Polymerase fidelity estimates

Load required libraries, load and merge MAGERI results.

```
library(plyr); library(ggplot2); library(reshape2); library(gplots); library(knitr); library(RColorBrewe
```

```
##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

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

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

```
load_variant_table <- function(path, project, sample) {
  fname <- paste(path, paste(project, sample, "variant.caller.txt", sep ="."), sep = "/")
  df.1 <- read.table(fname, header=T, sep="\t")
  df.1$project <- project
  df.1$sample <- sample
  df.1
}

# mutation signatures
sign.rep <- data.frame(mutation.signature = 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.signature.rep = 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"))

path <- "data/"
df.meta <- read.table(paste(path, "metadata.txt", sep="/"), header=T, sep = "\t")

# load and concatenate all samples
df.0 <- data.frame()

for (i in 1:nrow(df.meta)) {
  df.0 <- rbind(df.0, load_variant_table(path, df.meta$project[i], df.meta$sample[i]))
}

# append metadata
df.0 <- merge(df.0, df.meta, all.x=T, all.y=F)
```

```
# split mutation signature
df.0$mut.split <- sapply(df.0$mutation, function(x) strsplit(as.character(x),"[S:>]"))
df.0$mutation.pos <- as.integer(sapply(df.0$mut.split, function(x) x[2]))
df.0$mutation.from <- sapply(df.0$mut.split, function(x) x[3])
df.0$mutation.to <- sapply(df.0$mut.split, function(x) x[4])
df.0$mut.split  <- NULL

# mutation signature groups
df.0$mutation.signature <- paste(df.0$mutation.from, df.0$mutation.to, sep =">")
df.0 <- merge(df.0, sign.rep, all.x=T, all.y=F)


# exclude bases that are not in reference
df.0 <- subset(df.0, !(mutation.pos %in% 22:25))

# shift back mutations
df.0$mutation.pos <- ifelse(df.0$mutation.pos > 25, df.0$mutation.pos - 4,
                            df.0$mutation.pos)

df.0$mutation <- with(df.0, paste("S", mutation.pos, ":", mutation.from, ">",
                                  mutation.to, sep=""))

df <- subset(df.0, project %in% c("polerr2016-1", "polerr2016-2"))
df.linpcr <- subset(df.0, project %in% c("polerr73", "polerr82"))

template <- paste("TAGCGTGAAGACGACAGAACCATGGGATCCATTATCGGCGGCGAATTTACCACCATTGAAAACCAGCCGTGGTTT",
                  "GCGGCGATTTATCGTCGTCATCGTGGCGGCAGCGTGACCTATGTGTGCGGCGGCAGCCTGATTAGCCCGTGCTGG",
                  sep="") # 4 index bases removed
```

Estimating error rates

In this section we compute linear PCR error rate from Proj73/82 and error rate in conventional PCR from Polerr2016 project. Note that the linear PCR error rate is substantially higher than rate per cycle of conentional PCR. For some cases, e.g. Phusion it accounts for ~1/2 of errors observed in Polerr2016.

```
# Compute linear PCR error rate

df.er.linpcr <- ddply(df.linpcr, .(project, name), summarize, mismatches = sum(count.major),
                      umi.count = mean(coverage))
df.er.linpcr <- ddply(df.er.linpcr, .(name), summarize, mismatches = sum(mismatches),
                      umi.count = round(sum(umi.count)))

df.er.linpcr <- data.frame(name = df.er.linpcr$name,
                           linpcr.er = df.er.linpcr$mismatches/df.er.linpcr$umi.count /
                             nchar(template))

# Compute uncrorrected error rate

df.er <- ddply(df, .(project, name, cycles), summarize,
               mismatches = sum(count.major), umi.count = round(mean(coverage)))

df.er <- merge(df.er, df.er.linpcr, by = "name", all.x = T)
df.er$err.rate <- with(df.er, mismatches / umi.count / nchar(template) / mean(cycles))
df.er$delta <- with(df.er,
                    sqrt(mismatches / umi.count * (1 - mismatches / umi.count) / umi.count) /
                      nchar(template) / mean(cycles))
df.er$err.lb <- df.er$err.rate - 1.96 * df.er$delta
df.er$err.ub <- df.er$err.rate + 1.96 * df.er$delta

# Error rates corrected for linear PCR errors

df.er$mismatches.corr <- with(df.er, mismatches - linpcr.er * umi.count * nchar(template))
df.er$err.rate.corr <- with(df.er, mismatches.corr / umi.count / nchar(template) / mean(cycles))
df.er$delta.corr <- with(df.er, sqrt(mismatches.corr / umi.count *
                                       (1 - mismatches.corr / umi.count) / umi.count) /
                 nchar(template) / mean(cycles))
df.er$err.lb.corr <- df.er$err.rate.corr - 1.96 * df.er$delta.corr
df.er$err.ub.corr <- df.er$err.rate.corr + 1.96 * df.er$delta.corr

df.er$cycles <- NULL

kable(df.er)
```

name	project	mismatches	umi.count	linpcr.er	err.rate	delta	err.lb	err.ub	mismatche
Encyclo	polerr2016-1	24557	185560	0.0001279	4.30e-05	3.0e-07	4.25e-05	4.35e-05	20996.
Encyclo	polerr2016-2	14211	101516	0.0001279	4.55e-05	4.0e-07	4.48e-05	4.62e-05	12262.
Kappa HF	polerr2016-1	339	7876	0.0000817	1.40e-05	7.0e-07	1.25e-05	1.55e-05	242.
Kappa HF	polerr2016-2	2519	57052	0.0000817	1.44e-05	3.0e-07	1.38e-05	1.49e-05	1819.
KTN	polerr2016-2	6298	44331	0.0001292	4.62e-05	5.0e-07	4.51e-05	4.73e-05	5438.
KTN	polerr2016-1	16802	132733	0.0001292	4.12e-05	3.0e-07	4.06e-05	4.17e-05	14229.
Phusion	polerr2016-2	23	1351	0.0000499	5.50e-06	1.1e-06	3.30e-06	7.80e-06	12.
Phusion	polerr2016-1	30	1348	0.0000499	7.20e-06	1.3e-06	4.70e-06	9.80e-06	19.
SD	polerr2016-1	6714	33076	0.0002689	6.60e-05	7.0e-07	6.46e-05	6.74e-05	5380.
SD	polerr2016-2	10362	58518	0.0002689	5.76e-05	5.0e-07	5.66e-05	5.86e-05	8001.
SNP-detect	polerr2016-1	457	13870	0.0000504	1.07e-05	5.0e-07	9.70e-06	1.17e-05	352.
SNP-detect	polerr2016-2	848	32310	0.0000504	8.50e-06	3.0e-07	8.00e-06	9.10e-06	603.
Taq-HS	polerr2016-2	3113	24082	0.0001836	4.20e-05	7.0e-07	4.07e-05	4.34e-05	2449.
Taq-HS	polerr2016-1	1875	15137	0.0001836	4.03e-05	9.0e-07	3.86e-05	4.20e-05	1458.
Tersus-buffer1	polerr2016-2	1299	30683	0.0000510	1.38e-05	4.0e-07	1.30e-05	1.45e-05	1064.
Tersus-buffer1	polerr2016-1	6282	130891	0.0000510	1.56e-05	2.0e-07	1.52e-05	1.60e-05	5280.
Tersus-buffer2	polerr2016-1	2550	46927	0.0000598	1.77e-05	3.0e-07	1.70e-05	1.83e-05	2128.
Tersus-buffer2	polerr2016-2	5504	154226	0.0000598	1.16e-05	2.0e-07	1.13e-05	1.19e-05	4119.
TruSeq	polerr2016-1	312	14164	0.0000410	7.20e-06	4.0e-07	6.40e-06	7.90e-06	224.
TruSeq	polerr2016-2	362	16705	0.0000410	7.00e-06	4.0e-07	6.30e-06	7.80e-06	259.

