Load libraries

Load libraries needed for the analyses.

```
#library(googlesheets4)
#library(httpuv)
#gs4_deauth()
#gs4_user()

library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)

library(ggplot2)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

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

```
library(wesanderson)

library(lme4)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'

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

Settings

Define variables for the analyses.

Palette: colors for figures.
se: function for standard errors.
percent: format numbers as percentage.

```
Palette <- wes_palette("FantasticFox1", 3, type = c("discrete"))

se <- function(x) sqrt(var(x) / length(x))

percent <- function(x, digits = 2, format = "f", ...) {
  paste0(formatC(100 * x, format = format, digits = digits, ...), "%")
}
```

Load data: line survival

```
line_survival <- read.csv(file = "extinction_all.csv")
line_survival$Regime <- as.factor(line_survival$Regime)

str(line_survival)
```

```
## 'data.frame':    65 obs. of  3 variables:
##  $ Generation: int  0 1 2 3 4 5 6 7 8 9 ...
##  $ Regime    : Factor w/ 8 levels "T1 (set 1)","T1 (set 2)",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Number    : int  60 55 51 42 36 30 20 13 6 3 ...
```

Load data: coverage

coverage: table containing mean coverage for each sample.

```
coverage <- read.table(file = "DP_range")[,c(1,2,4)]

colnames(coverage) <- c("Regime", "Sample", "coverage")
coverage$coverage <- round(coverage$coverage, digits = 2)

mean(coverage$coverage)
```

```
## [1] 66.00222
```

```
aggregate(coverage$coverage, by = list(coverage$Regime), FUN = mean)
```

```
##   Group.1        x
## 1      T1 68.49833
## 2      T2 76.59167
## 3      T5 52.91667
```

Load data: callability

callability: table containing the size of callable regions for each sample.

```
callability <- read.table(file = "callability.txt", header = T)

callability$Callability <- as.integer(callability$Callability)
callability$Sample <- as.factor(callability$Sample)

str(callability)
```

```
## 'data.frame':    18 obs. of  2 variables:
##  $ Callability: int  90566554 90566554 90566554 90566554 90566554 90566554 93817854 93817854 9381785
##  $ Sample     : Factor w/ 18 levels "T1-10","T1-19",..: 1 2 3 4 5 6 7 8 9 10 ...
```

Load data: manually curated singletons

singleton_raw: table containing information for manually curated singletons; each row is occupied by one singleton.

meaning of each column:
Regime: treatment;
Sample: sample ID, or MA-line ID;
chromosome: the scaffold in which the singleton is found;
position: the position in the scaffold where the singleton is found;
reference_allele: the reference allele in this scaffold_position;
alternate_allele: the alternate_allele in this scaffold_position;
approach: the variant discovery approach by which the singleton is found;
screenshot_id: the id of the screenshot for this singleton;
filter: whether the singleton passes (PASS) or fails (FAIL) manual inspection;
filter_description: reasons for failing the singleton (for singletons that passes, the value is PASS).

```
singleton_raw <- read.table(file = "singleton_raw")

colnames(singleton_raw) <- c("Regime", "Sample", "chromosome", "position",
                             "reference_allele", "alternate_allele",
                             "approach", "screenshot_id",
                             "filter", "filter_description")

singleton_raw$Regime <- as.factor(singleton_raw$Regime)
singleton_raw$Sample <- as.factor(singleton_raw$Sample)
singleton_raw$chromosome <- as.factor(singleton_raw$chromosome)
singleton_raw$position <- as.factor(singleton_raw$position)
```

```
singleton_raw$reference_allele <- as.factor(singleton_raw$reference_allele)
singleton_raw$alternate_allele <- as.factor(singleton_raw$alternate_allele)
singleton_raw$approach <- as.factor(singleton_raw$approach)
singleton_raw$screenshot_id <- as.factor(singleton_raw$screenshot_id)
singleton_raw$filter <- as.factor(singleton_raw$filter)
singleton_raw$filter_description <- as.factor(singleton_raw$filter_description)

singleton_raw$singleton_ID <- as.factor(paste(singleton_raw$chromosome,singleton_raw$position, sep = "_

str(singleton_raw)
```

```
## 'data.frame':    1283 obs. of  11 variables:
##  $ Regime            : Factor w/ 3 levels "T1","T2","T5": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Sample            : Factor w/ 18 levels "T1-10","T1-19",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ chromosome        : Factor w/ 7 levels "CM050494.1","CM050495.1",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ position          : Factor w/ 1225 levels "16897","31045",..: 724 777 1008 158 28 238 334 385 386
##  $ reference_allele  : Factor w/ 4 levels "A","C","G","T": 2 4 3 2 3 2 2 1 3 2 ...
##  $ alternate_allele  : Factor w/ 4 levels "A","C","G","T": 4 1 1 4 1 4 4 4 1 1 ...
##  $ approach          : Factor w/ 2 levels "consensus","probabilistic": 2 1 1 2 1 1 2 1 1 1 ...
##  $ screenshot_id     : Factor w/ 121 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 8 8 ...
##  $ filter            : Factor w/ 2 levels "FAIL","PASS": 2 2 1 1 1 1 2 1 1 1 ...
##  $ filter_description: Factor w/ 4 levels "ALIGNMENT_ARTEFACT",..: 3 3 4 1 4 4 3 1 1 1 ...
##  $ singleton_ID      : Factor w/ 1225 levels "CM050494.1_10031035",..: 14 22 76 82 86 88 102 108 109
```

Mutation rate: generate data frames

From **singleton_raw**, subset to generate the following three data frames:
singleton_PASS,
singleton_PASS_consensus,
singleton_PASS_probabilistic.

singleton_PASS contains the number of singletons that pass manual inspection for each approach for each sample.

singleton_PASS_consensus contains the number of singletons discovered by the consensus approach and pass manual inspection for each sample.

singleton_PASS_probabilistic contains the number of singletons discovered by the probabilistic approach and pass manual inspection for each sample.

```
singleton_PASS <- singleton_raw[which(singleton_raw$filter == "PASS"),] %>% group_by(Regime, Sample, ap

singleton_PASS_consensus <- singleton_PASS[which(singleton_PASS$approach == "consensus"),c(1,2,4)]
colnames(singleton_PASS_consensus) <- c("Regime", "Sample", "consensus")

singleton_PASS_probabilistic <- singleton_PASS[which(singleton_PASS$approach == "probabilistic"),c(1,2,
colnames(singleton_PASS_probabilistic) <- c("Regime", "Sample", "probabilistic")
```

From **singleton_raw**, subset to generate **singleton_PASS_overlap**, which contains the number of singletons discovered by both approaches and pass manual inspection for each sample.

```
singleton_PASS_overlap <- subset(singleton_raw, duplicated(singleton_raw$singleton_ID) & singleton_raw$

colnames(singleton_PASS_overlap) <- c("Regime", "Sample", "overlap")
```

Mutation rate: merge data frames

df: working data frame that merges callability, singleton_PASS_consensus, singleton_PASS_probabilistic, singleton_PASS_overlap.

```
df <- merge(singleton_PASS_probabilistic, merge(singleton_PASS_consensus,merge(singleton_PASS_overlap, 

df[is.na(df)] <- 0
```

Mutation rate: calculate

derive mutation rate for the two approaches and for overlapping mutations.

```
df$mu_probabilistic <- (df$probabilistic/df$Callability)/3
df$mu_consensus <- (df$consensus/df$Callability)/3
df$mu_overlap <- (df$overlap/df$Callability)/3

str(df)
```

```
## 'data.frame':    18 obs. of  9 variables:
##  $ Sample          : Factor w/ 18 levels "T1-10","T1-19",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Regime          : Factor w/ 3 levels "T1","T2","T5": 1 1 1 1 1 1 2 2 2 2 ...
##  $ probabilistic   : int  61 14 22 44 6 7 13 20 4 3 ...
##  $ consensus       : num  20 6 9 18 14 2 6 8 0 6 ...
##  $ overlap         : num  4 2 2 3 1 0 1 3 0 1 ...
##  $ Callability     : int  90566554 90566554 90566554 90566554 90566554 90566554 93817854 93817854 938
##  $ mu_probabilistic: num  2.25e-07 5.15e-08 8.10e-08 1.62e-07 2.21e-08 ...
##  $ mu_consensus    : num  7.36e-08 2.21e-08 3.31e-08 6.62e-08 5.15e-08 ...
##  $ mu_overlap      : num  1.47e-08 7.36e-09 7.36e-09 1.10e-08 3.68e-09 ...
```

Mutation rate: descriptions

overlapping

```
min(df$mu_overlap)
```

```
## [1] 0
```

```
max(df$mu_overlap)
```

```
## [1] 3.594185e-08
```

```
cbind(aggregate(df$mu_overlap, by = list(df$Regime), FUN = mean),aggregate(df$mu_overlap, by = list(df$
```

```
##   Group.1            x
## 1      T1 7.361069e-09
## 2      T2 2.960820e-09
## 3      T5 1.198062e-08
##   aggregate(df$mu_overlap, by = list(df$Regime), FUN = se)[, 2]
## 1                                                 2.124958e-09
## 2                                                 1.695700e-09
## 3                                                 6.540143e-09
```

consensus

```
min(df$mu_consensus)
```

```
## [1] 0
```

```
max(df$mu_consensus)
```

```
## [1] 1.222023e-07
```

```
cbind(aggregate(df$mu_consensus, by = list(df$Regime), FUN = mean),aggregate(df$mu_consensus, by = list
```

```
##   Group.1            x
## 1      T1 4.232615e-08
## 2      T2 1.361977e-08
## 3      T5 4.971956e-08
##   aggregate(df$mu_consensus, by = list(df$Regime), FUN = se)[,
## 1                                                 1.057153e-08
## 2                                                 4.890285e-09
## 3                                                 2.283400e-08
```

probabilistic

```
min(df$mu_probabilistic)
```

```
## [1] 3.552984e-09
```

```
max(df$mu_probabilistic)
```

```
## [1] 2.245126e-07
```

```
cbind(aggregate(df$mu_probabilistic, by = list(df$Regime), FUN = mean),aggregate(df$mu_probabilistic, by
```

```
##   Group.1            x
## 1      T1 9.446705e-08
## 2      T2 2.546305e-08
## 3      T5 7.308176e-08
##   aggregate(df$mu_probabilistic, by = list(df$Regime), FUN = se)[,
## 1                                                   3.339182e-08
## 2                                                   1.106254e-08
## 3                                                   3.270072e-08
```

Mutation rate: glm

m1: Sasani et al., 2019; m2, m3, m4: our models.

Report results produced by m2 and m4 (m2: absolute numbers of mutations; m4: relative number of mutations).

```
m1 <- anova(glm(df$overlap ~ df$Regime, family = poisson(link = 'identity')), test = 'Chisq')

m2 <- Anova(glmer(df$overlap ~ df$Regime + (1 | df$Regime), weights = df$Callability, family = poisson))
```

```
## boundary (singular) fit: see help('isSingular')
```

```
m3 <- anova(glm(cbind(df$overlap, df$Callability - df$overlap) ~ df$Regime, family = binomial(link = 'lo

m4 <- Anova(glmer(cbind(df$overlap, df$Callability - df$overlap) ~ df$Regime + (1 | df$Regime), family =
```

```
## boundary (singular) fit: see help('isSingular')
```

```
#print(m1)
print(m2)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: df$overlap
##                Chisq Df Pr(>Chisq)
## df$Regime 759046551  2  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#print(m3)
print(m4)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: cbind(df$overlap, df$Callability - df$overlap)
##             Chisq Df Pr(>Chisq)
## df$Regime 8.1648  2    0.01687 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
m1_consensus <- anova(glm(df$consensus ~ df$Regime, family = poisson(link = 'identity')), test = 'Chisq

m2_consensus <- Anova(glmer(df$consensus ~ df$Regime + (1 | df$Regime), family = poisson))
```

```
## boundary (singular) fit: see help('isSingular')
```

```
m3_consensus <- anova(glm(cbind(df$consensus, df$Callability - df$consensus) ~ df$Regime, family = binor

m4_consensus <- Anova(glmer(cbind(df$consensus, df$Callability - df$consensus) ~ df$Regime + (1 | df$Reg
```

```
## boundary (singular) fit: see help('isSingular')
```

```
#print(m1_consensus)
print(m2_consensus)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: df$consensus
##            Chisq Df Pr(>Chisq)
## df$Regime 30.028  2  3.016e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#print(m3_consensus)
print(m4_consensus)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: cbind(df$consensus, df$Callability - df$consensus)
##            Chisq Df Pr(>Chisq)
## df$Regime 30.798  2  2.053e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
m1_probabilistic <- anova(glm(df$probabilistic ~ df$Regime, family = poisson(link = 'identity')), test =

m2_probabilistic <- Anova(glmer(df$probabilistic ~ df$Regime + (1 | df$Regime), family = poisson))
```

```
## boundary (singular) fit: see help('isSingular')
```

```
m3_probabilistic <- anova(glm(cbind(df$probabilistic, df$Callability - df$probabilistic) ~ df$Regime, fa

m4_probabilistic <- Anova(glmer(cbind(df$probabilistic, df$Callability - df$probabilistic) ~ df$Regime +
```

```
## boundary (singular) fit: see help('isSingular')
```

```
#print(m1_probabilistic)
print(m2_probabilistic)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: df$probabilistic
##            Chisq Df Pr(>Chisq)
## df$Regime 54.865  2   1.22e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#print(m3_probabilistic)
print(m4_probabilistic)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: cbind(df$probabilistic, df$Callability - df$probabilistic)
##            Chisq Df Pr(>Chisq)
## df$Regime 57.841  2  2.754e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Pearson's correlation

Correlation test between callability and number of mutations (overlap) across lines.

```
cor.test(df$Callability, df$overlap, method = 'pearson')
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$Callability and df$overlap
## t = -0.43226, df = 16, p-value = 0.6713
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.5468777  0.3784122
## sample estimates:
##        cor
## -0.1074391
```

```
cor.test(df$Callability, df$consensus, method = 'pearson')
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$Callability and df$consensus
## t = -1.0259, df = 16, p-value = 0.3202
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.6409648  0.2470902
## sample estimates:
##        cor
## -0.2484396
```

```
cor.test(df$Callability, df$probabilistic, method = 'pearson')
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$Callability and df$probabilistic
## t = -1.6248, df = 16, p-value = 0.1237
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.7171973  0.1098267
## sample estimates:
##        cor
## -0.3763402
```

Mutation spectrum: generate data frames

Three data frames will be generated; one for the consensus approach (**spectrum_consensus_df**), one for the probabilistic approach (**spectrum_probabilistic_df**), and one for the overlap (**spectrum_overlap_df**).

column category:
category_1: G to A, C to T;
category_2: A to G, T to C;
category_3: G to C, C to G;
category_4: G to T, C to A;
category_5: T to G, A to C;
category_6: T to A, A to T.

