Dark Kinases (KRSA Report)

Khaled Alganem

20 January, 2022

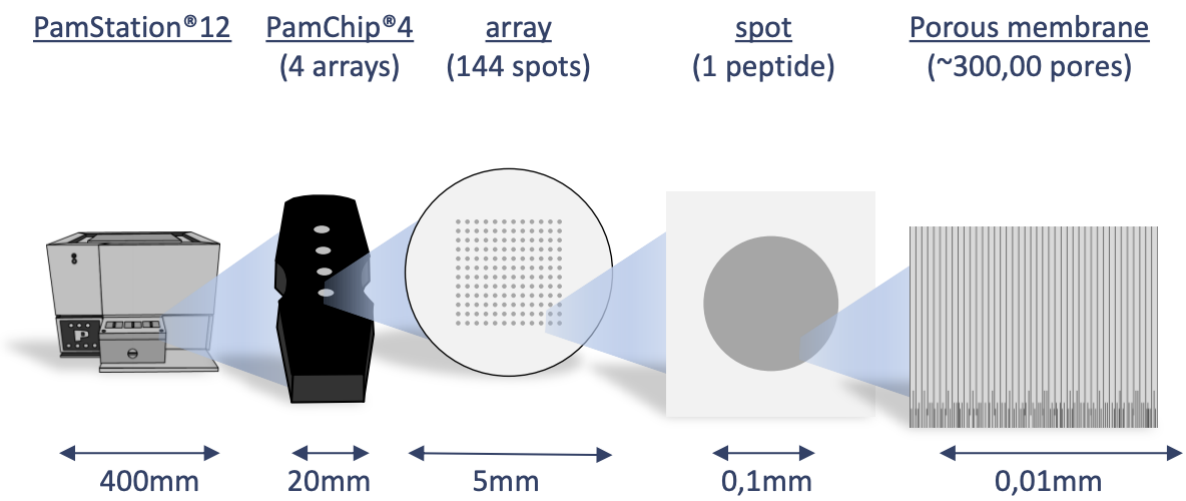
```
library(KRSA)
library(knitr)
library(tidyverse)

library(gt) # can be used to view tables
library(furrr) # can be used for parallel computing

theme_set(theme_bw())
```

Introduction

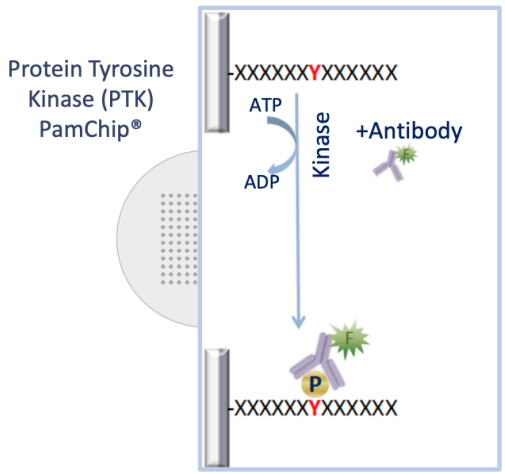
Background

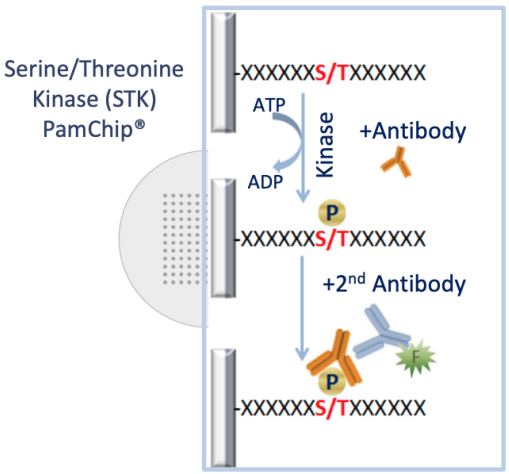
The Pamstation12 instrument provides a profiling of kinase activity of cell or tissue samples. The device is loaded with either serine/threonine or tyrosine microarray chips. Each chip has 4 wells so four samples can be loaded on a single chip, and the Pamstation12 can accommodate 3 chips per run. The microarray represents 144 (STK chip) or 196 (PTK chip) reporter peptides that can be phosphorylated by serine/threonine or tyrosine kinases. The device measures the degree of the phosphorylation in real time by detecting fluorescently labeled antibodies at different exposure times. The list of peptides present in each microarray can be viewed here: STK chip, PTK chip

```
knitr::include_graphics("pamgene_workflow.png")
```




```
knitr::include_graphics("pamgene_detectionFig.png")
```





Results

Image Analysis

The first step of analyzing the run is to convert the images taken by the PamStation of each array at different exposure times to numerical values This is done by the Bionavigator software developed by Pamgene. The software recognizes the grid of the array with the aid of the searching algorithm (Pamgrid) to correctly identify each spot on the array. The numbers produced by this software represent the median value of the foreground pixels minus the median value of the background pixels to produce the median signal minus background (Median_SigmBg).

Reading Data

The first step will be reading the crosstab view bionavigator files (Median_SigmBg and Signal_Saturation) and defining the PamChip type (STK or PTK). The raw data is read and then transformed to be in tidy format for an easier analysis, modeling, and visualizing.

```
# Define chip type. STK or PTK
chipType <- "PTK"

# Read crosstab view bionavigator files and tidy data
data <- krsa_read("input_data/_ExportstosDBstsCopy_Median_SigmBg_220118170521.txt", "input_data/_Exports
```

QC Initial Steps and Groups Assignments

We will perform a couple of quality control steps to deal with negative values in the data and adjust based on signal saturation (optional). Next, we will define a new column to represent the grouping. And then, we will extract end point signal values

```
# qc processing
krsa_qc_steps(data, sat_qc = T) -> data

# Define sample groups
data %>% mutate(Group = str_extract(SampleName, "[^_]+")) -> data

# extract end level signal values @ all max exposure time (200ms)

#krsa_extractEndPointMaxExp(data, chipType) -> data_pw_200

filter(data, Cycle == 94, ExposureTime == 100) %>%
  dplyr::select(SampleName, Peptide ,Signal, Group) -> data_pw_200

# extract end level signal values end level @ all exposure times
krsa_extractEndPoint(data, chipType) -> data_pw
```

QC Steps and Model Fitting

We will filter out peptides with low signals. In order to combine the values from different exposure times into a single value, a simple linear regression model of the *Medain_SigmBg* as a function of exposure time is fitted. The slope of of the model fit and R^2 are then used for quality control and samples comparison. The

slope is also scaled by multiplying by 100 and log2 transformed (*Slope_Transformed*). We then filter out peptides with poor linear fit and references peptides.

```
# Filter out peptides with low signals
krsa_filter_lowPeps(data_pw_200, 0) -> ppPassAll


# Fit the linear model
# This will produce a list of data frames:
# scaled: the Slope_Transformed values (see above for more info on Slope_Transformed)
# normalized: the Slope_Transformed values but normalized by Chip/Barcode
# Grouped: The mean of Slope_Transformed values for all samples within a group
krsa_scaleModel(data_pw, ppPassAll) -> data_modeled

# Filter out peptides weak linear fit
krsa_filter_nonLinear(data_modeled$scaled, 0) -> ppPassR2

# Filter out reference peptides
krsa_filter_ref_pep(ppPassR2) -> new_pep


data_modeled$scaled <- data_modeled$scaled %>%
             mutate(SampleName = factor(SampleName,
                                        levels = c("INSRR_250NG_HI","TNK1_250NG_HI", "EPHA6_250NG_HI"
                                                   "INSRR_2.5NG","TNK1_2.5NG", "EPHA6_2.5NG",
                                                    "INSRR_25NG","TNK1_25NG", "EPHA6_25NG",
                                                   "INSRR_250NG","TNK1_250NG", "EPHA6_250NG"))
                    )
```

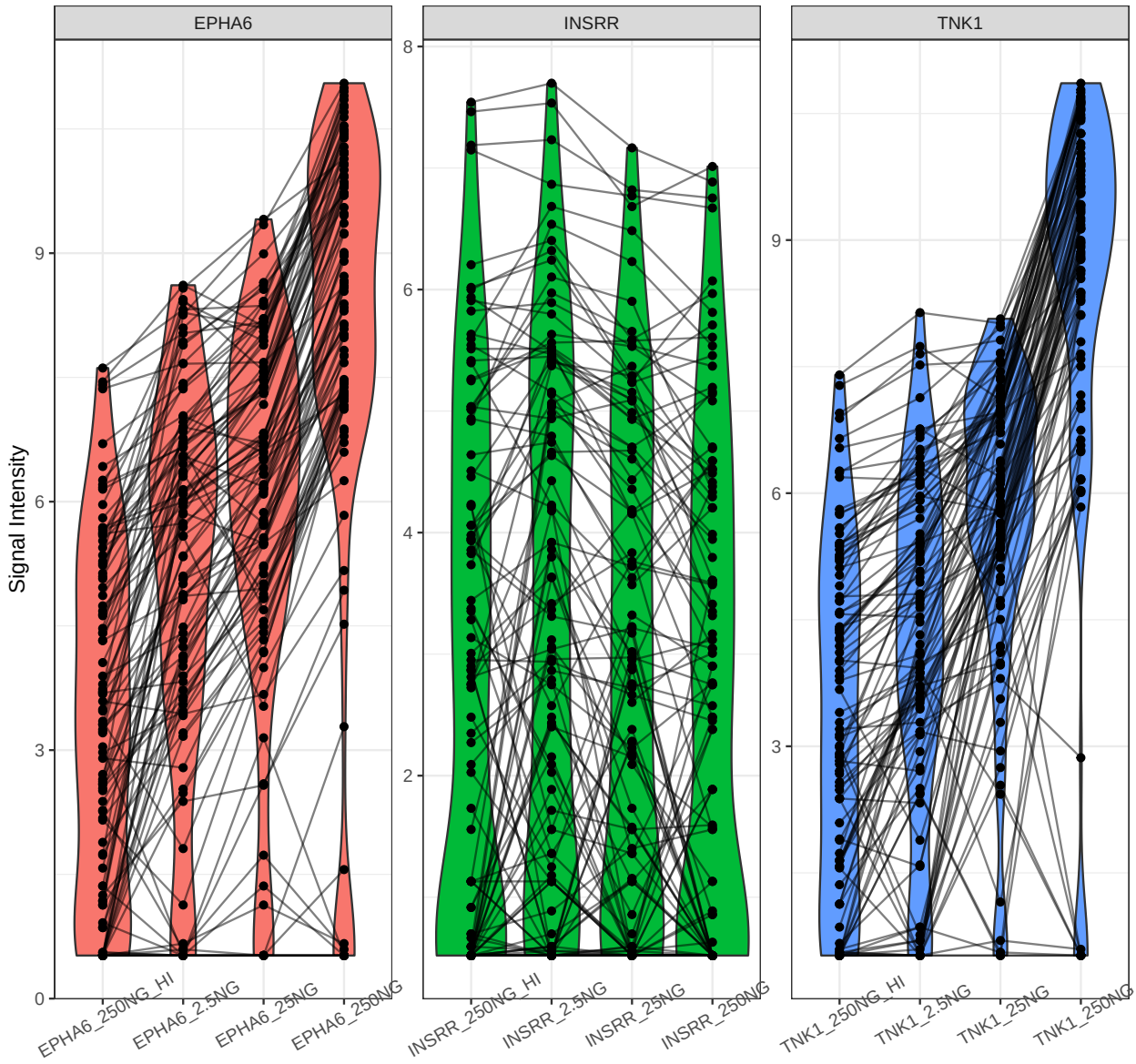
Global Signal Intensity

For a global signal intensity across all samples/groups, few figures can be plotted based on the *Slope_Transformed* values.

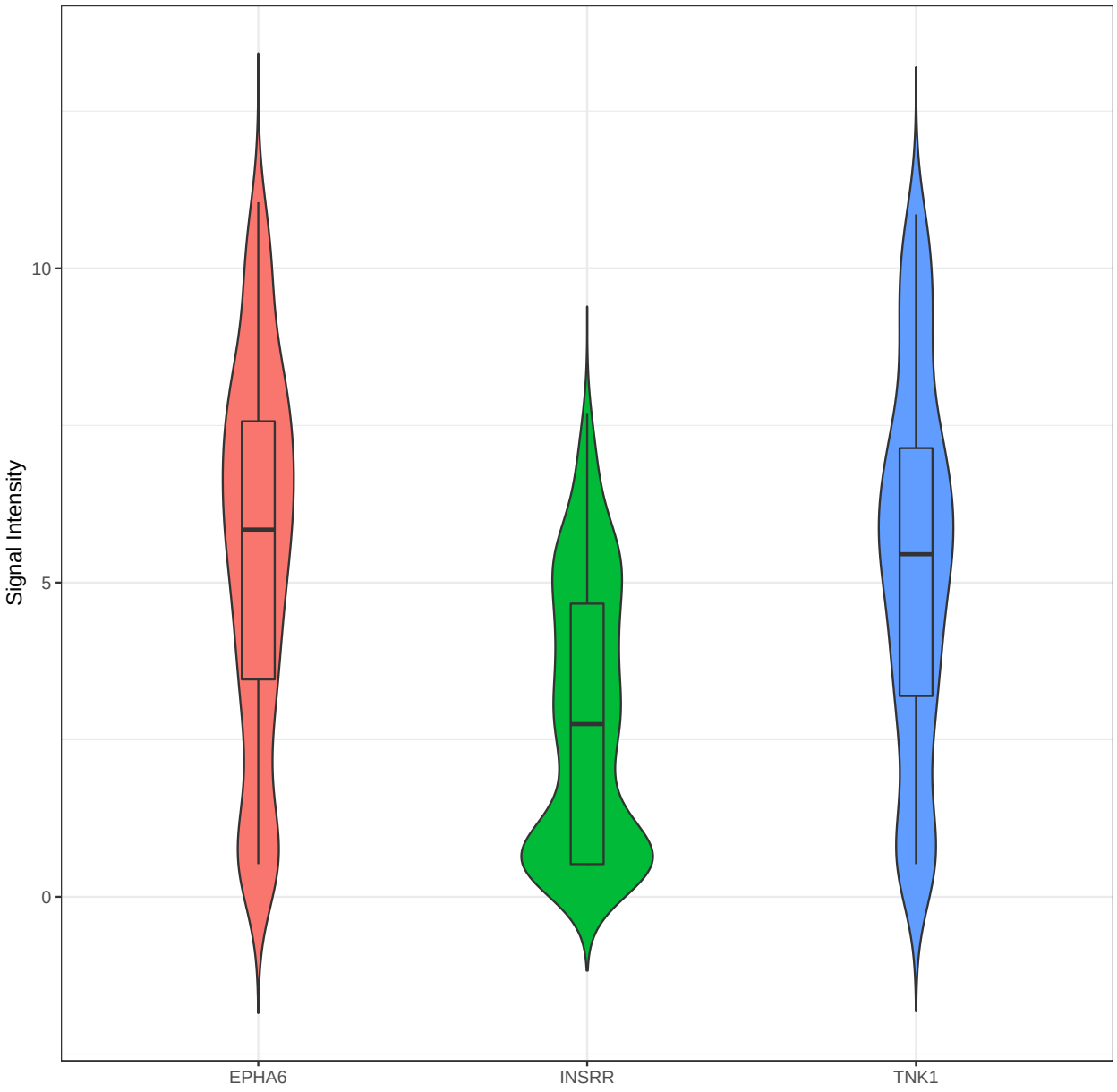
Global Violin Plots

We will plot violin figures to examine global signal differences between groups/samples.

```
# Plot a violin figure and facet by the (Group) variable
krsa_violin_plot(data_modeled$scaled, new_pep, "Group") +
  theme(axis.text.x = element_text(angle = 30))
```



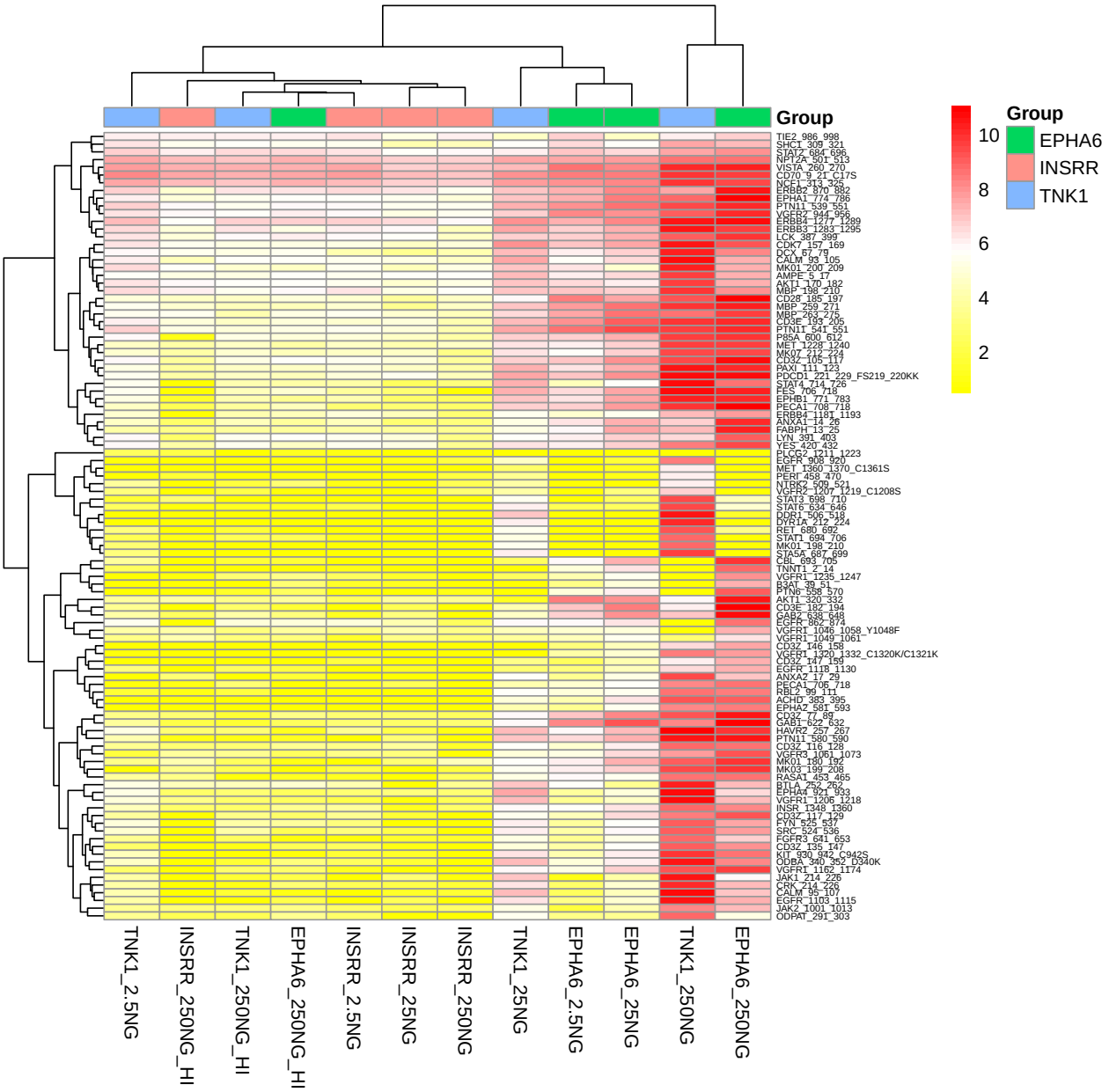

```
# Plot a grouped violin figure
krsa_violin_plot_grouped(data_modeled$scaled, new_pep, avg_line = F)
```