```
write.table(df.er, file="er.txt", quote=F, sep="\t", row.names = F)
```

Patterns and recurrent errors

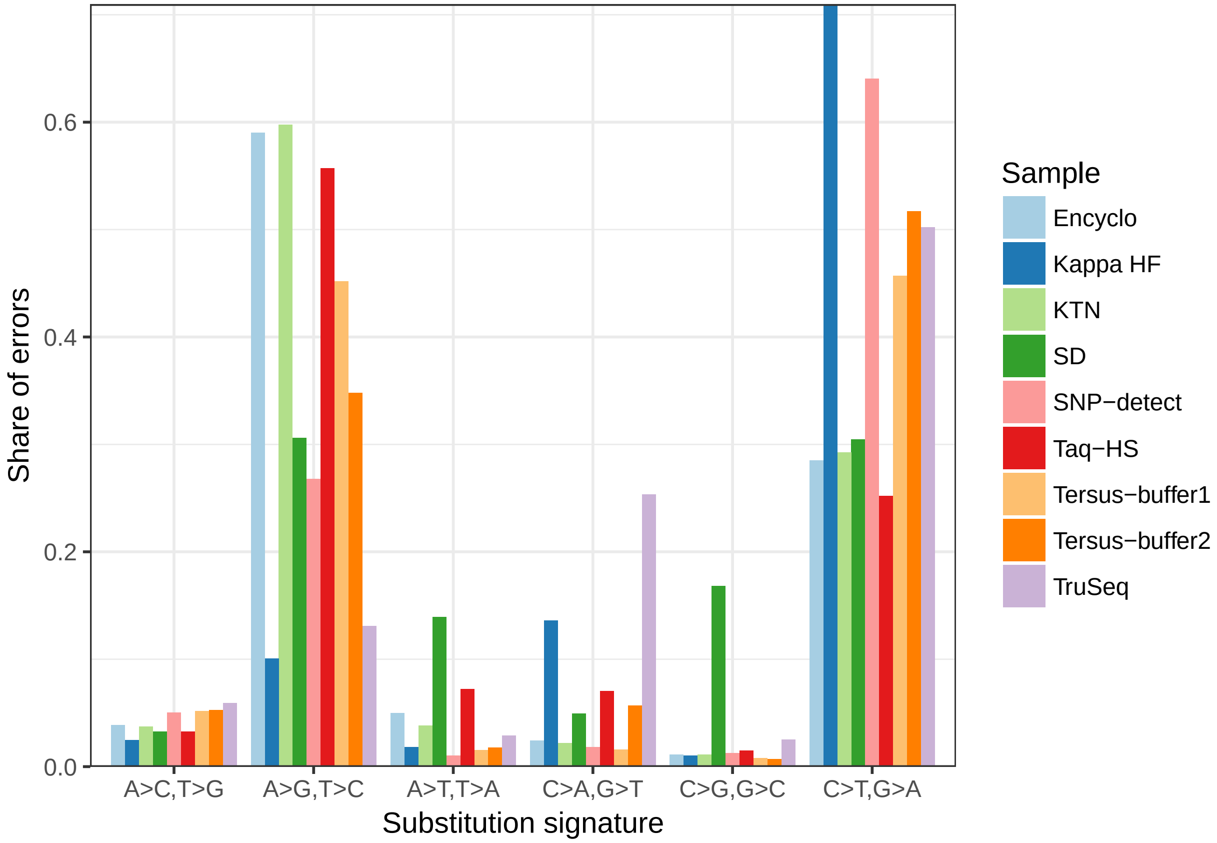
Substitution signature preferences:

```
base.freqs <- ddply(data.frame(base = strsplit(template, "")[[1]]), .(base), summarize,
                    count=length(base))
```

```
bases <- base.freqs$base
base.freqs <- base.freqs$count
names(base.freqs) <- bases

df.pattern <- ddply(df, .(name, mutation.signature.rep, mutation.from), summarize,
                    count.sum = sum(count.major))
df.pattern <- ddply(df.pattern, .(mutation.from), transform, count.sum = count.sum /
                      base.freqs[mutation.from])
df.pattern <- ddply(df.pattern, .(name), transform, freq = count.sum / sum(count.sum))

p1 = ggplot(subset(df.pattern, name != "Phusion"), aes(x = mutation.signature.rep, weight = freq,
               fill = name)) + geom_bar(position = position_dodge()) +
  xlab("Substitution signature") + scale_y_continuous("Share of errors", expand=c(0,0)) +
  scale_fill_brewer("Sample", palette = "Paired") + theme_bw()

pdf("figures/fig2a.pdf")
p1
dev.off()
```

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

```
p1
```

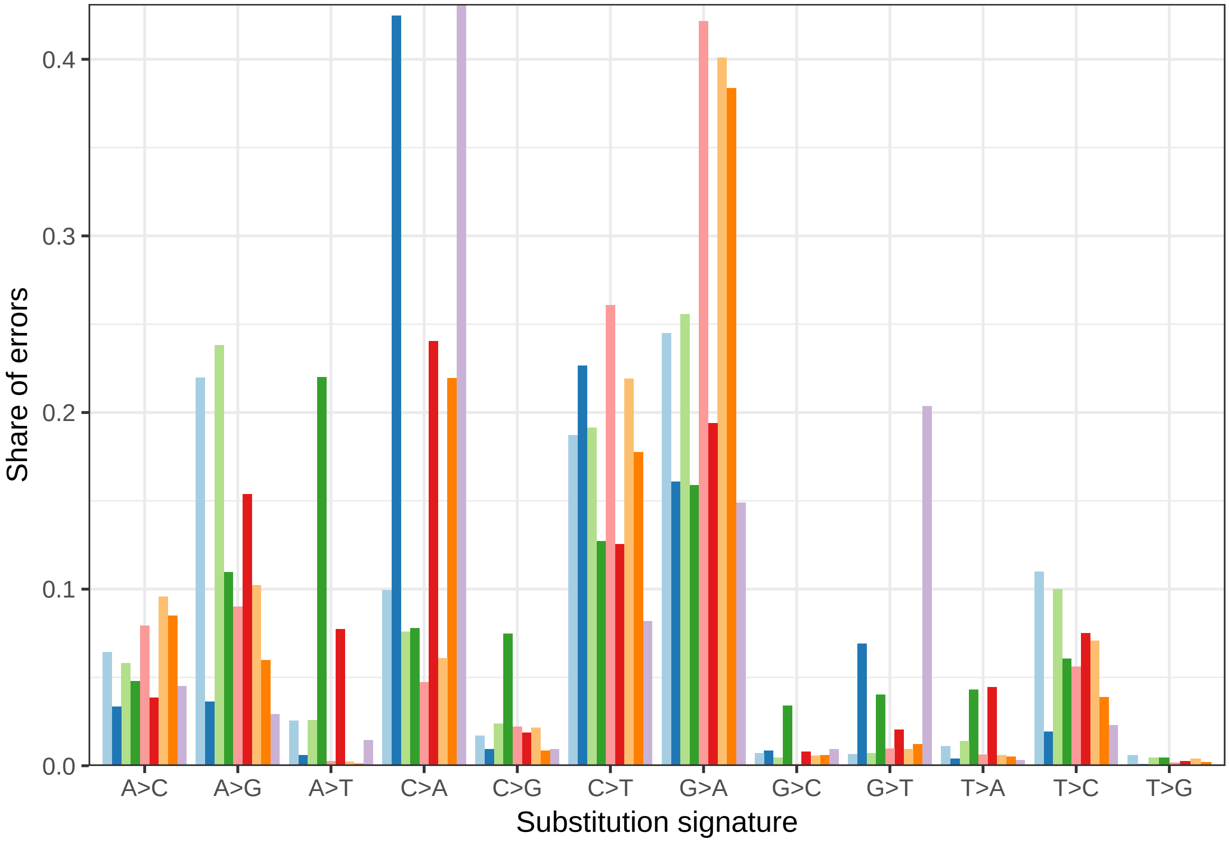


```
write.table(dcast(df.pattern, name ~ mutation.signature.rep, value.var = "freq",
                  fun.aggregate = sum),
            file="pattern.txt", quote=F, sep="\t", row.names = F)
```

```r
# same for linpcr

df.pattern.linpcr <- ddply(df.linpcr, .(name, mutation.signature, mutation.from), summarize,
                           count.sum = sum(count.major))
df.pattern <- ddply(df.pattern, .(mutation.from), transform,
                    count.sum = count.sum / base.freqs[mutation.from])
df.pattern.linpcr <- ddply(df.pattern.linpcr, .(name), transform,
                           freq = count.sum / sum(count.sum))

p2 = ggplot(subset(df.pattern.linpcr, name != "Phusion"), aes(x = mutation.signature, weight = freq,
               fill = name)) + geom_bar(position = position_dodge()) +
  xlab("Substitution signature") + scale_y_continuous("Share of errors", expand=c(0,0)) +
  scale_fill_brewer(palette = "Paired", guide = F) + theme_bw()

pdf("figures/fig2b.pdf")
p2
dev.off()
```

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

```r
p2
```



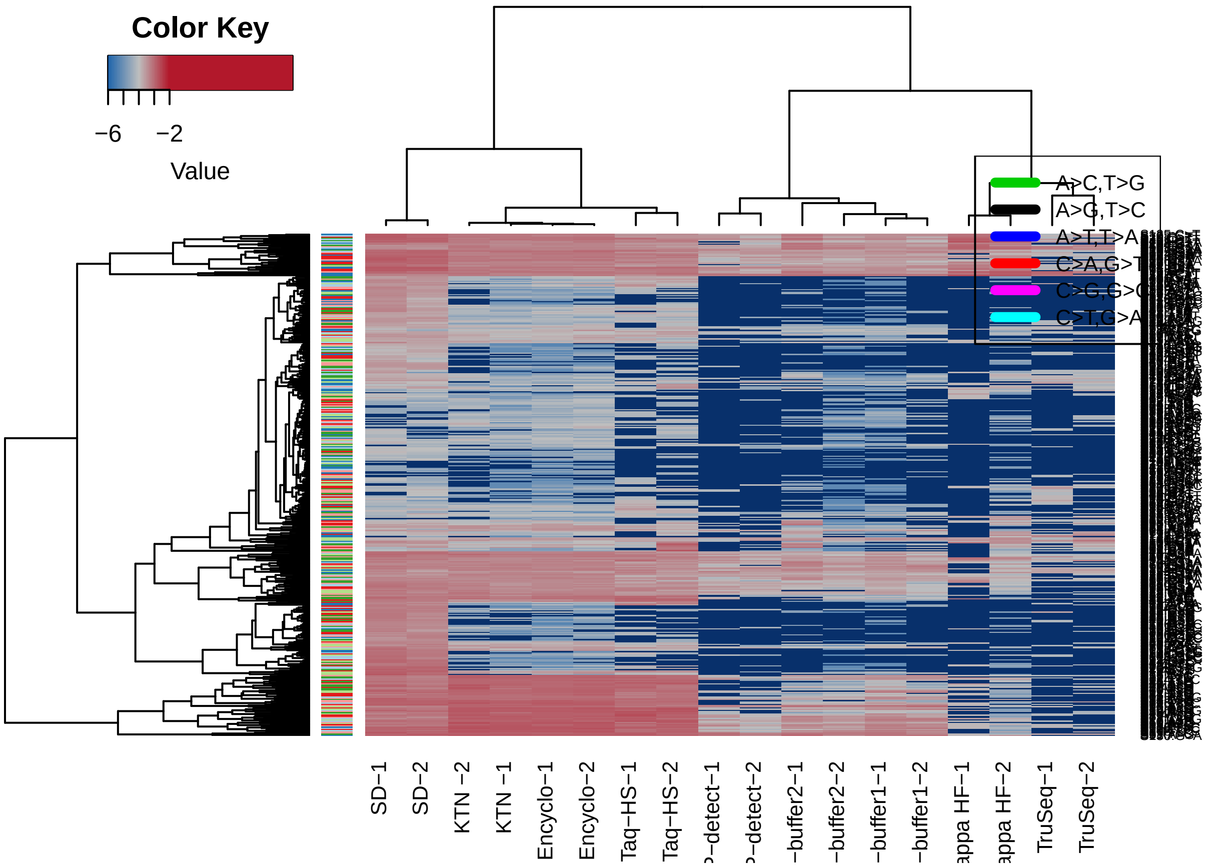
```r
write.table(dcast(df.pattern.linpcr, name ~ mutation.signature, value.var = "freq",
                  fun.aggregate = sum),
            file="pattern.lpcr.txt", quote=F, sep="\t", row.names = F)
```

Clustering of polymerase error profiles:

