Stoat Report (08/2025)

Stoat in a nutshell

STOAT is a versatile GWAS (Genome-Wide Association Study) tool designed to work with pangenome graphs.

It supports binary phenotypes both with and without covariates: using Fisher's exact test or Chi-squared test when no covariates are present, and logistic regression when covariates are included.

For quantitative traits, STOAT performs linear regression, again with or without covariates. Additionally, it supports eQTL analysis, enabling association testing between genetic variants and gene expression levels.

It containt 2 modes : - VCF : that will take VCF file in input and test snarl variant in it - Graph : that will test snarl directly on the pangenome graph

Stoat version 0.2.0 (08/2025)

- stoat :
  - Cartoon update : added stoat graph
  - Unify format output (position, snarl_id, type)
- stoat vcf :
  - Linear regression : Correction for the near perfect collinearity case by merging same column toghever.
  - Logistic regression : Correction of the output format
  - Other correction : fix snarl paths parsing, fix eqtl parsing format file (case handlegraph too big)
- stoat graph :
  - ...

```
library(ggplot2)
library(readr)
library(dplyr)
library(tidyr)
library(CMplot)
library(stringr)

utils::globalVariables(c("Expected", "Observed"))
```

```
## [1] "Expected" "Observed"
```

```
# ---- File paths ----
file_list <- list(
  binary_snarl = "/home/mbagarre/Bureau/stoat/output_binary/snarl_analyse.tsv",
  binary_freq = "/home/mbagarre/Bureau/stoat/data/binary/pg.snarls.freq.tsv",
```

```
  binary_vcf = "/home/mbagarre/Bureau/stoat/output_binary/binary_table_vcf.tsv",
  binary_covar_vcf = "/home/mbagarre/Bureau/stoat/output_binary_covar/binary_table_vcf.tsv",

  quantitative_snarl = "/home/mbagarre/Bureau/stoat/output_quantitative/snarl_analyse.tsv",
  quantitative_freq = "/home/mbagarre/Bureau/stoat/data/quantitative/pg.snarls.freq.tsv",
  quantitative_vcf = "/home/mbagarre/Bureau/stoat/output_quantitative/quantitative_table_vcf.tsv",
  quantitative_covar_vcf = "/home/mbagarre/Bureau/stoat/output_quantitative_covar/quantitative_table_vc

  eqtl_snarl = "/home/mbagarre/Bureau/stoat/output_eqtl/snarl_analyse.tsv",
  eqtl_freq = "/home/mbagarre/Bureau/stoat/data/eqtl/pg.snarls.freq.tsv",
  eqtl_vcf = "/home/mbagarre/Bureau/stoat/output_eqtl/eqtl_table_vcf.tsv",
  eqtl_covar_vcf = "/home/mbagarre/Bureau/stoat/output_eqtl_covar/eqtl_table_vcf.tsv",

  binary_graph = "/home/mbagarre/Bureau/stoat/output_binary_graph/binary_table_graph.tsv"
)

# ---- Helper functions ----

# Read and clean table
read_stoat <- function(path) {
  df <- read_tsv(path, show_col_types = FALSE)
  if ("#CHR" %in% names(df)) names(df)[names(df) == "#CHR"] <- "CHR"
  if ("START_POS" %in% names(df) && !"POS" %in% names(df)) df$POS <- df$START_POS
  return(df)
}

# QQ plot with customizable column and subtitle
qq_plot <- function(df, col, subtitle = "") {
  if (!col %in% names(df)) return(NULL)

  pvec <- df[[col]]
  observed <- sort(pvec[!is.na(pvec)])
  expected <- -log10(ppoints(length(observed)))
  observed_log <- -log10(observed)
  data <- data.frame(Expected = expected, Observed = observed_log)

  ggplot(data, aes(x = Expected, y = Observed)) +
    geom_point(size = 1, alpha = 0.5) +
    geom_abline(intercept = 0, slope = 1, linetype = "dashed", color = "red") +
    labs(
      title = paste("QQ Plot of", col, subtitle),
      x = "Expected -log10(P)", y = "Observed -log10(P)"
    ) +
    theme_bw()
}

manhattan_plot <- function(df, subtitle = "", ...) {
  p_cols <- c(...)

  if (!all(c("CHR", "POS") %in% names(df))) {
    message("Skipping Manhattan plot: missing required columns 'CHR' or 'POS'.")
    return(NULL)
  }
```

```r
  # If no P columns specified, default to "P" if exists
  if (length(p_cols) == 0) {
    if ("P" %in% names(df)) {
      p_cols <- "P"
    } else {
      message("No P-value column specified or found. Skipping plot.")
      return(NULL)
    }
  }

  # Convert CHR to numeric for plotting
  df$CHR_numeric <- as.numeric(as.factor(df$CHR))

  # Keep only relevant columns
  df_plot <- df %>%
    select(CHR_numeric, POS, all_of(p_cols)) %>%
    pivot_longer(cols = all_of(p_cols), names_to = "P_column", values_to = "P") %>%
    filter(!is.na(P))

  # Plot
  ggplot(df_plot, aes(x = POS, y = -log10(P), color = P_column)) +
    geom_point(alpha = 0.6) +
    theme_bw() +
    labs(
      title = paste("Manhattan plot", subtitle),
      x = "Position (BP)",
      y = "-log10(P)",
      color = "P-value column"
    )
}

# Volcano plot
volcano_plot <- function(df, p_col = "P", beta_col = "BETA",
                         title = "Volcano Plot Beta vs P-value", subtitle = NULL,
                         log_y = TRUE, point_alpha = 0.6, point_color = "blue") {

  # Check required columns
  if (!all(c(beta_col, p_col) %in% names(df))) {
    message(sprintf("Skipping Volcano plot: '%s' or '%s' not found.", beta_col, p_col))
    return(NULL)
  }

  # Compute -log10(P)
  df$logP <- -log10(df[[p_col]])

  # Plot
  p <- ggplot(df, aes_string(x = beta_col, y = "logP")) +
    geom_point(alpha = point_alpha, color = point_color) +
    theme_bw() +
    labs(
      title = title,
      subtitle = subtitle,
      x = paste("Effect Size (", beta_col, ")", sep = ""),
```

```
      y = paste("-log10(", p_col, ")", sep = "")
    )

  if (log_y) {
    p <- p + scale_y_continuous(trans = "log10")
  }

  return(p)
}

# RSQUARE vs BETA plot
rsq_plot <- function(df, rsq_col = "RSQUARE", beta_col = "BETA",
                     title = "Effect Size vs RSQUARE", subtitle = NULL,
                     log_x = FALSE, log_y = FALSE, point_alpha = 0.6, point_color = "blue") {

  # Check required columns
  if (!all(c(rsq_col, beta_col) %in% names(df))) {
    message(sprintf("Skipping RSQ plot: '%s' or '%s' not found.", rsq_col, beta_col))
    return(NULL)
  }

  # Plot
  p <- ggplot(df, aes_string(x = rsq_col, y = beta_col)) +
    geom_point(alpha = point_alpha, color = point_color) +
    theme_bw() +
    labs(
      title = title,
      subtitle = subtitle,
      x = rsq_col,
      y = beta_col
    )

  if (log_x) p <- p + scale_x_continuous(trans = "log10")
  if (log_y) p <- p + scale_y_continuous(trans = "log10")

  return(p)
}

# Histogram of P-values with custom columns and subtitle
histogram_plot <- function(df, subtitle = "", ...) {
  cols <- c(...)
  if (!all(cols %in% names(df))) return(NULL)

  df_long <- df %>%
    select(all_of(cols)) %>%
    pivot_longer(cols = everything(), names_to = "Type", values_to = "PValue")

  ggplot(df_long, aes(x = PValue, fill = Type)) +
    geom_histogram(alpha = 0.6, position = "dodge", bins = 50) +
    theme_bw() +
    labs(
      title = paste("Distribution of P-values", subtitle),
      x = "P-value", y = "Count"
```

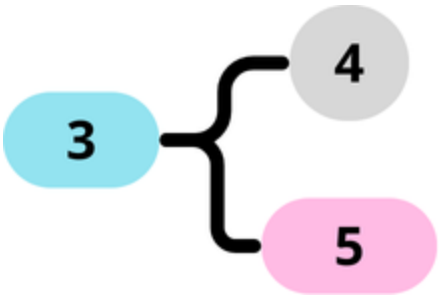
```
    ) +
    scale_fill_brewer(palette = "Set1")
}

# ---- Run Python verification truth VCF ----
run_verify_truth <- function(freq_file, pvalue_file, paths_file, flag, output_dir) {
  cmd <- paste(
    "python3 /home/mbagarre/Bureau/stoat/tests/scripts/verify_truth.py",
    "--freq", freq_file,
    "--p_value", pvalue_file,
    "--paths", paths_file,
    flag,
    "--output", output_dir
  )
  output <- system(cmd, intern = TRUE)
  return(output)
}

run_scatter_plot_graph <- function(input, output) {
  cmd <- paste(
    "python3 /home/mbagarre/Bureau/stoat/tests/scripts/plot_scatter.py",
    input,
    output
  )
  output <- system(cmd, intern = TRUE)
  return(output)
}

run_histo_plot_graph <- function(input, output) {
  cmd <- paste(
    "python3 /home/mbagarre/Bureau/stoat/tests/scripts/plot_histogram.py",
    input,
    output
  )
  output <- system(cmd, intern = TRUE)
  return(output)
}
```

Simulation description

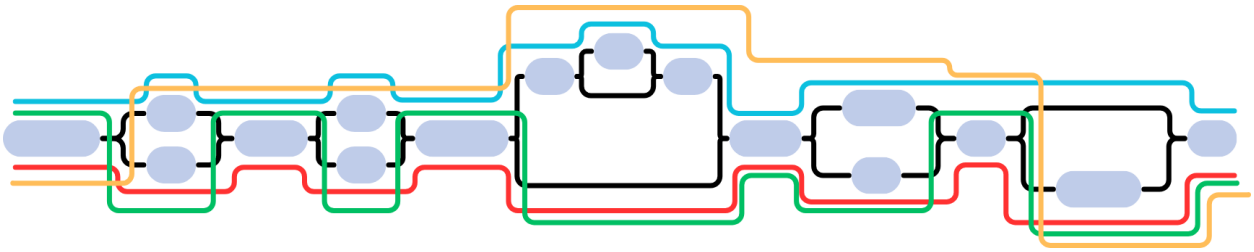
We constructed three simulated datasets: binary, quantitative, and eQTL.

The binary and quantitative simulations each have their own pangenome graph containing a single chromosome that incorporates various types of variation, such as SNPs, INDELs, and complex variants. However, the graph structure, based on a fork pattern, remains consistent across all simulations.

A fork structure is defined as a boundary snarl with exactly two paths, which can themselves contain nested forks (see fork representation below).



Once the pangenome graph was constructed, we simulated individual haplotypes by generating sample genomes that traverse different paths in the graph. To introduce phenotype-genotype relationships, we assigned each haplotype to a binary or quantitative phenotype according to specific simulation rules.



The eQTL simulation differs in that it includes 10 chromosomes, but no pangenome graph is constructed. Instead, we directly simulate the path-snarl file required by Stoat, and the variations consist only of simple SNPs.

```r
library(knitr)

# Create the data frame
df <- data.frame(
  Type = c("Binary", "Quantitative", "eqtl"),
  `Number of samples` = c(200, 200, 200),
  `Number of variant type (SNP/INDEL/COMPLEX)` = c("2444/446/158", "2478/382/138", "200000/0/0"),
  `Number of snarl/paths` = c("1524/3048", "1499/2998", "100000/200000"),
  check.names = FALSE
)

# Print as R Markdown table
kable(df, format = "markdown", align = "c")
```

Type	Number of samples	Number of variant type (SNP/INDEL/COMPLEX)	Number of snarl/paths
Binary	200	2444/446/158	1524/3048
Quantitative	200	2478/382/138	1499/2998
eqtl	200	200000/0/0	100000/200000

Binary simulation

200 Samples were divided into two cohorts (100 case and 100 control), each corresponding to a different phenotypic state containing 1000 variations. Each group had a correlated probability of traversing a specific path within snarls. This probability (e.g., 50/50) could be equal between the two cohorts or skewed in favor of one group (e.g., 20/80), simulating an association between variation and phenotype.

Quantitative simulation

Similar to the binary simulation, 200 samples were divided into two cohorts, with each group having a correlated probability of traversing specific paths within snarls. However, in this case, an additional probability factor was introduced, where the likelihood of passing through a given path was influenced by the individual's phenotype value.

Eqtl simulation

Here, we simulated 200 samples with 200 000 simple SNP variations and 100 genes. In this simulation, there are no significant variants, everything was generated randomly. The goal was to create a very simple simulation to test STOAT's VCF eQTL implementation.

Covariate Simulation

Covariates are simulated with no effect. They include **SEX**, **PC1**, **PC2**, and **PC3**. Each sample has the same values across these covariates: `0`, `25.215`, `75.84`, `45.0`.

Verifying the Truth Simulation Description

Frequency File Description The **binary** and **quantitative** simulations produce a file containing the following elements:

- The **frequency probability (freq)** of a haplotype, depending on its group, to follow a specific path/edge in a fork.
- The **group** associated with each frequency.
- The **start** and **end** node of the fork.

An identical probability between two groups on the same edge suggests that the edge may appear significant due to randomness, but it will be considered **non-significant** in downstream analysis. On the other hand, a difference in frequency between the two groups even if small is treated as **significant**.

Example `freq` file:

start_node	next_node	group	freq
2	3	0	0.53
2	3	1	0.53
2	4	0	0.47
2	4	1	0.47

Computing Stoat Output with the Frequency File To assess whether a snarl is correctly identified as significant, we merge the snarl's path information with the frequency probabilities from the `freq` file.

A **correct match** is a snarl that contains **both** start/end node pairs in two **separate paths** (one for each group).

Graph Information

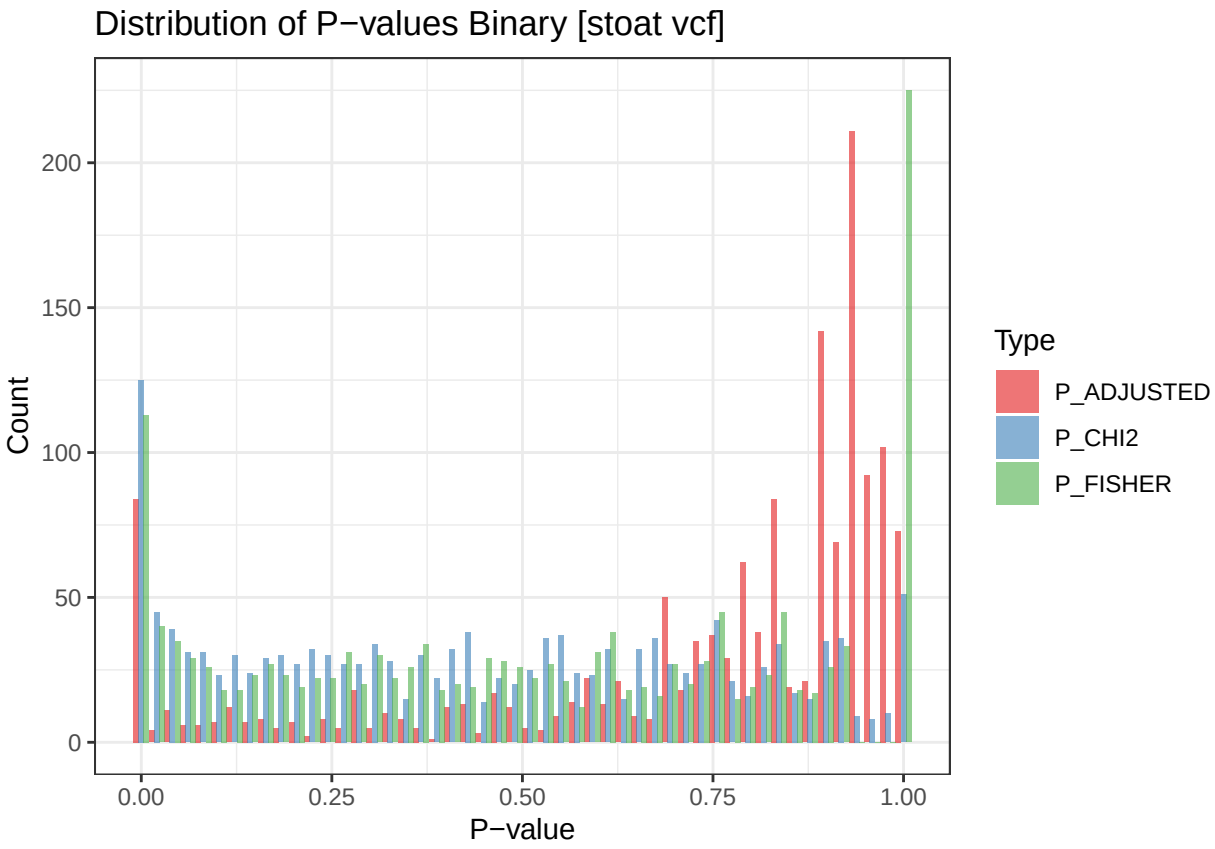
- A snarl is considered **true** if it contains a fork where **at least one edge** shows a frequency difference between the two groups.
- A snarl is considered **positive** if its **P-value is below 0.01**.

Binary stoat VCF

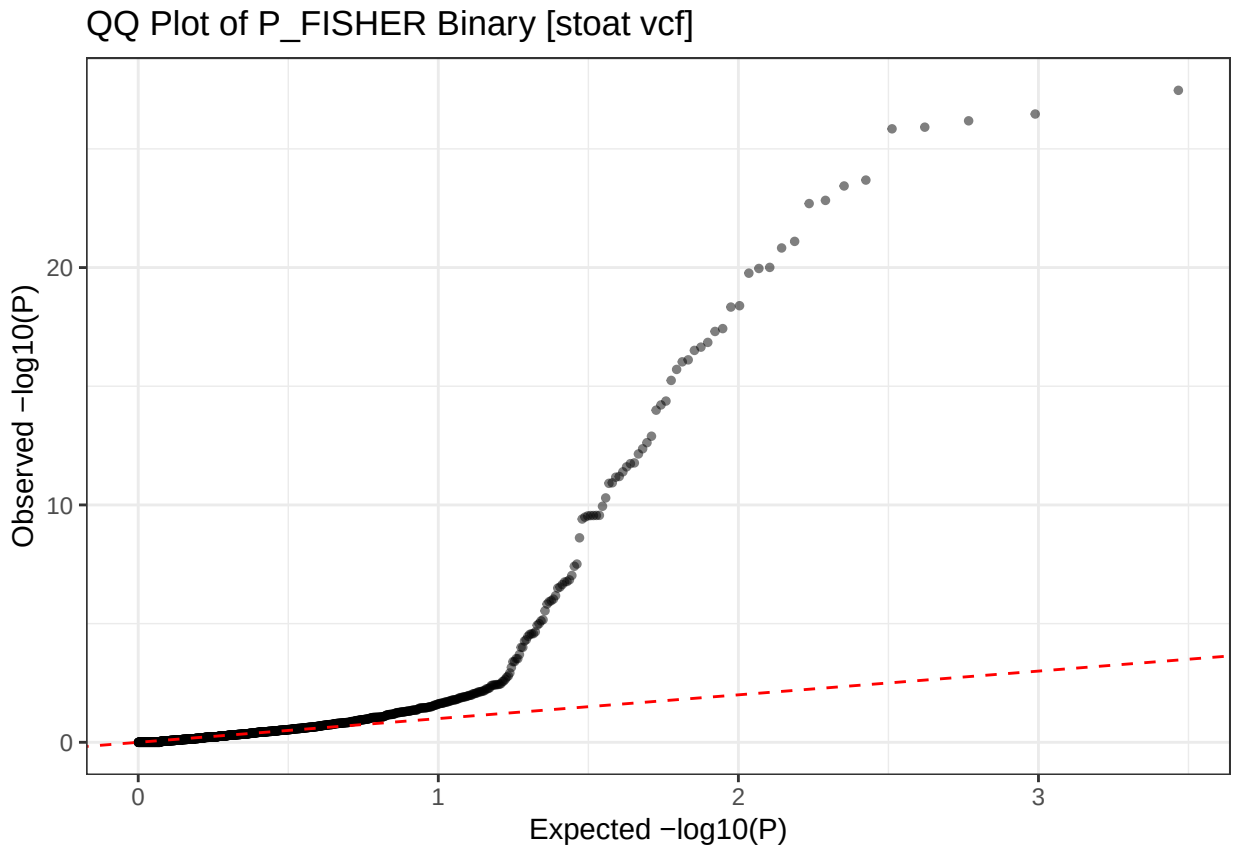
```
# Execute verification for binary VCF
binary_output <- run_verify_truth(
  freq_file = file_list$binary_freq,
  pvalue_file = file_list$binary_vcf,
  paths_file = file_list$binary_snarl,
  flag = " -b ",
  output_dir = "../binary_output"
)

cat(binary_output, sep = "\n")
```

```
## Output directory already exists: ../binary_output
##
## Metrics for p-value < 0.01:
## Confusion Matrix for p-value [P] < 0.01:
## [[ 101   22]
##  [  30 1310]]
## Precision: 0.983
## Recall: 0.978
## F1 Score: 0.981
##
## Metrics for p-value < 1e-05:
## Confusion Matrix for p-value [P] < 1e-05:
## [[  68    1]
##  [  63 1331]]
## Precision: 0.999
## Recall: 0.955
## F1 Score: 0.977
##
## Metrics for p-value < 1e-08:
## Confusion Matrix for p-value [P] < 1e-08:
## [[  50    0]
##  [  81 1332]]
## Precision: 1.000
## Recall: 0.943
## F1 Score: 0.970
##
## Pourcentage of paths (present in freq file) tested :  95.99737532808399
```

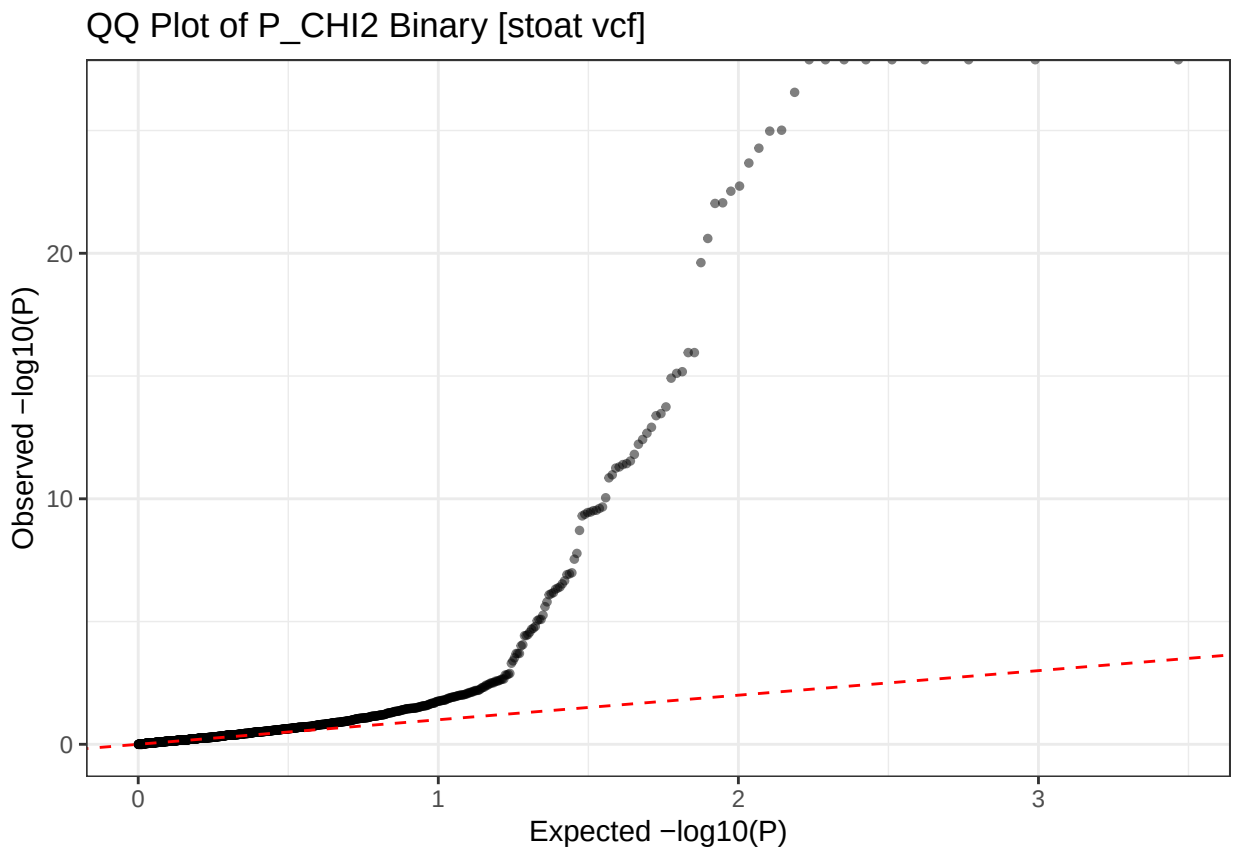
```
df_binary_vcf <- read_stoat(file_list$binary_vcf)

# Plots
histogram_plot(df=df_binary_vcf, subtitle="Binary [stoat vcf]", "P_FISHER", "P_CHI2", "P_ADJUSTED")
```



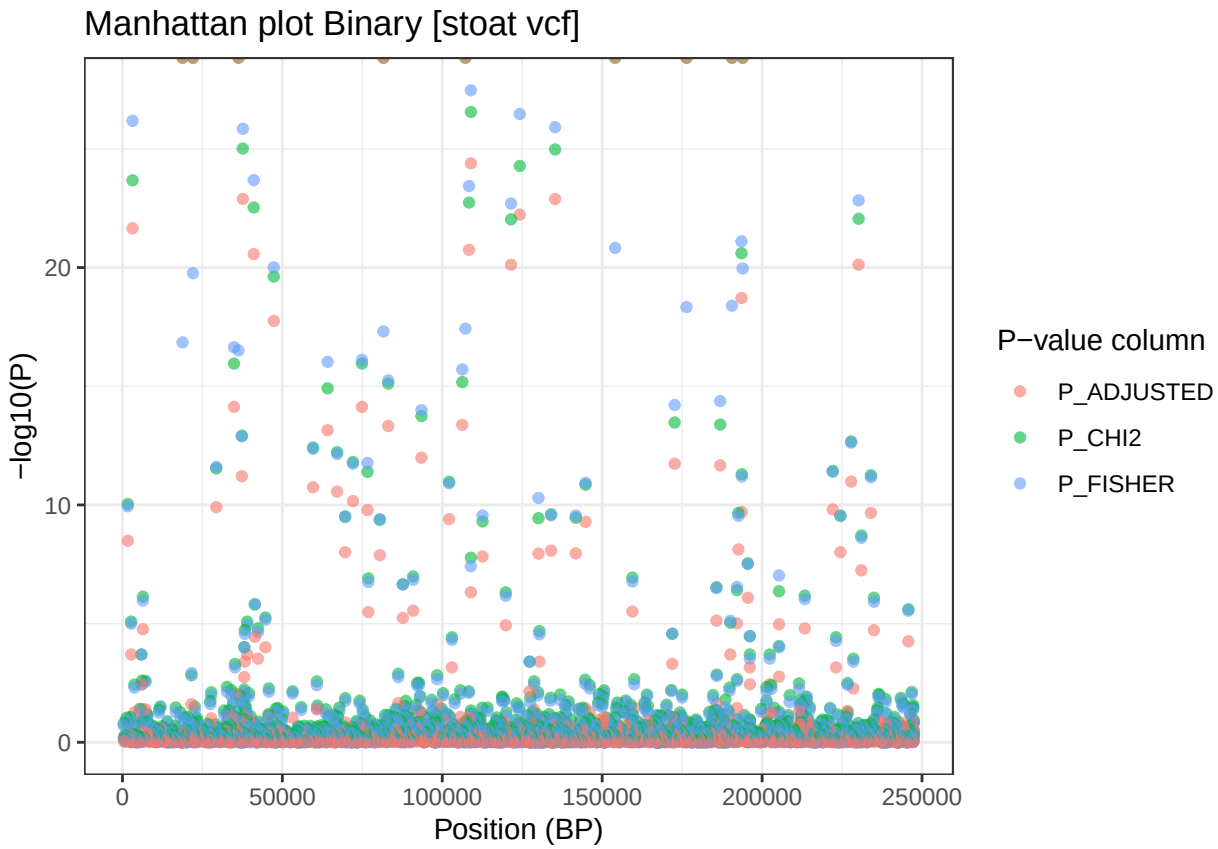

```
qq_plot(df_binary_vcf, "P_FISHER", subtitle="Binary [stoat vcf]")
```