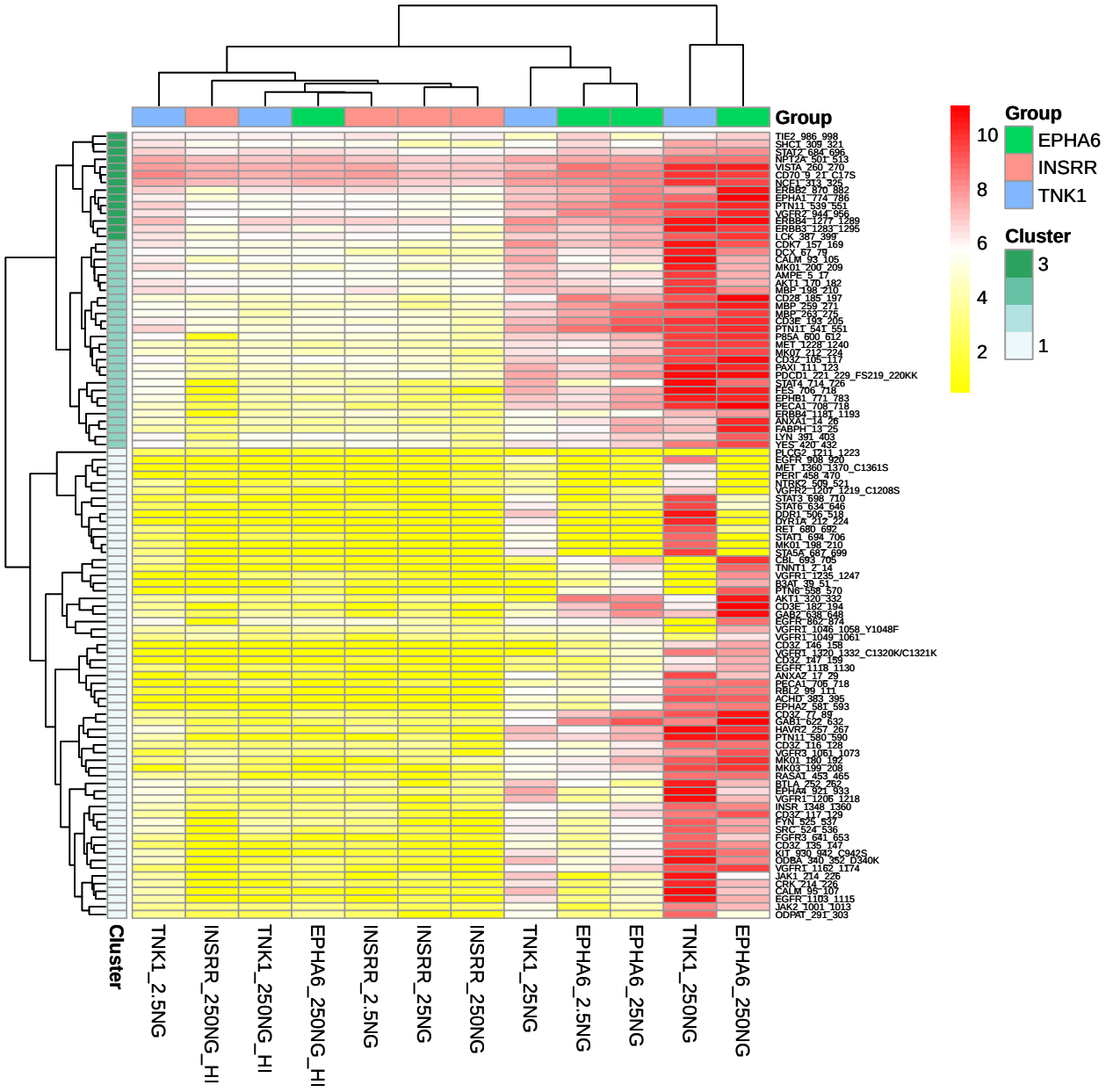
Global Heatmaps

The heatmap represent all the peptides present on the chip except the positive/internal controls and peptides that failed to pass QC. The heatmaps are scaled by row to highlight the peptide signal differences across the samples. A hierarchical unsupervised clustering is applied both on the peptides and the samples to potentially group similar signatures.

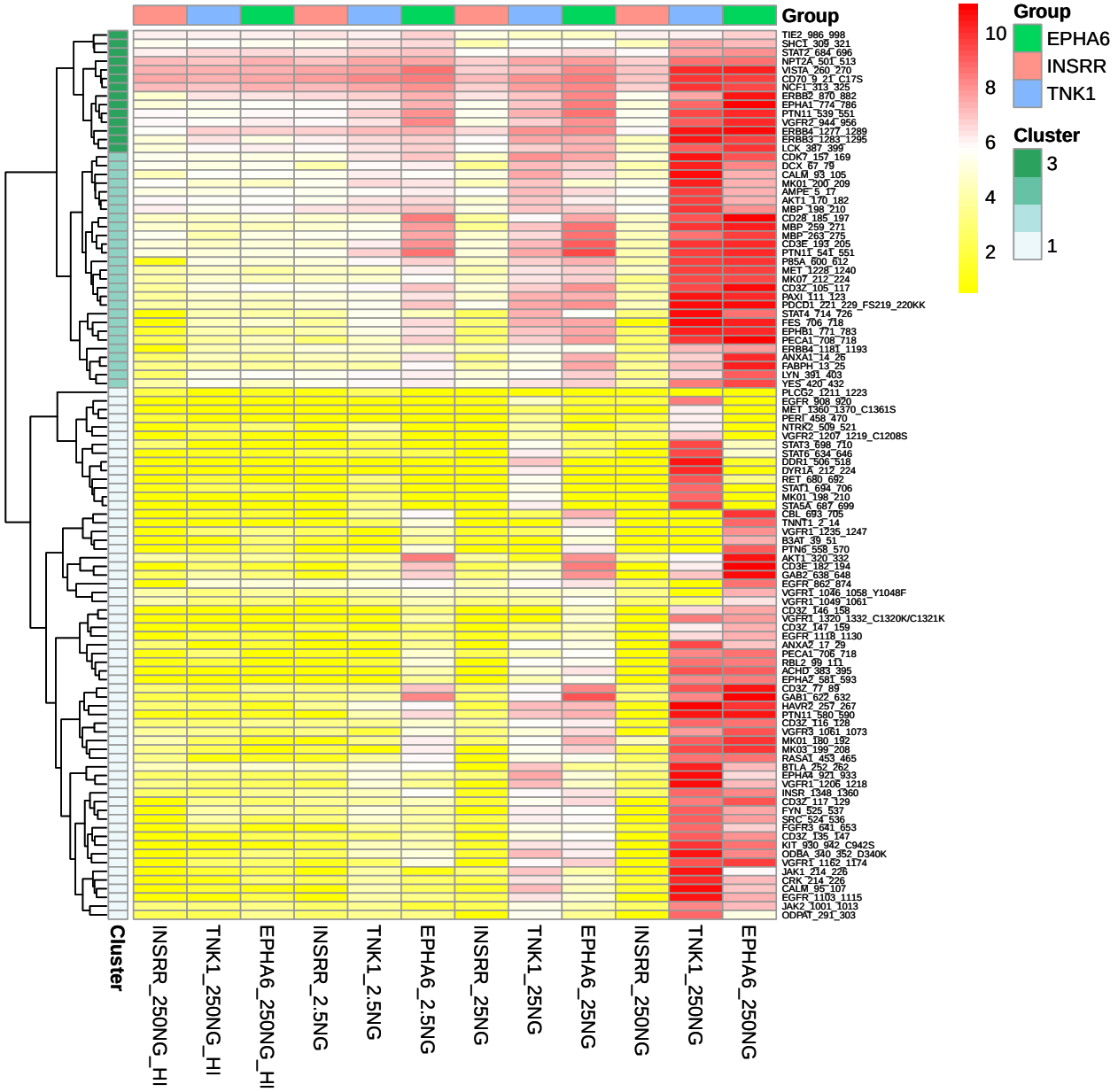
```
krsa_heatmap(data_modeled$scaled ,new_pep, scale = "none", cluster_col = T) -> p1
```



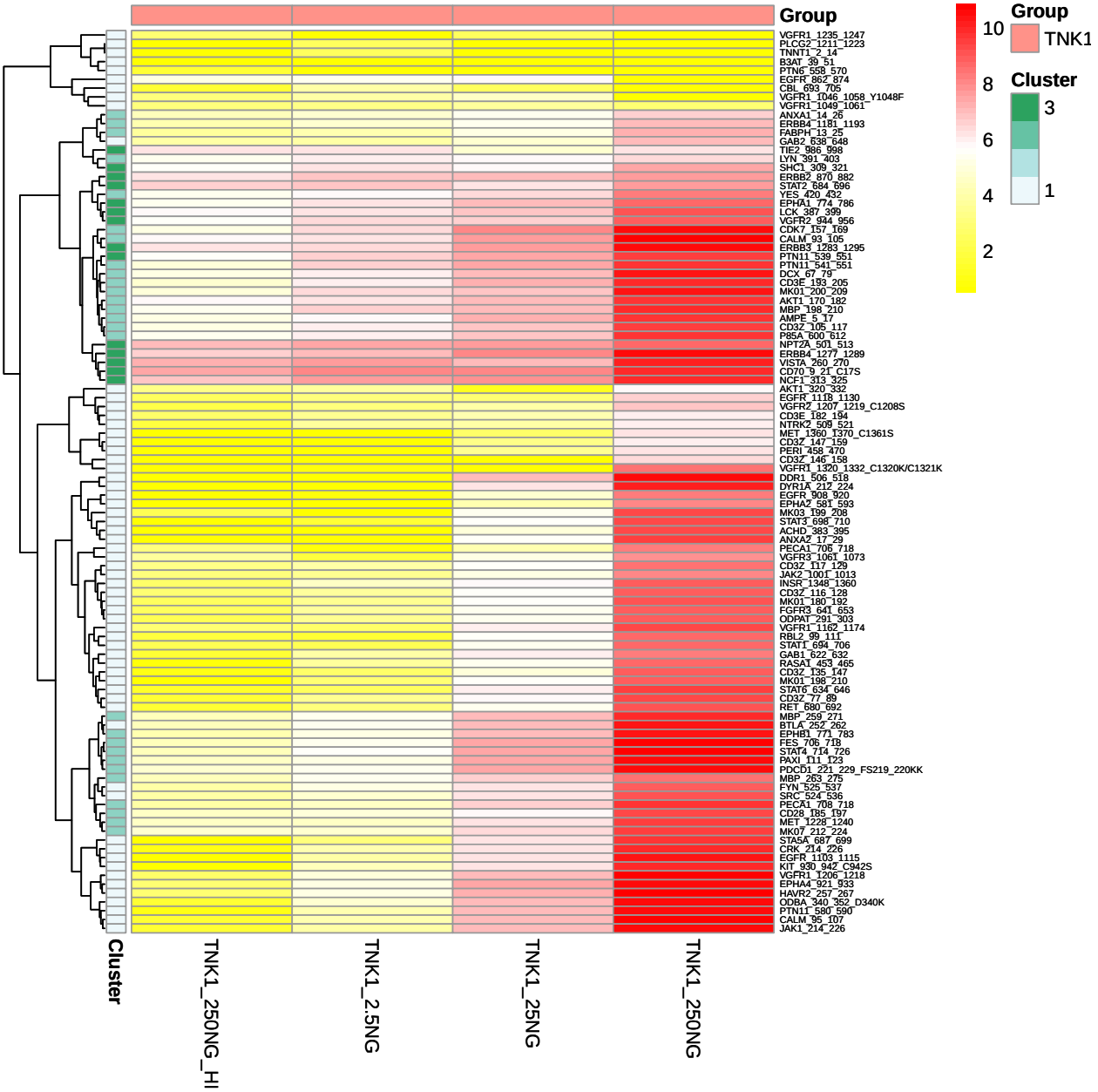

```
cutree(p1$tree_row,6) %>% as.data.frame() %>% rename(Cluster = 1) %>%
  mutate(Cluster = case_when(
    Cluster == 2 ~ 2,
    Cluster == 5 ~ 3,
    T ~ 1
  )) -> pep_annotations


pep_annotations %>% filter(Cluster == 1 ) %>% rownames() -> good_peptides

krsa_heatmap(data_modeled$scaled ,new_pep, scale = "none", cluster_col = T, annotation_row = pep_annota
```



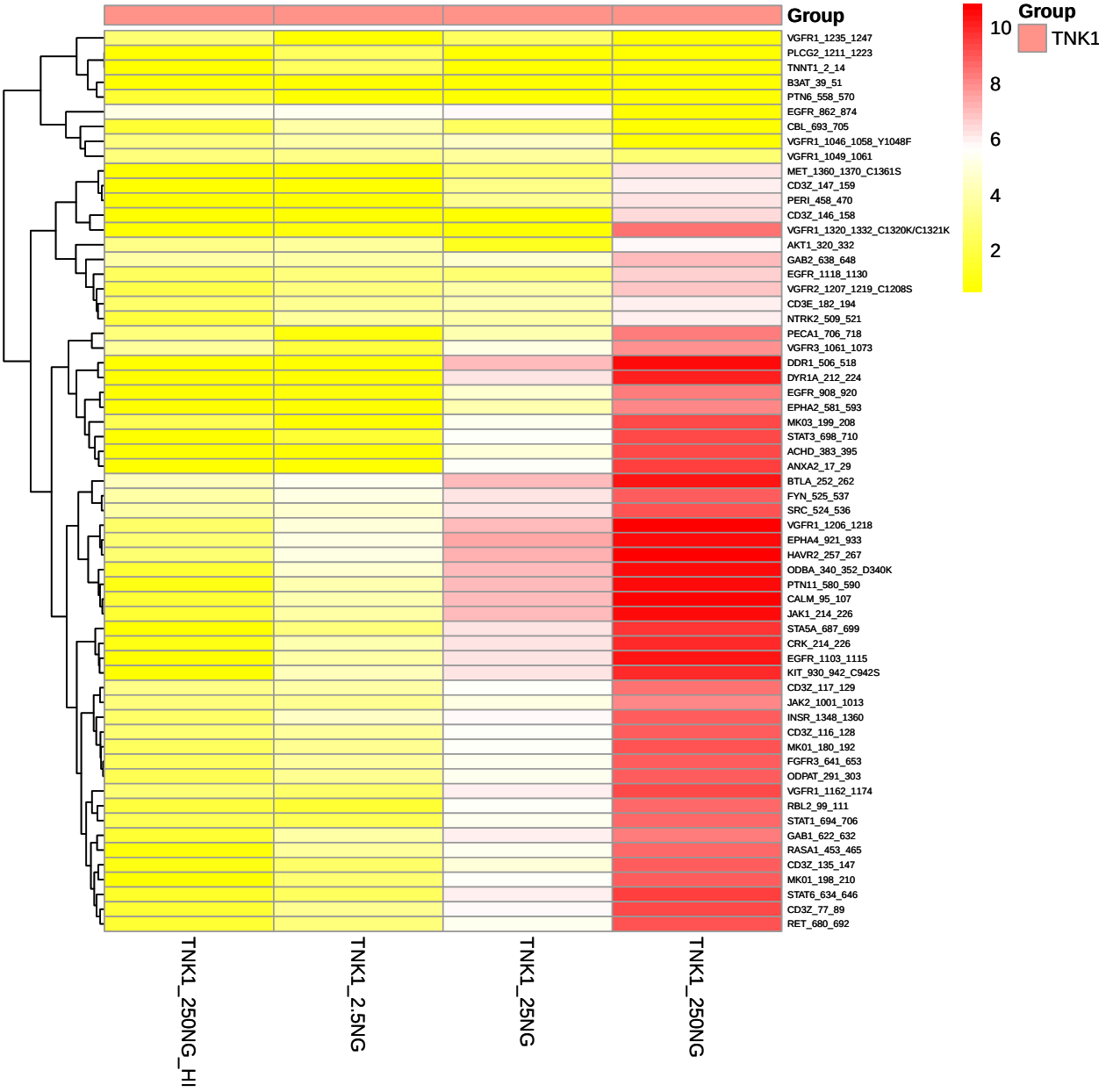

```
krsa_heatmap(data_modeled$scaled ,new_pep, scale = "none", cluster_col = F, annotation_row = pep_annota
```