```
spectrum_overlap <- subset(singleton_raw, duplicated(singleton_raw$singleton_ID) & singleton_raw$filter

spectrum_overlap$category <- ifelse(spectrum_overlap$reference_allele == "G" &
spectrum_overlap$alternate_allele == "A", "category_1",
ifelse(spectrum_overlap$reference_allele == "C" &
spectrum_overlap$alternate_allele == "T", "category_1",
ifelse(spectrum_overlap$reference_allele == "A" &
spectrum_overlap$alternate_allele == "G", "category_2",
ifelse(spectrum_overlap$reference_allele == "T" &
spectrum_overlap$alternate_allele == "C", "category_2",
ifelse(spectrum_overlap$reference_allele == "G" &
spectrum_overlap$alternate_allele == "C", "category_3",
ifelse(spectrum_overlap$reference_allele == "C" &
spectrum_overlap$alternate_allele == "G", "category_3",
ifelse(spectrum_overlap$reference_allele == "G" &
spectrum_overlap$alternate_allele == "T", "category_4",
ifelse(spectrum_overlap$reference_allele == "C" &
spectrum_overlap$alternate_allele == "A", "category_4",
ifelse(spectrum_overlap$reference_allele == "T" &
spectrum_overlap$alternate_allele == "G", "category_5",
ifelse(spectrum_overlap$reference_allele == "A" &
spectrum_overlap$alternate_allele == "C", "category_5",
ifelse(spectrum_overlap$reference_allele == "T" &
spectrum_overlap$alternate_allele == "A", "category_6",
"category_6"))))))))))))

spectrum_overlap$category <- as.factor(spectrum_overlap$category)

spectrum_overlap_df <- aggregate(spectrum_overlap$Sample,
                                 by = list(spectrum_overlap$Regime,
                                           spectrum_overlap$category), FUN = NROW)
colnames(spectrum_overlap_df) <- c("Regime", "category", "number")
```

```
spectrum_consensus <- subset(singleton_raw, singleton_raw$approach == "consensus" & singleton_raw$filte

spectrum_consensus$category <- ifelse(spectrum_consensus$reference_allele == "G" &
spectrum_consensus$alternate_allele == "A", "category_1",
ifelse(spectrum_consensus$reference_allele == "C" &
spectrum_consensus$alternate_allele == "T", "category_1",
ifelse(spectrum_consensus$reference_allele == "A" &
spectrum_consensus$alternate_allele == "G", "category_2",
```

```
ifelse(spectrum_consensus$reference_allele == "T" &
spectrum_consensus$alternate_allele == "C", "category_2",
ifelse(spectrum_consensus$reference_allele == "G" &
spectrum_consensus$alternate_allele == "C", "category_3",
ifelse(spectrum_consensus$reference_allele == "C" &
spectrum_consensus$alternate_allele == "G", "category_3",
ifelse(spectrum_consensus$reference_allele == "G" &
spectrum_consensus$alternate_allele == "T", "category_4",
ifelse(spectrum_consensus$reference_allele == "C" &
spectrum_consensus$alternate_allele == "A", "category_4",
ifelse(spectrum_consensus$reference_allele == "T" &
spectrum_consensus$alternate_allele == "G", "category_5",
ifelse(spectrum_consensus$reference_allele == "A" &
spectrum_consensus$alternate_allele == "C", "category_5",
ifelse(spectrum_consensus$reference_allele == "T" &
spectrum_consensus$alternate_allele == "A", "category_6",
"category_6")))))))))))

spectrum_consensus$category <- as.factor(spectrum_consensus$category)

spectrum_consensus_df <- aggregate(spectrum_consensus$Sample,
                                   by = list(spectrum_consensus$Regime,
                                             spectrum_consensus$category), FUN = NROW)
colnames(spectrum_consensus_df) <- c("Regime", "category", "number")
```

```
spectrum_probabilistic <- subset(singleton_raw, singleton_raw$approach == "probabilistic" & singleton_ra

spectrum_probabilistic$category <- ifelse(spectrum_probabilistic$reference_allele == "G" &
spectrum_probabilistic$alternate_allele == "A", "category_1",
ifelse(spectrum_probabilistic$reference_allele == "C" &
spectrum_probabilistic$alternate_allele == "T", "category_1",
ifelse(spectrum_probabilistic$reference_allele == "A" &
spectrum_probabilistic$alternate_allele == "G", "category_2",
ifelse(spectrum_probabilistic$reference_allele == "T" &
spectrum_probabilistic$alternate_allele == "C", "category_2",
ifelse(spectrum_probabilistic$reference_allele == "G" &
spectrum_probabilistic$alternate_allele == "C", "category_3",
ifelse(spectrum_probabilistic$reference_allele == "C" &
spectrum_probabilistic$alternate_allele == "G", "category_3",
ifelse(spectrum_probabilistic$reference_allele == "G" &
spectrum_probabilistic$alternate_allele == "T", "category_4",
ifelse(spectrum_probabilistic$reference_allele == "C" &
spectrum_probabilistic$alternate_allele == "A", "category_4",
ifelse(spectrum_probabilistic$reference_allele == "T" &
spectrum_probabilistic$alternate_allele == "G", "category_5",
ifelse(spectrum_probabilistic$reference_allele == "A" &
spectrum_probabilistic$alternate_allele == "C", "category_5",
ifelse(spectrum_probabilistic$reference_allele == "T" &
spectrum_probabilistic$alternate_allele == "A", "category_6",
"category_6")))))))))))

spectrum_probabilistic$category <- as.factor(spectrum_probabilistic$category)
```

```
spectrum_probabilistic_df <- aggregate(spectrum_probabilistic$Sample,
                                by = list(spectrum_probabilistic$Regime,
                                          spectrum_probabilistic$category), FUN = NROW)
colnames(spectrum_probabilistic_df) <- c("Regime", "category", "number")
```

Mutation spectrum: chi-square test

chisq.test for mutation spectrum for each approach and the overlapping.

```
spectrum_overlap_df
```

```
##   Regime   category number
## 1     T1 category_1     12
## 2     T2 category_1      5
## 3     T5 category_1     18
## 4     T5 category_2      1
## 5     T5 category_4      1
```

```
# update the numbers below


spectrum_overlap_df_chisq.test <- cbind(c(12,5,8),c(0,0,1),c(0,0,0),c(0,0,1),c(0,0,0),c(0,0,0))
dimnames(spectrum_overlap_df_chisq.test) <-
  list(Regime = c("T1", "T2", "T5"),
       category = c("category_1", "category_2", "category_3",
                    "category_4", "category_5", "category_6"))

spectrum_overlap_df_chisq.test <- as.data.frame(spectrum_overlap_df_chisq.test)

chisq.test(spectrum_overlap_df_chisq.test,
           simulate.p.value = TRUE, B = 10000)
```

```
## Warning in chisq.test(spectrum_overlap_df_chisq.test, simulate.p.value = TRUE,
## : cannot compute simulated p-value with zero marginals
```

```
## Warning in chisq.test(spectrum_overlap_df_chisq.test, simulate.p.value = TRUE,
## : Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  spectrum_overlap_df_chisq.test
## X-squared = NaN, df = 10, p-value = NA
```

```
chisq.test(spectrum_overlap_df_chisq.test)$observed
```

```
## Warning in chisq.test(spectrum_overlap_df_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    category_1 category_2 category_3 category_4 category_5 category_6
## T1         12          0          0          0          0          0
## T2          5          0          0          0          0          0
## T5          8          1          0          1          0          0
```

```
chisq.test(spectrum_overlap_df_chisq.test)$expected
```

```
## Warning in chisq.test(spectrum_overlap_df_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    category_1 category_2 category_3 category_4 category_5 category_6
## T1  11.111111  0.4444444          0  0.4444444          0          0
## T2   4.629630  0.1851852          0  0.1851852          0          0
## T5   9.259259  0.3703704          0  0.3703704          0          0
```

```
chisq.test(spectrum_overlap_df_chisq.test$category_1)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_overlap_df_chisq.test$category_1
## X-squared = 2.96, df = 2, p-value = 0.2276
```

```
chisq.test(spectrum_overlap_df_chisq.test$category_2)
```

```
## Warning in chisq.test(spectrum_overlap_df_chisq.test$category_2): Chi-squared
## approximation may be incorrect
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_overlap_df_chisq.test$category_2
## X-squared = 2, df = 2, p-value = 0.3679
```

```
chisq.test(spectrum_overlap_df_chisq.test$category_4)
```

```
## Warning in chisq.test(spectrum_overlap_df_chisq.test$category_4): Chi-squared
## approximation may be incorrect
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_overlap_df_chisq.test$category_4
## X-squared = 2, df = 2, p-value = 0.3679
```

```
chisq.test(spectrum_overlap_df_chisq.test[,c(1,2,4)],
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  spectrum_overlap_df_chisq.test[, c(1, 2, 4)]
## X-squared = 3.672, df = NA, p-value = 0.4777
```

```
spectrum_consensus_df
```

```
##    Regime   category number
## 1      T1 category_1     61
## 2      T2 category_1     22
## 3      T5 category_1     76
## 4      T1 category_2      4
## 5      T5 category_2      1
## 6      T2 category_3      1
## 7      T5 category_3      1
## 8      T1 category_4      2
## 9      T5 category_4      3
## 10     T1 category_6      2
## 11     T5 category_6      2
```

```
# update the numbers below


spectrum_consensus_df_chisq.test <- cbind(c(61,22,76),c(4,0,1),c(0,1,1),c(2,0,3),c(0,0,0),c(2,0,2))
dimnames(spectrum_consensus_df_chisq.test) <-
  list(Regime = c("T1", "T2", "T5"),
       category = c("category_1", "category_2", "category_3",
                    "category_4", "category_5", "category_6"))

spectrum_consensus_df_chisq.test <- as.data.frame(spectrum_consensus_df_chisq.test)

chisq.test(spectrum_consensus_df_chisq.test,
           simulate.p.value = TRUE, B = 10000)
```

```
## Warning in chisq.test(spectrum_consensus_df_chisq.test, simulate.p.value =
## TRUE, : cannot compute simulated p-value with zero marginals
```

```
## Warning in chisq.test(spectrum_consensus_df_chisq.test, simulate.p.value =
## TRUE, : Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  spectrum_consensus_df_chisq.test
## X-squared = NaN, df = 10, p-value = NA
```

```
chisq.test(spectrum_consensus_df_chisq.test)$observed
```

```
## Warning in chisq.test(spectrum_consensus_df_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    category_1 category_2 category_3 category_4 category_5 category_6
## T1         61          4          0          2          0          2
## T2         22          0          1          0          0          0
## T5         76          1          1          3          0          2
```

```
chisq.test(spectrum_consensus_df_chisq.test)$expected
```

```
## Warning in chisq.test(spectrum_consensus_df_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    category_1 category_2 category_3 category_4 category_5 category_6
## T1   62.69143  1.9714286  0.7885714  1.9714286          0  1.5771429
## T2   20.89714  0.6571429  0.2628571  0.6571429          0  0.5257143
## T5   75.41143  2.3714286  0.9485714  2.3714286          0  1.8971429
```

```
chisq.test(spectrum_consensus_df_chisq.test$category_1)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_consensus_df_chisq.test$category_1
## X-squared = 29.321, df = 2, p-value = 4.296e-07
```

```
chisq.test(spectrum_consensus_df_chisq.test$category_2)
```

```
## Warning in chisq.test(spectrum_consensus_df_chisq.test$category_2): Chi-squared
## approximation may be incorrect
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_consensus_df_chisq.test$category_2
## X-squared = 5.2, df = 2, p-value = 0.07427
```

```
chisq.test(spectrum_consensus_df_chisq.test$category_4)
```

```
## Warning in chisq.test(spectrum_consensus_df_chisq.test$category_4): Chi-squared
## approximation may be incorrect
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_consensus_df_chisq.test$category_4
## X-squared = 2.8, df = 2, p-value = 0.2466
```

```
chisq.test(spectrum_consensus_df_chisq.test[,c(1,2,3,4,6)],
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  spectrum_consensus_df_chisq.test[, c(1, 2, 3, 4, 6)]
## X-squared = 7.9735, df = NA, p-value = 0.4223
```

```
spectrum_probabilistic_df
```

```
##    Regime   category number
## 1      T1 category_1    131
## 2      T2 category_1     36
## 3      T5 category_1    103
## 4      T1 category_2      6
## 5      T2 category_2      3
## 6      T5 category_2      6
## 7      T1 category_3      1
## 8      T2 category_3      2
## 9      T5 category_3      1
## 10     T1 category_4     12
## 11     T2 category_4      2
## 12     T5 category_4      4
## 13     T1 category_5      1
## 14     T5 category_5      1
## 15     T1 category_6      3
## 16     T5 category_6      7
```

```
# update the numbers below


spectrum_probabilistic_df_chisq.test <- cbind(c(131,36,103),c(6,3,6),c(1,2,1),c(12,2,4),c(1,1,3),c(3,0,
dimnames(spectrum_probabilistic_df_chisq.test) <-
  list(Regime = c("T1", "T2", "T5"),
       category = c("category_1", "category_2", "category_3",
                    "category_4", "category_5", "category_6"))

spectrum_probabilistic_df_chisq.test <- as.data.frame(spectrum_probabilistic_df_chisq.test)

chisq.test(spectrum_probabilistic_df_chisq.test,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  spectrum_probabilistic_df_chisq.test
## X-squared = 14.013, df = NA, p-value = 0.1695
```

```
chisq.test(spectrum_probabilistic_df_chisq.test)$observed
```

```
## Warning in chisq.test(spectrum_probabilistic_df_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    category_1 category_2 category_3 category_4 category_5 category_6
## T1        131          6          1         12          1          3
## T2         36          3          2          2          1          0
## T5        103          6          1          4          3          7
```

```
chisq.test(spectrum_probabilistic_df_chisq.test)$expected
```

```
## Warning in chisq.test(spectrum_probabilistic_df_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    category_1 category_2 category_3 category_4 category_5 category_6
## T1  129.13043   7.173913  1.9130435   8.608696  2.3913043   4.782609
## T2   36.89441   2.049689  0.5465839   2.459627  0.6832298   1.366460
## T5  103.97516   5.776398  1.5403727   6.931677  1.9254658   3.850932
```

```
chisq.test(spectrum_probabilistic_df_chisq.test$category_1)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_probabilistic_df_chisq.test$category_1
## X-squared = 52.956, df = 2, p-value = 3.168e-12
```

```
chisq.test(spectrum_probabilistic_df_chisq.test$category_2)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_probabilistic_df_chisq.test$category_2
## X-squared = 1.2, df = 2, p-value = 0.5488
```

```
chisq.test(spectrum_probabilistic_df_chisq.test$category_4)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  spectrum_probabilistic_df_chisq.test$category_4
## X-squared = 9.3333, df = 2, p-value = 0.009404
```

```
chisq.test(spectrum_probabilistic_df_chisq.test[,c(1,2,3,4,5,6)],
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  spectrum_probabilistic_df_chisq.test[, c(1, 2, 3, 4, 5, 6)]
## X-squared = 14.013, df = NA, p-value = 0.163
```

Transition-transversion: generate data frames

Three data frames will be generated; one for the consensus approach (**TsTv_consensus_df**), one for the probabilistic approach (**TsTv_probabilistic_df**), and one for the overlap (**TsTv_overlap_df**).

column TsTv:
A to G, G to A, C to T and T to C mutations are transitions; the rest are transversions.

