Bispecific_Antibody_Target_Selection_ProstateCancer

2024-02-21

```
library(edgeR)
```

```
## Loading required package: limma
```

```
library(bsabsfinder)
```

```
## Warning: replacing previous import 'ensembldb::filter' by 'stats::filter' when
## loading 'bsabsfinder'
```

```
library(cluster)
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(ggpubr)
```

```
suppressWarnings({
    case = subset(phenoDF, cancer == "prostate adenocarcinoma" &
        sample.type == "primary")  #Select Prostate cancer as case
    case_id = case$sample.id  #getting case IDS
    control = subset(phenoDF, sample.type == "normal" & biopsy.site ==
        "PROSTATE")  #Select Normal Prostate samples
    control_id = control$sample.id  #getting control IDS

    case_expr = loadOctadCounts(case_id, type = "tpm", file = "~/Downloads/octad.counts.and.tpm.h5")
    case_expr = as.data.frame(case_expr)
    control_expr = loadOctadCounts(control_id, type = "tpm",
        file = "~/Downloads/octad.counts.and.tpm.h5")
    control_expr = as.data.frame(control_expr)

    # final data
    case_with_control_expr = cbind(case_expr, control_expr)

    # convert ensg to hgnc and select surface-expressed
    # genes according to compartments.jensenlab.org
    case_with_control_expr = ensg_to_hgnc(case_with_control_expr,
        select_surface = TRUE)

    phenotype_vector = as.factor(c(rep("case", ncol(case_expr)),
        rep("control", ncol(control_expr))))
})
```

```
## [1] "loading 60498 TPM expression values for 494 samples"
## [1] "loading 60498 TPM expression values for 100 samples"
```

#Perform Differential Gene Expression to filter out non-significant genes to speed up the computation

```
annotation = data.frame(sample = c(colnames(case_expr), colnames(control_expr)),
    phenotype = c(rep("cancer", length(colnames(case_expr))),
        rep("control", length(colnames(control_expr)))))