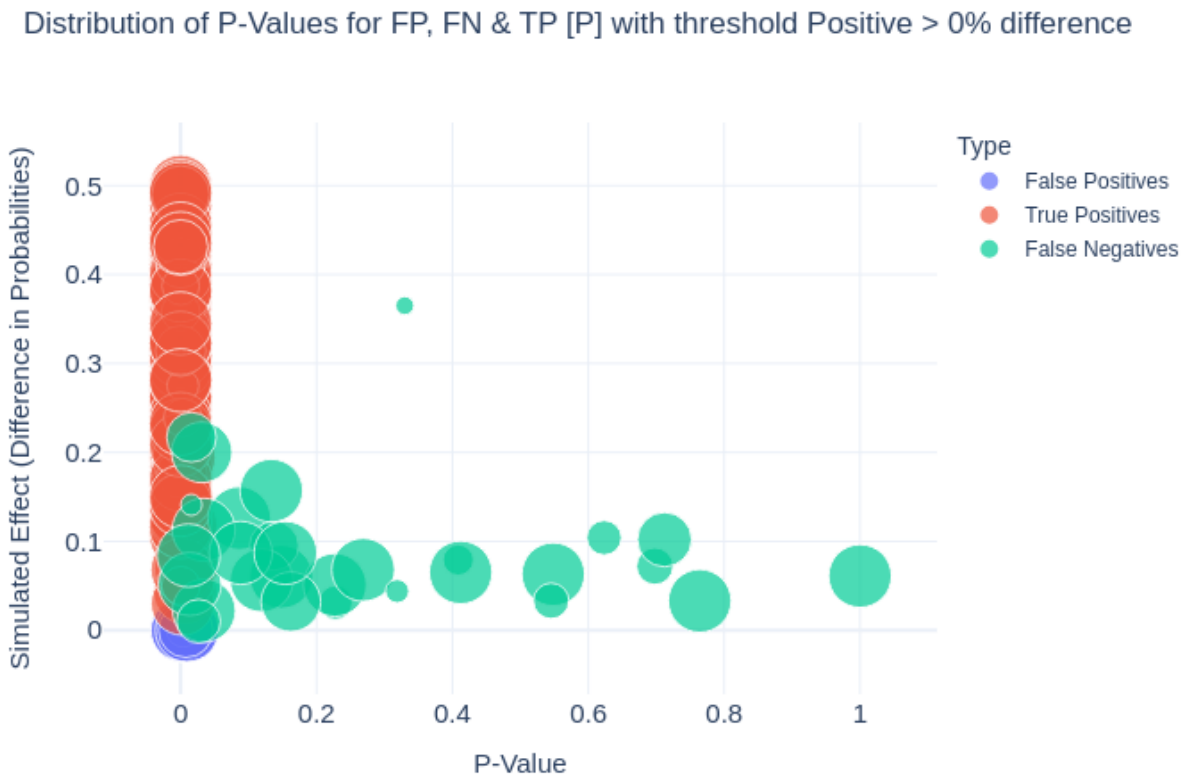


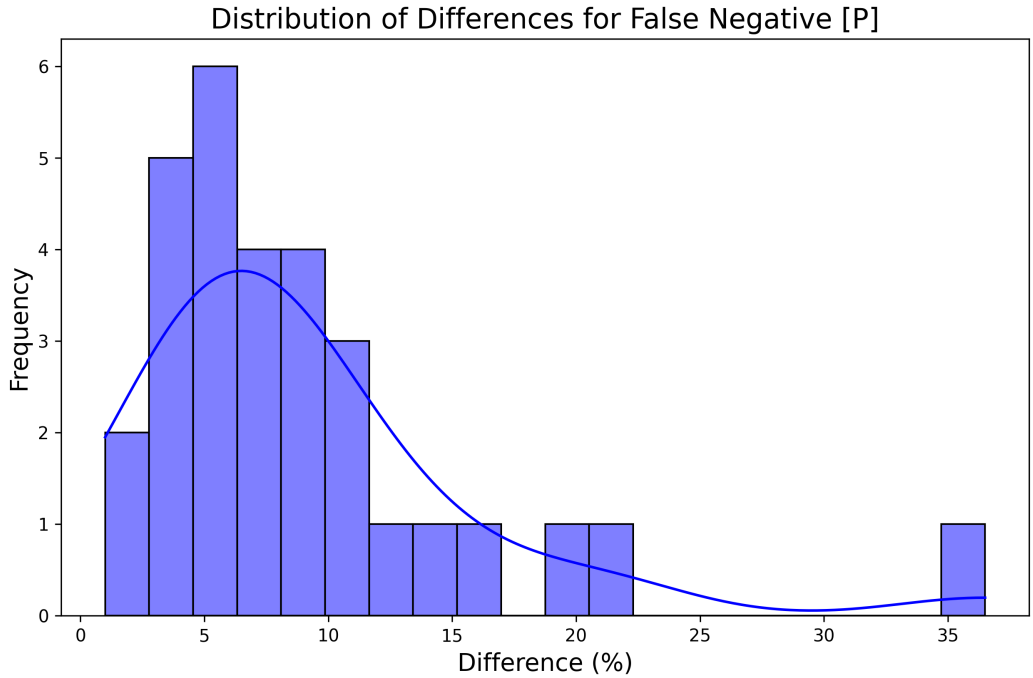

```
qq_plot(df_binary_vcf, "P_CHI2", subtitle="Binary [stoat vcf]")
```

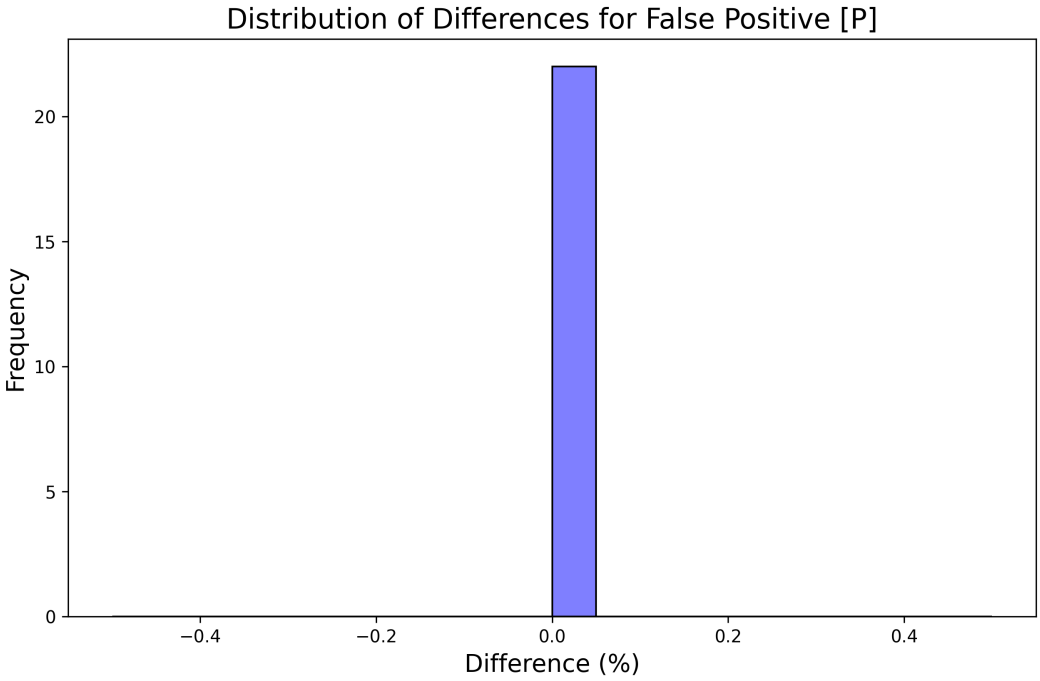


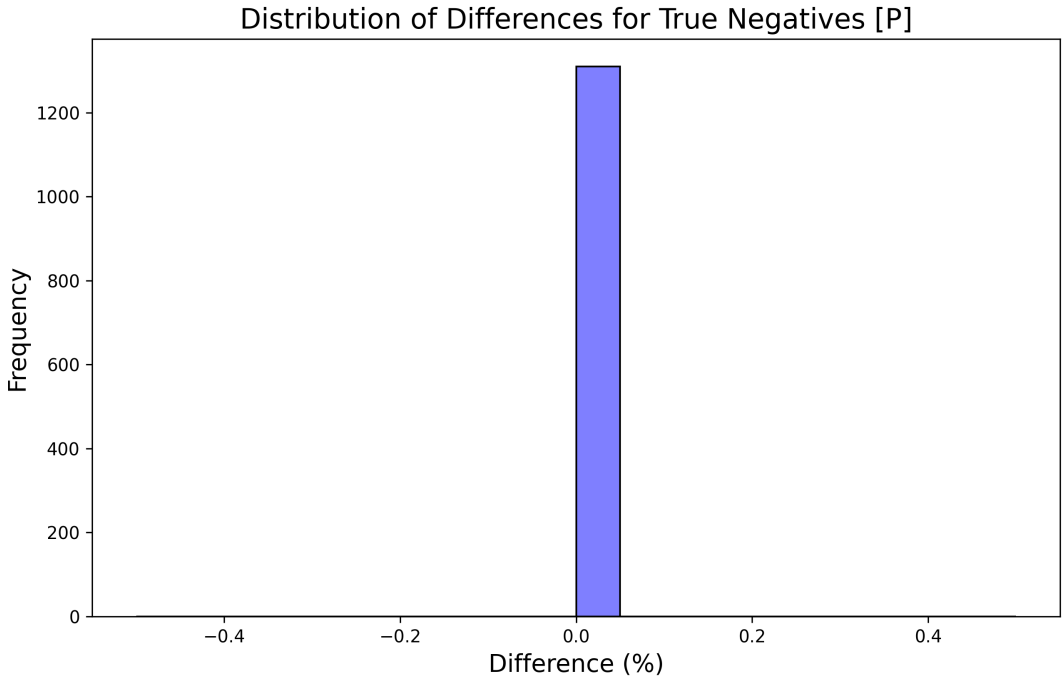

```
manhattan_plot(df_binary_vcf, subtitle="Binary [stoat vcf]", "P_CHI2", "P_FISHER", "P_ADJUSTED")
```

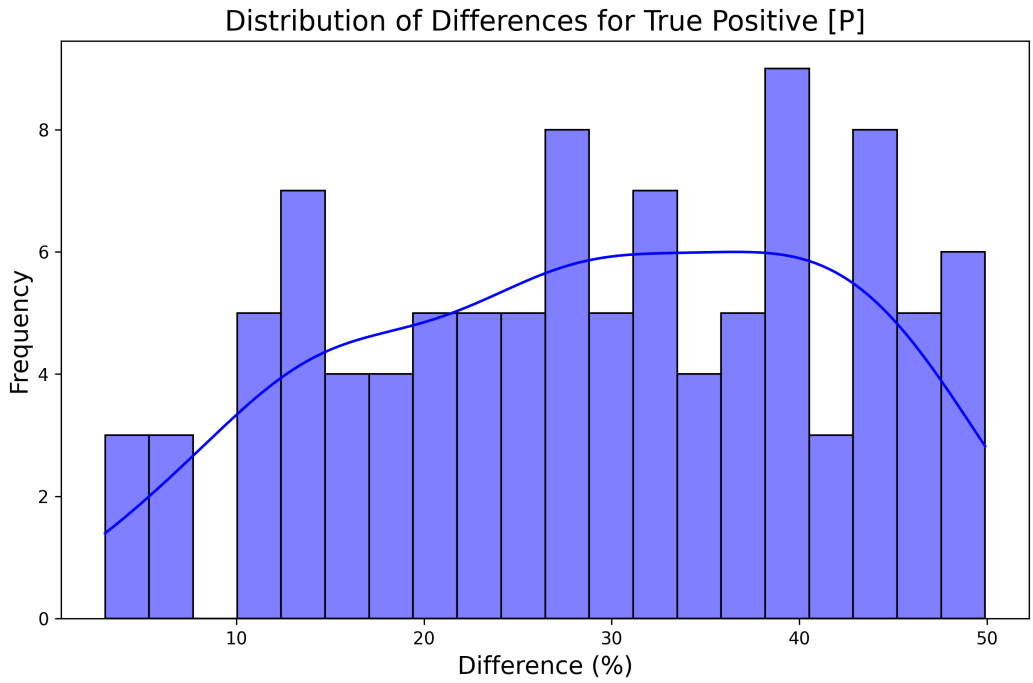

Manhattan plot Binary [stoat vcf]
−log10(P)
20
10
0
0
50000
100000
150000
200000
250000
Position (BP)
P−value column
P_ADJUSTED
P_CHI2
P_FISHER

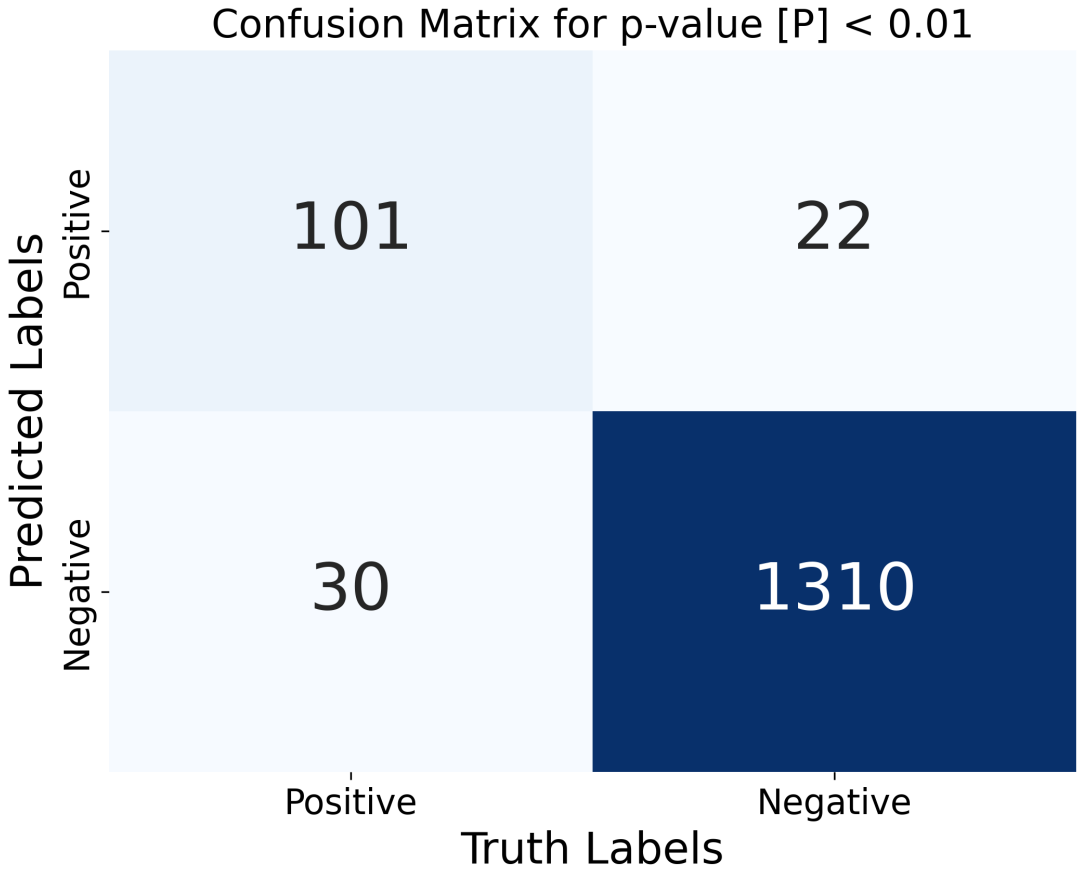

Distribution of P-Values for FP, FN & TP [P] with threshold Positive > 0% difference
Simulated Effect (Difference in Probabilities)
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1
P-Value
Type
False Positives
True Positives
False Negatives

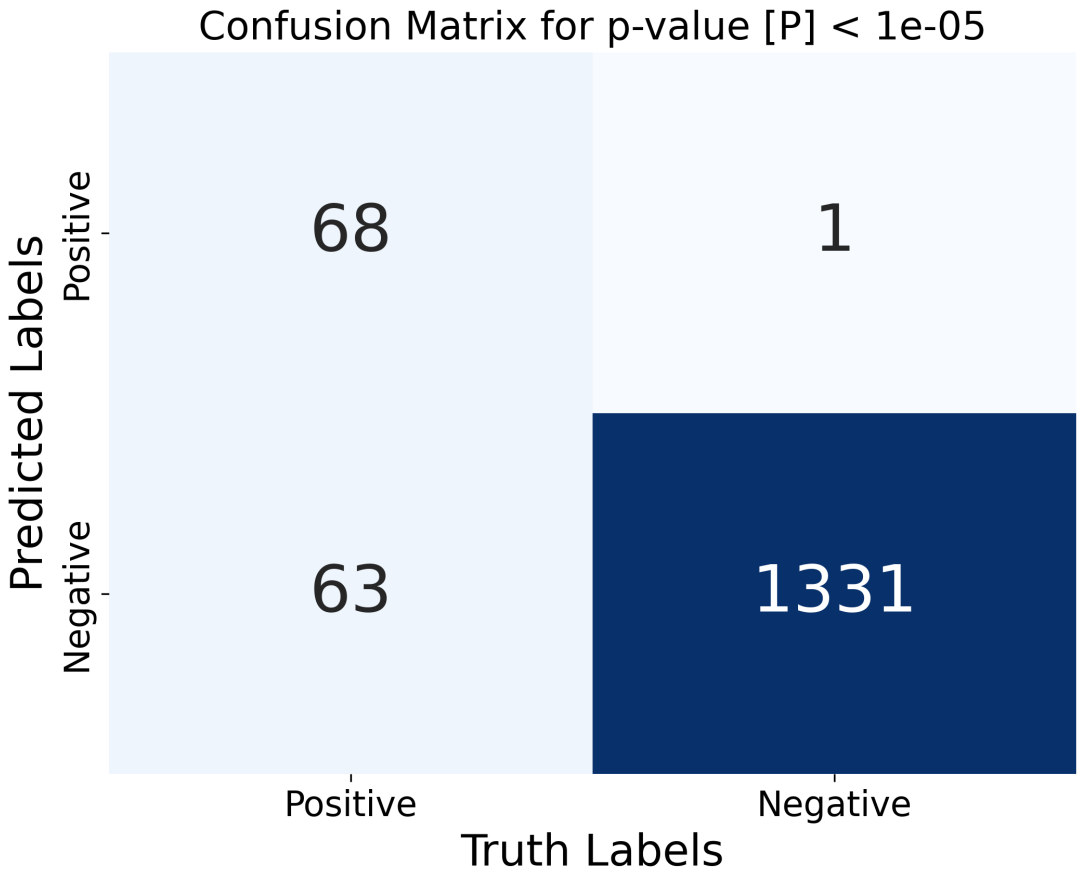

Distribution of Differences for False Negative [P]
Frequency
6
5
4
3
2
1
0
0
5
10
15
20
25
30
35
Difference (%)

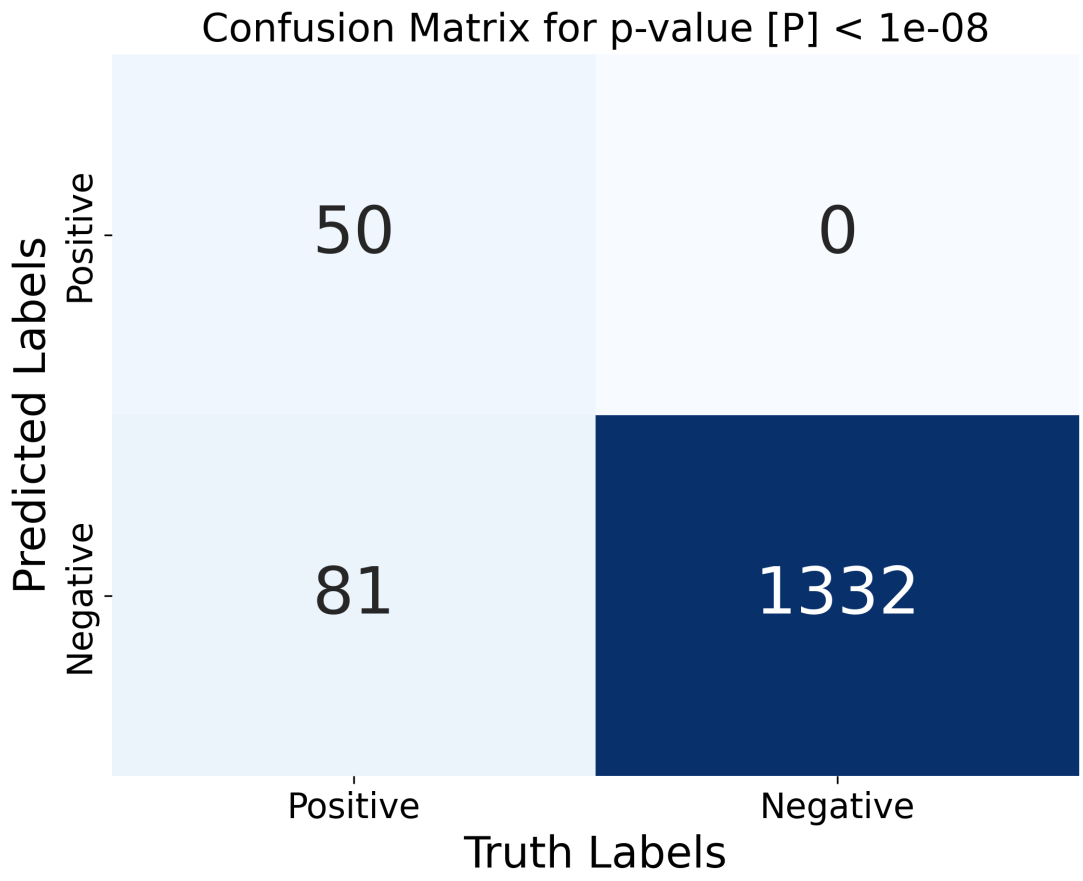

Distribution of Differences for False Positive [P]
Frequency
Difference (%)
0
5
10
15
20
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Negatives [P]
Frequency
Difference (%)
0
200
400
600
800
1000
1200
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Positive [P]
Frequency
Difference (%)
0
2
4
6
8
10
20
30
40
50


Confusion Matrix for p-value [P] < 0.01
Predicted Labels
Positive
Negative
101
22
30
1310
Positive
Negative
Truth Labels


Confusion Matrix for p-value [P] < 1e-05
Predicted Labels
Positive
Negative
68
1
63
1331
Positive
Negative
Truth Labels



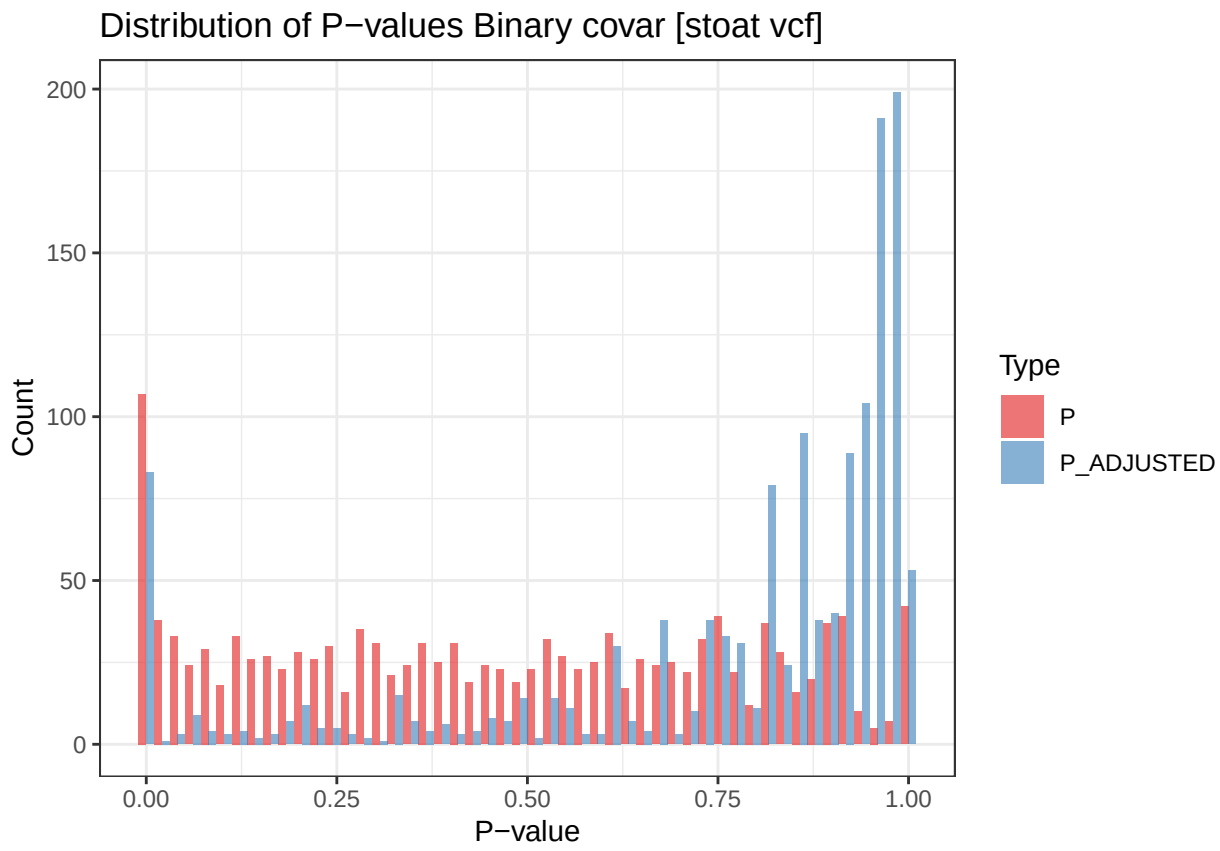
Binary stoat covar VCF

```
# Execute verification for binary VCF
binary_output <- run_verify_truth(
  freq_file = file_list$binary_freq,
  pvalue_file = file_list$binary_covar_vcf,
  paths_file = file_list$binary_snarl,
  flag = " -c ",
  output_dir = "../binary_covar_output"
)

cat(binary_output, sep = "\n")
```

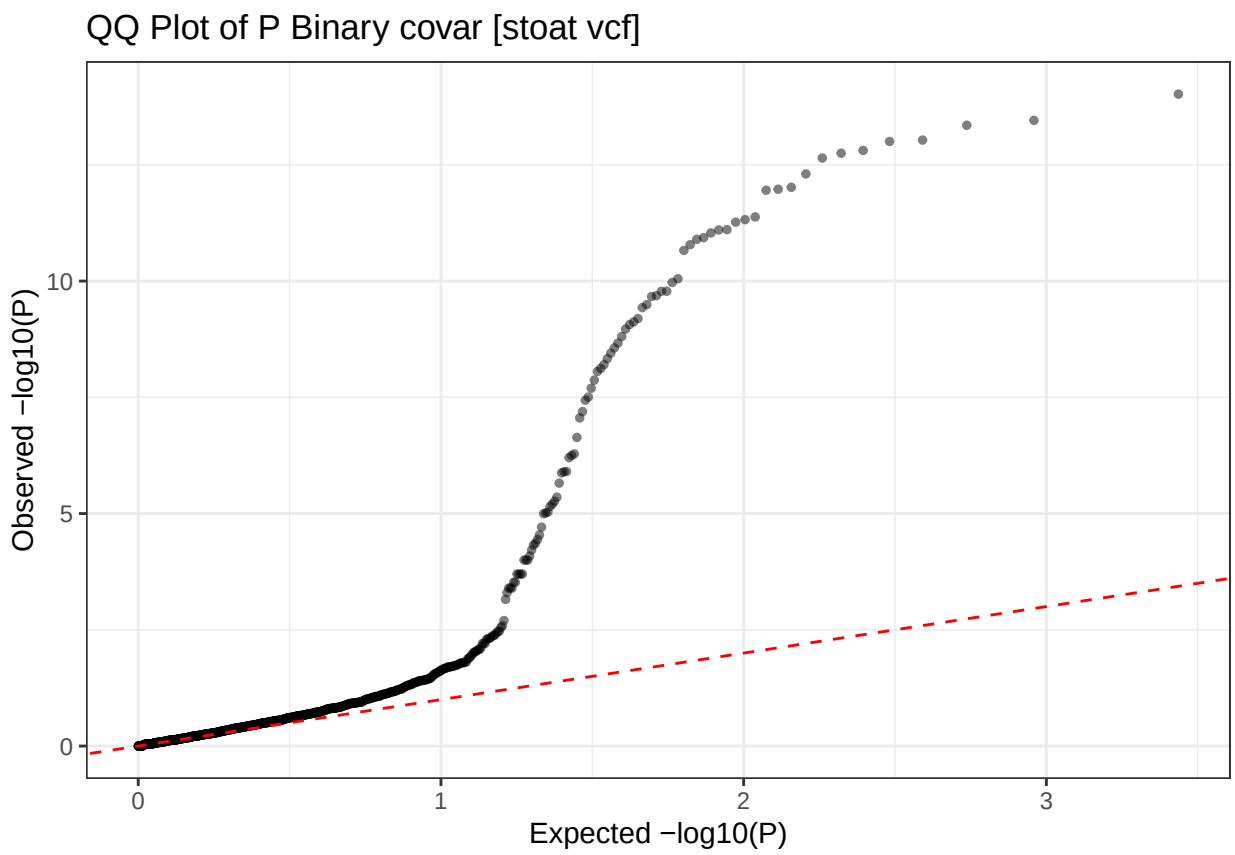
```
## Output directory already exists: ../binary_covar_output
##
## Metrics for p-value < 0.01:
## Confusion Matrix for p-value [P] < 0.01:
## [[  97   10]
##  [  41 1217]]
```

```
## Precision: 0.992
## Recall: 0.967
## F1 Score: 0.979
##
## Metrics for p-value < 1e-05:
## Confusion Matrix for p-value [P] < 1e-05:
## [[  62    0]
##  [  76 1227]]
## Precision: 1.000
## Recall: 0.942
## F1 Score: 0.970
##
## Metrics for p-value < 1e-08:
## Confusion Matrix for p-value [P] < 1e-08:
## [[  42    0]
##  [  96 1227]]
## Precision: 1.000
## Recall: 0.927
## F1 Score: 0.962
##
## Pourcentage of paths (present in freq file) tested :  89.56692913385827
```