```
df$freq <- df$count.major / df$coverage
df$name_proj <- paste(df$name, ifelse(df$project == "polerr2016-1", "1", "2"), sep="-")
mat.profile <- dcast(df, name_proj ~ mutation, value.var = "freq")
rownames(mat.profile) <- mat.profile[,1]
mat.profile[,1] <- NULL
mat.profile <- t(as.matrix(mat.profile))
mat.profile[is.na(mat.profile)] <- 0

df.aux <- unique(data.frame(mutation = df$mutation, signature = df$mutation.signature.rep))
df.aux$mutation <- as.character(df.aux$mutation)

df.color <- merge(data.frame(mutation = rownames(mat.profile)), df.aux)

df.color.legend <- data.frame(colors = brewer.pal(6, "Paired"),
                              signature = levels(df.color$signature))

df.color <- merge(df.color, df.color.legend)

rowcolor <- as.character(df.color$colors)
names(rowcolor) <- df.color$mutation

mat.profile = as.data.frame(mat.profile)
mat.profile[,"Phusion-1"] = NULL
mat.profile[,"Phusion-2"] = NULL
mat.profile = as.matrix(mat.profile)

rowDend <- hclust(dist(mat.profile))
colDend <- hclust(as.dist((1-cor(mat.profile))/2), method = "ward")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
heatmap.2(log10(mat.profile), col=c("#08306b", colorpanel(99, "#2166ac", "grey", "#b2182b")),
          dendrogram = "both", RowSideColors = rowcolor, breaks = seq(-6,-2,length.out = 101),
          Rowv = as.dendrogram(rowDend),
          Colv = as.dendrogram(colDend),
          density.info = "none", trace="none")

legend(x="topright", xpd=TRUE,
    legend = df.color.legend$signature,
    col = df.color.legend$colors,
    lty= 1,
    lwd = 5,
    cex=.7
    )
```



```
postscript("figures/fig2c.ps", horizontal = F)
heatmap.2(log10(mat.profile), col=c("#08306b", colorpanel(99, "#2166ac", "grey", "#b2182b")),
          dendrogram = "both", RowSideColors = rowcolor, breaks = seq(-6,-2,length.out = 101),
          Rowv = as.dendrogram(rowDend),
          Colv = as.dendrogram(colDend),
          density.info = "none", trace="none")
legend(x="topright", xpd=TRUE,
    legend = df.color.legend$signature,
    col = df.color.legend$colors,
    lty= 1,
    lwd = 5,
    cex=.7
    )
dev.off()
```

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

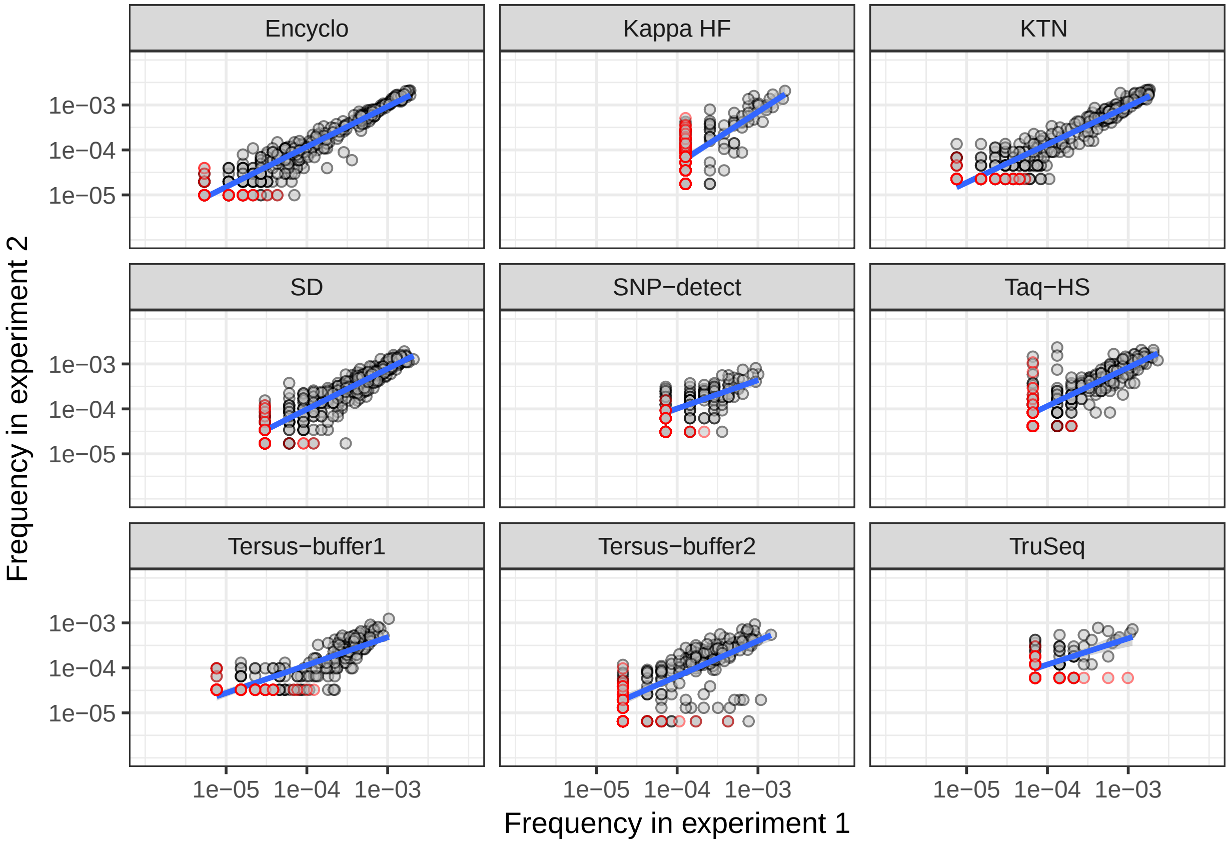
Recurrent errors

```
df.1x <- ddply(subset(df, project == "polerr2016-1"), .(name, mutation), summarize,
               freq = count.major / coverage, coverage = coverage)

df.1y <- ddply(subset(df, project == "polerr2016-2"), .(name, mutation), summarize,
               freq = count.major / coverage, coverage = coverage)

df.1 <- merge(df.1x, df.1y, by=c("name","mutation"), all = T)
mask1 <- is.na(df.1$freq.x)
```

```
mask2 <- is.na(df.1$freq.y)
df.1$miss <- mask1 | mask2

df.1 <- ddply(df.1, .(name), transform,
              freq.x = ifelse(is.na(freq.x), 1/mean(coverage.x, na.rm = T), freq.x),
              freq.y = ifelse(is.na(freq.y), 1/mean(coverage.y, na.rm = T), freq.y))

df.1 = subset(df.1, name != "Phusion")

p3 = ggplot() +
  geom_point(data=subset(df.1, !miss), aes(x=freq.x, y=freq.y), shape=21, fill="grey", alpha=0.5) +
  geom_smooth(data=subset(df.1, !miss & name != "Phusion"), aes(x=freq.x, y=freq.y), method="lm") +
  geom_point(data=subset(df.1, miss), aes(x=freq.x, y=freq.y), shape=21, color="red", fill="grey",
             alpha=0.5) +
  facet_wrap(~name) +
  scale_x_log10("Frequency in experiment 1", limits=c(1e-6,1e-2), breaks = c(1e-5, 1e-4, 1e-3)) +
  scale_y_log10("Frequency in experiment 2", limits=c(1e-6,1e-2), breaks = c(1e-5, 1e-4, 1e-3)) +
  theme_bw()

pdf("figures/fig3.pdf")
p3
dev.off()
```

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

```
p3
```



Indibidual mutations

Frequency distribution of individual mutations, grouped by their pattern.