```
TsTv_overlap <- subset(singleton_raw, duplicated(singleton_raw$singleton_ID) & singleton_raw$filter ==

TsTv_overlap$TsTv <- ifelse(TsTv_overlap$reference_allele == "A" &
                              TsTv_overlap$alternate_allele == "G", "Transition",
                       ifelse(TsTv_overlap$reference_allele == "G" &
                              TsTv_overlap$alternate_allele == "A", "Transition",
                       ifelse(TsTv_overlap$reference_allele == "C" &
                              TsTv_overlap$alternate_allele == "T", "Transition",
                       ifelse(TsTv_overlap$reference_allele == "T" &
                              TsTv_overlap$alternate_allele == "C", "Transition",
                              "Transversion"))))

TsTv_overlap_df <-
  aggregate(TsTv_overlap$Sample,
            by = list(TsTv_overlap$Regime, TsTv_overlap$Sample, TsTv_overlap$TsTv),
            FUN = NROW)
colnames(TsTv_overlap_df) <- c("Regime", "Sample", "TsTv", "number")

temp <-
aggregate(TsTv_overlap$Sample, by = list(TsTv_overlap$Regime, TsTv_overlap$Sample),
          FUN = NROW)[,c(2,3)]
colnames(temp) <- c("Sample", "total")

TsTv_overlap_df <- merge(TsTv_overlap_df, temp, by = "Sample", all = T)
TsTv_overlap_df$perc <- TsTv_overlap_df$number/TsTv_overlap_df$total
```

```
TsTv_consensus <- singleton_raw[which(singleton_raw$filter == "PASS" & singleton_raw$approach == "conse

TsTv_consensus$TsTv <- ifelse(TsTv_consensus$reference_allele == "A" &
                              TsTv_consensus$alternate_allele == "G", "Transition",
                       ifelse(TsTv_consensus$reference_allele == "G" &
                              TsTv_consensus$alternate_allele == "A", "Transition",
                       ifelse(TsTv_consensus$reference_allele == "C" &
                              TsTv_consensus$alternate_allele == "T", "Transition",
                       ifelse(TsTv_consensus$reference_allele == "T" &
                              TsTv_consensus$alternate_allele == "C", "Transition",
                              "Transversion"))))

TsTv_consensus_df <-
  aggregate(TsTv_consensus$Sample,
            by = list(TsTv_consensus$Regime, TsTv_consensus$Sample, TsTv_consensus$TsTv),
            FUN = NROW)
colnames(TsTv_consensus_df) <- c("Regime", "Sample", "TsTv", "number")

temp <-
aggregate(TsTv_consensus$Sample, by = list(TsTv_consensus$Regime, TsTv_consensus$Sample),
```

```
          FUN = NROW)[,c(2,3)]
colnames(temp) <- c("Sample", "total")

TsTv_consensus_df <- merge(TsTv_consensus_df, temp, by = "Sample", all = T)
TsTv_consensus_df$perc <- TsTv_consensus_df$number/TsTv_consensus_df$total
```

```
TsTv_probabilistic <- singleton_raw[which(singleton_raw$filter == "PASS" & singleton_raw$approach == "p

TsTv_probabilistic$TsTv <- ifelse(TsTv_probabilistic$reference_allele == "A" &
                                  TsTv_probabilistic$alternate_allele == "G", "Transition",
                           ifelse(TsTv_probabilistic$reference_allele == "G" &
                                  TsTv_probabilistic$alternate_allele == "A", "Transition",
                           ifelse(TsTv_probabilistic$reference_allele == "C" &
                                  TsTv_probabilistic$alternate_allele == "T", "Transition",
                           ifelse(TsTv_probabilistic$reference_allele == "T" &
                                  TsTv_probabilistic$alternate_allele == "C", "Transition",
                                  "Transversion"))))

TsTv_probabilistic_df <-
  aggregate(TsTv_probabilistic$Sample,
            by = list(TsTv_probabilistic$Regime, TsTv_probabilistic$Sample, TsTv_probabilistic$TsTv),
            FUN = NROW)
colnames(TsTv_probabilistic_df) <- c("Regime", "Sample", "TsTv", "number")

temp <-
aggregate(TsTv_probabilistic$Sample, by = list(TsTv_probabilistic$Regime, TsTv_probabilistic$Sample),
          FUN = NROW)[,c(2,3)]
colnames(temp) <- c("Sample", "total")

TsTv_probabilistic_df <- merge(TsTv_probabilistic_df, temp, by = "Sample", all = T)
TsTv_probabilistic_df$perc <- TsTv_probabilistic_df$number/TsTv_probabilistic_df$total
```

Transition-transversion: chi-square test

chisq.test for the occurrence of transitions and transversions for each approach and the overlapping.

```
TsTv <- rep(c("Transition", "Transversion"), times = 18)
Regime <- rep(c("T1", "T2", "T5"), each = 36)
Sample <- rep(c("T1-10", "T1-19", "T1-37", "T1-46", "T1-57", "T1-6",
                "T2-1", "T2-22", "T2-34", "T2-4", "T2-42", "T2-52",
                "T5-45", "T5-5", "T5-70", "T5-81", "T5-86", "T5-87"), each = 2)

temp <- cbind(cbind(Regime, Sample), TsTv)

TsTv_overlap_df_temp1 <- merge(TsTv_overlap_df, temp,
                               by = c("Regime", "Sample", "TsTv"), all = T)

TsTv_overlap_df_temp1[is.na(TsTv_overlap_df_temp1)] <- 0

TsTv_overlap_df_temp2 <- aggregate(TsTv_overlap_df_temp1$number,
                                   by = list(TsTv_overlap_df_temp1$Regime,
```

```
                                                    TsTv_overlap_df_temp1$TsTv),
                                             FUN = sum)
colnames(TsTv_overlap_df_temp2) <- c("Regime", "TsTv", "count")

# update the numbers below


TsTv_overlap_df_chisq.test <- rbind(c(12,5,19),c(0,0,1))
dimnames(TsTv_overlap_df_chisq.test) <-
  list(category = c("Transition", "Transversion"),
       Regime = c("T1", "T2", "T5"))

TsTv_overlap_df_chisq.test <- as.data.frame(TsTv_overlap_df_chisq.test)

chisq.test(TsTv_overlap_df_chisq.test,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  TsTv_overlap_df_chisq.test
## X-squared = 0.87361, df = NA, p-value = 1
```

```
chisq.test(TsTv_overlap_df_chisq.test)$observed
```

```
## Warning in chisq.test(TsTv_overlap_df_chisq.test): Chi-squared approximation
## may be incorrect
```

```
##              T1 T2 T5
## Transition   12  5 19
## Transversion  0  0  1
```

```
chisq.test(TsTv_overlap_df_chisq.test)$expected
```

```
## Warning in chisq.test(TsTv_overlap_df_chisq.test): Chi-squared approximation
## may be incorrect
```

```
##                      T1        T2         T5
## Transition   11.6756757 4.8648649 19.4594595
## Transversion  0.3243243 0.1351351  0.5405405
```

```
chisq.test(TsTv_overlap_df_chisq.test$T1,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_overlap_df_chisq.test$T1
## X-squared = 12, df = NA, p-value = 5e-04
```

```
chisq.test(TsTv_overlap_df_chisq.test$T2,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_overlap_df_chisq.test$T2
## X-squared = 5, df = NA, p-value = 0.06479
```

```
chisq.test(TsTv_overlap_df_chisq.test$T5,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_overlap_df_chisq.test$T5
## X-squared = 16.2, df = NA, p-value = 2e-04
```

```
TsTv <- rep(c("Transition", "Transversion"), times = 18)
Regime <- rep(c("T1", "T2", "T5"), each = 36)
Sample <- rep(c("T1-10", "T1-19", "T1-37", "T1-46", "T1-57", "T1-6",
                "T2-1", "T2-22", "T2-34", "T2-4", "T2-42", "T2-52",
                "T5-45", "T5-5", "T5-70", "T5-81", "T5-86", "T5-87"), each = 2)

temp <- cbind(cbind(Regime, Sample), TsTv)

TsTv_consensus_df_temp1 <- merge(TsTv_consensus_df, temp,
                                 by = c("Regime", "Sample", "TsTv"), all = T)

TsTv_consensus_df_temp1[is.na(TsTv_consensus_df_temp1)] <- 0

TsTv_consensus_df_temp2 <- aggregate(TsTv_consensus_df_temp1$number,
                                     by = list(TsTv_consensus_df_temp1$Regime,
                                               TsTv_consensus_df_temp1$TsTv),
                                     FUN = sum)
colnames(TsTv_consensus_df_temp2) <- c("Regime", "TsTv", "count")

TsTv_consensus_df_temp2
```

```
##   Regime         TsTv count
## 1     T1   Transition    65
## 2     T2   Transition    22
## 3     T5   Transition    77
## 4     T1 Transversion     4
## 5     T2 Transversion     1
## 6     T5 Transversion     6
```

```
# update the numbers below
```

```
TsTv_consensus_df_chisq.test <- rbind(c(65,22,77),c(4,1,6))
dimnames(TsTv_consensus_df_chisq.test) <-
  list(category = c("Transition", "Transversion"),
       Regime = c("T1", "T2", "T5"))

TsTv_consensus_df_chisq.test <- as.data.frame(TsTv_consensus_df_chisq.test)

chisq.test(TsTv_consensus_df_chisq.test,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  TsTv_consensus_df_chisq.test
## X-squared = 0.29995, df = NA, p-value = 0.8384
```

```
chisq.test(TsTv_consensus_df_chisq.test)$observed
```

```
## Warning in chisq.test(TsTv_consensus_df_chisq.test): Chi-squared approximation
## may be incorrect
```

```
##              T1 T2 T5
## Transition   65 22 77
## Transversion  4  1  6
```

```
chisq.test(TsTv_consensus_df_chisq.test)$expected
```

```
## Warning in chisq.test(TsTv_consensus_df_chisq.test): Chi-squared approximation
## may be incorrect
```

```
##                     T1        T2        T5
## Transition   64.662857 21.554286 77.782857
## Transversion  4.337143  1.445714  5.217143
```

```
chisq.test(TsTv_consensus_df_chisq.test$T1,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_consensus_df_chisq.test$T1
## X-squared = 53.928, df = NA, p-value = 9.999e-05
```

```
chisq.test(TsTv_consensus_df_chisq.test$T2,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_consensus_df_chisq.test$T2
## X-squared = 19.174, df = NA, p-value = 9.999e-05
```

```
chisq.test(TsTv_consensus_df_chisq.test$T5,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_consensus_df_chisq.test$T5
## X-squared = 60.735, df = NA, p-value = 9.999e-05
```

```
TsTv <- rep(c("Transition", "Transversion"), times = 18)
Regime <- rep(c("T1", "T2", "T5"), each = 36)
Sample <- rep(c("T1-10", "T1-19", "T1-37", "T1-46", "T1-57", "T1-6",
                "T2-1", "T2-22", "T2-34", "T2-4", "T2-42", "T2-52",
                "T5-45", "T5-5", "T5-70", "T5-81", "T5-86", "T5-87"), each = 2)

temp <- cbind(cbind(Regime, Sample), TsTv)

TsTv_probabilistic_df_temp1 <- merge(TsTv_probabilistic_df, temp,
                                     by = c("Regime", "Sample", "TsTv"), all = T)

TsTv_probabilistic_df_temp1[is.na(TsTv_probabilistic_df_temp1)] <- 0

TsTv_probabilistic_df_temp2 <- aggregate(TsTv_probabilistic_df_temp1$number,
                                         by = list(TsTv_probabilistic_df_temp1$Regime,
                                                   TsTv_probabilistic_df_temp1$TsTv),
                                         FUN = sum)
colnames(TsTv_probabilistic_df_temp2) <- c("Regime", "TsTv", "count")

TsTv_probabilistic_df_temp2
```

```
##   Regime         TsTv count
## 1     T1   Transition   137
## 2     T2   Transition    39
## 3     T5   Transition   109
## 4     T1 Transversion    17
## 5     T2 Transversion     4
## 6     T5 Transversion    13
```

```
# update the numbers below


TsTv_probabilistic_df_chisq.test <- rbind(c(137,39,109),c(17,4,13))
dimnames(TsTv_probabilistic_df_chisq.test) <-
  list(category = c("Transition", "Transversion"),
```

```
       Regime = c("T1", "T2", "T5"))

TsTv_probabilistic_df_chisq.test <- as.data.frame(TsTv_probabilistic_df_chisq.test)

chisq.test(TsTv_probabilistic_df_chisq.test,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  TsTv_probabilistic_df_chisq.test
## X-squared = 0.10646, df = NA, p-value = 0.9713
```

```
chisq.test(TsTv_probabilistic_df_chisq.test)$observed
```

```
## Warning in chisq.test(TsTv_probabilistic_df_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##               T1 T2  T5
## Transition   137 39 109
## Transversion  17  4  13
```

```
chisq.test(TsTv_probabilistic_df_chisq.test)$expected
```

```
## Warning in chisq.test(TsTv_probabilistic_df_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##                     T1        T2        T5
## Transition   137.58621 38.416928 108.99687
## Transversion  16.41379  4.583072  13.00313
```

```
chisq.test(TsTv_probabilistic_df_chisq.test$T1,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_probabilistic_df_chisq.test$T1
## X-squared = 93.506, df = NA, p-value = 9.999e-05
```

```
chisq.test(TsTv_probabilistic_df_chisq.test$T2,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_probabilistic_df_chisq.test$T2
## X-squared = 28.488, df = NA, p-value = 9.999e-05
```

```
chisq.test(TsTv_probabilistic_df_chisq.test$T5,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  TsTv_probabilistic_df_chisq.test$T5
## X-squared = 75.541, df = NA, p-value = 9.999e-05
```

VEP (Variant Effect Prediction): prepare input

Prepare the input file for blast. This step is necessary because the reference assembly we use (ASM164373v4) is not available for VEP on WormBase.

Specifically, prepare the file as below, then use the file "~/My Drive/Nematode/Nematode_ASM164373v4_data/VEP_input.t to blast against GCF_010183535.1 assembly (which is available for VEP on WormBase) to derive corresponding chromosome and position for variant effect prediction (VEP).

Blast each site and store the scaffold name, the beginning and the end of the fragment, then save it as "~/My Drive/Nematode/Nematode_ASM164373v4_data/VEP_input_blast.txt"

```
VEP_input <- subset(singleton_raw, duplicated(singleton_raw$singleton_ID) & singleton_raw$filter == "PA
VEP_input$position <- as.integer(as.character(VEP_input$position))

VEP_input$beg <- VEP_input$position - 200
VEP_input$end <- VEP_input$position + 200

write.table(VEP_input, file = "~/My Drive/Nematode/Nematode_ASM164373v4_data/VEP_input.txt", append = F
```

Prepare the input file for VEP.

Specifically, prepare the file as below, then upload the file "~/My Drive/Nematode/Nematode_ASM164373v4_data/VEP_inp to WomBase for VEP.

Download the output of VEP as txt, save it as "~/My Drive/Nematode/Nematode_ASM164373v4_data/VEP_output.txt" and proceed to the next step.