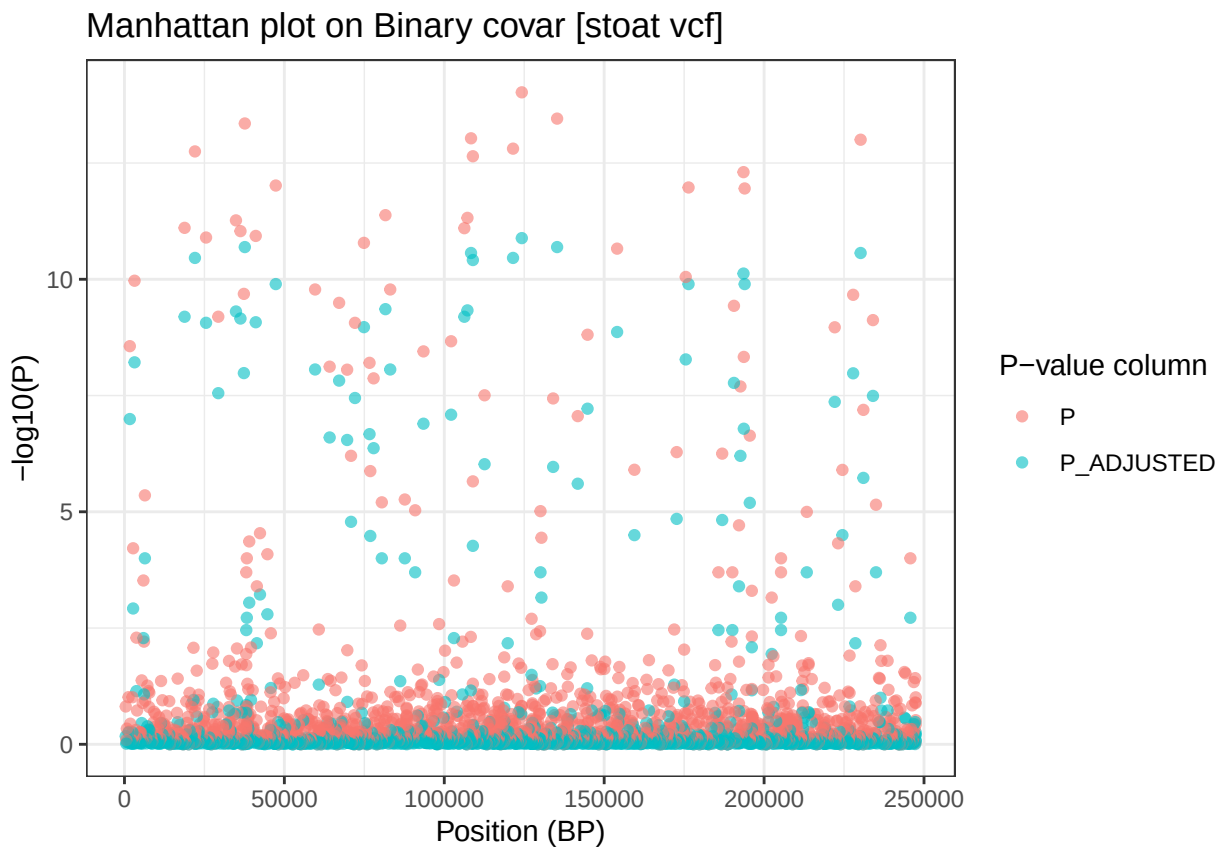
```r
df_binary_covar_vcf <- read_stoat(file_list$binary_covar_vcf)

# Plots
histogram_plot(df=df_binary_covar_vcf, subtitle="Binary covar [stoat vcf]", "P", "P_ADJUSTED")
```



```
qq_plot(df_binary_covar_vcf, "P", subtitle="Binary covar [stoat vcf]")
```

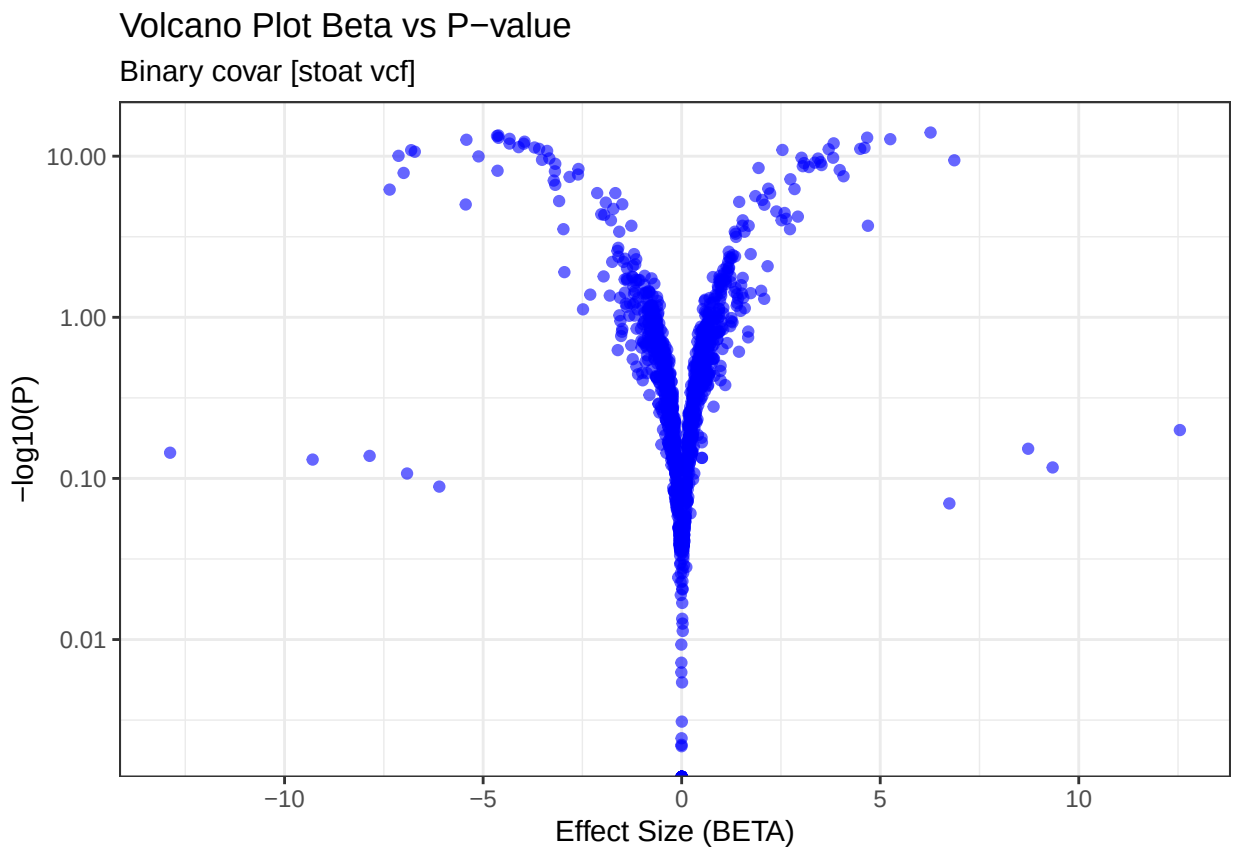


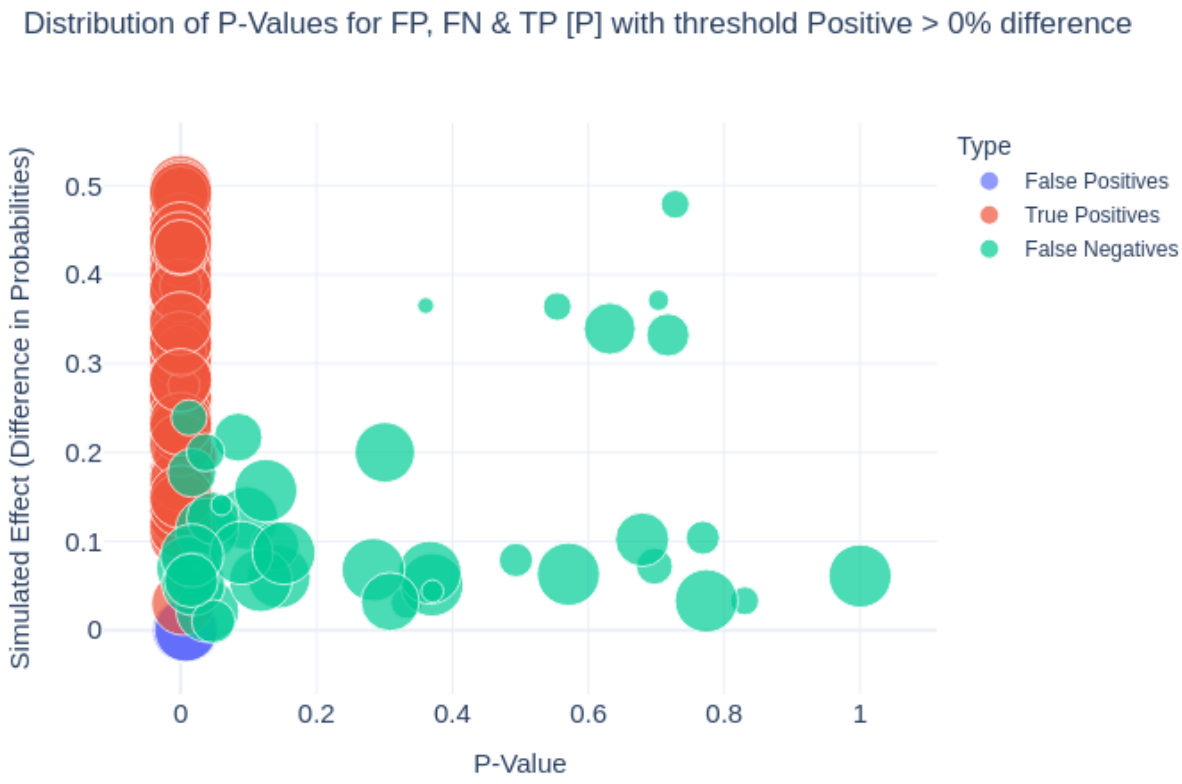

```
manhattan_plot(df_binary_covar_vcf, subtitle="on Binary covar [stoat vcf]", "P", "P_ADJUSTED")
```

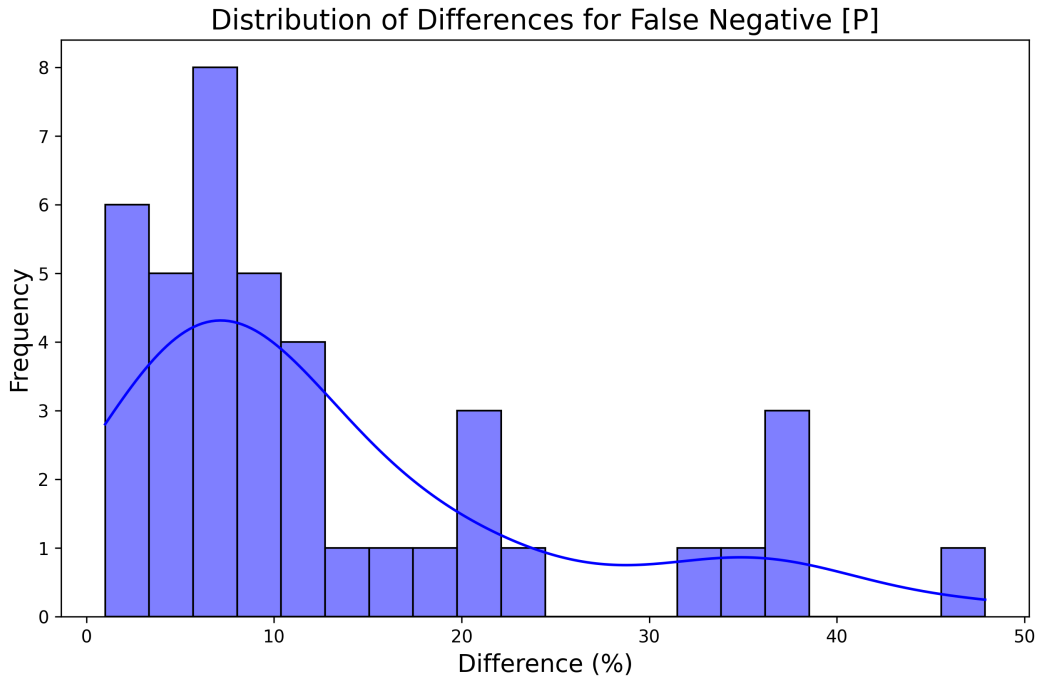


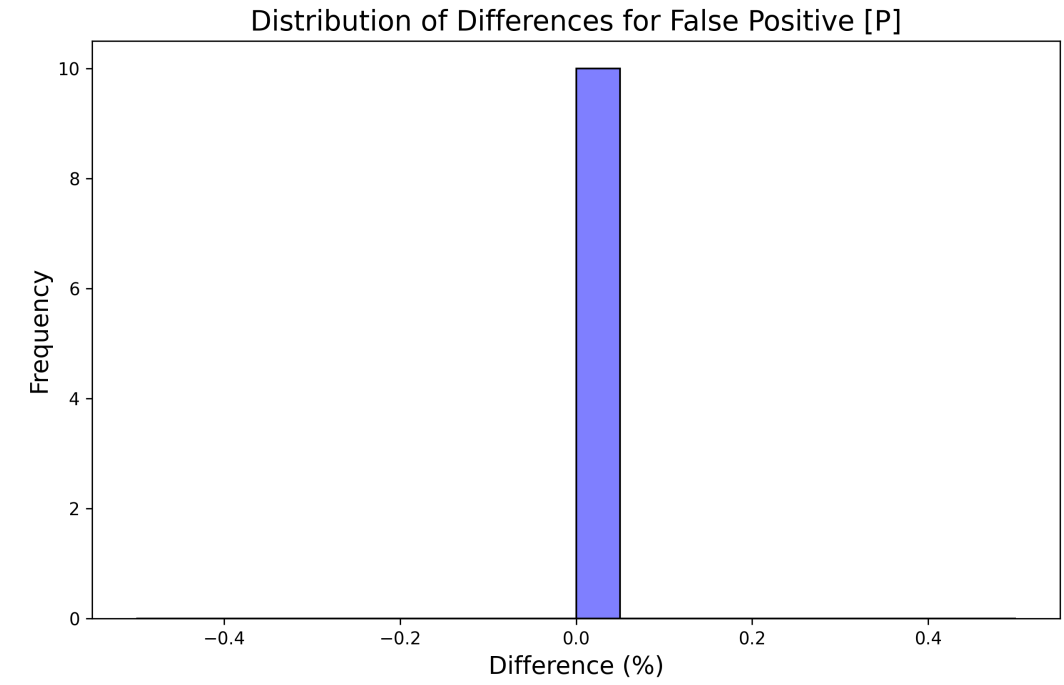

```
volcano_plot(df_binary_covar_vcf, subtitle="Binary covar [stoat vcf]")
```

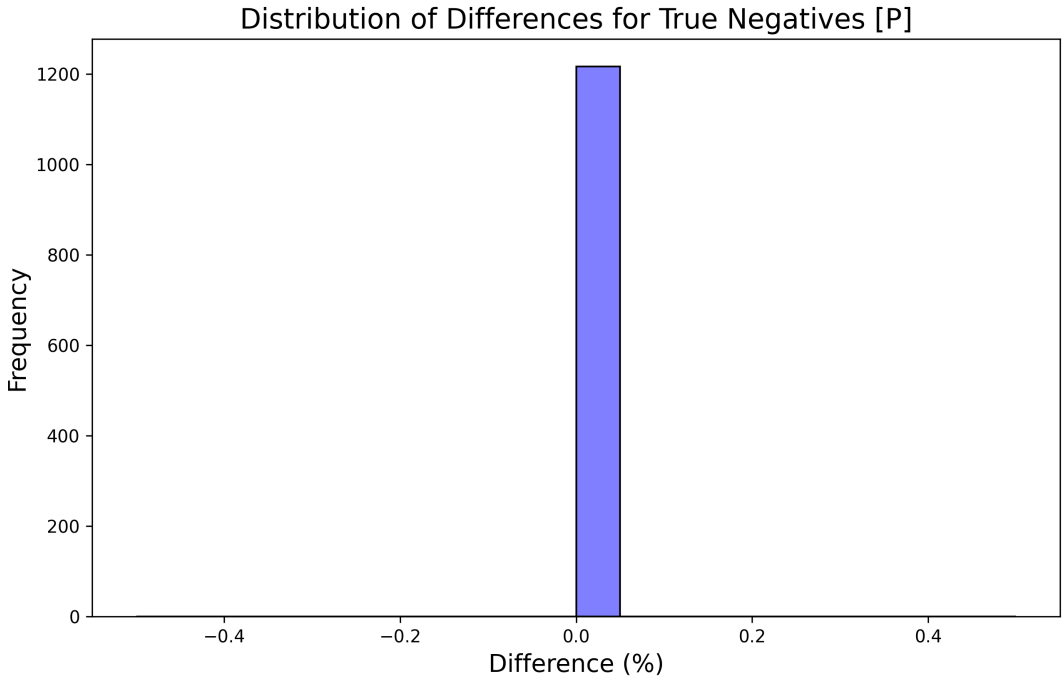
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## Warning in scale_y_continuous(trans = "log10"): log-10 transformation
## introduced infinite values.
```

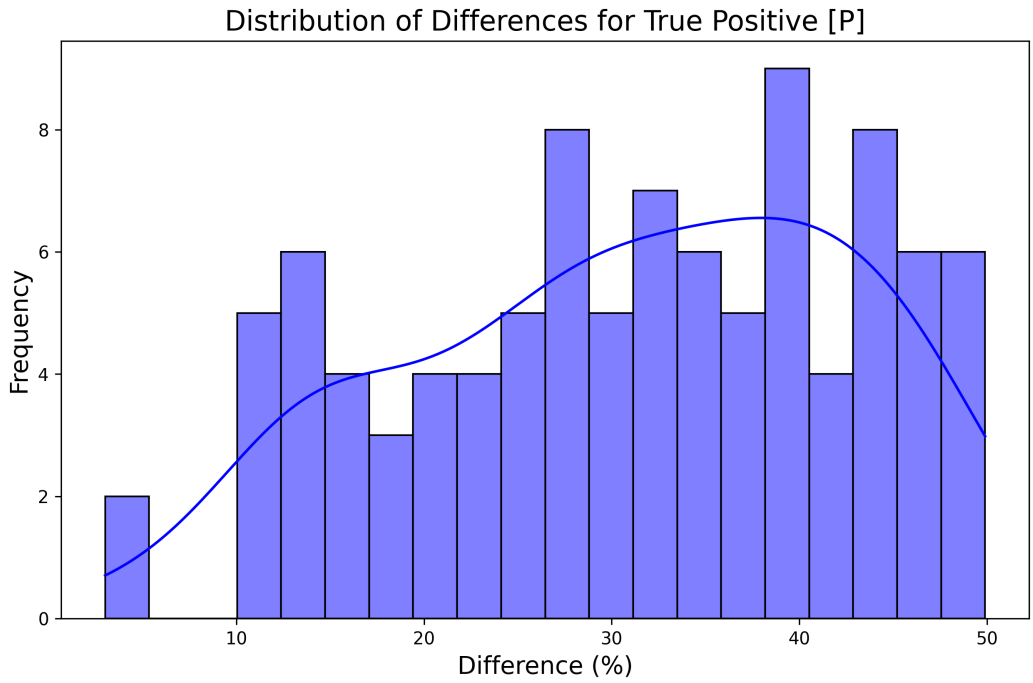

Volcano Plot Beta vs P−value
Binary covar [stoat vcf]
10.00
1.00
0.10
0.01
−log10(P)
−10
−5
0
5
10
Effect Size (BETA)

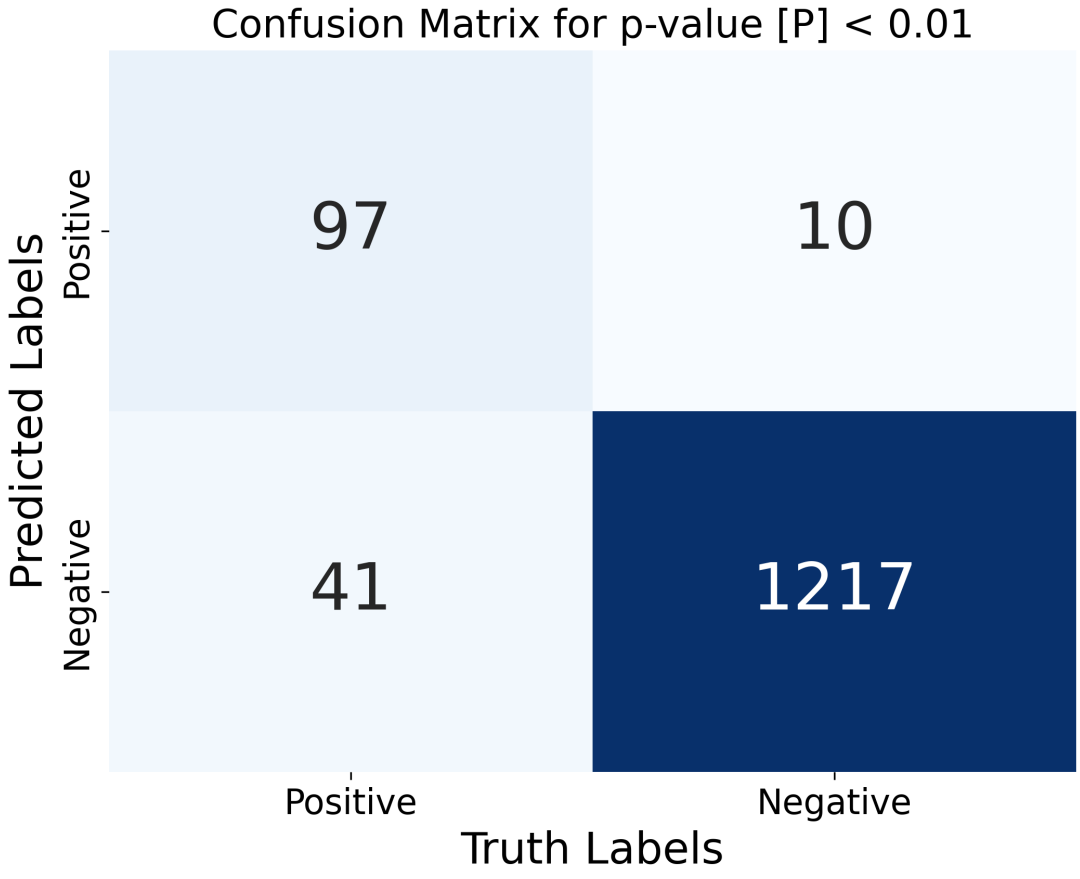

Distribution of P-Values for FP, FN & TP [P] with threshold Positive > 0% difference
Simulated Effect (Difference in Probabilities)
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1
P-Value
Type
False Positives
True Positives
False Negatives

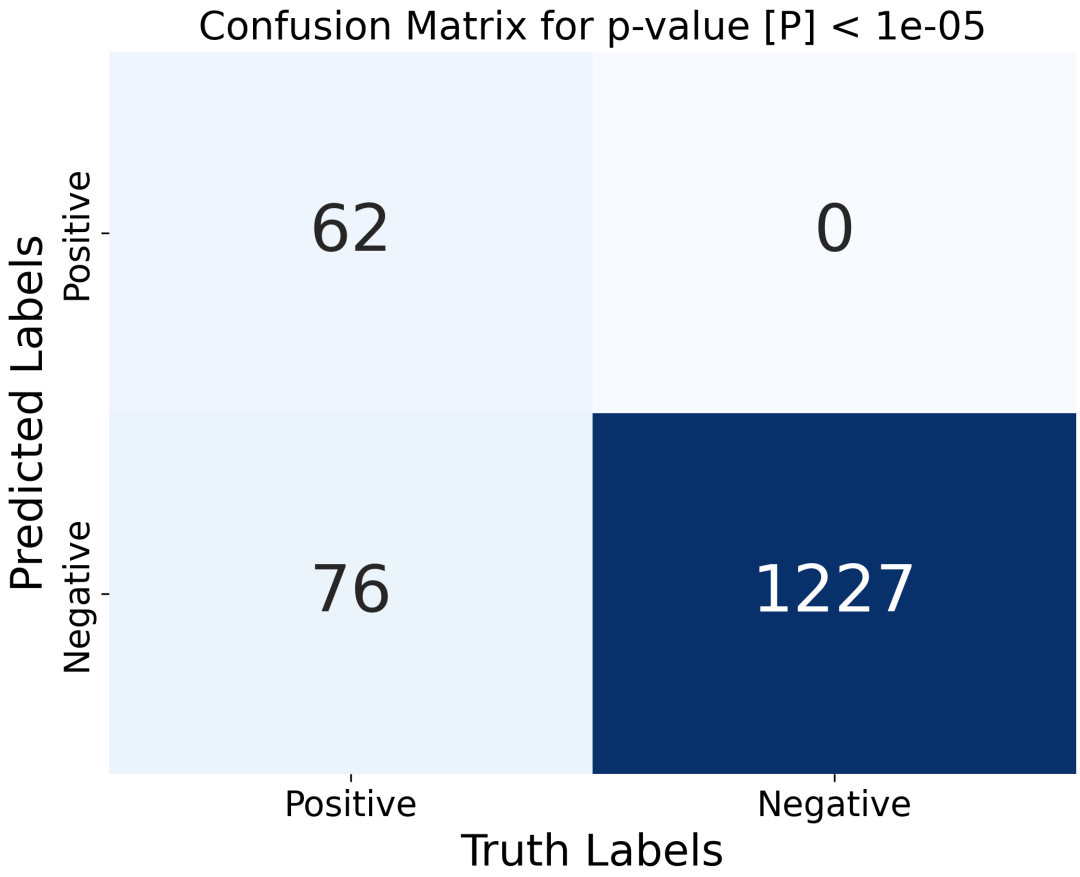

Distribution of Differences for False Negative [P]
Frequency
8
7
6
5
4
3
2
1
0
0
10
20
30
40
50
Difference (%)

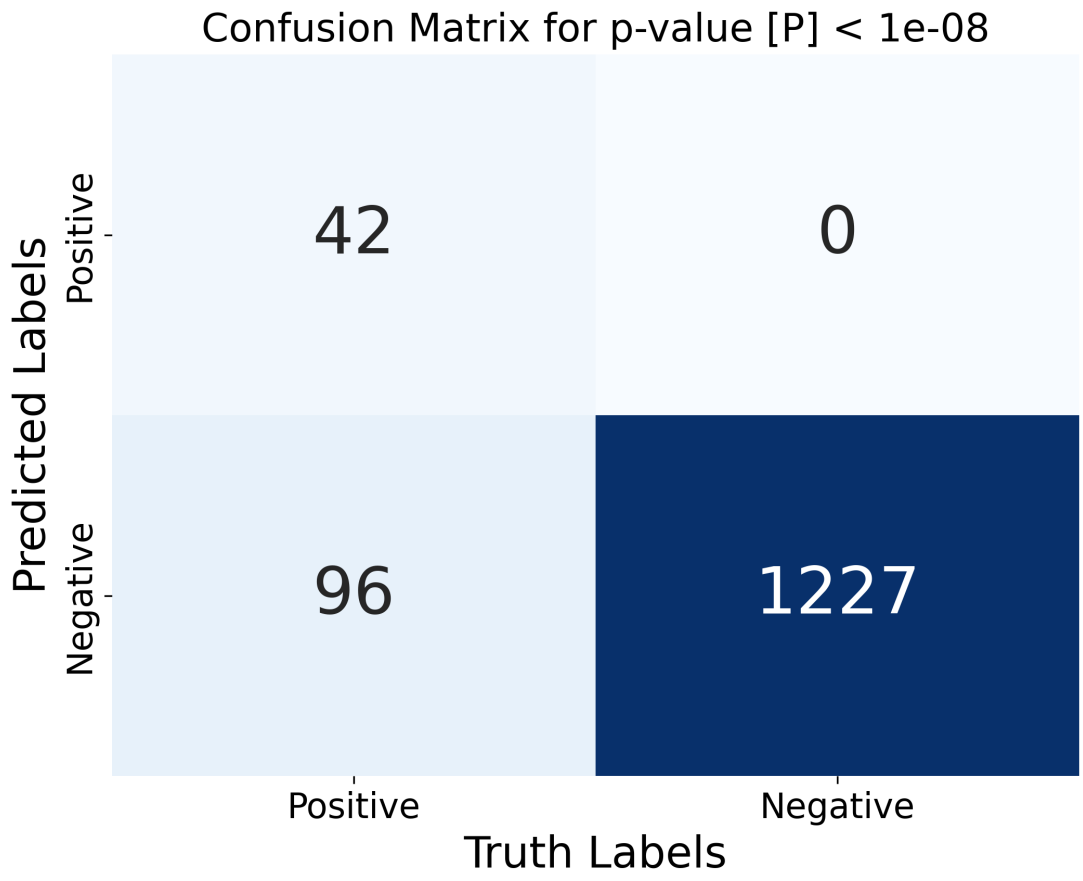

Distribution of Differences for False Positive [P]
Frequency
Difference (%)
0
2
4
6
8
10
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Negatives [P]
Frequency
Difference (%)
0
200
400
600
800
1000
1200
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Positive [P]
Frequency
Difference (%)
0
2
4
6
8
10
20
30
40
50


Confusion Matrix for p-value [P] < 0.01
Predicted Labels
Positive
Negative
97
10
41
1217
Positive
Negative
Truth Labels


Confusion Matrix for p-value [P] < 1e-05
62
0
76
1227
Positive
Negative
Predicted Labels
Positive
Negative
Truth Labels



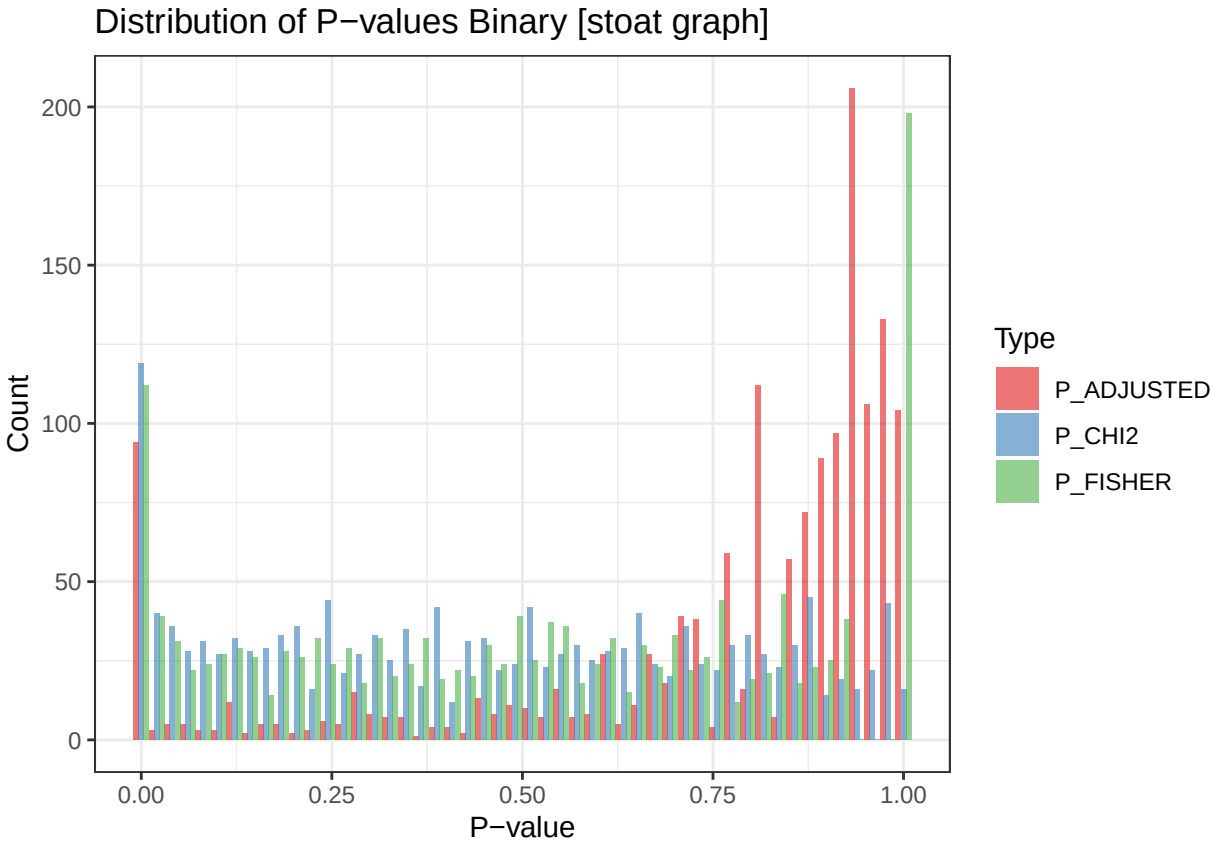
Binary stoat GRAPH