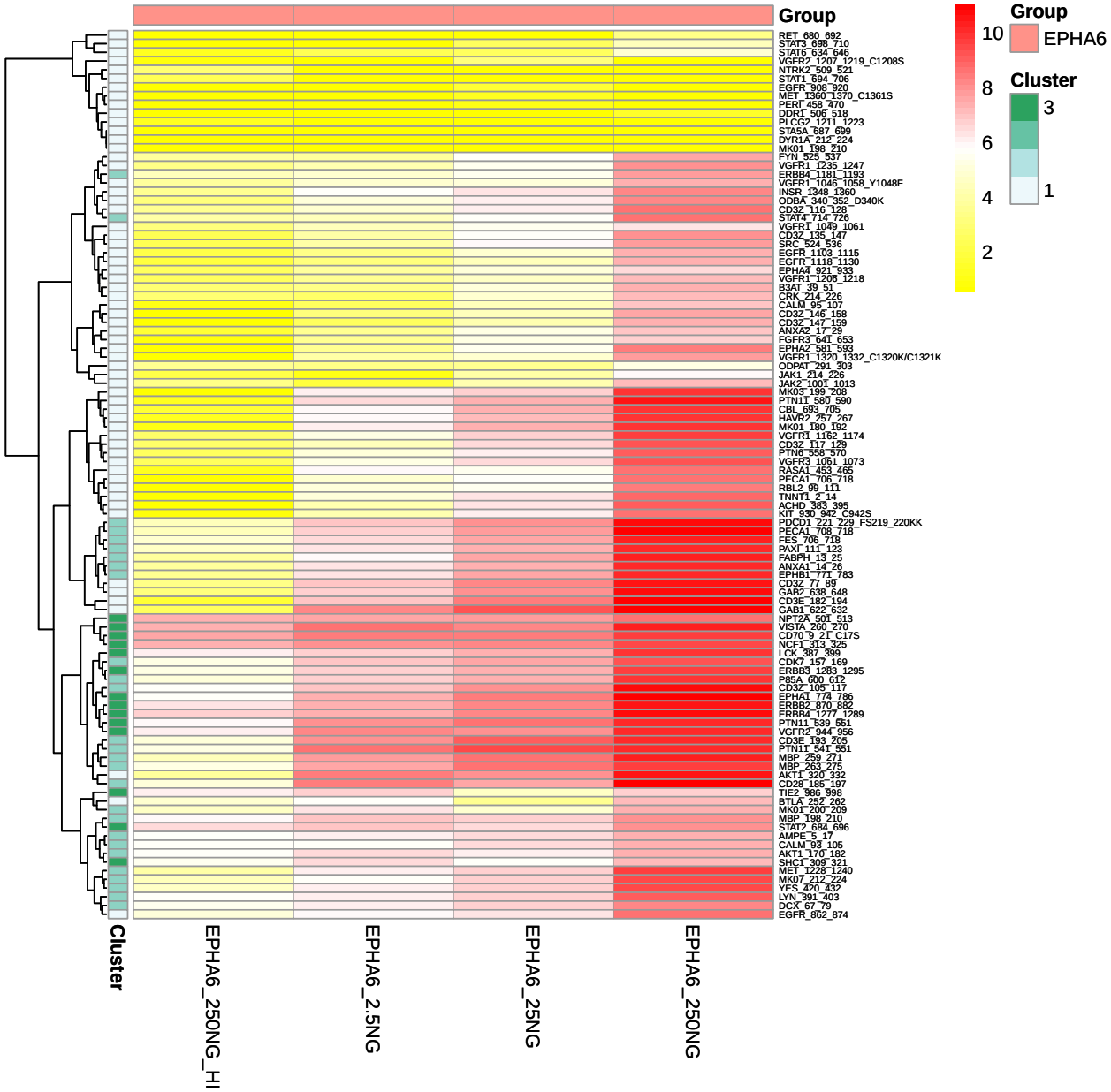

```
# Generates a heatmap using the modeled scaled data
krsa_heatmap(data_modeled$scaled %>% filter(Group == "TNK1"),

             new_pep, scale = "none", cluster_col = F, annotation_row = pep_annotations)
```



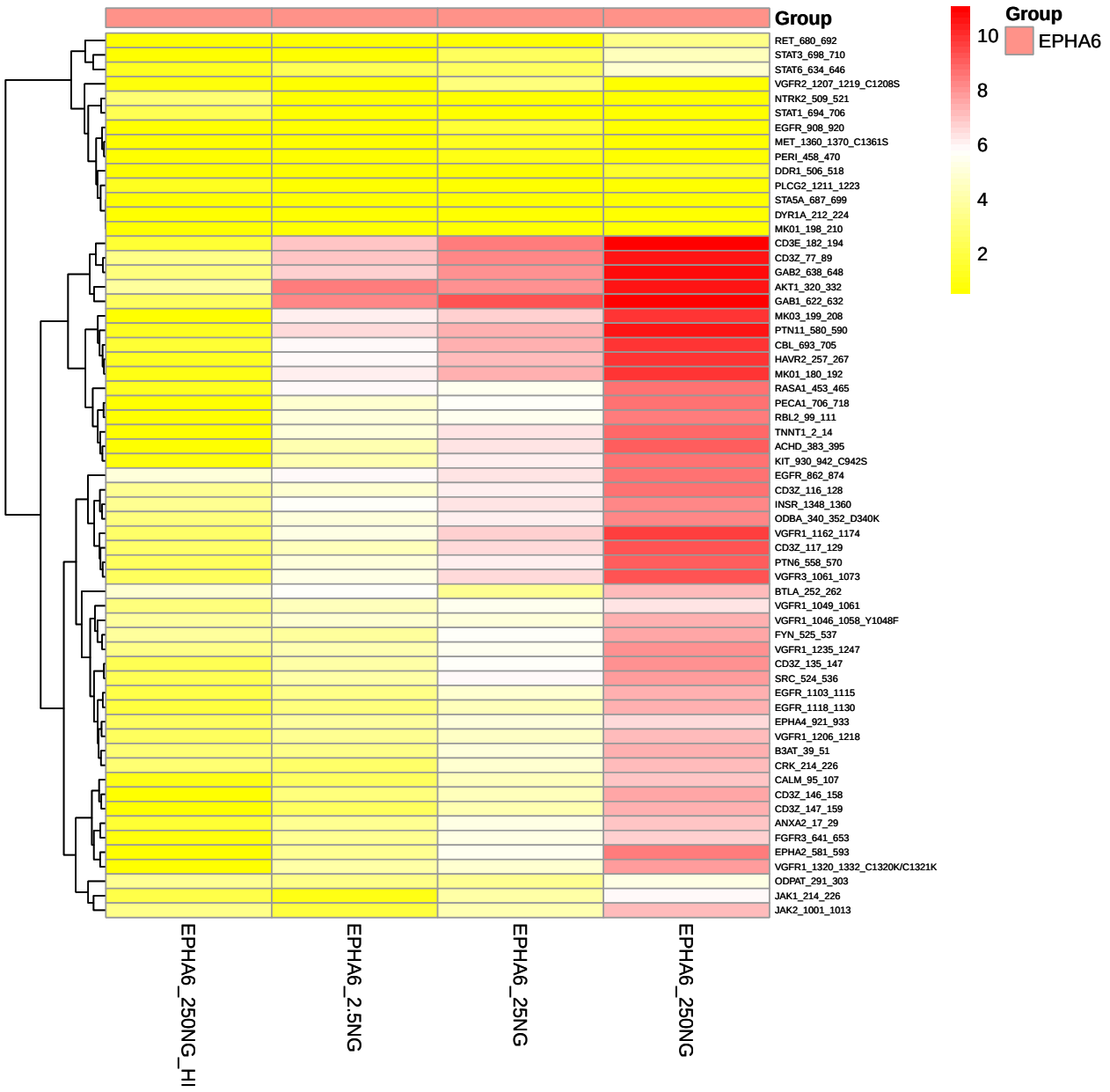

```
krsa_heatmap(data_modeled$scaled %>% filter(Group == "TNK1"),
             good_peptides, scale = "none", cluster_col = F)
```



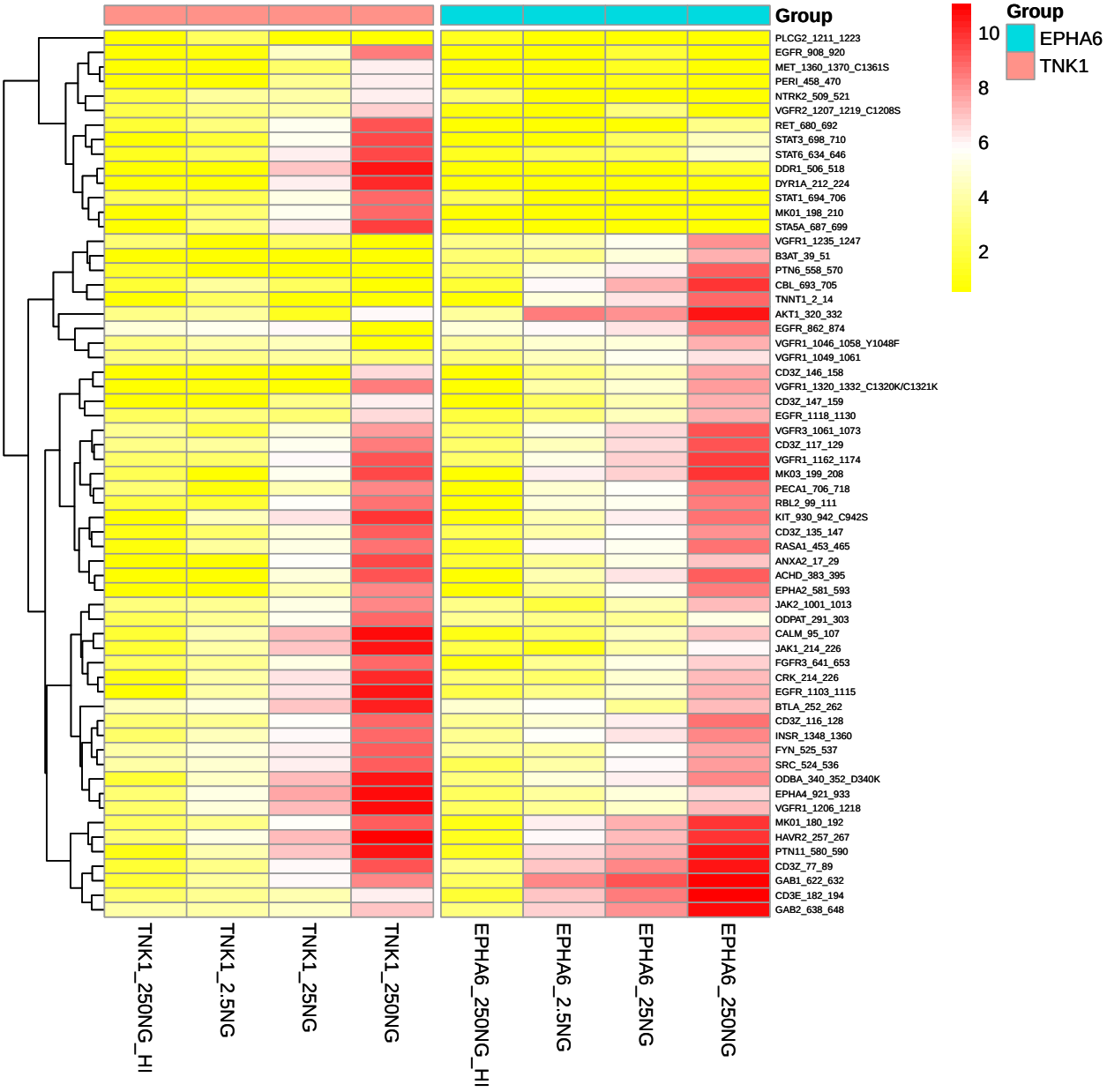

```
krsa_heatmap(data_modeled$scaled %>% filter(Group == "EPHA6"),
             new_pep, scale = "none", cluster_col = F, annotation_row = pep_annotations)
```




```
krsa_heatmap(data_modeled$scaled %>% filter(Group == "EPHA6"),
             good_peptides, scale = "none", cluster_col = F)
```




```
krsa_heatmap(data_modeled$scaled %>% filter(Group %in% c("EPHA6", "TNK1")) %>%
               mutate(SampleName = factor(SampleName, levels = c(
                 "TNK1_250NG_HI", "TNK1_2.5NG", "TNK1_25NG", "TNK1_250NG",
                 "EPHA6_250NG_HI", "EPHA6_2.5NG", "EPHA6_25NG", "EPHA6_250NG"
               ))),
             good_peptides, scale = "none", cluster_col = F, gaps_col = 4)
```



Group Comparison

To compare between samples, a two-group comparison is performed. In this case, the two group comparisons are:

- **TNK1**
- **EPHA6**

The *Slope_Transforemed* ratio between each group, paired by chip, is calculated to the fold change. Based on the fold change, peptides that pass a certain fold change threshold are considered significant hits. Also, quality control steps applied in each comparison to filter out peptides that do not reach specific criteria:

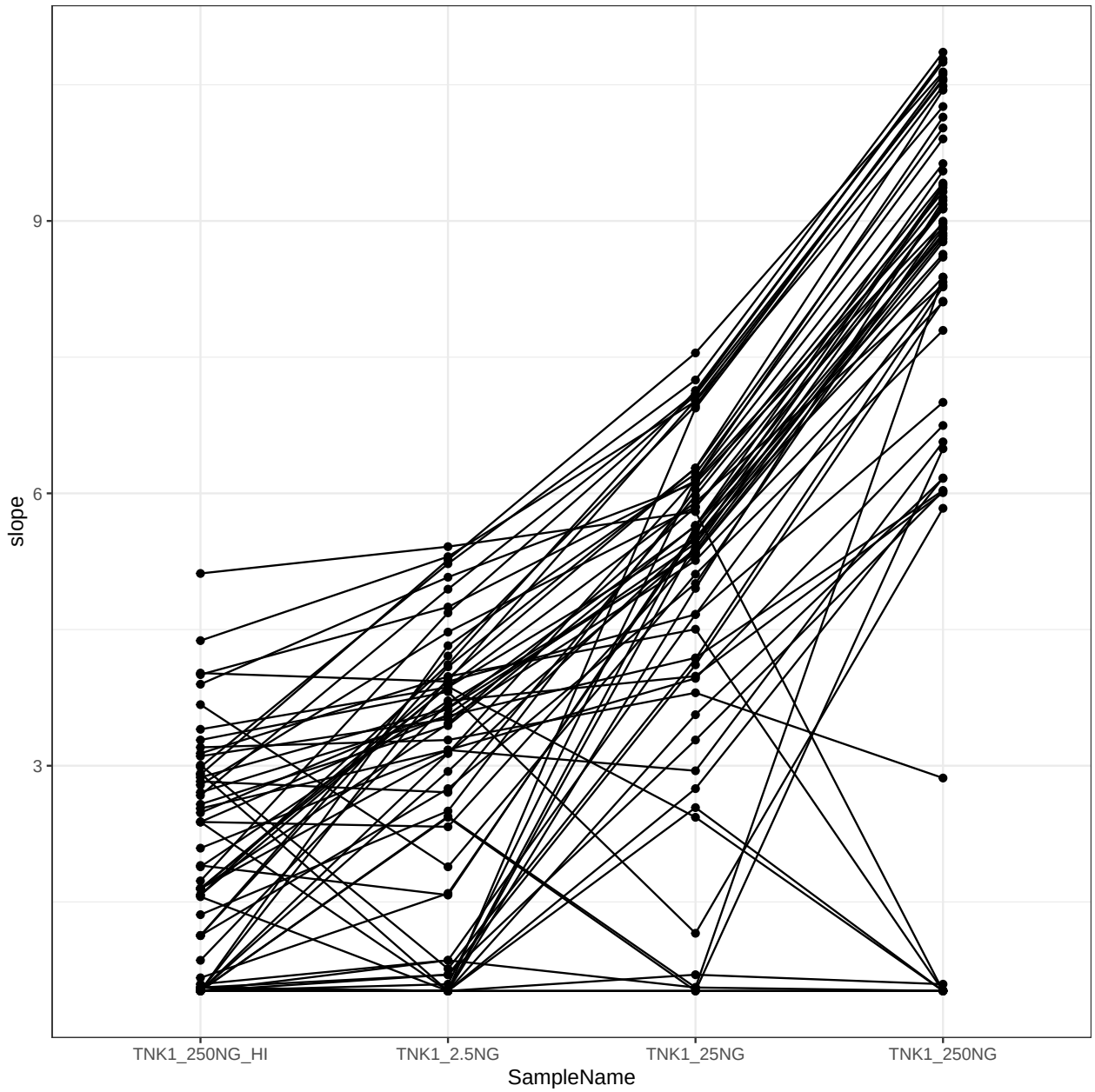
- The *Medain_SigmBg* at max exposure *100ms* must be above a certain value

- R^2 of the linear model fit must be above a threshold value

These *Filtering Parameters* (fold change threshold, QC criteria) can be modified to adjust the stringency of the analysis. The *Filtering Parameters* that are used for this analysis:

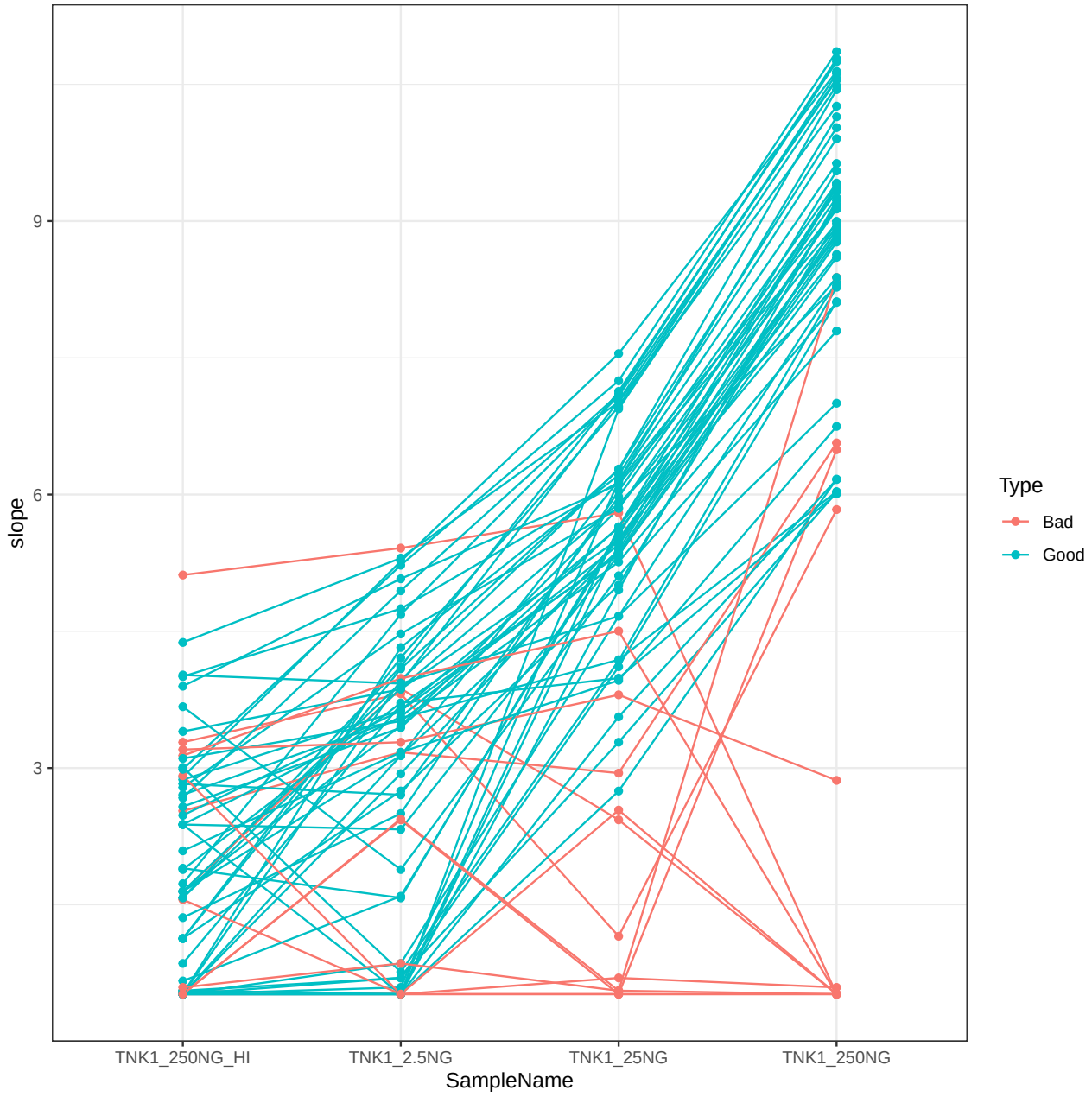
- The *Medain_SigmBg* at max exposure *100ms* must be equal or above 5
- R^2 of the linear model fit must be above or equal 0.9
- Excluded non-specific peptides
- Log fold change (LFC) cutoffs at (0.2,0.3,0.4) between IH and highest concentration

```
data_modeled$scaled %>%
  filter(Group == "TNK1", Peptide %in% good_peptides) %>%
  ggplot(aes(SampleName, slope, group = Peptide), alpha = 1/3) +
  geom_line() +
  geom_point()
```