```r
df.1 <- subset(df, project %in% c("polerr2016-1", "polerr2016-2"))
df.1 <- ddply(df.1, .(name, mutation.pos, mutation.signature.rep), summarize, count.major=sum(count.maj
              coverage=sum(coverage))
df.mean.err <- ddply(df.1, .(name), summarize, mean.err.rate = sum(count.major) / mean(coverage) /
                       nchar(template))
# ^ this one is the gloal mean

df.1 <- merge(df.1, df.mean.err, all.x=T, all.y=F)

df.1 = subset(df.1, name != "Phusion")

p4 = ggplot(df.1) +
  geom_histogram(aes(x = count.major / coverage, fill = mutation.signature.rep)) +
  geom_linerange(aes(x = mean.err.rate, ymin = 0, ymax=50), linetype = "dashed", color="black") +
  scale_fill_brewer("Substitution signature", palette = "Paired") +
  scale_x_log10("Error frequency") +
  scale_y_continuous("", expand=c(0,0)) + facet_wrap(~name) + theme_bw() +
  theme(legend.position="bottom")

pdf("figures/fig4.pdf")
p4
```

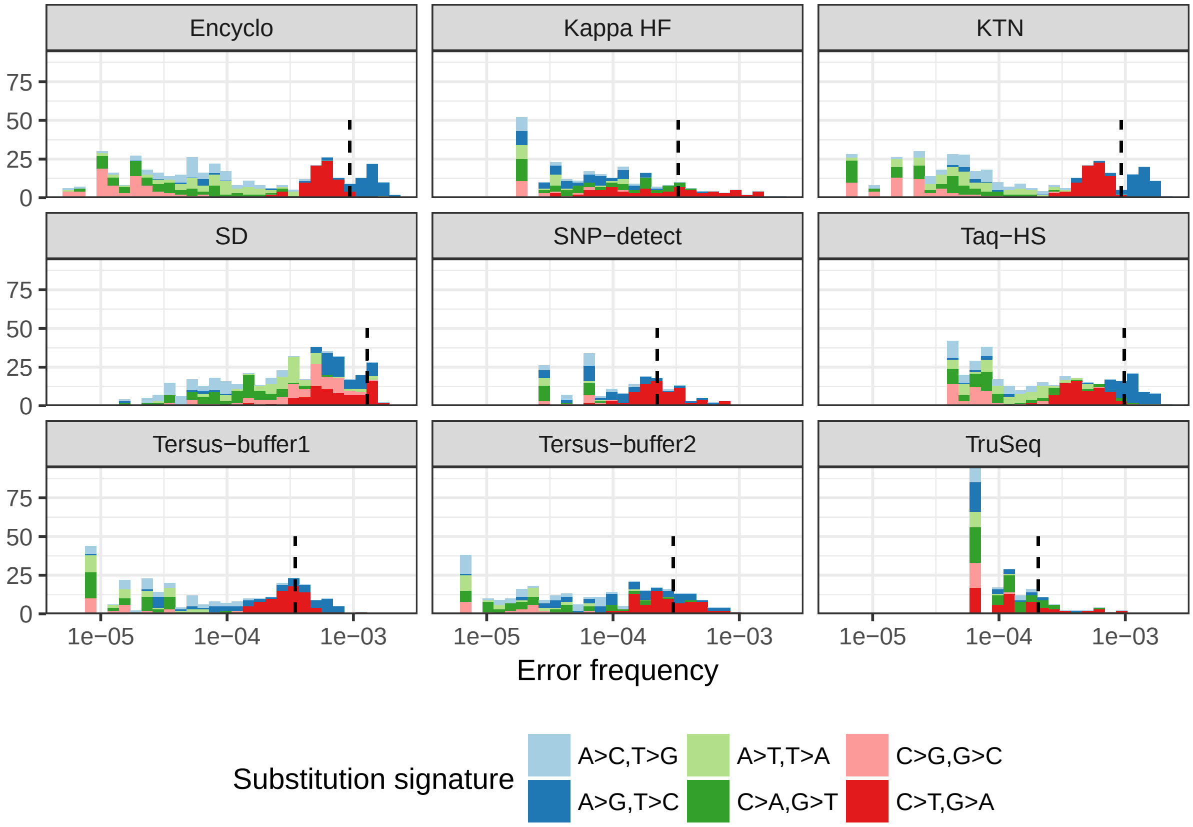
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```r
dev.off()
```

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

```r
p4
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Sequence quality and error patterns

Error frequency for different substitution type and quality threshold.

```
df.qq <- read.table("data/mmqc.txt", header=T, sep="\t", stringsAsFactors = F)
df.qq$mutation.signature <- as.factor(paste(df.qq$from, df.qq$to, sep = ">"))
df.qq <- merge(df.qq, sign.rep, all.x=TRUE)

df.qq.cov <- ddply(subset(df.qq, from == to), .(from, qual), summarize, total=sum(count))

df.qq.1 <- merge(df.qq, df.qq.cov)

df.qq.1 <- ddply(subset(df.qq.1, from != to), .(mutation.signature, qual), summarize,
                 count = sum(count), freq = count / total[1])

p5 = ggplot(df.qq.1, aes(x=qual, y=freq, color=mutation.signature)) +
  geom_smooth(fill="grey80") + geom_point() +
  scale_color_brewer("Substitution type", palette = "Paired") +
  scale_y_log10("Error rate") +
  scale_x_continuous("Quality threshold") +
  theme_bw()

pdf("figures/fig5a.pdf")
p5
```

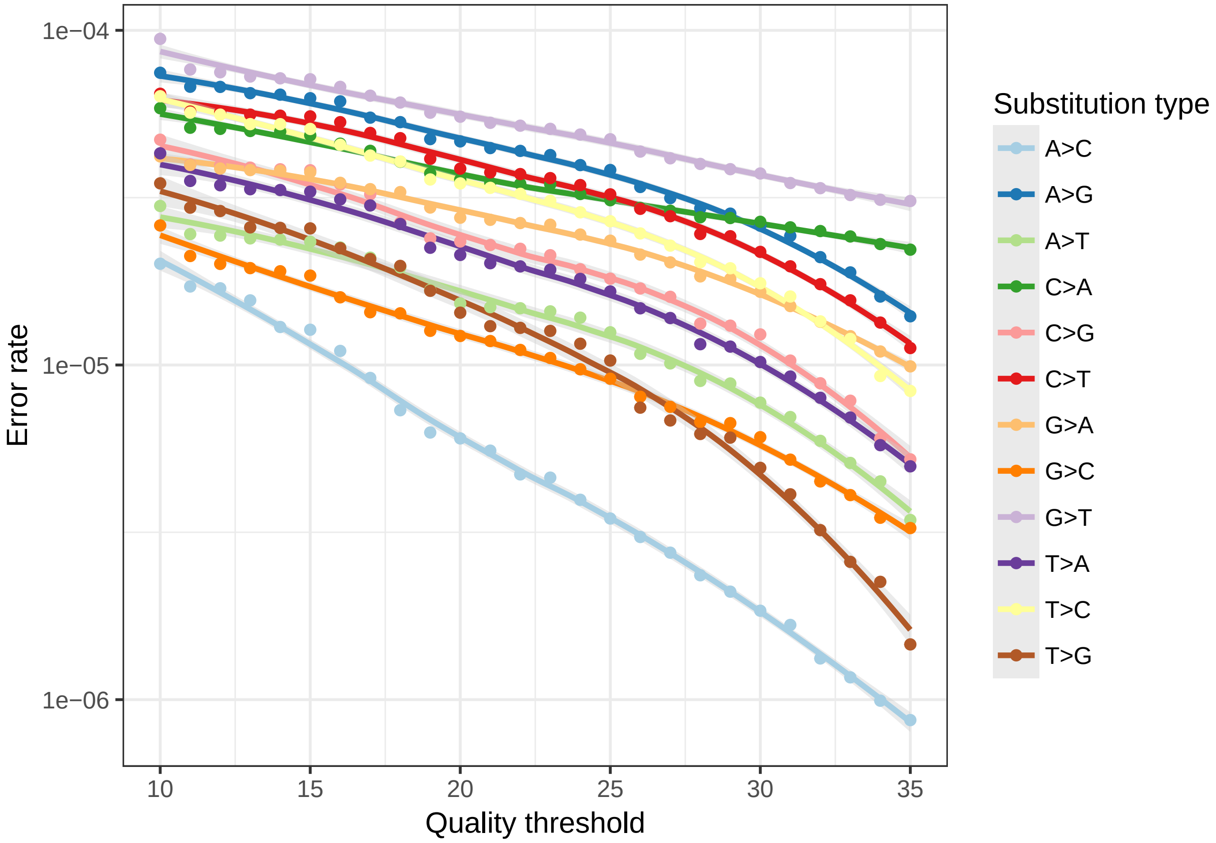
```
## `geom_smooth()` using method = 'loess'
```