```
binary_output <- run_verify_truth(
  freq_file = file_list$binary_freq,
  pvalue_file = file_list$binary_graph,
  paths_file = file_list$binary_snarl,
  flag = " -b ",
  output_dir = "../binary_graph_output"
)

cat(binary_output, sep = "\n")
```

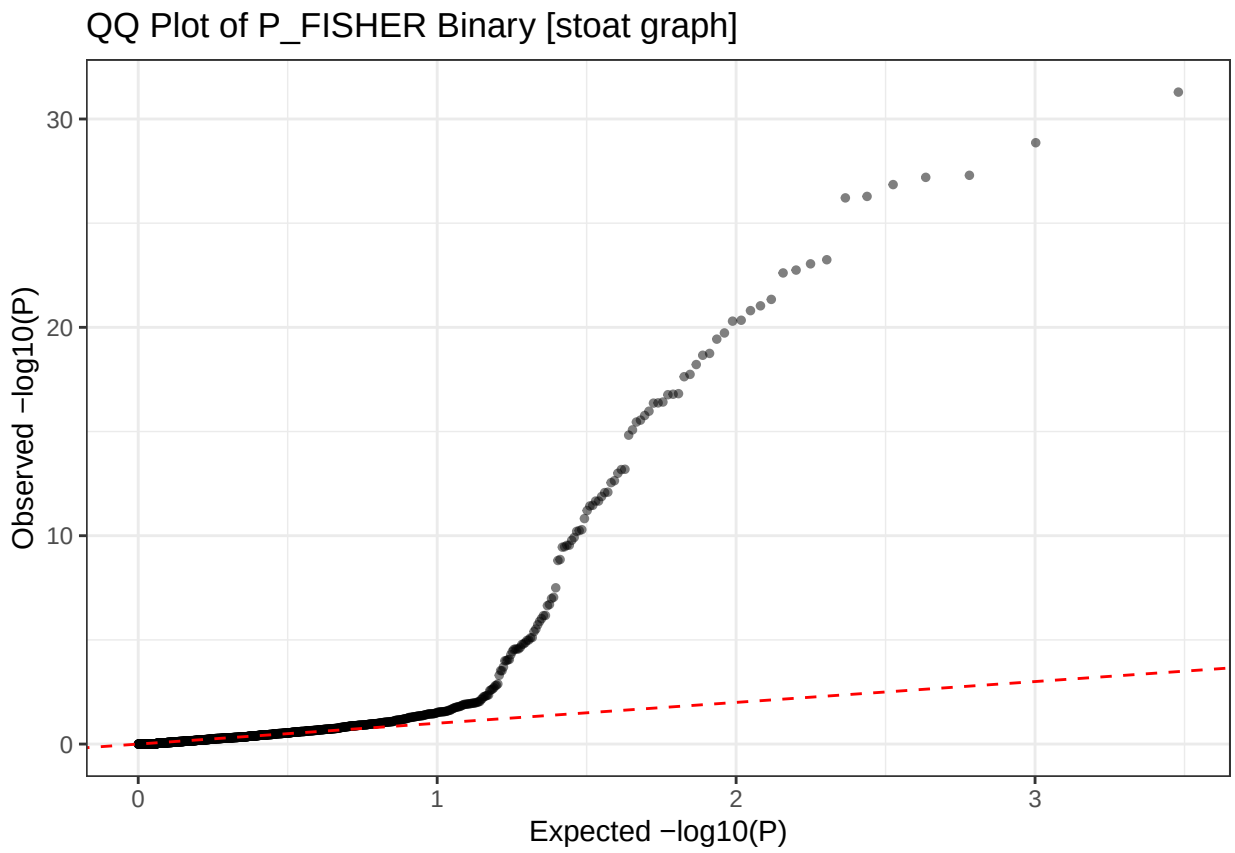
```
## Output directory already exists: ../binary_graph_output
##
## Metrics for p-value < 0.01:
## Confusion Matrix for p-value [P] < 0.01:
## [[ 108    9]
##  [  33 1358]]
## Precision: 0.993
```

```
## Recall: 0.976
## F1 Score: 0.985
##
## Metrics for p-value < 1e-05:
## Confusion Matrix for p-value [P] < 1e-05:
## [[  76    0]
##  [  65 1367]]
## Precision: 1.000
## Recall: 0.955
## F1 Score: 0.977
##
## Metrics for p-value < 1e-08:
## Confusion Matrix for p-value [P] < 1e-08:
## [[  60    0]
##  [  81 1367]]
## Precision: 1.000
## Recall: 0.944
## F1 Score: 0.971
##
## Pourcentage of paths (present in freq file) tested :  98.9501312335958
```

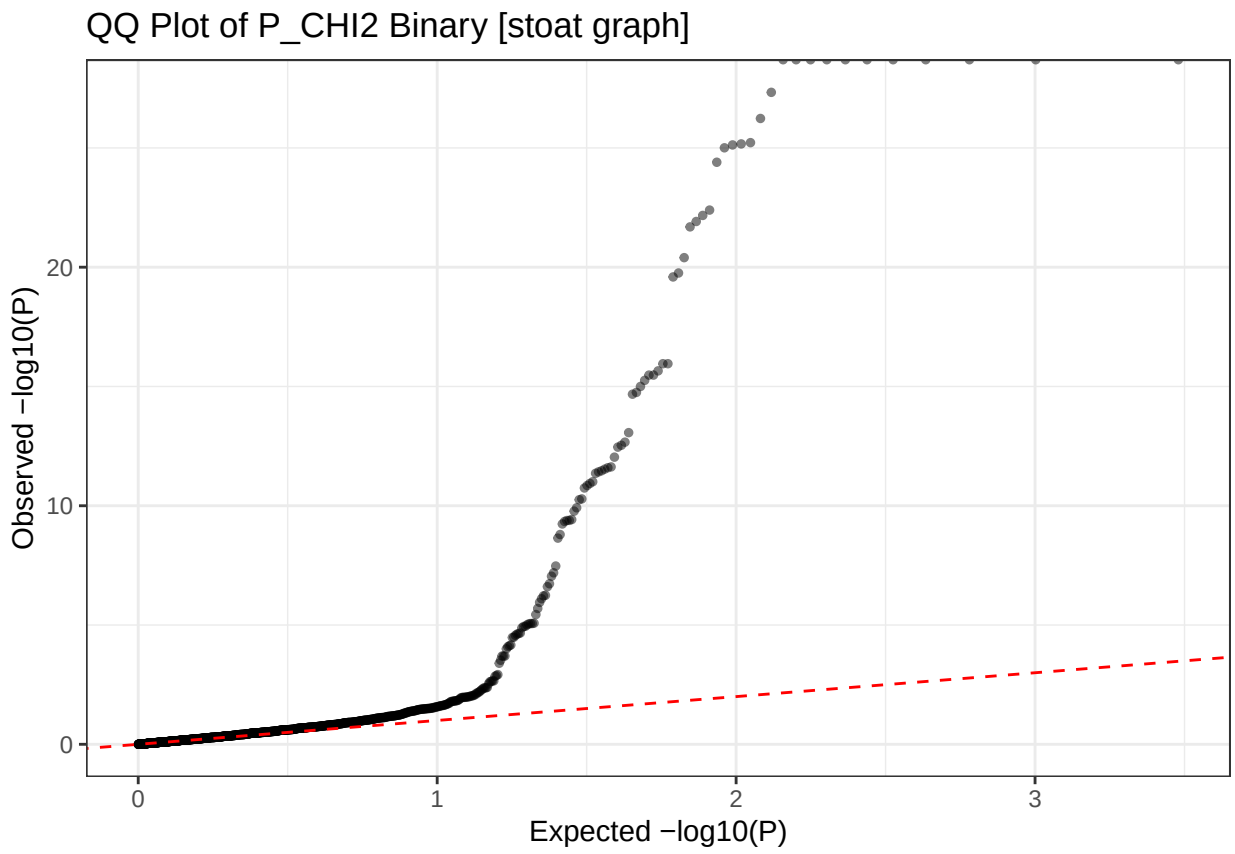
```r
df_binary_graph <- read_stoat(file_list$binary_graph)

# Plots
histogram_plot(df=df_binary_graph, subtitle="Binary [stoat graph]", "P_FISHER", "P_CHI2", "P_ADJUSTED")
```



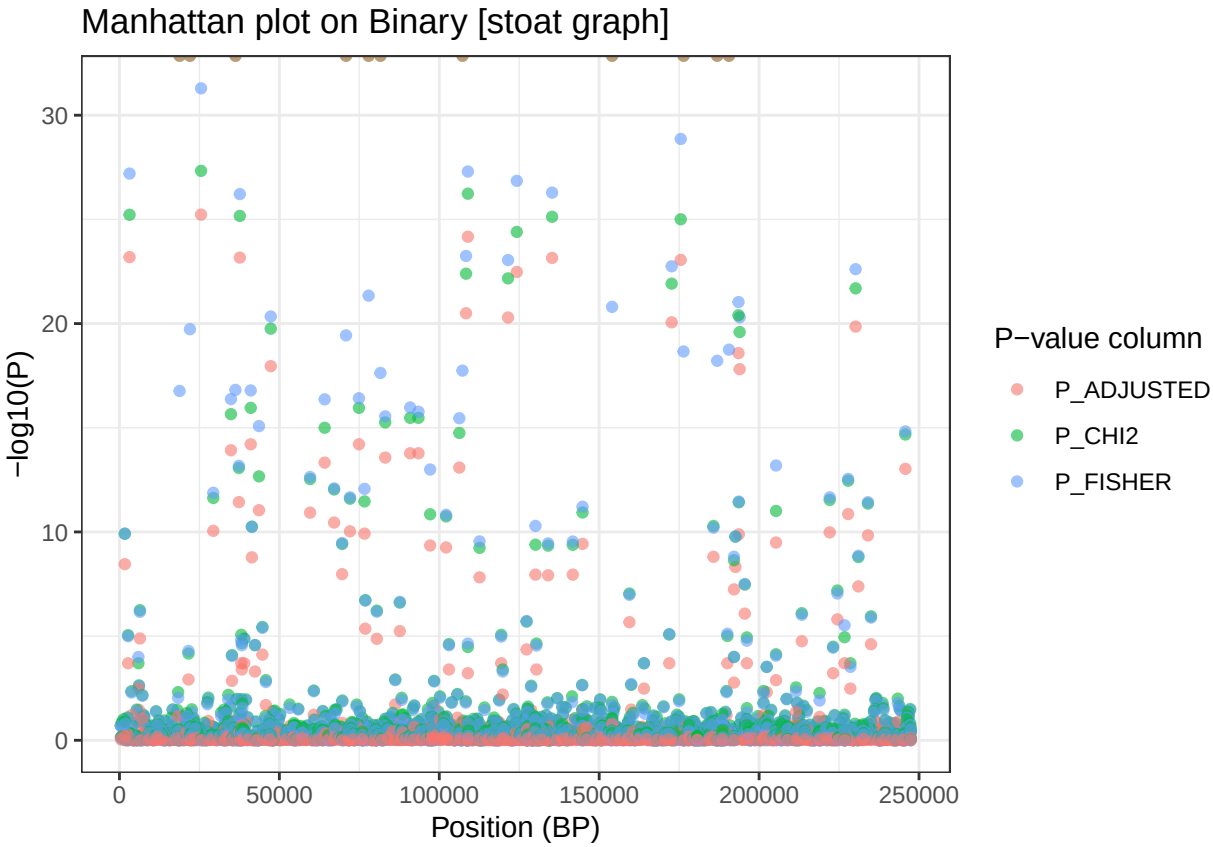
```
qq_plot(df_binary_graph, "P_FISHER", subtitle="Binary [stoat graph]")
```

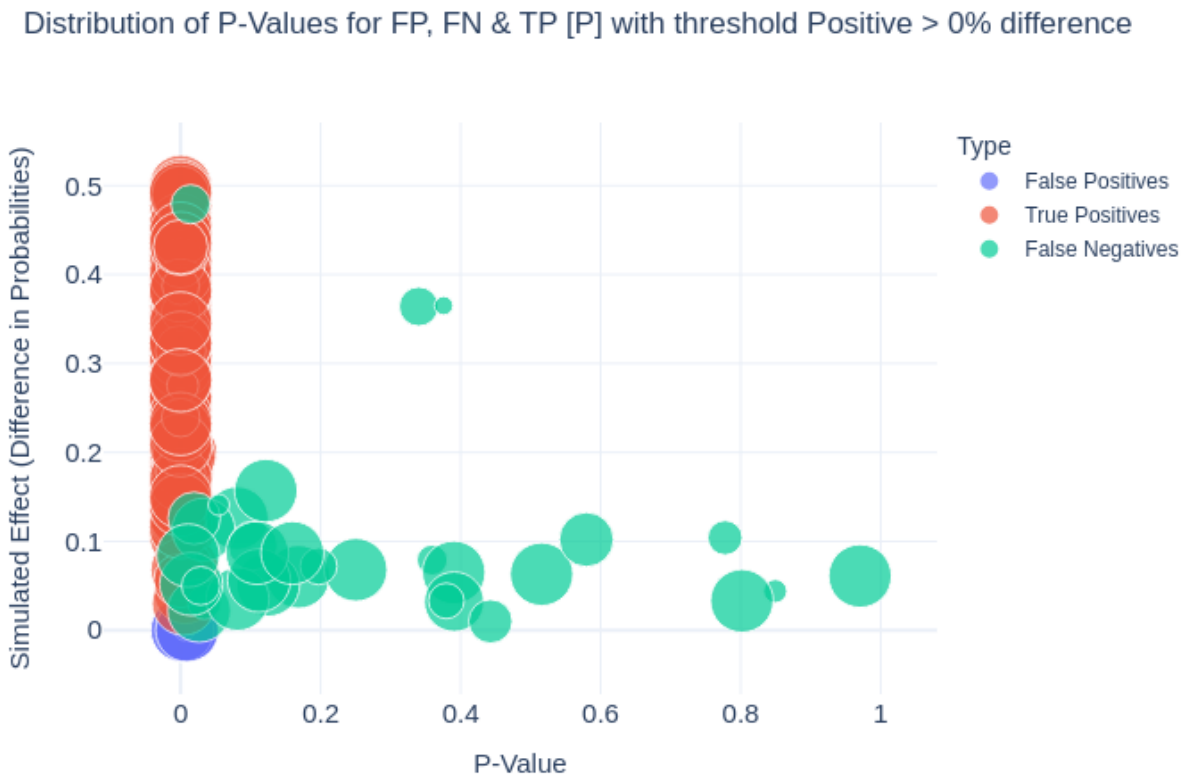


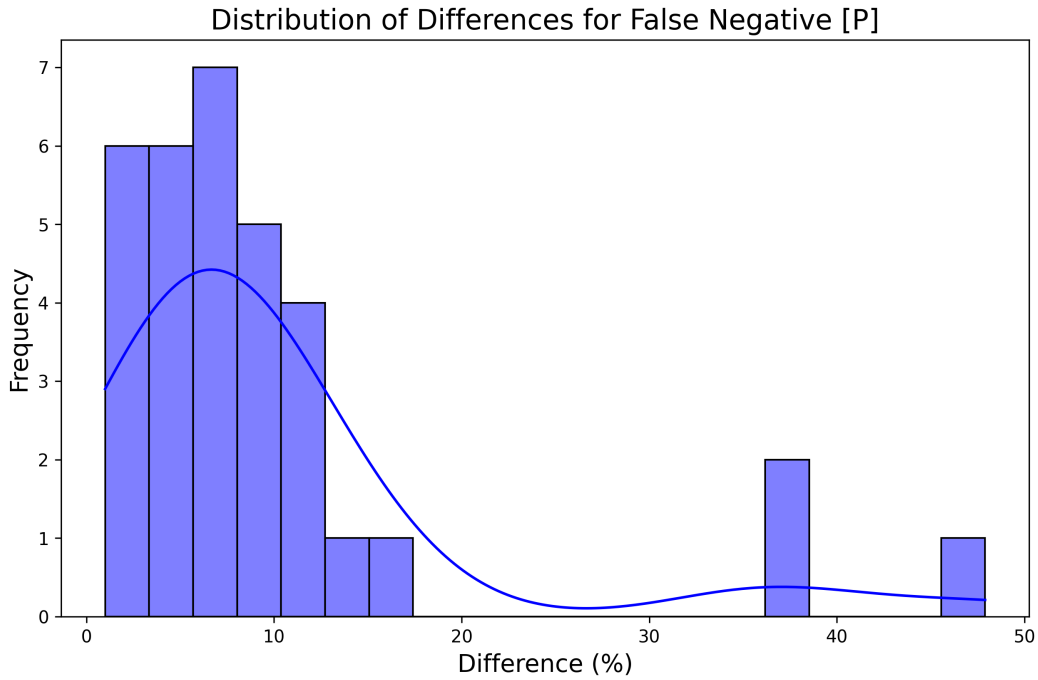

```
qq_plot(df_binary_graph, "P_CHI2", subtitle="Binary [stoat graph]")
```

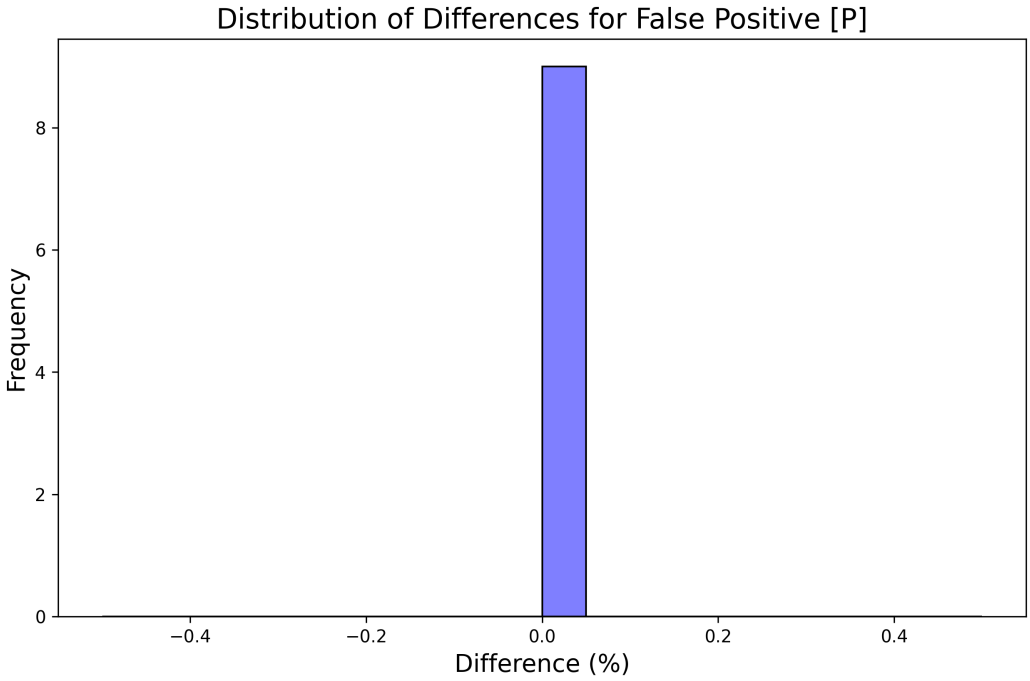


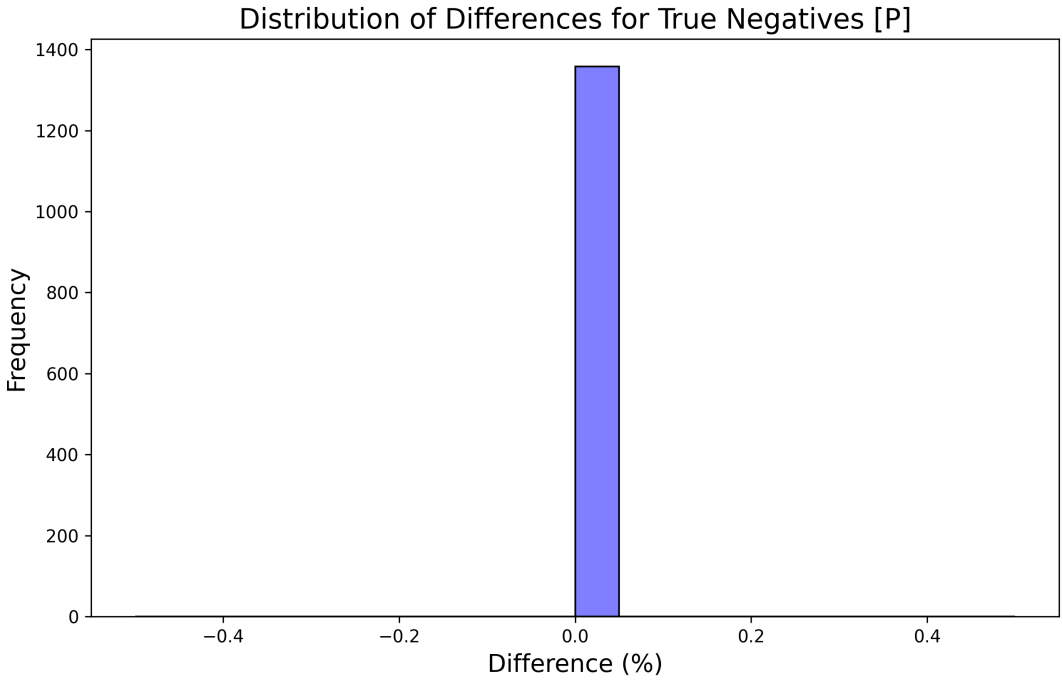

```
manhattan_plot(df_binary_graph, subtitle="on Binary [stoat graph]", "P_CHI2", "P_FISHER", "P_ADJUSTED")
```

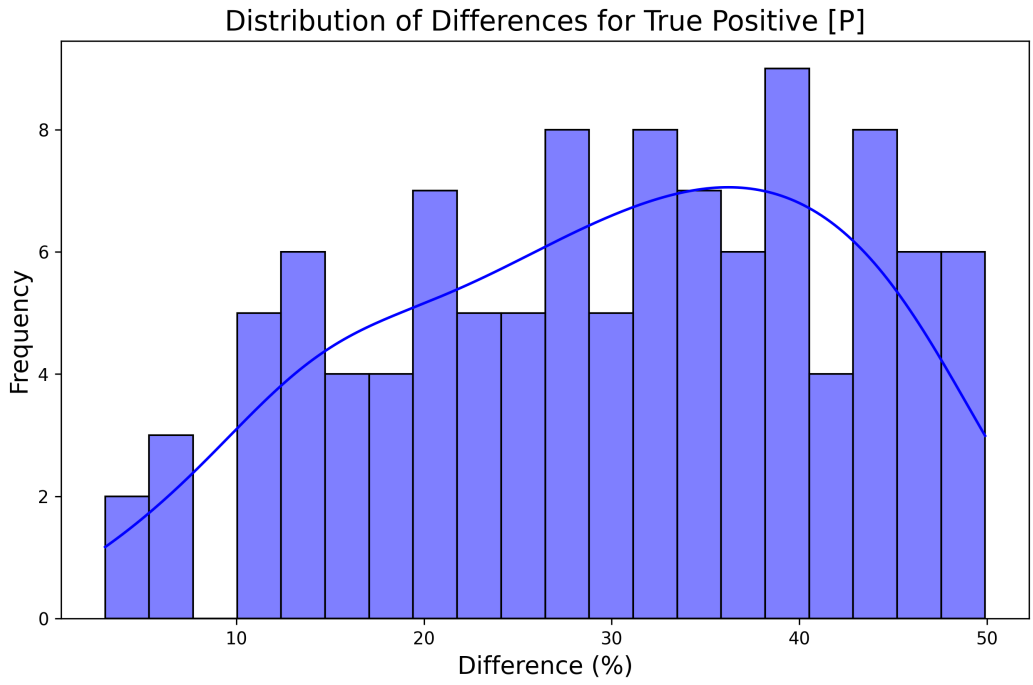

Manhattan plot on Binary [stoat graph]
−log10(P)
30
20
10
0
0
50000
100000
150000
200000
250000
Position (BP)
P−value column
P_ADJUSTED
P_CHI2
P_FISHER


Distribution of P-Values for FP, FN & TP [P] with threshold Positive > 0% difference
Simulated Effect (Difference in Probabilities)
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1
P-Value
Type
False Positives
True Positives
False Negatives


Distribution of Differences for False Negative [P]
Frequency
7
6
5
4
3
2
1
0
0
10
20
30
40
50
Difference (%)


Distribution of Differences for False Positive [P]
Frequency
Difference (%)
0
2
4
6
8
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Negatives [P]
Frequency
Difference (%)
0
200
400
600
800
1000
1200
1400
−0.4
−0.2
0.0
0.2
0.4



Compare stoat graph vs stoat vcf binary output

```
# Compare each columns by matching SNARL
# Merge the data frames by SNARL
merged_df <- merge(df_binary_vcf, df_binary_graph, by = "SNARL", suffixes = c("_df1", "_df2"))

# Get the list of columns to compare (excluding SNARL)
columns_to_compare <- setdiff(names(df_binary_vcf), "SNARL")

# Initialize a named vector to store number of differing rows per column
column_diff_counts <- setNames(integer(length(columns_to_compare)), columns_to_compare)

# Loop through each column and count differing rows
for (col in columns_to_compare) {
  col1 <- paste0(col, "_df1")
  col2 <- paste0(col, "_df2")

  # Count where values differ (using `!=` and NA-safe logic)
  diffs <- merged_df[[col1]] != merged_df[[col2]]

  # Handle NAs: count rows where one is NA and the other is not, or values differ
  diffs[is.na(diffs)] <- xor(is.na(merged_df[[col1]]), is.na(merged_df[[col2]]))[is.na(diffs)]

  column_diff_counts[col] <- sum(diffs, na.rm = TRUE)
}

# Convert to data frame for pretty output
result <- data.frame(Column = names(column_diff_counts),
```

```
                    Num_Different_Rows = as.integer(column_diff_counts))

