

# Cancer

Lieven Clement

statOmics, Ghent University (<https://statomics.github.io>)

## Contents

<b>1</b>	<b>Background</b>	<b>1</b>
<b>2</b>	<b>Data</b>	<b>1</b>
2.1	Data exploration . . . . .	3
<b>3</b>	<b>Preprocessing</b>	<b>3</b>
3.1	Log transform the data . . . . .	4
3.2	Filtering . . . . .	4
3.3	Normalize the data using median centering . . . . .	4
3.4	Explore normalized data . . . . .	4
3.5	Summarization to protein level . . . . .	6
<b>4</b>	<b>Data Analysis</b>	<b>7</b>
4.1	Estimation . . . . .	7
4.2	Inference . . . . .	7
4.3	Plots . . . . .	8
<b>5</b>	<b>Session Info</b>	<b>222</b>

This is part of the online course [Proteomics Data Analysis \(PDA\)](#)

## 1 Background

Eighteen Estrogen Receptor Positive Breast cancer tissues from from patients treated with tamoxifen upon recurrence have been assessed in a proteomics study. Nine patients had a good outcome (OR) and the other nine had a poor outcome (PD). The proteomes have been assessed using an LTQ-Orbitrap and the thermo output .RAW files were searched with MaxQuant (version 1.4.1.2) against the human proteome database (FASTA version 2012-09, human canonical proteome).

## 2 Data

We first import the data from peptide.txt file. This is the file containing your peptide-level intensities. For a MaxQuant search [6], this peptide.txt file can be found by default in the “path\_to\_raw\_files/combined/txt/” folder from the MaxQuant output, with “path\_to\_raw\_files” the folder where the raw files were saved.

We generate the object peptideFile with the path to the peptide.txt file. Using the `grepCols` function, we find the columns that contain the expression data of the peptide in the peptide.txt file.

```
library(tidyverse)
library(limma)
library(QFeatures)
```

```
library(msqrob2)
library(plotly)

peptidesFile <- "https://raw.githubusercontent.com/statOmics/PDA22GTPB/data/quantification/cancer/peptidesFile.csv"

ecols <- grep(
  "Intensity\\.\\.",
  names(read.delim(peptidesFile))
)
```

Next, we read the data and store it in QFeatures object

```
pe <- readQFeatures(
  assayData = read.delim(peptidesFile),
  fnames = 1,
  quantCols = ecols,
  name = "peptideRaw")
```

```
## Checking arguments.
## Loading data as a 'SummarizedExperiment' object.
## Formatting sample annotations (colData).
## Formatting data as a 'QFeatures' object.
```

The QFeatures object pe currently contains a single assay, named peptideRaw.

We extract the column names from the peptideRaw assay and see that this contains information about the prognosis.

```
colnames(pe[["peptideRaw"]])

## [1] "Intensity.OR.01" "Intensity.OR.04" "Intensity.OR.07" "Intensity.OR.09"
## [5] "Intensity.OR.10" "Intensity.OR.13" "Intensity.OR.20" "Intensity.OR.23"
## [9] "Intensity.OR.25" "Intensity.PD.02" "Intensity.PD.03" "Intensity.PD.04"
## [13] "Intensity.PD.06" "Intensity.PD.07" "Intensity.PD.08" "Intensity.PD.09"
## [17] "Intensity.PD.10" "Intensity.PD.11"
```

We rename the colnames by dropping the “Intensity.” from the name.

```
(newNames <- sub(
  pattern = "Intensity\\.\\.",
  replacement = "",
  colnames(pe[["peptideRaw"]]))
)

## [1] "OR.01" "OR.04" "OR.07" "OR.09" "OR.10" "OR.13" "OR.20" "OR.23" "OR.25"
## [10] "PD.02" "PD.03" "PD.04" "PD.06" "PD.07" "PD.08" "PD.09" "PD.10" "PD.11"

pe <- renameColname(pe,
  i = "peptideRaw",
  newNames)
pe <- renamePrimary(pe, newNames)
colnames(pe[["peptideRaw"]])

## [1] "OR.01" "OR.04" "OR.07" "OR.09" "OR.10" "OR.13" "OR.20" "OR.23" "OR.25"
## [10] "PD.02" "PD.03" "PD.04" "PD.06" "PD.07" "PD.08" "PD.09" "PD.10" "PD.11"
```

In the following code chunk, we add the prognosis of the patients that we can read in the raw file name to the colData.

```
colData(pe)$prognosis <-
  colnames(pe[["peptideRaw"]]) %>%
  substr(start = 1, stop = 2) %>%
  as.factor
colData(pe)$prognosis
```

```
## [1] OR OR OR OR OR OR OR OR OR OR PD PD PD PD PD PD PD PD PD PD
## Levels: OR PD
```

We calculate how many non zero intensities we have per peptide and this will be useful for filtering.

```
rowData(pe[["peptideRaw"]])$nNonZero <- rowSums(assay(pe[["peptideRaw"]]) > 0)
```

Peptides with zero intensities are missing peptides and should be represent with a NA value rather than 0.

```
pe <- zeroIsNA(pe, "peptideRaw") # convert 0 to NA
```

Look at the column names of the data to know the variables that you can use for filtering.

```
pe[["peptideRaw"]] %>% rowData %>% names
```

```
## [1] "Sequence"          "Proteins"          "Leading.razor.protein"
## [4] "Gene.names"        "Protein.names"     "Unique..Groups."
## [7] "Unique..Proteins." "Charges"           "PEP"
## [10] "Score"             "Slice.Average"     "Slice.Std..Dev."
## [13] "Slice.1"           "Unique.Slice.Average" "Unique.Slice.Std..Dev."
## [16] "Unique.Slice.1"     "Experiment.OR.01"   "Experiment.OR.04"
## [19] "Experiment.OR.07"   "Experiment.OR.09"   "Experiment.OR.10"
## [22] "Experiment.OR.13"   "Experiment.OR.20"   "Experiment.OR.23"
## [25] "Experiment.OR.25"   "Experiment.PD.02"   "Experiment.PD.03"
## [28] "Experiment.PD.04"   "Experiment.PD.06"   "Experiment.PD.07"
## [31] "Experiment.PD.08"   "Experiment.PD.09"   "Experiment.PD.10"
## [34] "Experiment.PD.11"   "Intensity"          "Reverse"
## [37] "Contaminant"        "id"                 "Protein.group.IDs"
## [40] "Mod..peptide.IDs"   "Evidence.IDs"       "MS.MS.IDs"
## [43] "Best.MS.MS"         "Oxidation..M..site.IDs" "nNonZero"
```

So we will filter on the “Reverse”, “Contaminant” and “nNonZero” column.

## 2.1 Data exploration

47% of all peptide intensities are missing and for some peptides we do not even measure a signal in any sample.

## 3 Preprocessing

This section performs preprocessing for the peptide data. This include

- log transformation,
- filtering and
- summarisation of the data.

### 3.1 Log transform the data

```
pe <- logTransform(pe, base = 2, i = "peptideRaw", name = "peptideLog")
```

### 3.2 Filtering

1. Handling overlapping protein groups

In our approach a peptide can map to multiple proteins, as long as there is none of these proteins present in a smaller subgroup.

```
pe <- filterFeatures(pe, ~ Proteins %in% smallestUniqueGroups(rowData(pe[["peptideLog"]])$Proteins))  
  
## 'Proteins' found in 2 out of 2 assay(s)
```

2. Remove reverse sequences (decoys) and contaminants

We now remove the contaminants and peptides that map to decoy sequences.

```
pe <- filterFeatures(pe, ~Reverse != "+")  
  
## 'Reverse' found in 2 out of 2 assay(s)  
pe <- filterFeatures(pe, ~Contaminant != "+")  
  
## 'Contaminant' found in 2 out of 2 assay(s)
```

3. Drop peptides that were only identified in one sample

We keep peptides that were observed at last twice.

```
pe <- filterFeatures(pe, ~ nNonZero >=2)  
  
## 'nNonZero' found in 2 out of 2 assay(s)  
nrow(pe[["peptideLog"]])  
  
## [1] 26696
```

We keep 26696 peptides upon filtering.

### 3.3 Normalize the data using median centering

We normalize the data by subtracting the sample median from every intensity for peptide  $p$  in a sample  $i$ :

$$y_{ip}^{\text{norm}} = y_{ip} - \hat{\mu}_i$$

with  $\hat{\mu}_i$  the median intensity over all observed peptides in sample  $i$ .

```
pe <- normalize(pe,  
               i = "peptideLog",  
               name = "peptideNorm",  
               method = "center.median")
```

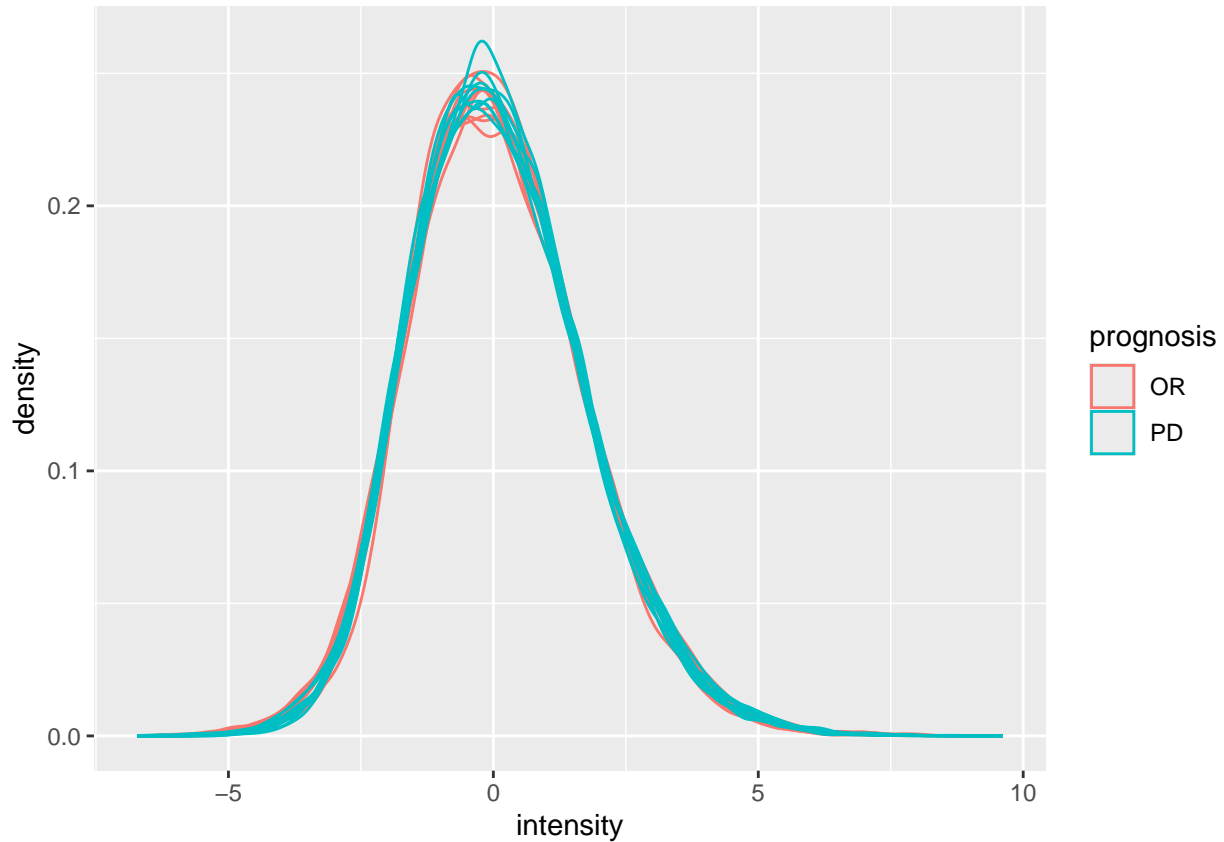
### 3.4 Explore normalized data

Upon the normalisation the density curves are nicely registered

```
pe[["peptideNorm"]] %>%  
  assay %>%  
  as.data.frame() %>%
```

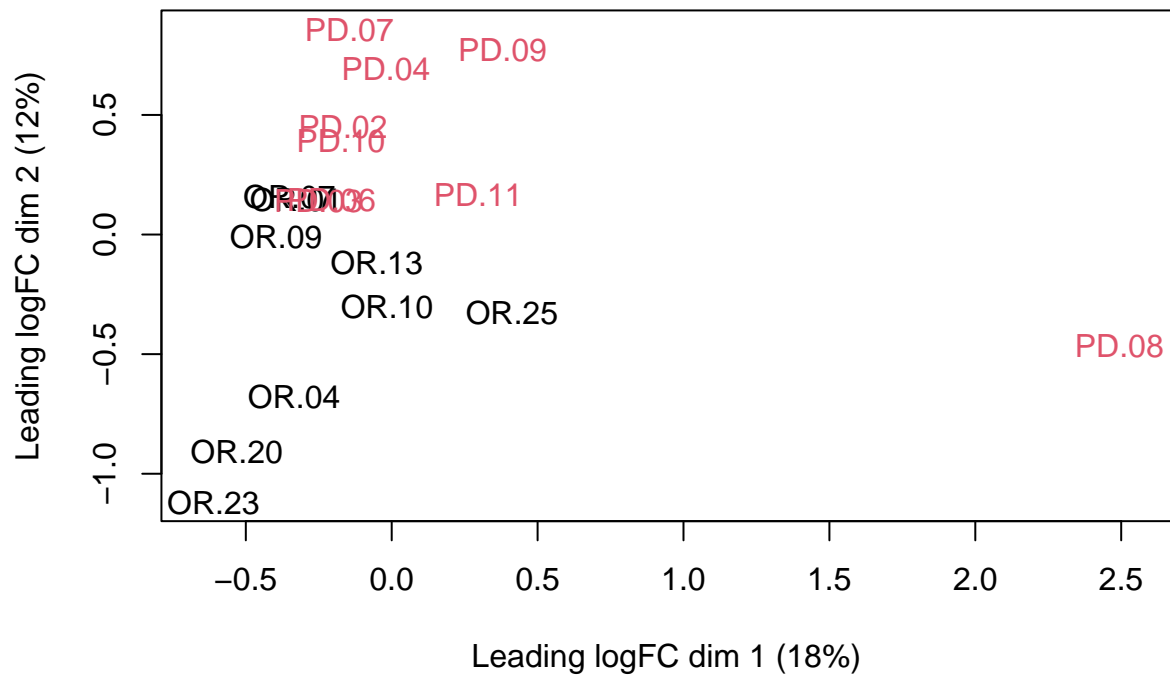
```
gather(sample, intensity) %>%
mutate(prognosis = colData(pe)[sample,"prognosis"]) %>%
ggplot(aes(x = intensity,group = sample,color = prognosis)) +
  geom_density()
```

```
## Warning: Removed 188395 rows containing non-finite outside the scale range
## (`stat_density()`).
```



We can visualize our data using a Multi Dimensional Scaling plot, eg. as provided by the `limma` package.

```
pe[["peptideNorm"]] %>%
  assay %>%
  limma::plotMDS(col = as.numeric(colData(pe)$prognosis))
```



The first axis in the plot is showing the leading log fold changes (differences on the log scale) between the samples. We observe one outlying sample. In the second dimension we observe a separation according to prognosis.

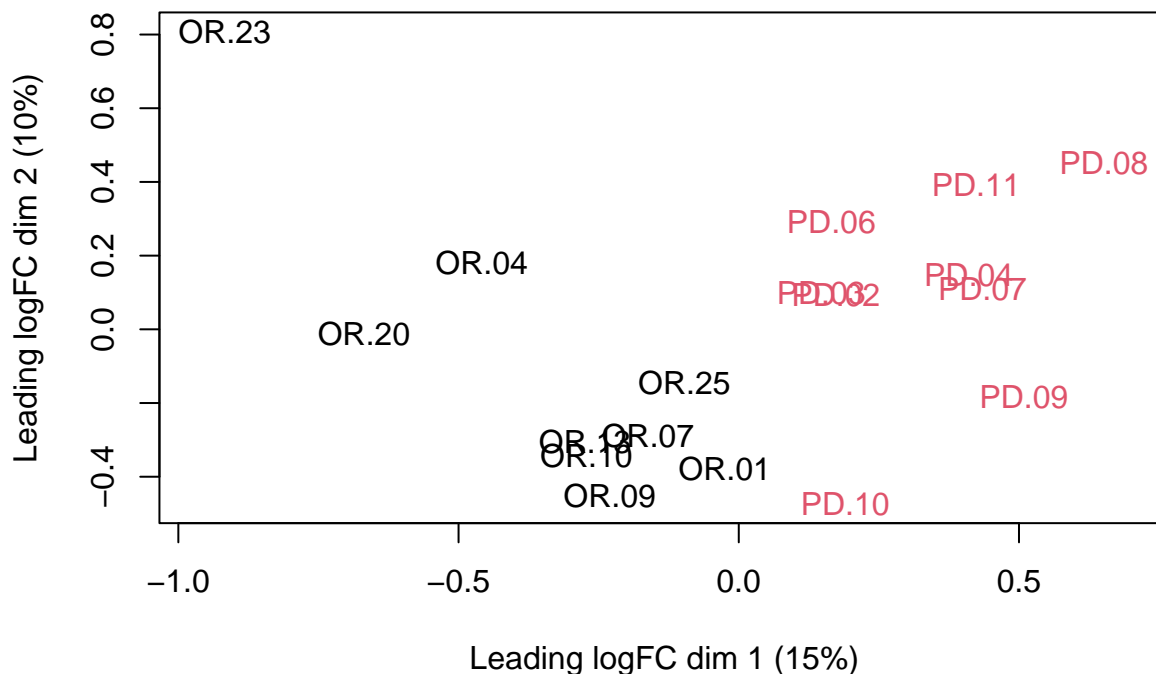
### 3.5 Summarization to protein level

- By default robust summarization is used: `fun = MsCoreUtils::robustSummary()`

```
pe <- aggregateFeatures(pe,
  i = "peptideNorm",
  fcol = "Proteins",
  na.rm = TRUE,
  name = "protein")
```

```
## Your quantitative and row data contain missing values. Please read the
## relevant section(s) in the aggregateFeatures manual page regarding the
## effects of missing values on data aggregation.
```

```
plotMDS(assay(pe[["protein"]]), col = as.numeric(colData(pe)$prognosis))
```



Note that the samples upon robust summarisation show a separation according to the prognosis.

## 4 Data Analysis

### 4.1 Estimation

We model the protein level expression values using `msqrob`. By default `msqrob2` estimates the model parameters using robust regression.

We will model the data with a different group mean. The group is incoded in the variable `prognosis` of the `colData`. We can specify this model by using a formula with the factor condition as its predictor: `formula = ~prognosis`.

Note, that a formula always starts with a symbol `~`.

```
pe <- msqrob(object = pe, i = "protein", formula = ~prognosis)
```

### 4.2 Inference

First, we extract the parameter names of the model by looking at the first model. The models are stored in the row data of the assay under the default name `msqrobModels`.