```
dev.off()
```

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

```
p5
```

```
## `geom_smooth()` using method = 'loess'
```

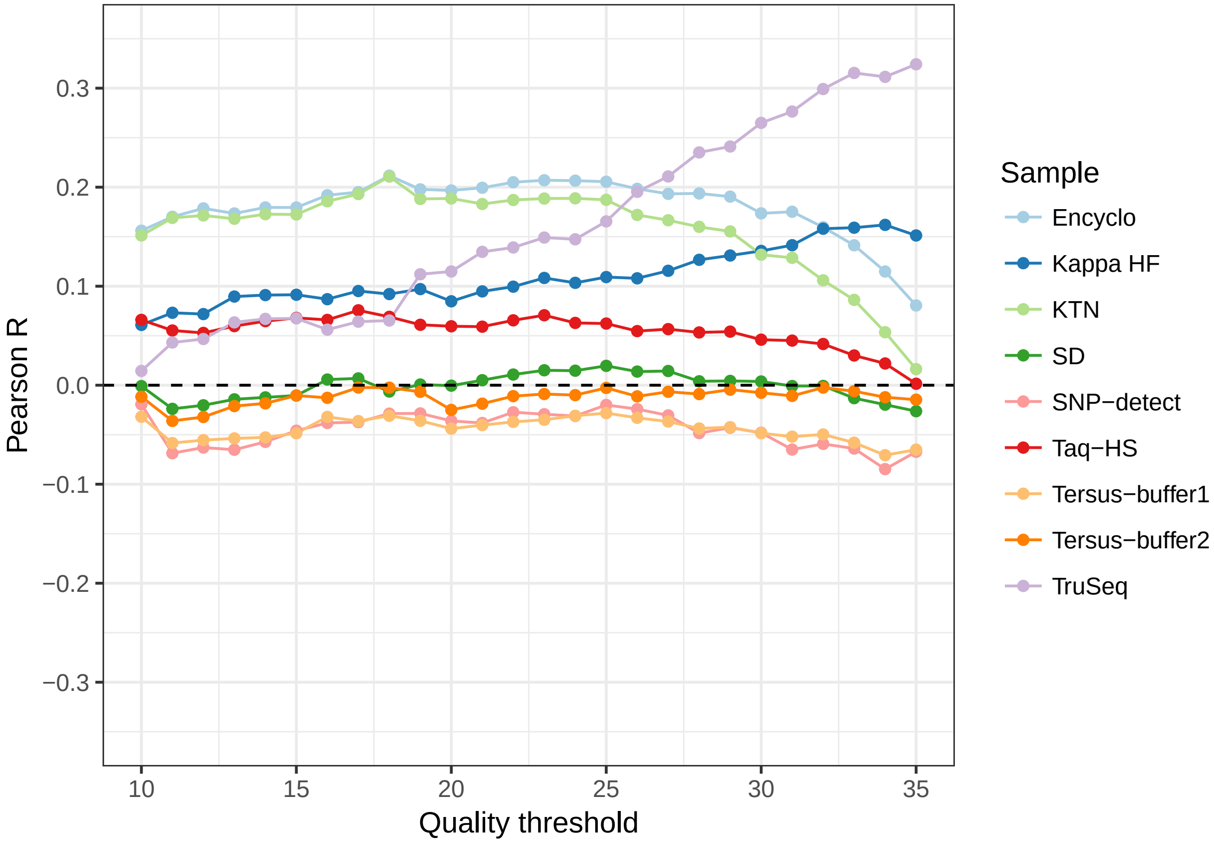


Correlation across polymerase types.

```
df.linpcr.2 <- ddply(df.linpcr, .(name, mutation.pos, mutation.from, mutation.to), summarize,
                     count = sum(count.major))

colnames(df.linpcr.2) <- c("name", "pos", "from", "to", "count.pcr")

df.qq.2 <- merge(df.qq, df.linpcr.2)

df.qq.corr <- ddply(df.qq.2, .(qual, name), summarize, r = cor(count, count.pcr))

df.qq.corr = subset(df.qq.corr, name != "Phusion")

p6 = ggplot(df.qq.corr, aes(x=qual, color=name, y = r)) +
  geom_line() + geom_point() + geom_hline(yintercept = 0, linetype = "dashed") +
  scale_color_brewer("Sample", palette = "Paired") +
  scale_y_continuous("Pearson R", limits = c(-0.35, 0.35),
                     breaks = c(-0.3, -0.2, -0.1, 0, 0.1, 0.2, 0.3)) +
  xlab("Quality threshold") +
  theme_bw()

pdf("figures/fig5b.pdf")
p6
dev.off()
```

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

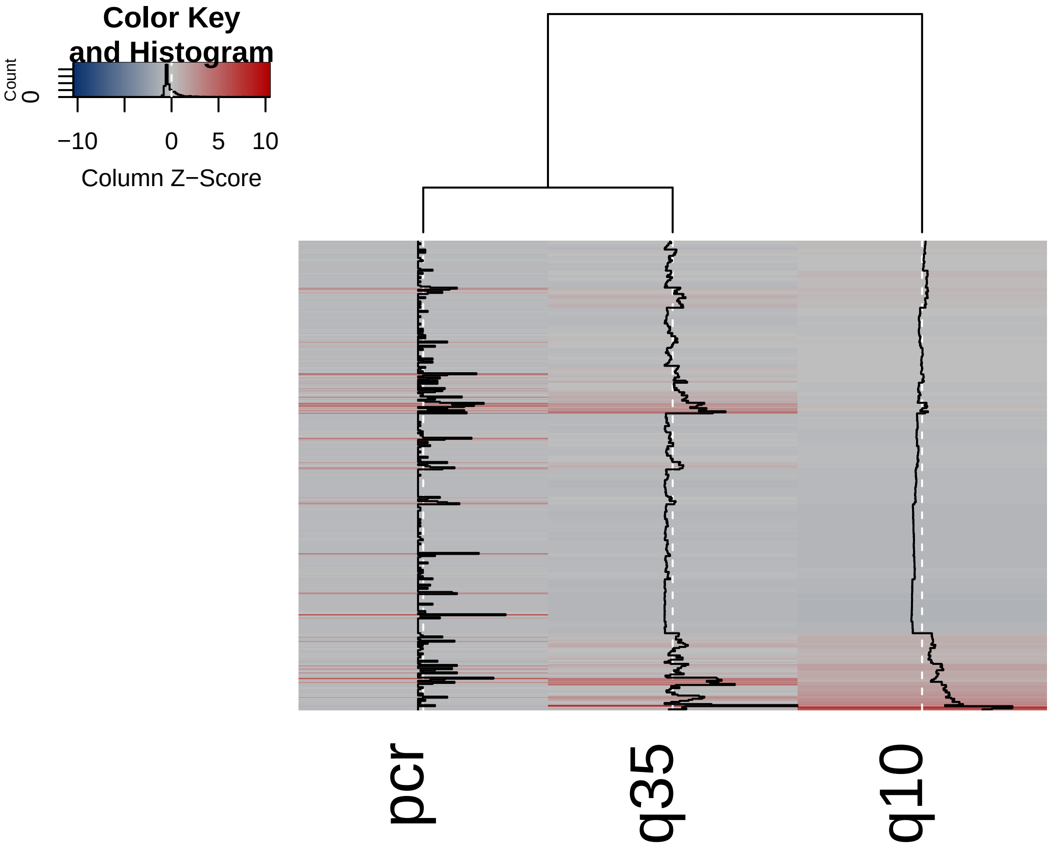
```
p6
```



Clustering TruSeq, Q10 and Q35 error profiles.

```
df.qq.trs <- subset(df.linpcr.2, name=="TruSeq")
df.qq.trs <- data.frame(pos = df.qq.trs$pos, to = df.qq.trs$to, count.trs = df.qq.trs$count.pcr)
df.qq.q10 <- subset(df.qq, qual==10 & from != to)
df.qq.q35 <- subset(df.qq, qual==35 & from != to)
df.qq.clust <- merge(df.qq.trs, df.qq.q10, by = c("pos", "to"), all.y=T)
df.qq.clust <- merge(df.qq.clust, df.qq.q35, by = c("pos", "to"))
df.qq.clust <- data.frame(subst = paste(df.qq.clust$pos, df.qq.clust$to),
                          pcr=df.qq.clust$count.trs,
                          q10=df.qq.clust$count.x,
                          q35=df.qq.clust$count.y)
rownames(df.qq.clust) <- df.qq.clust$subst
df.qq.clust$subst <- NULL

mat.qq.clust <- as.matrix(df.qq.clust)
mat.qq.clust[is.na(mat.qq.clust)] <- 0
```

```
heatmap.2(mat.qq.clust, dendrogram = "column",  labRow = FALSE, scale="col",
          col=colorpanel(100, "#08306b", "grey", "#b30000"), tracecol="black", linecol = "white")
```



```
postscript("figures/fig5c.ps", horizontal = F)
heatmap.2(mat.qq.clust, dendrogram = "column",  labRow = FALSE, scale="col",
          col=colorpanel(100, "#08306b", "grey", "#b30000"), tracecol="black", linecol = "white")
dev.off()
```

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

Sequence context

Load data

```
df.kmer = read.table("data/kmer_errors.txt", header = T, sep = "\t")
```

Normalize error rate for each polymerase and base

```
df.kmer.posnorm = df.kmer %>%
  dplyr::group_by(name, mut.from, mut.pos) %>%
  dplyr::summarize(freq = log10(sum(count) / sum(coverage)), gc = ifelse(mut.from %in% c("G", "C"), 1,

df.kmer.posnorm = df.kmer.posnorm %>%
  dplyr::group_by(name, mut.from) %>%
  dplyr::mutate(freq.norm = scale(freq)[,1])
```