```
VEP_input_blast <- read.table(file = "VEP_input_blast.txt")

colnames(VEP_input_blast) <- c("chromosome", "position", "reference_allele", "alternate_allele", "blast_

VEP_input_blast$chromosome <- as.factor(VEP_input_blast$chromosome)
VEP_input_blast$reference_allele <- as.factor(VEP_input_blast$reference_allele)
VEP_input_blast$alternate_allele <- as.factor(VEP_input_blast$alternate_allele)
VEP_input_blast$GCF_010183535.1_RefSeq_ID <- as.factor(VEP_input_blast$GCF_010183535.1_RefSeq_ID)

VEP_input_blast$GCF_010183535.1_RefSeq_chromosome <-
  ifelse(VEP_input_blast$GCF_010183535.1_RefSeq_ID == "NC_071328.1", "I",
  ifelse(VEP_input_blast$GCF_010183535.1_RefSeq_ID == "NC_071329.1", "II",
  ifelse(VEP_input_blast$GCF_010183535.1_RefSeq_ID == "NC_071330.1", "III",
  ifelse(VEP_input_blast$GCF_010183535.1_RefSeq_ID == "NC_071331.1", "IV",
  ifelse(VEP_input_blast$GCF_010183535.1_RefSeq_ID == "NC_071332.1", "V", "X")))))
```

```
VEP_input_blast$GCF_010183535.1_RefSeq_chromosome <- as.factor(VEP_input_blast$GCF_010183535.1_RefSeq_ch

VEP_input_blast$GCF_010183535.1_RefSeq_position <- VEP_input_blast$GCF_010183535.1_RefSeq_beg+200

VEP_input_blast$variable = "."

VEP_input_blast_submit <- VEP_input_blast[,c(10,11,12,3,4,12,12,12)]

write.table(VEP_input_blast_submit, file = "~/My Drive/Nematode/Nematode_ASM164373v4_data/VEP_input_blas
```

Load data: VEP output

load VEP output and generate a data frame for analysis (**VEP_analysis**).

```
VEP_output <- read.table(file = "VEP_output.txt")
colnames(VEP_output) <- VEP_output[c(1),]
VEP_output <- VEP_output[c(2:104),]

VEP_output_temp <- subset(singleton_raw, duplicated(singleton_raw$singleton_ID) & singleton_raw$filter =

VEP_output_temp$ID <- paste(VEP_output_temp$chromosome, VEP_output_temp$position, sep = ":")
VEP_input_blast_temp <- VEP_input_blast
VEP_input_blast_temp$ID <- paste(VEP_input_blast_temp$chromosome, VEP_input_blast_temp$position, sep =
VEP_input_blast_temp$Location_temp <- paste(VEP_input_blast_temp$GCF_010183535.1_RefSeq_chromosome, VEP
VEP_input_blast_temp$Location <- paste(VEP_input_blast_temp$Location_temp, VEP_input_blast_temp$GCF_010

VEP_analysis <- merge(VEP_input_blast_temp, VEP_output_temp[,c(1,2,5)], by = "ID", all = T)
VEP_analysis <- merge(VEP_analysis, VEP_output, by = "Location", all = T)

VEP_analysis <- subset(VEP_analysis, select = c("Regime", "Sample", "Location", "ID", "Consequence", "I
```

VEP analysis

Impact

Prepare input file

```
VEP_analysis$Location <- as.factor(VEP_analysis$Location)
VEP_analysis$ID <- as.factor(VEP_analysis$ID)
VEP_analysis$Consequence <- as.factor(VEP_analysis$Consequence)
VEP_analysis$IMPACT <- as.factor(VEP_analysis$IMPACT)

VEP_analysis_IMPACT <-
  aggregate(VEP_analysis$Sample,
            by = list(VEP_analysis$Regime, VEP_analysis$Sample, VEP_analysis$IMPACT),
            FUN = NROW)
colnames(VEP_analysis_IMPACT) <- c("Regime", "Sample", "IMPACT", "Number")

VEP_analysis_IMPACT_df_temp1 <- aggregate(VEP_analysis_IMPACT$Number,
```

```
                                          by = list(VEP_analysis_IMPACT$Regime),
                                          FUN = sum)
colnames(VEP_analysis_IMPACT_df_temp1) <- c("Regime", "Total")

VEP_analysis_IMPACT_df_temp2 <- aggregate(VEP_analysis_IMPACT$Number,
                                        by = list(VEP_analysis_IMPACT$Regime,
                                                  VEP_analysis_IMPACT$IMPACT),
                                        FUN = sum)
colnames(VEP_analysis_IMPACT_df_temp2) <- c("Regime", "IMPACT", "Number")


VEP_analysis_IMPACT_df <- merge(VEP_analysis_IMPACT_df_temp1,
                                    VEP_analysis_IMPACT_df_temp2,
                                    by = "Regime", all = T)

VEP_analysis_IMPACT_df$perc <- VEP_analysis_IMPACT_df$Number/VEP_analysis_IMPACT_df$Total
```

chisq.test

```
VEP_analysis_IMPACT_df
```

```
##    Regime Total   IMPACT Number       perc
## 1      T1    45     HIGH      2 0.04444444
## 2      T1    45      LOW      1 0.02222222
## 3      T1    45 MODIFIER     39 0.86666667
## 4      T1    45 MODERATE      3 0.06666667
## 5      T2    11      LOW      1 0.09090909
## 6      T2    11 MODERATE      2 0.18181818
## 7      T2    11 MODIFIER      8 0.72727273
## 8      T5    47 MODERATE      4 0.08510638
## 9      T5    47      LOW      6 0.12765957
## 10     T5    47 MODIFIER     37 0.78723404
```

```
# update the numbers below

VEP_analysis_IMPACT_chisq.test <- cbind(c(2,0,0),c(1,1,6),c(3,2,4),c(39,8,37))
dimnames(VEP_analysis_IMPACT_chisq.test) <-
  list(Regime = c("T1", "T2", "T5"),
       IMPACT = c("HIGH", "LOW", "MODERATE", "MODIFIER"))

VEP_analysis_IMPACT_chisq.test <- as.data.frame(VEP_analysis_IMPACT_chisq.test)


chisq.test(VEP_analysis_IMPACT_chisq.test,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  VEP_analysis_IMPACT_chisq.test
## X-squared = 7.538, df = NA, p-value = 0.2508
```

```
chisq.test(VEP_analysis_IMPACT_chisq.test)$observed
```

```
## Warning in chisq.test(VEP_analysis_IMPACT_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    HIGH LOW MODERATE MODIFIER
## T1    2   1        3       39
## T2    0   1        2        8
## T5    0   6        4       37
```

```
chisq.test(VEP_analysis_IMPACT_chisq.test)$expected
```

```
## Warning in chisq.test(VEP_analysis_IMPACT_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##         HIGH       LOW MODERATE  MODIFIER
## T1 0.8737864 3.4951456 3.932039 36.699029
## T2 0.2135922 0.8543689 0.961165  8.970874
## T5 0.9126214 3.6504854 4.106796 38.330097
```

```
chisq.test(VEP_analysis_IMPACT_chisq.test$HIGH,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_IMPACT_chisq.test$HIGH
## X-squared = 4, df = NA, p-value = 0.3404
```

```
chisq.test(VEP_analysis_IMPACT_chisq.test$LOW,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_IMPACT_chisq.test$LOW
## X-squared = 6.25, df = NA, p-value = 0.05949
```

```
chisq.test(VEP_analysis_IMPACT_chisq.test$MODERATE,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_IMPACT_chisq.test$MODERATE
## X-squared = 0.66667, df = NA, p-value = 0.9184
```

```
chisq.test(VEP_analysis_IMPACT_chisq.test$MODIFIER,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_IMPACT_chisq.test$MODIFIER
## X-squared = 21.5, df = NA, p-value = 2e-04
```

Consequence

Prepare input file

```
VEP_analysis$Location <- as.factor(VEP_analysis$Location)
VEP_analysis$ID <- as.factor(VEP_analysis$ID)
VEP_analysis$Consequence <- as.factor(VEP_analysis$Consequence)
VEP_analysis$IMPACT <- as.factor(VEP_analysis$IMPACT)

VEP_analysis_Consequence <-
  aggregate(VEP_analysis$Sample,
            by = list(VEP_analysis$Regime, VEP_analysis$Sample, VEP_analysis$Consequence),
            FUN = NROW)
colnames(VEP_analysis_Consequence) <- c("Regime", "Sample", "Consequence", "Number")

VEP_analysis_Consequence_df_temp1 <- aggregate(VEP_analysis_Consequence$Number,
                                               by = list(VEP_analysis_Consequence$Regime),
                                               FUN = sum)
colnames(VEP_analysis_Consequence_df_temp1) <- c("Regime", "Total")

VEP_analysis_Consequence_df_temp2 <- aggregate(VEP_analysis_Consequence$Number,
                                           by = list(VEP_analysis_Consequence$Regime,
                                                     VEP_analysis_Consequence$Consequence),
                                           FUN = sum)
colnames(VEP_analysis_Consequence_df_temp2) <- c("Regime", "Consequence", "Number")


VEP_analysis_Consequence_df <- merge(VEP_analysis_Consequence_df_temp1,
                                     VEP_analysis_Consequence_df_temp2,
                                     by = "Regime", all = T)

VEP_analysis_Consequence_df$perc <- VEP_analysis_Consequence_df$Number/VEP_analysis_Consequence_df$Total
```

chisq.test

```
VEP_analysis_Consequence_df
```

```
##    Regime Total             Consequence Number       perc
## 1      T1    45 downstream_gene_variant     18 0.40000000
## 2      T1    45    splice_donor_variant      1 0.02222222
## 3      T1    45        missense_variant      3 0.06666667
```

```
## 4      T1    45          intron_variant      2 0.04444444
## 5      T1    45             stop_gained      1 0.02222222
## 6      T1    45      synonymous_variant      1 0.02222222
## 7      T1    45   upstream_gene_variant     19 0.42222222
## 8      T2    11 downstream_gene_variant      4 0.36363636
## 9      T2    11          intron_variant      1 0.09090909
## 10     T2    11      synonymous_variant      1 0.09090909
## 11     T2    11        missense_variant      2 0.18181818
## 12     T2    11   upstream_gene_variant      3 0.27272727
## 13     T5    47 downstream_gene_variant     18 0.38297872
## 14     T5    47          intron_variant      2 0.04255319
## 15     T5    47        missense_variant      4 0.08510638
## 16     T5    47      synonymous_variant      6 0.12765957
## 17     T5    47      intergenic_variant      1 0.02127660
## 18     T5    47   upstream_gene_variant     16 0.34042553
```

```
# update the numbers below

VEP_analysis_Consequence_chisq.test <- cbind(c(18,4,18),c(0,0,1),c(2,1,2),
                                             c(3,2,4),c(1,0,0),c(1,0,0),
                                             c(1,1,6),c(19,3,16))
dimnames(VEP_analysis_Consequence_chisq.test) <-
  list(Regime = c("T1", "T2", "T5"),
       Consequence = c("downstream_gene_variant", "intergenic_variant", "intron_variant",
                       "missense_variant", "splice_donor_variant" ,"stop_gained",
                       "synonymous_variant","upstream_gene_variant"))

VEP_analysis_Consequence_chisq.test <- as.data.frame(VEP_analysis_Consequence_chisq.test)

chisq.test(VEP_analysis_Consequence_chisq.test,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  VEP_analysis_Consequence_chisq.test
## X-squared = 9.6534, df = NA, p-value = 0.7739
```

```
chisq.test(VEP_analysis_Consequence_chisq.test)$observed
```

```
## Warning in chisq.test(VEP_analysis_Consequence_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    downstream_gene_variant intergenic_variant intron_variant missense_variant
## T1                      18                  0              2                3
## T2                       4                  0              1                2
## T5                      18                  1              2                4
##    splice_donor_variant stop_gained synonymous_variant upstream_gene_variant
## T1                    1           1                  1                   19
## T2                    0           0                  1                    3
## T5                    0           0                  6                   16
```

```
chisq.test(VEP_analysis_Consequence_chisq.test)$expected
```

```
## Warning in chisq.test(VEP_analysis_Consequence_chisq.test): Chi-squared
## approximation may be incorrect
```

```
##    downstream_gene_variant intergenic_variant intron_variant missense_variant
## T1               17.475728          0.4368932      2.1844660         3.932039
## T2                4.271845          0.1067961      0.5339806         0.961165
## T5               18.252427          0.4563107      2.2815534         4.106796
##    splice_donor_variant stop_gained synonymous_variant upstream_gene_variant
## T1            0.4368932   0.4368932          3.4951456             16.601942
## T2            0.1067961   0.1067961          0.8543689              4.058252
## T5            0.4563107   0.4563107          3.6504854             17.339806
```

```
chisq.test(VEP_analysis_Consequence_chisq.test$downstream_gene_variant,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_Consequence_chisq.test$downstream_gene_variant
## X-squared = 9.8, df = NA, p-value = 0.008199
```

```
chisq.test(VEP_analysis_Consequence_chisq.test$intergenic_variant,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_Consequence_chisq.test$intergenic_variant
## X-squared = 2, df = NA, p-value = 1
```

```
chisq.test(VEP_analysis_Consequence_chisq.test$intron_variant,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_Consequence_chisq.test$intron_variant
## X-squared = 0.4, df = NA, p-value = 1
```

```
chisq.test(VEP_analysis_Consequence_chisq.test$missense_variant,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
```

```
##
## data:  VEP_analysis_Consequence_chisq.test$missense_variant
## X-squared = 0.66667, df = NA, p-value = 0.9132
```

```
chisq.test(VEP_analysis_Consequence_chisq.test$splice_donor_variant,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_Consequence_chisq.test$splice_donor_variant
## X-squared = 2, df = NA, p-value = 1
```

```
chisq.test(VEP_analysis_Consequence_chisq.test$stop_gained,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_Consequence_chisq.test$stop_gained
## X-squared = 2, df = NA, p-value = 1
```

```
chisq.test(VEP_analysis_Consequence_chisq.test$synonymous_variant,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_Consequence_chisq.test$synonymous_variant
## X-squared = 6.25, df = NA, p-value = 0.05879
```

```
chisq.test(VEP_analysis_Consequence_chisq.test$upstream_gene_variant,
           simulate.p.value = TRUE, B = 10000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 10000 replicates)
##
## data:  VEP_analysis_Consequence_chisq.test$upstream_gene_variant
## X-squared = 11.421, df = NA, p-value = 0.0031
```

Figure 1