# Print result
print(result)
```

```
##          Column Num_Different_Rows
## 1           CHR                  0
## 2     START_POS                  0
## 3       END_POS                  0
## 4  PATH_LENGTHS                  0
## 5      P_FISHER               1295
## 6        P_CHI2               1439
## 7    P_ADJUSTED               1449
## 8   GROUP_PATHS               1451
## 9         DEPTH                  0
## 10          POS                  0
```

This change can be explain because stoat graph use 'ref' haplotype as an correct haplotype where in the simulation it's not.

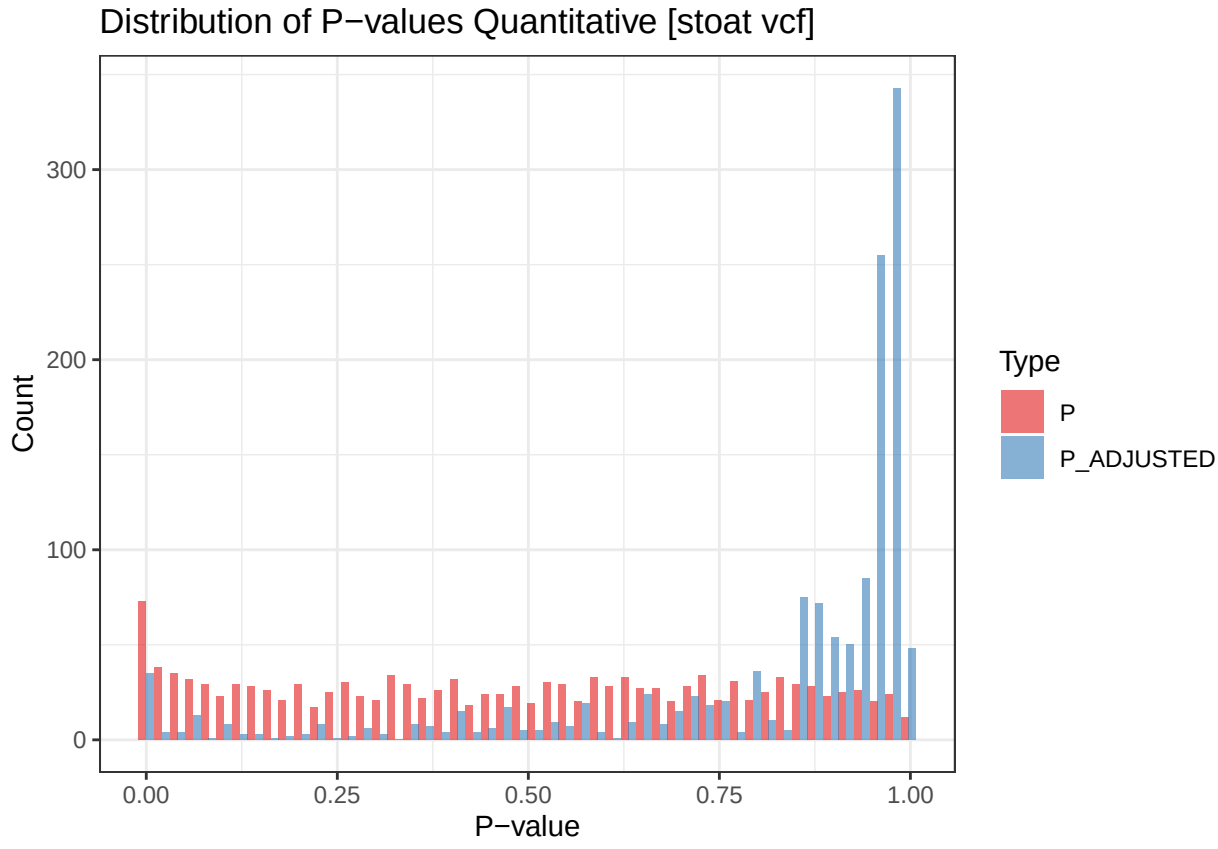
Quantitative stoat VCF

```
# Execute verification for quantitative VCF
quantitative_output <- run_verify_truth(
  freq_file = file_list$quantitative_freq,
  pvalue_file = file_list$quantitative_vcf,
  paths_file = file_list$quantitative_snarl,
  flag = " -q ",
  output_dir = "../quantitative_output"
)

cat(quantitative_output, sep = "\n")
```

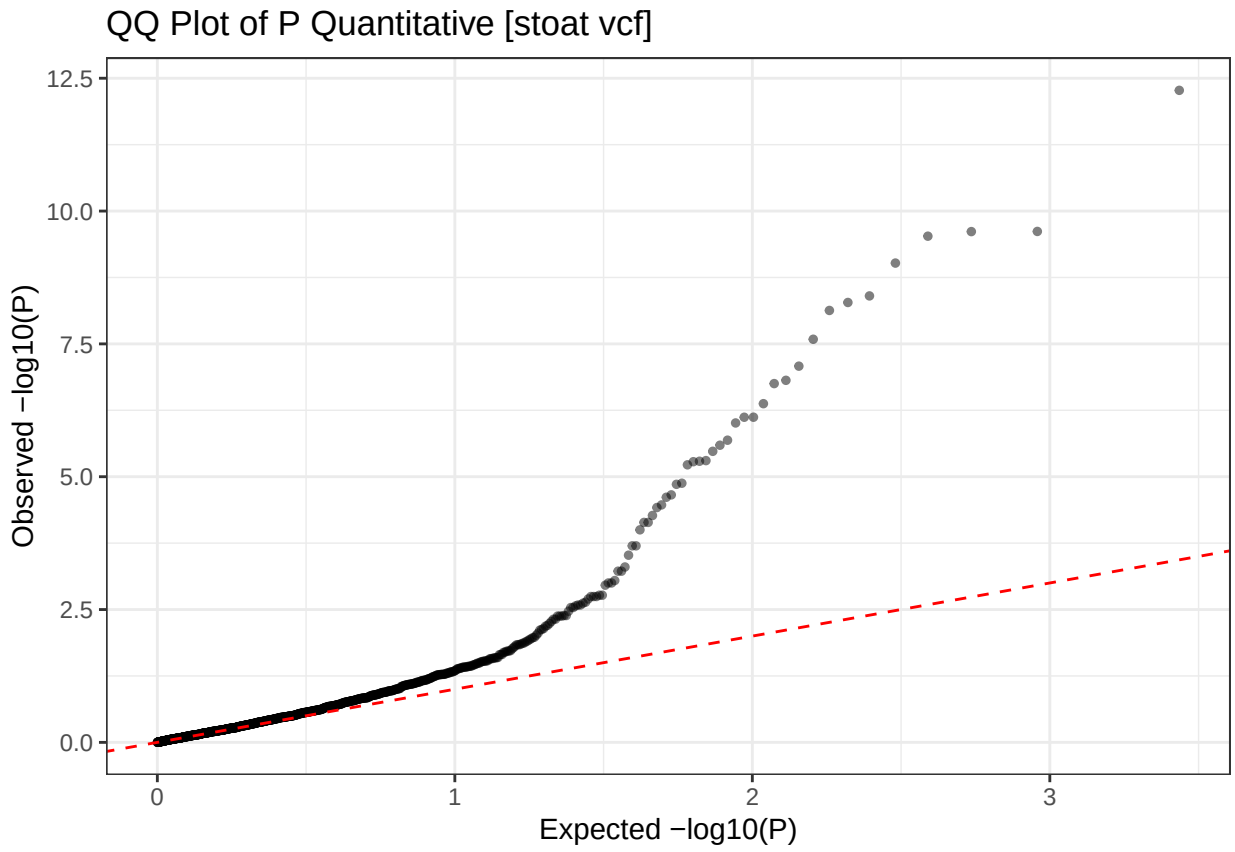
```
## Output directory already exists: ../quantitative_output
##
## Metrics for p-value < 0.01:
## Confusion Matrix for p-value [P] < 0.01:
## [[  63   10]
##  [  84 1205]]
## Precision: 0.992
## Recall: 0.935
## F1 Score: 0.962
##
## Metrics for p-value < 1e-05:
## Confusion Matrix for p-value [P] < 1e-05:
## [[  23    0]
##  [ 124 1215]]
## Precision: 1.000
```

```
## Recall: 0.907
## F1 Score: 0.951
##
## Metrics for p-value < 1e-08:
## Confusion Matrix for p-value [P] < 1e-08:
## [[   8    0]
##  [ 139 1215]]
## Precision: 1.000
## Recall: 0.897
## F1 Score: 0.946
##
## Pourcentage of paths (present in freq file) tested :  90.86057371581055
```

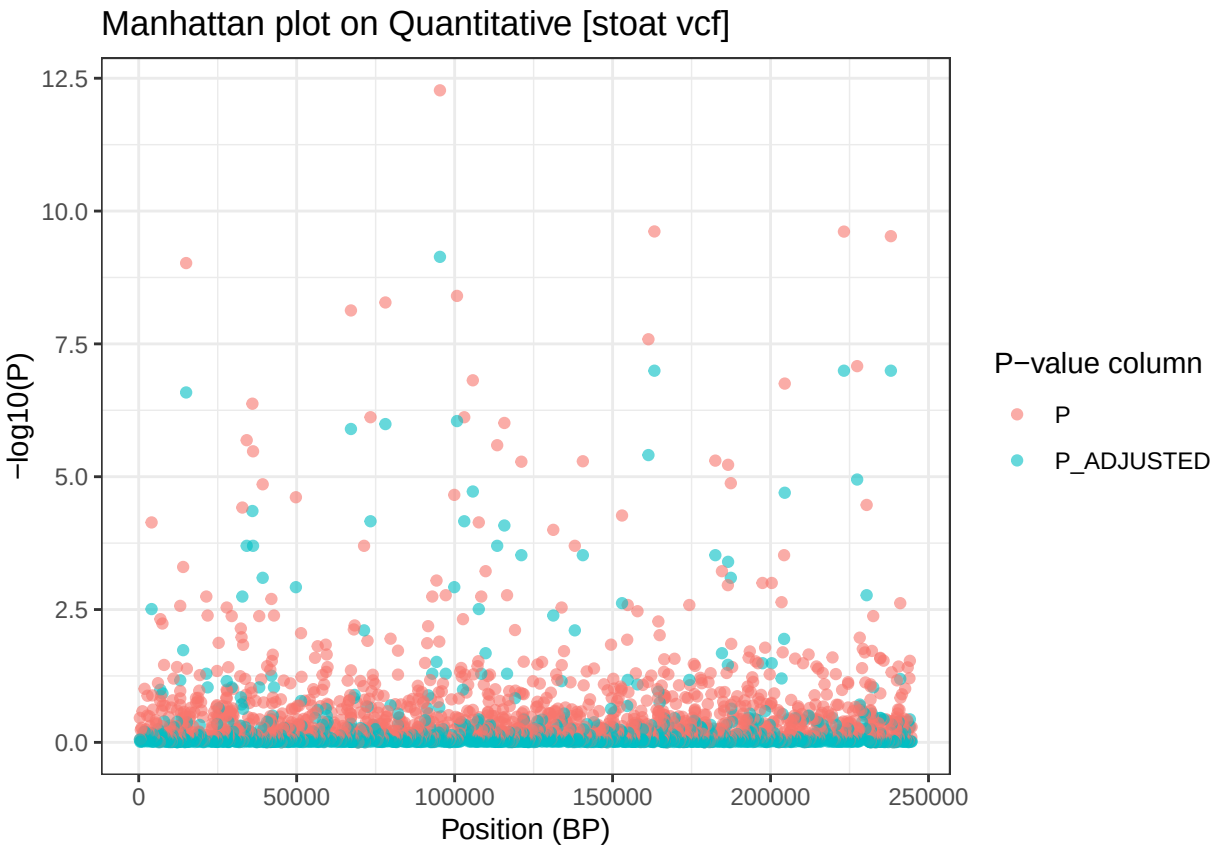
```
df_quantitative_vcf <- read_stoat(file_list$quantitative_vcf)

# ---- Plots ----
histogram_plot(df=df_quantitative_vcf, subtitle="Quantitative [stoat vcf]", "P", "P_ADJUSTED")
```



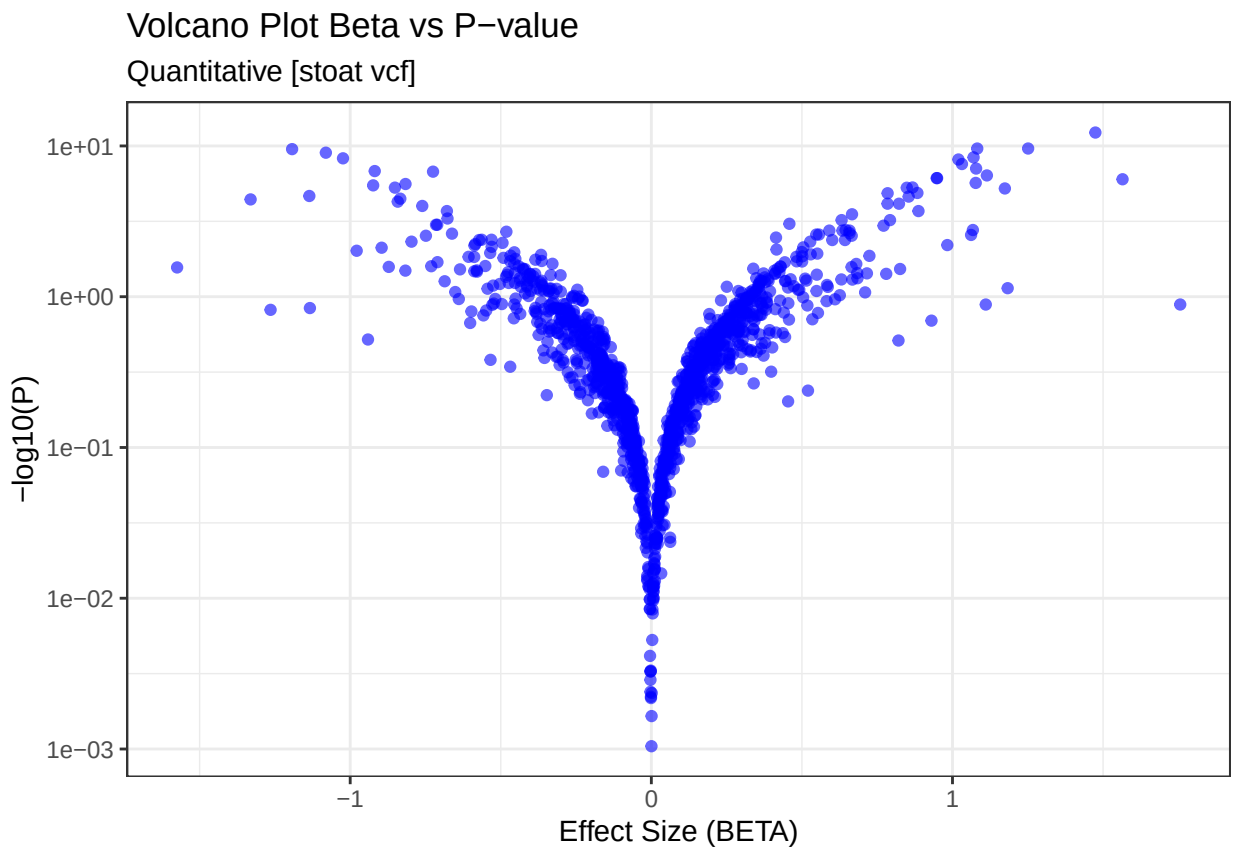

```
qq_plot(df_quantitative_vcf, "P", subtitle="Quantitative [stoat vcf]")
```



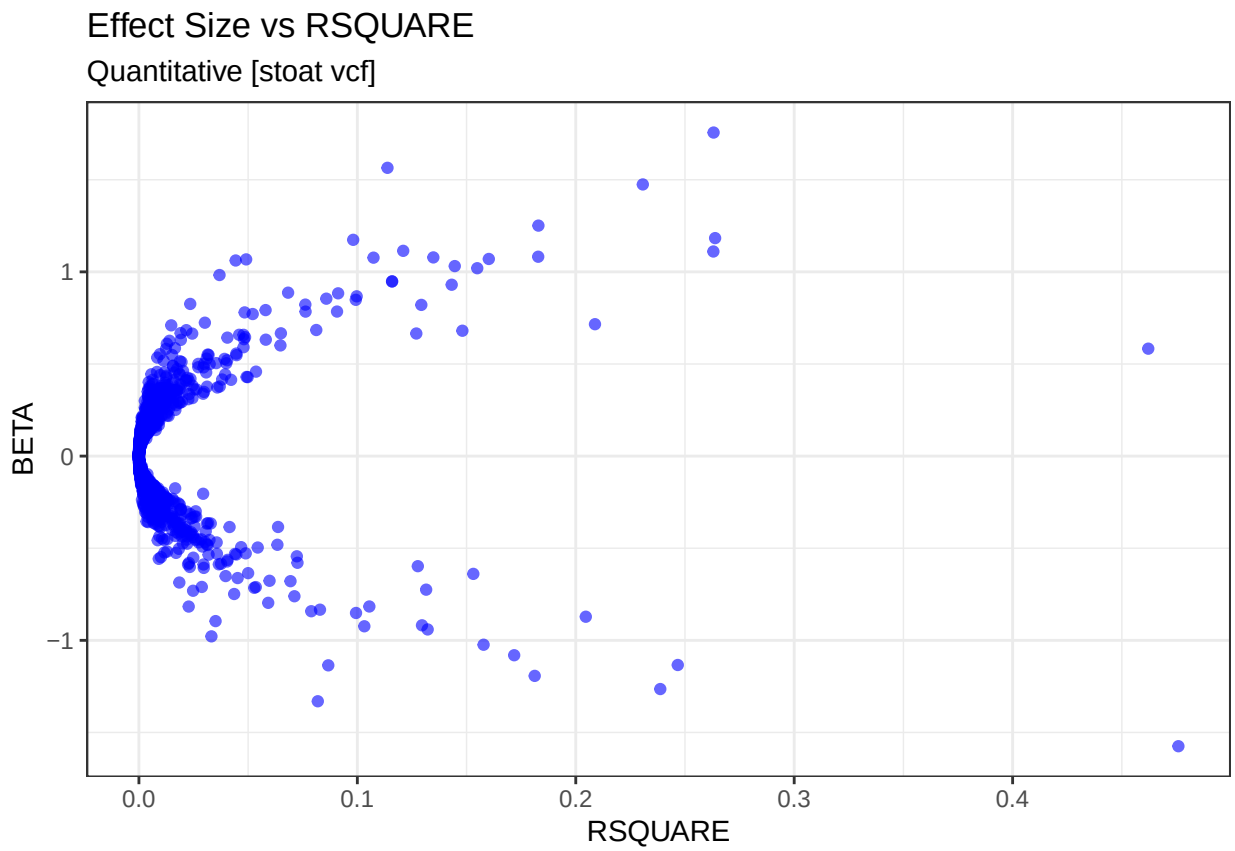

```
manhattan_plot(df_quantitative_vcf, subtitle="on Quantitative [stoat vcf]", "P", "P_ADJUSTED")
```

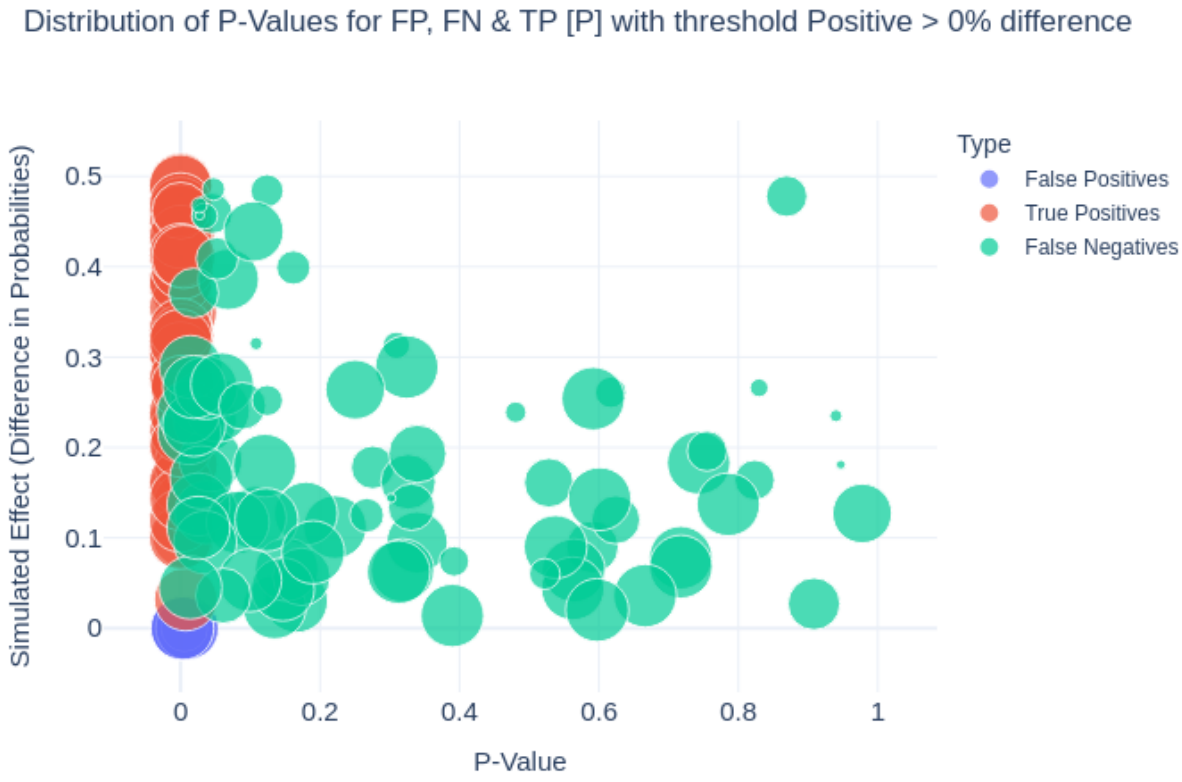


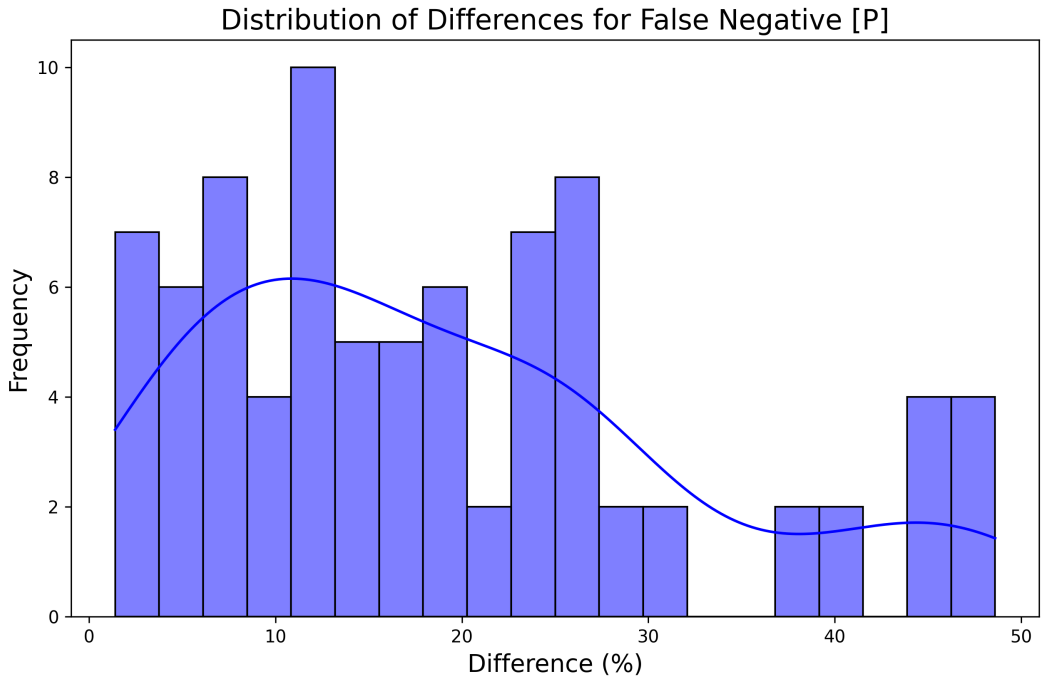

```
volcano_plot(df_quantitative_vcf, subtitle="Quantitative [stoat vcf]")
```

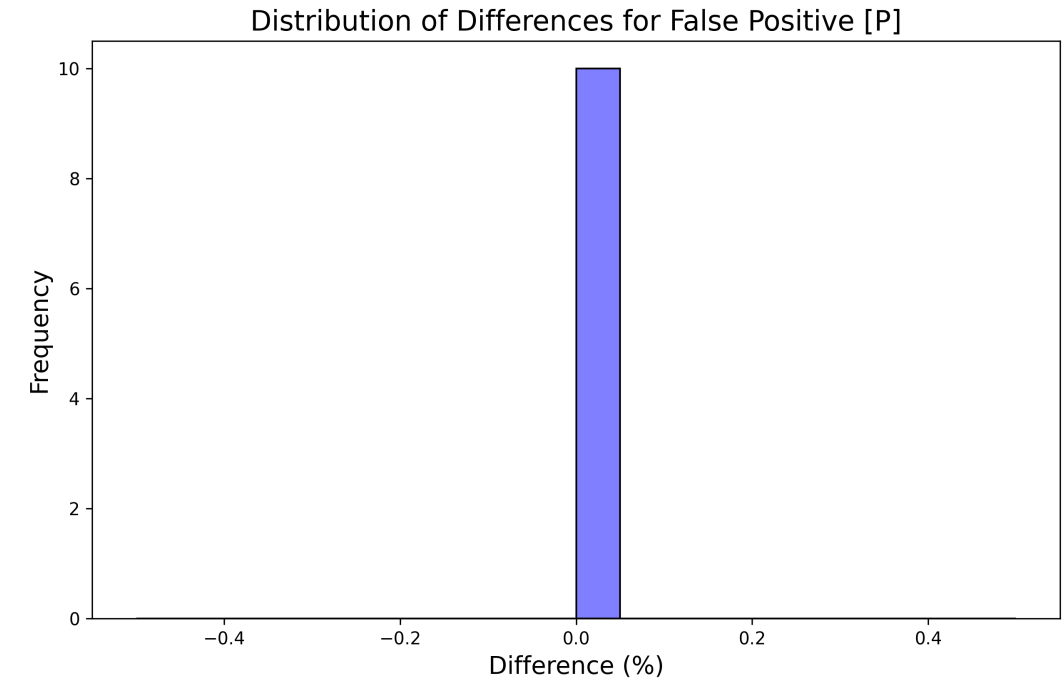


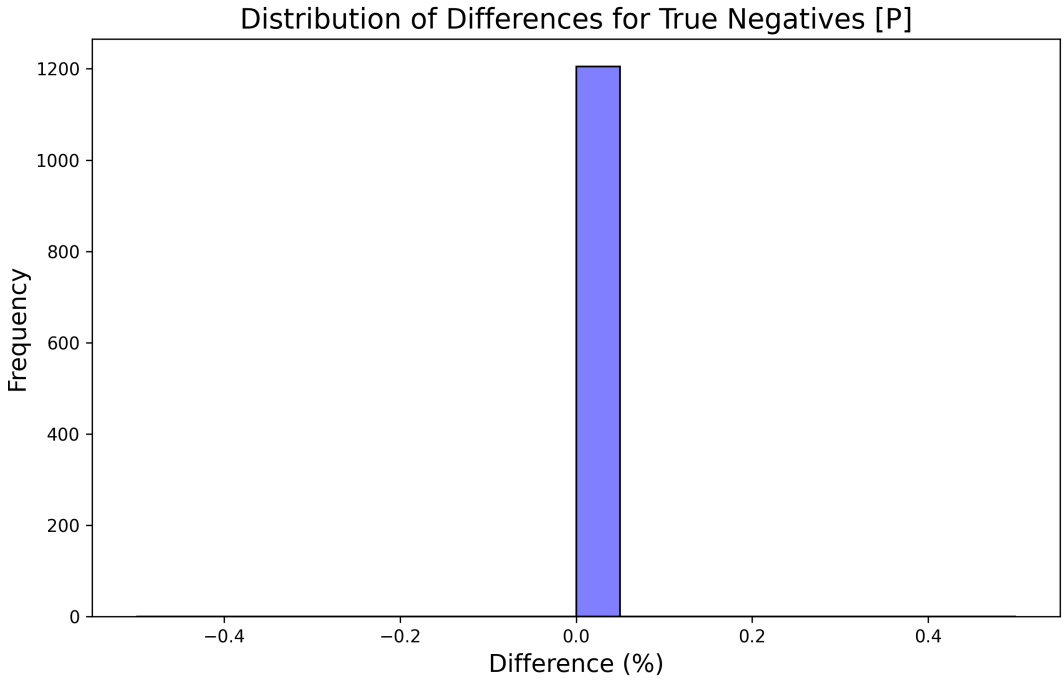