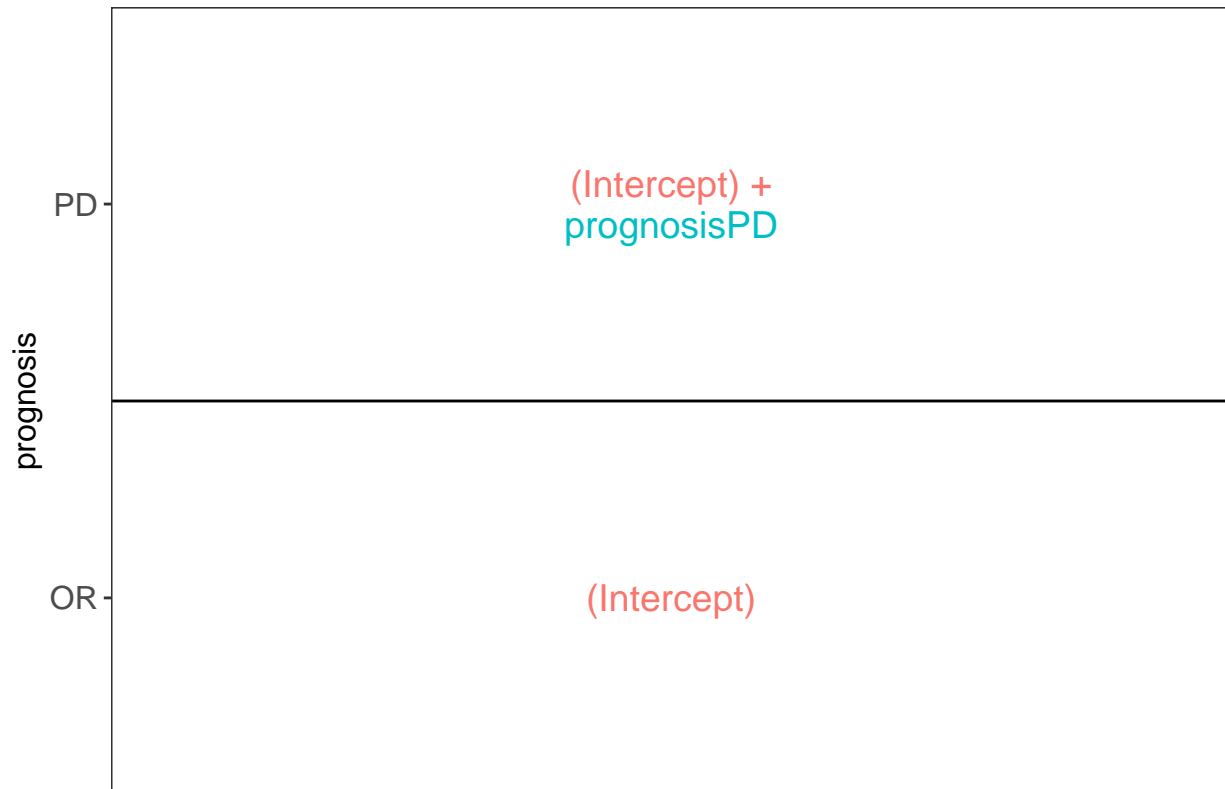
```
getCoef(rowData(pe[["protein"]])$msqrobModels[[1]])
```

```
## (Intercept) prognosisPD
## -1.1185468 0.4007461
```

We can also explore the design of the model that we specified using the the package `ExploreModelMatrix`

```
library(ExploreModelMatrix)
VisualizeDesign(colData(pe), ~prognosis)$plotlist
```

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



Spike-in condition A is the reference class. So the mean log2 expression for samples from good prognosis (OR) is '(Intercept)'. The mean log2 expression for samples from poor prognosis (PD) is '(Intercept)+prognosisPD'. Hence, the average log2 fold change between prognosis PD and prognosis OR is modelled using the parameter 'conditionPD'. Thus, we assess the contrast 'conditionPD = 0' with our statistical test.

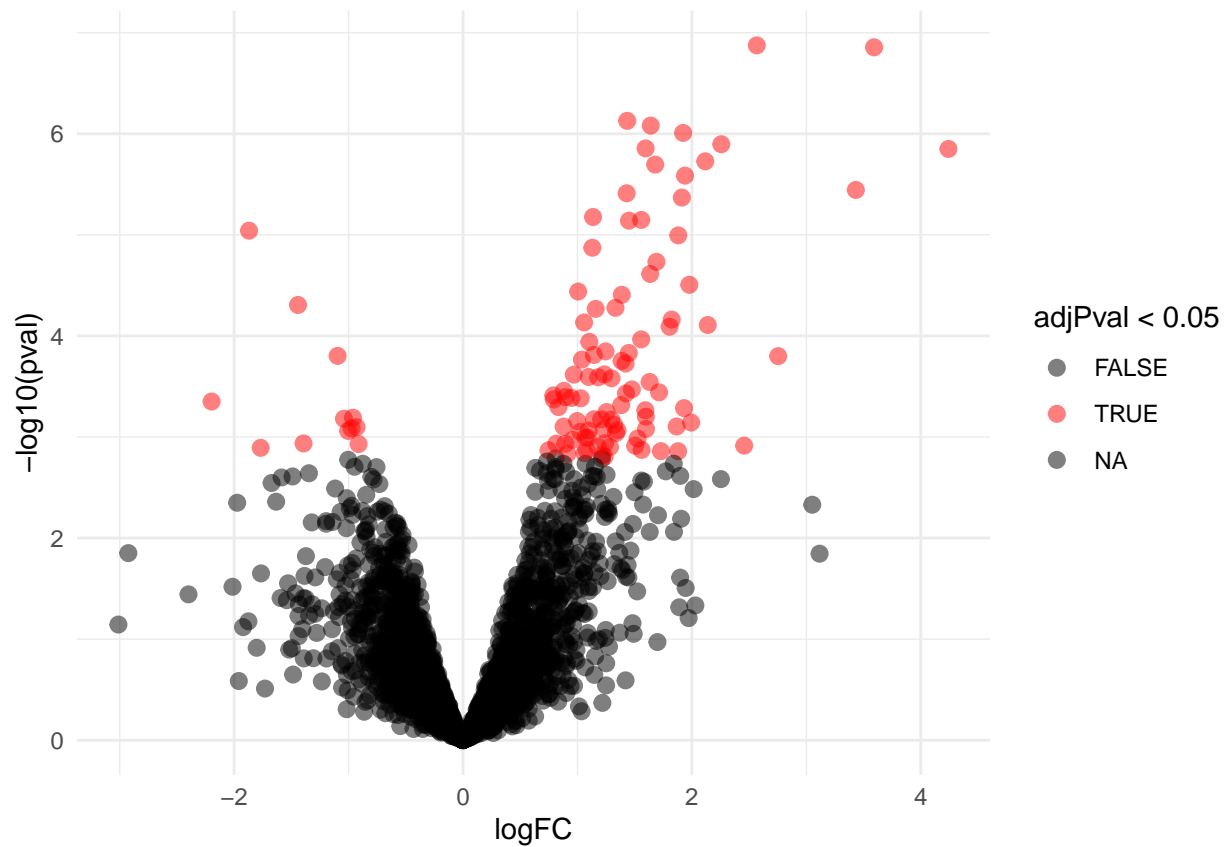
```
L <- makeContrast("prognosisPD=0", parameterNames = c("prognosisPD"))
pe <- hypothesisTest(object = pe, i = "protein", contrast = L)
```

## 4.3 Plots

### 4.3.1 Volcano-plot

```
volcano <- ggplot(rowData(pe[["protein"]])$prognosisPD,
  aes(x = logFC, y = -log10(pval), color = adjPval < 0.05)) +
  geom_point(cex = 2.5) +
  scale_color_manual(values = alpha(c("black", "red"), 0.5)) + theme_minimal()
volcano
```



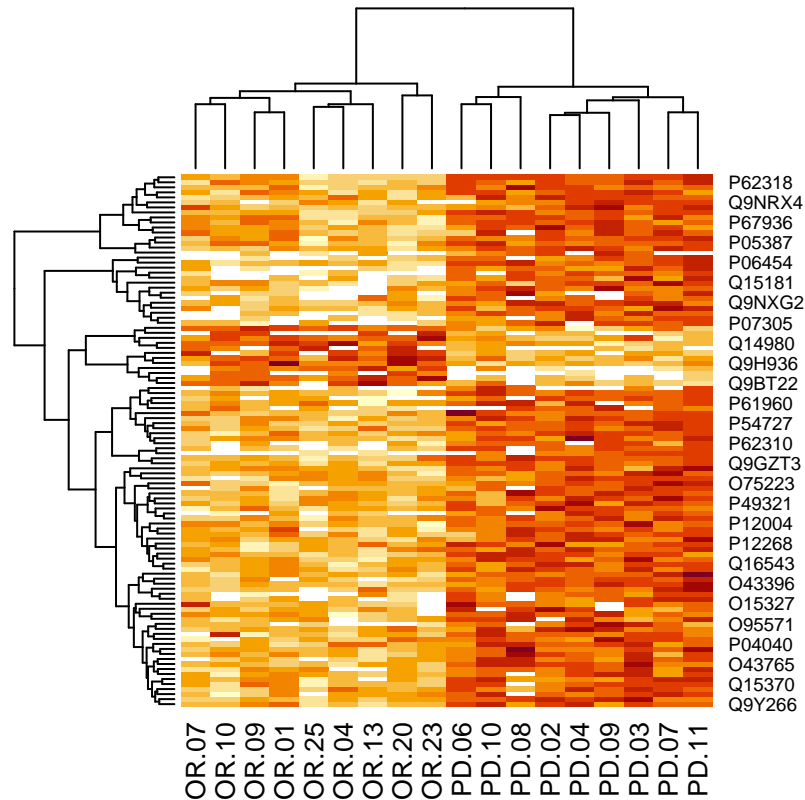


Note, that 106 proteins are found to be differentially abundant.

#### 4.3.2 Heatmap

Note, that we also order the sigNames according to statistical significance.

```
sigNames <- rowData(pe[["protein"]])$prognosisPD %>%
  rownames_to_column("protein") %>%
  arrange(pval) %>%
  filter(adjPval<0.05) %>%
  pull(protein)
heatmap(assay(pe[["protein"]])[sigNames, ])
```



### 4.3.3 Detail plots

We make detail plots for the top 10 proteins to restrict the number of detail plots.

```
for (protName in sigNames)
#for (protName in orderProt[1:10])
{
  pePlot <- pe[protName, , c("peptideNorm","protein")]
  pePlotDf <- data.frame(longFormat(pePlot))
  pePlotDf$assay <- factor(pePlotDf$assay,
                          levels = c("peptideNorm", "protein"))
  pePlotDf$prognosis <- as.factor(colData(pePlot)[pePlotDf$colname, "prognosis"])

  # plotting
  p1 <- ggplot(data = pePlotDf,
               aes(x = colname, y = value, group = rowname)) +
    geom_line() +
    geom_point() +
    theme(axis.text.x = element_text(angle = 70, hjust = 1, vjust = 0.5)) +
    facet_grid(~assay) +
    ggtitle(protName)
  print(p1)

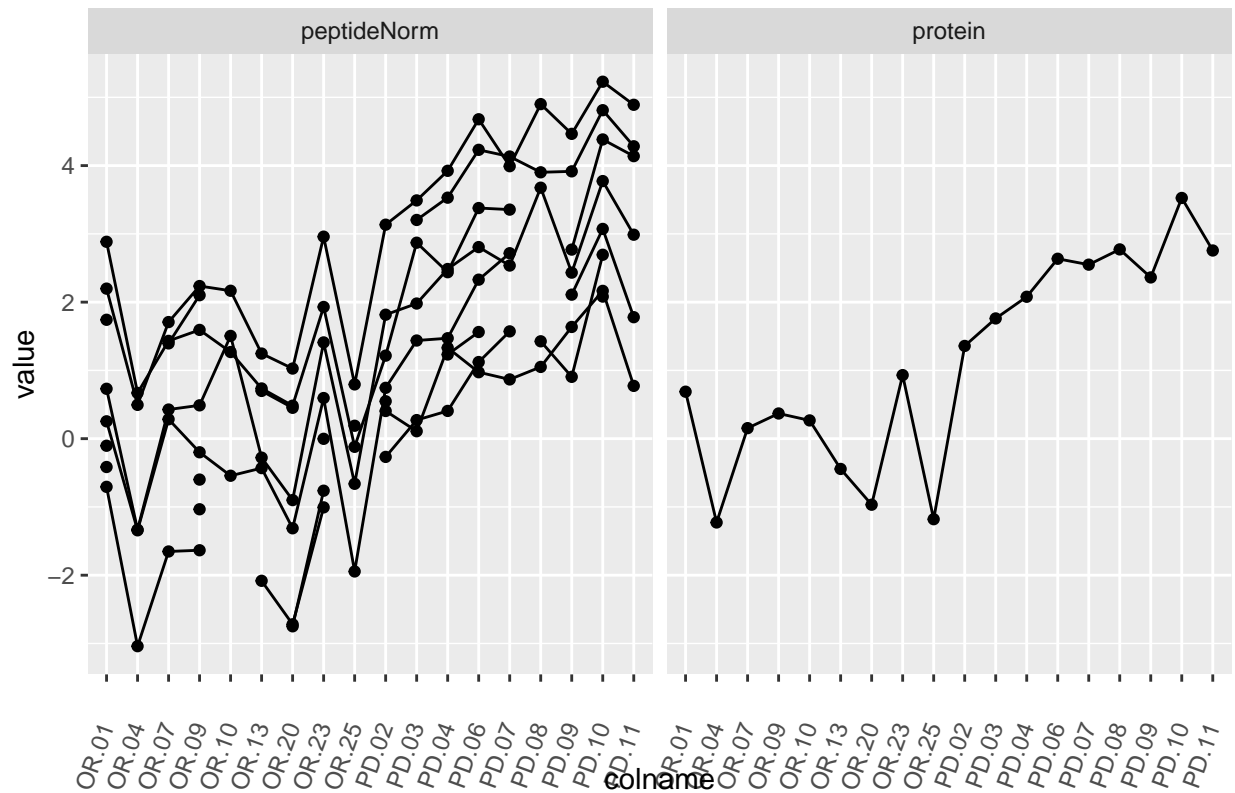
  # plotting 2
  p2 <- ggplot(pePlotDf, aes(x = colname, y = value, fill = prognosis)) +
    geom_boxplot(outlier.shape = NA) +
```

```

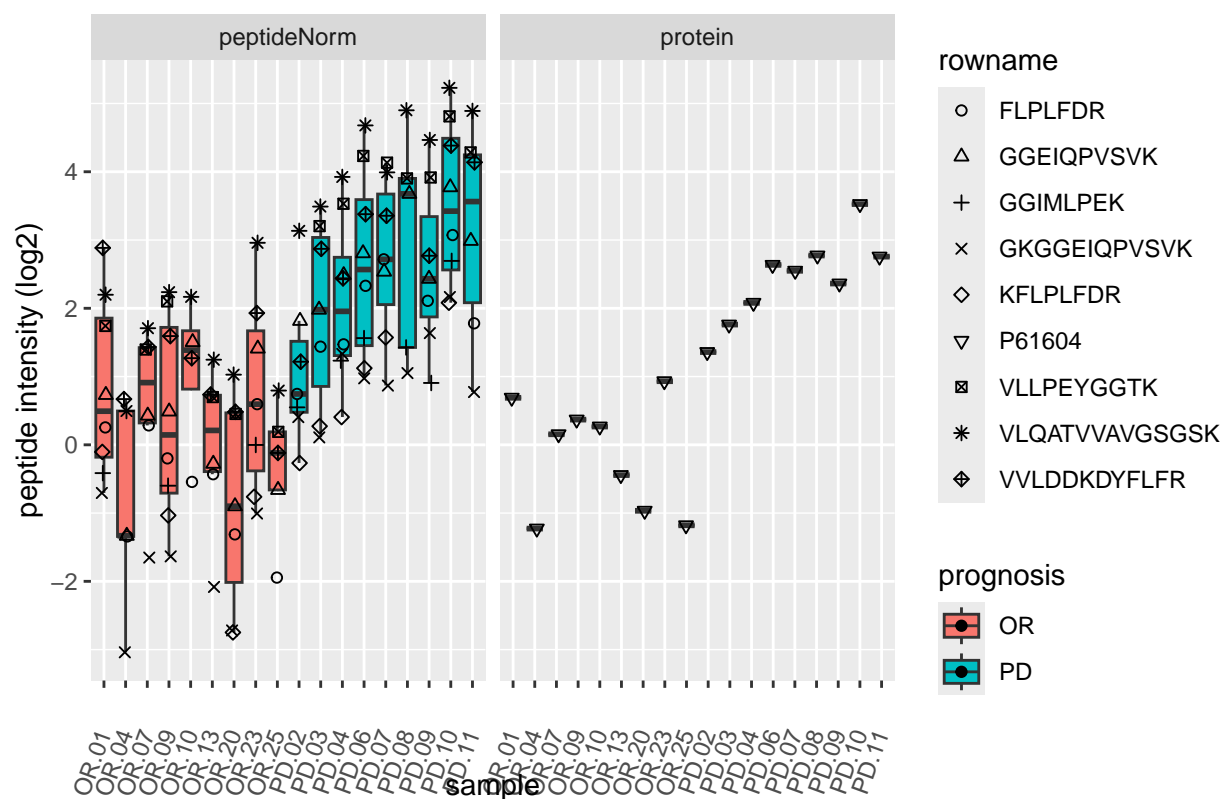
geom_point(
  position = position_jitter(width = .1),
  aes(shape = rowname)) +
scale_shape_manual(values = 1:nrow(pePlotDf)) +
labs(title = protName, x = "sample", y = "peptide intensity (log2)") +
theme(axis.text.x = element_text(angle = 70, hjust = 1, vjust = 0.5)) +
facet_grid(~assay)
print(p2)
}