```r
line_survival_df <- subset(line_survival, line_survival$Regime == "T1 (set 1)" | line_survival$Regime ==

line_survival_df <- line_survival_df[line_survival_df$Generation < 4,]

line_survival_df$Regime <- c("Young T1", "Young T1", "Young T1", "Young T1",
                             "Peak T2", "Peak T2", "Peak T2", "Peak T2",
                             "Old T5", "Old T5", "Old T5", "Old T5")
line_survival_df$Generation_factor <- c("M0", "M1", "M2", "M3", "M0", "M1", "M2", "M3", "M0", "M1", "M2
line_survival_df$survival <- line_survival_df$Number/60


ggplot(line_survival_df, aes(x = Generation_factor, y = survival, group = Regime, color = Regime))+
  geom_line()+
  geom_point(size = 3)+
  scale_color_manual(values = Palette)+
  scale_y_continuous(name = expression("% of lines alive"),
                     breaks = c(0,0.2,0.4,0.6,0.8,1.0),
                     labels = c("0","20","40","60","80","100"),
                     limits = c(0,1)) +
  scale_x_discrete(name = "Generation")+
  theme_classic()+
  theme(axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12),
        legend.justification = c(0, 0), legend.position = c(0, 0),
        legend.background = element_rect(fill='transparent'))+
  guides(color = guide_legend(reverse=TRUE))
```

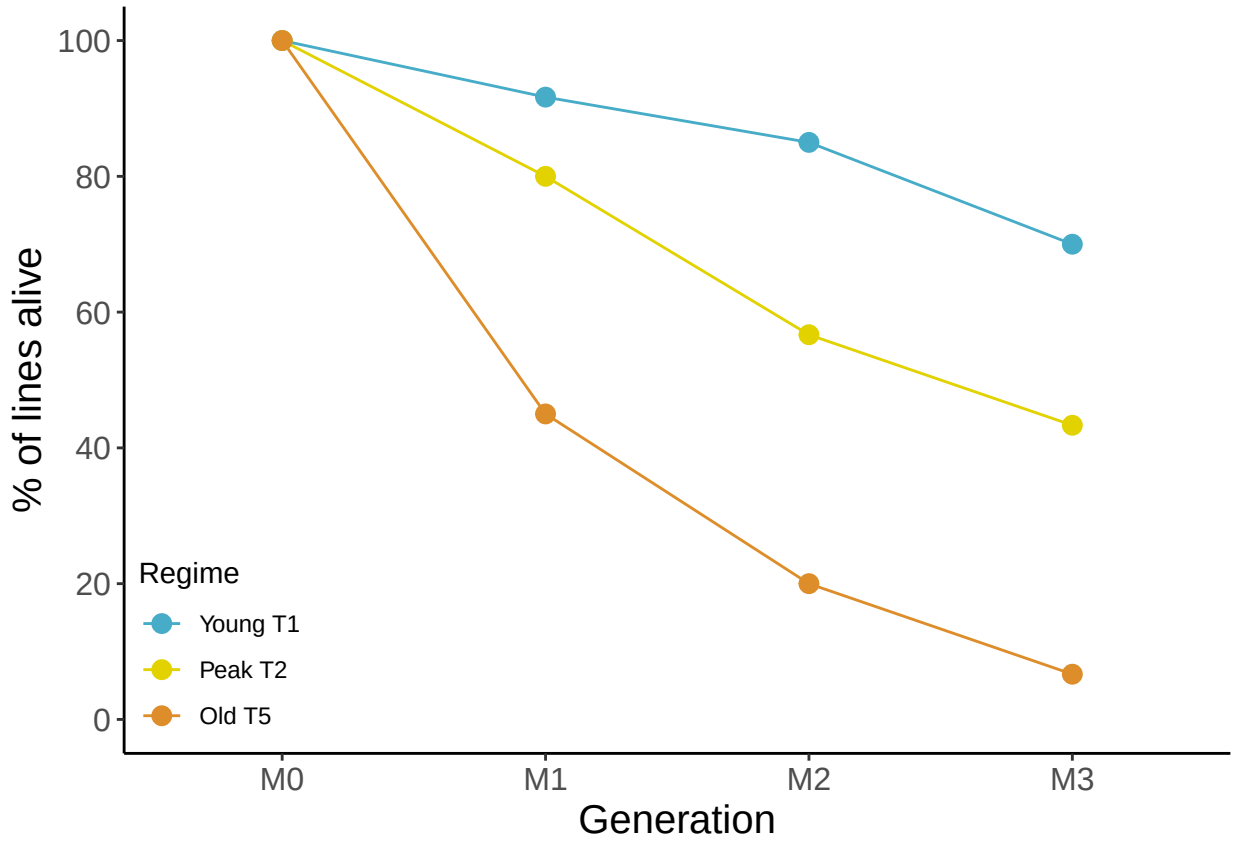

Figure 2

```
Figure_2 <- cbind(aggregate(df$mu_overlap, list(df$Regime), FUN = mean),
                  aggregate(df$mu_overlap, list(df$Regime), FUN = se)[,2])
colnames(Figure_2) <- c("Regime","Mean_mu","SE_mu")


ggplot()+
  geom_bar(data = Figure_2,
       aes(x = Regime, y = Mean_mu,  fill = Regime),
       stat="identity", position=position_dodge())+
  geom_errorbar(data = Figure_2,
                aes(x = Regime, y = Mean_mu, ymin = Mean_mu-SE_mu, ymax = Mean_mu+SE_mu),
                stat="identity", position=position_dodge(), width=0.4)+
  geom_jitter(data = df, aes(x = Regime, y = mu_overlap))+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous(name = expression("Mutation rate (X10"^{-8}~")"),
                     breaks = c(-0.1E-8, 0, 0.5E-8, 1E-8, 1.5E-8, 2E-8, 2.5E-8, 3E-8, 3.5E-8, 4E-8),
                     labels = c("","0","0.5","1","1.5","2","2.5","3","3.5","4"),
                     limits = c(-0.1E-8,4E-8)) +
  theme_classic()+
  theme(axis.title.x = element_text(size = 15),
```

```
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12),
        legend.position = "none")
```

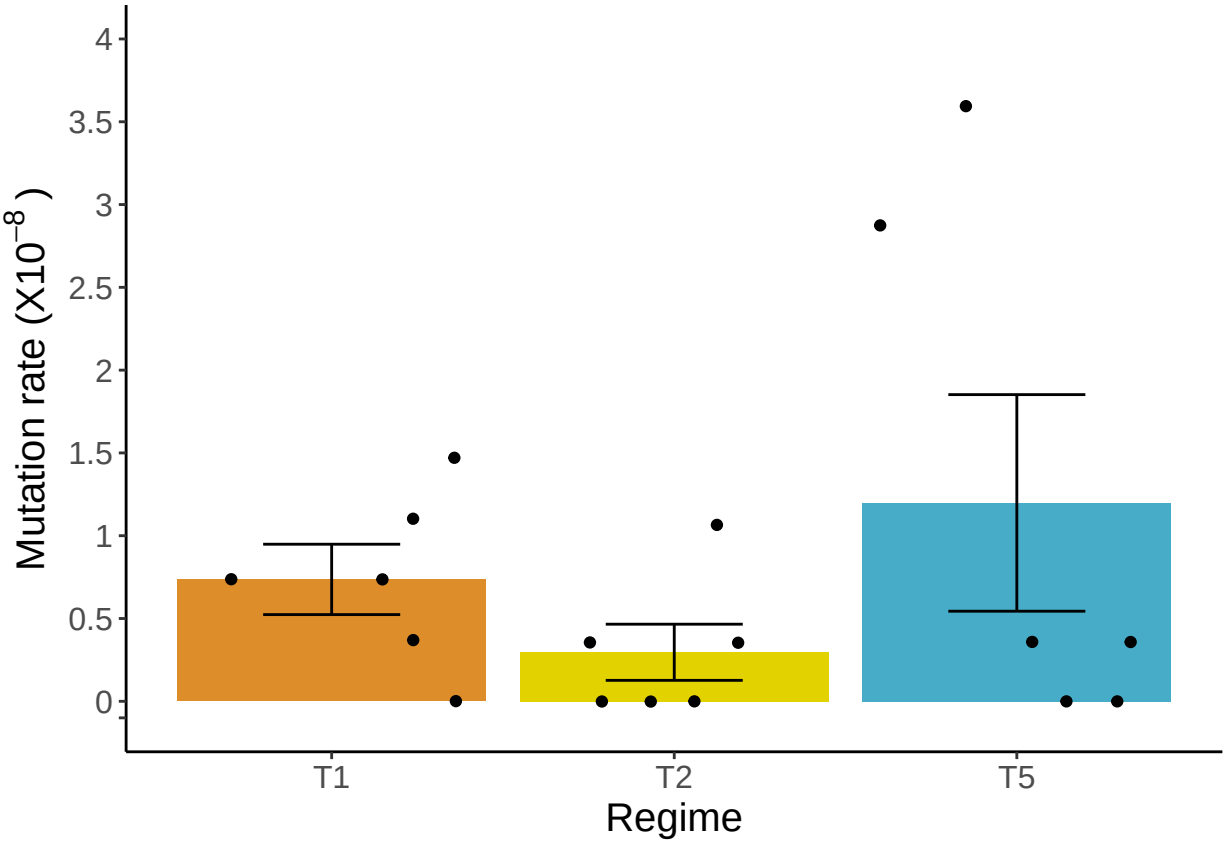

Figure 3

```
Figure_3_temp <- cbind(aggregate(TsTv_overlap_df$perc,
                                 by = list(TsTv_overlap_df$Regime,
                                           TsTv_overlap_df$TsTv), FUN = mean),
                       aggregate(TsTv_overlap_df$perc,
                                 by = list(TsTv_overlap_df$Regime,
                                           TsTv_overlap_df$TsTv), FUN = se)[,3])
colnames(Figure_3_temp) <- c("Regime", "TsTv", "mean", "se")

Regime <- c("T1", "T2", "T5")
TsTv <- c("Transition", "Transversion", "Transition", "Transversion", "Transition", "Transversion")

temp <- data.frame(rep(Regime, each = 2), TsTv)
colnames(temp) <- c("Regime", "TsTv")

Figure_3 <- merge(temp, Figure_3_temp,
```

```
                  by.x = c("Regime", "TsTv"), by.y = c("Regime", "TsTv"), all = T)


Figure_3[is.na(Figure_3)] <- 0

Figure_3$Regime <- as.factor(Figure_3$Regime)
Figure_3$TsTv <- as.factor(Figure_3$TsTv)


ggplot(Figure_3, aes(x = TsTv, y = mean, fill = Regime))+
  geom_bar(stat="identity", position=position_dodge())+
  geom_errorbar(aes(ymin=mean-se, ymax=mean+se), width=.2,
                position=position_dodge(.9))+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous((name = "% of mutations"),
                     breaks = c(0,0.2,0.4,0.6,0.8,1),
                     labels = c("0","20","40","60","80","100"),
                     limits = c(0,1)) +
  scale_x_discrete(name = "Category of mutations")+
  theme_classic()+
  theme(axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12))
```

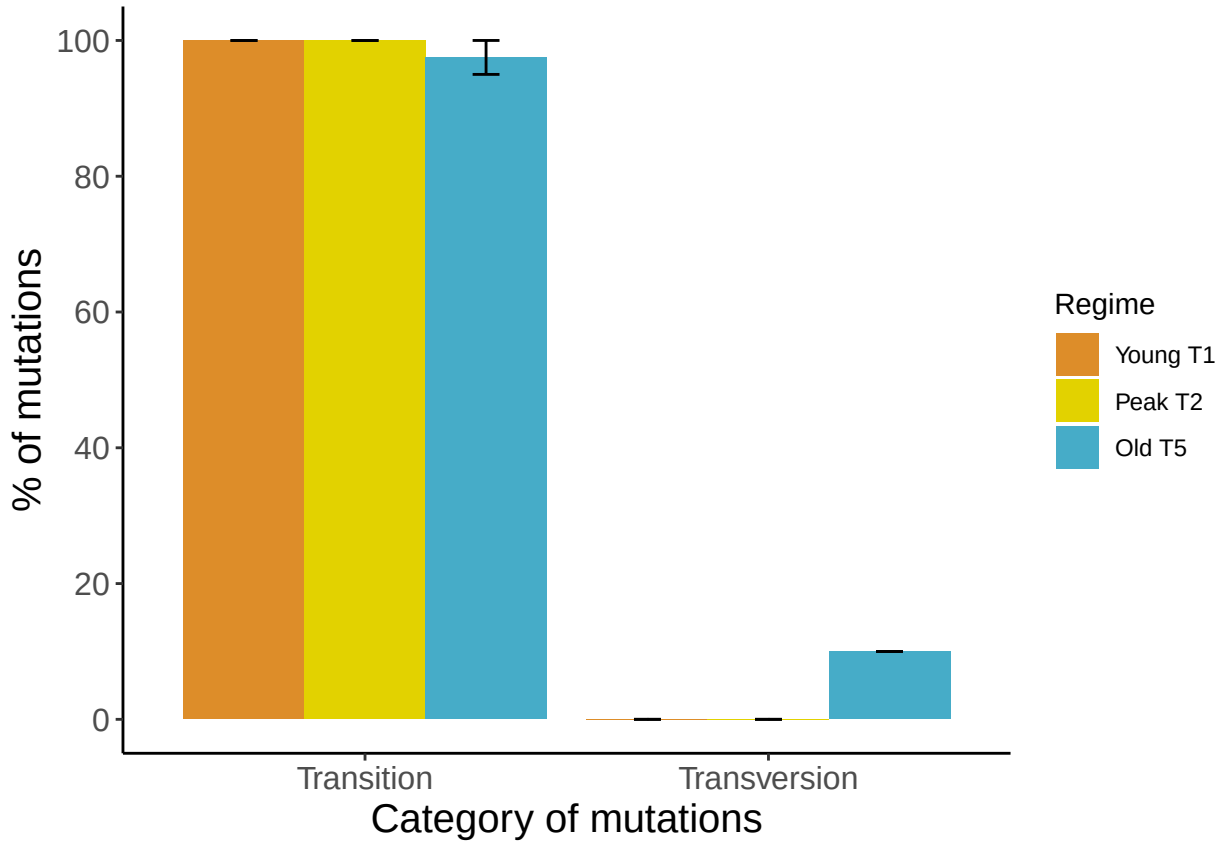

Figure 4

```
missing = list(Regime = c("T1","T2",
                          "T2","T5",
                          "T2","T5"),
               Total = c(0,0,0,0,0,0),
               Consequence = c("intergenic_variant","intergenic_variant",
                               "splice_donor_variant","splice_donor_variant",
                               "stop_gained","stop_gained"),
               Number = c(0,0,0,0,0,0),
               perc = c(0,0,0,0,0,0))

VEP_analysis_Consequence_plot <- rbind(VEP_analysis_Consequence_df,missing)

Figure_4a <-
ggplot(VEP_analysis_Consequence_plot, aes(x = Consequence, y = perc, fill = Regime) )+
  geom_bar(stat="identity", position=position_dodge())+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous((name = "% of mutations"),
                     breaks = c(0,0.2,0.4,0.6,0.8,1.0),
                     labels = c("0","20","40","60","80","100"),
                     limits = c(0,1)) +
  scale_x_discrete(name = "Consequence")+
  theme_classic()+
  theme(axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 8),
        axis.text.y = element_text(size = 12),
        legend.position = "none")+
  labs(tag = "A)")
```

```
missing = list(Regime = c("T2","T5"),
               Total = c(0,0),
               IMPACT = c("HIGH","HIGH"),
               Number = c(0,0),
               perc = c(0,0))

VEP_analysis_IMPACT_plot <- rbind(VEP_analysis_IMPACT_df,missing)