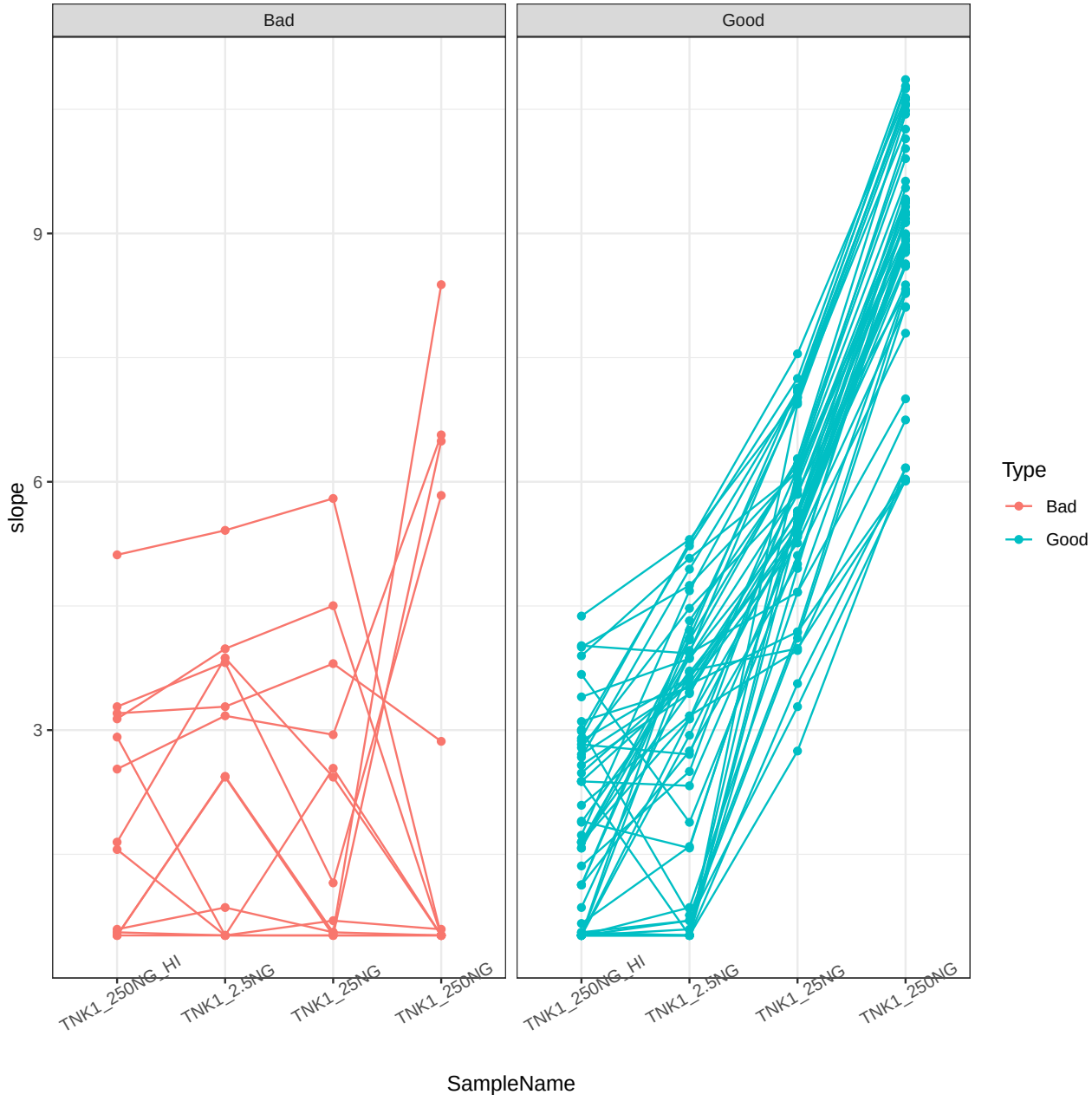

```
data_modeled$scaled %>%
  filter(Group == "TNK1", Peptide %in% good_peptides) %>%
  select(Peptide,SampleName, slope) %>%
  pivot_wider(names_from = SampleName, values_from = slope) %>%
  mutate(low = TNK1_2.5NG - TNK1_250NG_HI,
         mid = TNK1_25NG - TNK1_2.5NG,
         high = TNK1_250NG - TNK1_25NG) %>%
  select(Peptide, mid, high) %>%
  mutate(Avg = rowSums(across(where(is.numeric)) > 0, na.rm = T)) -> data_modeled$signal_direction


tnk1_top_peptides <- filter(data_modeled$signal_direction, Avg == 2) %>% pull(Peptide)
tnk1_bad_peptides <- filter(data_modeled$signal_direction, Avg != 2) %>% pull(Peptide)
```

```
data_modeled$scaled %>%
  filter(Group == "TNK1", Peptide %in% good_peptides) %>%
  mutate(Type = ifelse(Peptide %in% tnk1_top_peptides, "Good", "Bad")) %>%
  ggplot(aes(SampleName, slope, group = Peptide, color = Type),  alpha = 1/3) +
  geom_line() +
  geom_point()
```



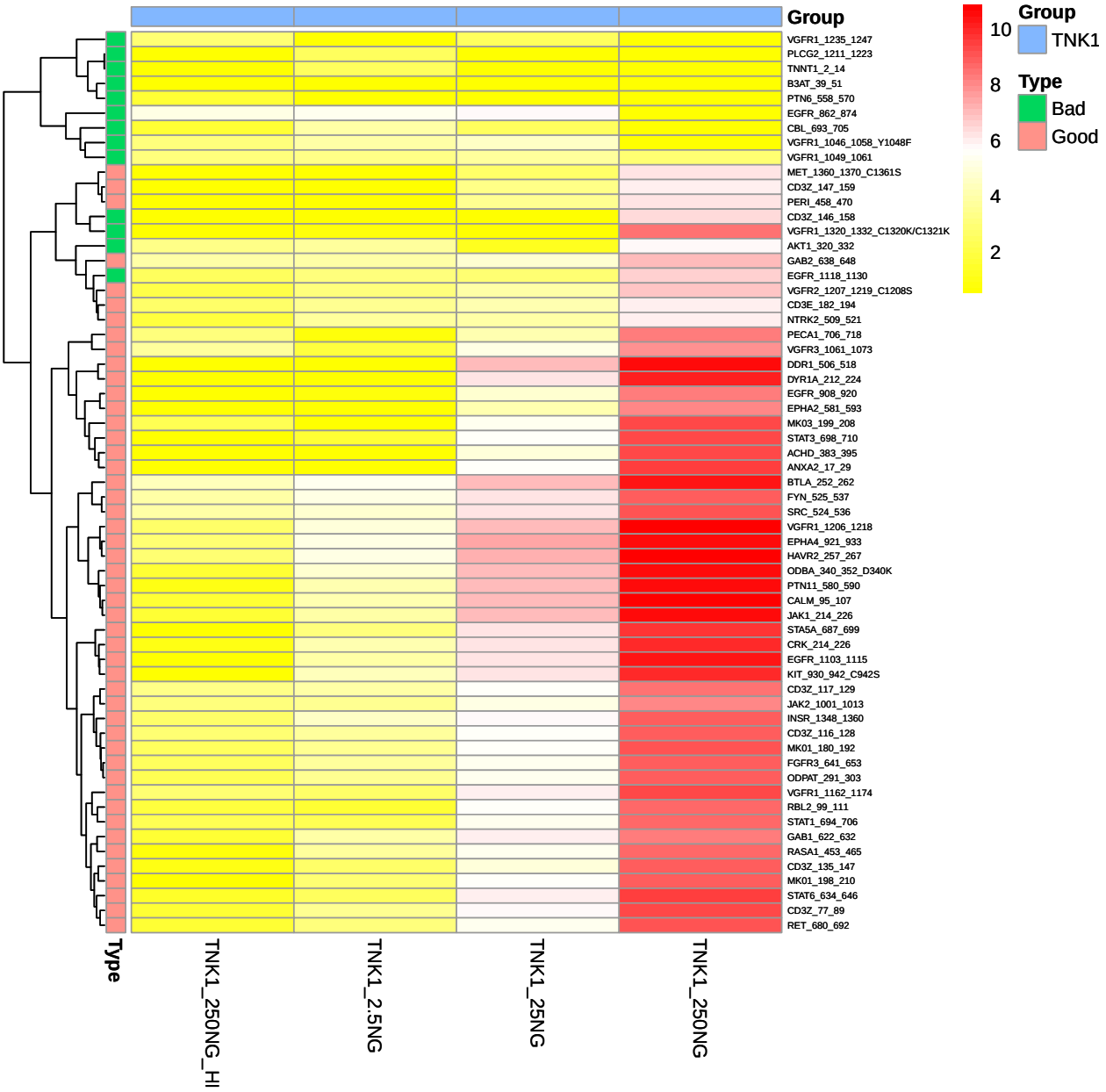

```
data_modeled$scaled %>%
  filter(Group == "TNK1", Peptide %in% good_peptides) %>%
  mutate(Type = ifelse(Peptide %in% tnk1_top_peptides, "Good", "Bad")) %>%
```

```
  ggplot(aes(SampleName, slope, group = Peptide, color = Type), alpha = 1/3) +
  geom_line() +
  geom_point() +
  facet_wrap(~Type) +
  theme(axis.text.x = element_text(angle = 30))
```




```
tibble(
  Peptide = good_peptides
) %>%
  mutate(Type = ifelse(Peptide %in% tnk1_top_peptides, "Good", "Bad")) %>%
  column_to_rownames("Peptide") -> pep_annotations_tnk1
```

```
krsa_heatmap(data_modeled$scaled %>% filter(Group == "TNK1"),
             good_peptides, scale = "none", cluster_col = F, annotation_row = pep_annotations_tnk1)
```

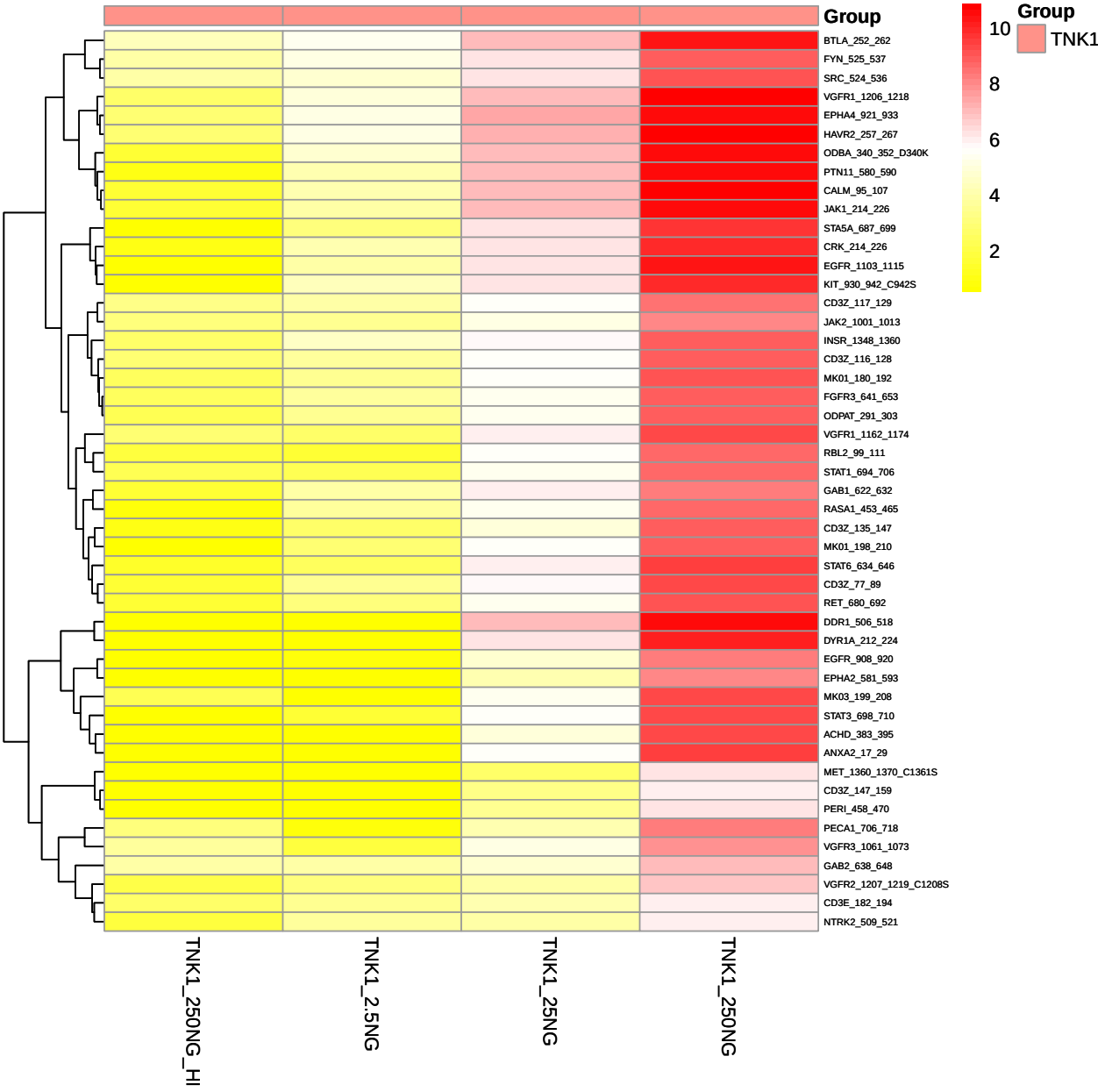



```
# Define Groups to be compared
comparisons <- list(Comp1 = c("TNK1_250NG", "TNK1_250NG_HI"))

# This function calculates log2 fold change values between the defined groups
```

```r
# The byChip argument lets you calculates the log2 fold change the results within each chip
krsa_group_diff(data_modeled$scaled %>% mutate(Group = SampleName),
                comparisons$Comp1, tnk1_top_peptides, byChip = F) -> diff_df
```

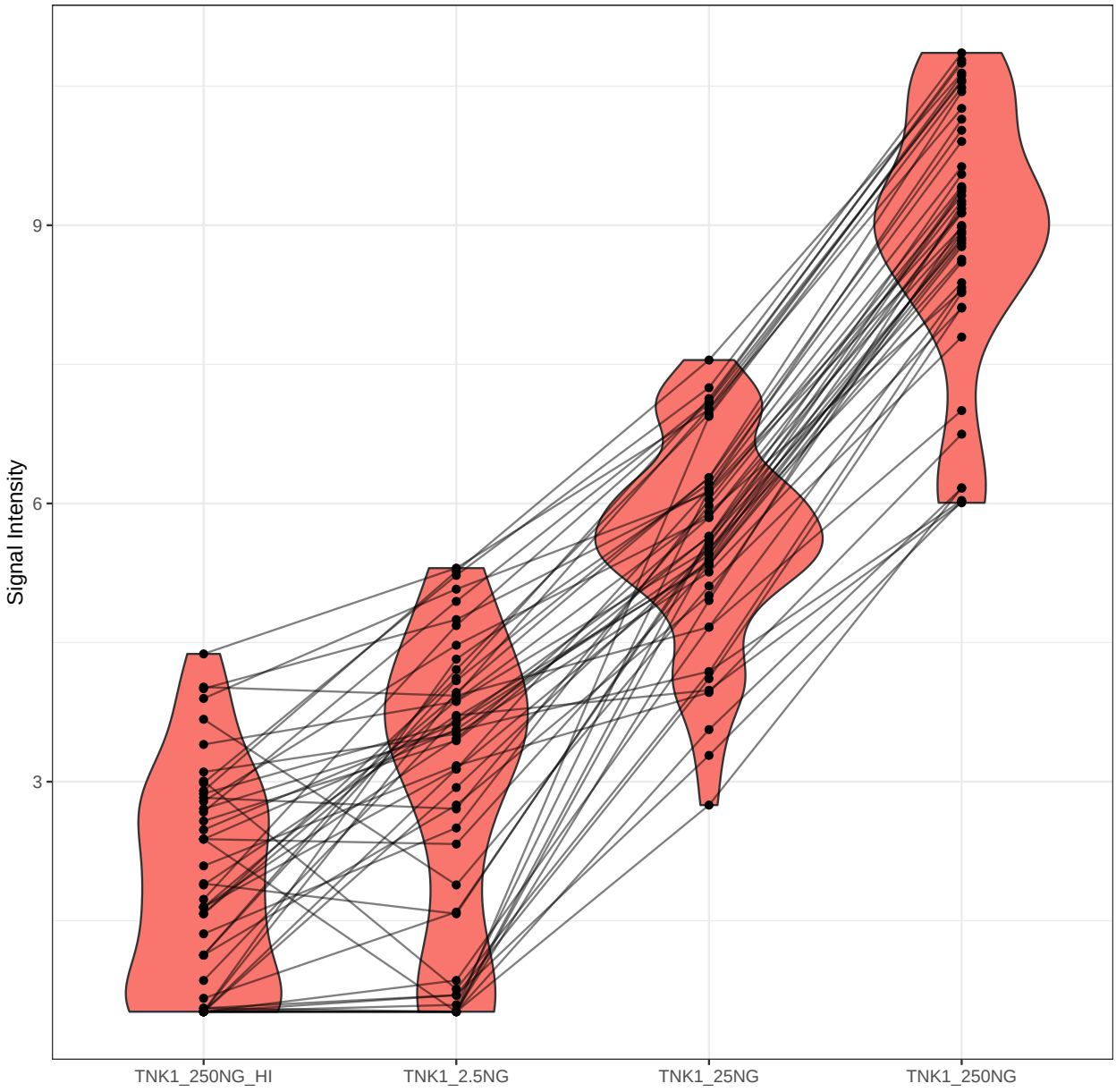
TNK1 (Heatmap)

```
# generates a heatmap using the selected groups and peptides
krsa_heatmap(data_modeled$scaled %>% filter(Group == "TNK1"),
             peptides = tnk1_top_peptides,
             scale = "none",
             cluster_col = F
             )
```