```

P61604



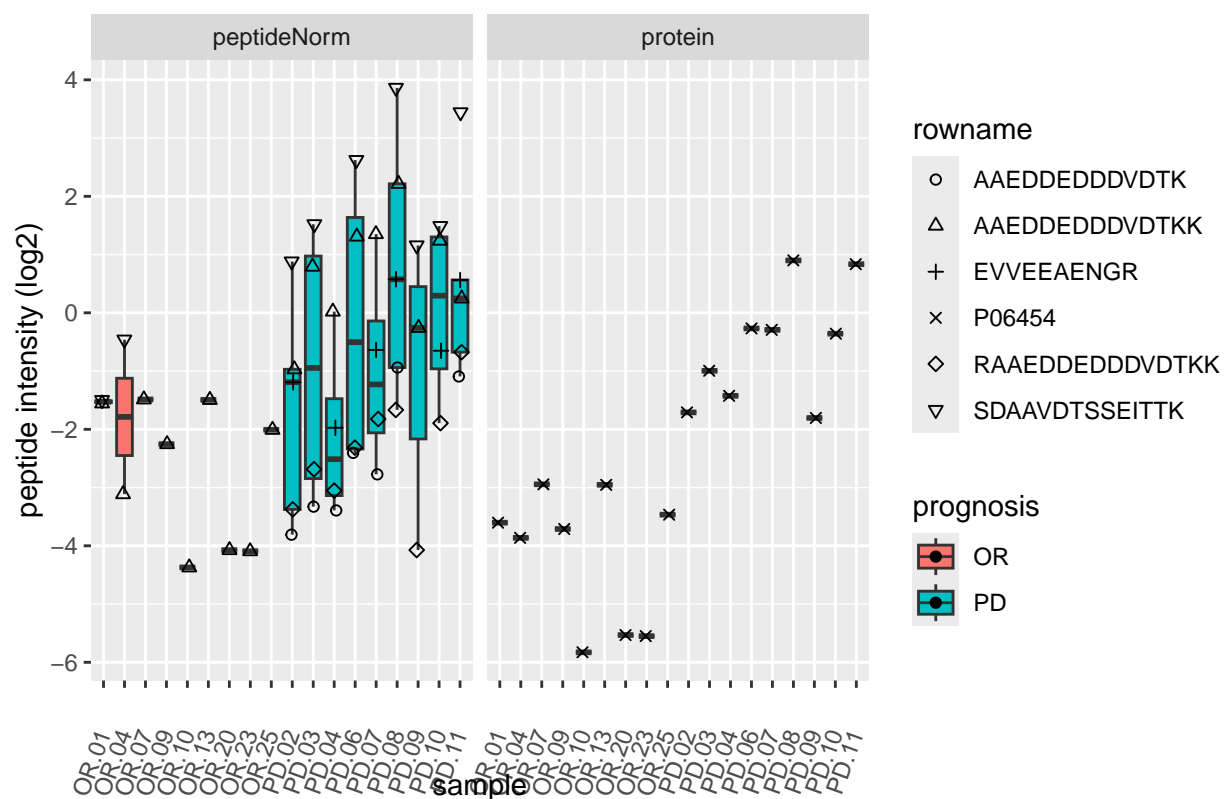
P61604



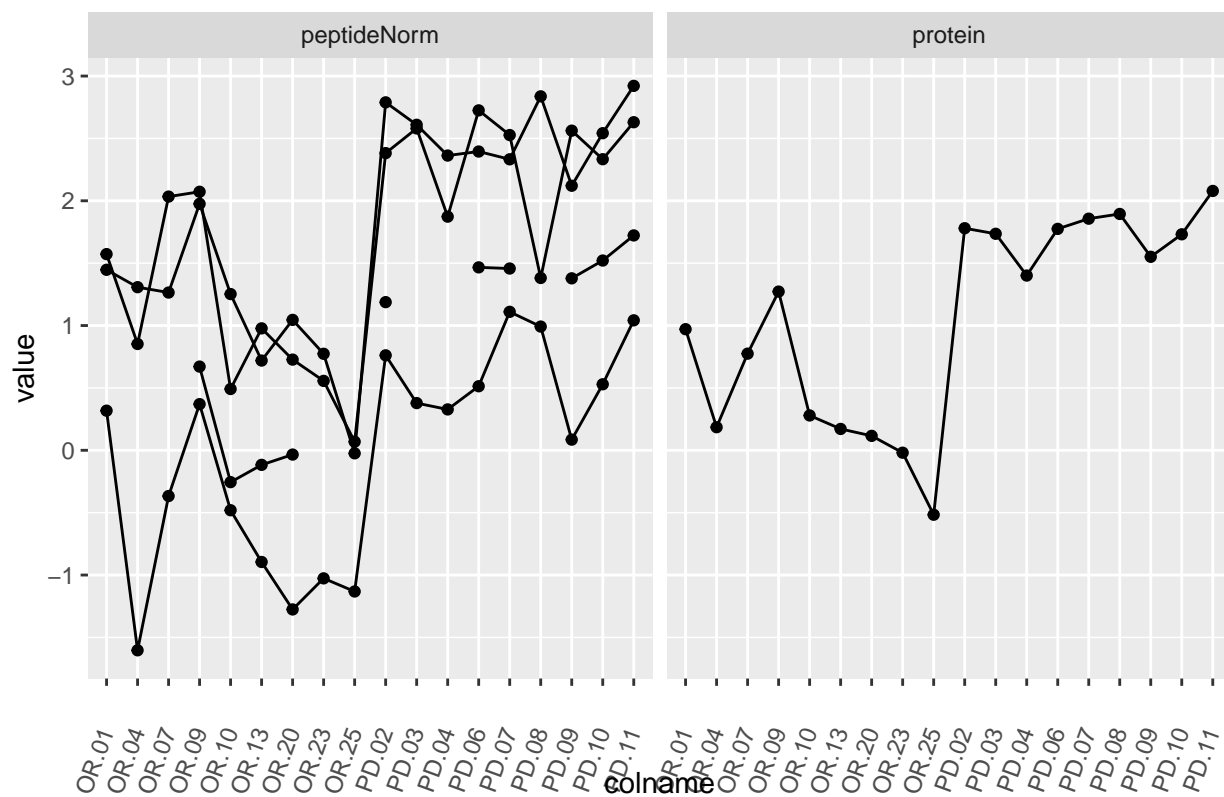
P06454



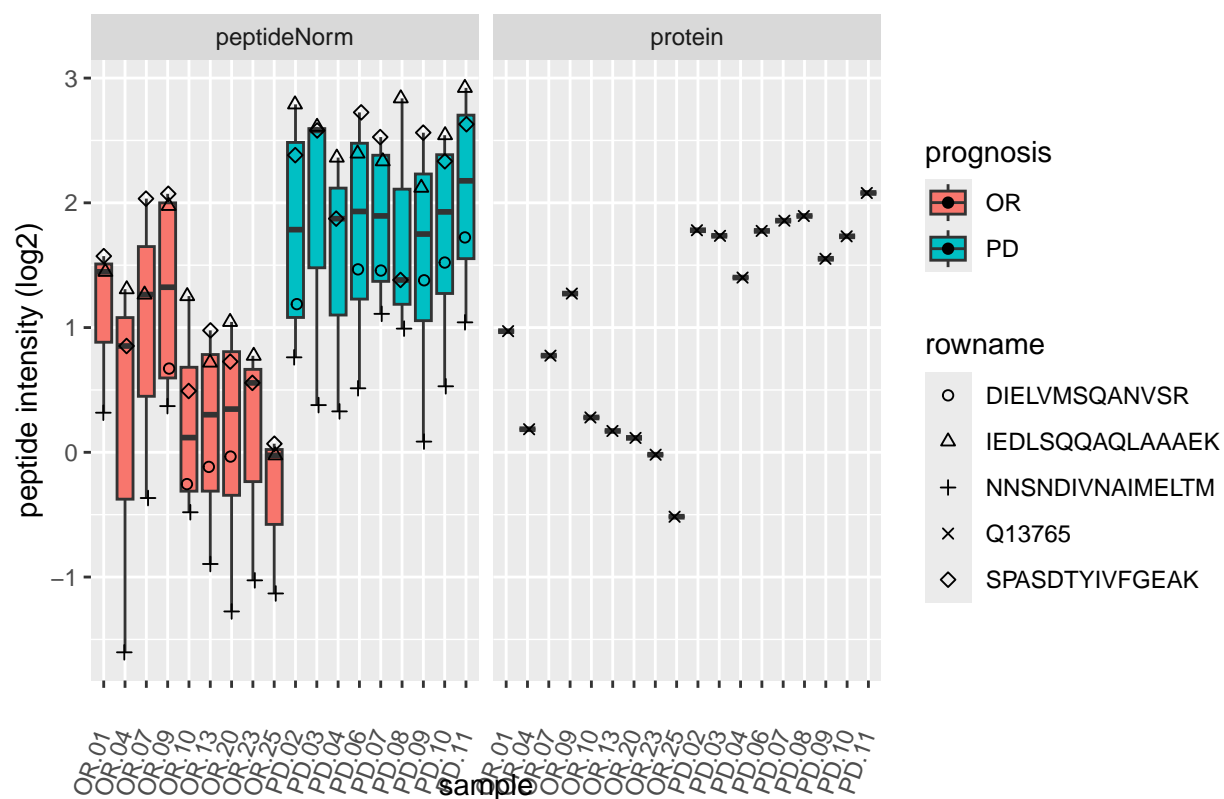
P06454



Q13765

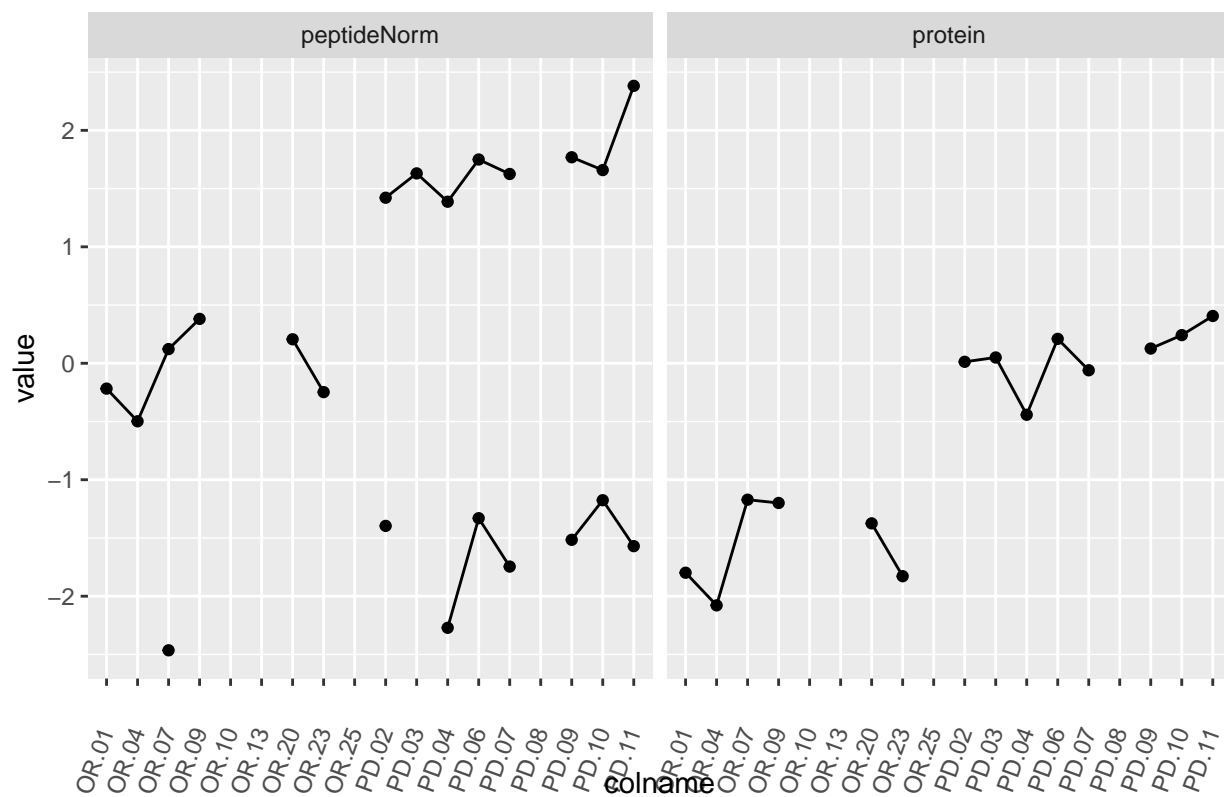


Q13765

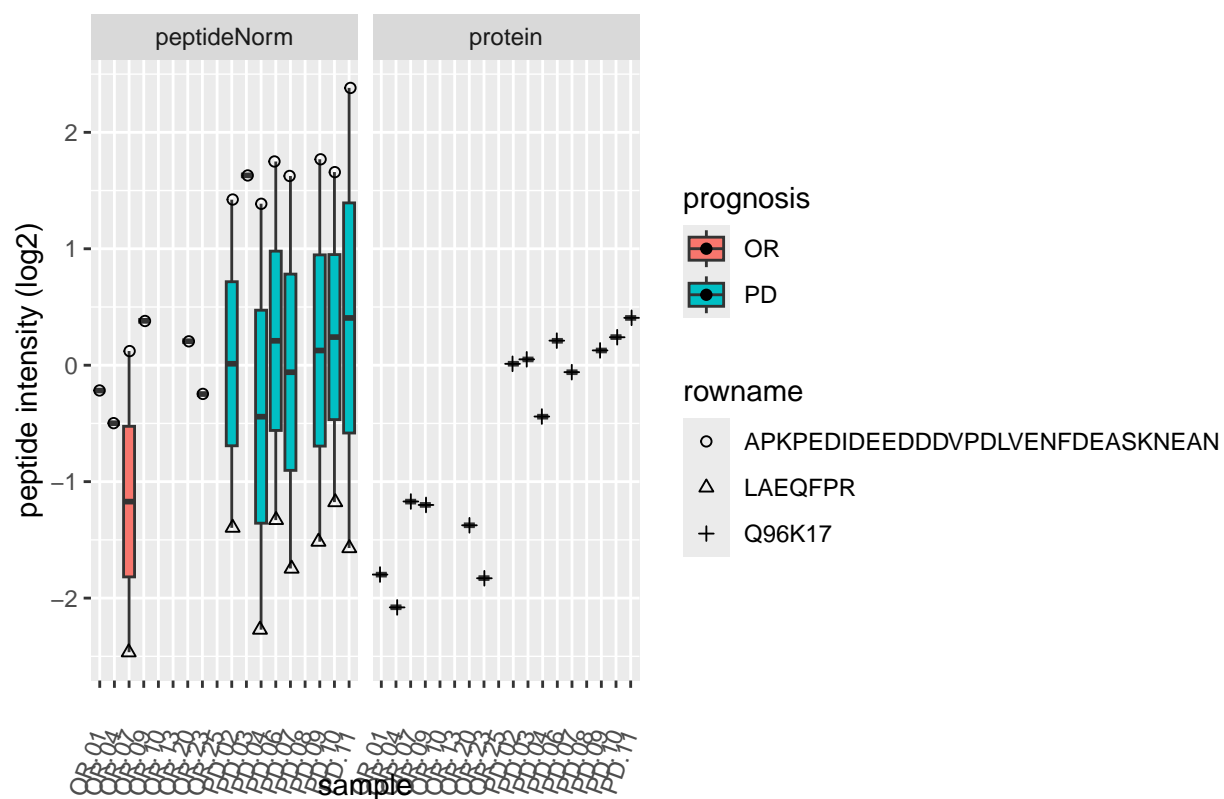




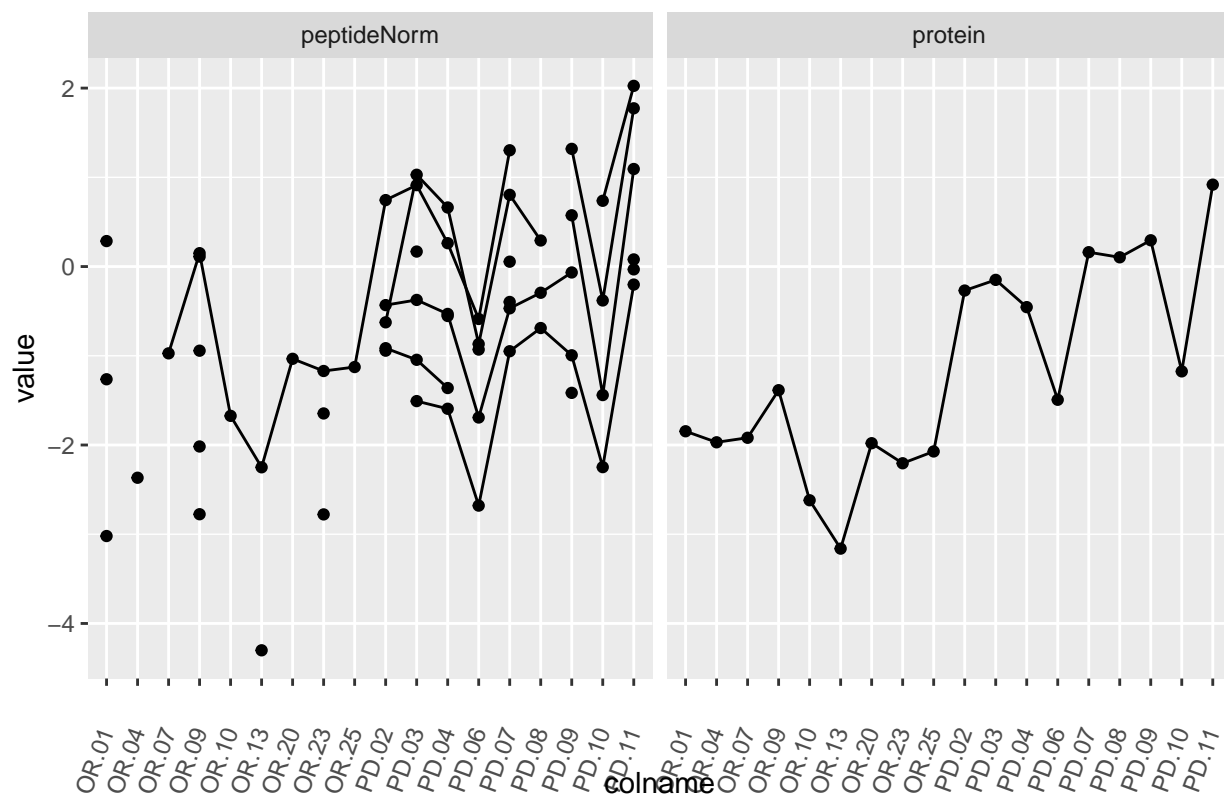