Figure_4b <-
ggplot(VEP_analysis_IMPACT_plot, aes(x = IMPACT, y = perc, fill = Regime) )+
  geom_bar(stat="identity", position=position_dodge())+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous((name = "% of mutations"),
                     breaks = c(0,0.2,0.4,0.6,0.8,1),
                     labels = c("0","20","40","60","80","100"),
                     limits = c(0,1)) +
  scale_x_discrete(name = "Impact")+
  theme_classic()+
  theme(axis.title.x = element_text(size = 15),
```

```
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 10),
        axis.text.y = element_text(size = 12))+
  labs(tag = "B)")
```

```
grid.arrange(Figure_4a, Figure_4b, ncol = 2)
```

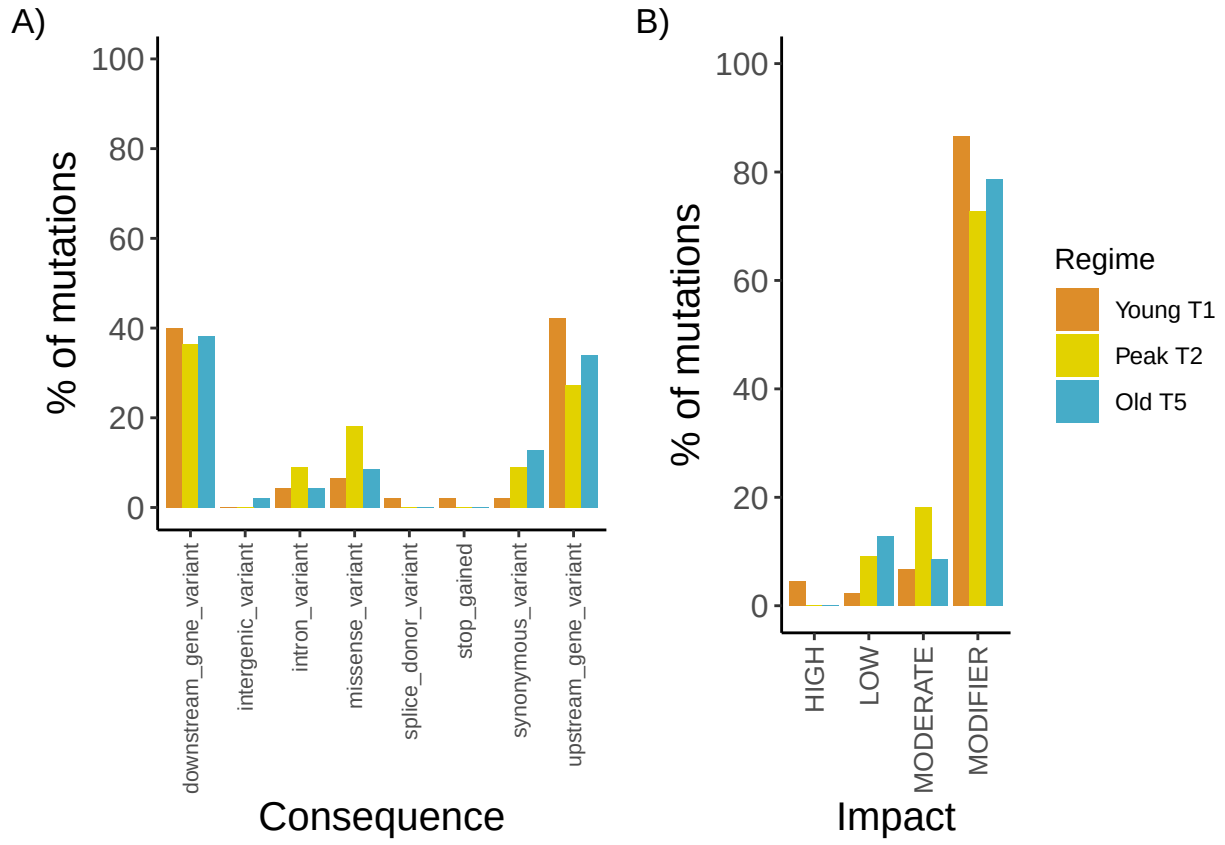

Figure S2

```
line_survival_1 <- line_survival[which(line_survival$Regime != "T5 (set 1)"),]
line_survival_2 <- line_survival[which(line_survival$Regime == "T5 (set 1)"),]

line_survival_1$survival <- line_survival_1$Number/60
line_survival_2$survival <- line_survival_2$Number/89

line_survival_all <- rbind(line_survival_1, line_survival_2)

str(line_survival_all)
```

```
## 'data.frame':    65 obs. of  4 variables:
##  $ Generation: int  0 1 2 3 4 5 6 7 8 9 ...
##  $ Regime    : Factor w/ 8 levels "T1 (set 1)","T1 (set 2)",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
##  $ Number    : int  60 55 51 42 36 30 20 13 6 3 ...
##  $ survival  : num  1 0.917 0.85 0.7 0.6 ...
```

```
ggplot(line_survival_all, aes(x = factor(Generation), y = Number, group = Regime, color = Regime))+
  geom_line()+
  geom_point(aes(shape = Regime), size = 3)+
  scale_color_manual(
    values = c("#DD8D29","#DD8D29",
               "#E2D200",
               "#9986A5", "#F4B5BD",
               "#46ACC8", "#46ACC8",
               "#9C964A"))+
  scale_shape_manual(values = c(16,1,16,16,16,16,1,16)) +
  scale_y_continuous(name = "Number of lines alive") +
  scale_x_discrete(name = "Generation",
                   breaks = c(0,1,2,3,4,5,6,7,8,9,10),
                   labels = c("M0", "M1", "M2", "M3", "M4", "M5", "M6", "M7", "M8", "M9", "M10"))+
  theme_classic()+
  theme(axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12),
        #legend.justification = c(1, 1), legend.position = c(1, 1),
        legend.background = element_rect(fill='transparent'))
```

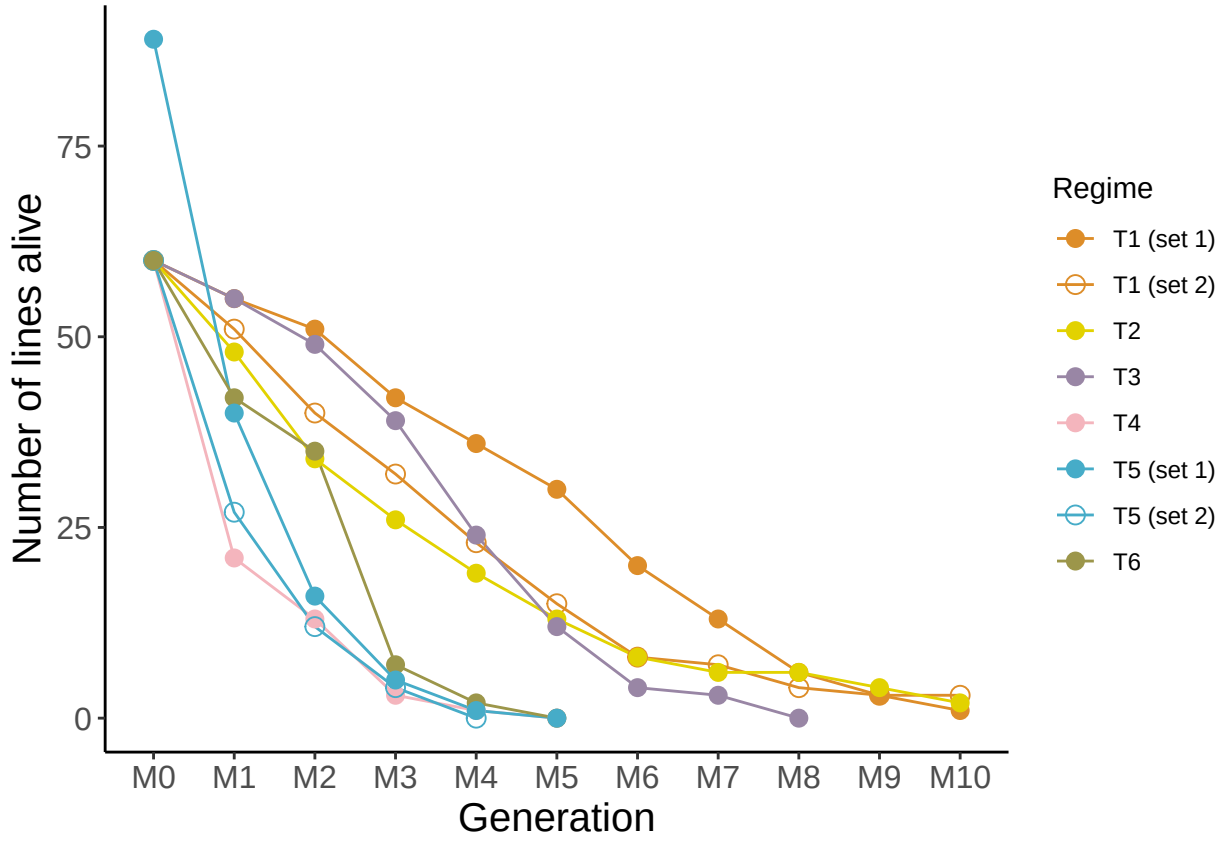

Figure S5

```
Figure_S5a <-
ggplot(df, aes(x = Sample, y = consensus, fill = Regime))+
  geom_bar(stat="identity", position=position_dodge())+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous((name = "Number of accepted mutations"),
                     limits = c(0,70)) +
  scale_x_discrete(name = "MA line ID")+
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 12),
        axis.text.y = element_text(size = 12))+
  labs(title = "Consensus approach",
       tag = "A)")
```

```
Figure_S5b <-
ggplot(df, aes(x = Sample, y = probabilistic, fill = Regime))+
  geom_bar(stat="identity", position=position_dodge())+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous((name = "Number of accepted mutations"),
                     limits = c(0,70)) +
  scale_x_discrete(name = "MA line ID")+
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 12),
        axis.text.y = element_text(size = 12))+
  labs(title = "Probabilistic approach",
       tag = "B)")
```

```
Figure_S5c<-
ggplot(df, aes(x = Sample, y = overlap, fill = Regime))+
  geom_bar(stat="identity", position=position_dodge())+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous((name = "Number of accepted mutations"),
                     limits = c(0,70)) +
  scale_x_discrete(name = "MA line ID")+
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 12),
        axis.text.y = element_text(size = 12))+
  labs(title = "Overlap",
       tag = "C)")
```

```
grid.arrange(Figure_S5a, Figure_S5b, Figure_S5c, ncol = 1)
```

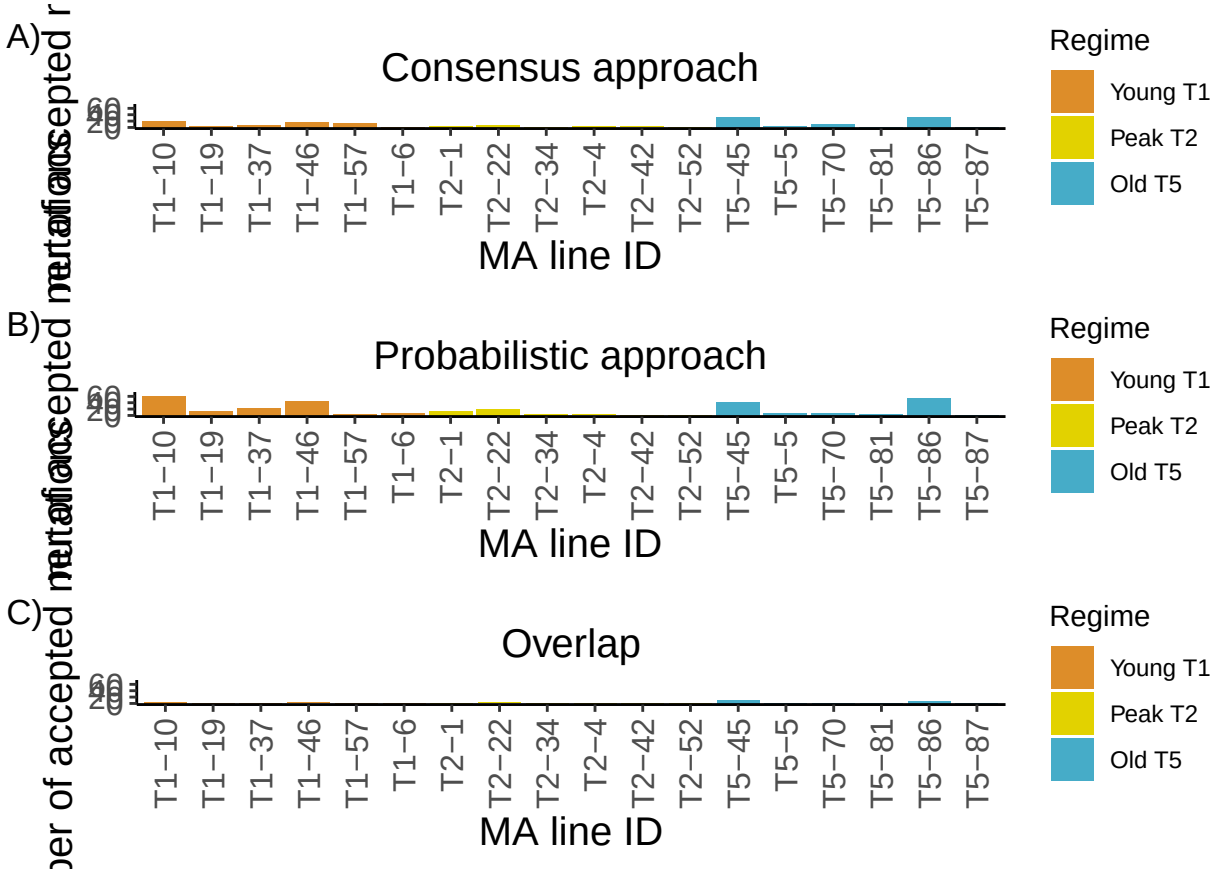

Figure S6

Consensus approach

```
Figure_7a_temp <- cbind(aggregate(TsTv_consensus_df$perc,
                            by = list(TsTv_consensus_df$Regime,
                                      TsTv_consensus_df$TsTv), FUN = mean),
                        aggregate(TsTv_consensus_df$perc,
                            by = list(TsTv_consensus_df$Regime,
                                      TsTv_consensus_df$TsTv), FUN = se)[,3])
colnames(Figure_7a_temp) <- c("Regime", "TsTv", "mean", "se")

Regime <- c("T1", "T2", "T5")
TsTv <- c("Transition", "Transversion", "Transition", "Transversion", "Transition", "Transversion")

temp <- data.frame(rep(Regime, each = 2), TsTv)
colnames(temp) <- c("Regime", "TsTv")

Figure_7a <- merge(temp, Figure_7a_temp,
                   by.x = c("Regime", "TsTv"), by.y = c("Regime", "TsTv"), all = T)
```

```
Figure_7a[is.na(Figure_7a)] <- 0

Figure_7a$Regime <- as.factor(Figure_7a$Regime)
Figure_7a$TsTv <- as.factor(Figure_7a$TsTv)