annotation$phenotype = as.factor(annotation$phenotype)
expression = DGEList(counts = as.matrix(case_with_control_expr),
    group = annotation$phenotype)
dim(expression)
```

```
## [1] 3736  594
```

```
keep <- rowSums(cpm(expression) > 100) >= 2
expression <- expression[keep, ]
dim(expression)
```

```
## [1] 3117  594
```

```
expression$samples$lib.size <- colSums(expression$counts)

suppressWarnings({
    expression <- calcNormFactors(expression)
})

expression_disp <- estimateCommonDisp(expression, verbose = T)
```

```
## Disp = 1e-04 , BCV = 0.01
```

```
expression_disp <- estimateTagwiseDisp(expression_disp)
DE <- exactTest(expression_disp, pair = c(1, 2))  # compare groups 1 and 2
DE = DE$table
DE$padj = p.adjust(DE$PValue, method = "BH")

DE1 = subset(DE, padj < 0.05 & abs(logFC) > 1)  # The cutoff criteria can be changed

# filter out only surface-expressed DE genes. Just to speed
```

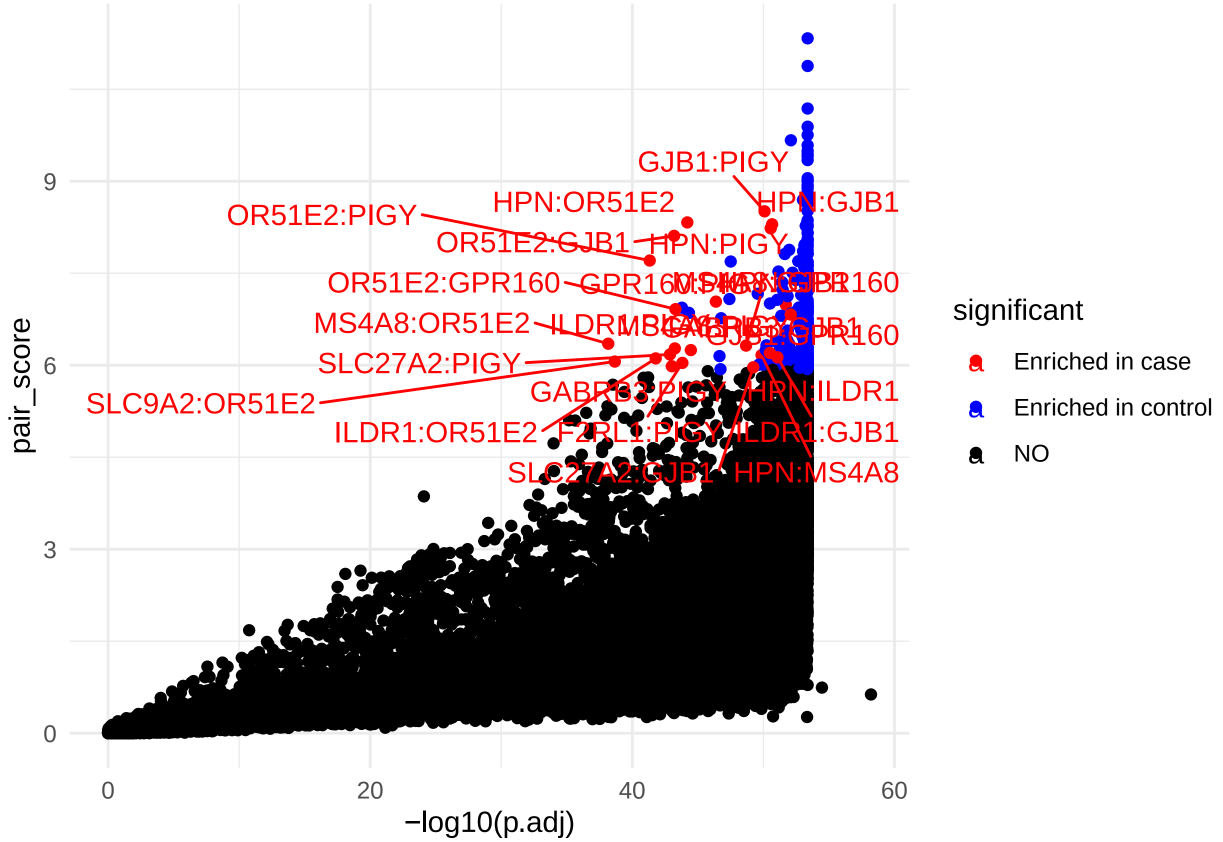
```r
# up.
case_with_control_expr = case_with_control_expr[row.names(case_with_control_expr) %in%
    row.names(DE1), ]
dataframe_for_computation = as.data.frame(t(case_with_control_expr))

# this step takes a while
small_res = compute_bsabs(antigene_1 = colnames(dataframe_for_computation),
    data_input = dataframe_for_computation, pheno_input = phenotype_vector)
```

```
##   |                                                                              |
```

```r
head(small_res)
```

```
##   antigen_1 antigen_2 distance   spread angle_cos pair_score      p.value
## 1    ITGA2B     SCN4A 1.266441 1.645612 0.7126953   1.485307 8.513137e-52
## 2    ITGA2B      GIPR 2.655597 1.627037 0.9112219   3.937166 8.630330e-50
## 3    ITGA2B    NPC1L1 1.327407 1.654277 0.7319016   1.607182 1.249741e-51
## 4    ITGA2B   SLC11A1 2.383090 1.665728 0.9500795   3.771416 9.820476e-55
## 5    ITGA2B   CYP3A43 1.436237 1.646023 0.8889049   