CodeS3: Characterisation of Psoralen signature

Sigurgeir Ólafsson

6/22/2022

Introduction

This document describes the various analyses carried out to characterise a mutational signature we identify and attribute to psoralen exposure.

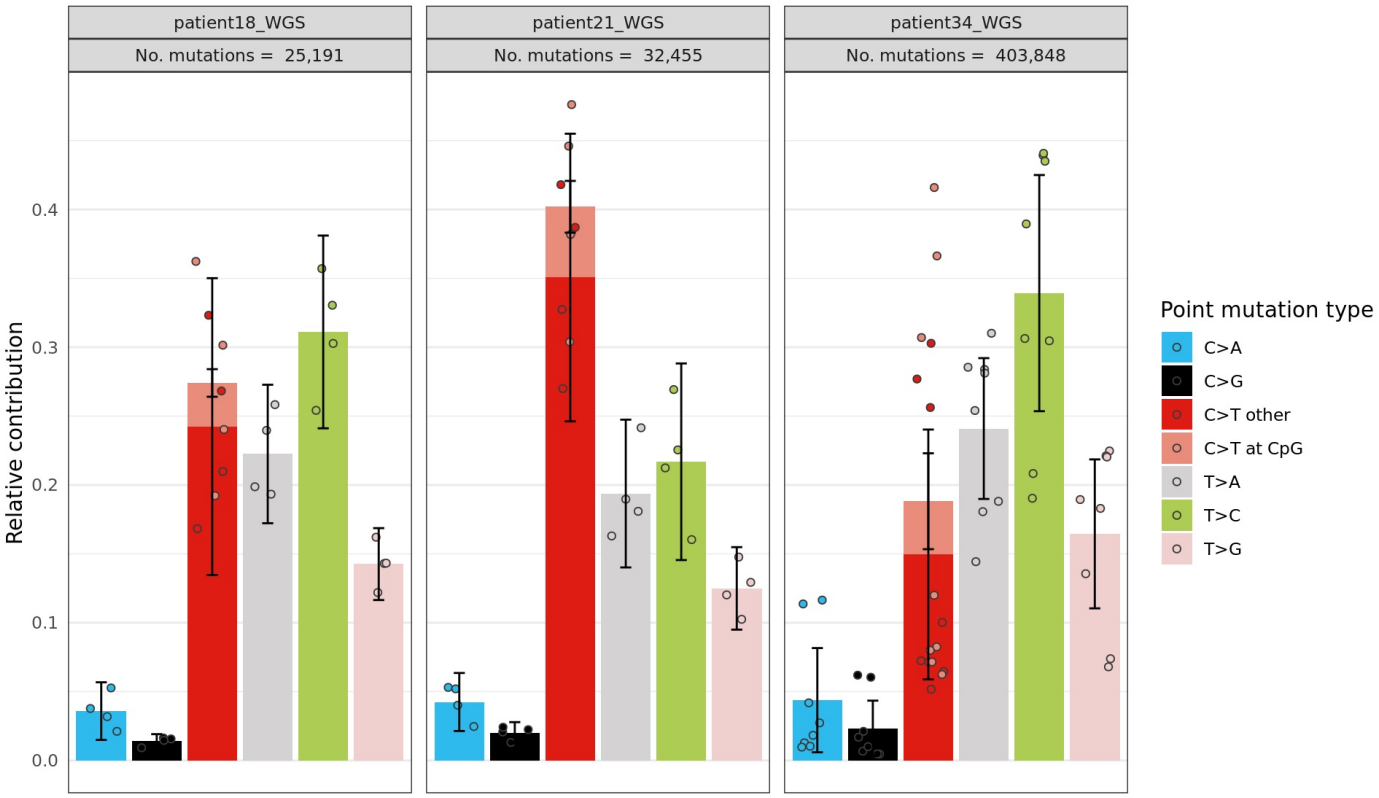
To enable detailed characterisation of the signature, we performed whole genome sequencing of 16 microbiopsies which showed clear evidence of the signature in the whole-exome data. The terms psoralen and PUVA exposure are sometimes used interchangibly throughout this document. Originally we attributed the signature to PUVA exposure and the names of some of the directories and early files reflect that. The “PUVA-signature” and the “Psoralen-signature” are one and the same.

```
.libPaths("/lustre/scratch126/humgen/projects/psoriasis/R_packages_farm5_R4.1.0_install/")

library("TxDb.Hsapiens.UCSC.hg38.knownGene")
library(grid)
library(gridExtra)
library(MutationalPatterns)
ref_genome <- "BSgenome.Hsapiens.UCSC.hg38"
library(ref_genome, character.only = TRUE)
library(ggplot2)
library(reshape2)
library(ggsignif)
library(dplyr)
options(stringsAsFactors = F)

## Define a few variables
genes_hg38 <- genes(TxDb.Hsapiens.UCSC.hg38.knownGene)
sample_meta <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/sample_info/sample_meta
_wgs.txt", h=T, stringsAsFactors = F)
binomial_dir="/lustre/scratch126/humgen/projects/psoriasis/binomial_filters/"
repl_timing_dir="/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/07_signature_extraction/replicati
on_timing/"
puva_dir="/lustre/scratch126/humgen/projects/psoriasis/signature_extraction/puva_characterization/"

## Read in the data
sample_names <- sample_meta$sampleID
patient_list <- sample_meta$patient_ID
vcf_files <- paste(puva_dir, sample_names, ".vcf", sep="")
grl <- read_vcfs_as_granges(vcf_files, sample_names, ref_genome, predefined_dbs_mbs=T)
```

Mutational profiles of psoralen exposed samples

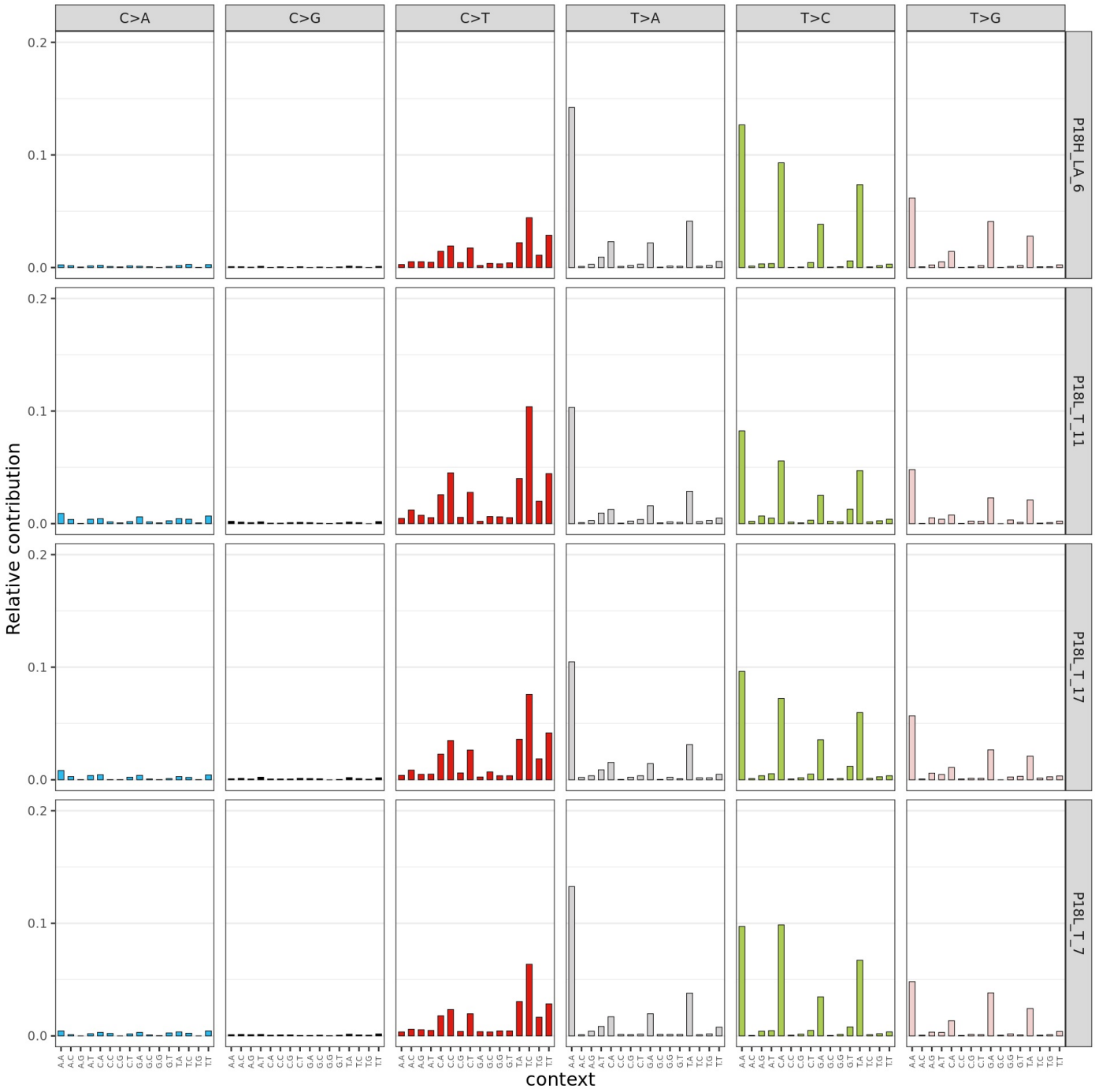
The mutational spectra of skin samples are typically dominated by the C>T mutations which characterise COSMIC signatures SBS7a and SBS7b. We can see that the samples highlighted here show evidence of some UV-exposure but their mutation spectra are dominated by T>A, T>C and T>G mutations at ApT sites.

```
type_occurrences <- mut_type_occurrences(grl, ref_genome)
plot_spectrum(type_occurrences, CT = TRUE,
              indv_points = TRUE, legend = T, by=patient_list)
```