Figure_S7a <-
ggplot(Figure_7a, aes(x = TsTv, y = mean, fill = Regime))+
  geom_bar(stat="identity", position=position_dodge())+
  geom_errorbar(aes(ymin=mean-se, ymax=mean+se), width=.2,
                position=position_dodge(.9))+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous((name = "% of mutations"),
                     breaks = c(0,0.2,0.4,0.6,0.8,1),
                     labels = c("0","20","40","60","80","100"),
                     limits = c(0,1)) +
  scale_x_discrete(name = "Category of mutations")+
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12))+
  labs(title = "Consensus approach",
       tag = "A)")
```

Probabilistic approach

```
Figure_7a_temp <- cbind(aggregate(TsTv_probabilistic_df$perc,
                            by = list(TsTv_probabilistic_df$Regime,
                                      TsTv_probabilistic_df$TsTv), FUN = mean),
                        aggregate(TsTv_probabilistic_df$perc,
                            by = list(TsTv_probabilistic_df$Regime,
                                      TsTv_probabilistic_df$TsTv), FUN = se)[,3])
colnames(Figure_7a_temp) <- c("Regime", "TsTv", "mean", "se")

Regime <- c("T1", "T2", "T5")
TsTv <- c("Transition", "Transversion", "Transition", "Transversion", "Transition", "Transversion")

temp <- data.frame(rep(Regime, each = 2), TsTv)
colnames(temp) <- c("Regime", "TsTv")

Figure_7a <- merge(temp, Figure_7a_temp,
                   by.x = c("Regime", "TsTv"), by.y = c("Regime", "TsTv"), all = T)


Figure_7a[is.na(Figure_7a)] <- 0

Figure_7a$Regime <- as.factor(Figure_7a$Regime)
Figure_7a$TsTv <- as.factor(Figure_7a$TsTv)

Figure_S7b <-
ggplot(Figure_7a, aes(x = TsTv, y = mean, fill = Regime))+
```

```
  geom_bar(stat="identity", position=position_dodge())+
  geom_errorbar(aes(ymin=mean-se, ymax=mean+se), width=.2,
                position=position_dodge(.9))+
  scale_fill_manual(values = Palette,
                    labels = c("Young T1", "Peak T2", "Old T5"))+
  scale_y_continuous((name = "% of mutations"),
                     breaks = c(0,0.2,0.4,0.6,0.8,1),
                     labels = c("0","20","40","60","80","100"),
                     limits = c(0,1)) +
  scale_x_discrete(name = "Category of mutations")+
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.title.x = element_text(size = 15),
        axis.title.y = element_text(size = 15),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12))+
  labs(title = "Probabilistic approach",
       tag = "B)")
```

```
grid.arrange(Figure_S7a, Figure_S7b, ncol = 1)
```

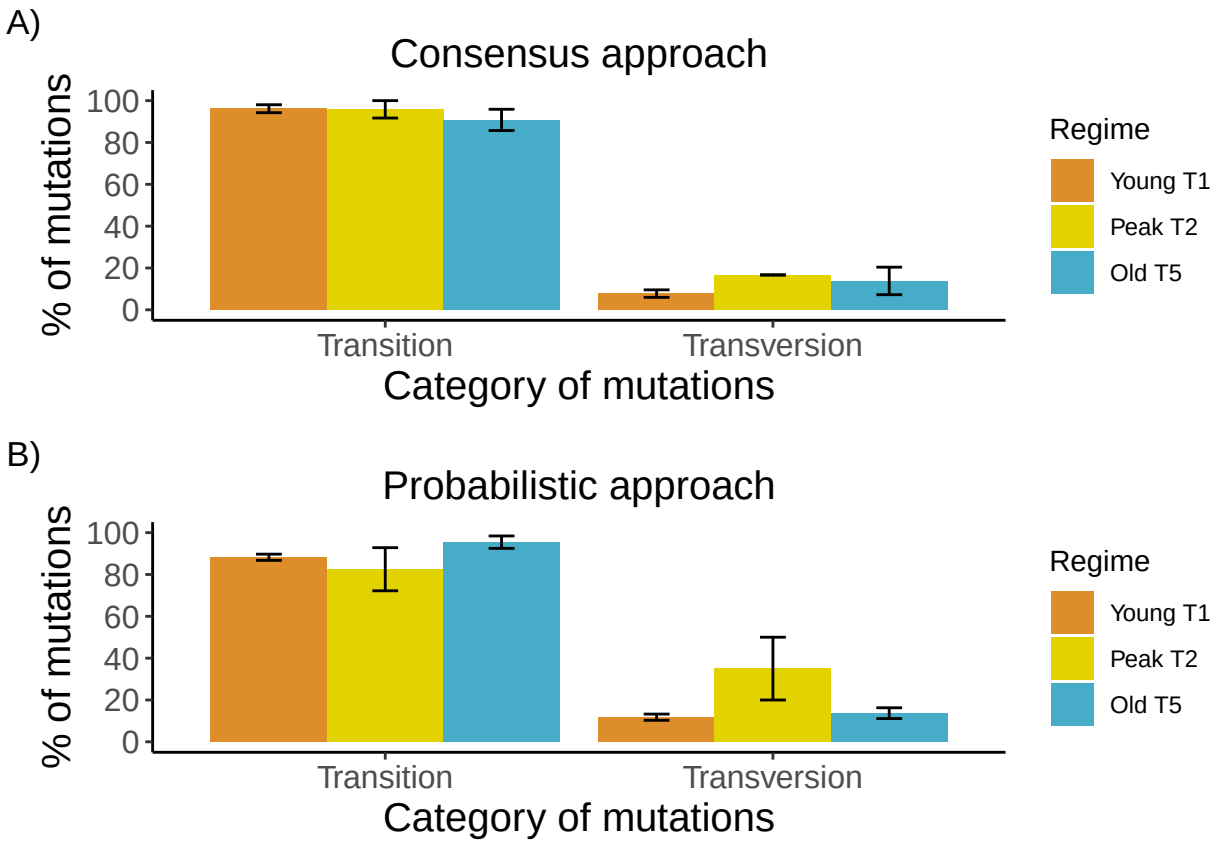

Table 1

mean number of mutations

```
format(cbind(aggregate(df$overlap, by = list(df$Regime), FUN = mean), aggregate(df$overlap, by = list(df
       format = "e", digits = 2)
```

```
##   Group.1    x aggregate(df$overlap, by = list(df$Regime), FUN = se)[, 2]
## 1      T1 2.00                                                      0.58
## 2      T2 0.83                                                      0.48
## 3      T5 3.33                                                      1.82
```

mean number of callable sites

```
aggregate(df$Callability, by = list(df$Regime), FUN = mean)
```

```
##   Group.1        x
## 1      T1 90566554
## 2      T2 93817854
## 3      T5 92742405
```

mean mutation rate

```
format(cbind(aggregate(df$mu_overlap, list(df$Regime), FUN = mean),
             aggregate(df$mu_overlap, list(df$Regime), FUN = se)[,2]),
       format = "e", digits = 3)
```

```
##   Group.1        x aggregate(df$mu_overlap, list(df$Regime), FUN = se)[, 2]
## 1      T1 7.36e-09                                                2.12e-09
## 2      T2 2.96e-09                                                1.70e-09
## 3      T5 1.20e-08                                                6.54e-09
```

mean number of transitions and transversions

```
TsTv <- c("Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion

temp <- data.frame(rep(df[,c(1,2)], each = 2), TsTv)[,c(1,3,5)]
colnames(temp) <- c("Sample", "Regime", "TsTv")

TsTv_overlap_df2 <- merge(temp, TsTv_overlap_df,
                  by.x = c("Sample", "TsTv"), by.y = c("Sample", "TsTv"), all = T)[,c(1,2,3,5,6)]
colnames(TsTv_overlap_df2) <- c("Sample", "TsTv", "Regime", "number", "total")
TsTv_overlap_df2[is.na(TsTv_overlap_df2)] <- 0


format(cbind(
  aggregate(TsTv_overlap_df2$number,
            by = list(TsTv_overlap_df2$Regime, TsTv_overlap_df2$TsTv),
            FUN = mean),
  aggregate(TsTv_overlap_df2$number,
```

```
            by = list(TsTv_overlap_df2$Regime, TsTv_overlap_df2$TsTv),
            FUN = se)[,3]),
        format = "e", digits = 2)
```

```
##   Group.1      Group.2    x
## 1      T1   Transition 2.00
## 2      T2   Transition 0.83
## 3      T5   Transition 3.17
## 4      T1 Transversion 0.00
## 5      T2 Transversion 0.00
## 6      T5 Transversion 0.17
##   aggregate(TsTv_overlap_df2$number, by = list(TsTv_overlap_df2$Regime,
## 1                                                                  0.58
## 2                                                                  0.48
## 3                                                                  1.70
## 4                                                                  0.00
## 5                                                                  0.00
## 6                                                                  0.17
```

Table S1

```
Table_S1 <- merge(coverage, callability, by = "Sample", all = T)
Table_S1$Percentage <- Table_S1$Callability/124491291

mean(Table_S1$Percentage)
```

```
## [1] 0.7420246
```

```
aggregate(Table_S1$Percentage, by = list(Table_S1$Regime), FUN = mean)
```

```
##   Group.1         x
## 1      T1 0.7274931
## 2      T2 0.7536098
## 3      T5 0.7449710
```

Table S2

```
singleton_count_all <- spread(aggregate(singleton_raw$Sample, by = list(singleton_raw$Sample, singleton_
colnames(singleton_count_all) <- c("Sample", "consensus_all", "probabilistic_all")

singleton_count_PASS <- spread(singleton_PASS, approach, total_count)
colnames(singleton_count_PASS) <- c("Regime", "Sample", "consensus_PASS", "probabilistic_PASS")
singleton_count_PASS[is.na(singleton_count_PASS)] <- 0

Table_S2 <- merge(singleton_count_PASS, singleton_count_all, by = "Sample", all = T)
Table_S2$consensus_perc <- percent(Table_S2$consensus_PASS/Table_S2$consensus_all)
```

```
Table_S2$probabilistic_perc <- percent(Table_S2$probabilistic_PASS/Table_S2$probabilistic_all)

Table_S2 <- merge(Table_S2, singleton_PASS_overlap[,c(2,3)], by = "Sample", all = T)
Table_S2[is.na(Table_S2)] <- 0

Table_S2 <- Table_S2[,c(2,1,5,3,7,6,4,8,9)]

sum(Table_S2$consensus_all)
```

```
## [1] 940
```

```
sum(Table_S2$consensus_PASS)
```

```
## [1] 175
```

```
sum(Table_S2$probabilistic_all)
```

```
## [1] 343
```

```
sum(Table_S2$probabilistic_PASS)
```

```
## [1] 319
```

```
sum(Table_S2$overlap)
```

```
## [1] 37
```

```
percent(mean(Table_S2$consensus_PASS/Table_S2$consensus_all))
```

```
## [1] "15.01%"
```

```
percent(mean(Table_S2$probabilistic_PASS/Table_S2$probabilistic_all))
```

```
## [1] "89.36%"
```

```
aggregate(Table_S2$overlap, by = list(Table_S2$Regime), FUN = sum)
```

```
##   Group.1  x
## 1      T1 12
## 2      T2  5
## 3      T5 20
```

```
cbind(aggregate(Table_S2$overlap, by = list(Table_S2$Regime), FUN = mean),
aggregate(Table_S2$overlap, by = list(Table_S2$Regime), FUN = se)[,2])
```

```
##   Group.1         x
## 1      T1 2.0000000
## 2      T2 0.8333333
## 3      T5 3.3333333
##   aggregate(Table_S2$overlap, by = list(Table_S2$Regime), FUN = se)[,
## 1                                                                 0.5773503
## 2                                                                 0.4772607
## 3                                                                 1.8196459
```

Table S3

mean number of mutations

```
format(cbind(aggregate(df$consensus, by = list(df$Regime), FUN = mean), aggregate(df$consensus, by = lis
       format = "e", digits = 3)
```

```
##   Group.1     x aggregate(df$consensus, by = list(df$Regime), FUN = se)[, 2]
## 1      T1 11.50                                                       2.87
## 2      T2  3.83                                                       1.38
## 3      T5 13.83                                                       6.35
```

```
format(cbind(aggregate(df$probabilistic, by = list(df$Regime), FUN = mean), aggregate(df$probabilistic,
       format = "e", digits = 3)
```

```
##   Group.1     x aggregate(df$probabilistic, by = list(df$Regime), FUN = se)[,
## 1      T1 25.67                                                         9.07
## 2      T2  7.17                                                         3.11
## 3      T5 20.33                                                         9.10
```

mean number of callable sites

```
aggregate(df$Callability, by = list(df$Regime), FUN = mean)
```

```
##   Group.1        x
## 1      T1 90566554
## 2      T2 93817854
## 3      T5 92742405
```

mean mutation rate

```
format(cbind(aggregate(df$mu_consensus, list(df$Regime), FUN = mean),
             aggregate(df$mu_consensus, list(df$Regime), FUN = se)[,2]),
       format = "e", digits = 4)
```

```
##   Group.1         x aggregate(df$mu_consensus, list(df$Regime), FUN = se)[, 2]
## 1      T1 4.233e-08                                                 1.057e-08
## 2      T2 1.362e-08                                                 4.890e-09
## 3      T5 4.972e-08                                                 2.283e-08
```

```
format(cbind(aggregate(df$mu_probabilistic, list(df$Regime), FUN = mean),
             aggregate(df$mu_probabilistic, list(df$Regime), FUN = se)[,2]),
       format = "e", digits = 4)
```

```
##   Group.1         x
## 1      T1 9.447e-08
## 2      T2 2.546e-08
## 3      T5 7.308e-08
##   aggregate(df$mu_probabilistic, list(df$Regime), FUN = se)[, 2]
## 1                                                     3.339e-08
## 2                                                     1.106e-08
## 3                                                     3.270e-08
```

mean number of transitions and transversions

```
TsTv <- c("Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion

temp <- data.frame(rep(df[,c(1,2)], each = 2), TsTv)[,c(1,3,5)]
colnames(temp) <- c("Sample", "Regime", "TsTv")

TsTv_consensus_df2 <- merge(temp, TsTv_consensus_df,
                  by.x = c("Sample", "TsTv"), by.y = c("Sample", "TsTv"), all = T)[,c(1,2,3,5,6)]
colnames(TsTv_consensus_df2) <- c("Sample", "TsTv", "Regime", "number", "total")
TsTv_consensus_df2[is.na(TsTv_consensus_df2)] <- 0


format(cbind(
  aggregate(TsTv_consensus_df2$number,
            by = list(TsTv_consensus_df2$Regime, TsTv_consensus_df2$TsTv),
            FUN = mean),
  aggregate(TsTv_consensus_df2$number,
            by = list(TsTv_consensus_df2$Regime, TsTv_consensus_df2$TsTv),
            FUN = se)[,3]),
        format = "e", digits = 2)
```

