Inference of global PCR error rates from minor subvariants of UMI-tagged template consensus sequence

Mikhail Shugay

November 21, 2016

Control experiment

Some auxiliary functions & setup

```
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(ggplot2)
library(RColorBrewer)
library(stringr)
library(nloptr)
library(scales)

select <- dplyr::select

mtypes <- data.frame(mutation.fromto = c("A>C","A>G","A>T","C>A","C>G","C>T","G>A",
                                         "G>C","G>T","T>A","T>C","T>G"),
                     mutation.type = c("A>C,T>G","A>G,T>C","A>T,T>A","C>A,G>T",
                                       "C>G,G>C","C>T,G>A","C>T,G>A","C>G,G>C",
                                       "C>A,G>T","A>T,T>A","A>G,T>C","A>C,T>G"))

read_variant_table <- function(file_name, pos_filter = function(x) T, count_filter = function(x) T) {
  .df <- read.table(file_name, header=T, sep="\t", stringsAsFactors = F)
  .df <- subset(.df, !grepl("^[DI]", mutation) & coverage > 100 & freq < 0.45)

  .df$mutation.fromto <- unlist(lapply(str_split(.df$mutation, ":"), function(x) x[2]))
  .df$mutation.pos <- as.integer(unlist(lapply(str_split(.df$mutation, ":"),
                                          function(x) str_sub(x[1], 2, nchar(x[1])))))
  .df <- merge(.df, mtypes)
  .df %>%
    filter(pos_filter(mutation.pos) & count_filter(count)) %>%
    select(mutation.pos, mutation.type, count, coverage, freq, error.rate)
}
```

Load data from 2 independent experiments with library preparation performed using 10 different PCR assays: a known synthetic template sequence that was UMI-tagged and amplified using a set of different polymerases. At this stage a single event is a combination of substituted nucleotide, its substitution, position in template, sample (polymerase) and project (replica). Samples are obtained by grouping all events by substitution type which is one of 6 from and to nucleotide combinations that can be observed when not knowing the exact strand at which a given error has happened. Note that the latter is done because in most practical applications this information is hard to obtain. Data was pre-processed with different quality and UMI size threshold.

```r
df <- data.frame()

samples <- c("encyclo", "kappa-hf-taq", "phusion", "sd-hs", "snp-detect",
                   "taq-hs", "tersus", "tersus-snp-buff", "truseq", "velox")

for (proj in c("73", "82")) {
  for (q in c(20, 25, 30)) {
     for (m in c(8, 16, 32)) {
        for (sample in samples) {
           file_name <- paste("data/", q, "_", m, "_polerr", proj, ".", sample,
                             ".variant.caller.txt", sep = "")

           # ignore positions with no errors
           # this includes primers/barcodes
           .df <- read_variant_table(file_name,
                                     count_filter=function(x) x>0)
           .df$q <- q
           .df$m <- m
           .df$proj <- proj
           .df$sample <- sample
           df <- rbind(df, .df)
        }
     }
  }
}
```

Minor-based error model (MBEM)

MBEM computes PCR error rate for n cycles as following

$$\phi^{(n)} = \pi\rho + (1-\pi)\,\rho'$$

where π is the fraction of MIGs with an error occupying a substantial fraction of reads, ρ is the overall fraction of erroneous reads in these MIGs and ρ' is the hidden fraction of errors in molecules for which no error was detected.

For computing π we count MIGs that contain a fraction of errors m/l that can not be explained by sequencing error rate p_{seq} under selected quality threshold Q alone, that is

$$F_{binom}\left(M \geq m | l, p_{seq}\right) < p_0$$
$$p_{seq} = \frac{1}{3} 10^{-Q/10}$$

where l is the number of reads in MIG (MIG size) and m is the count of a given minor variant. Note that we also compute false-discovery rate FDR under the specified P-value threshold p_0 and substitute π with $\pi * (1 - FDR)$.

The hidden fraction of errors is estimated from Poisson distribution as $\pi = 1 - \exp(-\rho' \hat{l})$ where $\hat{l}$ is the geometric mean of MIG sizes. Finally,

$$\phi^{(n)} = \pi\rho - \frac{(1-\pi)\log(1-\pi)}{\hat{l}}$$

Error rate per PCR cycle is computed as

$$\phi = \frac{\phi^{(n)}(1+\lambda)}{n\lambda}$$

where $1+\lambda$ is PCR efficiency.

In order to apply this model to datasets where only errors introduced at first PCR cycle (in contrast to designs when errors can happen during UMI attachment) that become dominant due to PCR stochastics can become major variants, one need to multiply ϕ by error propagation probability $P_{prop} = (1-\lambda)\lambda^2$.