2.101442 2.291172e-51
## 6    ITGA2B    SLC7A9 1.639132 1.656881 0.9921792   2.694607 1.994614e-52
##          p.adj case_greater
## 1 3.597593e-51  FALSE_FALSE
## 2 2.706100e-49  FALSE_FALSE
## 3 5.124317e-51  FALSE_FALSE
## 4 1.223267e-53  FALSE_FALSE
## 5 8.991584e-51  FALSE_FALSE
## 6 9.610274e-52  FALSE_FALSE
```

```r
## PLOT FIG.2D

suppressWarnings({
    plot_bsabs(small_res, label = "case", pval_cut_off = 0.01,
        pair_score_cut_off = quantile(small_res$pair_score, 0.99))
})
```



```
# subset result table to keep only those pairs where both
# markers have higher expression in case than control
small_res = small_res[small_res$case_greater == "TRUE_TRUE",
    ]

# ordering as per pair score , highest score should be at
# top
small_res = small_res[order(small_res$pair_score, decreasing = T),
    ]

# Subsetting top 20 pairs
small_res_selective = small_res[c(1:20), ]

# unique marker genes in top 20 pairs
marker_list = unique(c(small_res_selective$antigen_1, small_res_selective$antigen_2))
marker_list
```

```
##  [1] "GJB1"    "HPN"     "OR51E2"  "GPR160"  "MS4A8"   "ILDR1"   "GABRB3"
##  [8] "SLC27A2" "SLC9A2"  "PIGY"
```

```
# Checking the expression of marker genes in healthy tissue

healthy_tissues = subset(phenoDF, sample.type == "normal")
healthy_tissues = subset(healthy_tissues, grepl("BRAIN", biopsy.site) |
    biopsy.site == "LIVER" | biopsy.site == "LUNG" | grepl("HEART",
```

```r
    biopsy.site) | grepl("KIDNEY", biopsy.site))

healthy_tissues <- healthy_tissues %>%
    mutate(biopsy.site = ifelse(grepl("BRAIN", biopsy.site),
        "BRAIN", biopsy.site))
healthy_tissues <- healthy_tissues %>%
    mutate(biopsy.site = ifelse(grepl("HEART", biopsy.site),
        "HEART", biopsy.site))
healthy_tissues <- healthy_tissues %>%
    mutate(biopsy.site = ifelse(grepl("KIDNEY", biopsy.site),
        "KIDNEY", biopsy.site))