```
p10=ggplot(df.kmer.posnorm, aes(x = mut.pos, y = freq.norm)) +
  #geom_line(aes(group = name)) +
  geom_smooth(color = "black") +
  geom_point(aes(color = name), shape = 21) +
  xlab("Position on template") + ylab("Normalized log10 error rate") +
  #facet_grid(.~mut.from) +
  scale_color_brewer("PCR assay", palette = "Paired") +
  theme_bw()

pdf("figures/fig6a.pdf")
p10
```

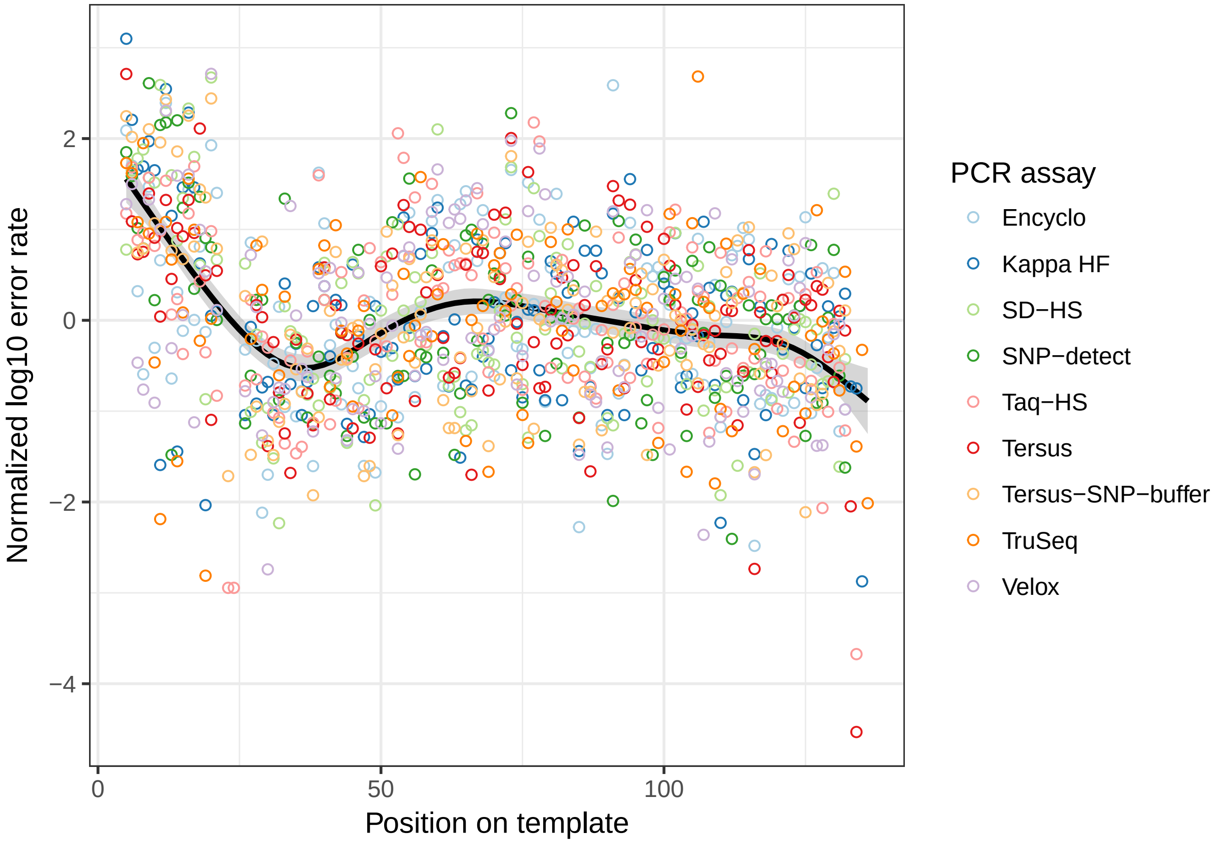
```
## `geom_smooth()` using method = 'gam'
```

```
dev.off()
```

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

```
p10
```

```
## `geom_smooth()` using method = 'gam'
```



```
cor.test(df.kmer.posnorm$mut.pos, df.kmer.posnorm$freq.norm, method = "spearman")
```

```
## Warning in cor.test.default(df.kmer.posnorm$mut.pos, df.kmer.posnorm
## $freq.norm, : Cannot compute exact p-value with ties
```

```
##
```

```
##  Spearman's rank correlation rho
##
## data:  df.kmer.posnorm$mut.pos and df.kmer.posnorm$freq.norm
## S = 248560000, p-value = 7.186e-12
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##        rho
## -0.2072036
```

K-mer model

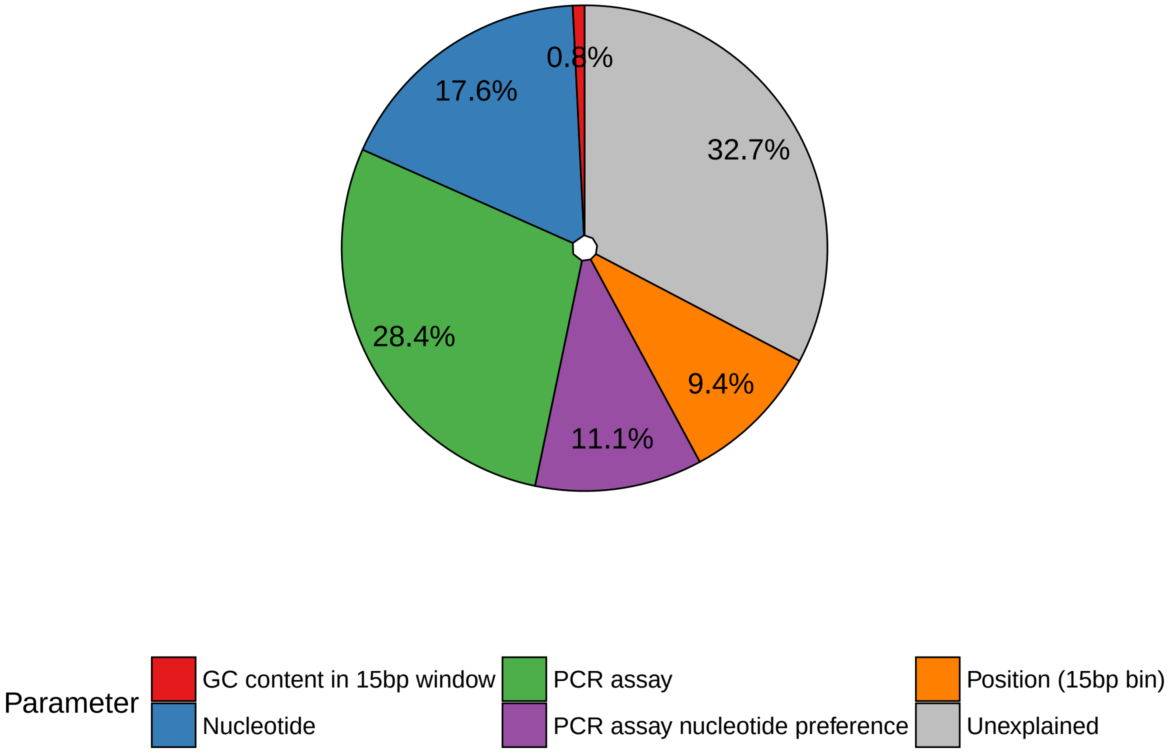
```
library(scales)

df.kmer.pos = df.kmer %>%
  dplyr::group_by(name, mut.pos, mut.from, region.gc) %>%
  dplyr::summarize(freq.pos = sum(count) / sum(coverage))

df.kmer.pos$mut.pos = factor(as.integer(df.kmer.pos$mut.pos / 10))

fit = lm(log10(freq.pos) ~ mut.pos + region.gc + name * mut.from, df.kmer.pos)
af <- anova(fit)
afss <- af$"Sum Sq"
afss <- cbind(af,PctExp=afss/sum(afss)*100)
print(afss)
```

```
##                 Df    Sum Sq     Mean Sq   F value        Pr(>F)    PctExp
## mut.pos         13 16.46515  1.26655018  22.72736  3.436255e-48  9.440777
## region.gc        1  1.35799  1.35798952  24.36818  9.286102e-07  0.778643
## name             8 49.49073  6.18634147 111.00960 3.431933e-133 28.376959
## mut.from         3 30.64862 10.21620511 183.32270  4.063892e-95 17.573280
## name:mut.from   24 19.43241  0.80968384  14.52921  3.802972e-50 11.142142
## Residuals     1023 57.00973  0.05572799        NA            NA 32.688198
```

```
afss$param <- factor(c("Position (15bp bin)", "GC content in 15bp window", "PCR assay", "Nucleotide", "

afss = afss %>%
  dplyr::arrange(param) %>%
  dplyr::mutate(label_pos = 100 - cumsum(PctExp) + PctExp / 2)

p11 = ggplot(afss, aes(x = "", y = PctExp, fill = param)) +
  geom_bar(stat = "identity", color = "black", size = 0.3) +
  geom_text(aes(x = 1.25, y = label_pos, label = percent(PctExp/100)), size = 4) +
  coord_polar(theta = "y") +
  xlab("") + ylab("") +
  scale_fill_manual("Parameter", values = c("#e41a1c", "#377eb8", "#4daf4a", "#984ea3", "#ff7f00", "gre
  theme_bw() +
  theme(axis.text.x=element_blank(), panel.border = element_blank(),
        axis.ticks = element_blank(), panel.grid.major = element_blank(), panel.grid.minor = element_bla