```
##   Group.1      Group.2     x
## 1      T1   Transition 10.83
## 2      T2   Transition  3.67
## 3      T5   Transition 12.83
## 4      T1 Transversion  0.67
## 5      T2 Transversion  0.17
## 6      T5 Transversion  1.00
##   aggregate(TsTv_consensus_df2$number, by = list(TsTv_consensus_df2$Regime,
## 1                                                                      2.60
## 2                                                                      1.33
## 3                                                                      6.02
## 4                                                                      0.33
## 5                                                                      0.17
## 6                                                                      0.37
```

```
aggregate(TsTv_consensus$Sample,
            by = list(TsTv_consensus$Sample, TsTv_consensus$TsTv),
            FUN = NROW)
```

```
##    Group.1      Group.2  x
## 1    T1-10   Transition 19
## 2    T1-19   Transition  6
## 3    T1-37   Transition  9
## 4    T1-46   Transition 16
## 5    T1-57   Transition 13
## 6     T1-6   Transition  2
```

```
## 7     T2-1   Transition  5
## 8    T2-22   Transition  8
## 9     T2-4   Transition  6
## 10   T2-42   Transition  3
## 11   T5-45   Transition 32
## 12    T5-5   Transition  2
## 13   T5-70   Transition  9
## 14   T5-81   Transition  1
## 15   T5-86   Transition 31
## 16   T5-87   Transition  2
## 17   T1-10 Transversion  1
## 18   T1-46 Transversion  2
## 19   T1-57 Transversion  1
## 20    T2-1 Transversion  1
## 21   T5-45 Transversion  2
## 22    T5-5 Transversion  1
## 23   T5-70 Transversion  1
## 24   T5-86 Transversion  2
```

```
TsTv <- c("Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transition", "Transition", "Transition", "Transition", "Transition", "Transition",
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion
          "Transversion", "Transversion", "Transversion", "Transversion", "Transversion", "Transversion

temp <- data.frame(rep(df[,c(1,2)], each = 2), TsTv)[,c(1,3,5)]
colnames(temp) <- c("Sample", "Regime", "TsTv")

TsTv_probabilistic_TsTv_df <- merge(temp, TsTv_probabilistic_df,
                  by.x = c("Sample", "TsTv"), by.y = c("Sample", "TsTv"), all = T)[,c(1,2,3,5,6)]
colnames(TsTv_probabilistic_TsTv_df) <- c("Sample", "TsTv", "Regime", "number", "total")
TsTv_probabilistic_TsTv_df[is.na(TsTv_probabilistic_TsTv_df)] <- 0


TsTv_probabilistic_TsTv_df[is.na(TsTv_probabilistic_TsTv_df)] <- 0

format(cbind(
  aggregate(TsTv_probabilistic_TsTv_df$number,
            by = list(TsTv_probabilistic_TsTv_df$Regime, TsTv_probabilistic_TsTv_df$TsTv),
            FUN = mean),
  aggregate(TsTv_probabilistic_TsTv_df$number,
            by = list(TsTv_probabilistic_TsTv_df$Regime, TsTv_probabilistic_TsTv_df$TsTv),
            FUN = se)[,3]),
        format = "e", digits = 2)
```

```
##   Group.1      Group.2     x
## 1      T1   Transition 22.83
## 2      T2   Transition  6.50
## 3      T5   Transition 18.17
## 4      T1 Transversion  2.83
## 5      T2 Transversion  0.67
## 6      T5 Transversion  2.17
##   aggregate(TsTv_probabilistic_TsTv_df$number, by = list(TsTv_probabilistic_TsTv_df$Regime,
```

```
## 1                                                                    8.11
## 2                                                                    3.12
## 3                                                                    7.79
## 4                                                                    1.01
## 5                                                                    0.33
## 6                                                                    1.38
```

Table S4

```
format(df[,c(1,7,8,9)], format = "e", digits = 4)[,c(1,3,2,4)]
```

```
##    Sample mu_consensus mu_probabilistic mu_overlap
## 1   T1-10    7.361e-08        2.245e-07  1.472e-08
## 2   T1-19    2.208e-08        5.153e-08  7.361e-09
## 3   T1-37    3.312e-08        8.097e-08  7.361e-09
## 4   T1-46    6.625e-08        1.619e-07  1.104e-08
## 5   T1-57    5.153e-08        2.208e-08  3.681e-09
## 6    T1-6    7.361e-09        2.576e-08  0.000e+00
## 7    T2-1    2.132e-08        4.619e-08  3.553e-09
## 8   T2-22    2.842e-08        7.106e-08  1.066e-08
## 9   T2-34    0.000e+00        1.421e-08  0.000e+00
## 10   T2-4    2.132e-08        1.066e-08  3.553e-09
## 11  T2-42    1.066e-08        7.106e-09  0.000e+00
## 12  T2-52    0.000e+00        3.553e-09  0.000e+00
## 13  T5-45    1.222e-07        1.545e-07  3.594e-08
## 14   T5-5    1.078e-08        3.235e-08  3.594e-09
## 15  T5-70    3.594e-08        3.235e-08  0.000e+00
## 16  T5-81    3.594e-09        2.157e-08  0.000e+00
## 17  T5-86    1.186e-07        1.941e-07  2.875e-08
## 18  T5-87    7.188e-09        3.594e-09  3.594e-09
```

Table S5

```
reshape(TsTv_consensus_df[,c(1,3,4)], idvar = "Sample", timevar = "TsTv", direction = "wide")
```

```
##    Sample number.Transition number.Transversion
## 1   T1-10                19                   1
## 3   T1-19                 6                  NA
## 4   T1-37                 9                  NA
## 5   T1-46                16                   2
## 7   T1-57                13                   1
## 9    T1-6                 2                  NA
## 10   T2-1                 5                   1
## 12  T2-22                 8                  NA
## 13   T2-4                 6                  NA
## 14  T2-42                 3                  NA
## 15  T5-45                32                   2
## 17   T5-5                 2                   1
```

```
## 19  T5-70                 9                   1
## 21  T5-81                 1                  NA
## 22  T5-86                31                   2
## 24  T5-87                 2                  NA
```

```
reshape(TsTv_probabilistic_df[,c(1,3,4)], idvar = "Sample", timevar = "TsTv", direction = "wide")
```

```
##    Sample number.Transition number.Transversion
## 1   T1-10                55                   6
## 3   T1-19                13                   1
## 5   T1-37                20                   2
## 7   T1-46                38                   6
## 9   T1-57                 5                   1
## 11   T1-6                 6                   1
## 13   T2-1                13                  NA
## 14  T2-22                19                   1
## 16  T2-34                 2                   2
## 18   T2-4                 3                  NA
## 19  T2-42                 1                   1
## 21  T2-52                 1                  NA
## 22  T5-45                36                   7
## 24   T5-5                 9                  NA
## 25  T5-70                 9                  NA
## 26  T5-81                 6                  NA
## 27  T5-86                48                   6
## 29  T5-87                 1                  NA
```

```
reshape(TsTv_overlap_df[,c(1,3,4)], idvar = "Sample", timevar = "TsTv", direction = "wide")
```

```
##    Sample number.Transition number.Transversion
## 1   T1-10                 4                  NA
## 2   T1-19                 2                  NA
## 3   T1-37                 2                  NA
## 4   T1-46                 3                  NA
## 5   T1-57                 1                  NA
## 6    T2-1                 1                  NA
## 7   T2-22                 3                  NA
## 8    T2-4                 1                  NA
## 9   T5-45                 9                   1
## 11   T5-5                 1                  NA
## 12  T5-86                 8                  NA
## 13  T5-87                 1                  NA
```

Supplementary Results

Overview

general description

```
aggregate(singleton_raw$Sample, by = list(singleton_raw$approach), FUN = NROW)
```

```
##         Group.1   x
## 1     consensus 940
## 2 probabilistic 343
```

```
aggregate(singleton_raw$Sample, by = list(singleton_raw$Regime, singleton_raw$approach), FUN = NROW)
```

```
##   Group.1       Group.2   x
## 1      T1     consensus 333
## 2      T2     consensus 243
## 3      T5     consensus 364
## 4      T1 probabilistic 167
## 5      T2 probabilistic  48
## 6      T5 probabilistic 128
```

```
cbind(aggregate(singleton_count_PASS$consensus_PASS, by = list(singleton_count_PASS$Regime), FUN = sum)
```

```
##   Group.1  x
## 1      T1 69
## 2      T2 23
## 3      T5 83
##   aggregate(singleton_count_PASS$probabilistic_PASS, by = list(singleton_count_PASS$Regime),
## 1                                                                                        154
## 2                                                                                         43
## 3                                                                                        122
```

```
cbind(aggregate(singleton_count_PASS$consensus_PASS, by = list(singleton_count_PASS$Regime), FUN = mean)
```

```
##   Group.1         x
## 1      T1 11.500000
## 2      T2  3.833333
## 3      T5 13.833333
##   aggregate(singleton_count_PASS$consensus_PASS, by = list(singleton_count_PASS$Regime),
## 1                                                                                2.872281
## 2                                                                                1.376388
## 3                                                                                6.353040
```

```
cbind(aggregate(singleton_count_PASS$probabilistic_PASS, by = list(singleton_count_PASS$Regime), FUN = m
```

```
##   Group.1         x
## 1      T1 25.666667
## 2      T2  7.166667
## 3      T5 20.333333
##   aggregate(singleton_count_PASS$probabilistic_PASS, by = list(singleton_count_PASS$Regime),
## 1                                                                                    9.072547
## 2                                                                                    3.113590
## 3                                                                                    9.098229
```

derive acceptance rate

```
percent(min(Table_S2$consensus_PASS/Table_S2$consensus_all))
```

```
## [1] "0.00%"
```

```
percent(max(Table_S2$consensus_PASS/Table_S2$consensus_all))
```

```
## [1] "35.71%"
```

```
percent(min(Table_S2$probabilistic_PASS/Table_S2$probabilistic_all))
```

```
## [1] "50.00%"
```

```
percent(max(Table_S2$probabilistic_PASS/Table_S2$probabilistic_all))
```

```
## [1] "100.00%"
```

```
cbind(aggregate(Table_S2$consensus_PASS/Table_S2$consensus_all, by = list(Table_S2$Regime), FUN = mean)
aggregate(Table_S2$consensus_PASS/Table_S2$consensus_all, by = list(Table_S2$Regime), FUN = se)[,2])
```

```
##   Group.1          x
## 1      T1 0.19501369
## 2      T2 0.08636219
## 3      T5 0.16889165
##   aggregate(Table_S2$consensus_PASS/Table_S2$consensus_all, by = list(Table_S2$Regime),
## 1                                                                                0.03794389
## 2                                                                                0.03126183
## 3                                                                                0.05246826
```

```
cbind(aggregate(Table_S2$probabilistic_PASS/Table_S2$probabilistic_all, by = list(Table_S2$Regime), FUN
aggregate(Table_S2$probabilistic_PASS/Table_S2$probabilistic_all, by = list(Table_S2$Regime), FUN = se)
```

```
##   Group.1         x
## 1      T1 0.9472048
## 2      T2 0.8520833
## 3      T5 0.8813629
##   aggregate(Table_S2$probabilistic_PASS/Table_S2$probabilistic_all,
## 1                                                       0.01764090
## 2                                                       0.08039076
## 3                                                       0.07973201
```

```
#Table_S2$consensus_perc <- percent(Table_S2$consensus_PASS/Table_S2$consensus_all)
#Table_S2$probabilistic_perc <- percent(Table_S2$probabilistic_PASS/Table_S2$probabilistic_all)
```

test whether acceptance rate differs between regimes: consensus approach

```
Table_S2_chisq_test <- subset(Table_S2, select = c("Regime"))
Table_S2_chisq_test$consensus_perc_2 <- Table_S2$consensus_PASS/Table_S2$consensus_all
Table_S2_chisq_test$probabilistic_perc_2 <- Table_S2$probabilistic_PASS/Table_S2$probabilistic_all

Table_S2_chisq_test_consensus <- aggregate(Table_S2_chisq_test$consensus_perc_2, by = list(Table_S2_chi

chisq.test(Table_S2_chisq_test_consensus$x)
```

```
## Warning in chisq.test(Table_S2_chisq_test_consensus$x): Chi-squared
## approximation may be incorrect

##
##  Chi-squared test for given probabilities
##
## data:  Table_S2_chisq_test_consensus$x
## X-squared = 0.04286, df = 2, p-value = 0.9788
```

test whether acceptance rate differs between regimes: probabilistic approach

```
Table_S2_chisq_test <- subset(Table_S2, select = c("Regime"))
Table_S2_chisq_test$probabilistic_perc_2 <- Table_S2$probabilistic_PASS/Table_S2$probabilistic_all
Table_S2_chisq_test$probabilistic_perc_2 <- Table_S2$probabilistic_PASS/Table_S2$probabilistic_all

Table_S2_chisq_test_probabilistic <- aggregate(Table_S2_chisq_test$probabilistic_perc_2, by = list(Table

chisq.test(Table_S2_chisq_test_probabilistic$x)
```

```
## Warning in chisq.test(Table_S2_chisq_test_probabilistic$x): Chi-squared
## approximation may be incorrect

##
##  Chi-squared test for given probabilities
##
## data:  Table_S2_chisq_test_probabilistic$x
## X-squared = 0.0053123, df = 2, p-value = 0.9973
```

correlation

```
cor.test(df$Callability, df$consensus, method = 'pearson')
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$Callability and df$consensus
## t = -1.0259, df = 16, p-value = 0.3202
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.6409648  0.2470902
## sample estimates:
##       cor
## -0.2484396
```

```
cor.test(df$Callability, df$probabilistic, method = 'pearson')
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$Callability and df$probabilistic
## t = -1.6248, df = 16, p-value = 0.1237
```

```
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.7171973  0.1098267
## sample estimates:
##       cor
## -0.3763402
```

parental-age-at-reproduction and mutation rates

```
min(df$mu_consensus)
```

```
## [1] 0
```

```
max(df$mu_consensus)
```

```
## [1] 1.222023e-07
```

```
min(df$mu_probabilistic)
```

```
## [1] 3.552984e-09
```

```
max(df$mu_probabilistic)
```

```
## [1] 2.245126e-07
```

```
temp <- df[,c(1,2,7,8,9)]

cbind(aggregate(temp$mu_consensus, by = list(temp$Regime), FUN = mean),
aggregate(temp$mu_consensus, by = list(temp$Regime), FUN = se)[,2])
```

```
##   Group.1           x
## 1      T1 4.232615e-08
## 2      T2 1.361977e-08
## 3      T5 4.971956e-08
##   aggregate(temp$mu_consensus, by = list(temp$Regime), FUN = se)[,
## 1                                                     1.057153e-08
## 2                                                     4.890285e-09
## 3                                                     2.283400e-08
```

```
cbind(aggregate(temp$mu_probabilistic, by = list(temp$Regime), FUN = mean),
aggregate(temp$mu_probabilistic, by = list(temp$Regime), FUN = se)[,2])
```

```
##   Group.1           x
## 1      T1 9.446705e-08
## 2      T2 2.546305e-08
## 3      T5 7.308176e-08
##   aggregate(temp$mu_probabilistic, by = list(temp$Regime), FUN = se)[,
## 1                                                       3.339182e-08
## 2                                                       1.106254e-08
## 3                                                       3.270072e-08
```