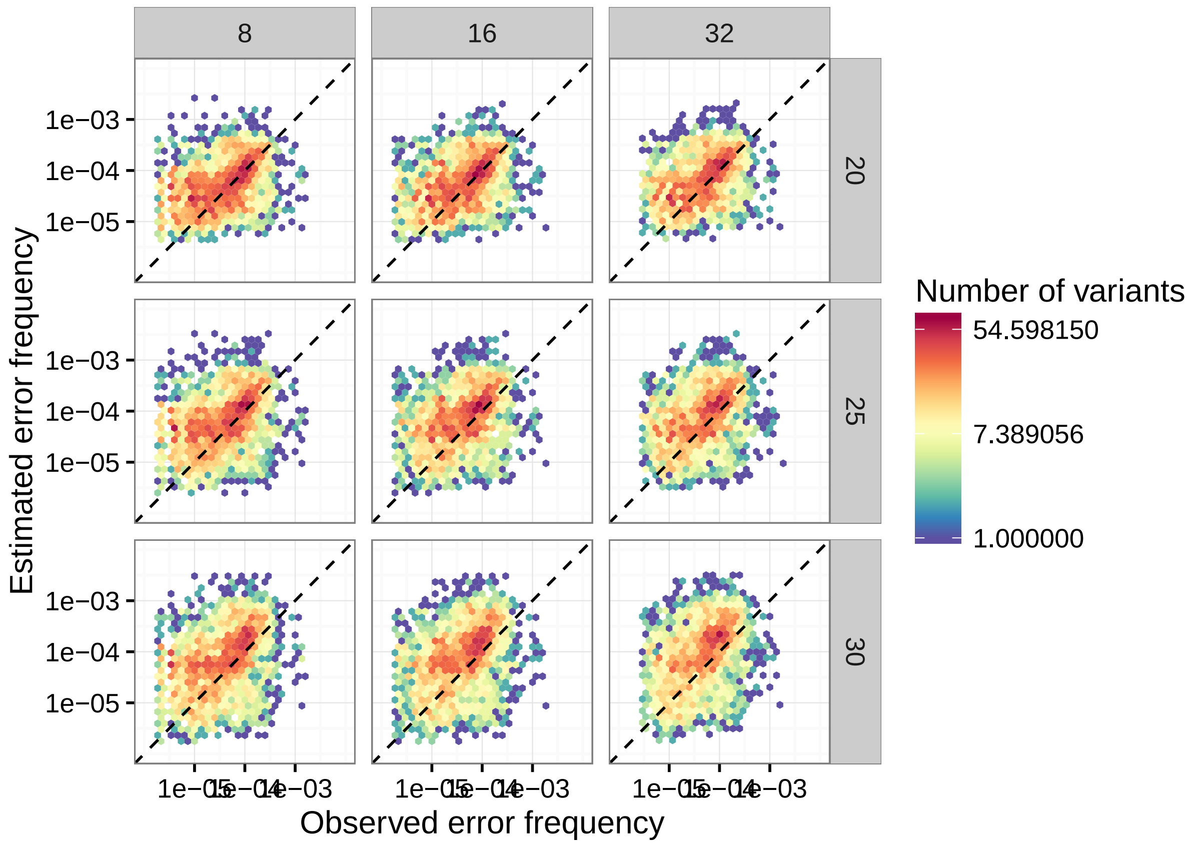
Note that this is not the case for this dataset: same polymerase was used during linear PCR to attach UMI and at subsequent PCR cycles. Thus per cycle error rate computed from errors within MIGs should nicely reflect observed error rate.

Model benchmark

In this section we compare MBEM predictions against observed PCR error rate in control experiment. First we check robustness w.r.t. pre-processing settings such as UMI coverage (aka MIG size) threshold `m` and quality threshold for minor variants `q`

```r
rf <- colorRampPalette(rev(brewer.pal(11, 'Spectral')))
r <- rf(32)

ggplot(df, aes(x=freq, y=error.rate)) +
  stat_binhex() +
  geom_abline(intercept = 0, slope=1, linetype="dashed") +
  scale_x_log10("Observed error frequency", limits=c(1e-6, 1e-2), breaks=c(1e-5,1e-4,1e-3)) +
  scale_y_log10("Estimated error frequency", limits=c(1e-6, 1e-2), breaks=c(1e-5,1e-4,1e-3)) +
  facet_grid(q~m) +
  scale_fill_gradientn("Number of variants", colors=r, trans="log") +
  theme_bw()
```



ANOVA for log-transformed data and nonparametric correlation

```
a <- aov(log(freq) ~ I(log(error.rate)) + q + m + proj, df)
summary(a)
```

```
##                       Df Sum Sq Mean Sq F value  Pr(>F)
## I(log(error.rate))     1   8169    8169 5674.83 < 2e-16 ***
## q                      1     40      40   27.87 1.3e-07 ***
## m                      1    183     183  127.36 < 2e-16 ***
## proj                   1      4       4    2.90  0.0886 .
## Residuals          37279  53663       1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
for (mm in unique(df$m)) {
  for (qq in unique(df$q)) {
    print(paste("m =", mm , "q =", qq, with(subset(df, q == qq & m == mm),
                                            cor(freq, error.rate, method = "spearman"))))
  }
}
```

```
## [1] "m = 8 q = 20 0.391378907595036"
## [1] "m = 8 q = 25 0.386114394453723"
## [1] "m = 8 q = 30 0.392749524829647"
## [1] "m = 16 q = 20 0.396904201852173"
## [1] "m = 16 q = 25 0.378896004227403"
## [1] "m = 16 q = 30 0.386560187419309"
## [1] "m = 32 q = 20 0.369973194115805"
## [1] "m = 32 q = 25 0.379033330277354"
## [1] "m = 32 q = 30 0.381186877043874"
```