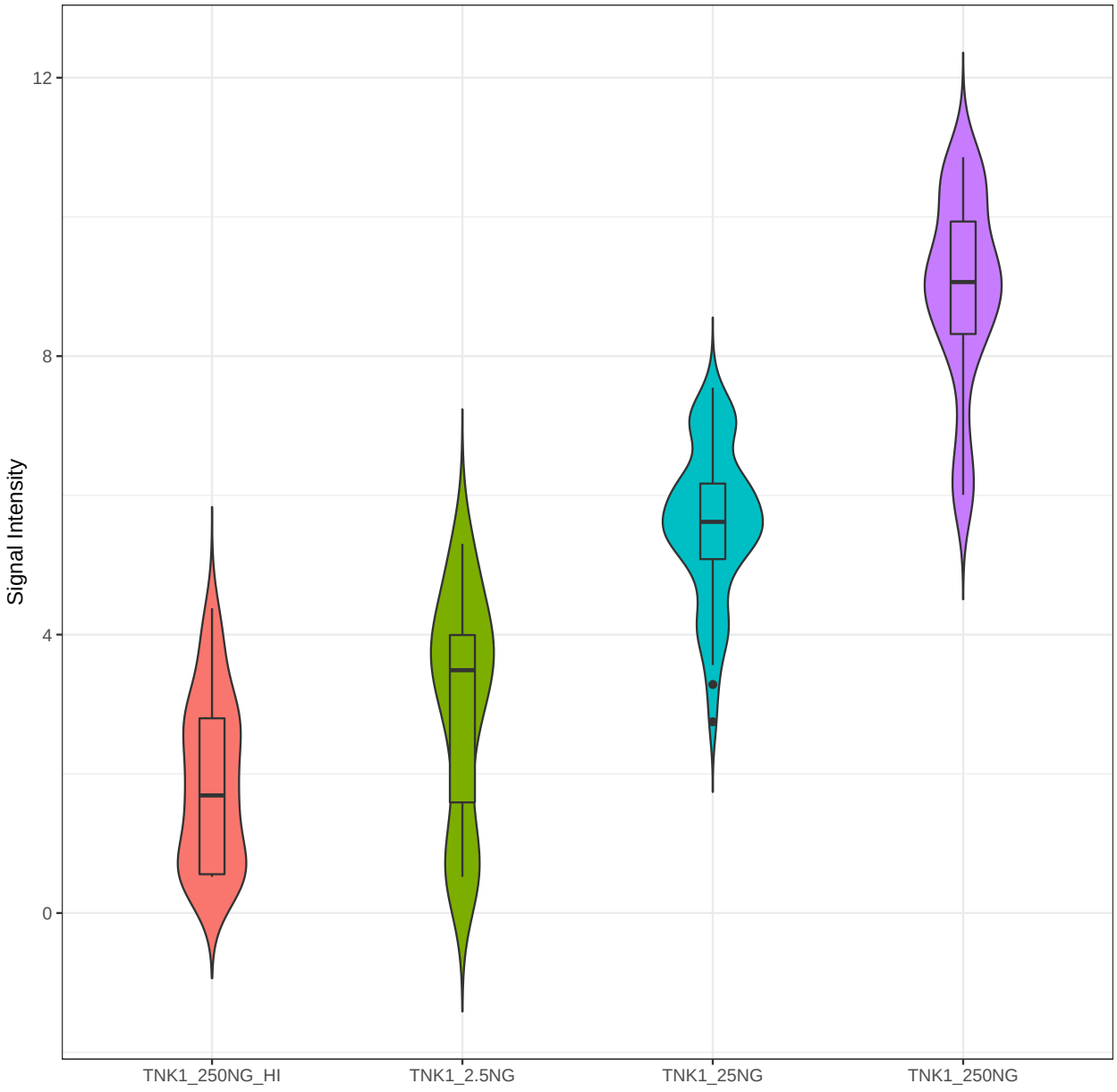


TNK1 (Violin Plot)

```
# generates a violin plot using the selected groups and peptides
krsa_violin_plot(data_modeled$scaled %>% filter(Group == "TNK1"),
                 tnk1_top_peptides, facet = F,
                   facet_factor = "Group", groups = "TNK1")
```



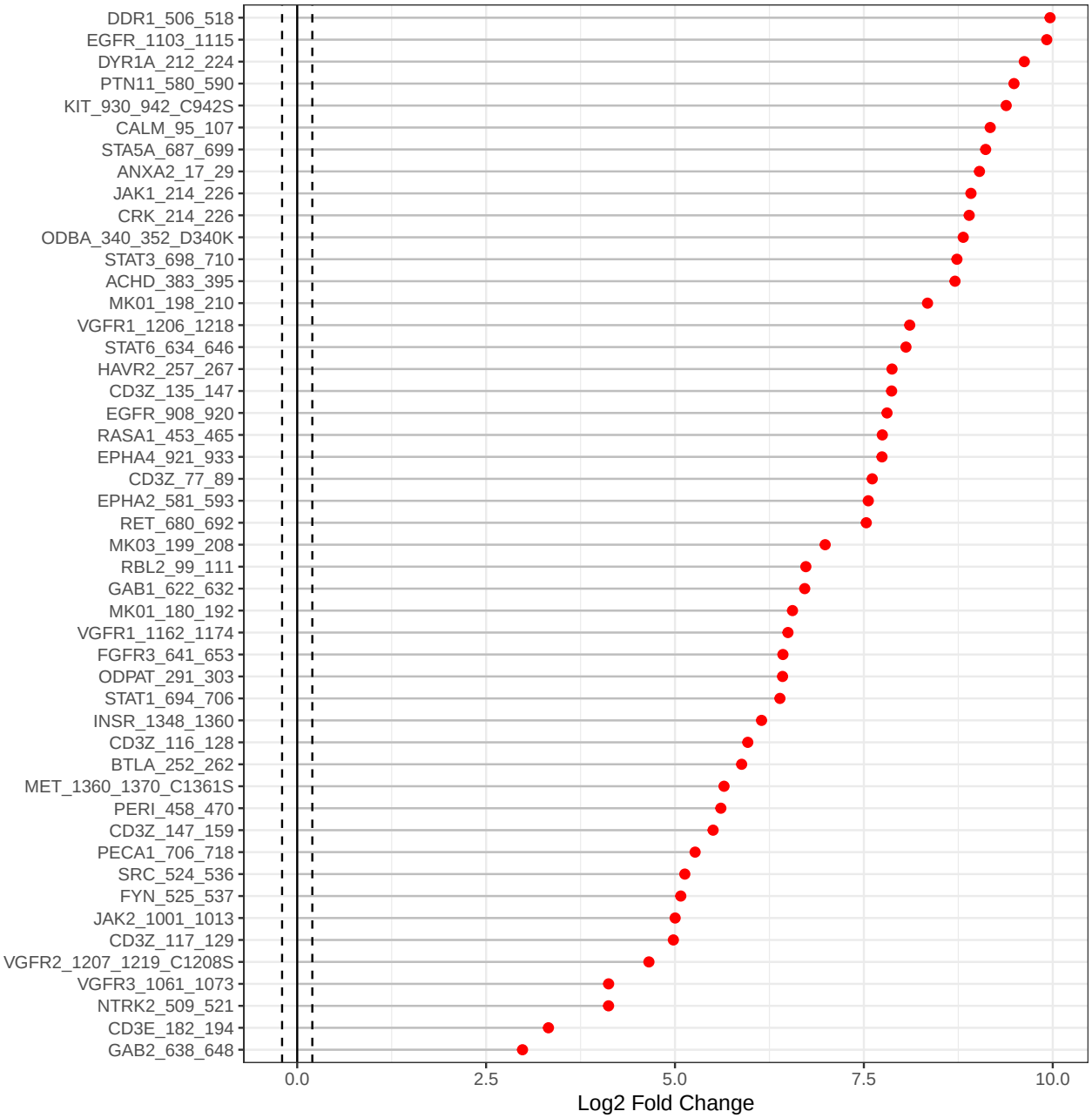

```
# generate a grouped violin/boxplot plot using the selected groups and peptides with more options, like
krsa_violin_plot_grouped(data_modeled$scaled %>% filter(Group == "TNK1") %>%
                  mutate(Group = SampleName),
                         tnk1_top_peptides, comparisons,
                         dots = F,
                test = F, avg_line = F)
```


Signal Intensity
12
8
4
0
TNK1_250NG_HI
TNK1_2.5NG
TNK1_25NG
TNK1_250NG

TNK1 (Waterfall Plot)

This waterfall represents the log2 fold changes between the two groups at each peptide.

```
# generates a waterfall of the log2 fold change values for the selected peptide (top peptides)
krsa_waterfall(diff_df, lfc_thr = 0.2, byChip = F)
```



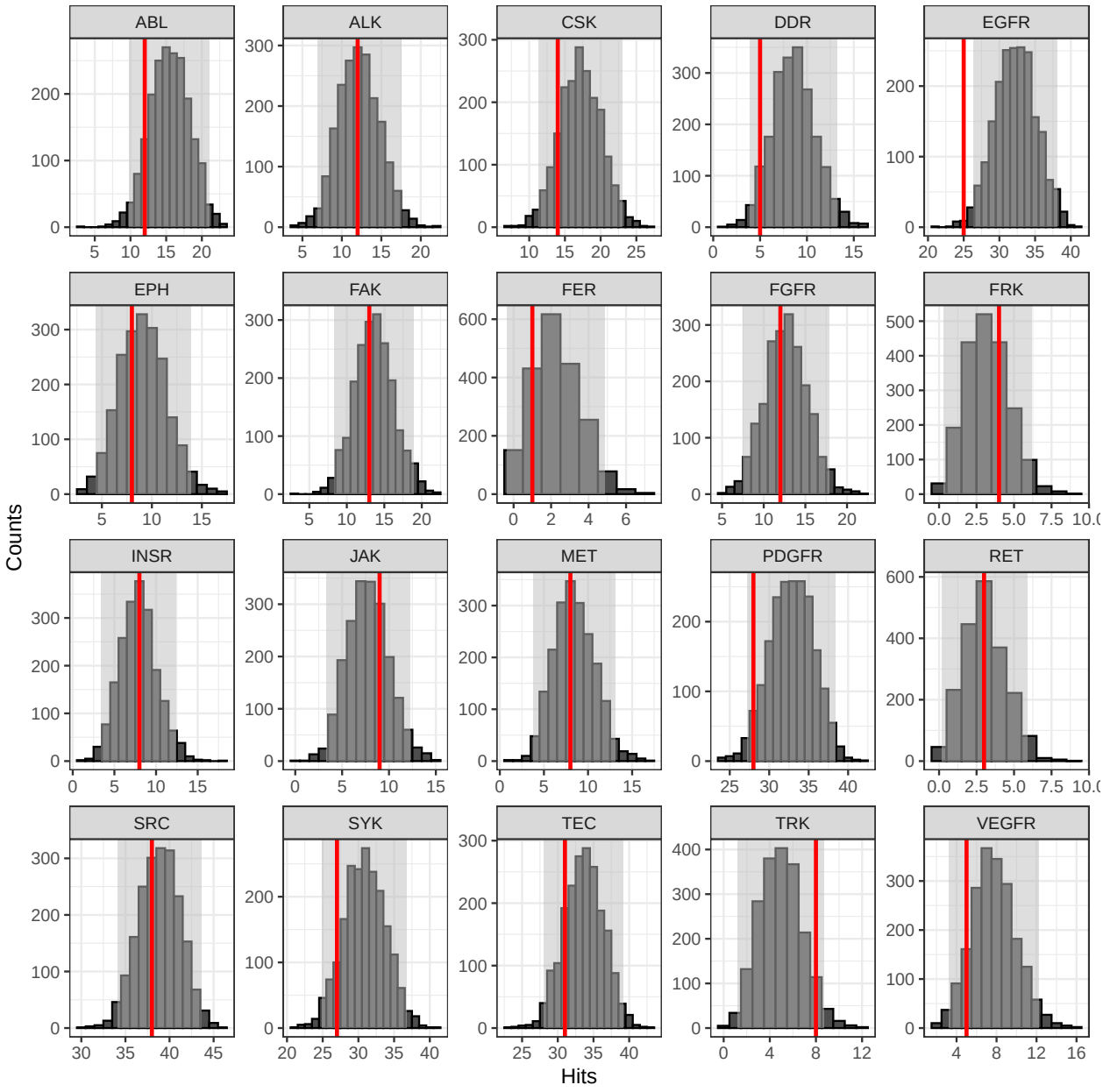
TNK1 (Upstream Kinase Analysis)

```
# load in chip coverage and kinase-substrate files OR upload your own files
# if PTK chip, use:
chipCov <- KRSA_coverage_PTK_PamChip_86402_v1
KRSA_file <- KRSA_Mapping_PTK_PamChip_86402_v1

# STK chip
# chipCov <- KRSA_coverage_STK_PamChip_87102_v2
# KRSA_file <- KRSA_Mapping_STK_PamChip_87102_v1



# run the KRSA function to do the random sampling analysis, set seed that can be used later to reproduc
krsa(tnk1_top_peptides, return_count = T, seed = 123, itr = 2000,
     map_file = KRSA_file, cov_file = chipCov) -> fin

# View the Z score table
kable(head(fin$KRSA_Table,25), digits = 3)
```

Kinase	Observed	SamplingAvg	SD	Z
EGFR	25	32.257	2.932	-2.475
PDGFR	28	32.834	2.802	-1.725
TRK	8	4.967	1.874	1.618
DDR	5	8.581	2.328	-1.538
SYK	27	30.863	2.951	-1.309
ABL	12	15.472	2.808	-1.236
VEGFR	5	7.740	2.242	-1.222
CSK	14	17.160	2.945	-1.073
FER	1	2.273	1.309	-0.972
TEC	31	33.614	2.787	-0.938
AXL	17	14.442	2.761	0.927
RYK	1	0.502	0.629	0.791
SEV	0	0.258	0.438	-0.590
ACK	0	0.250	0.433	-0.576
JAK	9	7.754	2.245	0.555
FRK	4	3.227	1.468	0.527
EPH	8	9.106	2.389	-0.463
SRC	38	38.847	2.366	-0.358
FGFR	12	12.674	2.602	-0.259
FAK	13	13.648	2.650	-0.245
MET	8	8.447	2.329	-0.192
ALK	12	12.187	2.646	-0.071
INSR	8	7.891	2.238	0.048
RET	3	3.042	1.432	-0.029

```
# to save file
#fin$KRSA_Table %>% write_delim("output/Z_Scores_tables/acrossChip_KRSA_FullTable_comp1.txt", delim = "

# find top and bottom kinases
```

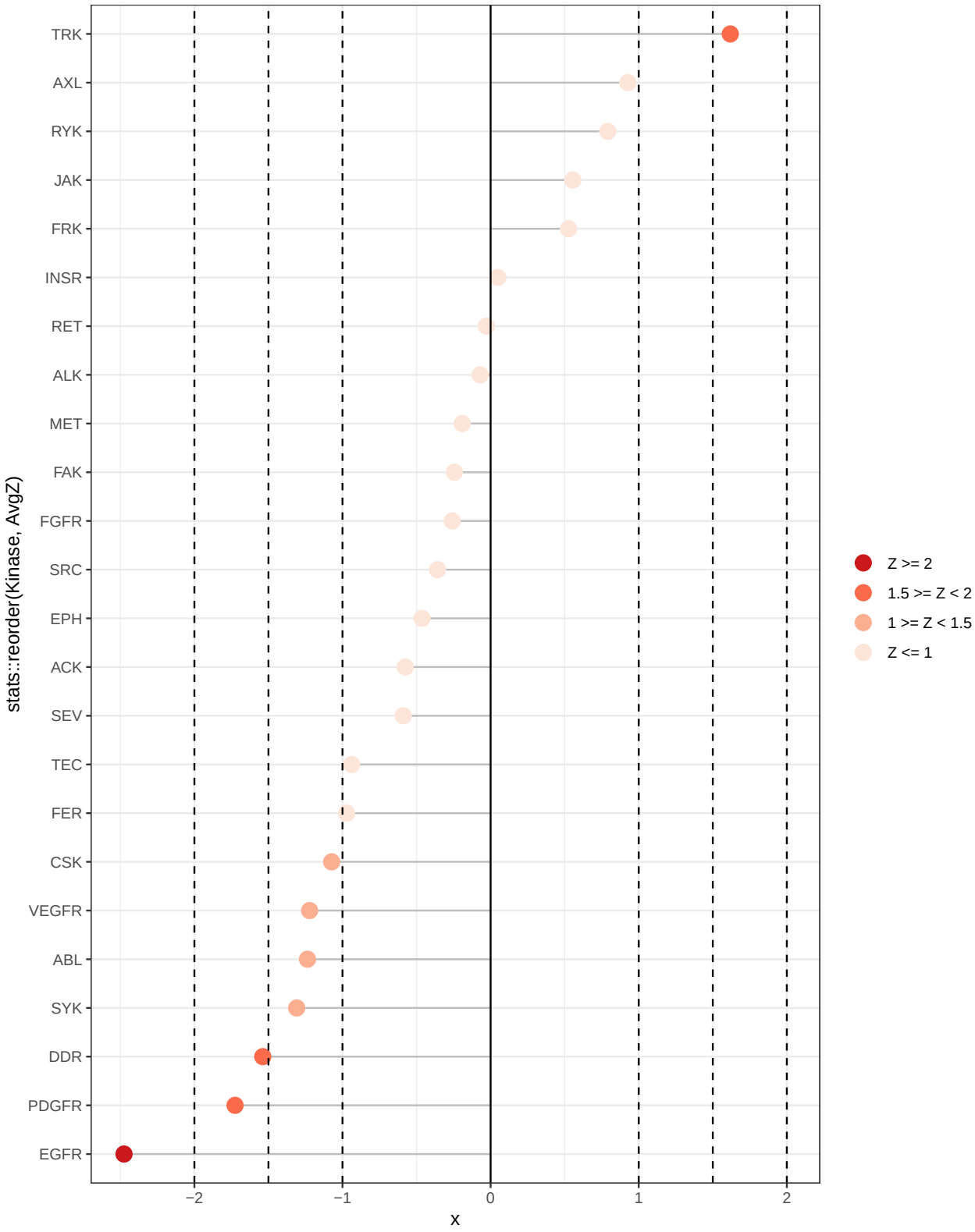
```
bothways <- c(pull(head(fin$KRSA_Table, 10), Kinase), pull(tail(fin$KRSA_Table, 10), Kinase))

# Use these kinase to generate histogram plots for each selected kinase
krsa_histogram_plot(fin$KRSA_Table, fin$count_mtx, bothways)
```



TNK1 (Z Scores Plot)

We will plot the individual and averaged Z scores using both the across and within chip analyses.