```
rsq_plot(df_quantitative_vcf, subtitle="Quantitative [stoat vcf]")
```

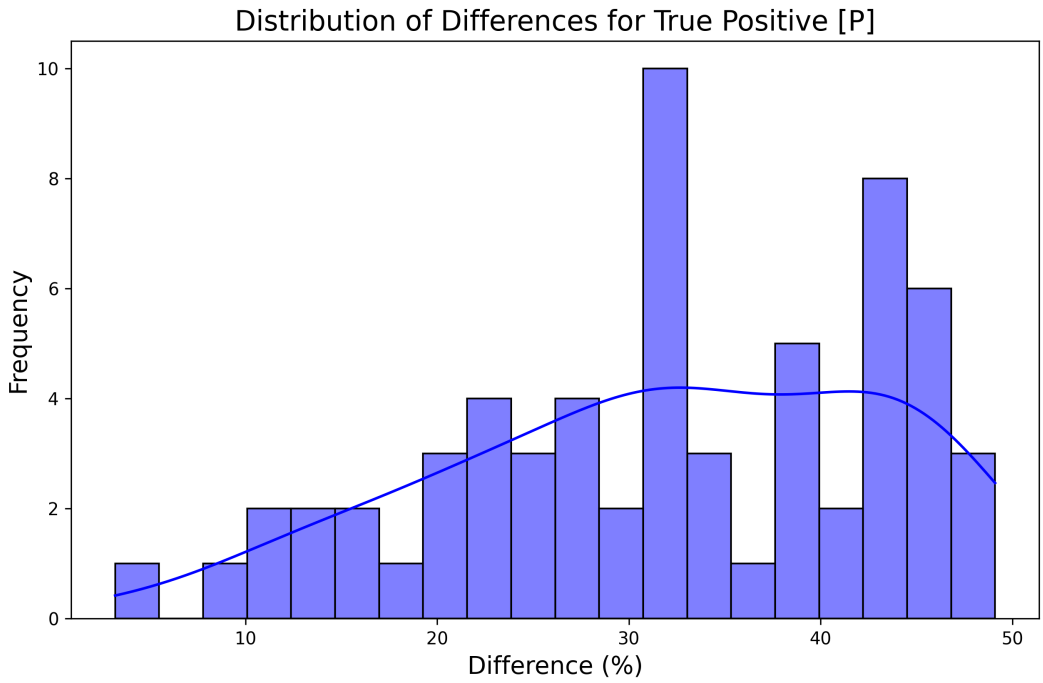

Effect Size vs RSQUARE
Quantitative [stoat vcf]
BETA
1
0
−1
0.0
0.1
0.2
0.3
0.4
RSQUARE

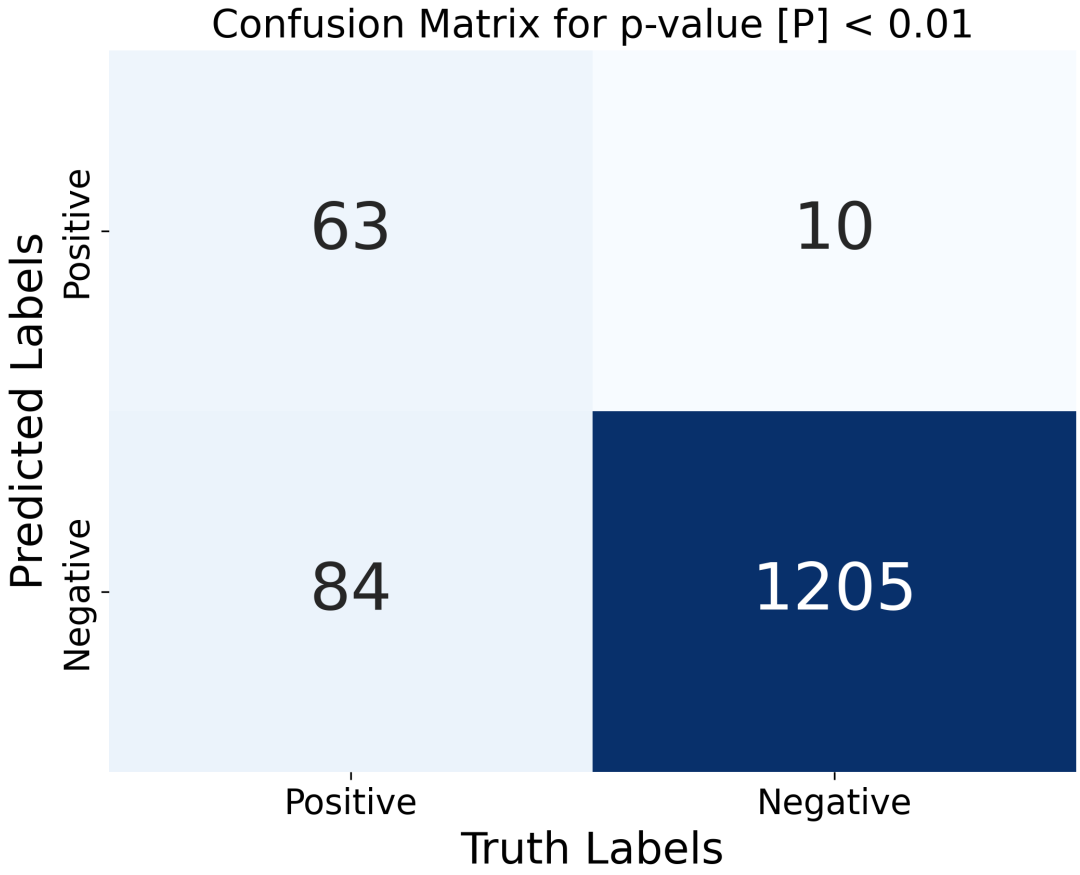

Distribution of P-Values for FP, FN & TP [P] with threshold Positive > 0% difference
Simulated Effect (Difference in Probabilities)
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1
P-Value
Type
False Positives
True Positives
False Negatives

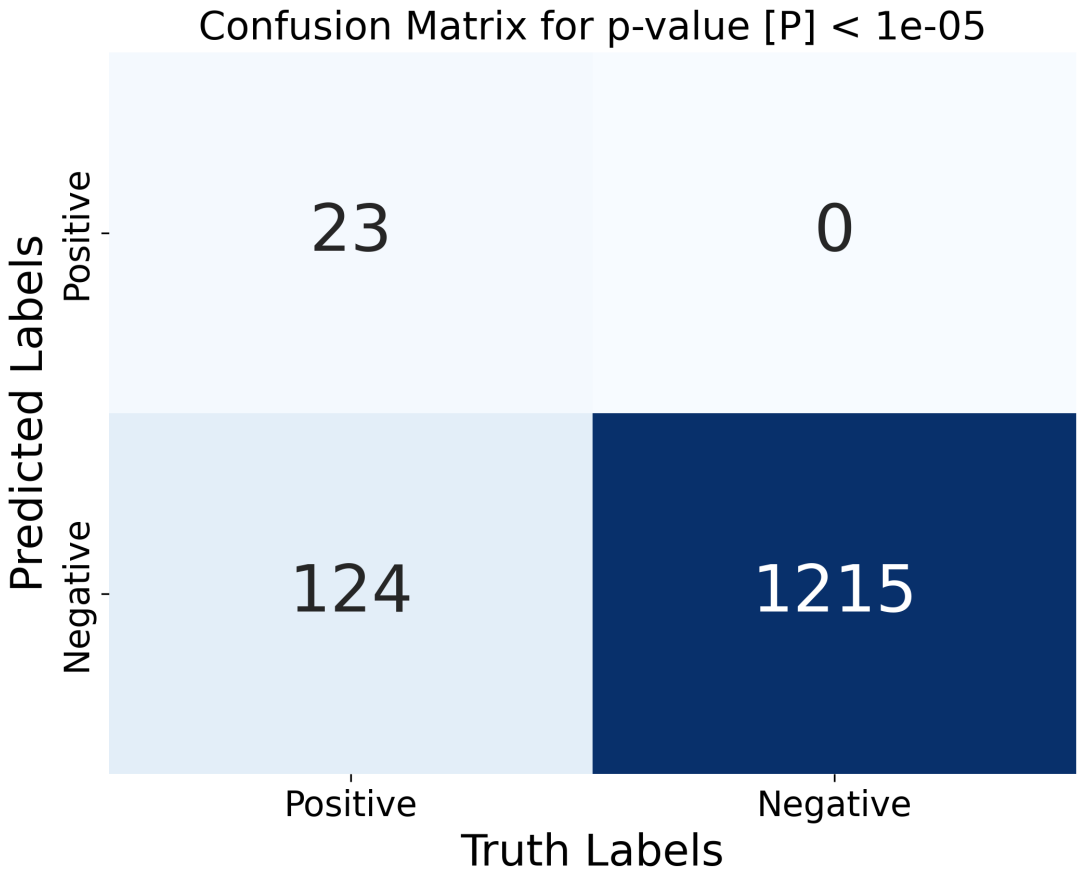

Distribution of Differences for False Negative [P]
Frequency
10
8
6
4
2
0
0
10
20
30
40
50
Difference (%)

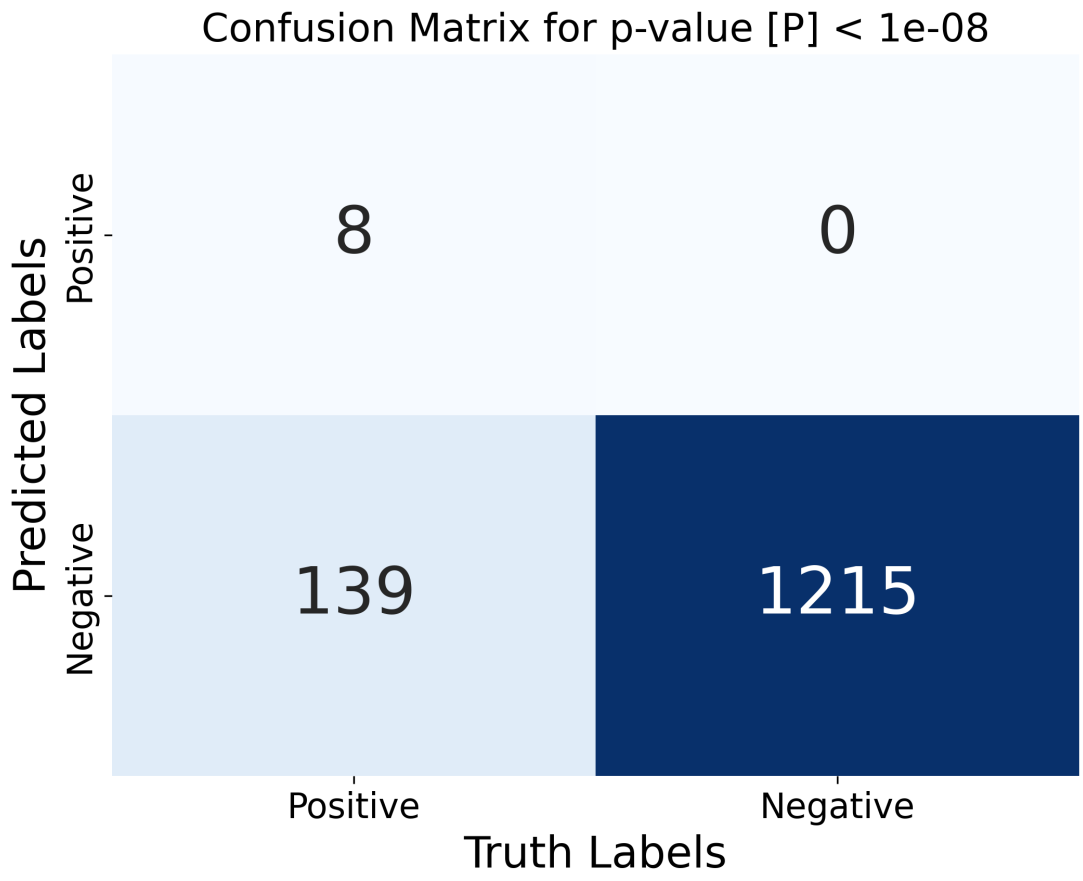

Distribution of Differences for False Positive [P]
Frequency
Difference (%)
0
2
4
6
8
10
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Negatives [P]
Frequency
Difference (%)
0
200
400
600
800
1000
1200
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Positive [P]
Frequency
Difference (%)
0
2
4
6
8
10
10
20
30
40
50


Confusion Matrix for p-value [P] < 0.01
Predicted Labels
Positive
Negative
63
10
84
1205
Positive
Negative
Truth Labels


Confusion Matrix for p-value [P] < 1e-05
Predicted Labels
Positive
Negative
23
0
124
1215
Positive
Negative
Truth Labels



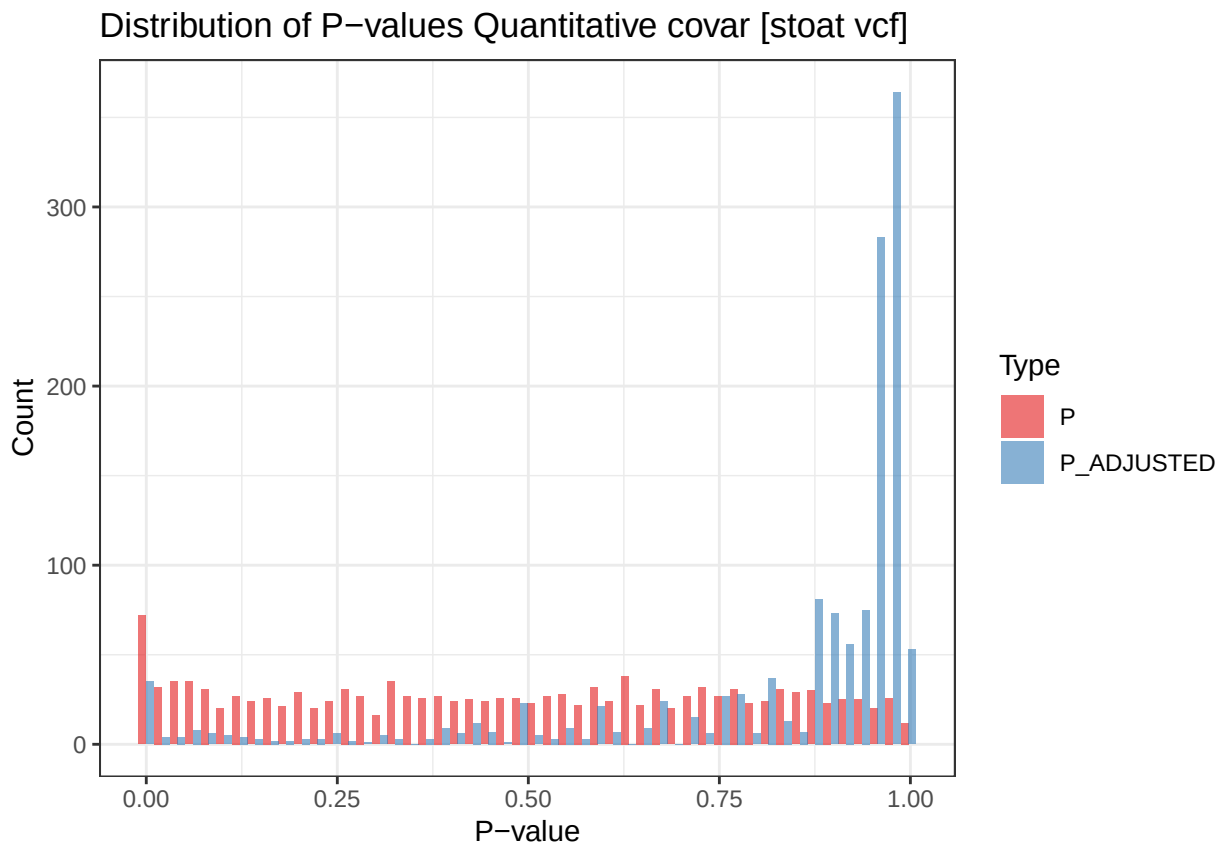
Quantitative covar stoat VCF

```
# Execute verification for quantitative VCF
quantitative_output <- run_verify_truth(
  freq_file = file_list$quantitative_freq,
  pvalue_file = file_list$quantitative_covar_vcf,
  paths_file = file_list$quantitative_snarl,
  flag = " -q ",
  output_dir = "../quantitative_covar_output"
)

cat(quantitative_output, sep = "\n")
```

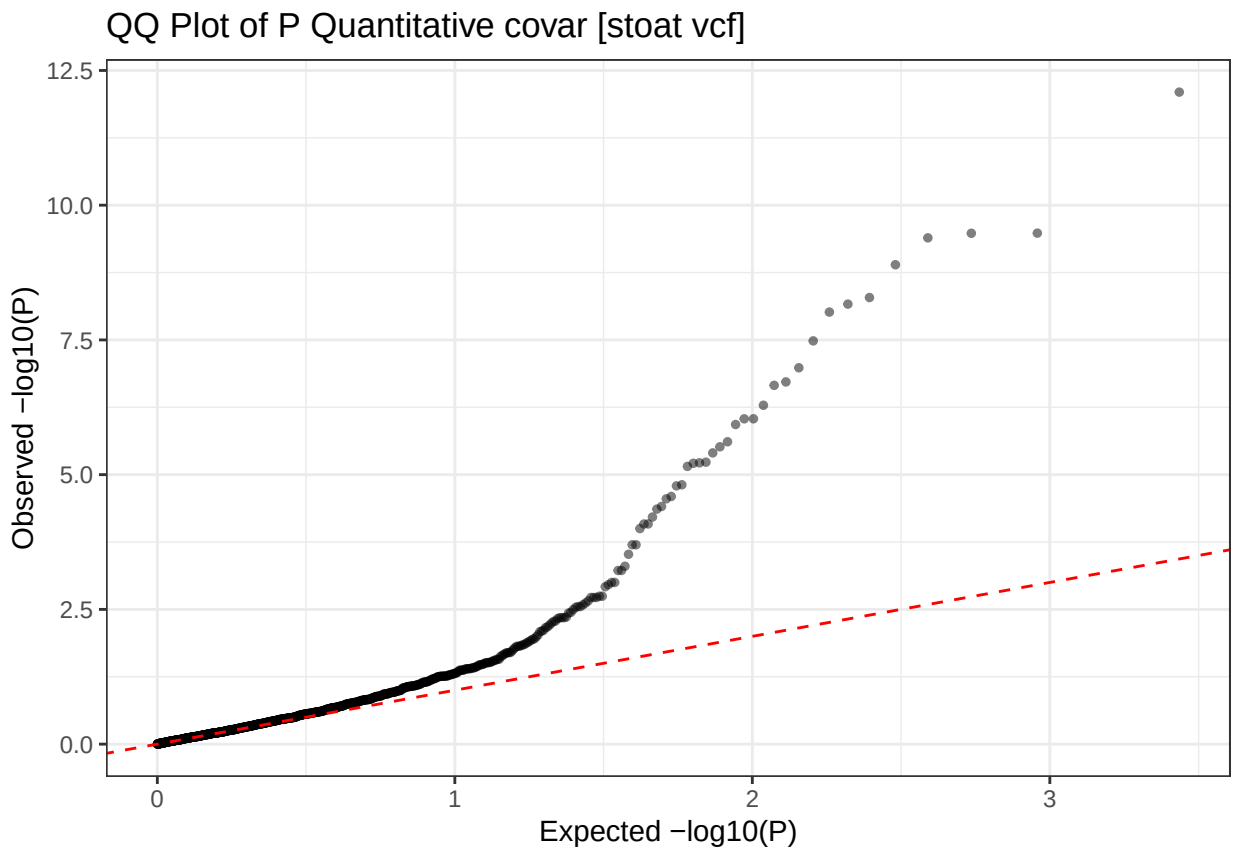
```
## Output directory already exists: ../quantitative_covar_output
##
## Metrics for p-value < 0.01:
## Confusion Matrix for p-value [P] < 0.01:
## [[  63    9]
##  [  84 1206]]
```

```
## Precision: 0.993
## Recall: 0.935
## F1 Score: 0.963
##
## Metrics for p-value < 1e-05:
## Confusion Matrix for p-value [P] < 1e-05:
## [[  23    0]
##  [ 124 1215]]
## Precision: 1.000
## Recall: 0.907
## F1 Score: 0.951
##
## Metrics for p-value < 1e-08:
## Confusion Matrix for p-value [P] < 1e-08:
## [[   8    0]
##  [ 139 1215]]
## Precision: 1.000
## Recall: 0.897
## F1 Score: 0.946
##
## Pourcentage of paths (present in freq file) tested :  90.86057371581055
```

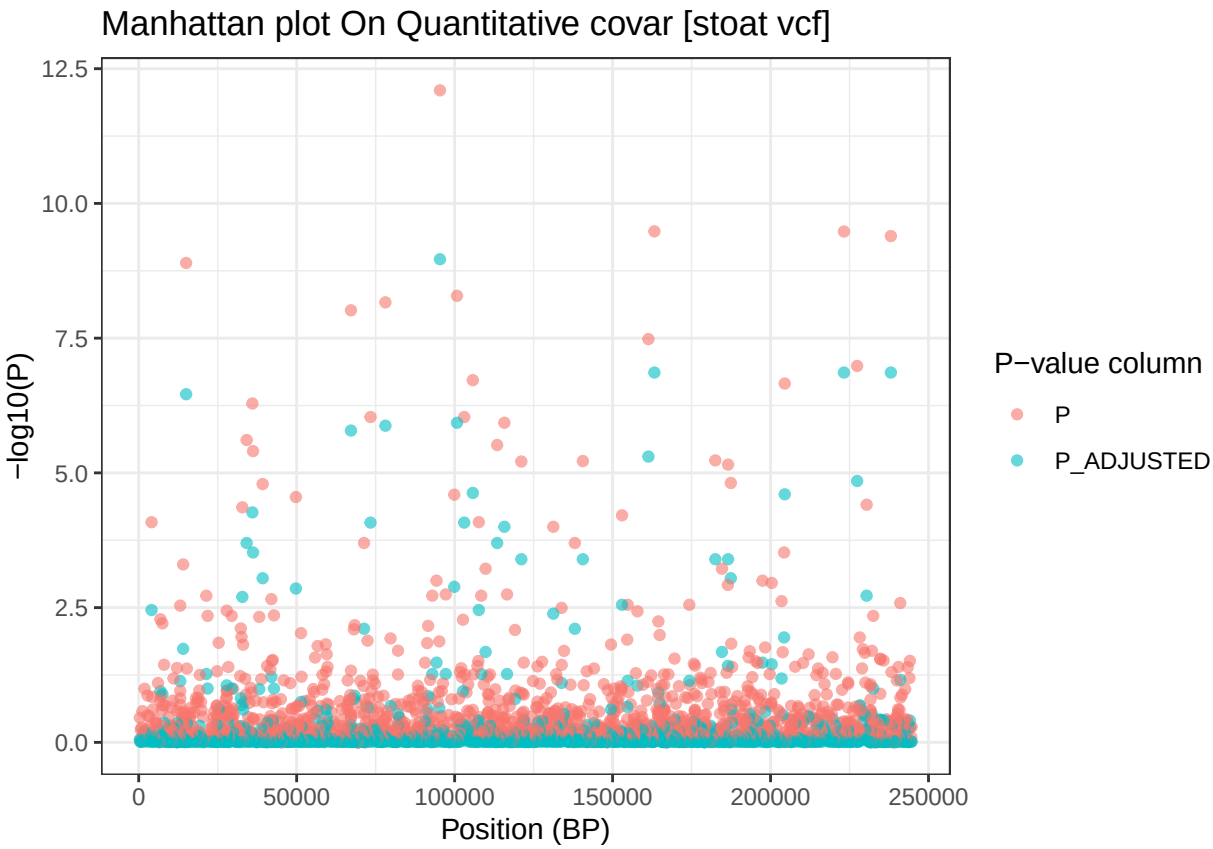
```
df_quantitative_covar_vcf <- read_stoat(file_list$quantitative_covar_vcf)

# ---- Plots ----
histogram_plot(df=df_quantitative_covar_vcf, subtitle="Quantitative covar [stoat vcf]", "P", "P_ADJUSTE
```



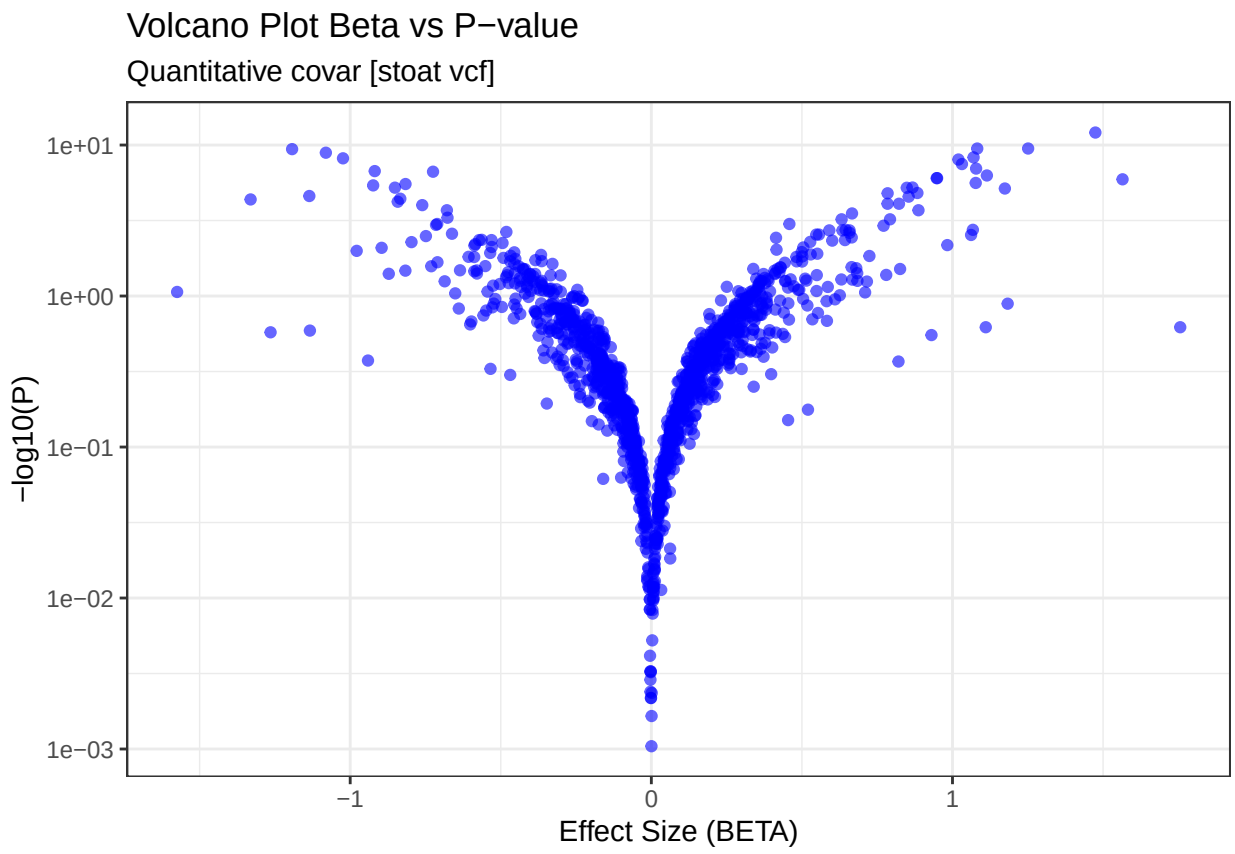
```
qq_plot(df_quantitative_covar_vcf, "P", subtitle="Quantitative covar [stoat vcf]")
```



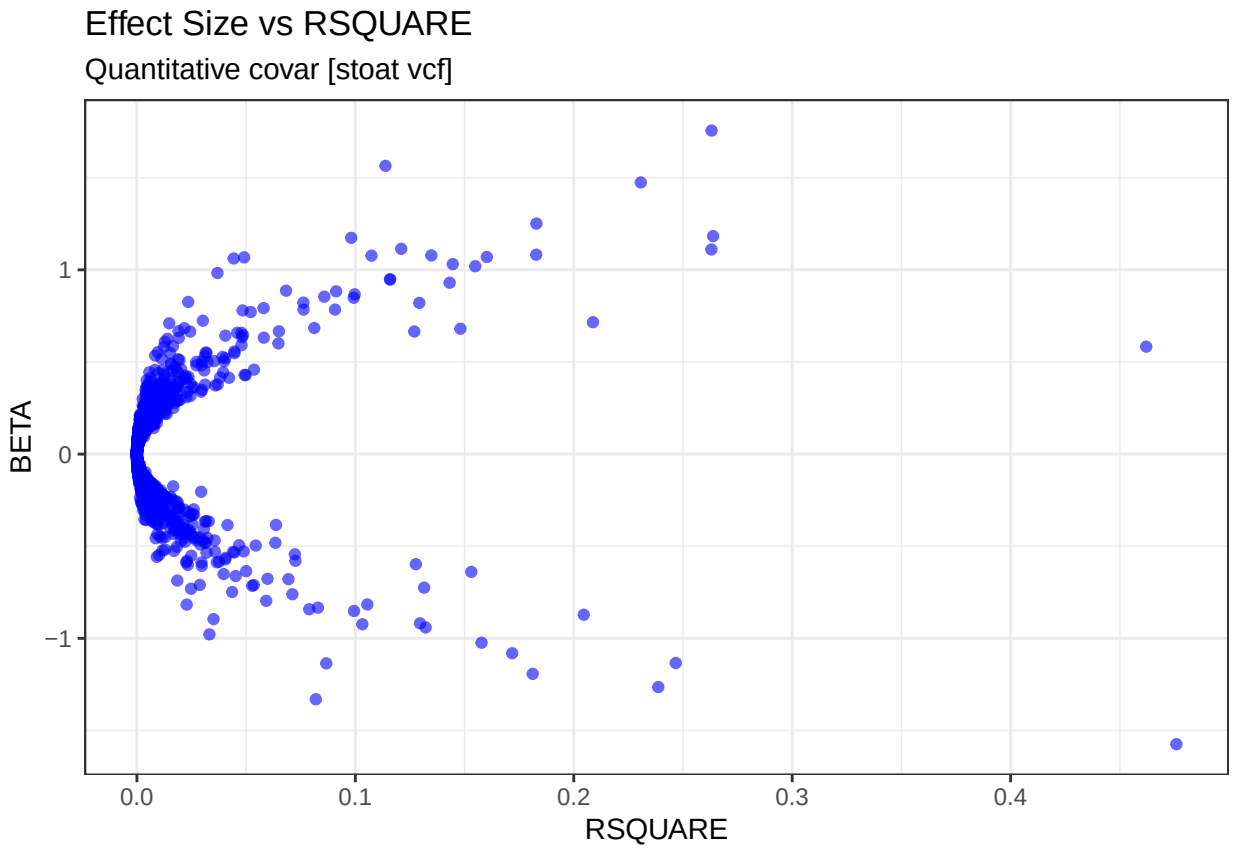