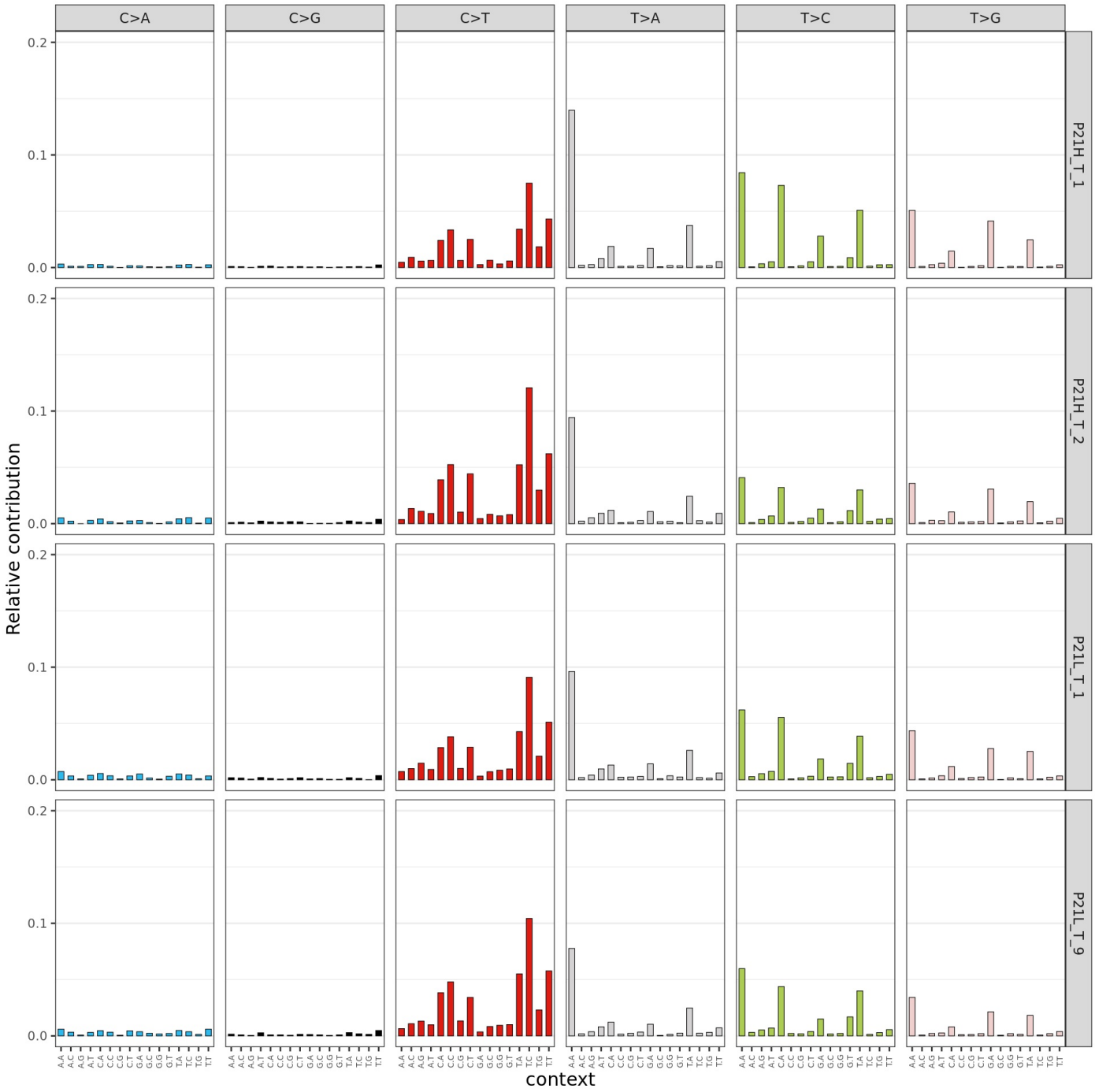

```
mut_mat <- mut_matrix(vcf_list = grl, ref_genome = ref_genome)
plot_96_profile(mut_mat[,c(1:4)])
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```



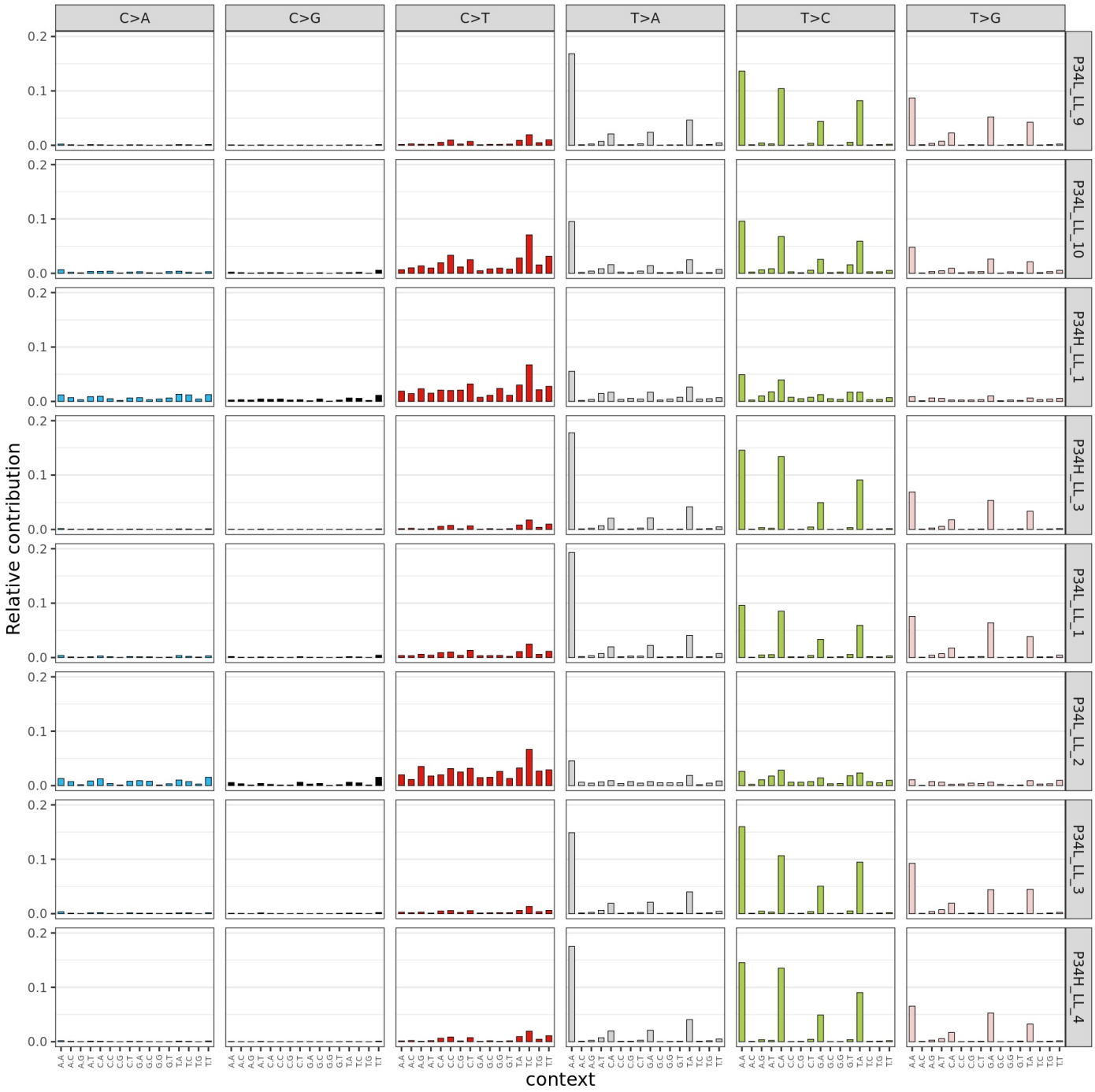

```
plot_96_profile(mut_mat[,c(5:8)])
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```



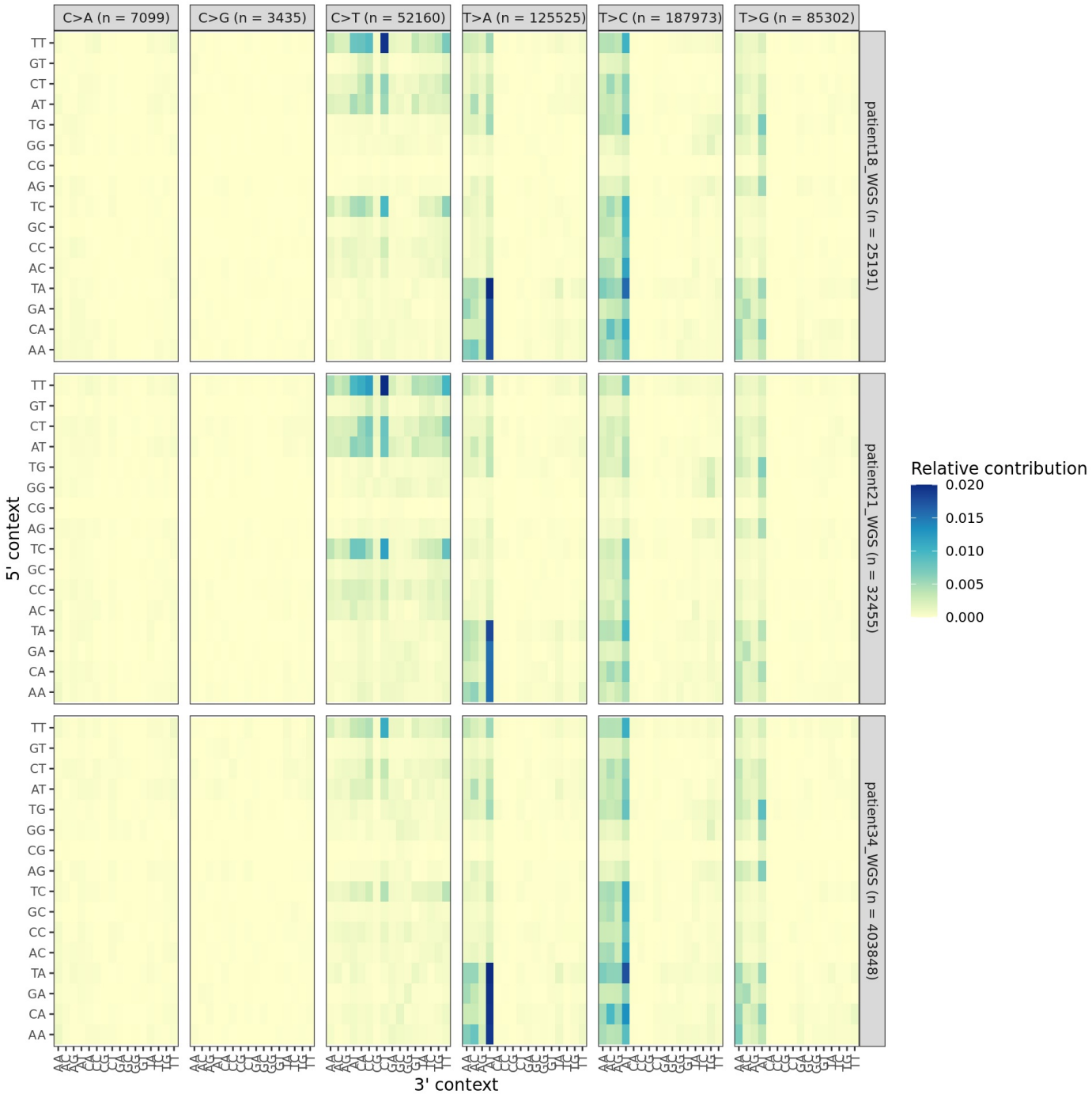

```
plot_96_profile(mut_mat[,c(9:16)])
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```



We may be interested in knowing if the trinucleotide model, which is classically used to describe mutational signatures, is sufficient to describe the context-dependence of this mutational process. We find that it is not so. The effect of sequence context extends beyond the trinucleotide model.

```
mut_mat_ext_context <- mut_matrix(grl, ref_genome, extension = 2)
plot_profile_heatmap(mut_mat_ext_context, by=patient_list)
```

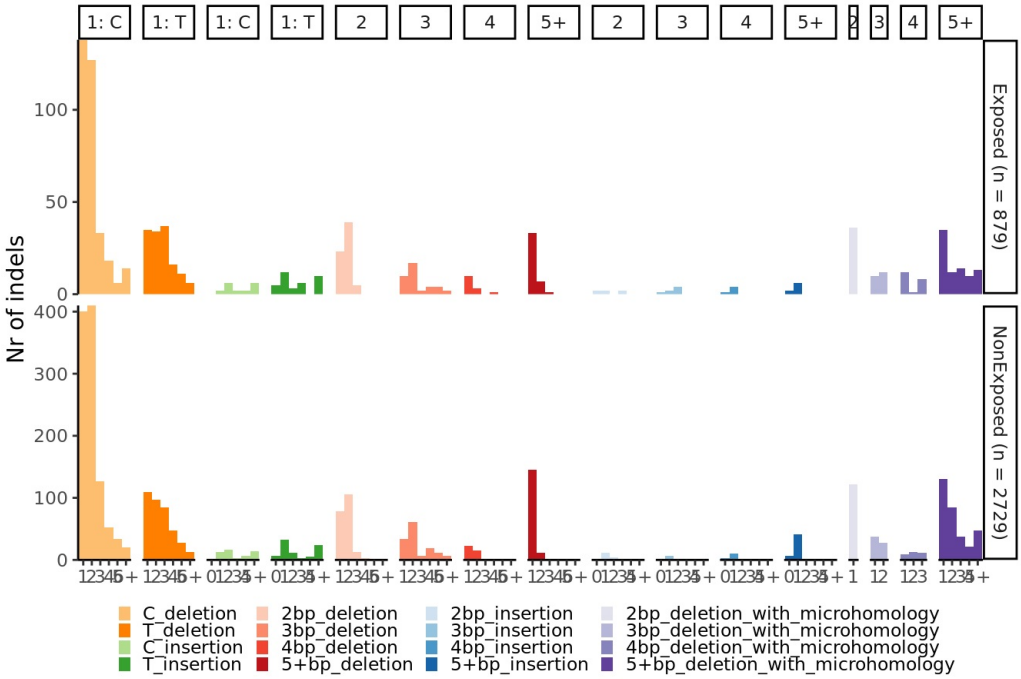


Indel spectra

We can compare the mutation spectra of indels from clones showing evidence of Psoralen exposure vs not. Because no non-exposed samples were whole-genome sequenced, this comparison will be done using the WES data.

I classify patients as psoralen-exposed if any clone from the patient has more than 100 psoralen-associated mutations. We then pool indels across patients in both types. This is because there are too few indels in individual clones (and even across individual patients) to do meaningful signature extraction.