Q96K17

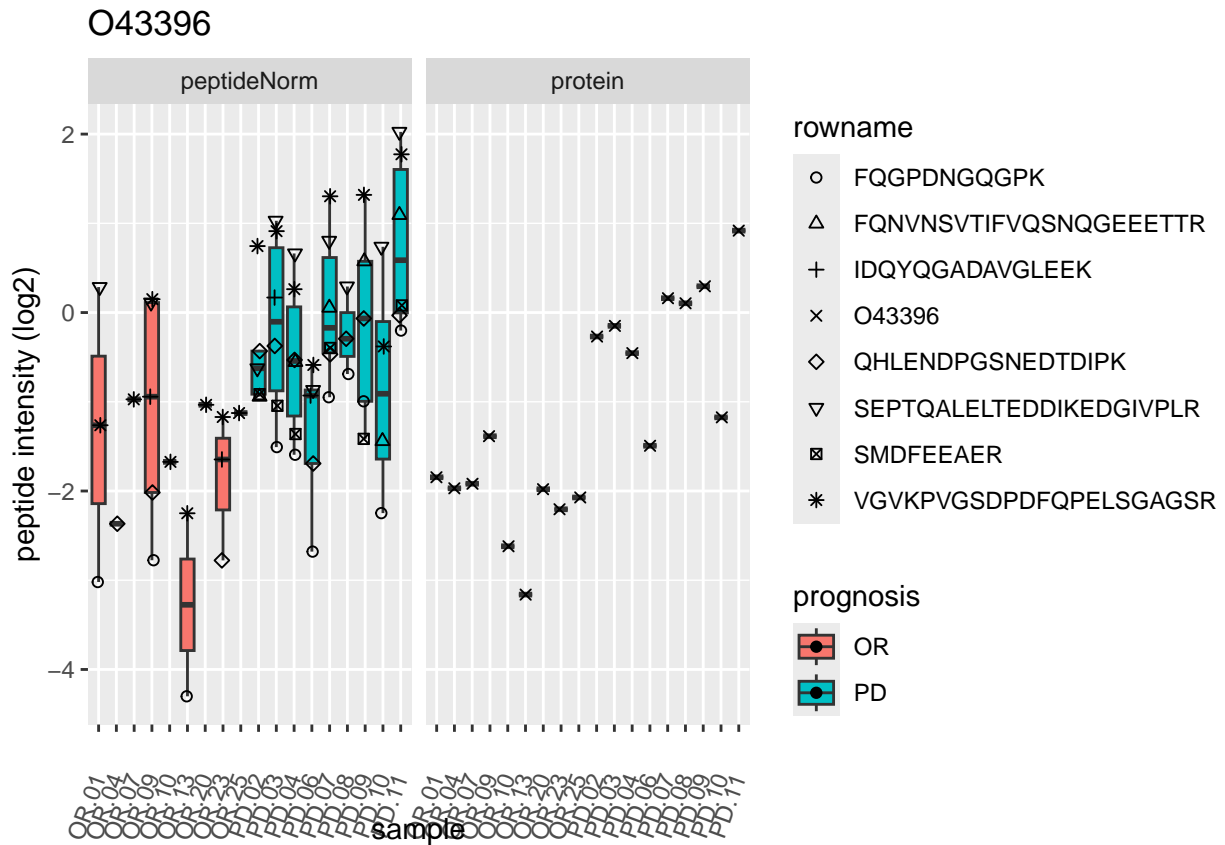


# Q96K17

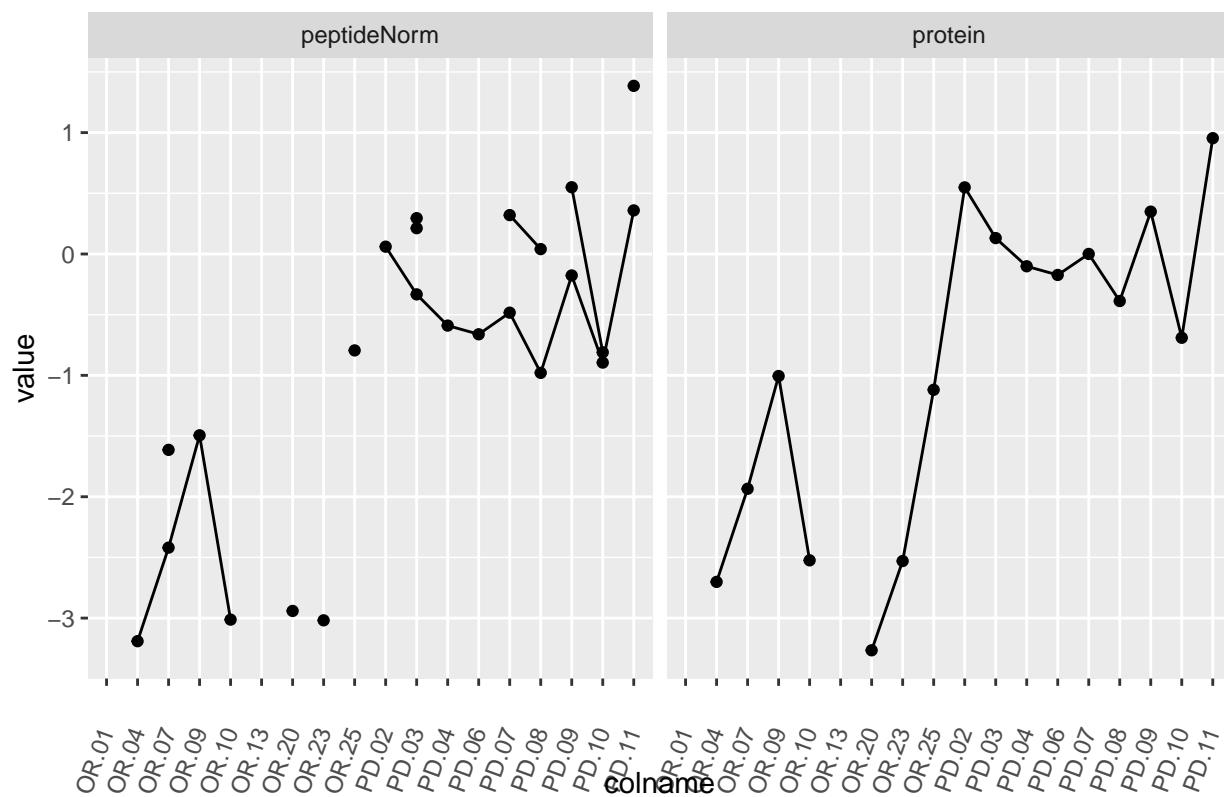


O43396

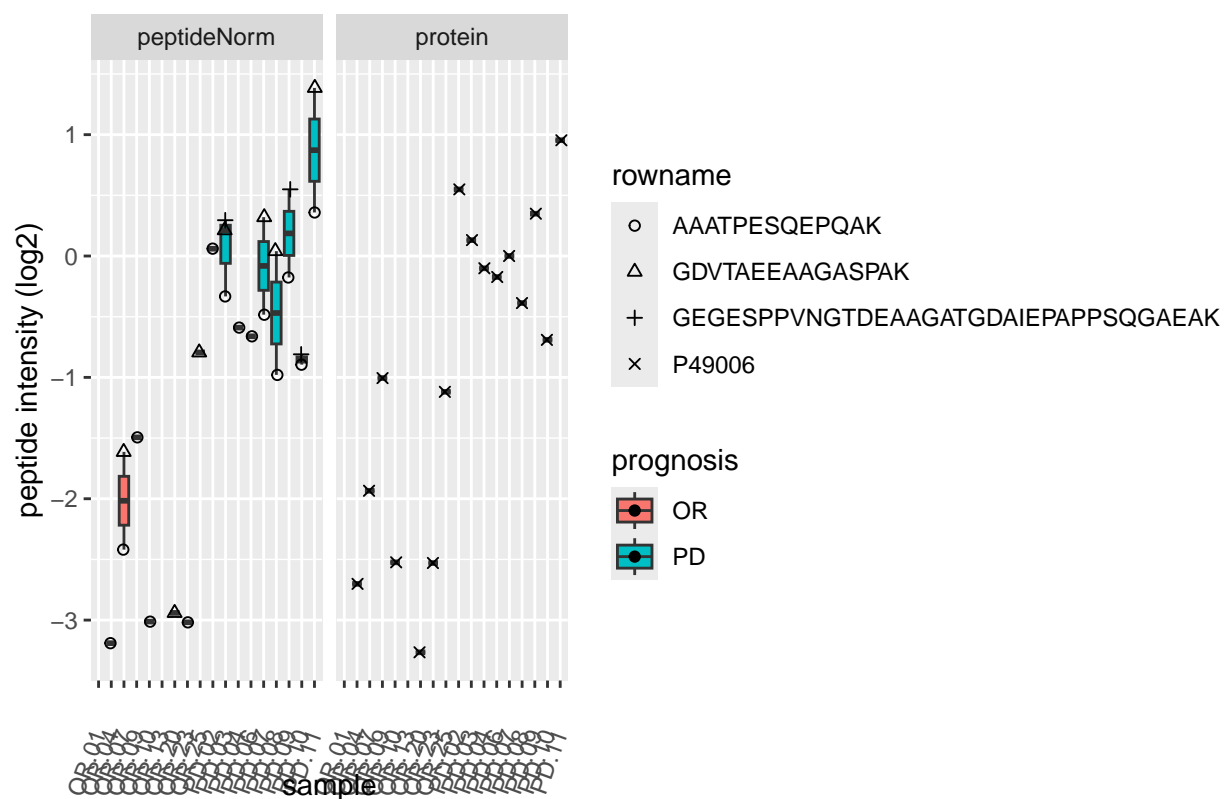




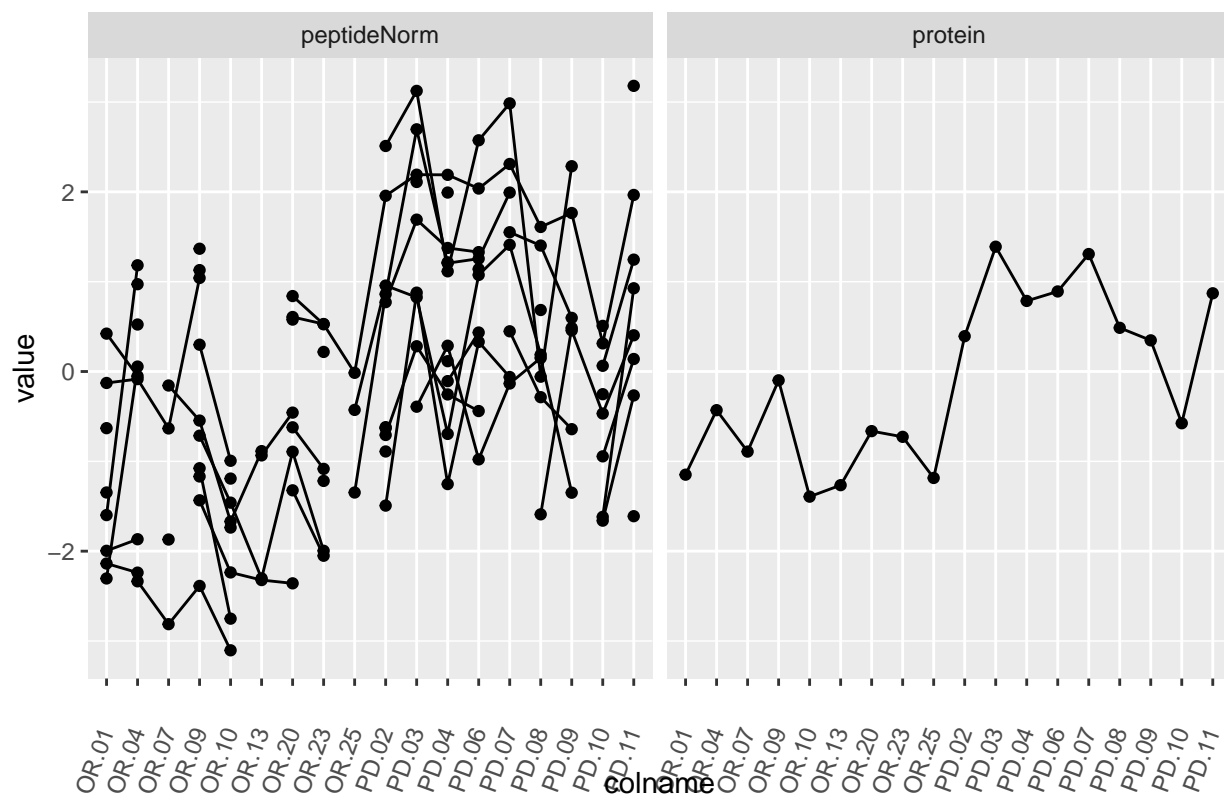
P49006

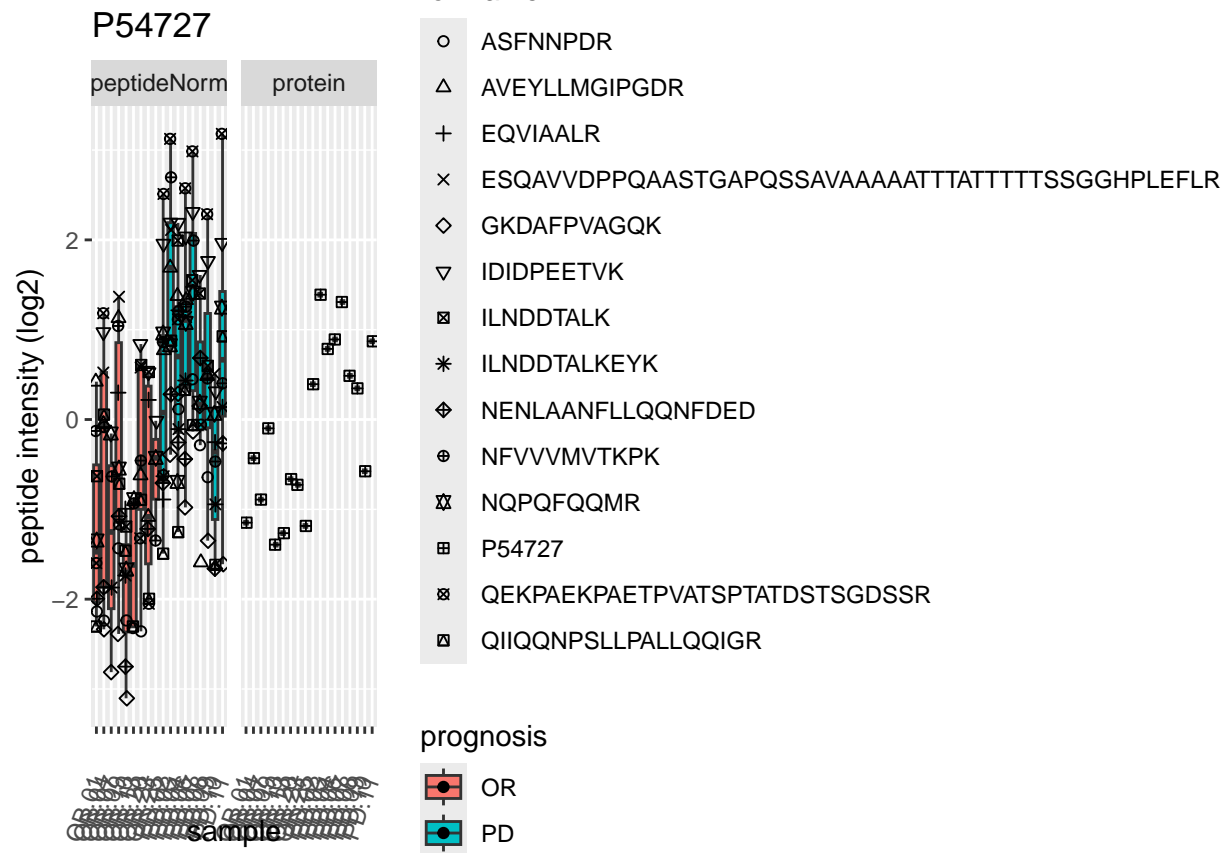


P49006



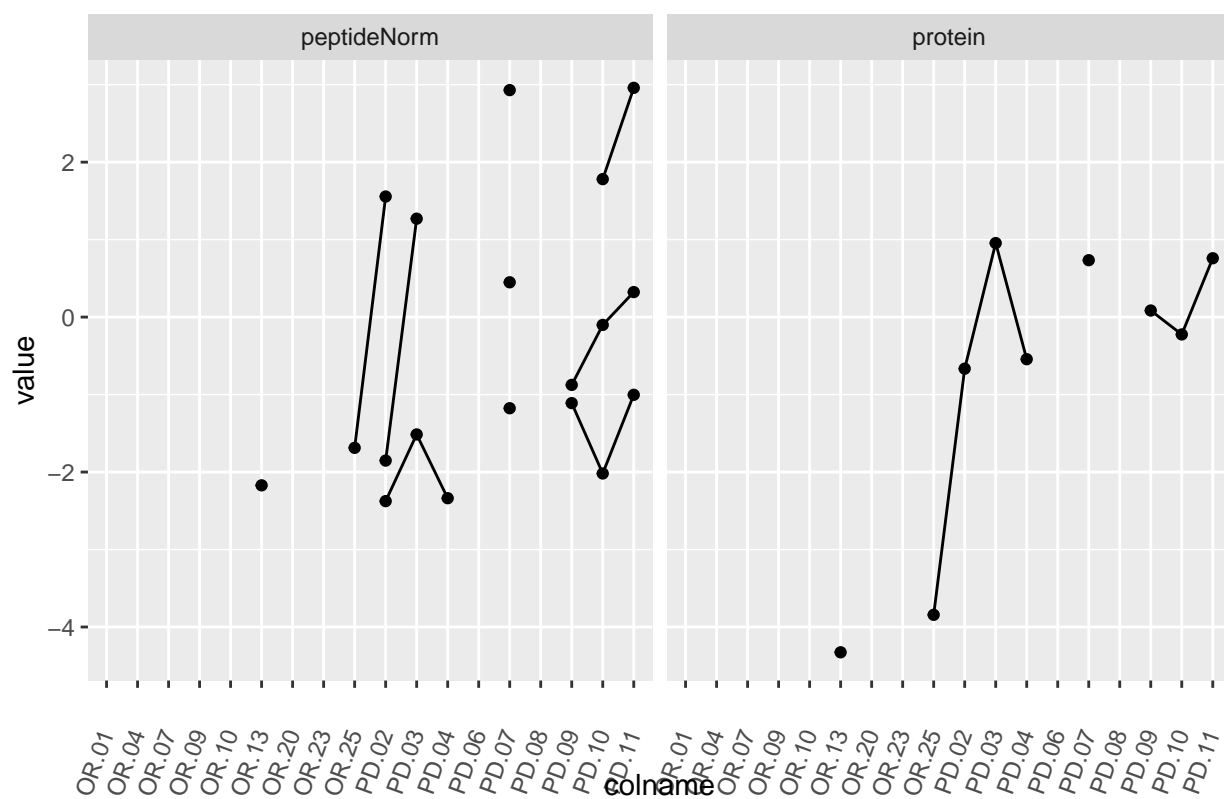
P54727