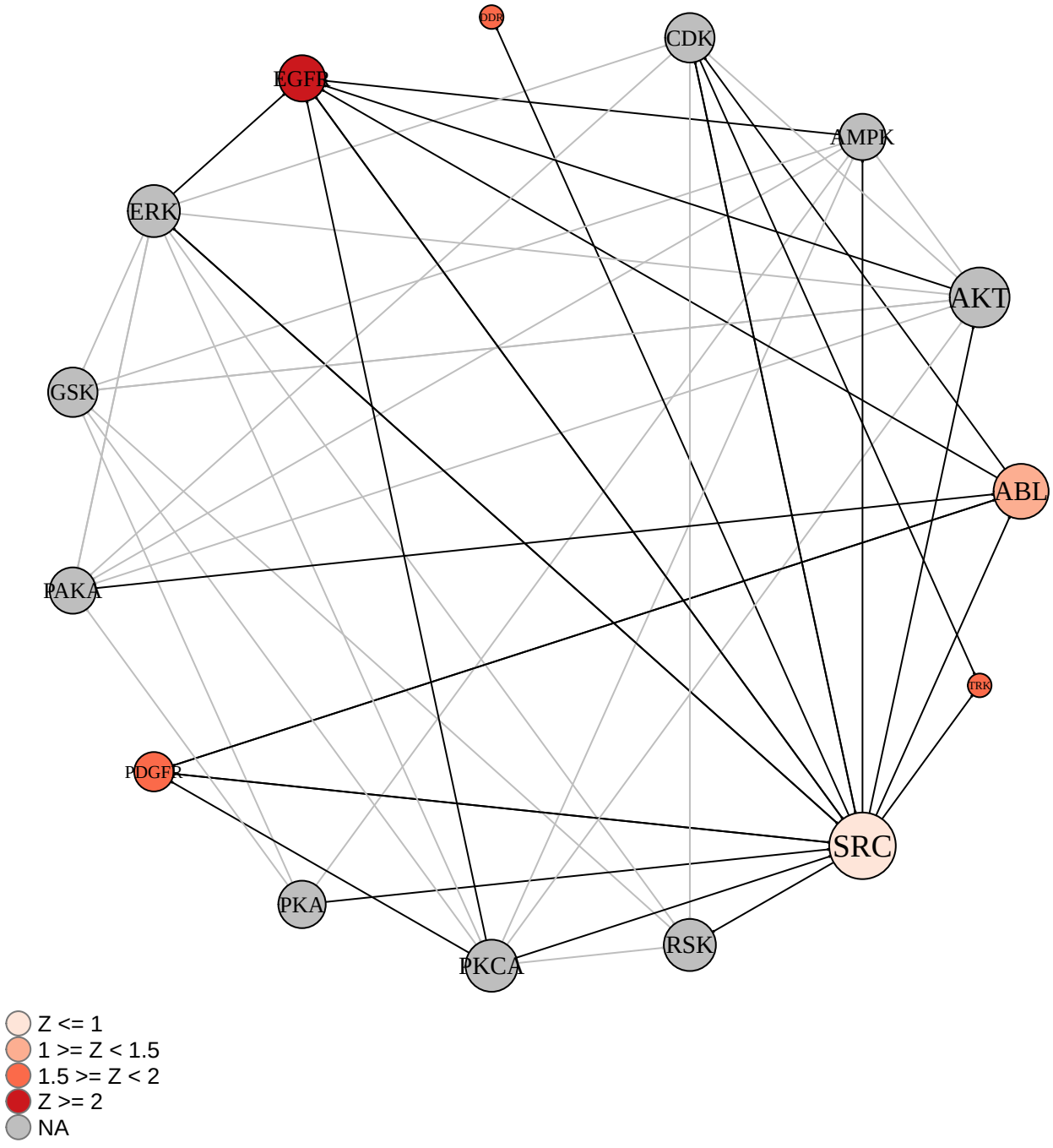
```
krsa_zscores_plot <- function(Ztable) {
  Ztable %>%
    dplyr::filter(!Kinase %in% c("VRK2", "BARK1")) %>%
    dplyr::mutate(
      AvgZ = Z,
      breaks = cut(
        abs(AvgZ),
        breaks = c(0, 1, 1.5, 2, Inf),
        right = F,
        labels = c("Z <= 1", "1 >= Z < 1.5", "1.5 >= Z < 2", "Z >= 2")
        )) -> x
  x %>%
    ggplot2::ggplot() +
    ggplot2::geom_segment(ggplot2::aes(x=0, xend= AvgZ,
                                       y= stats::reorder(Kinase,AvgZ),
                                       yend= stats::reorder(Kinase,AvgZ)),
                          color="grey") +
      ggplot2::geom_point(ggplot2::aes(AvgZ, stats::reorder(Kinase,AvgZ), color = breaks), size = 4) +
    ggplot2::geom_vline(xintercept = 0) +
    ggplot2::geom_vline(xintercept = c(-1, -1.5, -2, 1, 1.5, 2),linetype="dashed") +
    ggplot2::scale_color_brewer(palette="Reds", drop = F, guide = ggplot2::guide_legend(reverse=TRUE, t
}



krsa_zscores_plot(fin$KRSA_Table)
```


TRK
AXL
RYK
JAK
FRK
INSR
RET
ALK
MET
FAK
FGFR
SRC
EPH
ACK
SEV
TEC
FER
CSK
VEGFR
ABL
SYK
DDR
PDGFR
EGFR
stats::reorder(Kinase, AvgZ)
−2
−1
0
1
2
x
Z >= 2
1.5 >= Z < 2
1 >= Z < 1.5
Z <= 1

TNK1 (Ball Model Network)

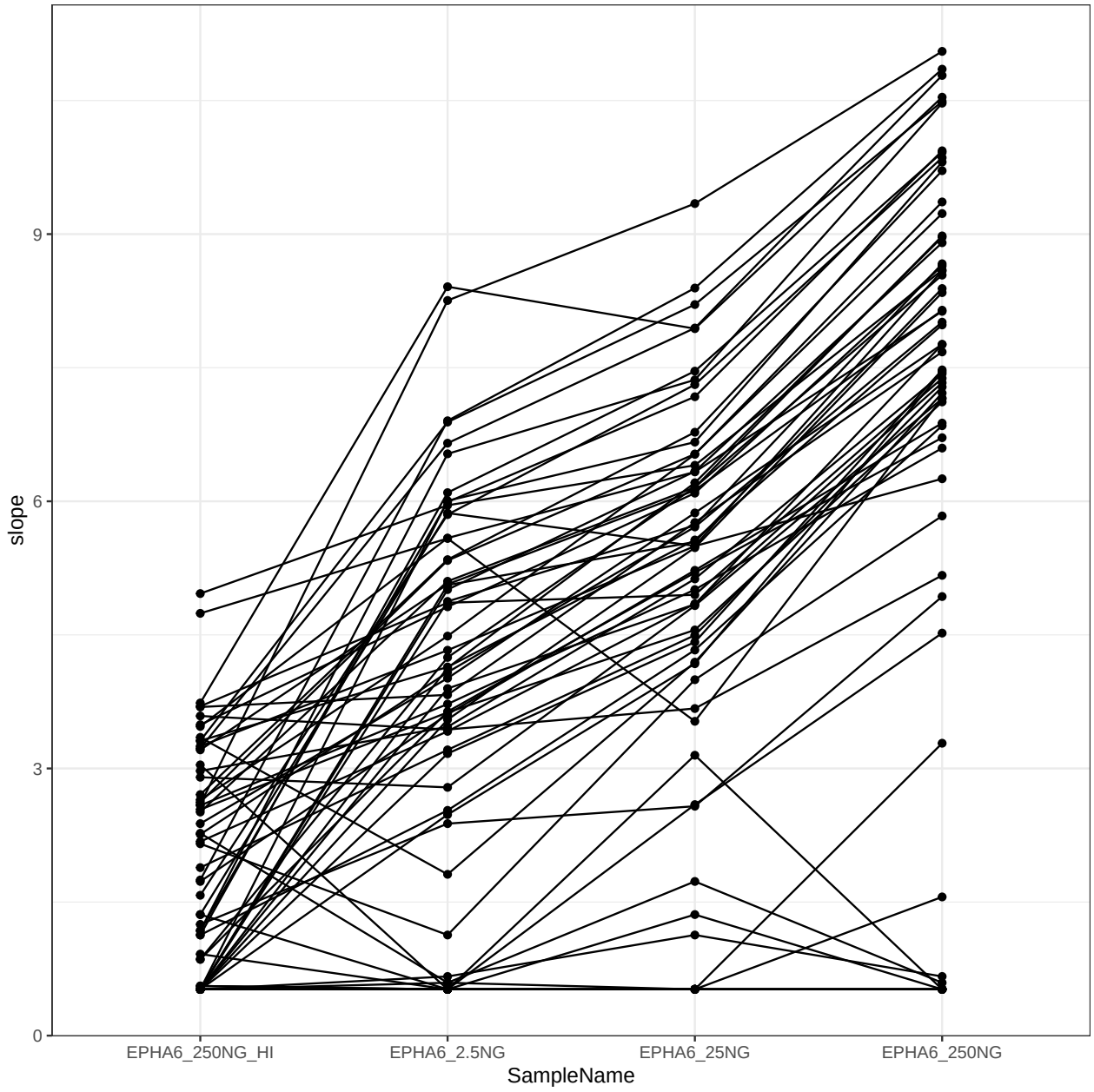
We will view the ball model network function, to generate a model representing the protein-protein interactions between kinases

```
# Plot the network ball model
krsa_ball_model(c("TRK"), fin$KRSA_Table %>% mutate(AvgZ = Z), 15, 2.5, 4.8)
```



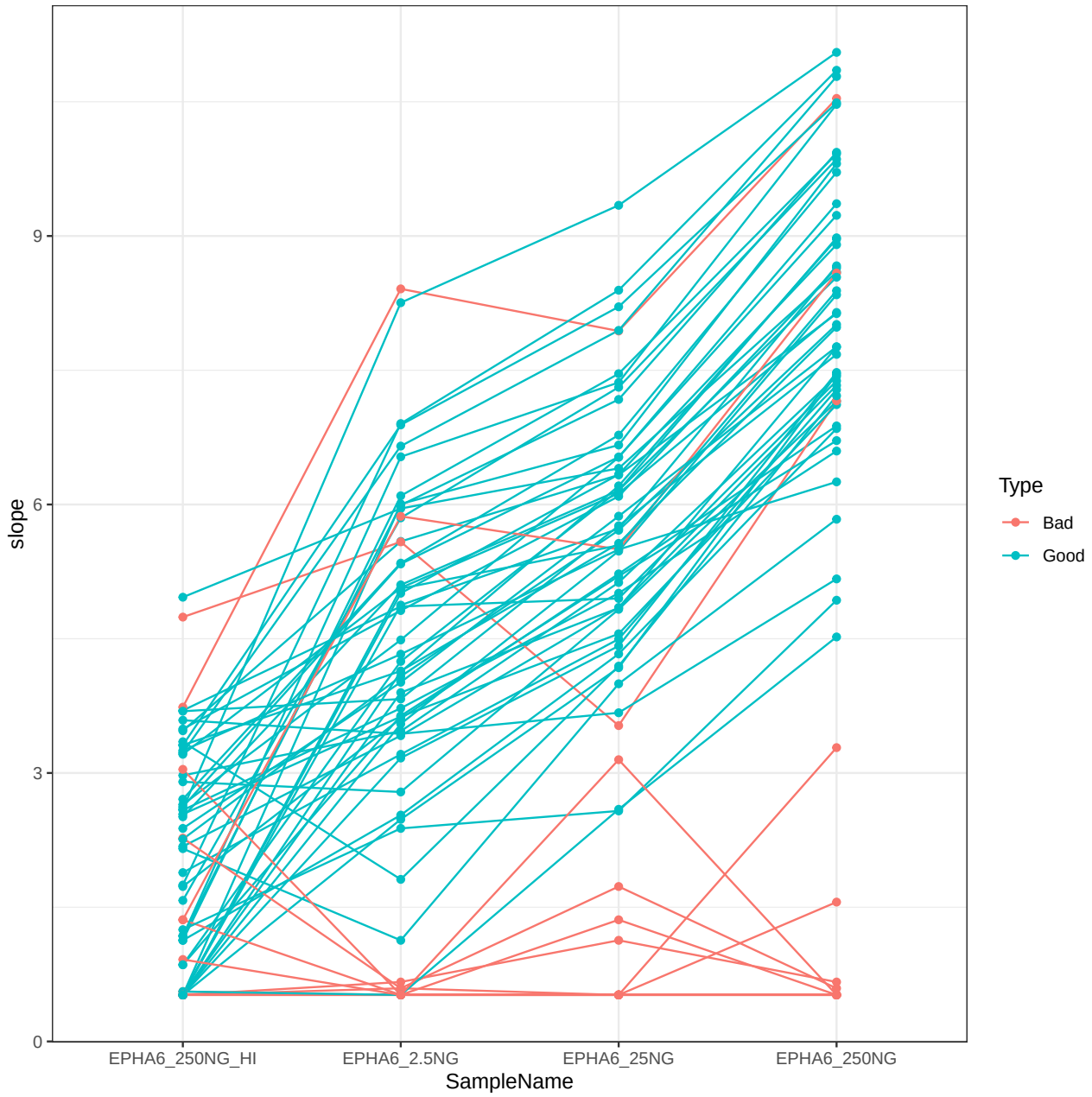
EPHA6

```
data_modeled$scaled %>%
  filter(Group == "EPHA6", Peptide %in% good_peptides)  %>%
  ggplot(aes(SampleName, slope, group = Peptide), alpha = 1/3) +
  geom_line() +
  geom_point()
```

