x =
         Distance, y = Freq)) +
             geom_point(size = 3) +
             geom_smooth(method = "lm", se = F, colour = "black", size = 1.05) +
             ylab("Mean frequency of HCN across
         1000 simulations") + xlab("Distance from urban-most population") + ng1
         FreqFirstGen_NoBot_NoMig + ng1
```



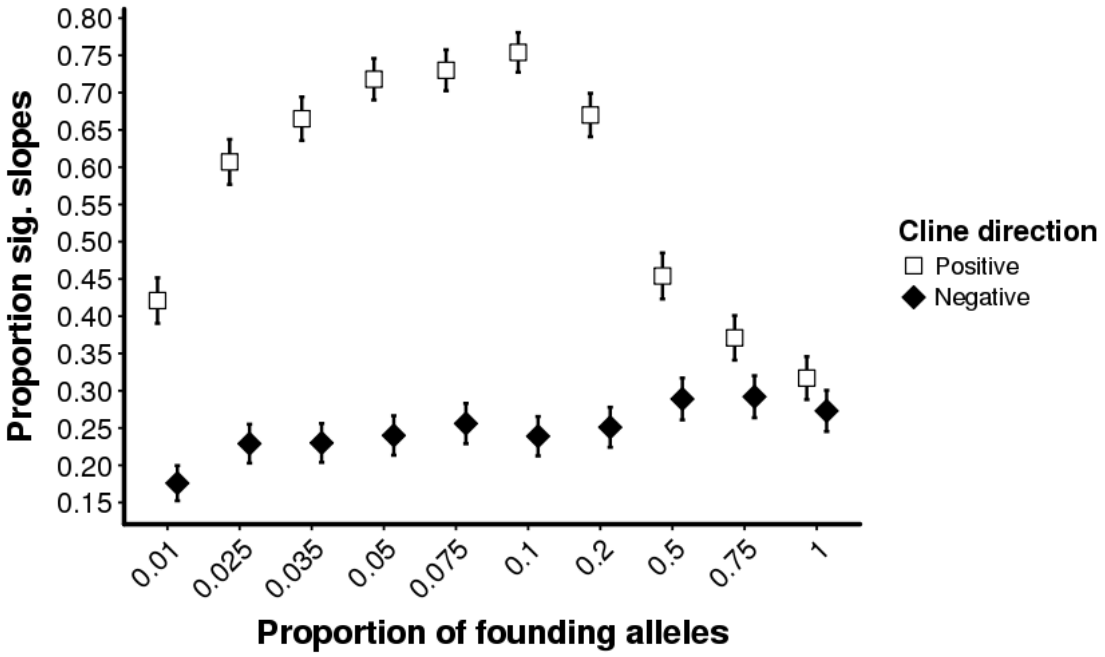
```
In [89]: PropNeg_BotMig <- ggplot(datSlopes_GenOne, aes(x = bot, y = prop_neg, group = Mig_rate))
         +
             geom_errorbar(aes(ymin = prop_neg - ci_neg, ymax = prop_neg + ci_neg), width=0.15,
         size=0.7,
             position = position_dodge(width = 0.5)) +
             geom_point(size = 3, aes(fill = Mig_rate, shape = Mig_rate), position =
         position_dodge(width = 0.5)) +
             geom_line(size = 1, aes(linetype = Mig_rate), position = position_dodge(width =
         0.5)) +
             scale_shape_manual(labels = c("0", "0.01", "0.05"), values = c(21, 22, 23)) +
             scale_fill_manual(labels = c("0", "0.01", "0.05"), values = c("white", "grey60",
         "black")) +
             scale_linetype_manual(labels = c("0", "0.01", "0.05"), values = c("dotted",
         "dashed", "solid")) +
             ylab("Proportion of negative slopes") + xlab("Proportion of founding alleles") +
         ng1.45
         PropNeg_BotMig + ng1.45
```



```
In [87]: #Extra columns that will be used to create melting dataset for plotting proportion of
         positive and negative slopes
         SlopeSum_Melt_NoMig <- dplyr::select(datSlopes_GenOne_NoMig, bot, prop_sigPos,
         ci_sigPos, prop_sigNeg,
           ci_sigNeg)

         #Melt dataframe
         dm1 <- melt(datSlopes_GenOne_NoMig[,c("bot", "prop_sigPos", "ci_sigPos")],
                     id=c("bot", "ci_sigPos"))
         dm2 <- melt(datSlopes_GenOne_NoMig[,c("bot", "prop_sigNeg", "ci_sigNeg")],
                     id=c("bot", "ci_sigNeg"))

         #Rename columns
         setnames(dm1, old = "ci_sigPos", new = "ci")
         setnames(dm2, old = "ci_sigNeg", new = "ci")

         #Merge melted dataframes
         SlopeSum_Melt_NoMig <- rbind(dm1, dm2)
         SlopeSum_Melt_NoMig
```

bot	ci	variable	value
0.01	0.03060106	prop_sigPos	0.421
0.025	0.03027239	prop_sigPos	0.607
0.035	0.02925427	prop_sigPos	0.665
0.05	0.02788964	prop_sigPos	0.718
0.075	0.02751689	prop_sigPos	0.730
0.1	0.02669373	prop_sigPos	0.754
0.2	0.02914409	prop_sigPos	0.670
0.5	0.03085889	prop_sigPos	0.454
0.75	0.02994114	prop_sigPos	0.371
1	0.02884005	prop_sigPos	0.317
0.01	0.02360348	prop_sigNeg	0.176
0.025	0.02604360	prop_sigNeg	0.229
0.035	0.02608347	prop_sigNeg	0.230
0.05	0.02647089	prop_sigNeg	0.240
0.075	0.02704970	prop_sigNeg	0.256
0.1	0.02643305	prop_sigNeg	0.239
0.2	0.02687409	prop_sigNeg	0.251
0.5	0.02809570	prop_sigNeg	0.289
0.75	0.02818150	prop_sigNeg	0.292
1	0.02761243	prop_sigNeg	0.273

```
In [90]: #Plot proportion of significant slopes by migration rate
         PropSig_Bot_NoMig <- ggplot(SlopeSum_Melt_NoMig, aes(x = factor(bot), y = value,
             shape = variable, fill = variable)) +
             geom_errorbar(aes(ymin = value - ci, ymax = value + ci), width=0.15, size = 0.7,
             position = position_dodge(width = 0.55)) +
             ylab("Proportion sig. slopes") + xlab("Proportion of founding alleles") +
         geom_point(size = 3.5, color = "black",
             position = position_dodge(width = 0.55)) +
             scale_shape_manual(labels = c("Positive", "Negative"),values=c(22, 23))+
             scale_fill_manual(labels = c("Positive", "Negative"),values=c("white", "black")) +
             scale_y_continuous(breaks = seq(from = 0, to = 1.0, by = 0.05)) +
             labs(shape = 'Cline direction', fill = 'Cline direction') + ng1.45
         PropSig_Bot_NoMig
```



0.5 Do we need to show what's happening to the individual alleles?