```
manhattan_plot(df_quantitative_covar_vcf, subtitle="On Quantitative covar [stoat vcf]", "P", "P_ADJUSTE
```

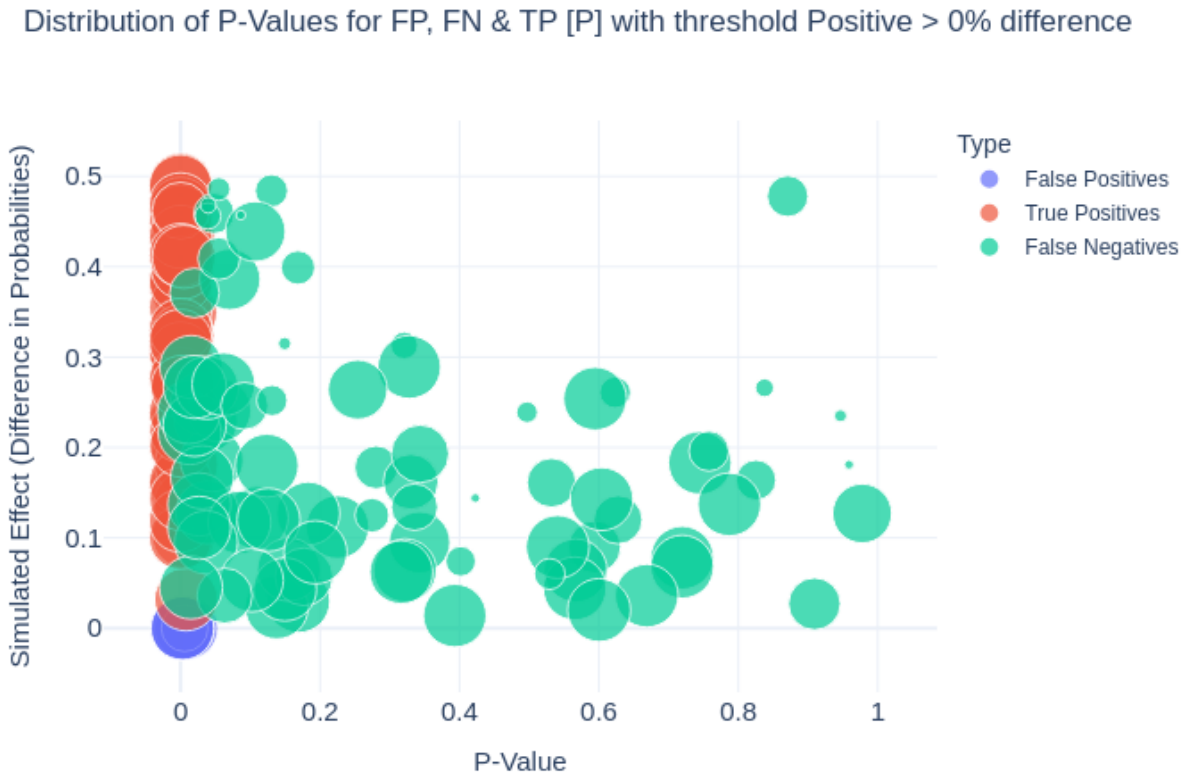


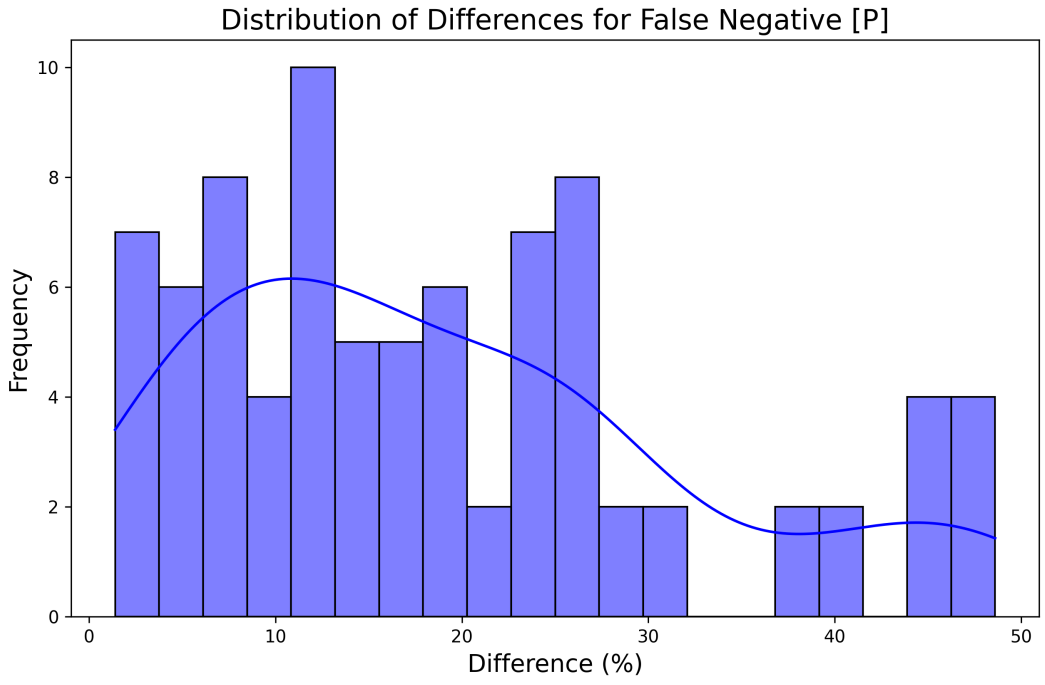

```
volcano_plot(df_quantitative_covar_vcf, subtitle="Quantitative covar [stoat vcf]")
```

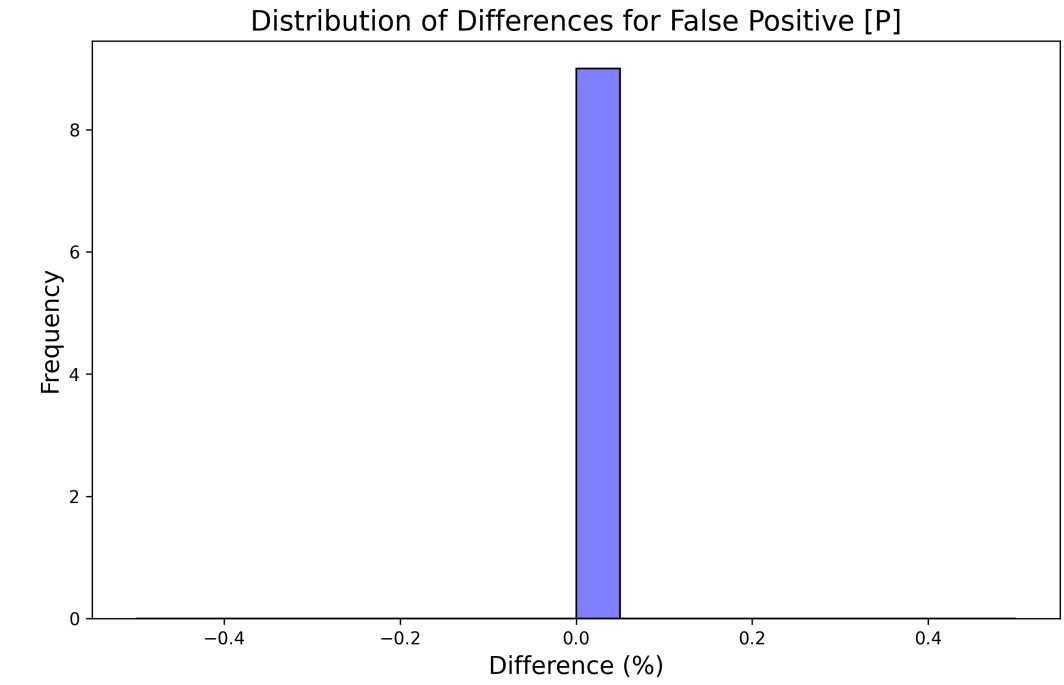


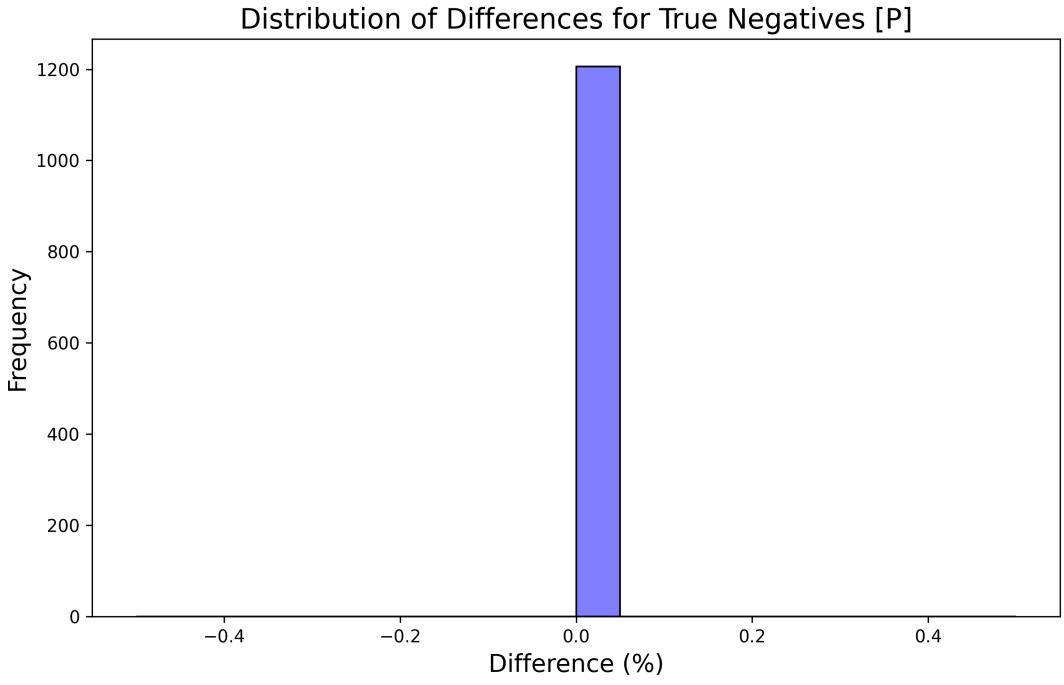

```
rsq_plot(df_quantitative_covar_vcf, subtitle="Quantitative covar [stoat vcf]")
```

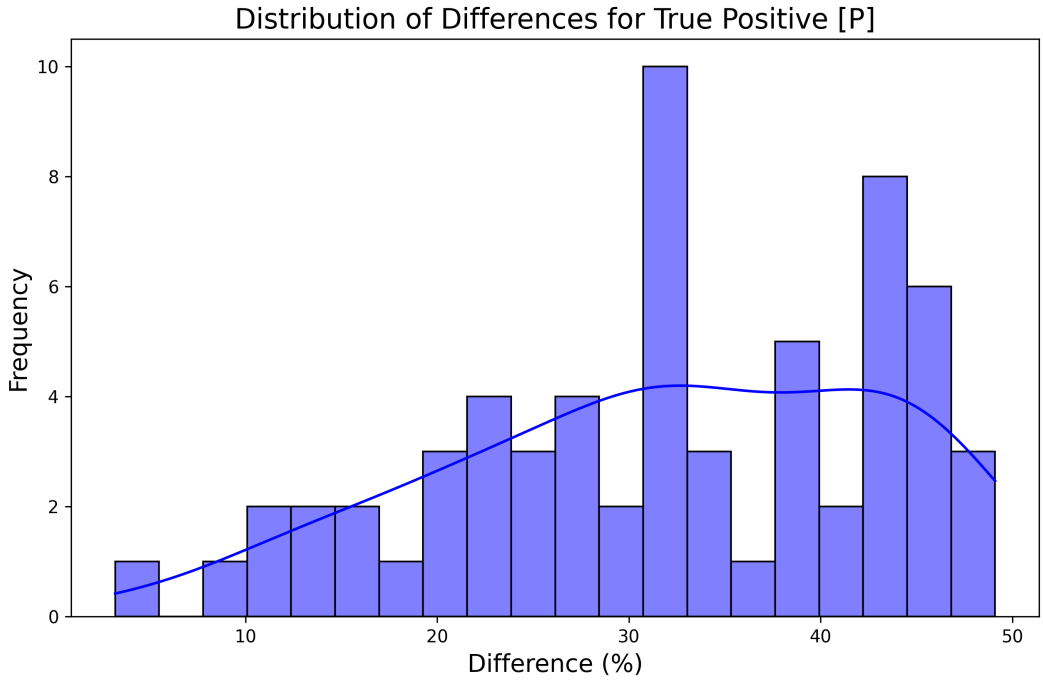

Effect Size vs RSQUARE
Quantitative covar [stoat vcf]
BETA
1
0
−1
0.0
0.1
0.2
0.3
0.4
RSQUARE

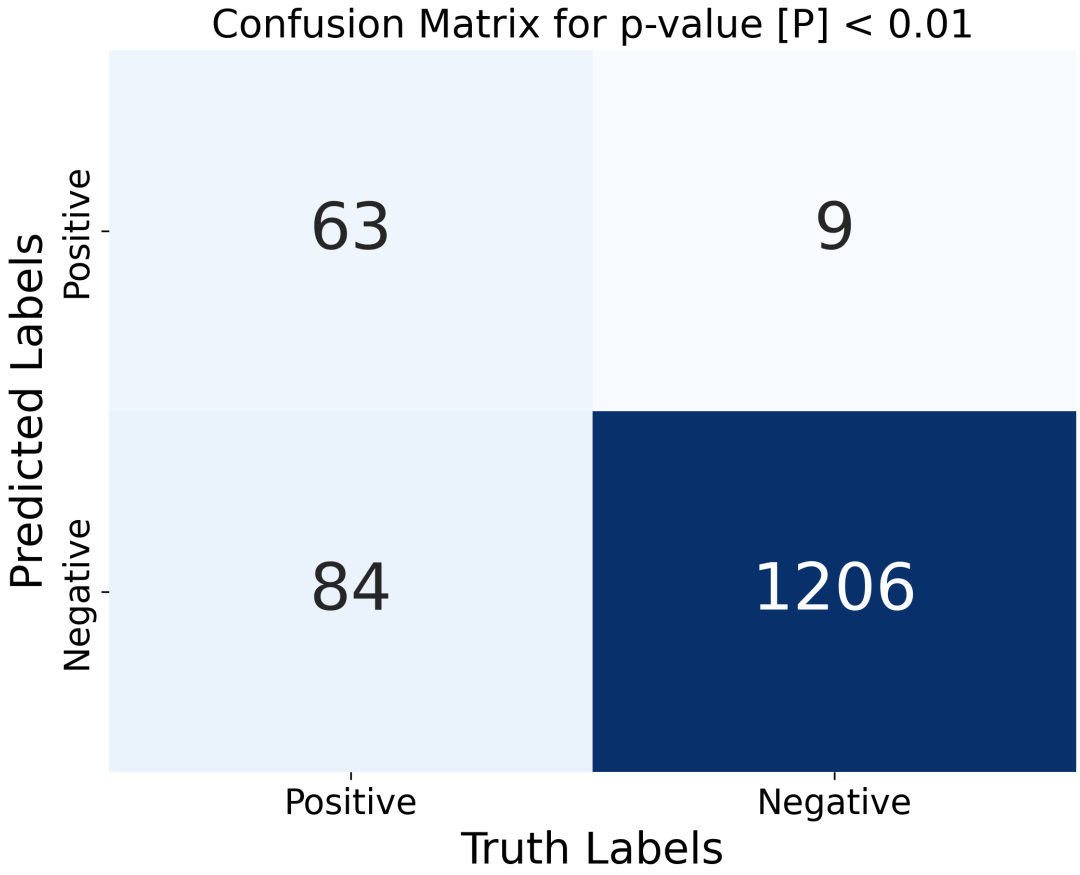

Distribution of P-Values for FP, FN & TP [P] with threshold Positive > 0% difference
Simulated Effect (Difference in Probabilities)
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1
P-Value
Type
False Positives
True Positives
False Negatives

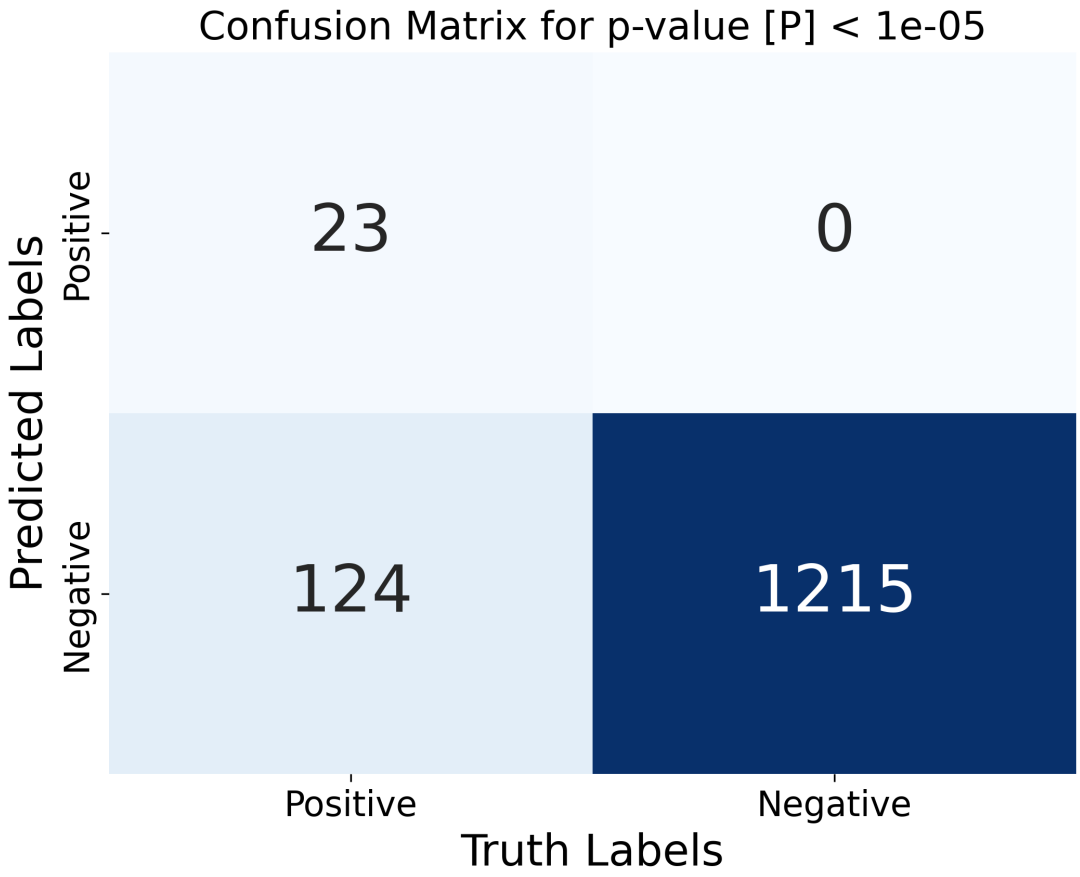

Distribution of Differences for False Negative [P]
Frequency
10
8
6
4
2
0
0
10
20
30
40
50
Difference (%)

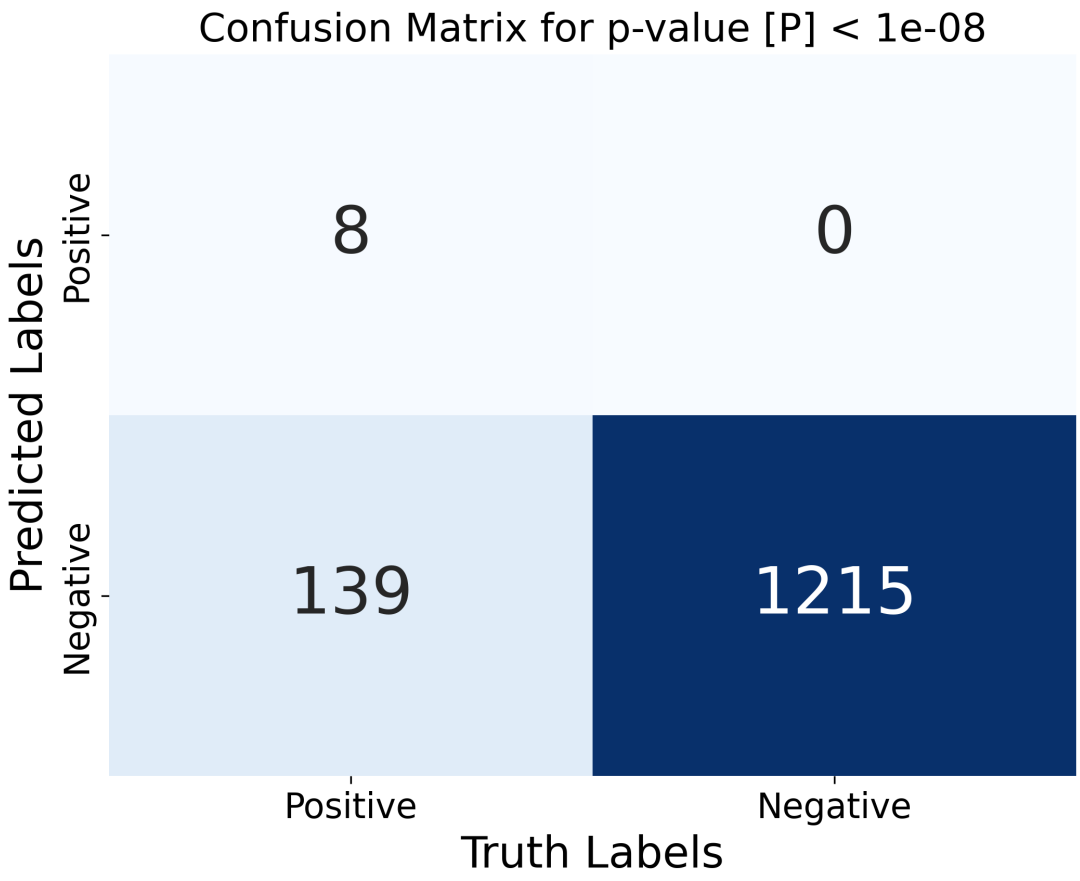

Distribution of Differences for False Positive [P]
Frequency
Difference (%)
0
2
4
6
8
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Negatives [P]
Frequency
Difference (%)
0
200
400
600
800
1000
1200
−0.4
−0.2
0.0
0.2
0.4


Distribution of Differences for True Positive [P]
Frequency
Difference (%)
0
2
4
6
8
10
10
20
30
40
50


Confusion Matrix for p-value [P] < 0.01
Predicted Labels
Positive
Negative
63
9
84
1206
Positive
Negative
Truth Labels

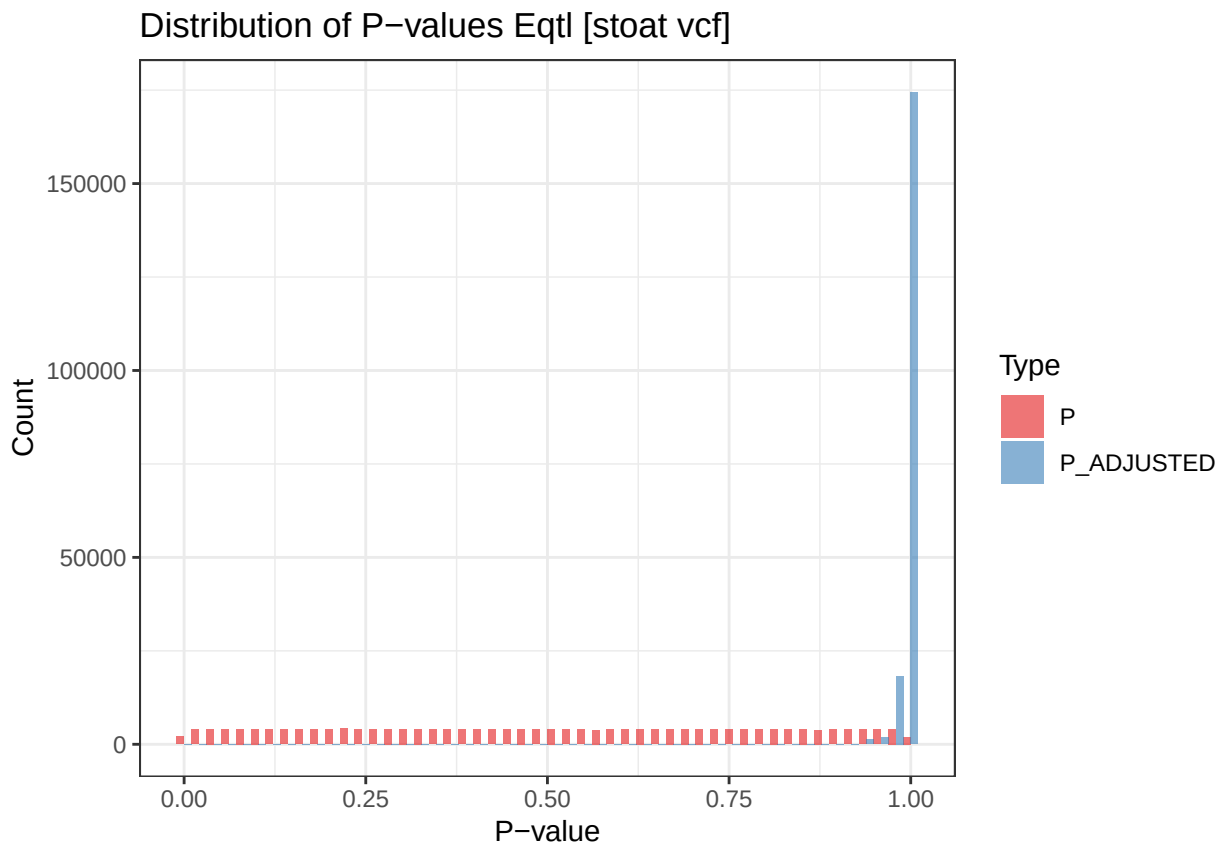

Confusion Matrix for p-value [P] < 1e-05
Predicted Labels
Positive
Negative
23
0
124
1215
Positive
Negative
Truth Labels



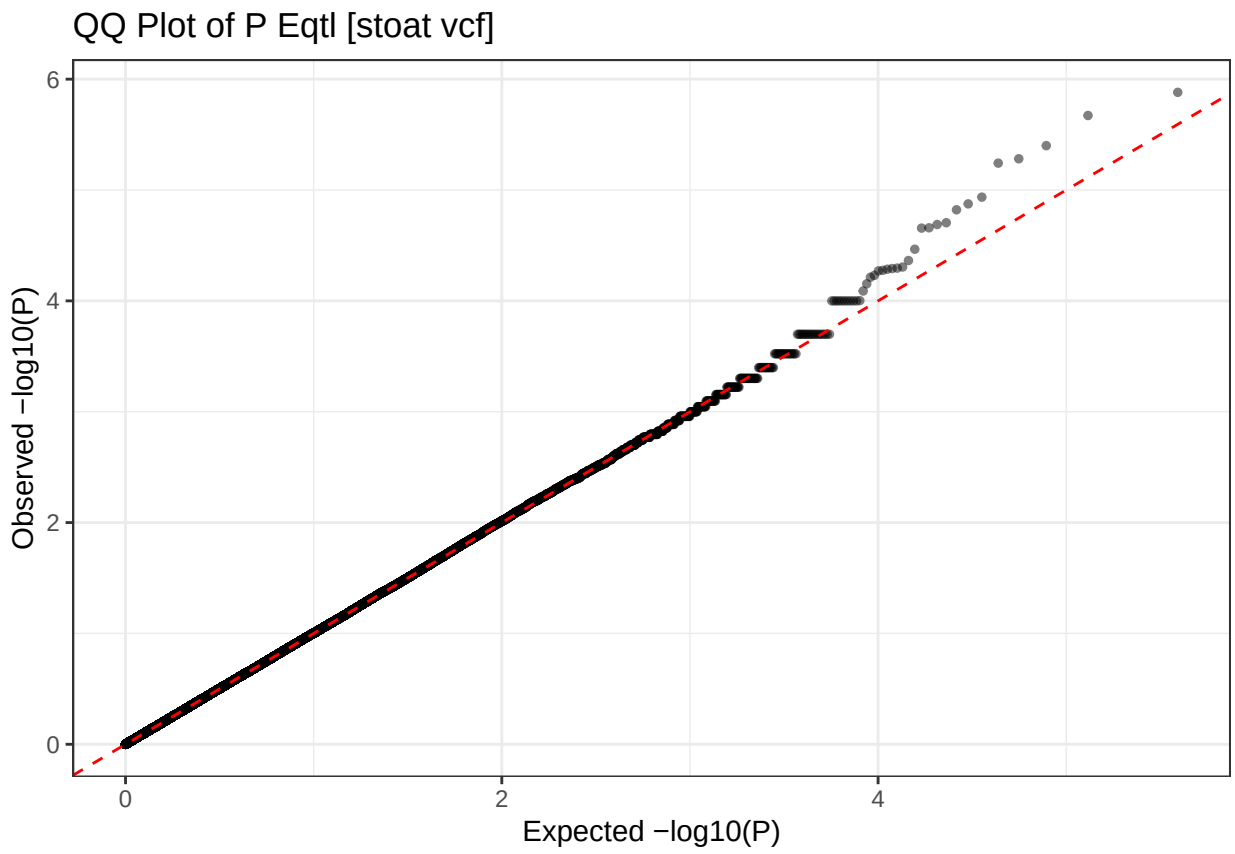
---

eQTL stoat VCF

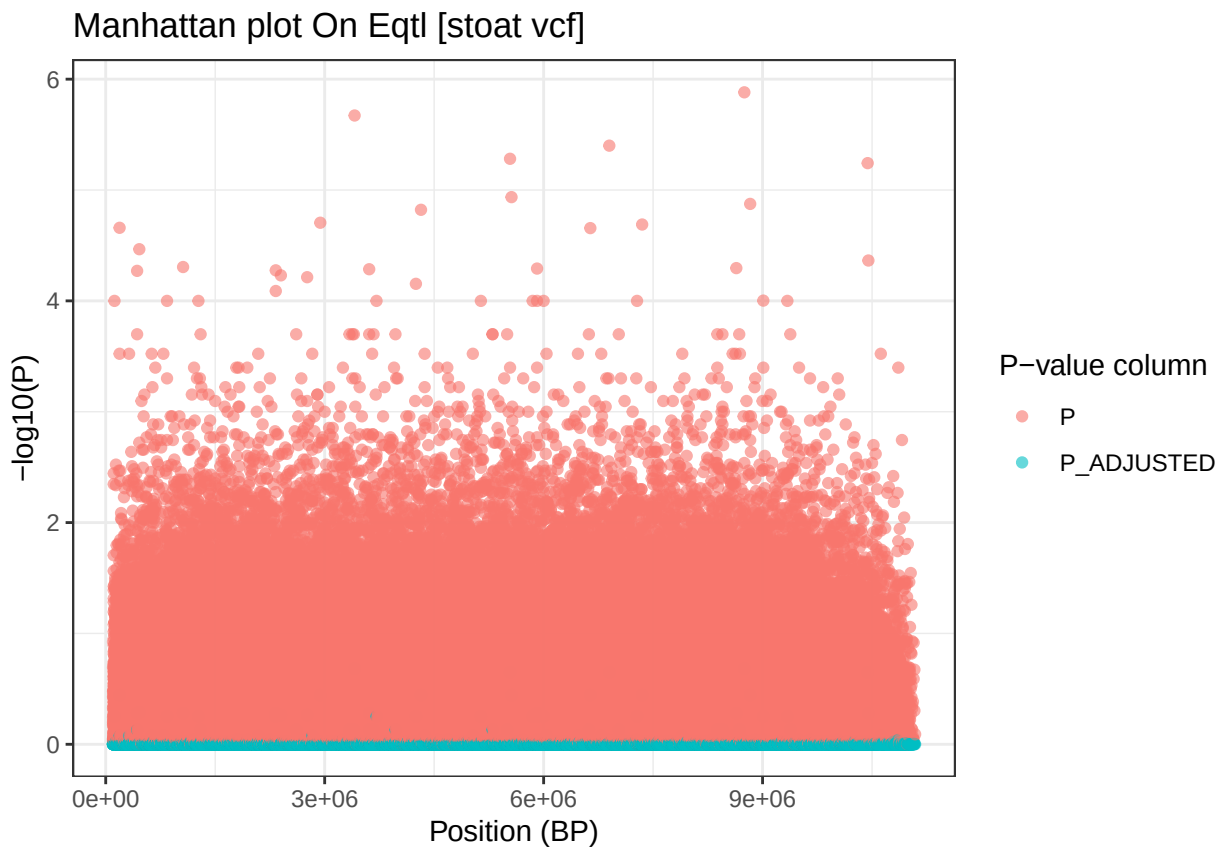
```
df_eqtl_vcf <- read_stoat(file_list$eqtl_vcf)