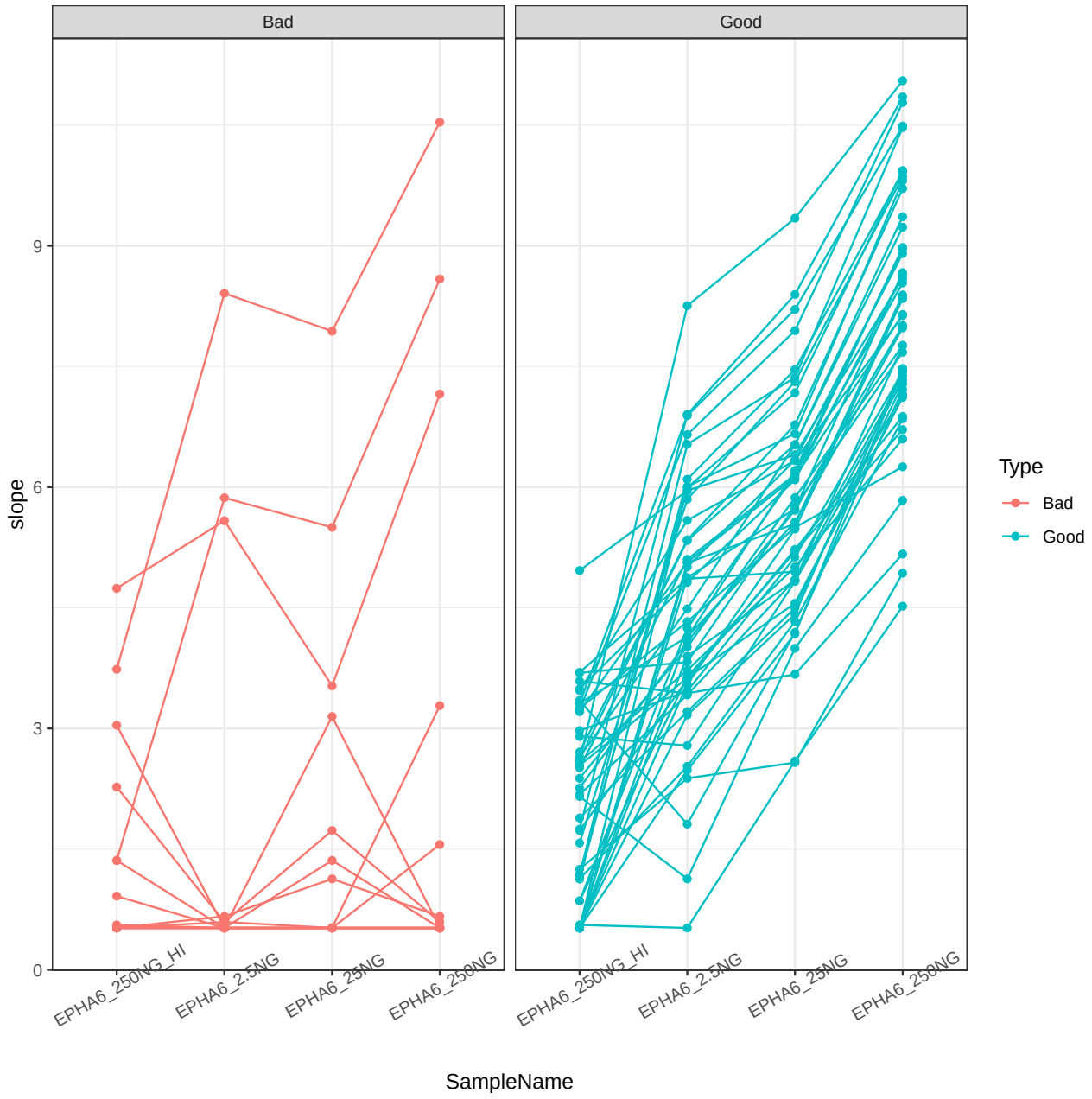



```
data_modeled$scaled %>%
```

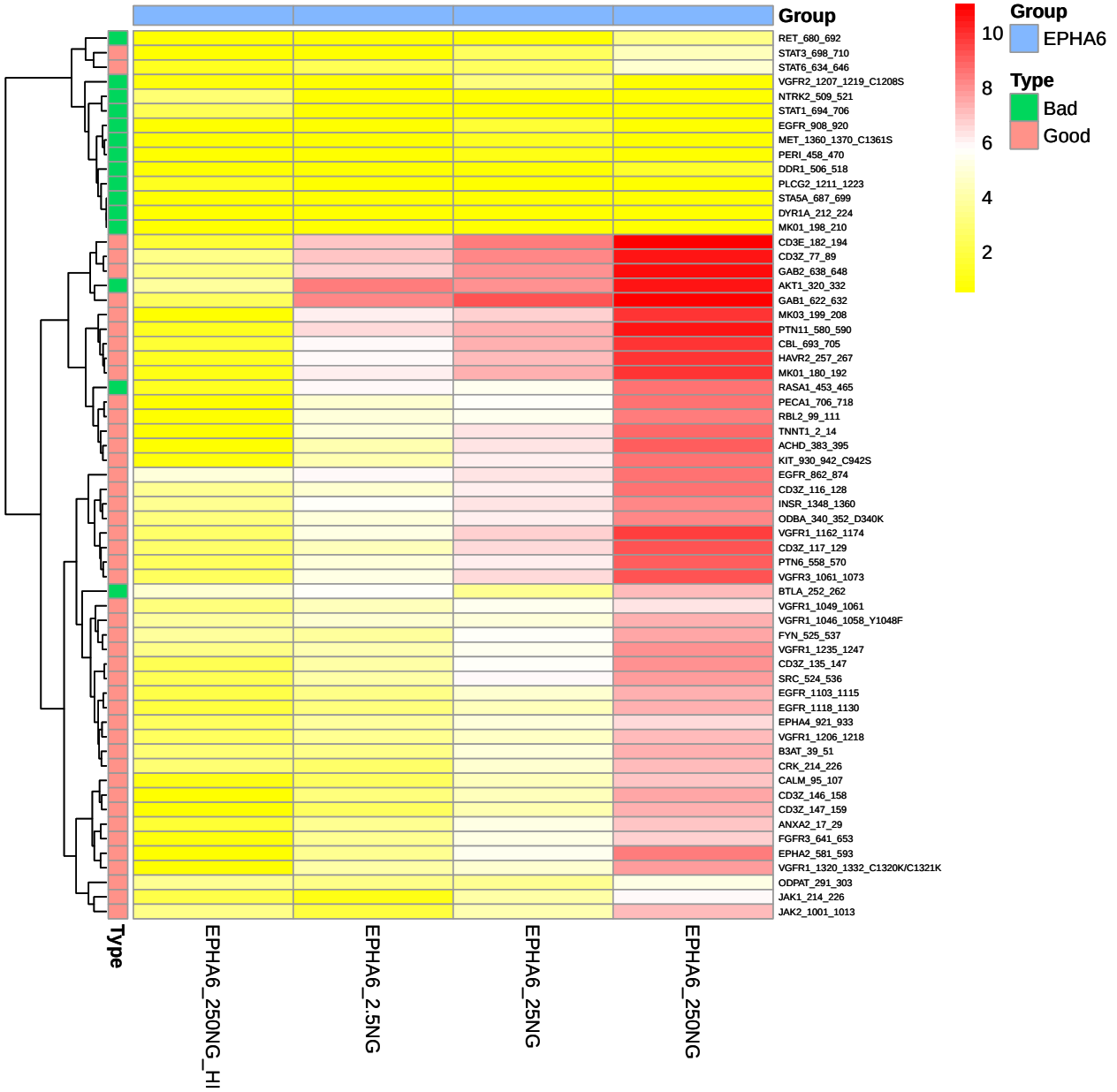
```
  filter(Group == "EPHA6", Peptide %in% good_peptides) %>%
  select(Peptide,SampleName, slope) %>%
  pivot_wider(names_from = SampleName, values_from = slope) %>%
  mutate(low = EPHA6_2.5NG - EPHA6_250NG_HI,
         mid = EPHA6_25NG - EPHA6_2.5NG,
         high = EPHA6_250NG - EPHA6_25NG) %>%
  select(Peptide, mid, high) %>%
  mutate(Avg = rowSums(across(where(is.numeric)) > 0, na.rm = T)) -> data_modeled$signal_direction_EPHA


EPHA6_top_peptides <- filter(data_modeled$signal_direction_EPHA6, Avg == 2) %>% pull(Peptide)
EPHA6_bad_peptides <- filter(data_modeled$signal_direction_EPHA6, Avg != 2) %>% pull(Peptide)



data_modeled$scaled %>%
  filter(Group == "EPHA6", Peptide %in% good_peptides) %>%
  mutate(Type = ifelse(Peptide %in% EPHA6_top_peptides, "Good", "Bad"))  %>%
  ggplot(aes(SampleName, slope, group = Peptide, color = Type),  alpha = 1/3) +
  geom_line() +
  geom_point()
```




```
data_modeled$scaled %>%
  filter(Group == "EPHA6", Peptide %in% good_peptides) %>%
  mutate(Type = ifelse(Peptide %in% EPHA6_top_peptides, "Good", "Bad")) %>%
  ggplot(aes(SampleName, slope, group = Peptide, color = Type), alpha = 1/3) +
  geom_line() +
  geom_point() +
  facet_wrap(~Type) +
  theme(axis.text.x = element_text(angle = 30))
```



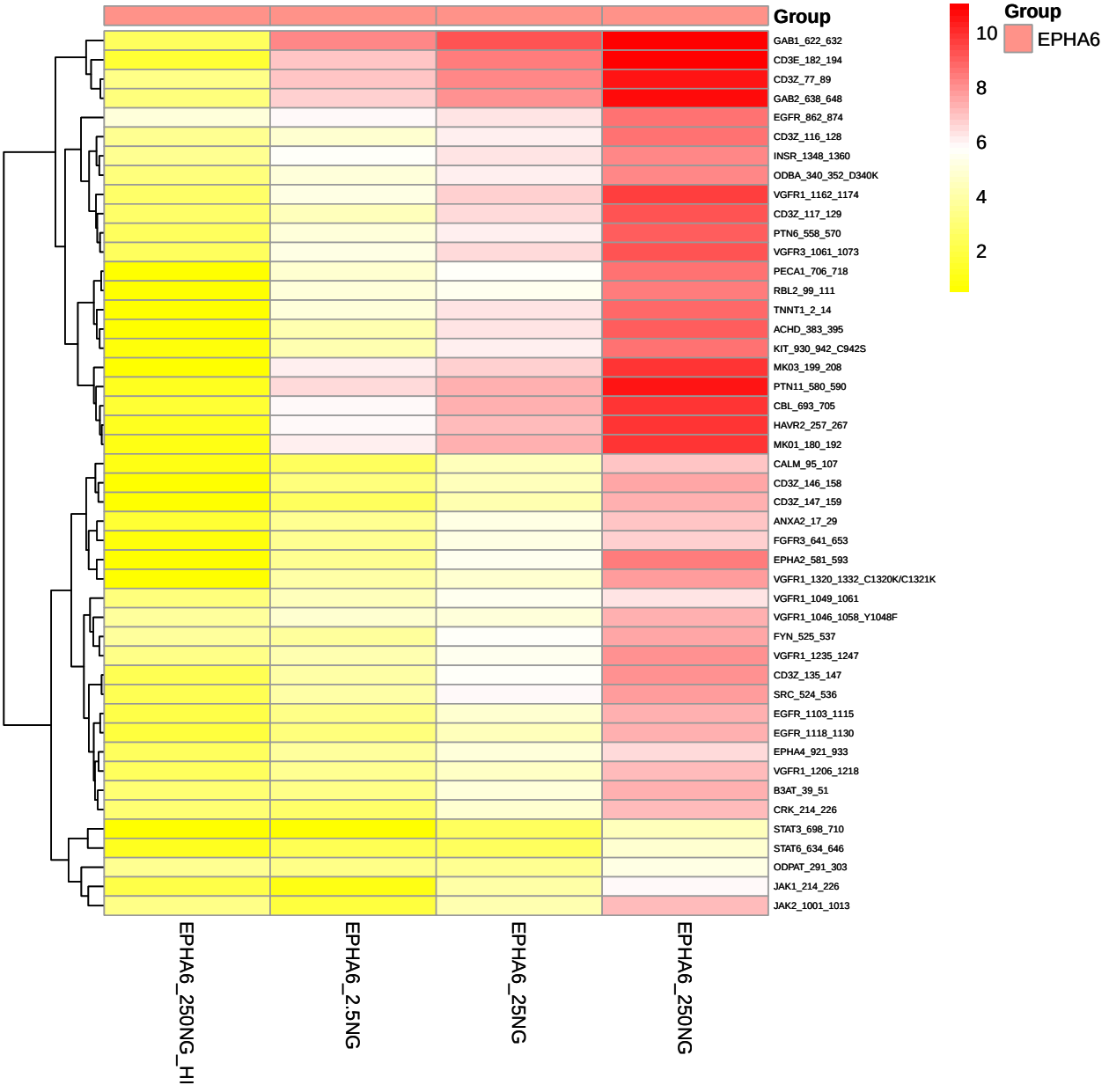

```
tibble(
  Peptide = good_peptides
) %>%
  mutate(Type = ifelse(Peptide %in% EPHA6_top_peptides, "Good", "Bad")) %>%
  column_to_rownames("Peptide") -> pep_annotations_EPHA6


krsa_heatmap(data_modeled$scaled %>% filter(Group == "EPHA6"),

             good_peptides, scale = "none", cluster_col = F, annotation_row = pep_annotations_EPHA6)
```




```
# Define Groups to be compared
comparisons <- list(Comp1 = c("EPHA6_250NG", "EPHA6_250NG_HI"))


# This function calculates log2 fold change values between the defined groups
# The byChip argument lets you calculates the log2 fold change the results within each chip
krsa_group_diff(data_modeled$scaled %>% mutate(Group = SampleName),
                comparisons$Comp1, EPHA6_top_peptides, byChip = F) -> diff_df
```

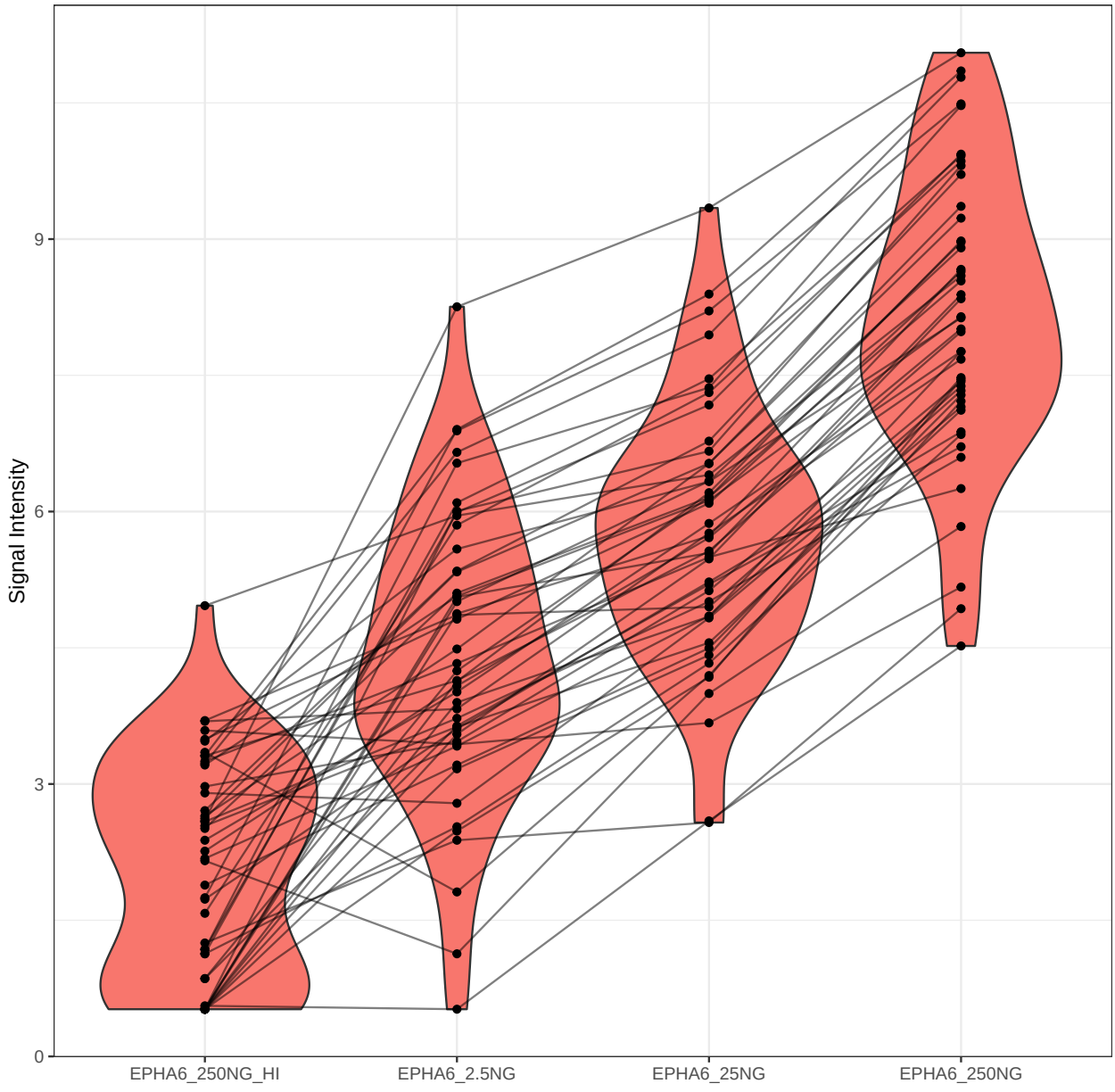
EPHA6 (Heatmap)

```
# generates a heatmap using the selected groups and peptides
krsa_heatmap(data_modeled$scaled %>% filter(Group == "EPHA6"),
             peptides = EPHA6_top_peptides,
             scale = "none",
             cluster_col = F
             )
```

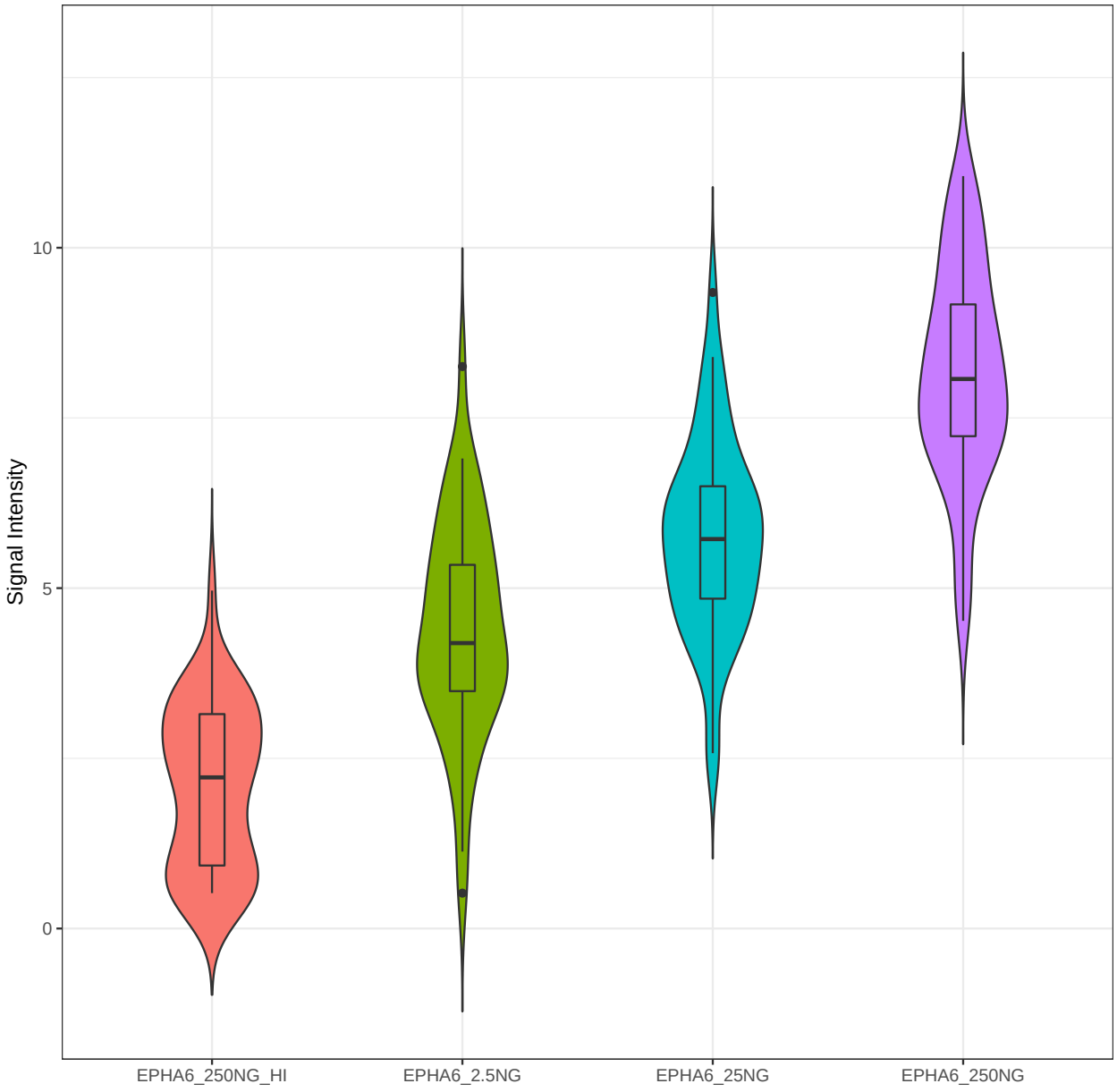


EPHA6 (Violin Plot)

```
# generates a violin plot using the selected groups and peptides
krsa_violin_plot(data_modeled$scaled %>% filter(Group == "EPHA6"),
                 EPHA6_top_peptides, facet = F,
                   facet_factor = "Group", groups = "EPHA6")
```