```
patients_wes <- read.table("/lustre/scratch126/humgen/projects/psoriasis/signature_extraction/hdp/snv/patient_list.txt")
patients_wes <- patients_wes$V1[patients_wes$V1!="patient25"] ## Removed as no microbiopsies from lesional skin passed qc.

indels <- data.frame()
for(pat in patients_wes) {
  pat_indels <- read.table(paste(binomial_dir, pat, "/", pat, "_genotype_indels.txt", sep=""), h=T)
  pat_indels$patientID <- pat
  indels <- rbind(indels, data.frame(rownames(pat_indels), pat_indels$patientID))
}

colnames(indels) <- c("mutationID", "patientID")

indels$Chr <- unlist(strsplit(indels$mutationID, split=":"))[c(T,F,F,F)]
indels$Pos <- unlist(strsplit(indels$mutationID, split=":"))[c(F,T,F,F)]
indels$Ref <- unlist(strsplit(indels$mutationID, split=":"))[c(F,F,T,F)]
indels$Alt <- unlist(strsplit(indels$mutationID, split=":"))[c(F,F,F,T)]
indels$Pos <- as.numeric(indels$Pos)
indels$end.pos <- indels$Pos
indels$end.pos[nchar(indels$Ref)>nchar(indels$Alt)] <- indels$Pos[nchar(indels$Ref)>nchar(indels$Alt)]+nchar(indels$Ref[nchar(indels$Ref)>nchar(indels$Alt)])-1

context_list <- indels  %>%
  dplyr::mutate(start = Pos - 3,
                end = Pos + 3) %>%
  dplyr::select(Chr, start,end)
this_range <- GenomicRanges::makeGRangesFromDataFrame(context_list)

this_seq <- getSeq(BSgenome.Hsapiens.UCSC.hg38, this_range)
out_seqs <- GenomicRanges::as.data.frame(this_seq)
out.table <- cbind(indels, out_seqs)

Grange_branches <- makeGRangesListFromDataFrame(indels, split.field = "patientID", keep.extra.columns = T, ignore.strand = T, seqnames.field = "Chr",
                                                start.field = "Pos", end.field = "end.pos")
GenomeInfoDb::genome(Grange_branches) = 'hg38'

indel_grl <- get_indel_context(Grange_branches, ref_genome)
indel_counts <- count_indel_contexts(indel_grl)
indel_counts <- data.frame(indel_counts)

patient_meta <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/manuscript_data_and_figures/Supplementary_Table1_final.txt", h=T)

exp <- indel_counts[,colnames(indel_counts) %in% patient_meta$Patient.ID[patient_meta$Shows_Psoralen_signature]]
nexp <- indel_counts[,!(colnames(indel_counts) %in% patient_meta$Patient.ID[patient_meta$Shows_Psoralen_signature])]

sums <- data.frame(Exposed=rowSums(exp), NonExposed=rowSums(nexp))

plot_indel_contexts(sums, condensed = TRUE) +
  theme_classic() +
  theme(legend.position = "bottom") + labs(fill="") + theme(legend.key.size = unit(0.25, "cm")) +
  theme(legend.box.spacing = unit(-10, "pt")) + scale_y_continuous(expand=c(0,0))
```



We see that the spectra are near-identical. For a formal comparison, we can calculate the cosine similarity of the probability vectors:

```
cos_sim(sums$Exposed, sums$NonExposed)
```

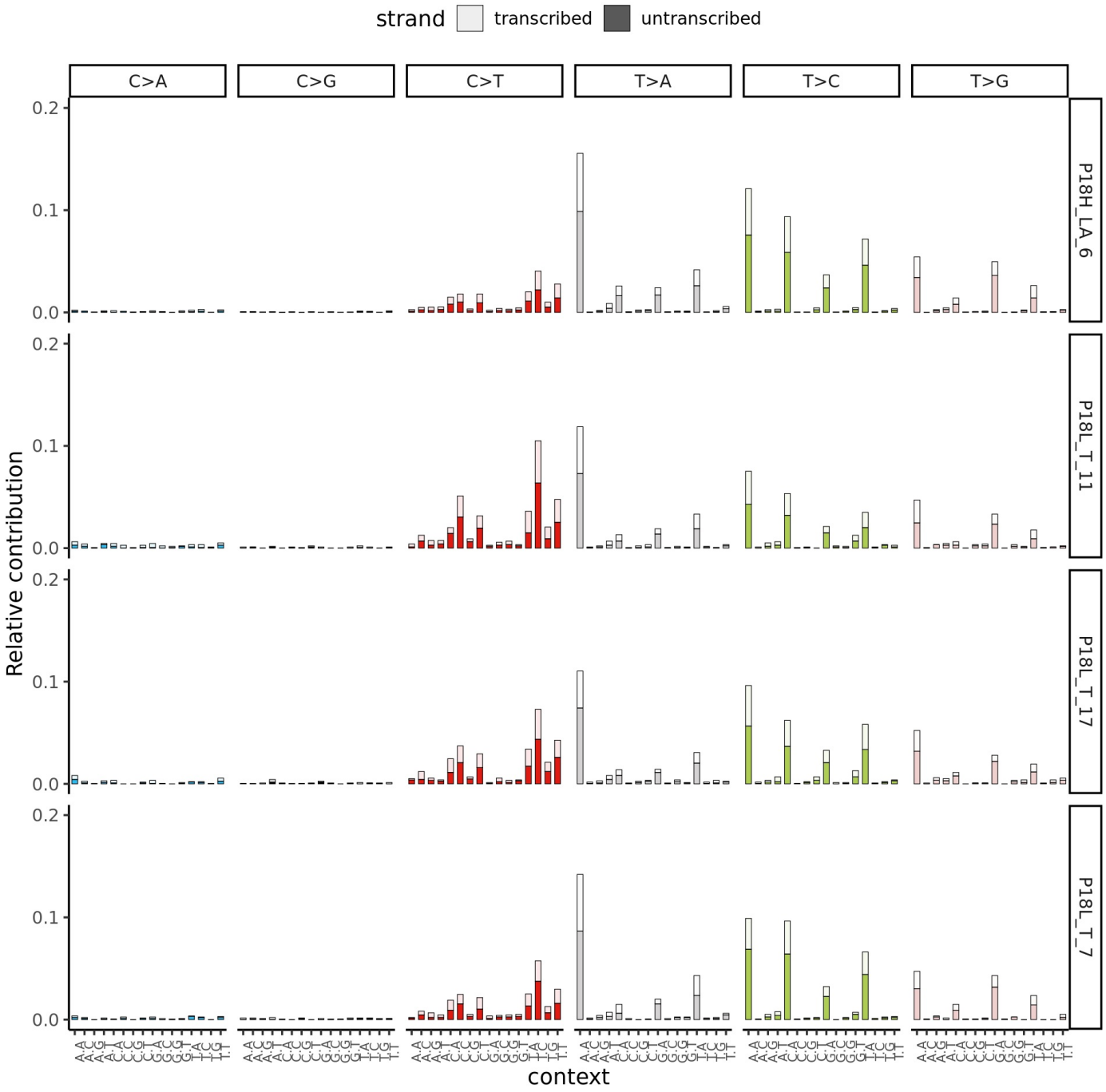
```
## [1] 0.9879375
```

Strand bias analyses

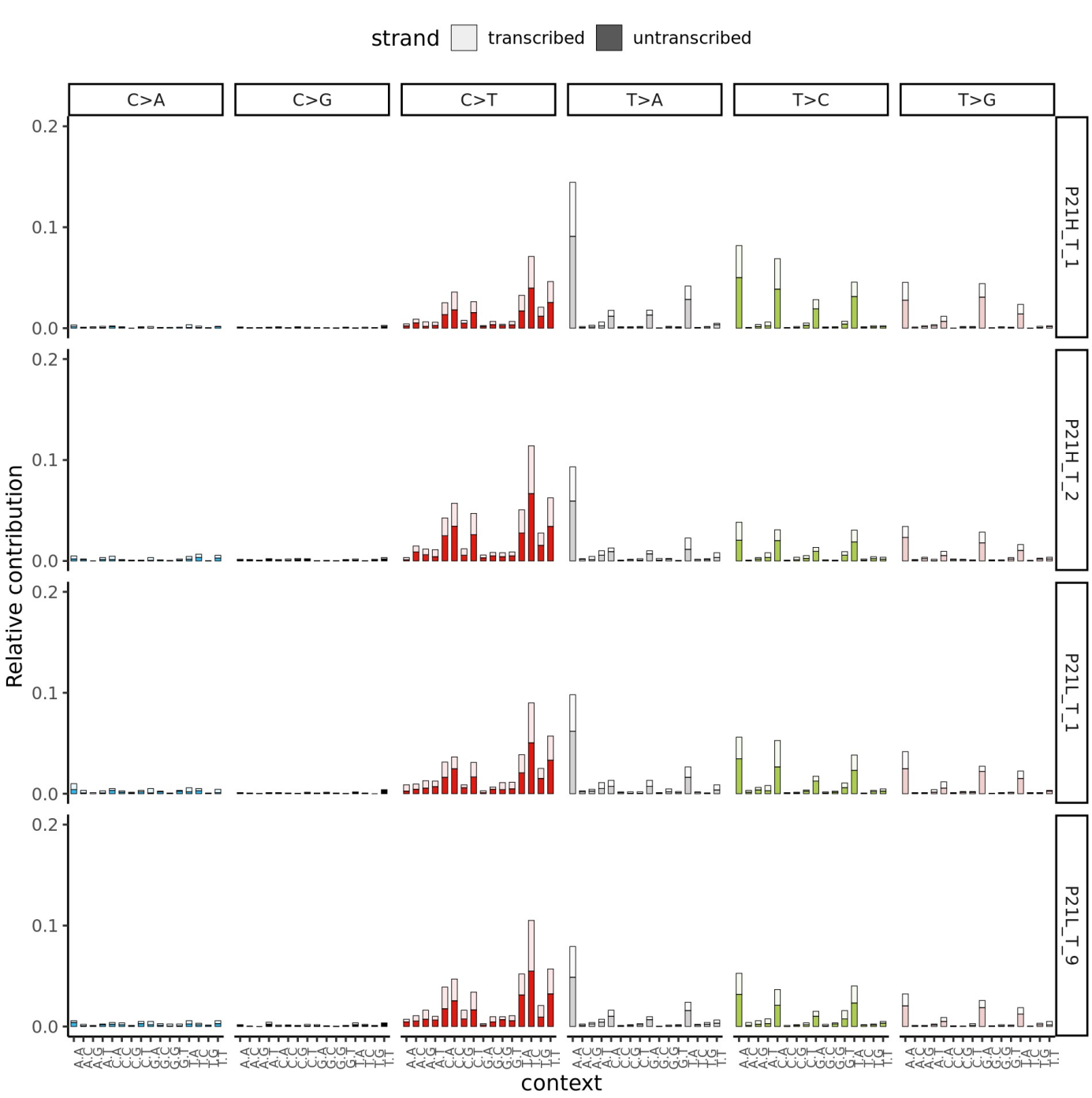
Transcriptional strand bias

We can start by plotting the mutation spectra of the samples as before but now with strand information. From the plot, it is obvious that there are many more T>[ACG] mutations on the untranscribed strand. This effect is much more pronounced than the known strand bias of UV-light exposure (C>T mutations below).

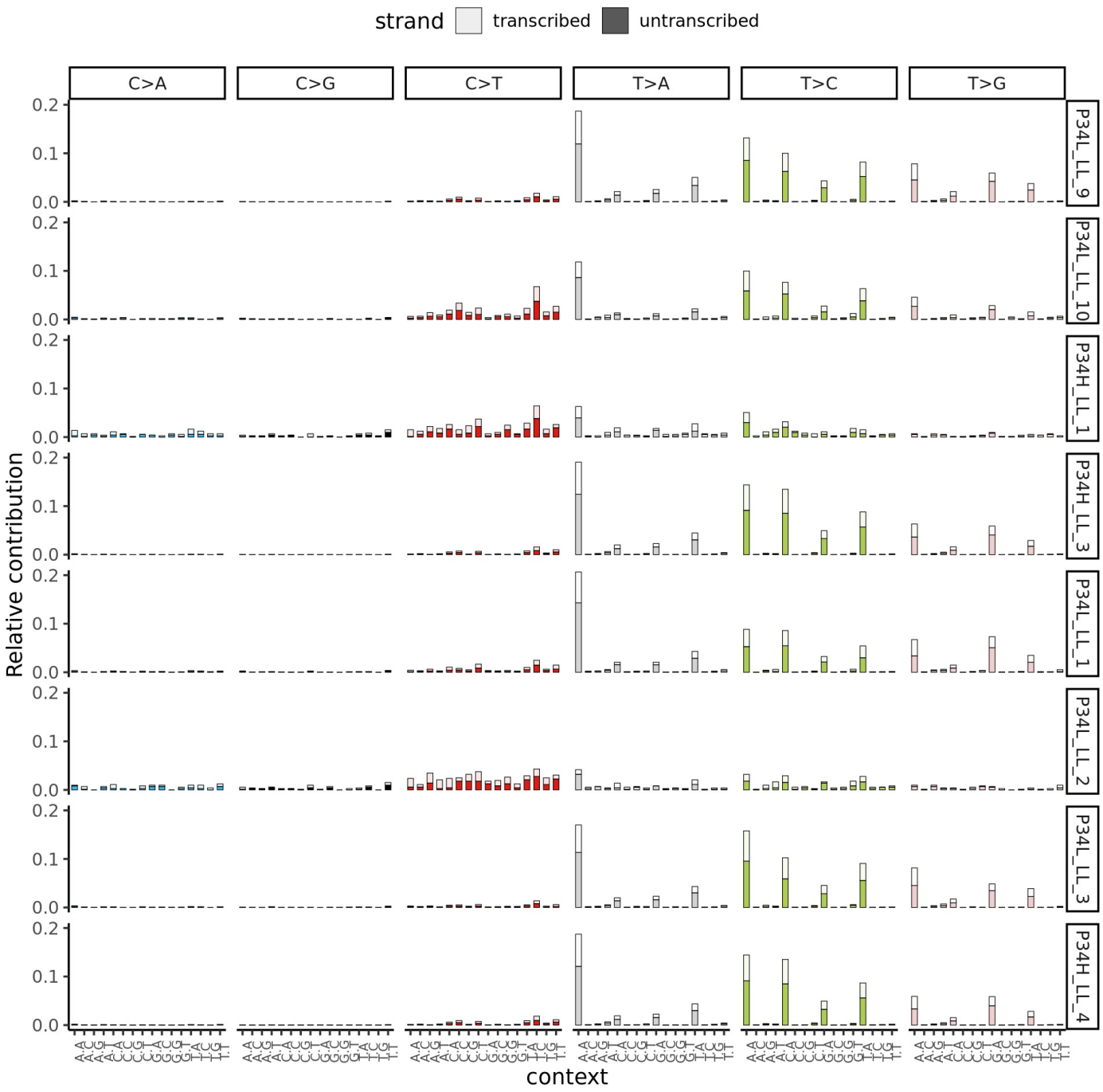
```
genes_hg38 <- genes(TxDb.Hsapiens.UCSC.hg38.knownGene)
mut_mat_s <- mut_matrix_stranded(grl, ref_genome, genes_hg38)
plot_192_profile(mut_mat_s[,1:4]) + theme_classic(base_size = 14) + theme(legend.position="top", axis.text.x = el
ement_text(angle=90, size=8))
```