healthy_tissues_expr = loadOctadCounts(healthy_tissues$sample.id,
    type = "tpm", file = "~/Downloads/octad.counts.and.tpm.h5")
```

```
## [1] "loading 60498 TPM expression values for 1950 samples"
```

```r
healthy_tissues_expr = as.data.frame(healthy_tissues_expr)

healthy_tissues_expr = ensg_to_hgnc(healthy_tissues_expr, select_surface = FALSE)
healthy_tissues_expr = healthy_tissues_expr[row.names(healthy_tissues_expr) %in%
    marker_list, ]

healthy_tissues_expr = healthy_tissues_expr[order(rownames(healthy_tissues_expr)),
    ]

healthy_tissues_expr = as.data.frame(t(healthy_tissues_expr))

healthy_tissues_expr$Sample = healthy_tissues$biopsy.site[match(rownames(healthy_tissues_expr),
    healthy_tissues$sample.id)]

case_with_control_expr2 = case_with_control_expr[row.names(case_with_control_expr) %in%
    marker_list, ]

case_with_control_expr2 = case_with_control_expr2[order(rownames(case_with_control_expr2)),
    ]

case_with_control_expr2 = as.data.frame(t(case_with_control_expr2))

case_with_control_expr2$Sample = ifelse(rownames(case_with_control_expr2) %in%
    case_id, "PROSTATE_CANCER", "PROSTATE")

colnames(case_with_control_expr2) == colnames(healthy_tissues_expr)
```

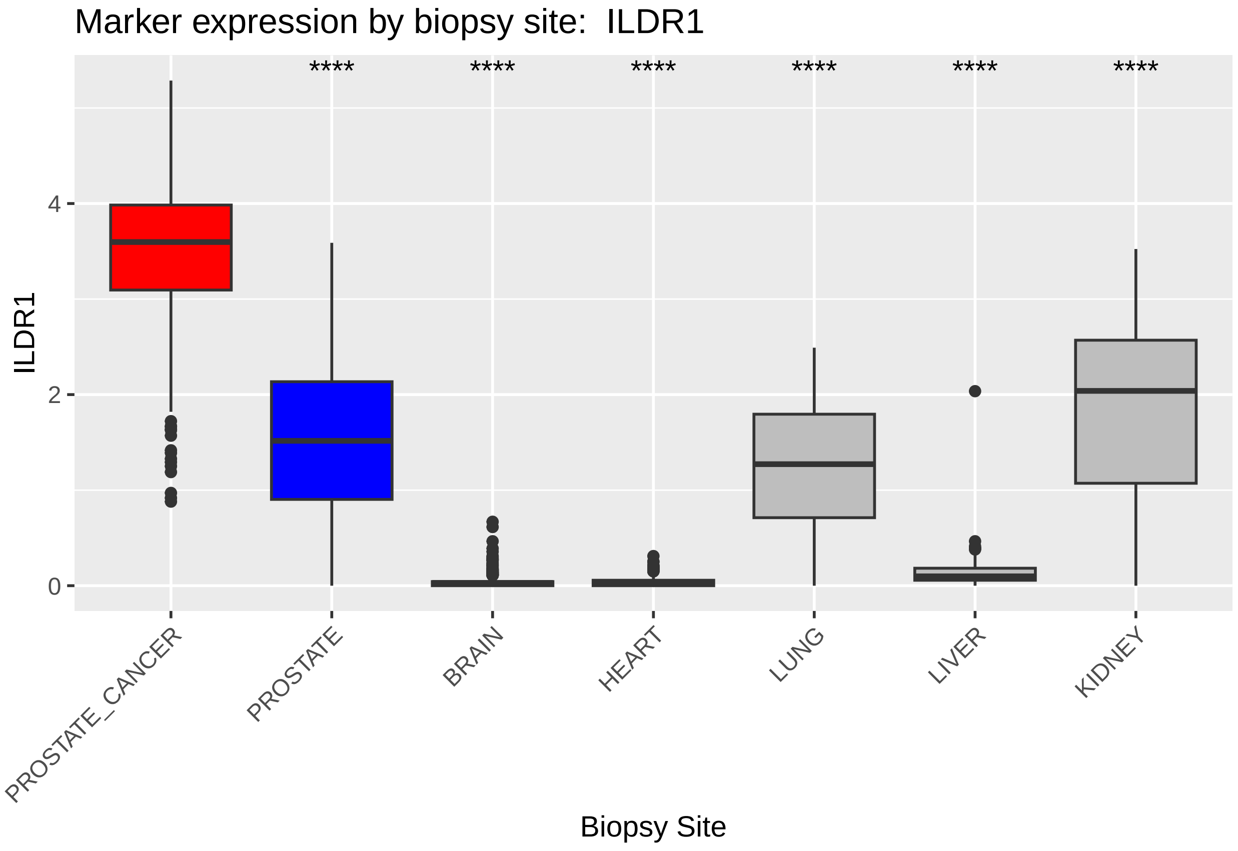
```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```r
marker_expr = rbind(case_with_control_expr2, healthy_tissues_expr)

table(marker_expr$Sample)
```

```
##
```

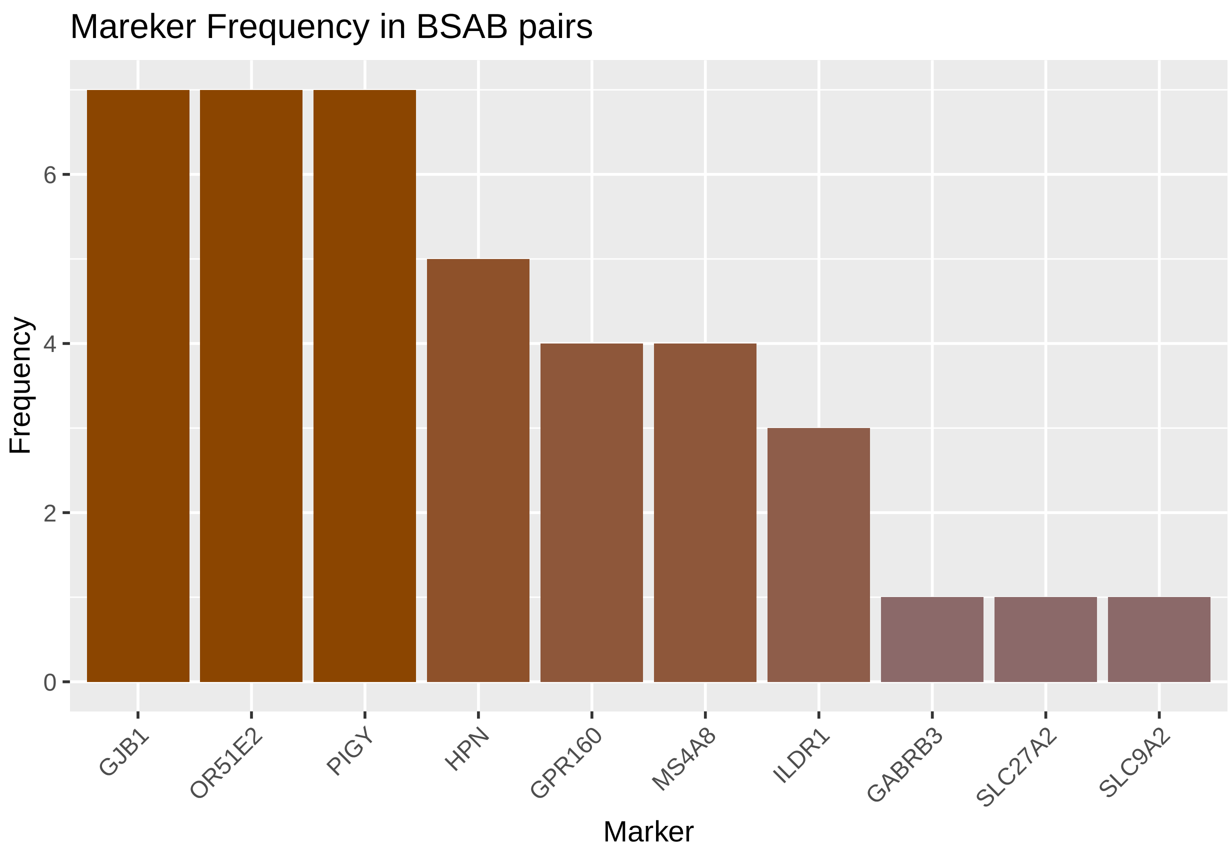
```
##           BRAIN           HEART          KIDNEY           LIVER            LUNG
##            1148             376              28             110             288
##        PROSTATE PROSTATE_CANCER
##             100             494
```

```
sample_order <- c("PROSTATE_CANCER", "PROSTATE", "BRAIN", "HEART",
    "LUNG", "LIVER", "KIDNEY")

marker_expr$Sample <- factor(marker_expr$Sample, levels = sample_order)