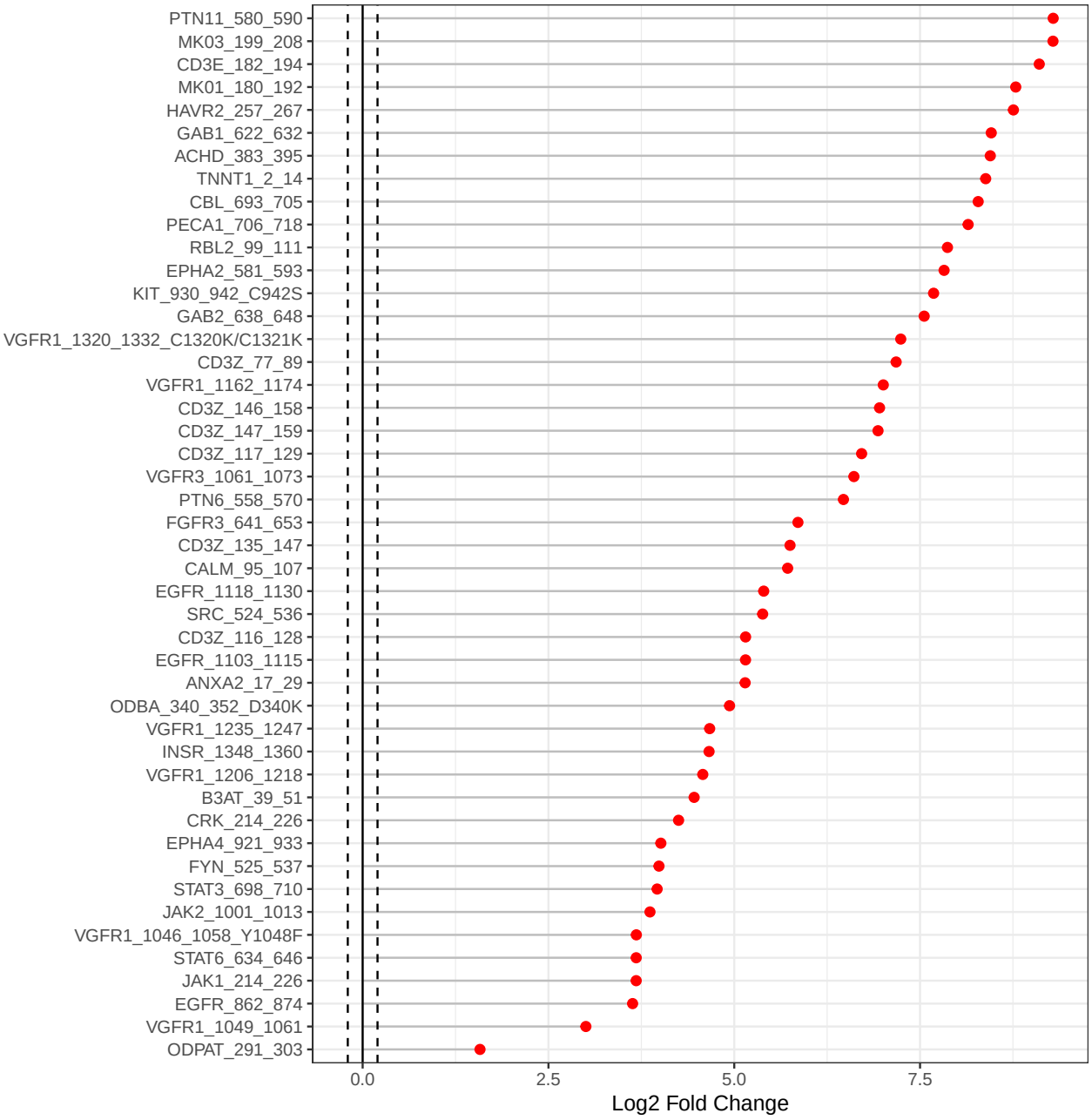

```
# generate a grouped violin/boxplot plot using the selected groups and peptides with more options, like
krsa_violin_plot_grouped(data_modeled$scaled %>% filter(Group == "EPHA6")  %>%
                  mutate(Group = SampleName),
                          EPHA6_top_peptides, comparisons,
                          dots = F,
                 test = F, avg_line = F)
```


Signal Intensity
10
5
0
EPHA6_250NG_HI
EPHA6_2.5NG
EPHA6_25NG
EPHA6_250NG

EPHA6 (Waterfall Plot)

This waterfall represents the log2 fold changes between the two groups at each peptide.

```
# generates a waterfall of the log2 fold change values for the selected peptide (top peptides)
krsa_waterfall(diff_df, lfc_thr = 0.2, byChip = F)
```

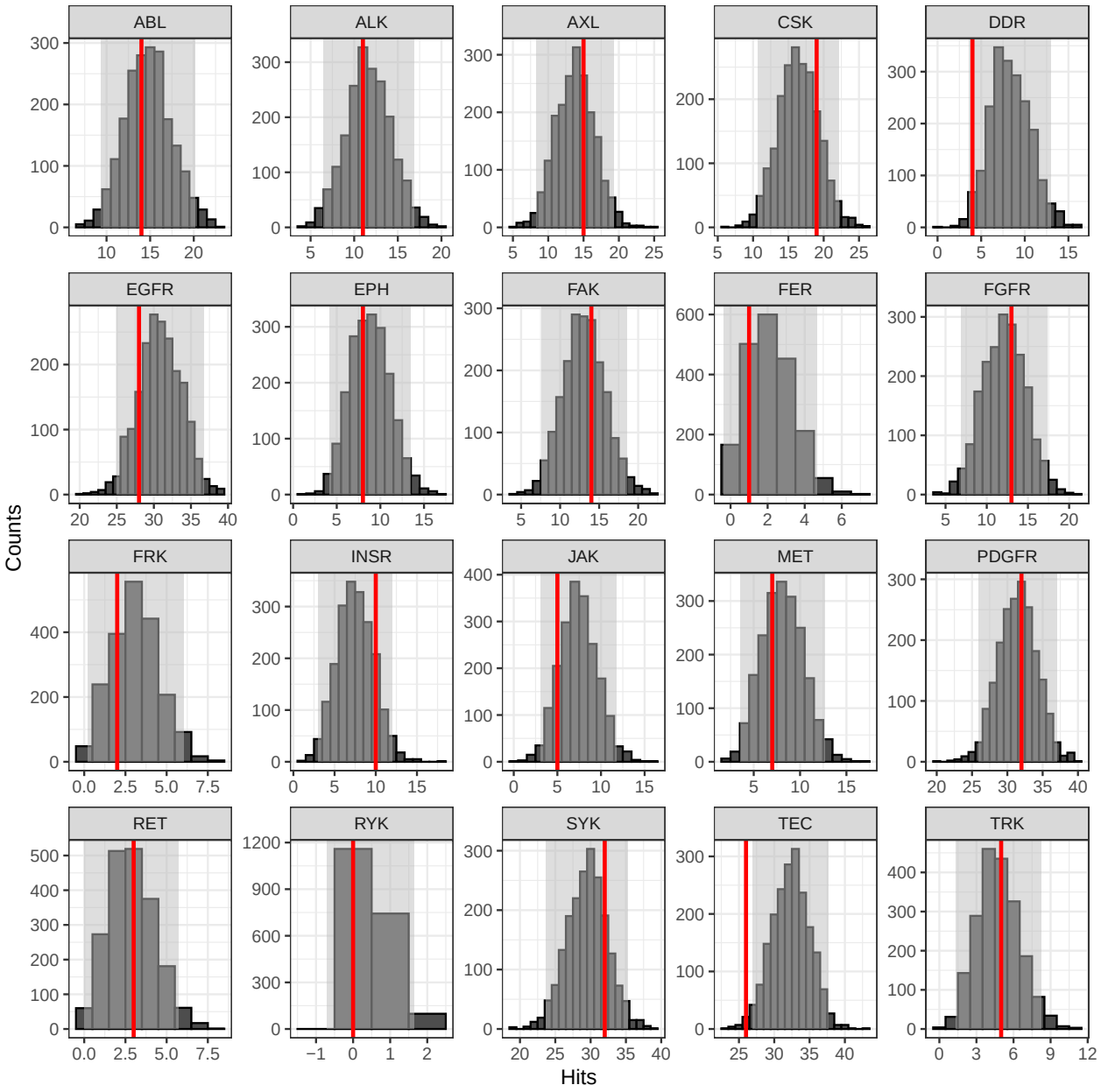


EPHA6 (Upstream Kinase Analysis)

```
# run the KRSA function to do the random sampling analysis, set seed that can be used later to reproduc
krsa(EPHA6_top_peptides, return_count = T, seed = 123, itr = 2000,
     map_file = KRSA_file, cov_file = chipCov) -> fin

# View the Z score table
kable(head(fin$KRSA_Table,25), digits = 3)
```

Kinase	Observed	SamplingAvg	SD	Z
TEC	26	32.331	2.697	-2.348
DDR	4	8.340	2.339	-1.856
JAK	5	7.416	2.167	-1.115
INSR	10	7.529	2.250	1.098
EGFR	28	30.838	2.937	-0.966
FER	1	2.129	1.253	-0.901
CSK	19	16.430	2.877	0.894
SYK	32	29.513	2.886	0.862
RYK	0	0.470	0.589	-0.797
FRK	2	3.102	1.455	-0.757
SRC	39	37.272	2.337	0.740
VEGFR	9	7.526	2.198	0.671
ACK	0	0.230	0.421	-0.547
SEV	0	0.228	0.420	-0.543
MET	7	8.118	2.281	-0.490
AXL	15	13.834	2.759	0.422
EPH	8	8.821	2.323	-0.353
FAK	14	13.060	2.737	0.343
FGFR	13	12.124	2.615	0.335
ABL	14	14.774	2.703	-0.286
ALK	11	11.652	2.591	-0.252
PDGFR	32	31.458	2.767	0.196
TRK	5	4.761	1.735	0.138
RET	3	2.877	1.425	0.086

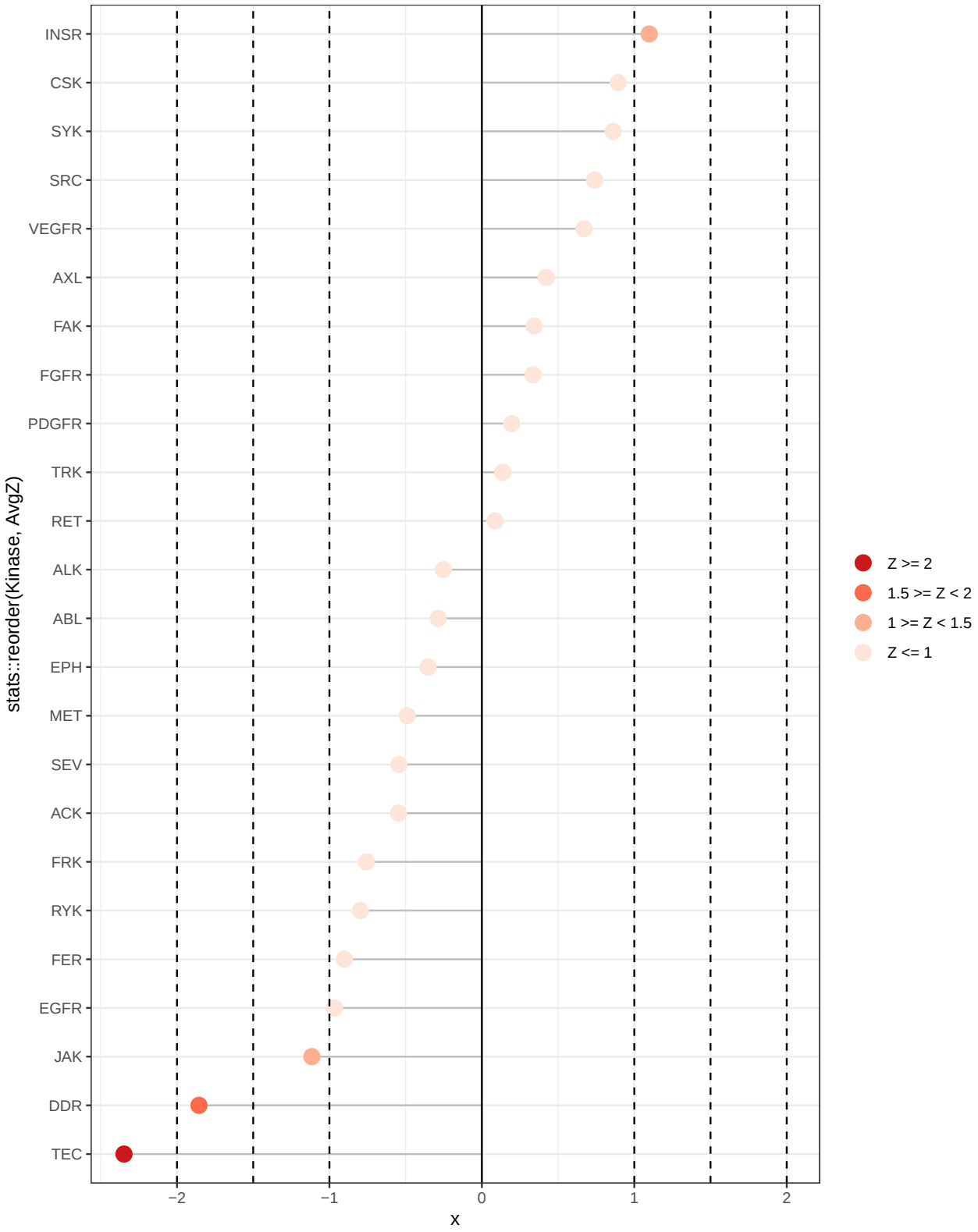
```
# to save file
#fin$KRSA_Table %>% write_delim("output/Z_Scores_tables/acrossChip_KRSA_FullTable_comp1.txt", delim = "

# find top and bottom kinases
bothways <- c(pull(head(fin$KRSA_Table, 10), Kinase), pull(tail(fin$KRSA_Table, 10), Kinase))

# Use these kinase to generate histogram plots for each selected kinase
krsa_histogram_plot(fin$KRSA_Table, fin$count_mtx, bothways)
```


ABL
ALK
AXL
CSK
DDR
EGFR
EPH
FAK
FER
FGFR
FRK
INSR
JAK
MET
PDGFR
RET
RYK
SYK
TEC
TRK
Counts
Hits

EPHA6 (Z Scores Plot)

We will plot the individual and averaged Z scores using both the across and within chip analyses.

```
krsa_zscores_plot(fin$KRSA_Table)
```

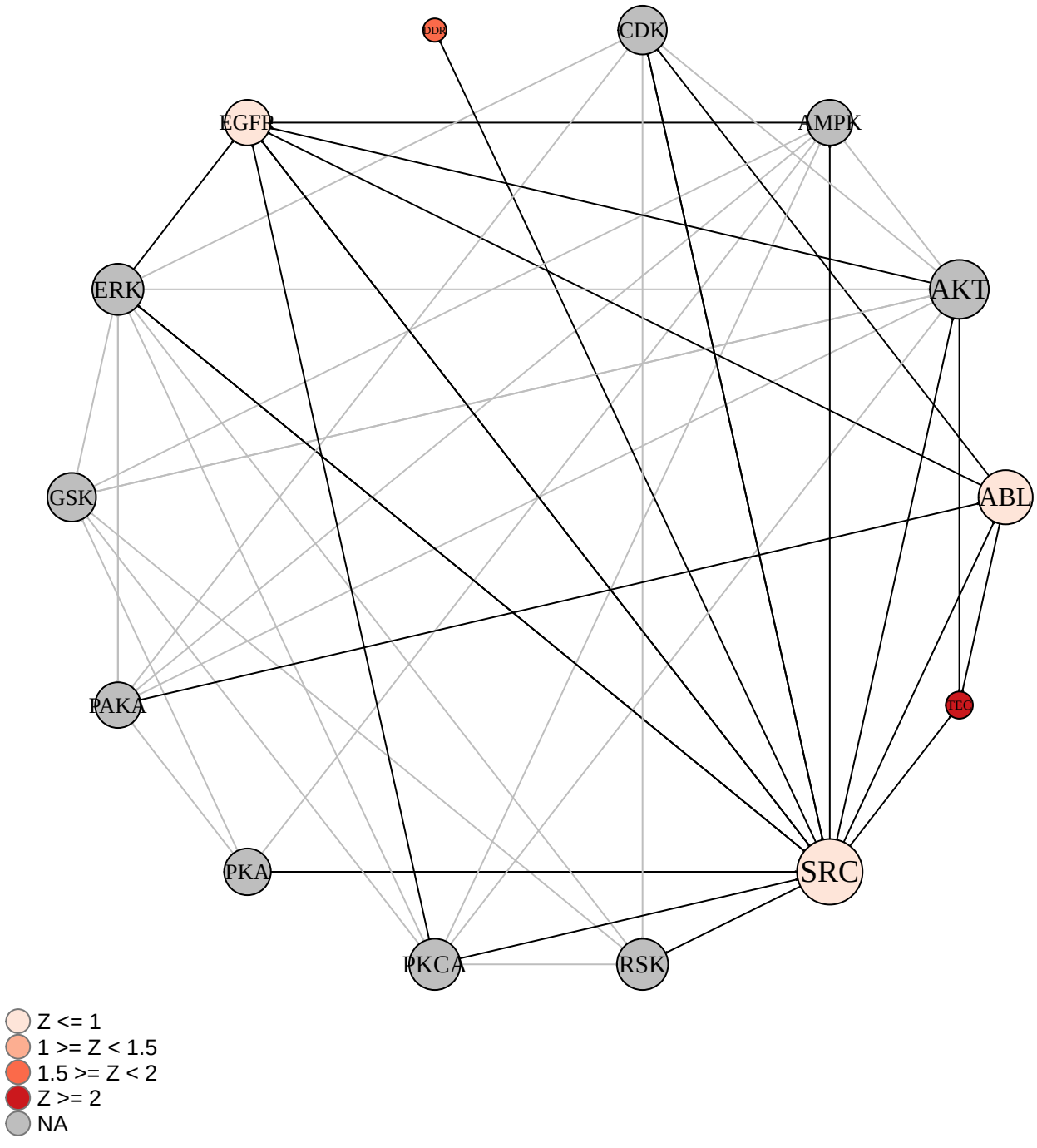

INSR
CSK
SYK
SRC
VEGFR
AXL
FAK
FGFR
PDGFR
TRK
RET
ALK
ABL
EPH
MET
SEV
ACK
FRK
RYK
FER
EGFR
JAK
DDR
TEC
stats::reorder(Kinase, AvgZ)
−2
−1
0
1
2
x
Z >= 2
1.5 >= Z < 2
1 >= Z < 1.5
Z <= 1

```
#
```

EPHA6 (Ball Model Network)

We will view the ball model network function, to generate a model representing the protein-protein interactions between kinases

```
# Plot the network ball model
krsa_ball_model(c("EPH"), fin$KRSA_Table %>% mutate(AvgZ = Z), 15, 2.5, 4.8)
```




```
installed.packages()[names(sessionInfo()$otherPkgs), "Version"]
```

```
#>     furrr    future        gt   forcats   stringr     dplyr     purrr     readr
#>   "0.2.1"  "1.22.1"   "0.2.2"   "0.5.0"   "1.4.0"   "1.0.4"   "0.3.4"   "1.4.0"
#>     tidyr    tibble   ggplot2 tidyverse     knitr      KRSA
#>   "1.1.2"   "3.1.2"   "3.3.3"   "1.3.0"    "1.30"  "0.9.44"
```

Session Info