# ---- Plots ----
histogram_plot(df=df_eqtl_vcf, subtitle="Eqtl [stoat vcf]", "P", "P_ADJUSTED")
```




```
qq_plot(df_eqtl_vcf, "P", subtitle="Eqtl [stoat vcf]")
```

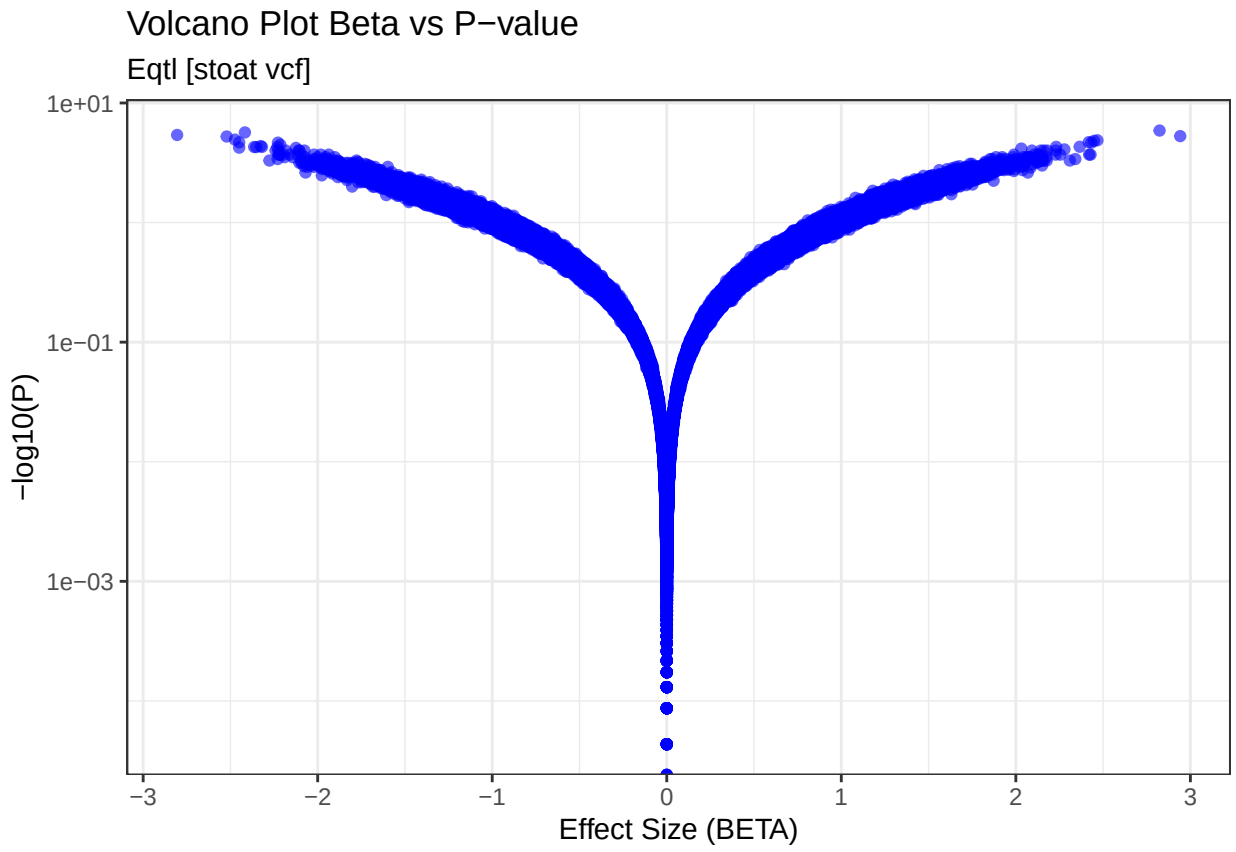



```
manhattan_plot(df_eqtl_vcf, subtitle="On Eqtl [stoat vcf]", "P_ADJUSTED", "P")
```

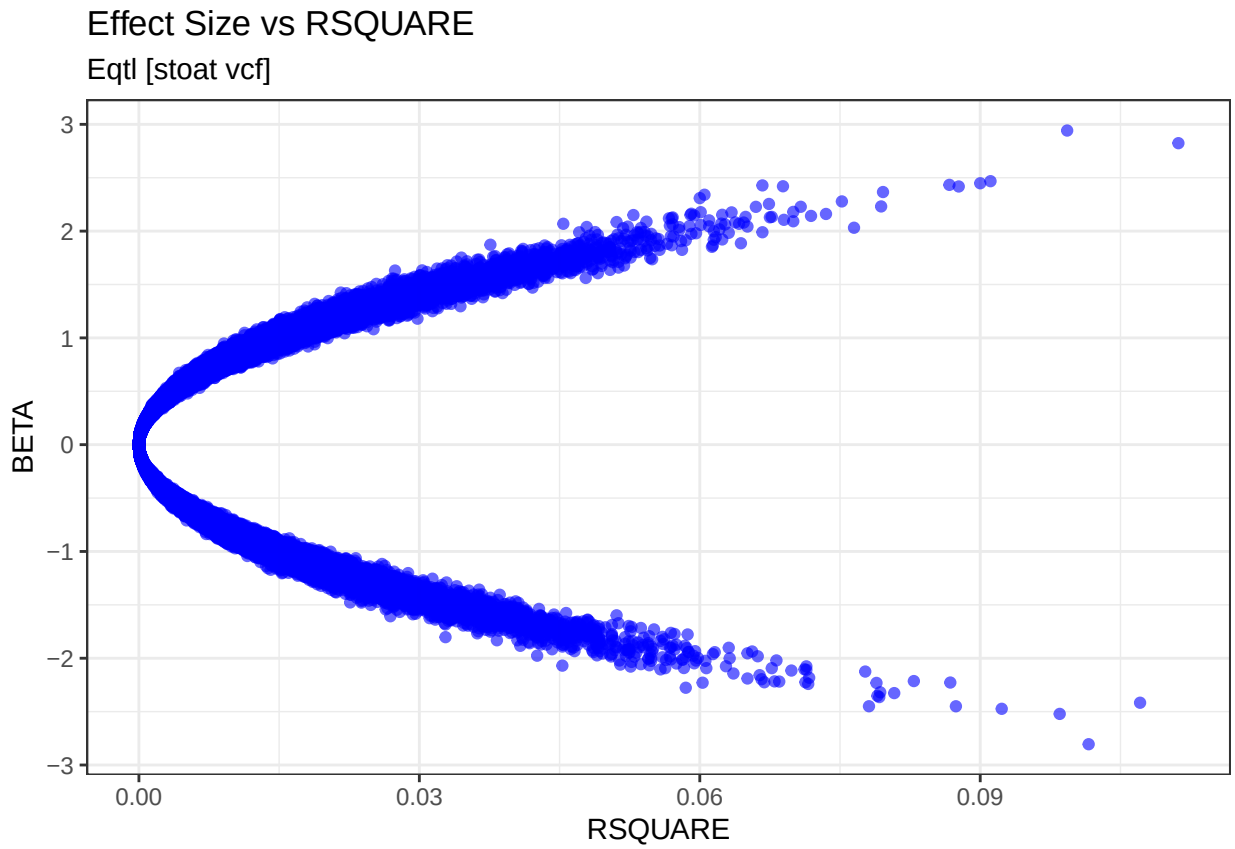



```
volcano_plot(df_eqtl_vcf, subtitle="Eqtl [stoat vcf]")
```

```
## Warning in scale_y_continuous(trans = "log10"): log-10 transformation
## introduced infinite values.
```



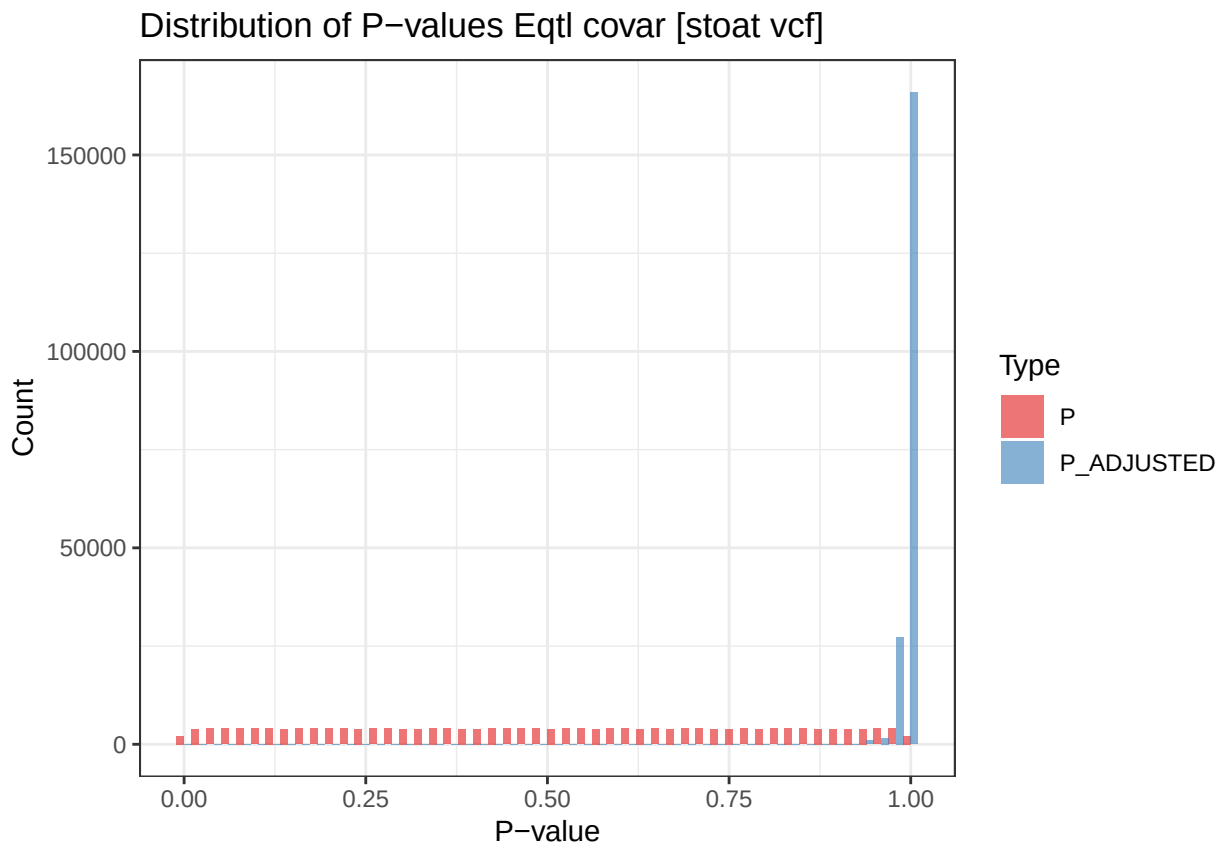

```
rsq_plot(df_eqtl_vcf, subtitle="Eqtl [stoat vcf]")
```



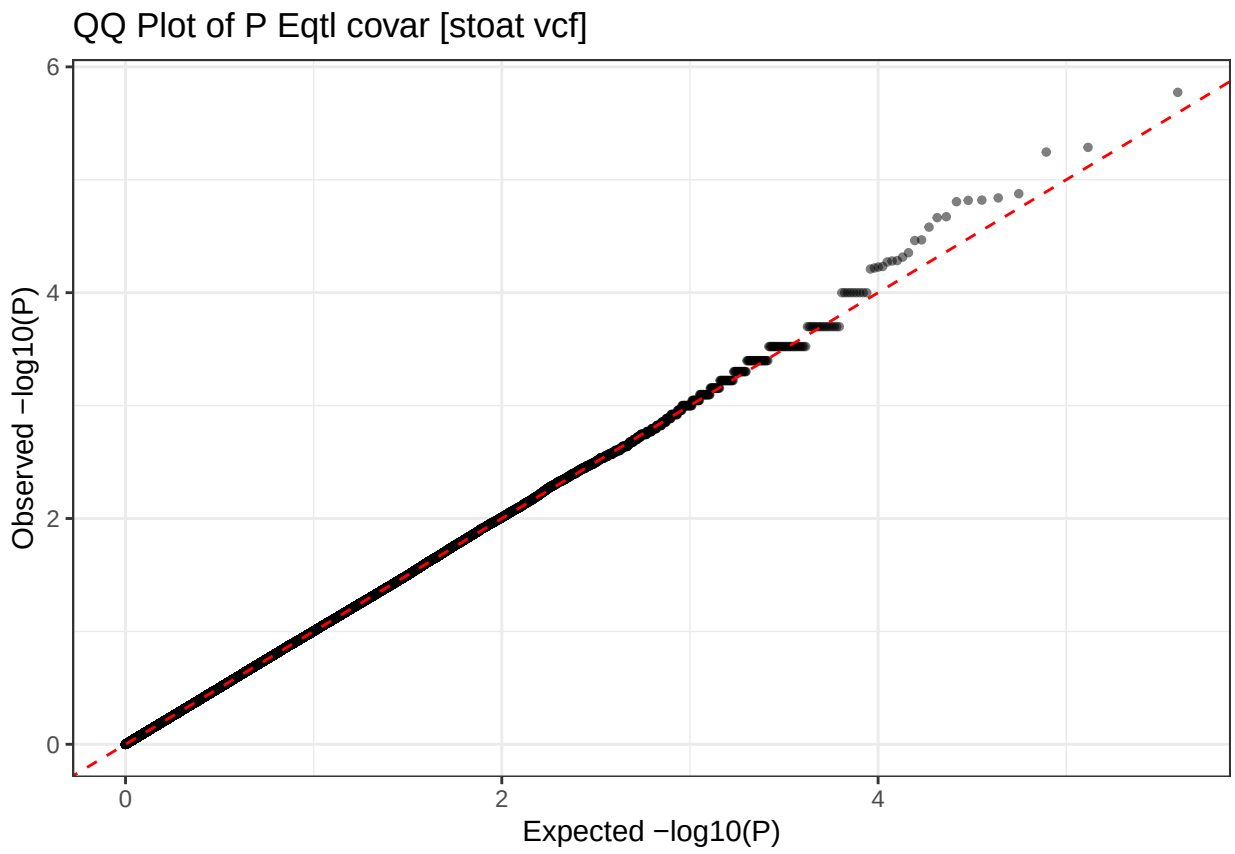
---

eQTL covar stoat VCF

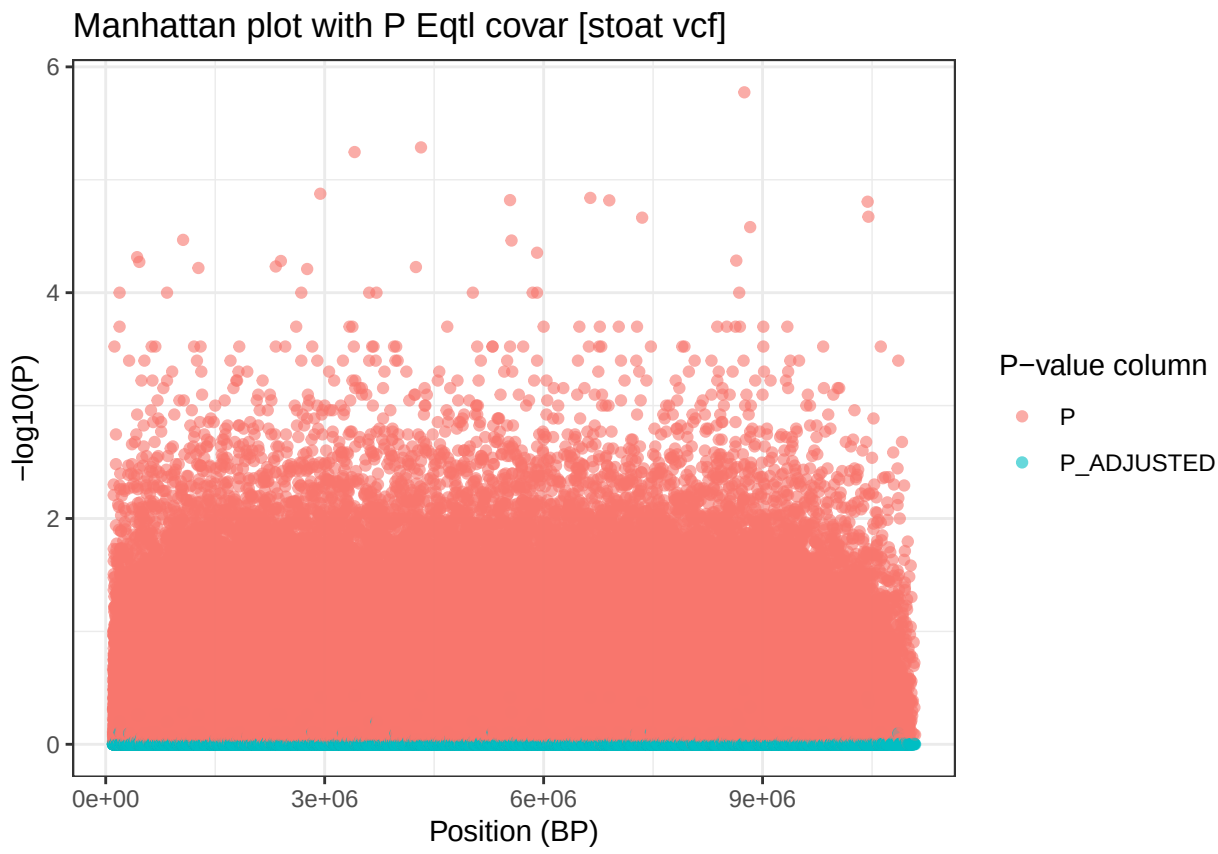
```
df_eqtl_covar_vcf <- read_stoat(file_list$eqtl_covar_vcf)

# ---- Plots ----
histogram_plot(df=df_eqtl_covar_vcf, subtitle="Eqtl covar [stoat vcf]", "P", "P_ADJUSTED")
```




```
qq_plot(df_eqtl_covar_vcf, "P", subtitle="Eqtl covar [stoat vcf]")
```

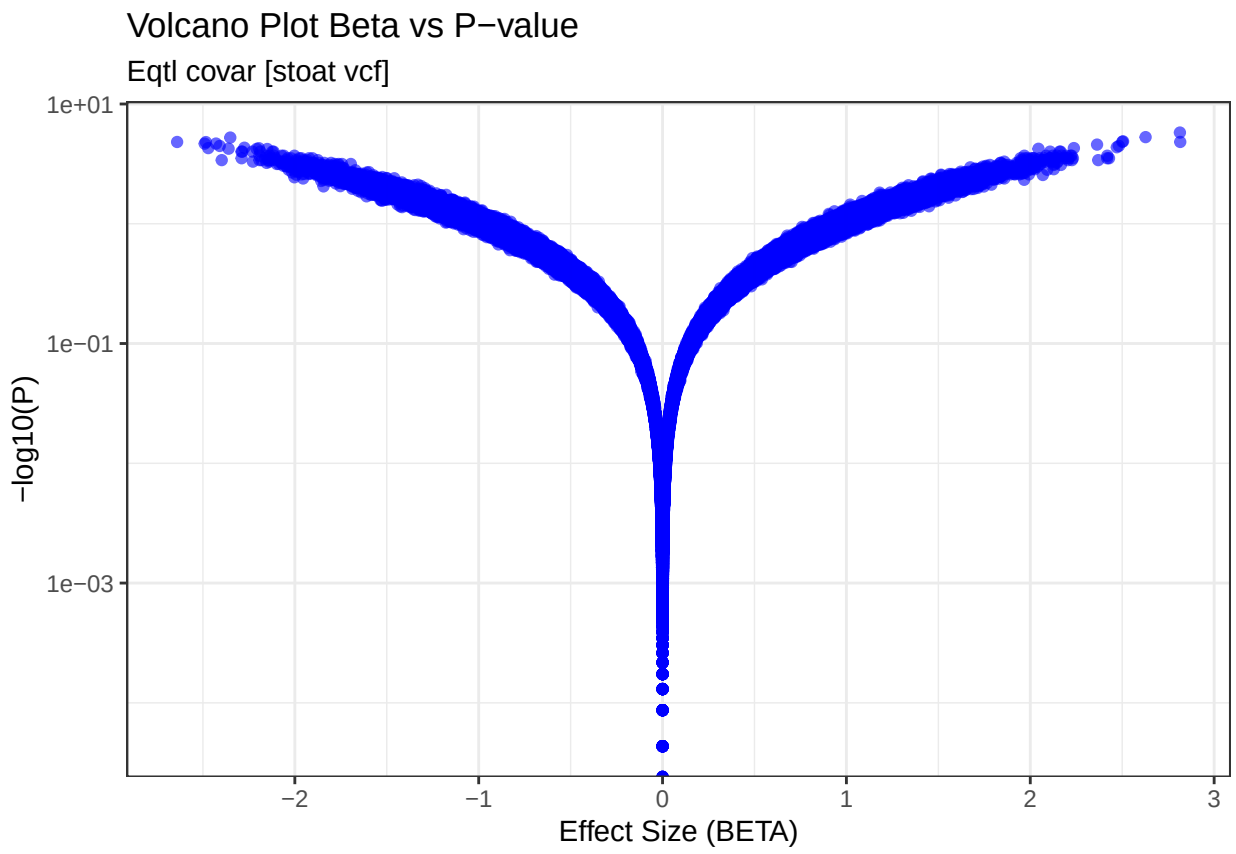



```
manhattan_plot(df_eqtl_covar_vcf, subtitle="with P Eqtl covar [stoat vcf]", "P", "P_ADJUSTED")
```

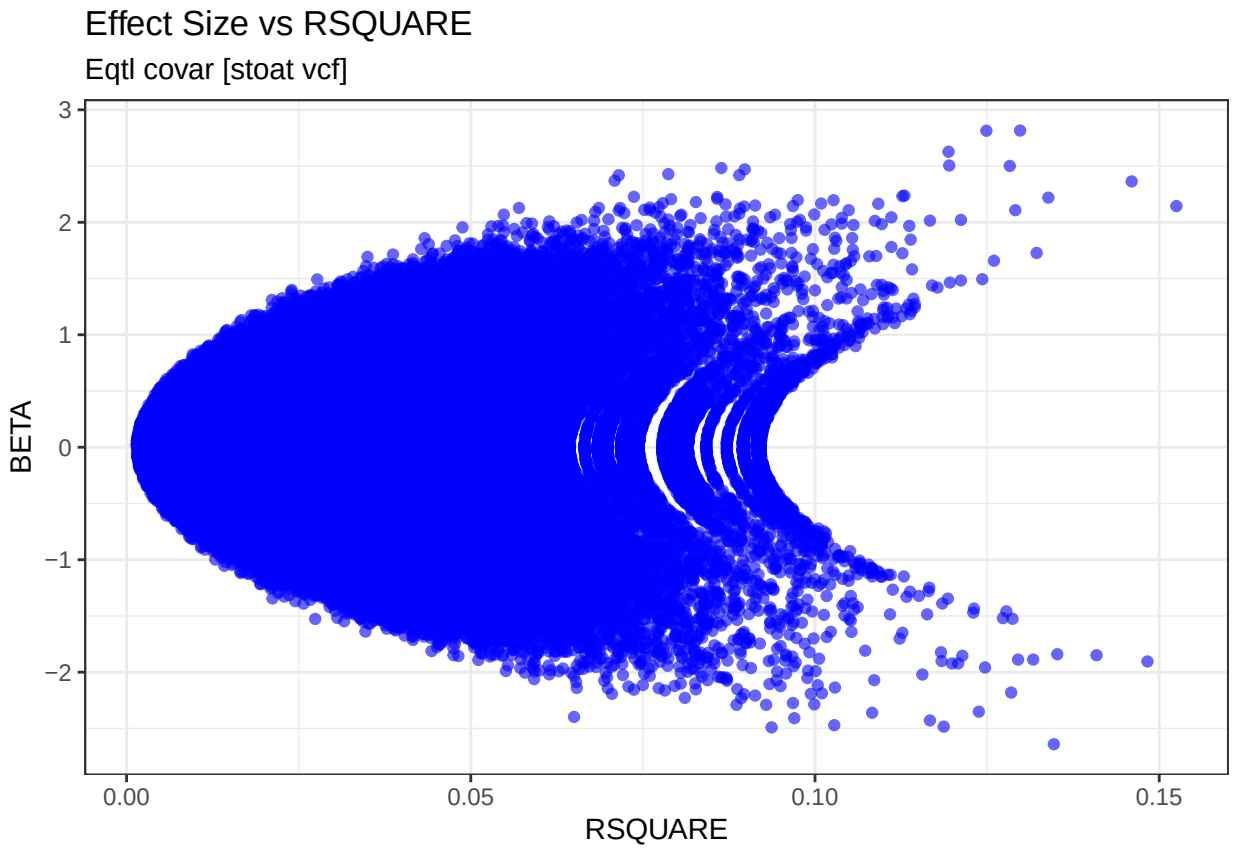



```
volcano_plot(df_eqtl_covar_vcf, subtitle="Eqtl covar [stoat vcf]")
```

```
## Warning in scale_y_continuous(trans = "log10"): log-10 transformation
## introduced infinite values.
```




```
rsq_plot(df_eqtl_covar_vcf, subtitle="Eqtl covar [stoat vcf]")
```


Effect Size vs RSQUARE
Eqtl covar [stoat vcf]
BETA
RSQUARE
3
2
1
0
−1
−2
0.00
0.05
0.10
0.15