site_comparisons = list(c("PROSTATE_CANCER", "PROSTATE"), c("PROSTATE_CANCER",
    "BRAIN"), c("PROSTATE_CANCER", "HEART"), c("PROSTATE_CANCER",
    "LUNG"), c("PROSTATE_CANCER", "LIVER"), c("PROSTATE_CANCER",
    "KIDNEY"))


# PLOT FIG. 2E with STATS

ggplot(marker_expr, aes(x = Sample, y = marker_expr[[5]], fill = Sample)) +
    geom_boxplot() + scale_fill_manual(values = c("red", "blue",
    rep("gray", length(sample_order) - 2)), guide = "none") +
    labs(x = "Biopsy Site", y = names(marker_expr)[5], title = paste("Marker expression by biopsy site:
        names(marker_expr)[5])) + theme(axis.text.x = element_text(angle = 45,
    hjust = 1)) + guides(fill = "none") + stat_compare_means(method = "t.test",
    ref.group = "PROSTATE_CANCER", label = "p.signif")
```



```
# Similar to above plot , this code will generate plots for
# all markers.  You can specify the working directory where
# you would like to save the pdf.

pdf("~/Downloads/BSAB_PROSTATE_marker.pdf")
{
    for (i in 1:length(marker_list)) {
        gg <- ggplot(marker_expr, aes(x = Sample, y = marker_expr[[i]],
            fill = Sample)) + geom_boxplot() + scale_fill_manual(values = c("red",
            "blue", rep("gray", length(sample_order) - 2)), guide = "none") +
            labs(x = "Biopsy Site", y = names(marker_expr)[i],
                title = paste("Marker expression by biopsy site: ",
                  names(marker_expr)[i])) + theme(axis.text.x = element_text(angle = 45,
            hjust = 1)) + stat_compare_means(method = "t.test",
            ref.group = "PROSTATE_CANCER", label = "p.signif")

        print(gg)

    }
}
```

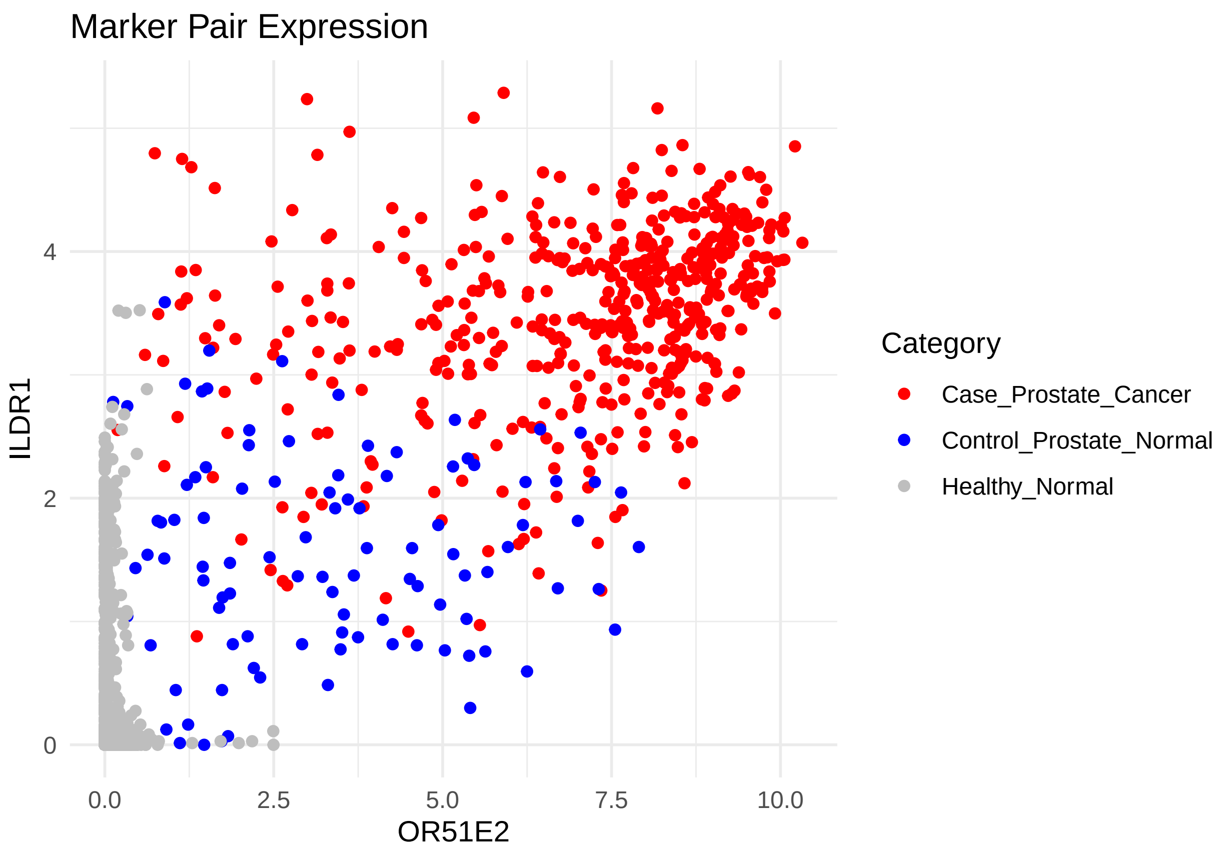
```
# PLOT FIG.2C Marker Frequency plot

markers <- c(small_res_selective$antigen_1, small_res_selective$antigen_2)

# Count the occurrences of each antigen in the combined
# vector
marker_counts <- table(markers)