```
plot_192_profile(mut_mat_s[,5:8]) + theme_classic(base_size = 14) + theme(legend.position="top", axis.text.x = el
ement_text(angle=90, size=8))
```




```
plot_192_profile(mut_mat_s[,9:16]) + theme_classic(base_size = 14) + theme(legend.position="top", axis.text.x = e
lement_text(angle=90, size=8))
```

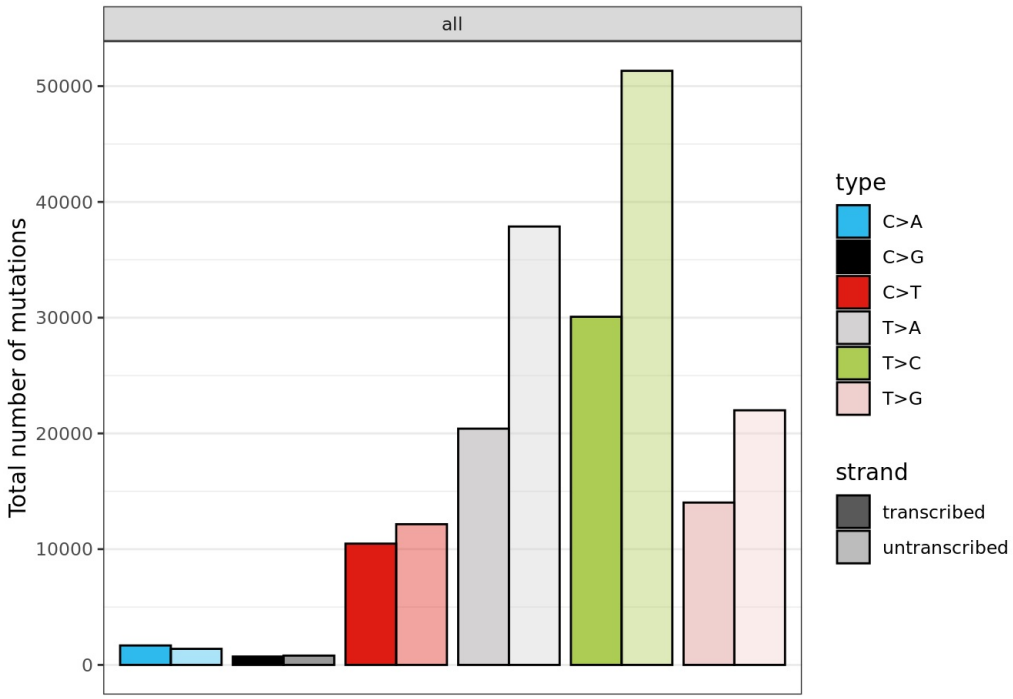


More formally, the strand_bias_test() function performs a two-sided Poisson test for the ratio between the number of mutation mapped on each strand. The difference is highly significant!

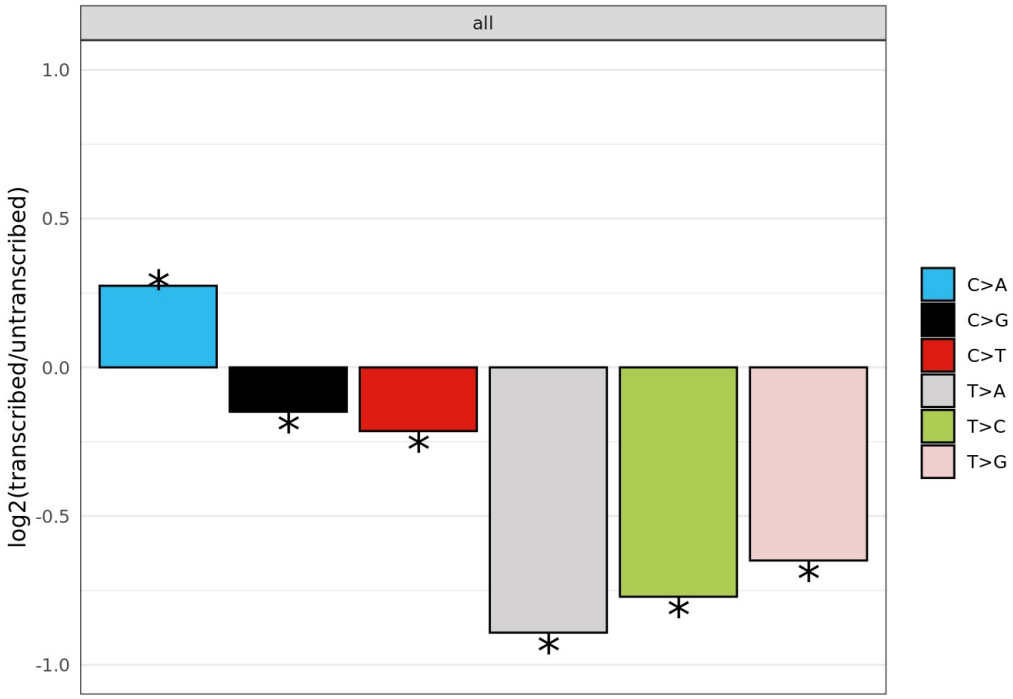
```
strand_counts <- strand_occurrences(mut_mat_s)
strand_bias <- strand_bias_test(strand_counts)
strand_bias
```

```
## # A tibble: 6 × 10
##   group type  transcribed untranscribed total ratio   p_poisson significant
##   <fct> <chr>       <dbl>         <dbl> <dbl> <dbl>       <dbl> <chr>
## 1 all   C>A          1680          1389  3069 1.21    1.62e-  7 *
## 2 all   C>G           725           804  1529 0.902   4.60e-  2 *
## 3 all   C>T         10478         12156 22634 0.862   7.01e- 29 *
## 4 all   T>A         20411         37874 58285 0.539 Inf.  e-324 *
## 5 all   T>C         30076         51325 81401 0.586 Inf.  e-324 *
## 6 all   T>G         14026         21998 36024 0.638 Inf.  e-324 *
## # … with 2 more variables: fdr <dbl>, significant_fdr <chr>
```

```
plot_strand(strand_counts, mode = "absolute")
```




```
plot_strand_bias(strand_bias, sig_type = "p")
```



Psoralen and transcription

To assess the relationship between psoralen-related mutagenesis and gene expression, we used gene expression data from sun-exposed skin from the GTEx project (https://gtexportal.org/home/%5D). The transcription start site of each gene we got from Gencode.

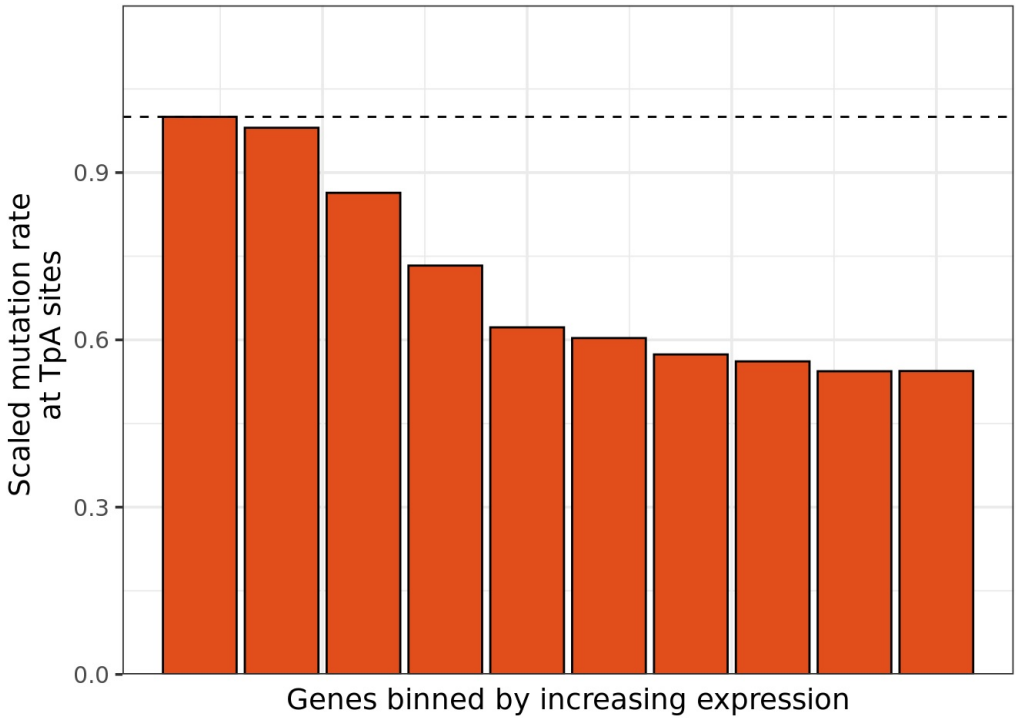
```
## First read in GTEx results and merge with gencode
gtex <- read.csv("/lustre/scratch126/humgen/projects/psoriasis/resources/GTEx_skin_expression.csv", h=T)
colnames(gtex) <- c("TranscriptID", "Gene", "Skin_no_sun", "Skin_sun")

tss<-read.table("/lustre/scratch126/humgen/projects/psoriasis/resources/gencode.v27.annotation_chr_pos_strand_gen
eid_gene_name.gtf",h=F)
colnames(tss)<-c("Chromosome_phe","START_GENE","END_GENE","STRAND","Phenotype_ID","HGNC")
tss$TSS<-ifelse(tss$STRAND=="+",tss$START_GENE-1,tss$END_GENE-1)
```

We next extracted only protein coding genes from the nuclear genome and divided them up into 10 equally sized bins in order of ascending expression.