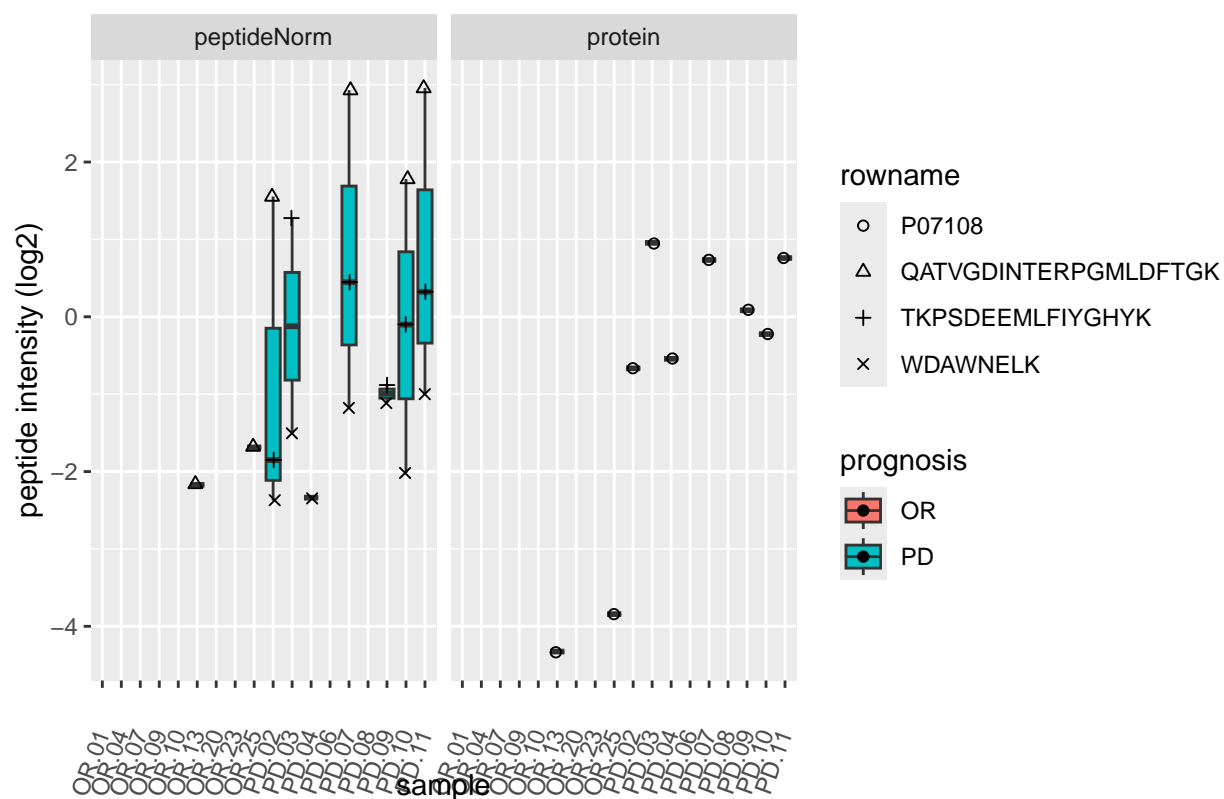




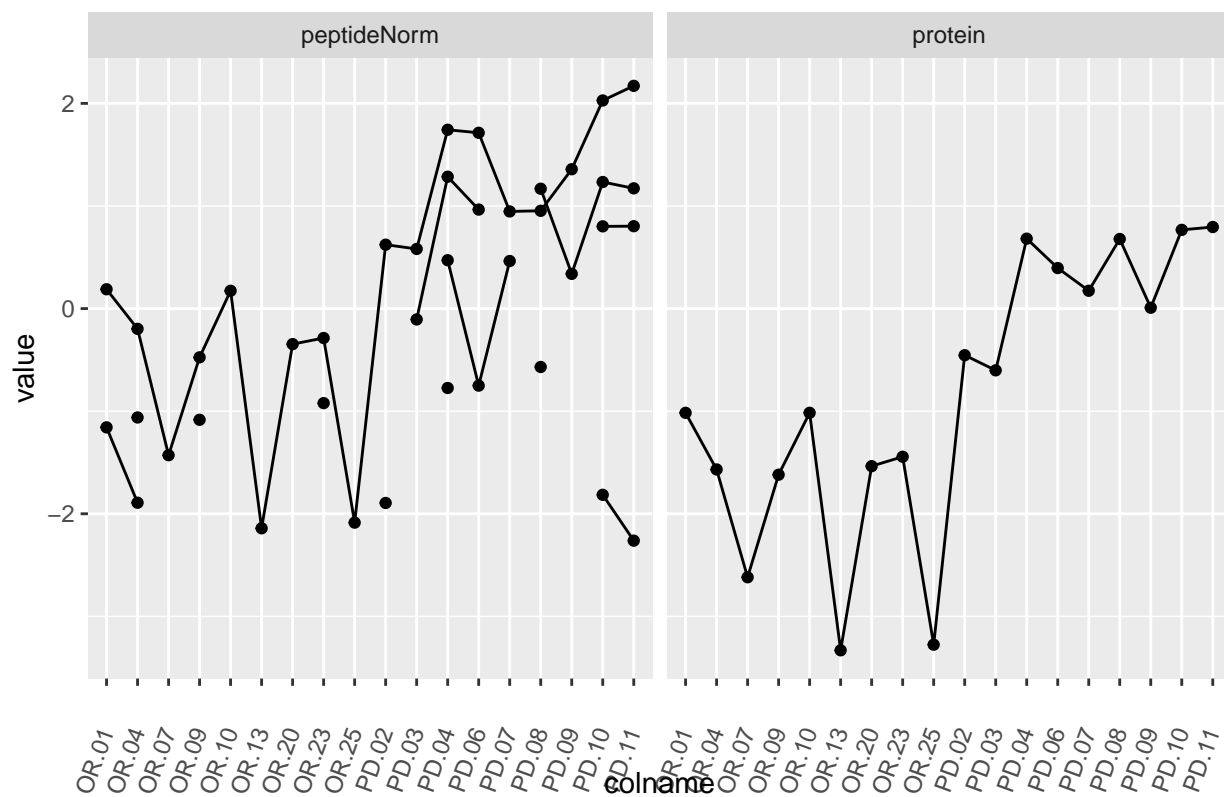
P07108



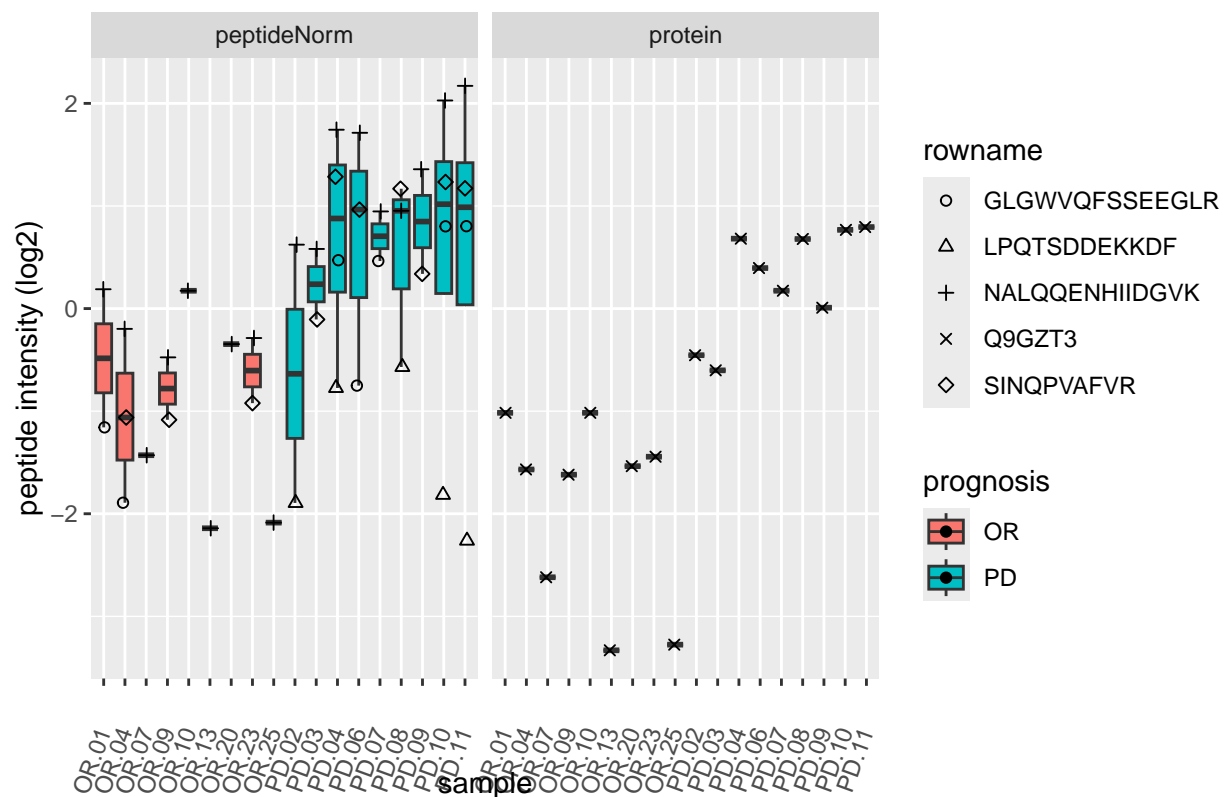
P07108



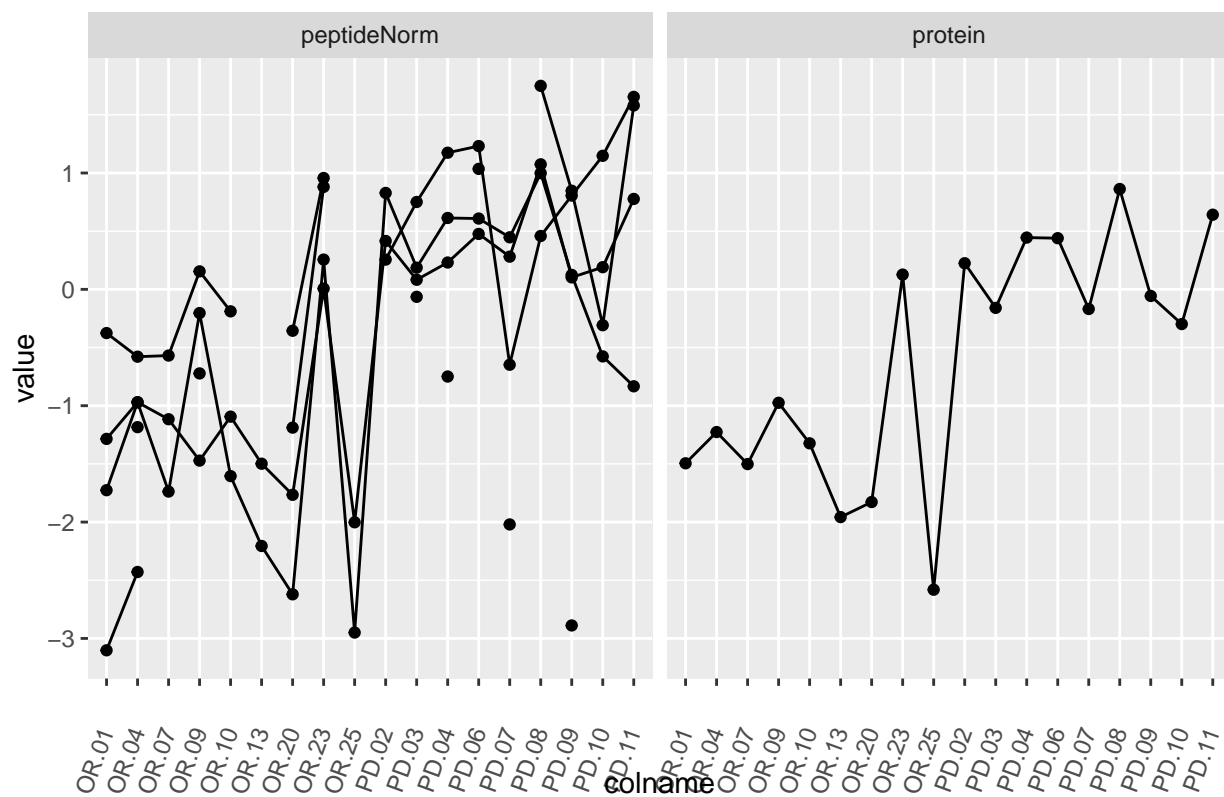
Q9GZT3

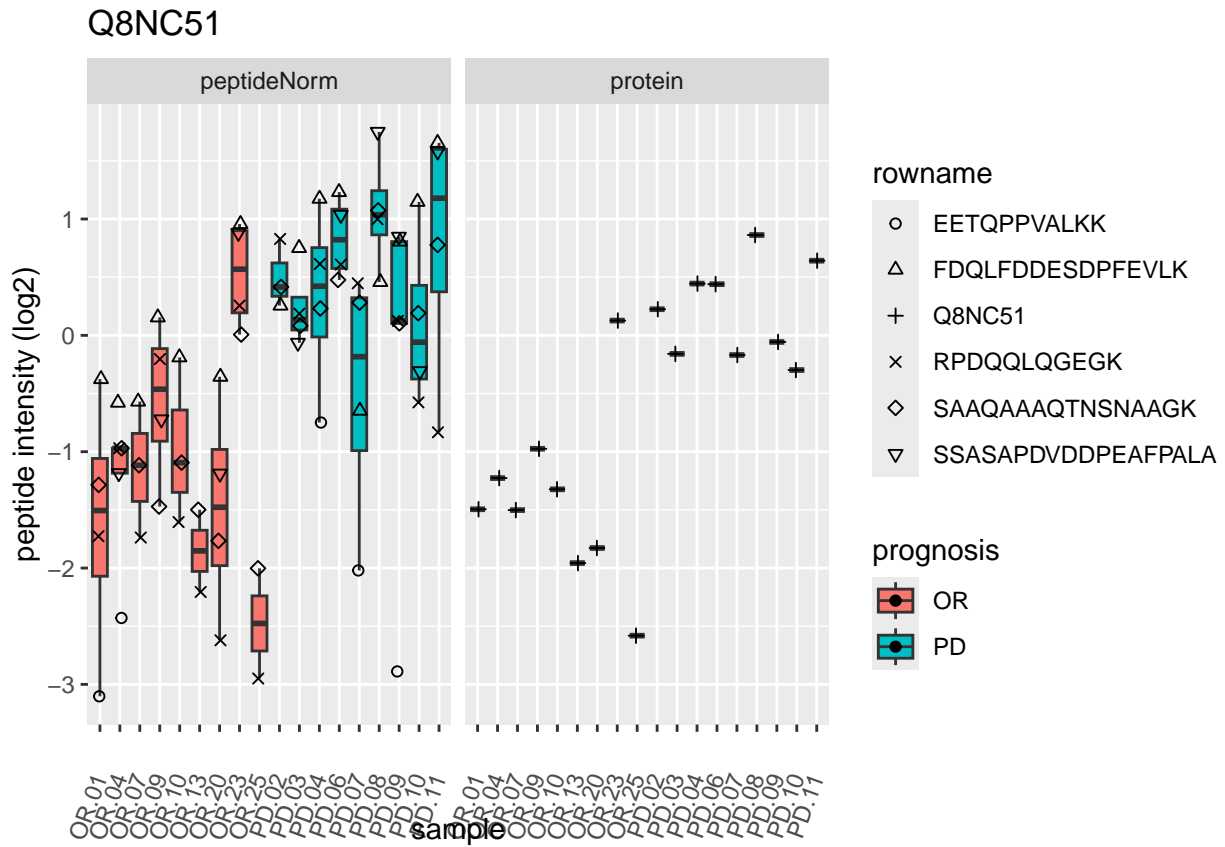


# Q9GZT3



Q8NC51

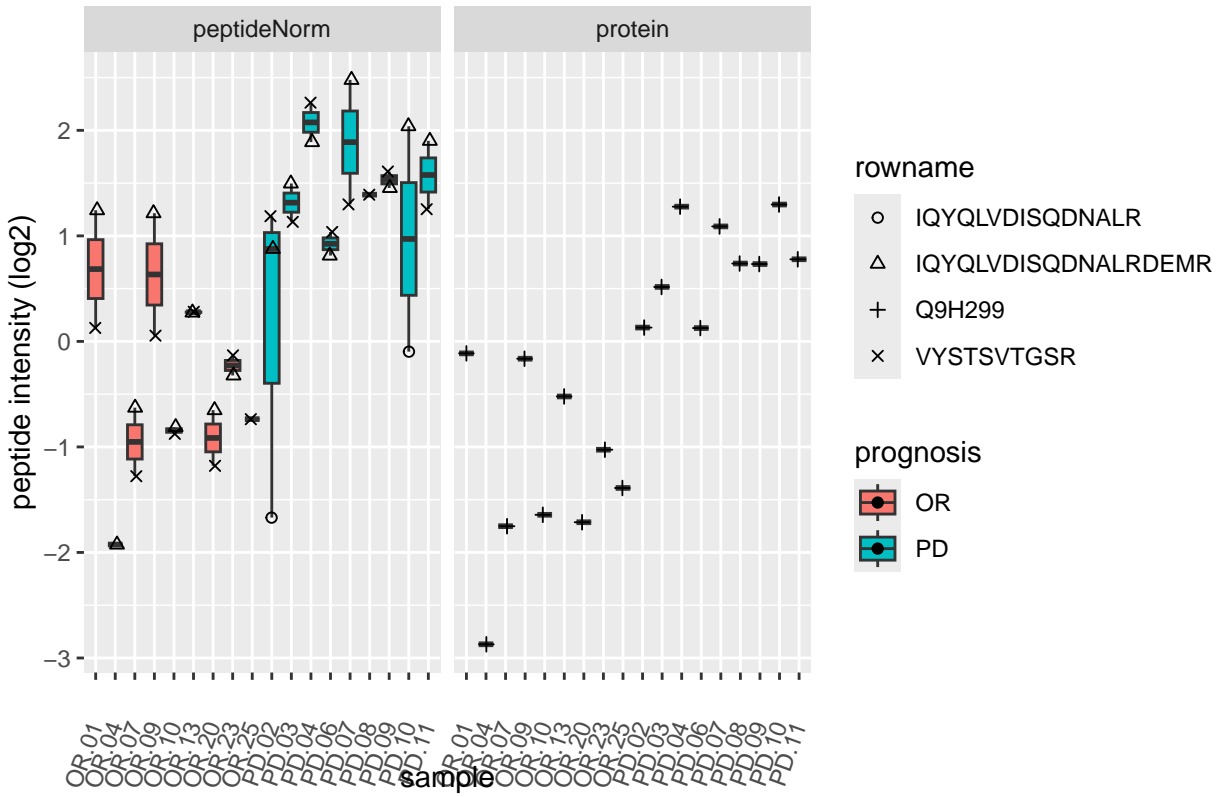




Q9H299