# Convert the result to a data frame
marker_counts_df <- as.data.frame(marker_counts)
names(marker_counts_df) <- c("Marker", "Frequency")

marker_counts_df <- marker_counts_df[order(-marker_counts_df$Frequency),
    ]
marker_counts_df$Marker <- factor(marker_counts_df$Marker, levels = marker_counts_df$Marker)

proxy_var <- as.numeric(marker_counts_df$Frequency)


gg_freq <- ggplot(marker_counts_df, aes(x = Marker, y = Frequency,
    fill = proxy_var)) + geom_bar(stat = "identity") + scale_y_continuous(breaks = seq(0,
    max(marker_counts_df$Frequency) + 2, by = 2), limits = c(0,
    max(marker_counts_df$Frequency))) + theme(axis.text.x = element_text(angle = 45,
    hjust = 1)) + labs(x = "Marker", y = "Frequency", title = "Mareker Frequency in BSAB pairs") +
    scale_fill_gradient(low = "rosybrown4", high = "darkorange4",
        guide = "none")

print(gg_freq)
```



```
# PLOT FIG.2B Marker Frequency plot

marker_expr$Category = NA
unique(marker_expr$Sample)
```

```
## [1] PROSTATE_CANCER PROSTATE        HEART           BRAIN
## [5] LUNG            LIVER           KIDNEY
## Levels: PROSTATE_CANCER PROSTATE BRAIN HEART LUNG LIVER KIDNEY
```

```
marker_expr$Category[marker_expr$Sample == "PROSTATE_CANCER"] = "Case_Prostate_Cancer"
marker_expr$Category[marker_expr$Sample == "PROSTATE"] = "Control_Prostate_Normal"
marker_expr$Category[marker_expr$Sample %in% c("HEART", "BRAIN",
    "LUNG", "LIVER", "KIDNEY")] = "Healthy_Normal"

ggplot(marker_expr, aes(x = OR51E2, y = ILDR1, color = Category)) +
    geom_point() + labs(x = "OR51E2", y = "ILDR1", title = "Marker Pair Expression") +
    scale_color_manual(values = c(Case_Prostate_Cancer = "red",
        Control_Prostate_Normal = "blue", Healthy_Normal = "gray")) +
    theme_minimal()
```



```
# Save all important generated results

# save(list=c('case_expr','control_expr','case_with_control_expr','case_with_control_expr2','DE','DE1',
```