```
comb <- merge(gtex, tss, by.x="Gene", by.y="HGNC")
comb <- comb[order(comb$Skin_sun, decreasing = T),]
comb <- comb[comb$Chromosome_phe!="chrM",]
d <- duplicated(comb$Gene)
comb <- comb[!d,]  ## Keeps the highest expressed transcript

## Create expression bins to check the mutation rate in bins of ascending
## expression
bins <- comb[,c("Chromosome_phe", "START_GENE", "END_GENE", "Gene", "Skin_sun")]

sel_cv <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/manuscript_data_and_figures/
sel_cv_dNdS_results.txt", h=T)
bins <- bins[bins$Gene %in% sel_cv$gene_name,]  ## Only consider protein coding genes.

bins$nr <- 1:nrow(bins)
bins$expr_bin <- cut(bins$nr, breaks=10)
bins$expr_bin <- factor(bins$expr_bin, labels=c("Bin10","Bin9","Bin8","Bin7","Bin6","Bin5","Bin4","Bin3","Bin2","
Bin1"))
bins$nr <- NULL
#write.table(bins, file="/lustre/scratch126/humgen/projects/psoriasis/signature_extraction/puva_characterization/
GTex_expression_bins.bed", sep="\t", col.names = F, row.names = F, quote = F)
```

To calculate the mutation rate in each bin, we used the script Psoralen_expression.sh, which is available in the GitHub repository which accompanies this manuscript (see the main text). Here, I read in and plot the results.

```
expression_res <- read.table("/lustre/scratch126/humgen/projects/psoriasis/signature_extraction/puva_characteriza
tion/Mutation_rate_pr_expr_bin.txt", h=T)
expression_res$scaled <- expression_res$Mutation_Rate/expression_res$Mutation_Rate[1]

ggplot(expression_res, aes(x=Bin, y=scaled)) + geom_bar(position="dodge",stat="identity", fill="#E24E1B", colour=
"black") +
  scale_y_continuous(expand=c(0,0), limits = c(0,1.2)) + theme_bw(base_size = 14) +
  labs(y="Scaled mutation rate \n at TpA sites", x="Genes binned by increasing expression") +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank()) +
  geom_hline(yintercept = 1, linetype=2)
```



Transcription coupled repair and damage

To check for evidence of transcription coupled repair and damage, we'll continue using the GTEx data above. The procedure is as follows:

i. Compile a list of genes highly expressed in the skin. Include strand information. ii) Create 1kb bins 10kb up and downstream of the TSSs. iii) Count the number of T>A, T>C and T>G mutations at TpA sites on the transcribed strand and the number of A>T, A>G and A>C mutations at ApT sites on the untranscribed strand. If the gene is on the (+) strand, the transcribed strand is the complement of the reference strand.

```
## Create expression quintiles:
expr_quintile1 <-comb[comb$Gene %in% bins$Gene[bins$expr_bin=="Bin1" | bins$expr_bin=="Bin2"],]
expr_quintile2 <-comb[comb$Gene %in% bins$Gene[bins$expr_bin=="Bin3" | bins$expr_bin=="Bin4"],]
expr_quintile3 <-comb[comb$Gene %in% bins$Gene[bins$expr_bin=="Bin5" | bins$expr_bin=="Bin6"],]
expr_quintile4 <-comb[comb$Gene %in% bins$Gene[bins$expr_bin=="Bin7" | bins$expr_bin=="Bin8"],]
expr_quintile5 <-comb[comb$Gene %in% bins$Gene[bins$expr_bin=="Bin9" | bins$expr_bin=="Bin10"],]
```

```
## Make a bed file for each quintile with all the information necessary for the analysis.
make_bed_file <- function(geneList) {

  toUse <- geneList
  ## For each of these genes, I want to write 20 entries to a bed file,
  ## ten 1kb segments upstream and downstream of the TSS.
  chr_vector <- as.character()
  start <- as.numeric()
  end <- as.numeric()
  tss <- as.character()
  strand <- as.character()
  label <- as.character()
  t <- 1
  for(i in 1:nrow(toUse)) {


    if(toUse$STRAND[i]=="-") {
      # Create 1kb bins upstream of the TSS
      for(j in 1:10) {
        chr_vector[t] <- toUse$Chromosome_phe[i]
        start[t] <- toUse$END_GENE[i] + (j-1)*1000
        end[t] <- toUse$END_GENE[i] + (j)*1000
        tss[t] <- "upstream"
        strand[t] <- "-"
        label[t] <- paste("-",j,"kb", sep="")
        t <- t+1
      }
      # Create 1kb bins downstream of the TSS
      for(k in 1:10) {
        chr_vector[t] <- toUse$Chromosome_phe[i]
        start[t] <- toUse$END_GENE[i] - (k)*1000
        end[t] <- toUse$END_GENE[i] - (k-1)*1000
        tss[t] <- "downstream"
        strand[t] <- "-"
        label[t] <- paste("+",k,"kb", sep="")
        t <- t+1
      }
    } else if(toUse$STRAND[i]=="+") {
      # Create 1kb bins upstream of the TSS
      for(j in 1:10) {
        chr_vector[t] <- toUse$Chromosome_phe[i]
        start[t] <- toUse$START_GENE[i] - (j)*1000
        end[t] <- toUse$START_GENE[i] - (j-1)*1000
        tss[t] <- "upstream"
        strand[t] <- "+"
        label[t] <- paste("-",j,"kb", sep="")
        t <- t+1
      }
      # Create 1kb bins downstream of the TSS
      for(k in 1:10) {
        chr_vector[t] <- toUse$Chromosome_phe[i]
        start[t] <- toUse$START_GENE[i] + (k-1)*1000
        end[t] <- toUse$START_GENE[i] + (k)*1000
        tss[t] <- "downstream"
        strand[t] <- "+"
        label[t] <- paste("+",k,"kb", sep="")
        t <- t+1
      }
    } else {
      stop("Strand info missing")
    }
  }

  bed_file <- data.frame(chr_vector, start, end, tss, strand, label)
  return(bed_file)
}

bed_q1 <- make_bed_file(expr_quintile1)
bed_q2 <- make_bed_file(expr_quintile2)
bed_q3 <- make_bed_file(expr_quintile3)
bed_q4 <- make_bed_file(expr_quintile4)
bed_q5 <- make_bed_file(expr_quintile5)

#write.table(bed_q5,file = "/lustre/scratch126/humgen/projects/psoriasis/signature_extraction/puva_characterizati
on/quintile5_expr_genes_skin.bed",  sep="\t", quote = F, row.names = F, col.names = F)
```

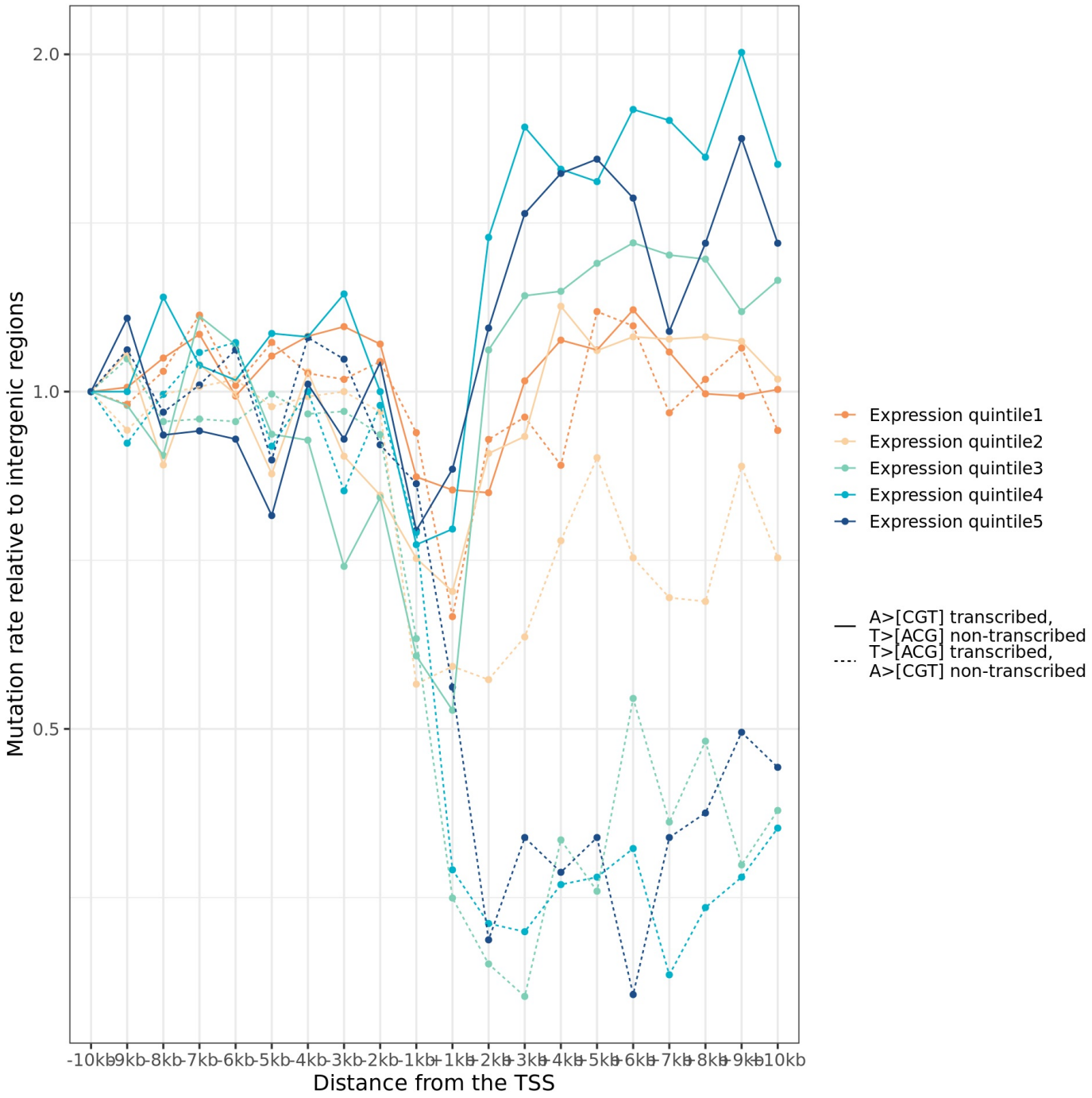
I next use the psoralen_tcd.sh script, available on the Github page accompanying the manuscript (see main text), to count the number of mutations in each bin.

We can then read in and plot the results:

```
for(q in c(1:5)) {
  assign(paste("quintile", q, "_Tmut_transcribed", sep=""),
         read.table(paste("/lustre/scratch126/humgen/projects/psoriasis/signature_extraction/puva_characterizatio
n/quintile", q, "_Tmut_transc.bed", sep="")))

  assign(paste("quintile", q, "_Amut_transcribed", sep=""),
         read.table(paste("/lustre/scratch126/humgen/projects/psoriasis/signature_extraction/puva_characterizatio
n/quintile", q, "_Amut_transc.bed", sep="")))