pdf("figures/fig6b.pdf")
p11
dev.off()
```

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

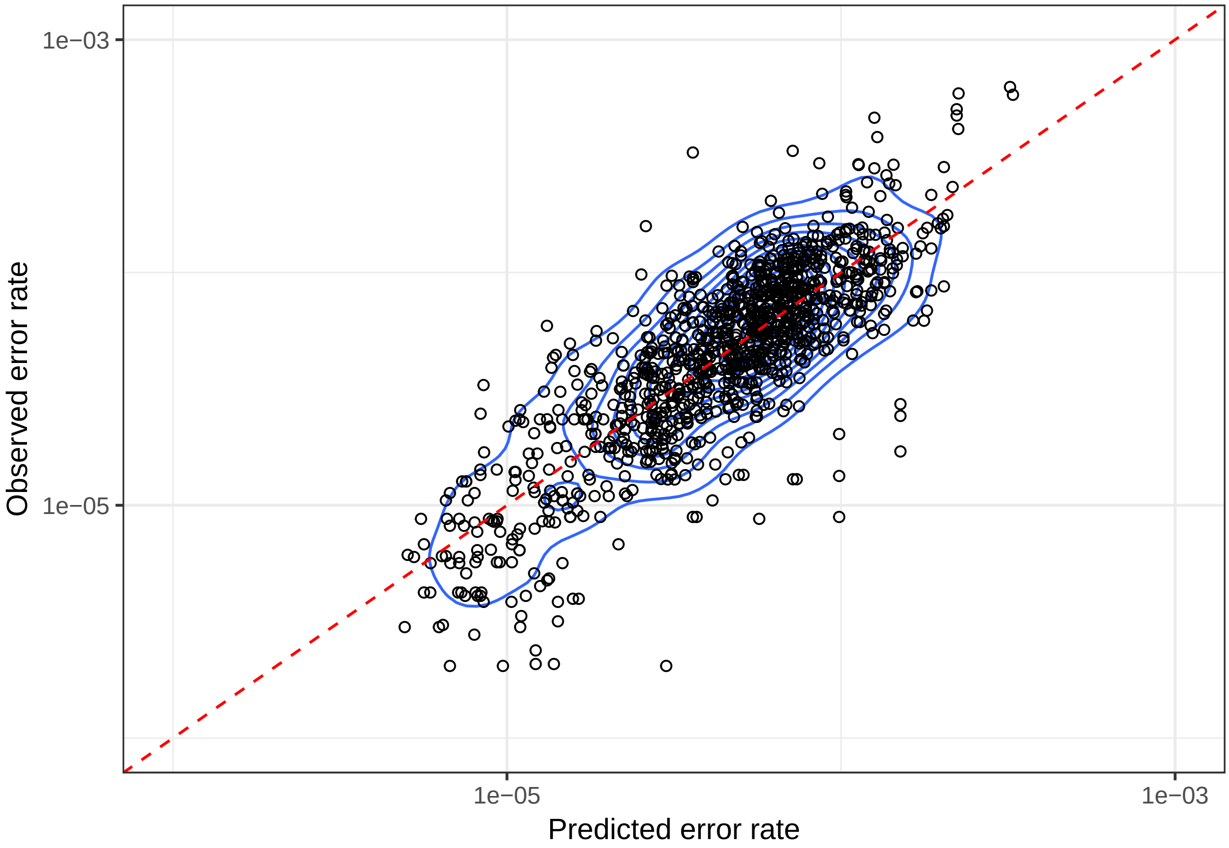
```
p11
```



```
df.kmer.pos$freq.mdl = predict(fit, df.kmer.pos, type="response")

p12 = ggplot(df.kmer.pos, aes(x = 10^freq.mdl, y = freq.pos)) +
  geom_density2d() +
  geom_point(shape=21) +
  geom_abline(slope = 1, intercept = 0, color = "red", linetype = "dashed") +
  scale_x_log10("Predicted error rate", limits = c(1e-6, 1e-3)) +
  scale_y_log10("Observed error rate", limits = c(1e-6, 1e-3)) +
  theme_bw()

pdf("figures/fig6c.pdf")
p12
dev.off()
```

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

```
p12
```



```
cor.test(df.kmer.pos$freq.mdl, df.kmer.pos$freq.pos)
```

```
##
##  Pearson's product-moment correlation
##
## data:  df.kmer.pos$freq.mdl and df.kmer.pos$freq.pos
## t = 26.252, df = 1071, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5878962 0.6608272
## sample estimates:
##       cor
## 0.6257272
```

Supplementary data

Error rate consistency between two independent experimental replicas from Polerr2016 dataset.

```
df.er.cast <- dcast(df.er, name ~ project,
                    value.var = "err.rate.corr")
df.er.cast2 <- dcast(df.er, name ~ project,
                     value.var = "delta")

df.er.cast <- merge(df.er.cast, df.er.cast2, by = "name")
colnames(df.er.cast) <- c("name", "replica1.x", "replica2.x", "replica1.y", "replica2.y")

m <- lm(replica1.x ~ replica2.x, df.er.cast);
```

```
eq <- substitute(italic(y) == a + b %.% italic(x)*","~~italic(R)~"="~r,
     list(a = format(coef(m)[1], digits = 2),
          b = format(coef(m)[2], digits = 2),
          r = format(sqrt(summary(m)$r.squared), digits = 2)))
lbl<-as.character(as.expression(eq))

p7 = ggplot(df.er.cast, aes(x=replica1.x, y=replica2.x)) +
  geom_errorbarh(aes(xmax=replica1.x+1.96*replica1.y,xmin=replica1.x-1.96*replica1.y)) +
  geom_errorbar(aes(ymax=replica2.x+1.96*replica2.y,ymin=replica2.x-1.96*replica2.y)) +
  geom_smooth(method = "lm", color="black", fill="grey80", fullrange = T) +
  geom_point(size=2, shape=21, fill="grey70") +
  geom_text(aes(label=name), vjust=1, hjust = .3, color="red") +
  geom_label(aes(x = 1e-6, y = 8e-5, label = lbl), hjust=-0.1, parse = TRUE)+
  scale_x_log10(name="Error rate in experiment 1, 1/bp/cycle", limits=c(5e-7,1e-4)) +
  scale_y_log10(name="Error rate in experiment 2, 1/bp/cycle", limits=c(5e-7,1e-4)) +
  theme_bw()

pdf("figures/figS1.pdf")
p7
```

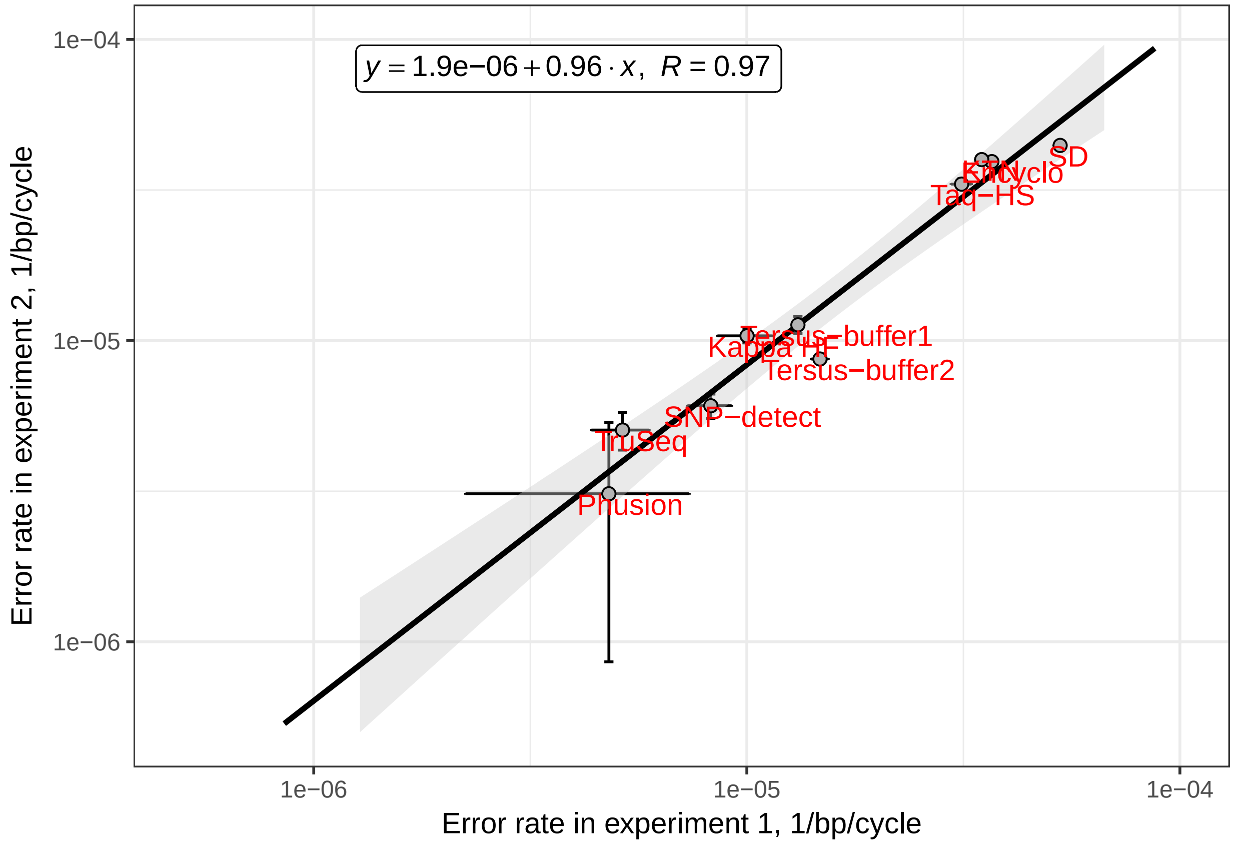
```
## Warning: Removed 10 rows containing missing values (geom_smooth).
```

```
dev.off()
```

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

```
p7
```

```
## Warning: Removed 10 rows containing missing values (geom_smooth).
```



UMI coverage histogram