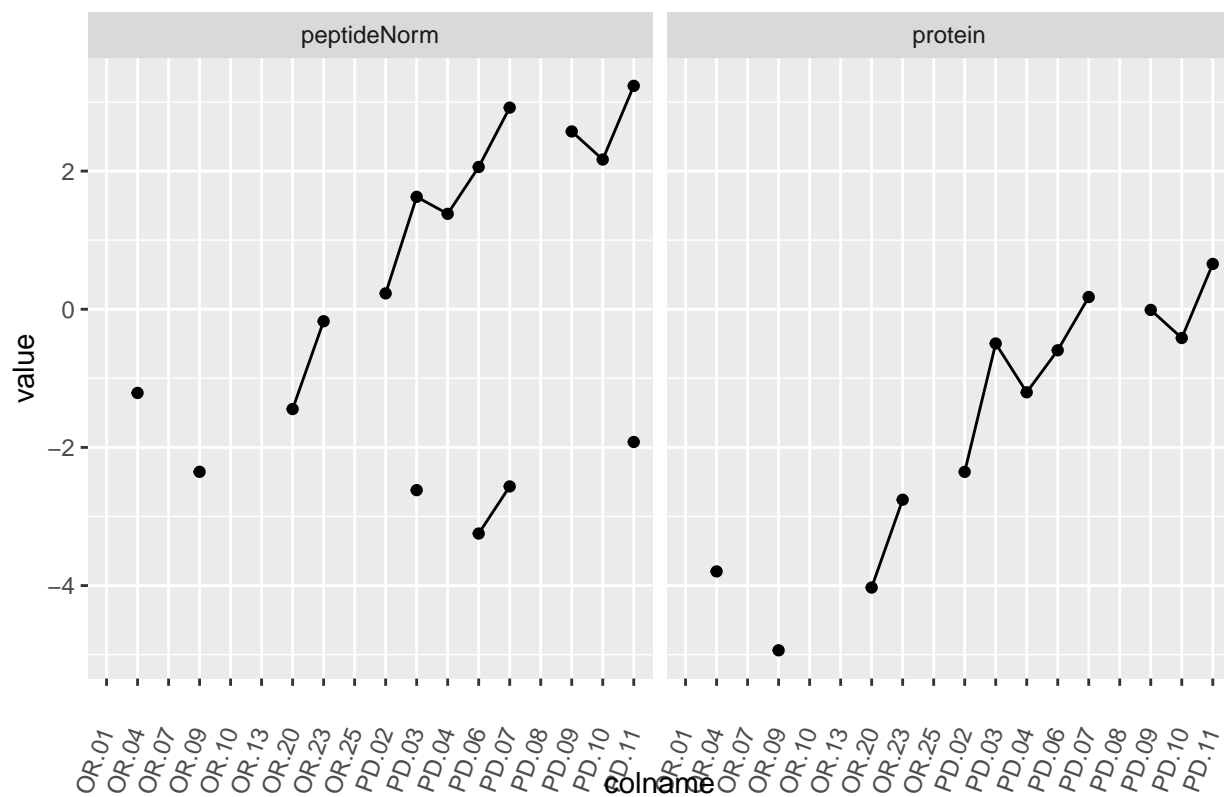


## Q9H299

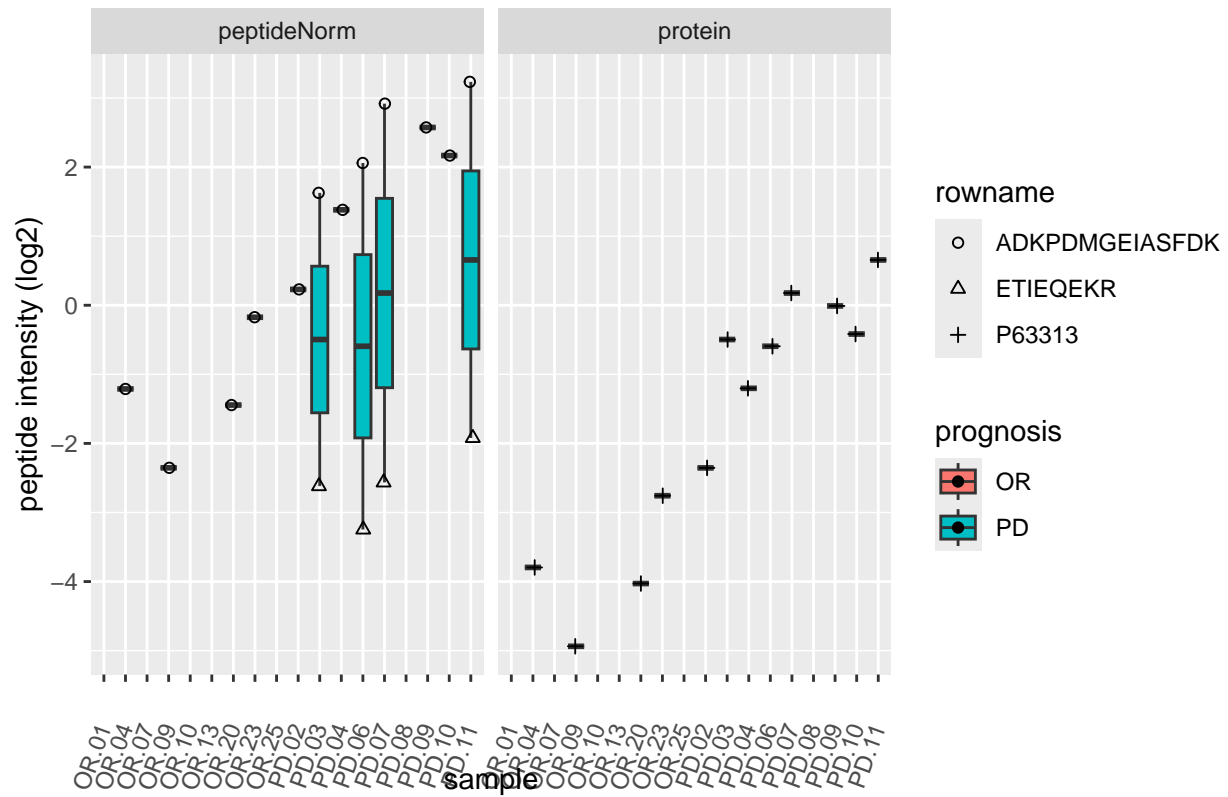




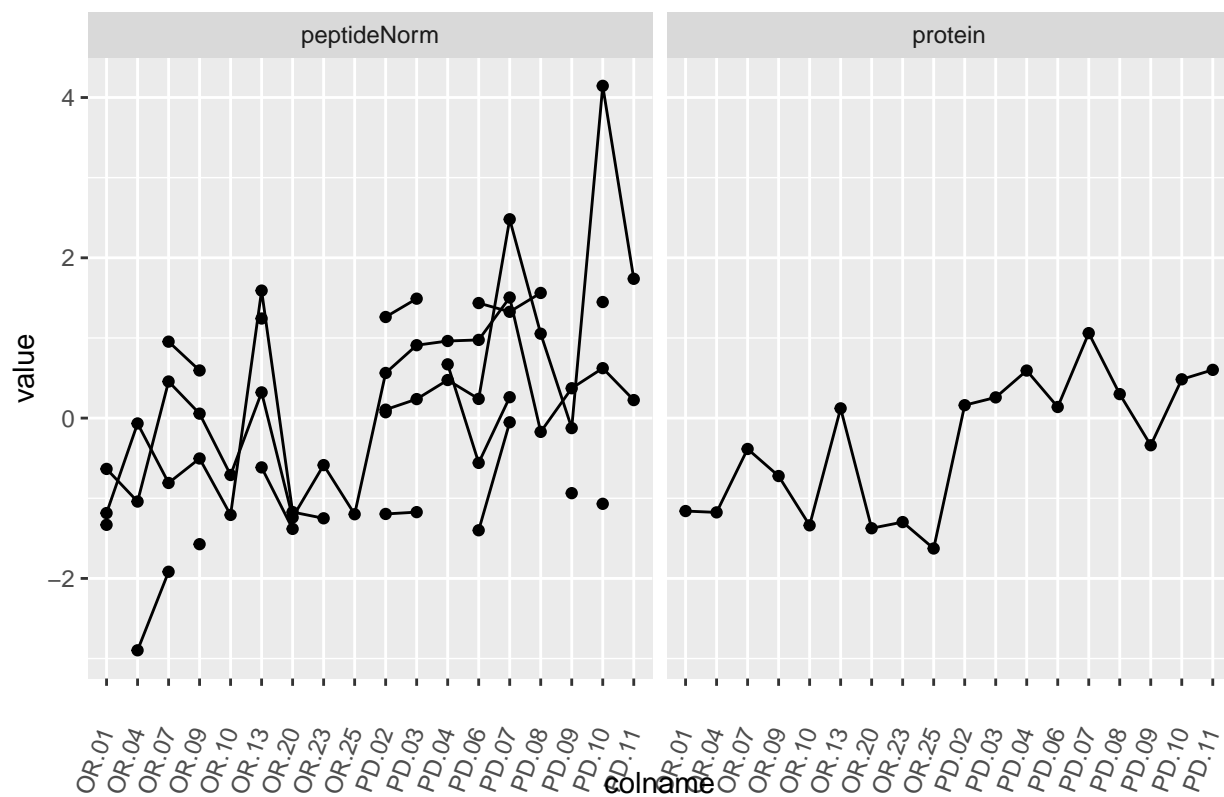
P63313



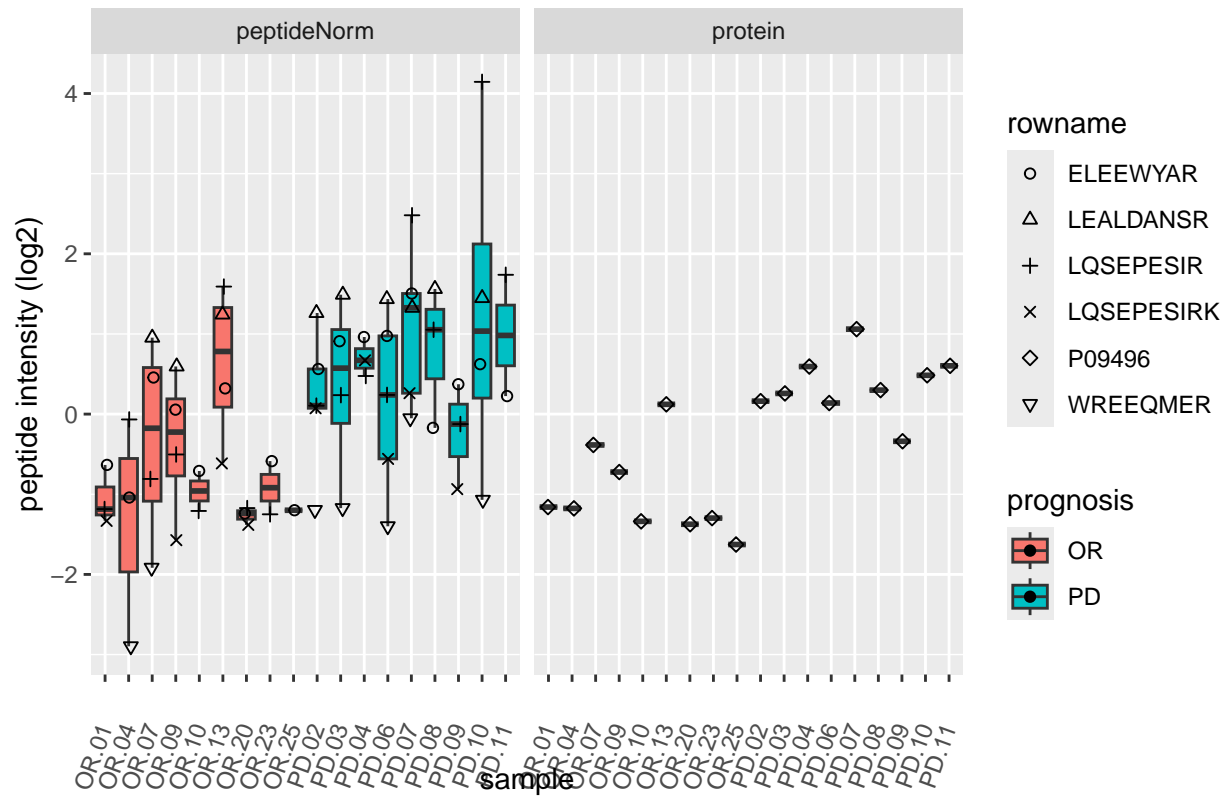
## P63313



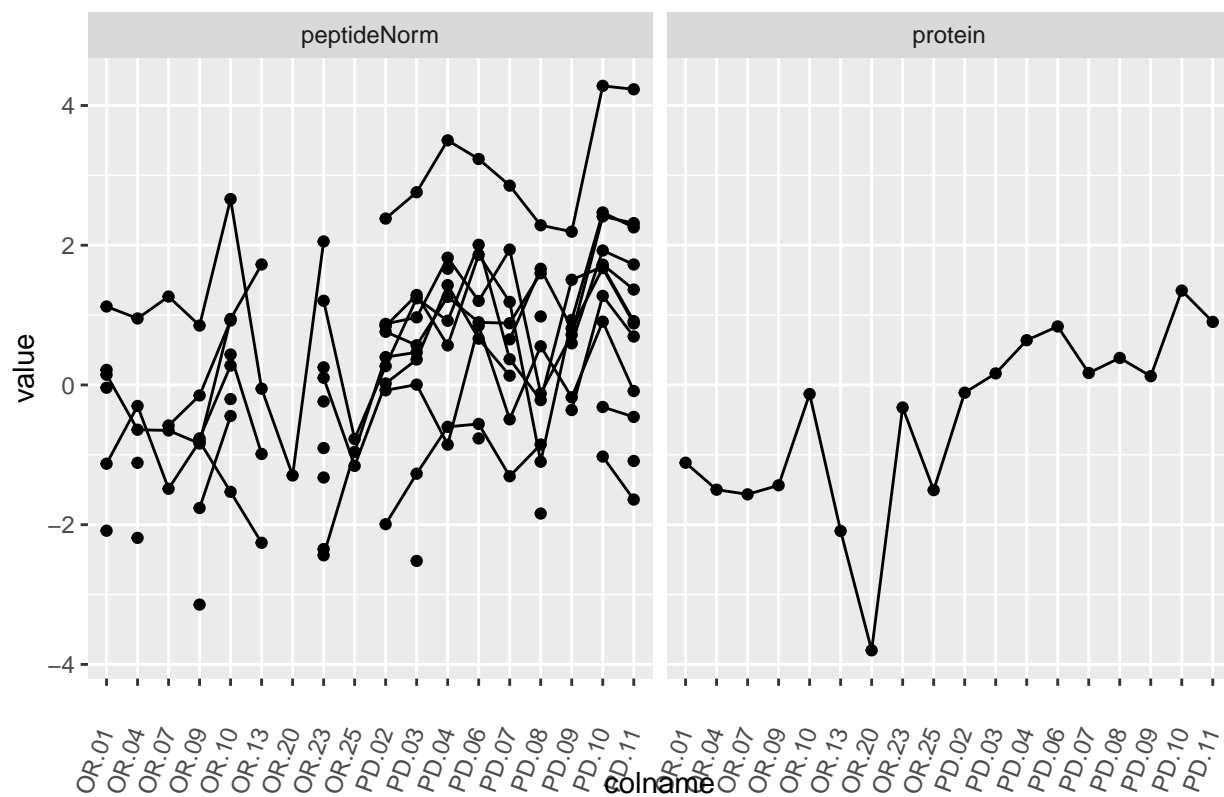
P09496

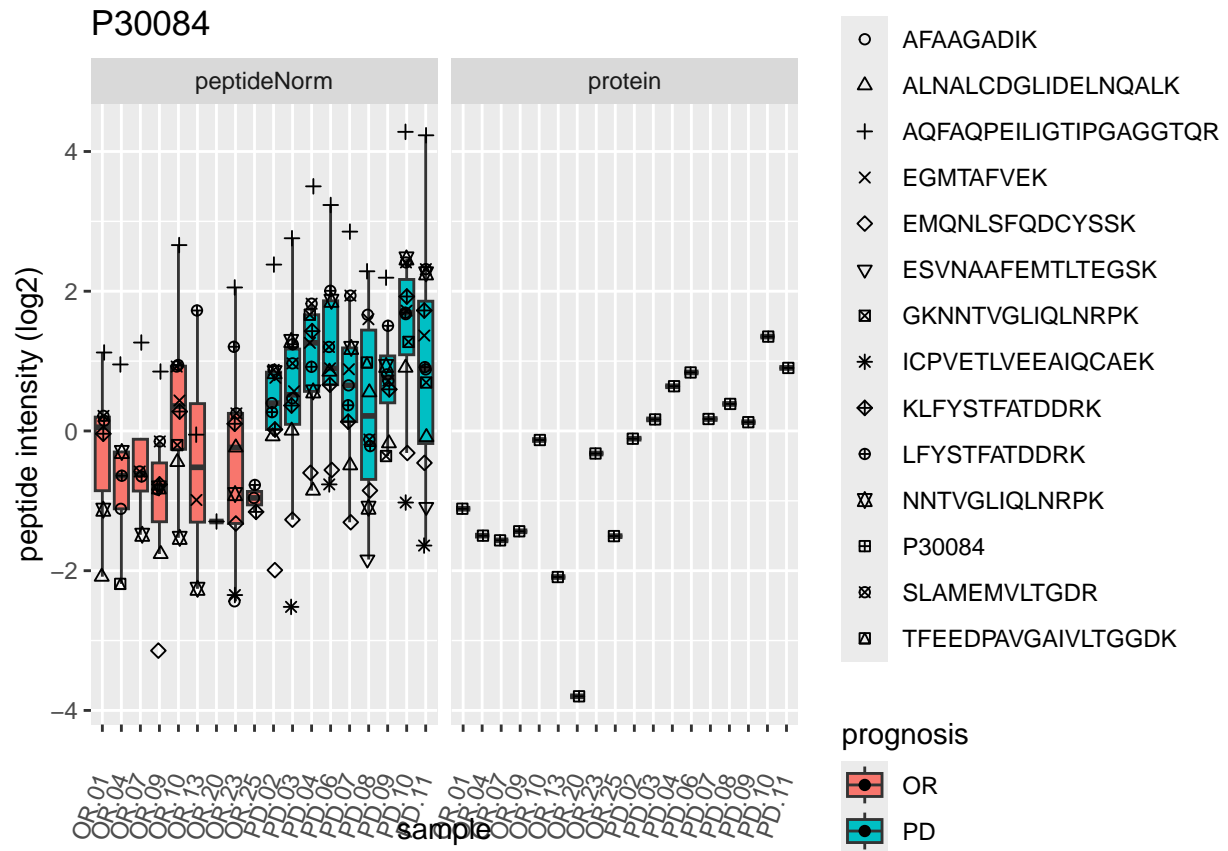


P09496