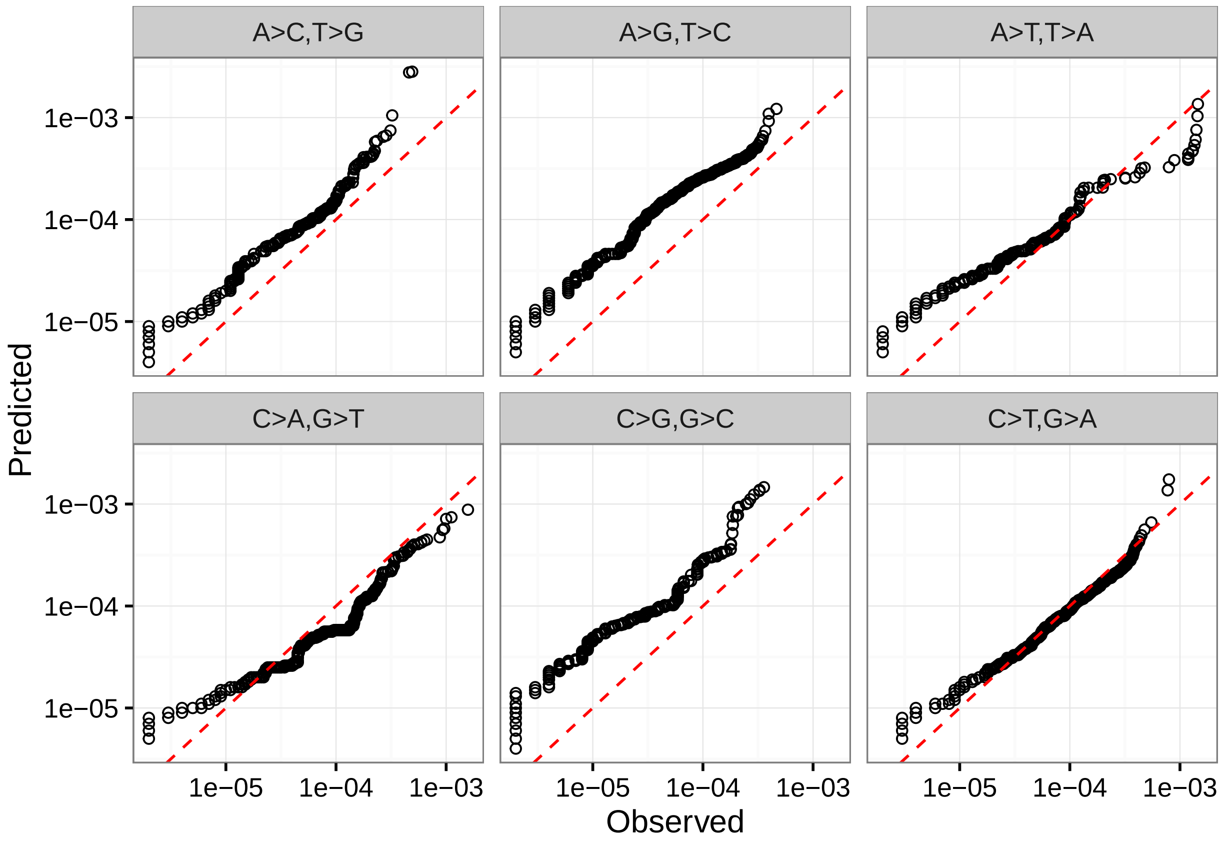
Comparing observed and predicted error frequencies using Q-Q plot

```
df.1 <- subset(df, q == 20 & m == 8)

roundsort <- function(x) {
  round(sort(x) * 10^6)/10^6
}

df.1.qq <- df.1 %>%
  group_by(mutation.type) %>%
  mutate(xx=roundsort(freq), yy=roundsort(error.rate)) %>%
  distinct(mutation.type, xx, yy)

ggplot() +
  geom_point(data=df.1.qq, aes(x=xx, y = yy), shape=21) +
  geom_abline(slope = 1, intercept = 0, color="red", linetype="dashed") +
  facet_wrap(~mutation.type) +
  scale_x_log10("Observed") + scale_y_log10("Predicted") +
  theme_bw()
```


A>C,T>G
A>G,T>C
A>T,T>A
C>A,G>T
C>G,G>C
C>T,G>A
Predicted
1e−03
1e−04
1e−05
Observed
1e−05
1e−04
1e−03