}

quintile1_Amut_transcribed$quintile=quintile1_Tmut_transcribed$quintile <- "Expression quintile1"
quintile2_Amut_transcribed$quintile=quintile2_Tmut_transcribed$quintile <- "Expression quintile2"
quintile3_Amut_transcribed$quintile=quintile3_Tmut_transcribed$quintile <- "Expression quintile3"
quintile4_Amut_transcribed$quintile=quintile4_Tmut_transcribed$quintile <- "Expression quintile4"
quintile5_Amut_transcribed$quintile=quintile5_Tmut_transcribed$quintile <- "Expression quintile5"

quintile1_Amut_transcribed$type=quintile2_Amut_transcribed$type=quintile3_Amut_transcribed$type=quintile4_Amut_tr
anscribed$type=quintile5_Amut_transcribed$type <- "A>[CGT] transcribed,\nT>[ACG] non-transcribed"
quintile1_Tmut_transcribed$type=quintile2_Tmut_transcribed$type=quintile3_Tmut_transcribed$type=quintile4_Tmut_tr
anscribed$type=quintile5_Tmut_transcribed$type <- "T>[ACG] transcribed,\nA>[CGT] non-transcribed"

comb <- rbind(quintile1_Amut_transcribed,quintile1_Tmut_transcribed,quintile2_Amut_transcribed,quintile2_Tmut_tra
nscribed,
              quintile3_Amut_transcribed,quintile3_Tmut_transcribed,quintile4_Amut_transcribed,
              quintile4_Tmut_transcribed,quintile5_Amut_transcribed,quintile5_Tmut_transcribed)

comb$V6 <- factor(comb$V6, levels=c("-10kb", "-9kb", "-8kb", "-7kb", "-6kb", "-5kb", "-4kb", "-3kb", "-2kb", "-1k
b",
                                    "+1kb", "+2kb", "+3kb", "+4kb", "+5kb", "+6kb", "+7kb", "+8kb", "+9kb", "+10k
b"))

m <- melt(data.frame(table(comb$V6, comb$quintile, comb$type)))
```

```
## Using Var1, Var2, Var3 as id variables
```

```
m$variable <- NULL
colnames(m) <- c("DistanceFromTSS","expression", "MutationType", "MutationCount")

standardized <- as.numeric()
for(i in 1:nrow(m)) {
  denom <- m$MutationCount[m$DistanceFromTSS=="-10kb" & m$MutationType==m$MutationType[i] & m$expression==m$expre
ssion[i]]
  standardized[i] <- m$MutationCount[i]/denom
}
m$standardized <- standardized

m$class <- paste(m$expression, m$MutationType)

#write.table(m, "/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/manuscript_data_and_figures/puva_
tcd.txt", row.names = F, sep="\t")

ggplot(m, aes(x=DistanceFromTSS, y=standardized, colour=expression)) + geom_point() + theme_bw(base_size = 14) +
  geom_line(aes(x=DistanceFromTSS,y=standardized, color = expression, group = class, linetype=MutationType)) +
  labs(x="Distance from the TSS", y="Mutation rate relative to intergenic regions", colour="", linetype="") +
  scale_y_continuous(trans="log2") + scale_colour_manual(values=c("#F79256","#FBD1A2", "#7DCFB6","#00B2CA","#1D4E
89"))
```



To test if the lines significantly depart from a straight line, we'll use a step function. We see that the effect of transcription coupled repair is highly significant from quintile 2 onwards and the effect of transcription coupled damage is highly significant from quintile 3 onwards.

```
counts <- rbind(data.frame(table(quintile1_Amut_transcribed$V6), quintile="Q1"),
                data.frame(table(quintile2_Amut_transcribed$V6), quintile="Q2"),
                data.frame(table(quintile3_Amut_transcribed$V6), quintile="Q3"),
                data.frame(table(quintile4_Amut_transcribed$V6), quintile="Q4"),
                data.frame(table(quintile5_Amut_transcribed$V6), quintile="Q5"))

counts$upstream_TSS <- 1
counts$upstream_TSS[grep("-", as.character(counts$Var1))] <- 0

P_vector_tcd <- as.numeric()
for(i in 1:5) {
  null_model <- lm(Freq ~ 1, data=counts[counts$quintile==paste("Q",i, sep=""),])
  alt_model <- lm(Freq ~ I(counts$upstream_TSS[counts$quintile==paste("Q",i, sep="")]), data=counts[counts$quinti
le==paste("Q",i, sep=""),])

  P_vector_tcd[i] <- anova(null_model, alt_model)$"Pr(>F)"[2]
}

counts <- rbind(data.frame(table(quintile1_Tmut_transcribed$V6), quintile="Q1"),
                data.frame(table(quintile2_Tmut_transcribed$V6), quintile="Q2"),
                data.frame(table(quintile3_Tmut_transcribed$V6), quintile="Q3"),
                data.frame(table(quintile4_Tmut_transcribed$V6), quintile="Q4"),
                data.frame(table(quintile5_Tmut_transcribed$V6), quintile="Q5"))

counts$upstream_TSS <- 1
counts$upstream_TSS[grep("-", as.character(counts$Var1))] <- 0

P_vector_tcr <- as.numeric()
for(i in 1:5) {
  null_model <- lm(Freq ~ 1, data=counts[counts$quintile==paste("Q",i, sep=""),])
  alt_model <- lm(Freq ~ I(counts$upstream_TSS[counts$quintile==paste("Q",i, sep="")]), data=counts[counts$quinti
le==paste("Q",i, sep=""),])

  P_vector_tcr[i] <- anova(null_model, alt_model)$"Pr(>F)"[2]
}
P_vector_tcr
```

```
## [1] 2.272361e-01 3.287565e-04 7.139100e-10 4.139985e-12 2.352863e-11
```

```
data.frame(quintile=paste("Q", c(1:5), sep=""), P_tcd=P_vector_tcd, P_tcr=P_vector_tcr)
```

```
##   quintile        P_tcd        P_tcr
## 1       Q1 0.4522413698 2.272361e-01
## 2       Q2 0.1555378104 3.287565e-04
## 3       Q3 0.0095397381 7.139100e-10
## 4       Q4 0.0002376581 4.139985e-12
## 5       Q5 0.0001896520 2.352863e-11
```

Replication strand bias

Next I want to see if there is evidence of replication strand bias. This is a bit trickier than the transcription strand bias, as the replication origins of the human genome are incompletely known. Nevertheless, there are some regions which are known to be conserved (see Methods).

Here I use a file containing replication time data from http://www.cell.com/abstract/S0092-8674(15)01714-6 (http://www.cell.com/abstract/S0092-8674(15)01714-6). In the file, left is synonymous to the leading strand because we are always using the reference as a point of reference. As shown below, there is an enrichment of Psoralen related mutations on the leading (left) strand.

```
repli_timing <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/07_signature_extractio
n/strand_asymmetries/replication_timing.repdirhg38.txt")
repli_timing$Class <- NA
repli_timing$Class[repli_timing$V5==1] <- "left"
repli_timing$Class[repli_timing$V6==1] <- "right"

repli_strand <- repli_timing[!is.na(repli_timing$Class), c("V1", "V2", "V3", "Class")]
repli_strand$Ratio <- 0
colnames(repli_strand)=c("Chr", "Start", "Stop", "Class", "Ratio")

# Store in GRanges object
repli_strand_granges <- GRanges(
  seqnames = repli_strand$Chr,
  ranges = IRanges(
    start = repli_strand$Start + 1,
    end = repli_strand$Stop
  ),
  strand_info = repli_strand$Class
)
# UCSC seqlevelsstyle
seqlevelsStyle(repli_strand_granges) <- "UCSC"
repli_strand_granges
```

```
## GRanges object with 10611 ranges and 1 metadata column:
##           seqnames            ranges strand | strand_info
##              <Rle>         <IRanges>  <Rle> | <character>
##       [1]     chr1   1164622-1264620      * |        left
##       [2]     chr1   1464622-1564620      * |       right
##       [3]     chr1   2368563-2468561      * |       right
##       [4]     chr1   2468563-2568561      * |       right
##       [5]     chr1   2568563-2668561      * |       right
##       ...      ...               ...    ... .         ...
##   [10607]    chr22 49406353-49506351      * |        left
##   [10608]    chr22 49506353-49606352      * |        left
##   [10609]    chr22 49606354-49706352      * |        left
##   [10610]    chr22 49706354-49806352      * |        left
##   [10611]    chr22 49806354-49906352      * |        left
##   -------
##   seqinfo: 23 sequences from an unspecified genome; no seqlengths
```

```
repli_strand_granges$strand_info <- factor(repli_strand_granges$strand_info,
                                           levels = c("right", "left")
)


mut_mat_s_rep <- mut_matrix_stranded(grl, ref_genome, repli_strand_granges,
                                     mode = "replication"
)
```

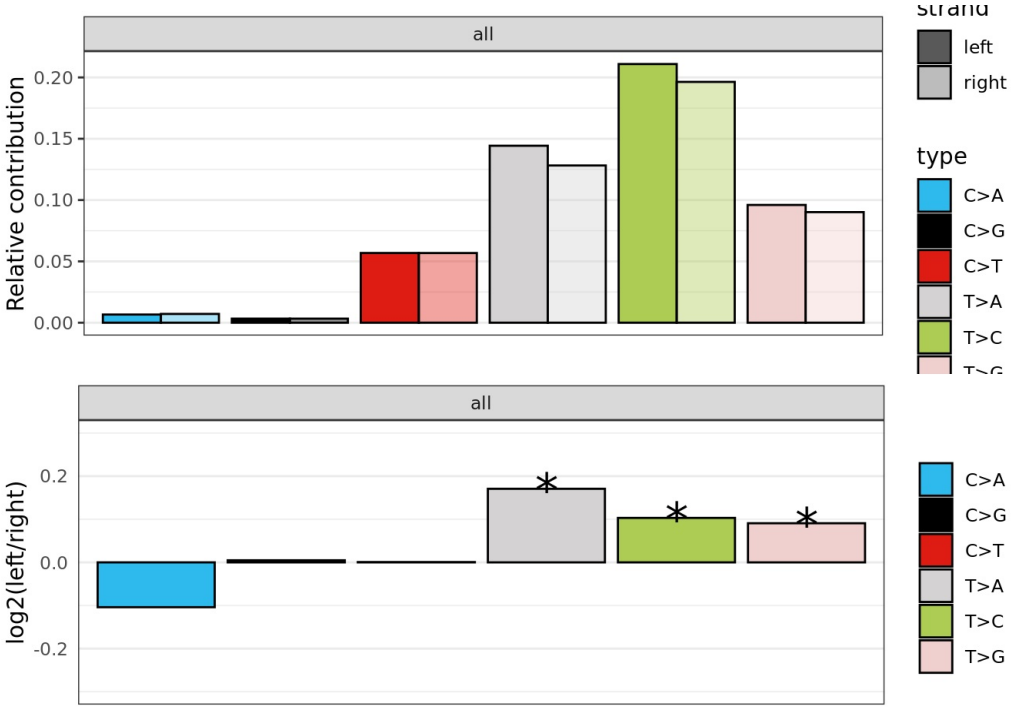
```
## Warning in .Seqinfo.mergexy(x, y): Each of the 2 combined objects has sequence levels not in the other:
##   - in 'x': chrX, chrY
##   - in 'y': chr7_KI270803v1_alt
##   Make sure to always combine/compare objects based on the same reference
##   genome (use suppressWarnings() to suppress this warning).
```

```
## Warning in mut_strand(gr, ranges, mode = mode): Some variants overlap with multiple genomic regions in the GRa
nges object.
##
##               These variants are assigned '-', as the strand cannot be determined.
##
##               To avoid this, make sure no genomic regions are overlapping in your GRanges object.
```

```
strand_counts_rep <- strand_occurrences(mut_mat_s_rep)
strand_bias_rep <- strand_bias_test(strand_counts_rep)
strand_bias_rep
```

```
## # A tibble: 6 × 10
##   group type   left right total ratio p_poisson significant      fdr
##   <fct> <chr> <dbl> <dbl> <dbl> <dbl>     <dbl> <chr>          <dbl>
## 1 all   C>A    1204  1294  2498 0.930  7.49e- 2 ""          1.12e- 1
## 2 all   C>G     600   598  1198 1.00   9.77e- 1 ""          9.78e- 1
## 3 all   C>T   10298 10293 20591 1.00   9.78e- 1 ""          9.78e- 1
## 4 all   T>A   26150 23232 49382 1.13   2.21e-39 "*"         1.32e-38
## 5 all   T>C   38220 35583 73803 1.07   2.90e-22 "*"         8.69e-22
## 6 all   T>G   17399 16338 33737 1.06   7.86e- 9 "*"         1.57e- 8
## # … with 1 more variable: significant_fdr <chr>
```