```
df.meta.1 <- subset(df.meta, project %in% c("polerr2016-1", "polerr2016-2"))

df.h <- data.frame(mig.size.bin = integer(), read.count = integer(), name=character(),
                   project=character())

for (proj in unique(df.meta.1$project)) {
  for (sample in unique(subset(df.meta.1, project == proj)$sample)) {
    mask <- which(df.meta.1$sample == sample & df.meta.1$project == proj)
    name <- df.meta.1$name[mask][1]
    cycles_2 <- df.meta.1$cycles_2[mask][1]
    df.hh <- read.table(paste(path, paste(proj, sample, "umi.histogram.txt", sep="."), sep="/"),
                        header=T, sep="\t")
    df.h <- rbind(df.h, data.frame(mig.size.bin = df.hh$mig.size.bin, read.count = df.hh$read.count,
                                   name=name, project=proj, cycles_2=cycles_2))
  }
}

p8 = ggplot(df.h) +
  geom_rect(aes(xmin=1, xmax=5, ymin=0, ymax=Inf), fill="grey") +
  geom_line(aes(x=mig.size.bin, y=read.count, color=project, linetype=name,
                group = interaction(project, name)), size=1) +
  scale_x_log10("UMI coverage", limits = c(1,1e5), expand=c(0,0)) +
  scale_y_continuous("Number of reads in UMIs", expand=c(0,0), limits=c(0,7.5e6)) +
  scale_color_brewer("Experiment", palette = "Set1") +
  scale_linetype("Sample") +
  theme_bw()

pdf("figures/figS2.pdf")
p8
```

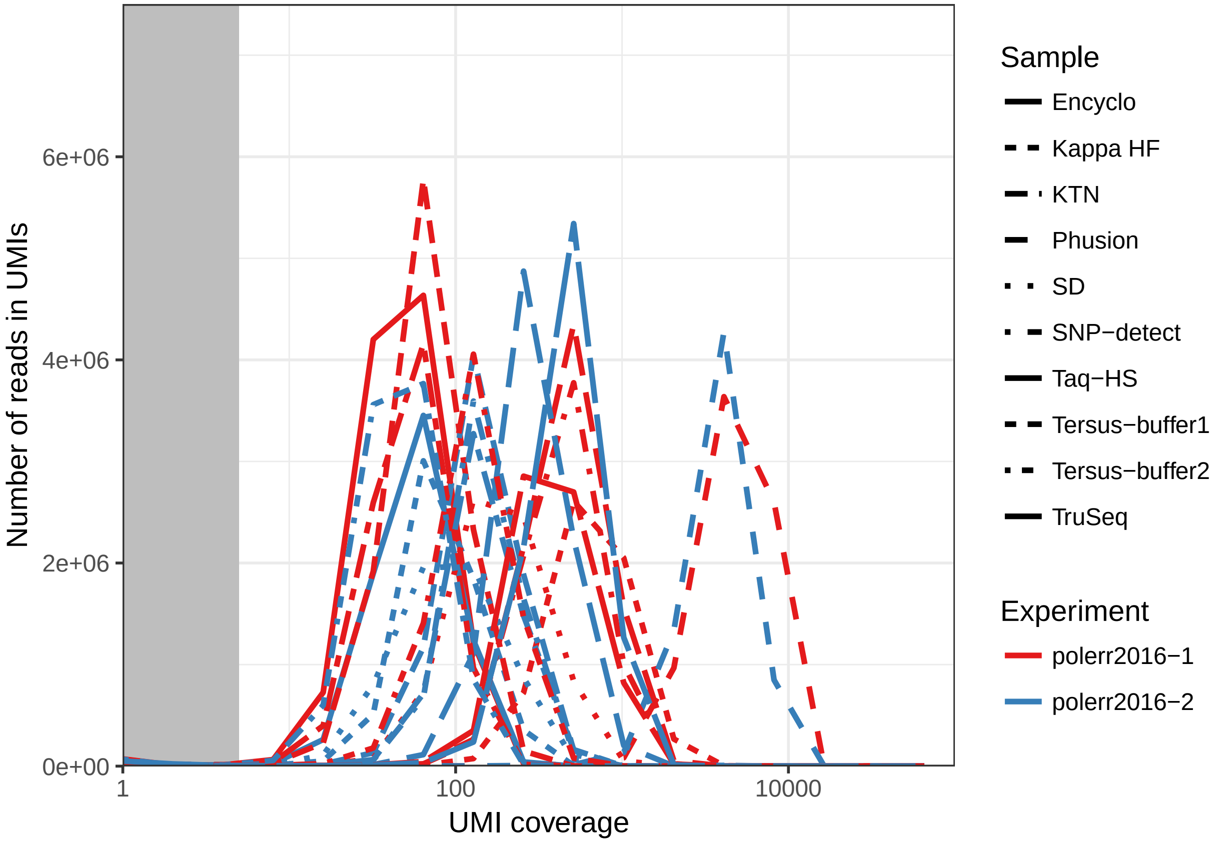
```
## Warning: Removed 60 rows containing missing values (geom_path).
```

```
dev.off()
```

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

```
p8
```

```
## Warning: Removed 60 rows containing missing values (geom_path).
```



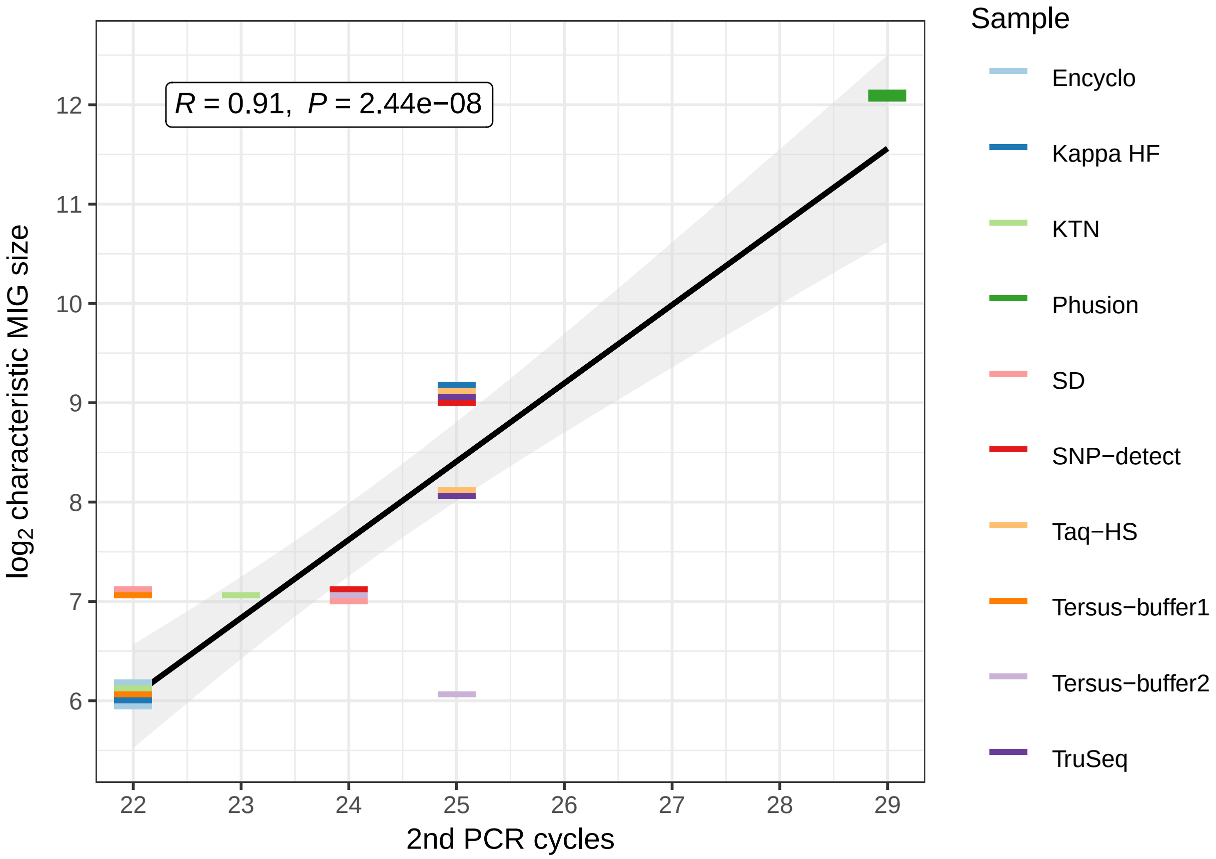
Coverage and PCR cycles

```
library(ggbeeswarm)
df.o <- ddply(df.h, .(project, name, cycles_2), summarize,
              peak = log2(mig.size.bin[which(read.count == max(read.count))]))

m <- lm(peak ~ cycles_2, df.o)
eq <- substitute(italic(R)~"="~r*","~~italic(P)~"="~p,
     list(
     r = format(sqrt(summary(m)$r.squared), digits = 2),
     p = format(summary(m)$coefficient[[8]], digits = 3)
     ))
lbl<-as.character(as.expression(eq))
```

```
p9 = ggplot()+
geom_label(aes(x = 22, y = 12, label = lbl), hjust=-0.1, parse = TRUE)+
stat_smooth(data=df.o, aes(cycles_2, peak), method=lm, color="black", fill="grey85") +
geom_beeswarm(data=df.o, aes(x=cycles_2, y=peak, group = interaction(name,cycles_2), color=name), group
            shape=45, size = 10) +
scale_x_continuous(name="2nd PCR cycles", breaks=10:30) +
scale_y_continuous(expression('log'[2]~'characteristic MIG size'), breaks=2:20) +
scale_color_brewer(name  ="Sample", palette = "Paired") +
theme_bw()

pdf("figures/figS3.pdf")
p9
dev.off()
```

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

```
p9
```


R = 0.91, P = 2.44e−08
log₂ characteristic MIG size
2nd PCR cycles
Sample
Encyclo
Kappa HF
KTN
Phusion
SD
SNP−detect
Taq−HS
Tersus−buffer1
Tersus−buffer2
TruSeq