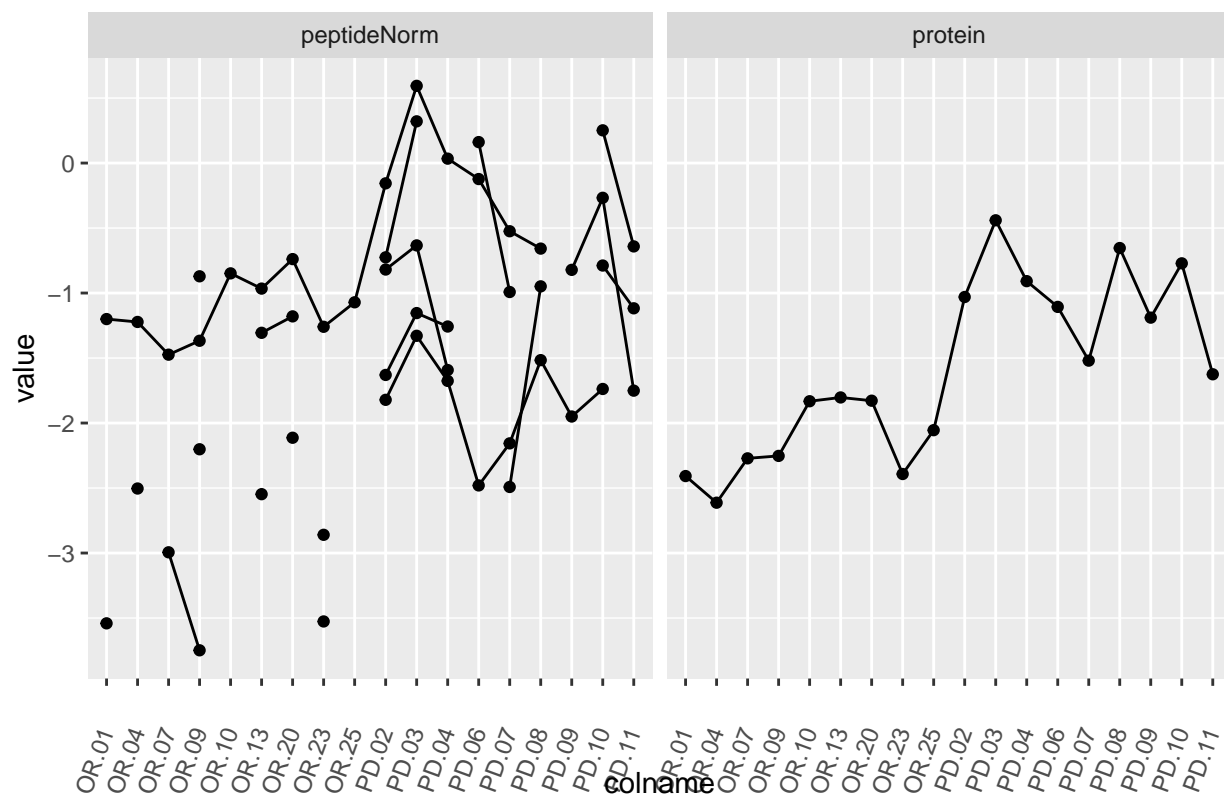


P30084

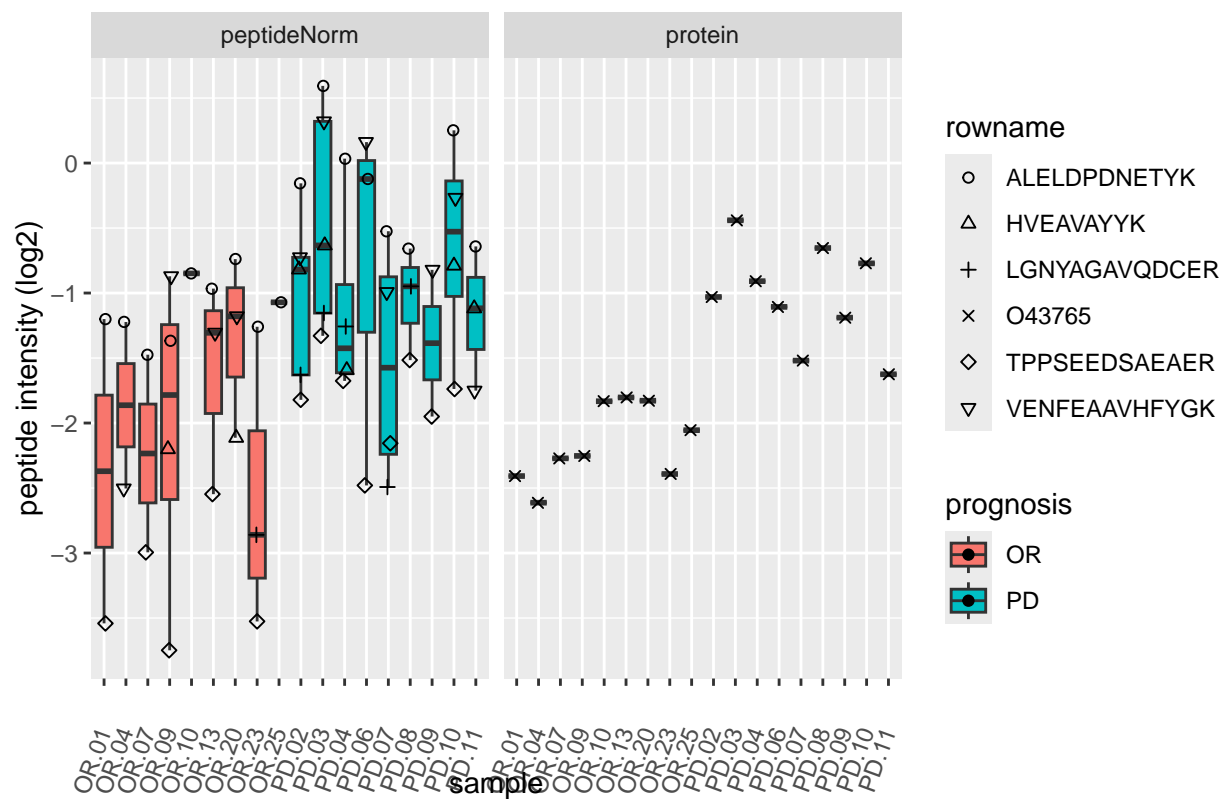




O43765

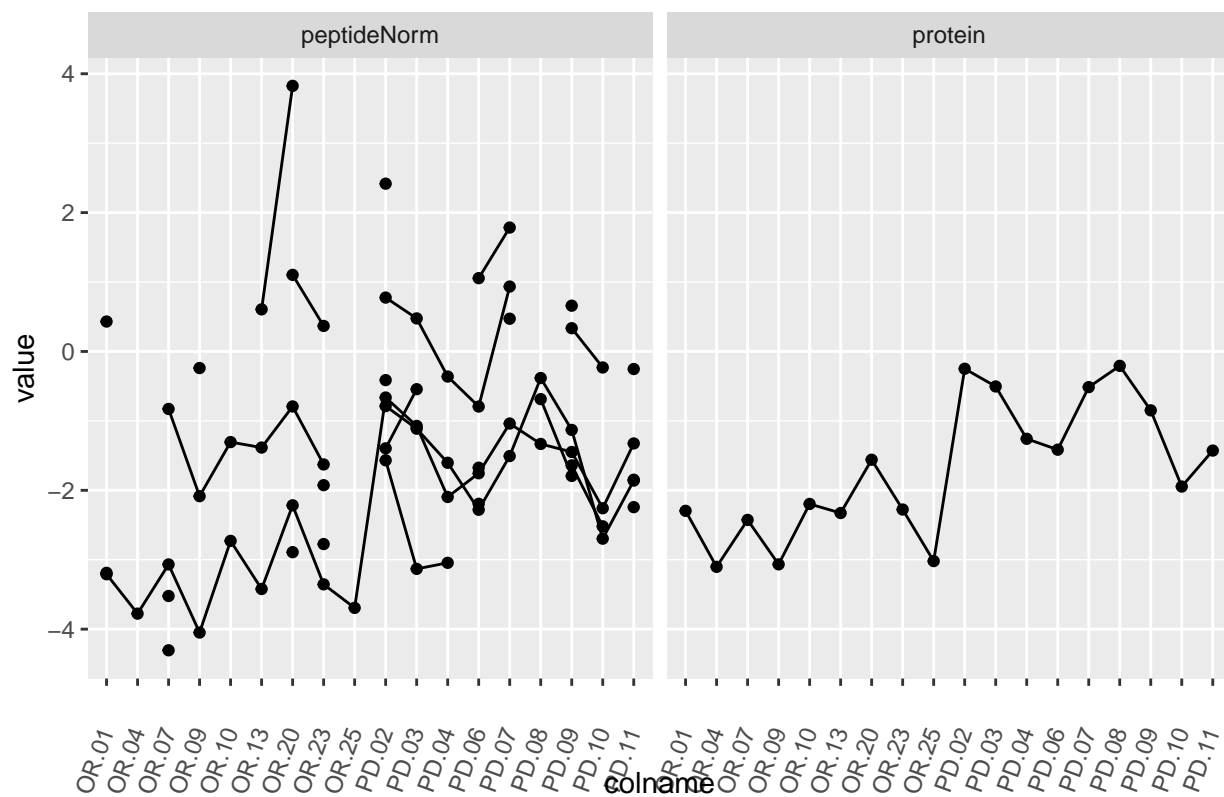


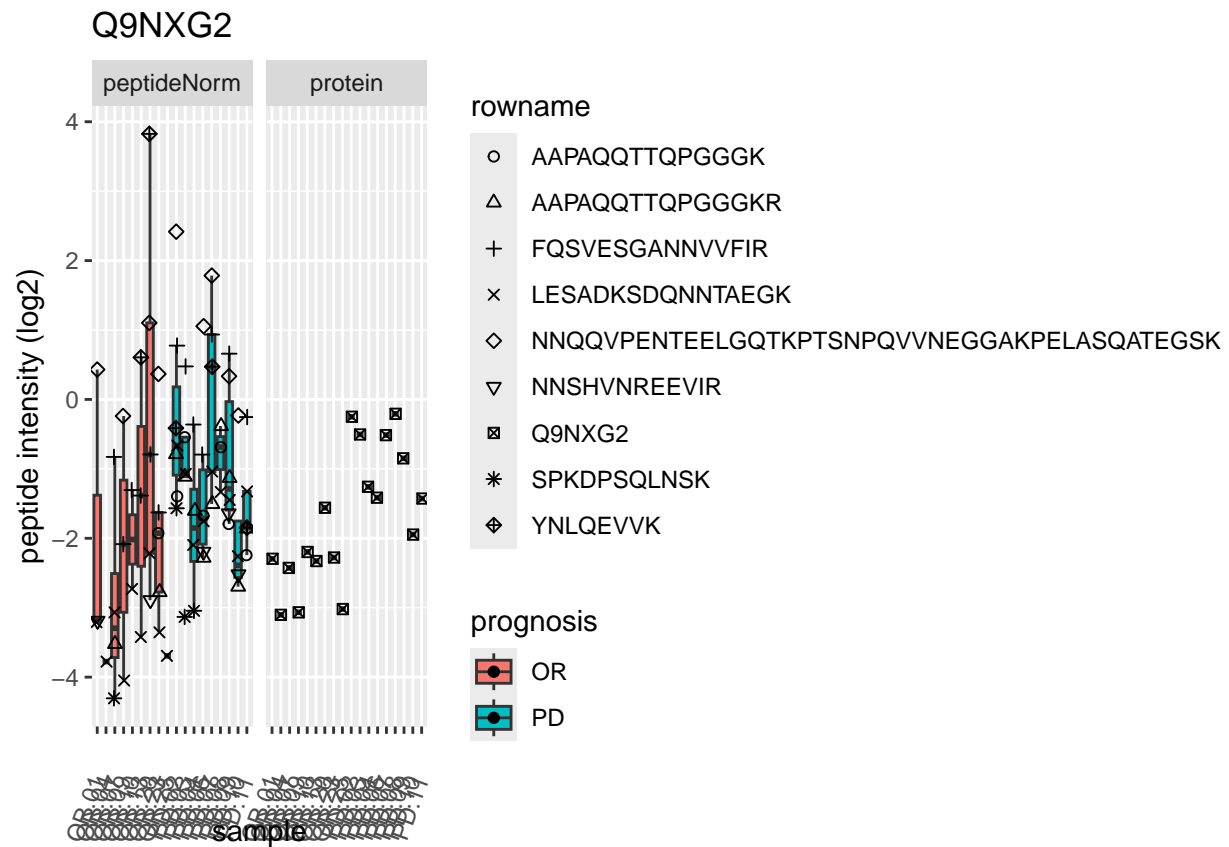
O43765



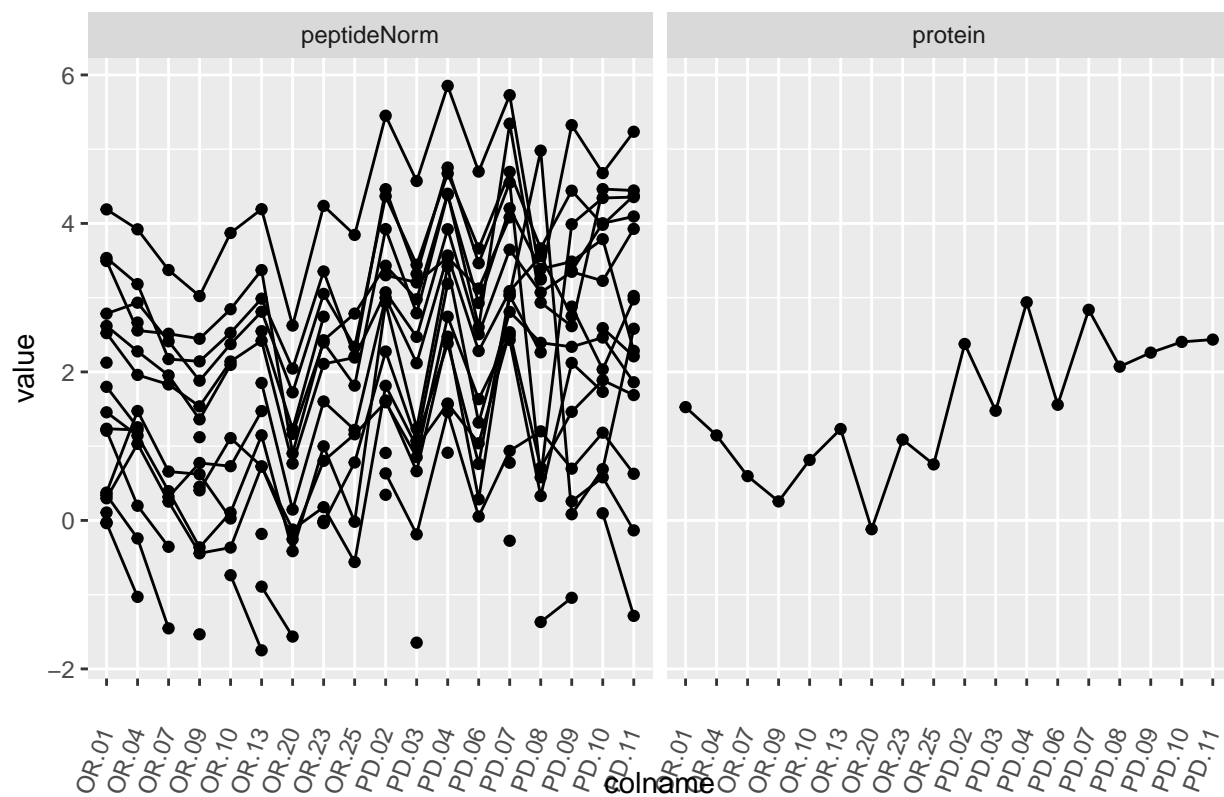


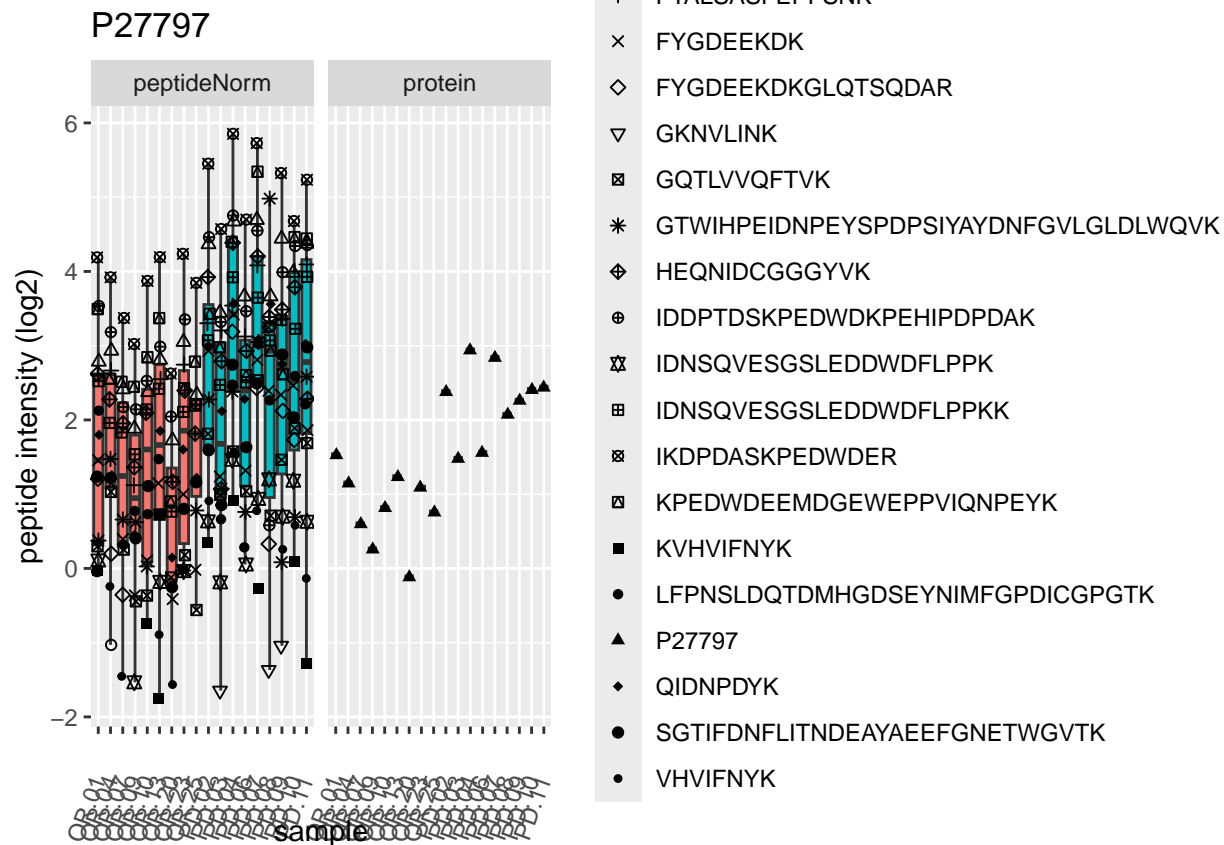
# Q9NXG2

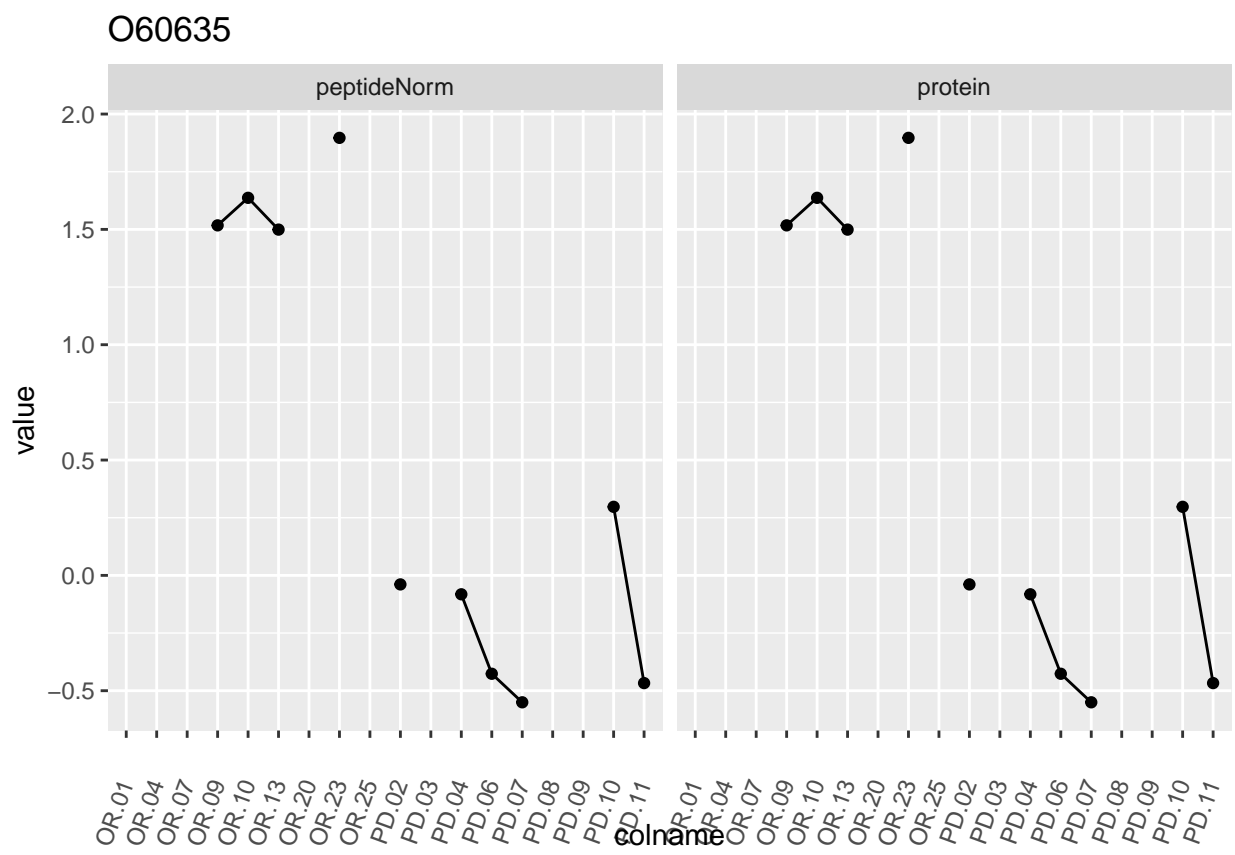