```
ps1 <- plot_strand(strand_counts_rep, mode = "relative")
ps2 <- plot_strand_bias(strand_bias_rep)
grid.arrange(ps1, ps2)
```



As an additional sanity check, we can use a different replication origin file, the one that comes with the MutationalPatterns package. This file is in hg19 so I first did a liftOver to convert the coordinates to hg38. We see broadly consistent results using this second source of replication origin information.

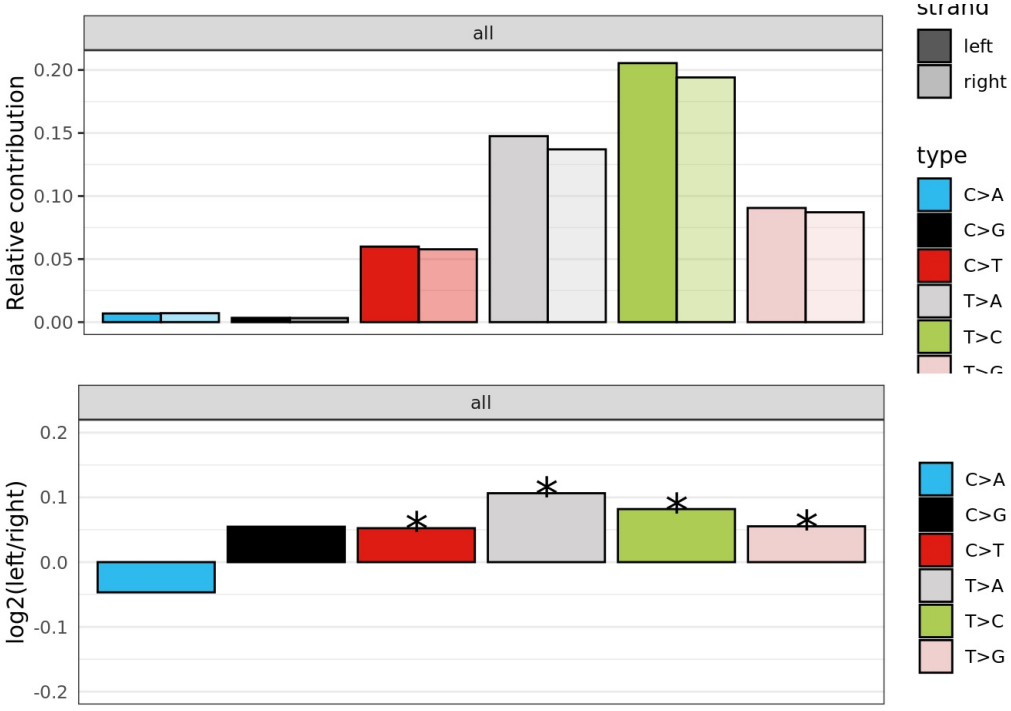
```
repli_strand <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/07_signature_extractio
n/strand_asymmetries/MutationalPatterns_repli_strandhg38.txt", h=T)


# Store in GRanges object
repli_strand_granges <- GRanges(
  seqnames = repli_strand$Chr,
  ranges = IRanges(
    start = repli_strand$Start + 1,
    end = repli_strand$Stop
  ),
  strand_info = repli_strand$Class
)
# UCSC seqlevelsstyle
seqlevelsStyle(repli_strand_granges) <- "UCSC"
repli_strand_granges
```

```
## GRanges object with 1936 ranges and 1 metadata column:
##          seqnames            ranges strand | strand_info
##             <Rle>         <IRanges>  <Rle> | <character>
##      [1]     chr1   3172437-3580436      * |        left
##      [2]     chr1   5162941-6367940      * |        left
##      [3]     chr1   6367941-7263940      * |       right
##      [4]     chr1   7263941-9551942      * |        left
##      [5]     chr1 11434944-12196943      * |        left
##      ...      ...               ...    ... .         ...
##   [1932]     chrY 21577115-21659114      * |       right
##   [1933]     chrY 21850854-22277853      * |       right
##   [1934]     chrY 22277854-26489853      * |        left
##   [1935]     chrY 26489854-26539853      * |       right
##   [1936]     chrY 26539854-26613853      * |        left
##   -------
##   seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

```
repli_strand_granges$strand_info <- factor(repli_strand_granges$strand_info,
                                           levels = c("right", "left")
)


mut_mat_s_rep <- mut_matrix_stranded(grl, ref_genome, repli_strand_granges,
                                     mode = "replication"
)
strand_counts_rep <- strand_occurrences(mut_mat_s_rep)
strand_bias_rep <- strand_bias_test(strand_counts_rep)
strand_bias_rep
```

```
## # A tibble: 6 × 10
##   group type   left right total ratio p_poisson significant      fdr
##   <fct> <chr> <dbl> <dbl> <dbl> <dbl>     <dbl> <chr>          <dbl>
## 1 all   C>A    1247  1288  2535 0.968  4.27e- 1 ""          5.12e- 1
## 2 all   C>G     618   595  1213 1.04   5.28e- 1 ""          5.28e- 1
## 3 all   C>T   10901 10512 21413 1.04   8.01e- 3 "*"         1.20e- 2
## 4 all   T>A   26847 24938 51785 1.08   5.05e-17 "*"         3.03e-16
## 5 all   T>C   37389 35325 72714 1.06   1.99e-14 "*"         5.98e-14
## 6 all   T>G   16480 15860 32340 1.04   5.77e- 4 "*"         1.15e- 3
## # … with 1 more variable: significant_fdr <chr>
```

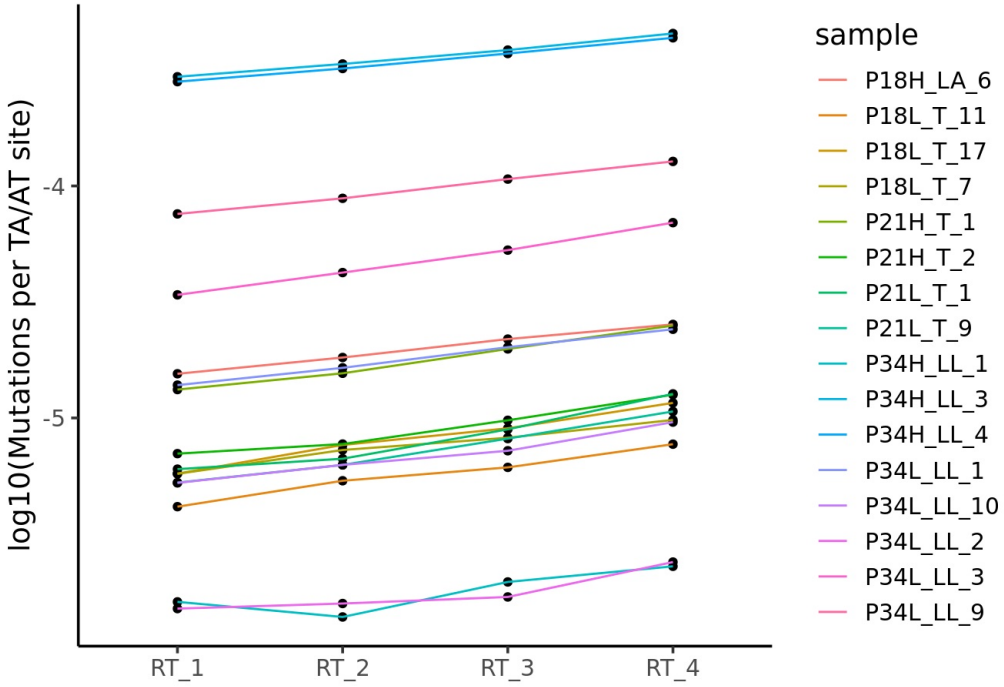
```
ps1 <- plot_strand(strand_counts_rep, mode = "relative")
ps2 <- plot_strand_bias(strand_bias_rep)
grid.arrange(ps1, ps2)
```



Replication timing analysis

To see if replication timing affects the mutation rate of psoralen exposure, we estimate the mutation rate after dividing the genome up into four bins of ascending replication timing. The code for doing this analysis is available in puva_replication_timing.sh in the github repository that accompanies this manuscript (see main text). Here I just read in and plot the results.

```
rt_results <- data.frame()
for(sample in sample_names) {
  rt <- read.table(paste(puva_dir, sample, "_RT_results.txt", sep=""))
  rt$sample <- sample
  rt_results <- rbind(rt_results, rt)
}

ggplot(rt_results, aes(x=V1, y=log10(V2))) +geom_point() +
  geom_line(aes(group=sample, colour=sample)) + theme_classic(base_size = 14) +
  labs(y="log10(Mutations per TA/AT site)", x="")
```



Effect of psoralen exposure on the clonal structure of the tissue

Psoralen exposure (and PUVA treatment in particular) results in interstrand cross links that are toxic to cells. It is possible that high exposure may result in differences in the clonal composition of the tissue and possibly different selection pressures compared with unexposed skin.

We test for a difference in clonality first on the level of individual microbiopsies.

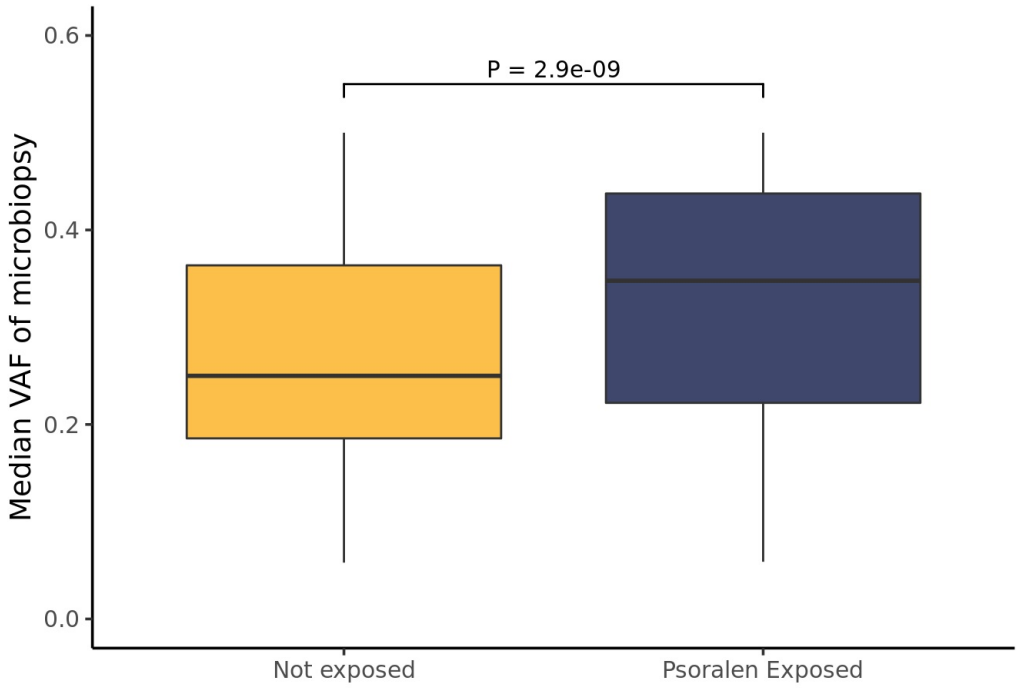
```
puva_colours <- c("#FCBF49", "#40476D")
clone_sigs <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/manuscript_data_and_figu
res/Supplementary_material/Supplementary_Table3_clone_mutationBurden.txt", h=T)
microd_meta <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/manuscript_data_and_fig
ures/Supplementary_material/Supplementary_Table2_microdissection_metadata.txt", h=T)
x <- clone_sigs[clone_sigs$PUVA>100,]
puva_exp_patients <- unique(unlist(strsplit(x$CloneID, spli="_"))[c(T,F)])
microd_meta$PUVA_exposed <- F
microd_meta$PUVA_exposed[microd_meta$PatientID %in% puva_exp_patients] <- T

puva_vaf_test <- wilcox.test(microd_meta$MedianVAF~microd_meta$PUVA_exposed)
puva_vaf_test
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  microd_meta$MedianVAF by microd_meta$PUVA_exposed
## W = 66641, p-value = 2.88e-09
## alternative hypothesis: true location shift is not equal to 0
```

```
ggplot(microd_meta, aes(x=PUVA_exposed, y=MedianVAF)) + geom_boxplot(aes(fill=PUVA_exposed)) +
  theme_classic(base_size = 14) + labs(x="", y="Median VAF of microbiopsy") +
  theme(legend.position = "none") +
  scale_x_discrete(labels=c("Not exposed", "Psoralen Exposed")) +
  scale_fill_manual(values=puva_colours) + scale_y_continuous(limits = c(0,0.6)) +
  geom_signif(annotation = paste("P = ", formatC(puva_vaf_test$p.value, digits=2), sep=""),
              y_position = c(0.55), xmin = c(1), xmax = c(2),
              tip_length = c(0.03, 0.03))
```

```
## Warning: Removed 8 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 8 rows containing non-finite values (stat_signif).
```



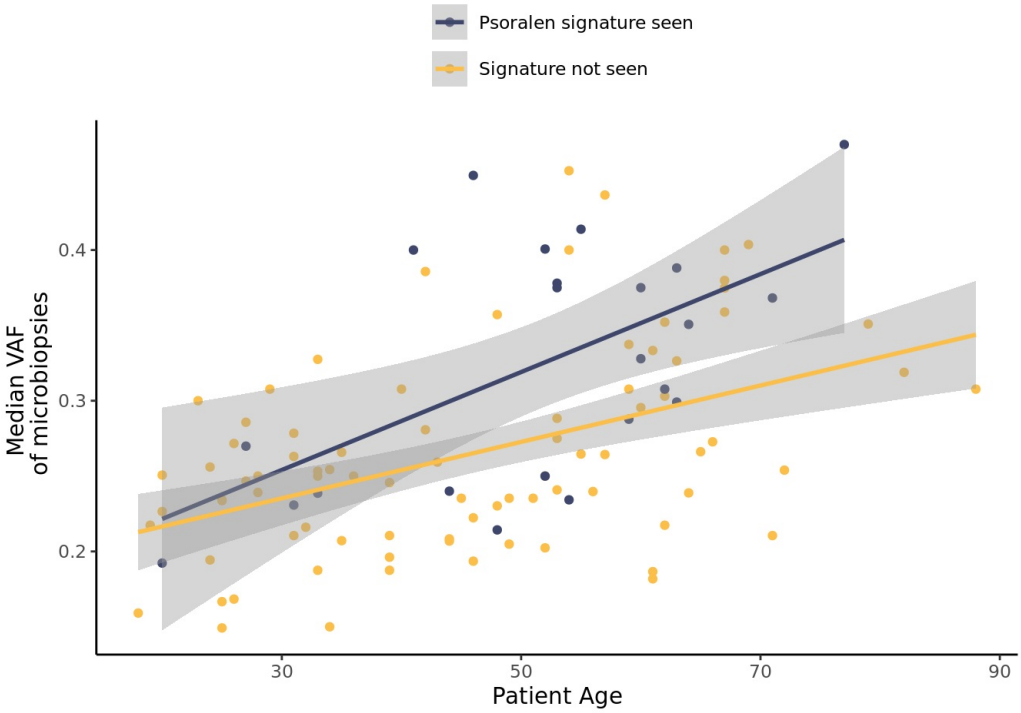
However, as pointed out by one of the reviewers, the patients who have received PUVA treatment are likely to be older than those who have not. As clones increase in size with age, it is important to take the age of the patient into account. We test if including a binary variable for the presence/absence of the psoralen signature significantly improves the fit of a linear model between mVAF and age.

```
microd_meta <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/manuscript_data_and_fig
ures/Microdissection_metaData.txt", h=T)

mVAFs <- data.frame(tapply(microd_meta$MedianVAF, microd_meta$PatientID, median, na.rm=T))
mVAFs$PatientID <- rownames(mVAFs)
colnames(mVAFs)[1] <- "MedianVAF"
patient_meta <- merge(patient_meta, mVAFs, by.x="Patient.ID", by.y="PatientID")
patient_meta$labels <- ifelse(patient_meta$Shows_Psoralen_signature, "Psoralen signature seen", "Signature not se
en")

puva_colours <- c("#FCBF49", "#40476D")

ggplot(patient_meta, aes(x=Age_at_sampling, y=MedianVAF, colour=labels)) + geom_point() +
  geom_smooth(method="lm") + theme_classic() +
  labs(y="Median VAF \n of microbiopsies", x="Patient Age", colour="") +
  theme(legend.position = "top") + scale_colour_manual(values=rev(puva_colours)) +
  guides(colour=guide_legend(nrow=2,byrow=TRUE))
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 5 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 5 rows containing missing values (geom_point).
```




```
## Do the likelihood ratio test:
model_null <- lm(MedianVAF~Age_at_sampling, data=patient_meta)
model_null_psoralen <- lm(MedianVAF~Age_at_sampling + Shows_Psoralen_signature, data=patient_meta)
anova(model_null,model_null_psoralen)
```

```
## Analysis of Variance Table
##
## Model 1: MedianVAF ~ Age_at_sampling
## Model 2: MedianVAF ~ Age_at_sampling + Shows_Psoralen_signature
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1    104 0.4322
## 2    103 0.3928  1  0.039402 10.332 0.001746 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Signature potential damage analysis

We would like to know if the psoralen signature is especially likely to cause any particular type of mutation in genes that are known drivers in squamous cell carcinomas or in normal skin.

Start by defining a list of entrez-gene IDs for the genes of interest.

```
## Genes mutated in SCCs and Normal skin:
# ARID2, ASXL1, CASP8, CDKN2A, FAT1, KMT2D, NOTCH1, NOTCH2, PPM1D, RB1, RBM10, TP53, TP63, AJUBA, NOTCH3
gene_ids <- c(196528,171023,841,1029,2195,8085,4851,4853,8493,5925,8241,7157,8626,84962,4854)
txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene

all_sigs <- read.csv("/lustre/scratch126/humgen/projects/psoriasis/resources/sigProfiler_SBS_signatures_summing_t
o_one.csv", h=T,row.names = 1, stringsAsFactors = F)
psoralen_component <- read.table("/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/manuscript_data_
and_figures/Supplementary_material/Supplementary_Table5_Psoralen_signature_hdp_component.txt", h=T)
colnames(psoralen_component) <- "Psoralen"
signatures <- data.frame(cbind(psoralen_component, all_sigs))
```

The signature_potential_damage_analysis() function in MutationalPatterns gives an estimate of the enrichment of particular mutations relative to a uniform mutational process. However, the mutational spectrum of the skin is not uniform and I would also like to give this relative to SBS7b, which is the dominant mutational process in our study. Need to make minor changes to the function.

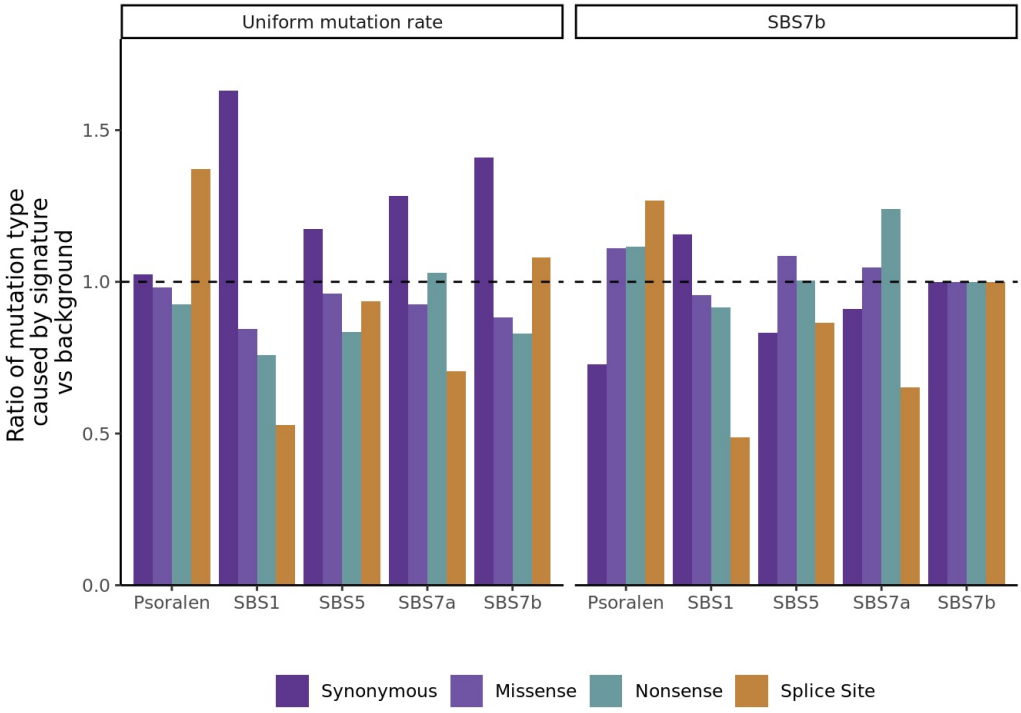
```
signature_potential_damage_analysis_sbs7b_ref <- function (signatures, contexts, context_mismatches)
{
  context <- ratio <- contribution <- sig <- flat <- n <- NULL
  ratio_by_background <- flat_ratio <- flat_blosum62 <- NULL
  blosum62 <- blosum62_min_background <- NULL
  signatures <- as.data.frame(signatures)
  signatures$context <- contexts
  nr_features <- nrow(signatures)
  signatures$flat_background <- rep(1/nr_features, nr_features)
  sig_context_mismatch <- signatures %>% dplyr::full_join(context_mismatches,
                                                          by = "context") %>% tidyr::pivot_longer(cols = c(-conte
xt,
                                                                                                  -type,
-ratio, -n, -blosum62), names_to = "sig", values_to = "contribution")
  sig_mismatch <- sig_context_mismatch %>% dplyr::mutate(ratio = contribution *
                                                           ratio, n = contribution * n, blosum62 = contribution *
                                                           blosum62) %>% dplyr::group_by(type, sig) %>% dplyr::su
mmarise(n = sum(n),

ratio = sum(ratio), blosum62 = sum(blosum62), .groups = "drop_last") %>%
    dplyr::ungroup()

  sbs7b_sig <- sig_mismatch %>% dplyr::filter(sig == "SBS7b") %>%
    dplyr::select(type, flat_ratio = ratio, flat_blosum62 = blosum62)

  norm_sig_mismatch <- sig_mismatch %>% dplyr::full_join(sbs7b_sig,
                                                         by = "type") %>% dplyr::mutate(ratio_by_background = rat
io/flat_ratio,
                                                                                        blosum62_min_background =
blosum62 - flat_blosum62) %>%
    dplyr::select(type, sig, ratio, ratio_by_background,
                  n, blosum62, blosum62_min_background) %>% dplyr::filter(sig !=
                                                                            "flat_background") %>% dplyr::arrange
(sig)
  return(norm_sig_mismatch)
}
```

Now we can look at the major signatures in the skin and plot the results.

```
contexts <- rownames(mut_mat)
context_mismatches <- context_potential_damage_analysis(contexts, txdb, ref_genome, gene_ids)
```

```
## Warning in .set_group_names(grl, use.names, txdb, by): some group names are NAs
## or duplicated
```

```
sig_damage_uniform <- signature_potential_damage_analysis(signatures[, c("SBS1", "SBS5","SBS7b","SBS7a","Psoralen
")], contexts, context_mismatches)
sig_damage_sbs7b <- signature_potential_damage_analysis_sbs7b_ref(signatures[, c("SBS1", "SBS5","SBS7b","SBS7a","
Psoralen")], contexts, context_mismatches)


sig_damage_uniform$Reference <- "Uniform mutation rate"
sig_damage_sbs7b$Reference <- "SBS7b"
sig_damage <- rbind(sig_damage_uniform, sig_damage_sbs7b)
sig_damage$sig[sig_damage$sig=="PUVA"] <- "Psoralen"

sig_damage$type <- factor(sig_damage$type,
                          levels=c("Synonymous", "Missense", "Stop_gain", "splice_site"),
                          labels=c("Synonymous", "Missense", "Nonsense", "Splice Site"))

sig_damage$Reference <- factor(sig_damage$Reference, levels=c("Uniform mutation rate", "SBS7b"))

write.table(sig_damage, "/nfs/users/nfs_s/so11/phd/psoriasis/bsub_jupyter_lab/psoriasis/manuscript_data_and_figur
es/signature_potential_damage.txt", sep="\t", quote=F)

## Nonsense,Splice, Missense, Indels, synonymous
annotation_cols <- c("#5D378E","#7155A5","#6A9A9E","#C0843E","#A9ACAB")

ggplot(sig_damage, aes(x=sig, y=ratio_by_background, fill=type)) +
  geom_bar(position="dodge", stat="identity") + theme_classic()+
  geom_hline(yintercept = 1, linetype="dashed") +
  scale_y_continuous(expand=c(0,0), limits = c(0,1.8)) + labs(x="", y="Ratio of mutation type \n caused by signat
ure \n vs background ", fill="") + facet_wrap(~Reference) +
  scale_fill_manual(values=annotation_cols) + theme(legend.position = "bottom")
```


Uniform mutation rate
SBS7b
Ratio of mutation type caused by signature vs background
1.5
1.0
0.5
0.0
Psoralen
SBS1
SBS5
SBS7a
SBS7b
Psoralen
SBS1
SBS5
SBS7a
SBS7b
Synonymous
Missense
Nonsense
Splice Site