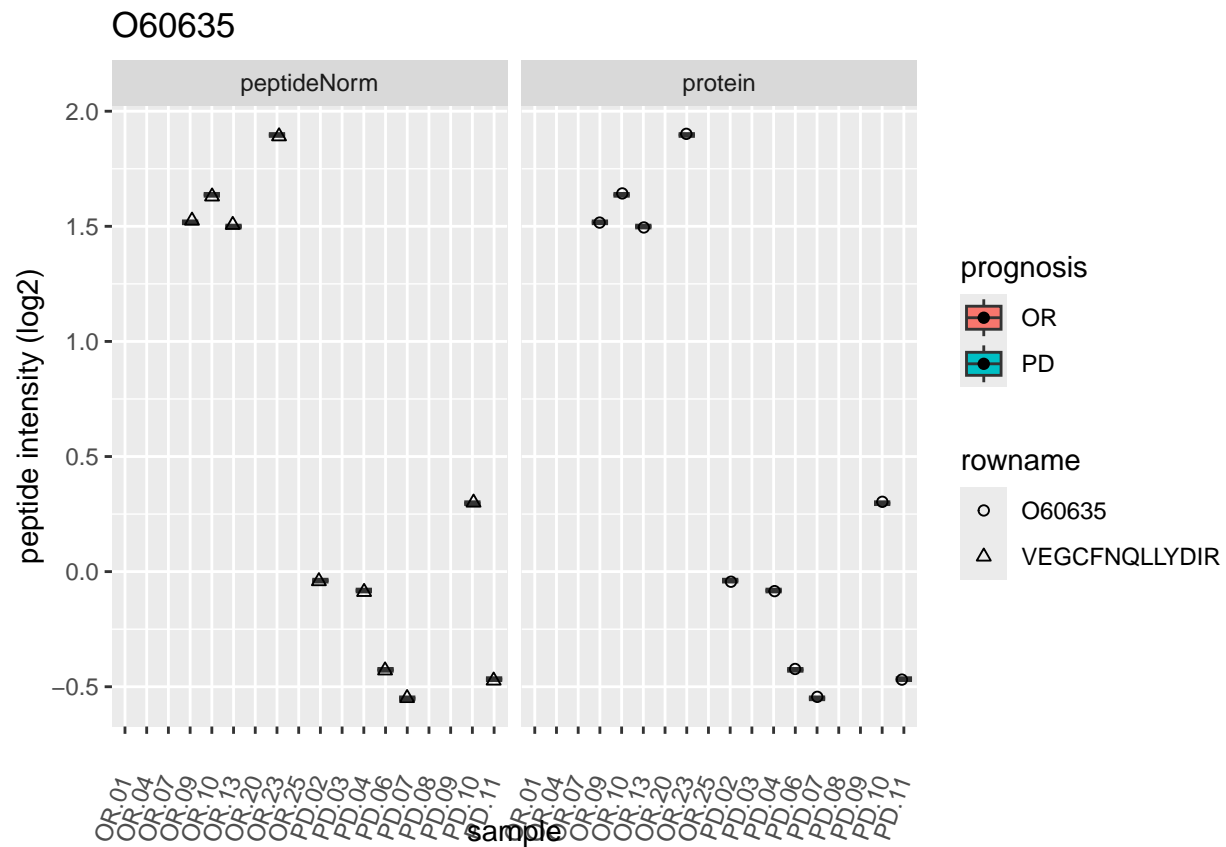


P27797

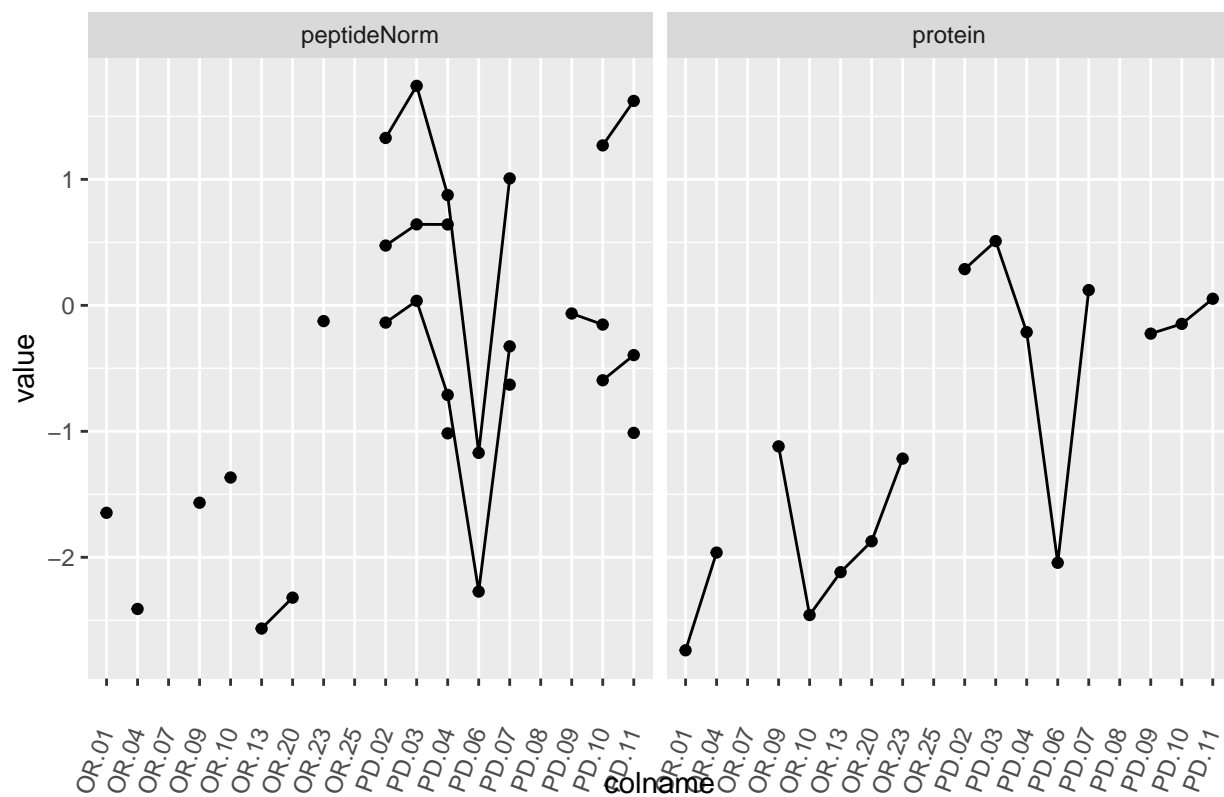




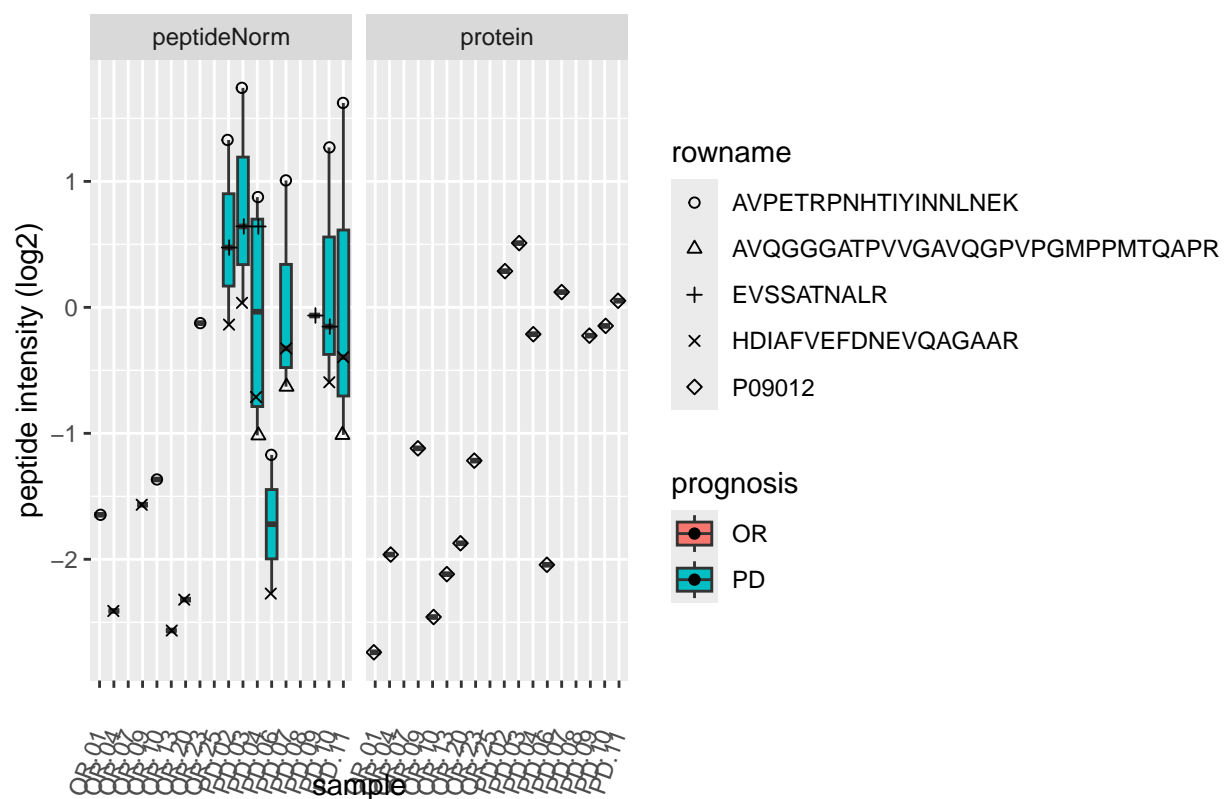




P09012

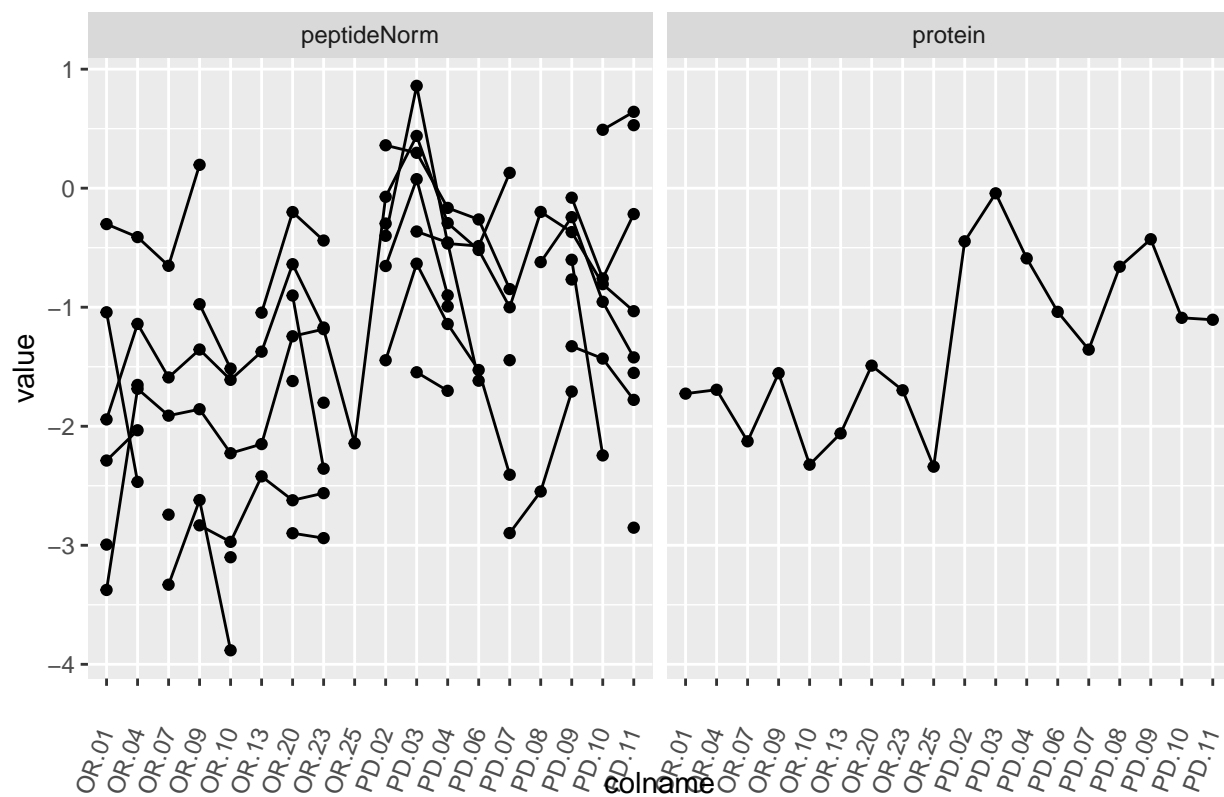


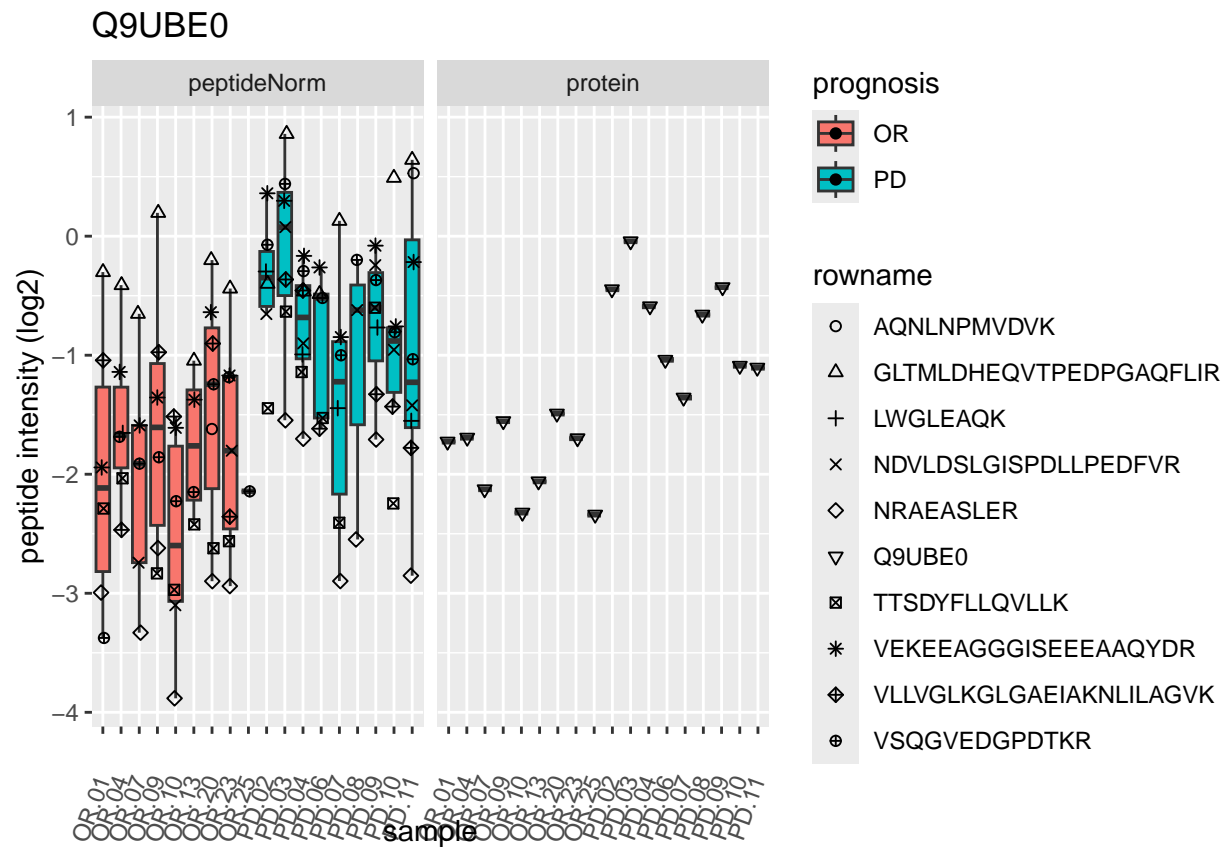
P09012



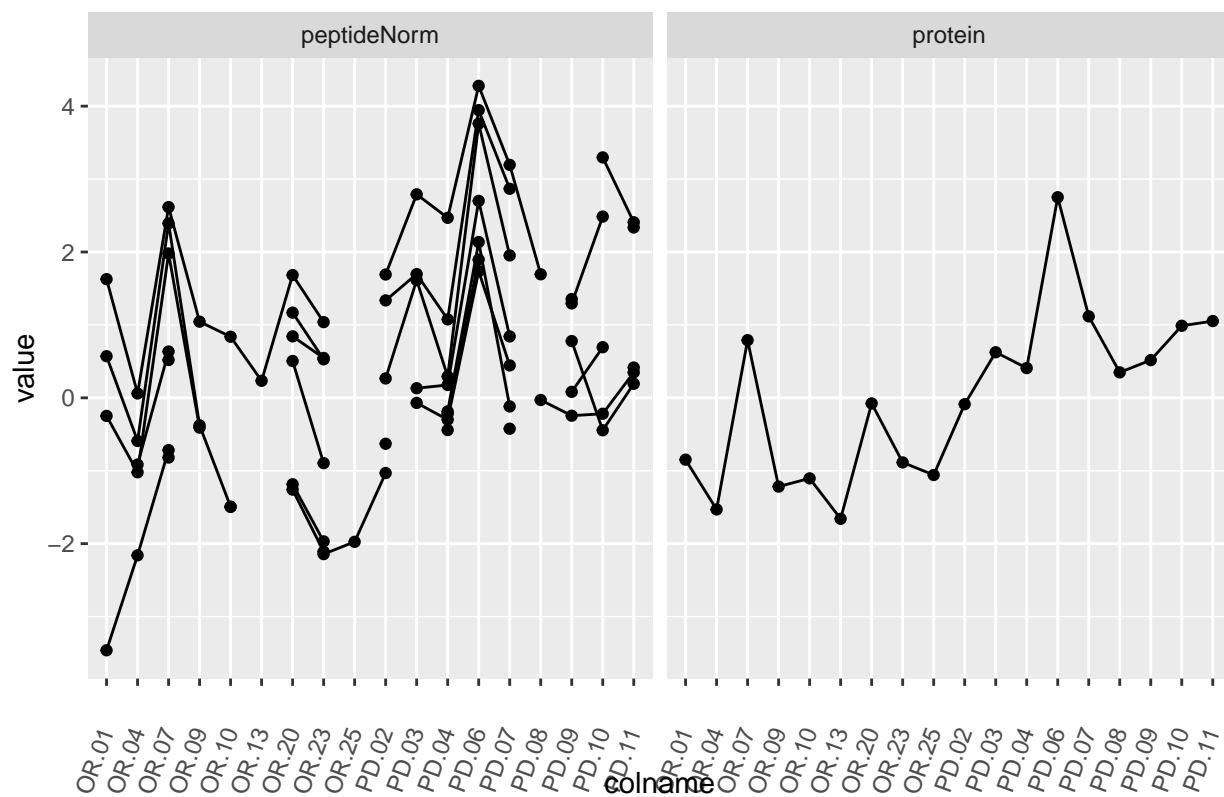


# Q9UBE0

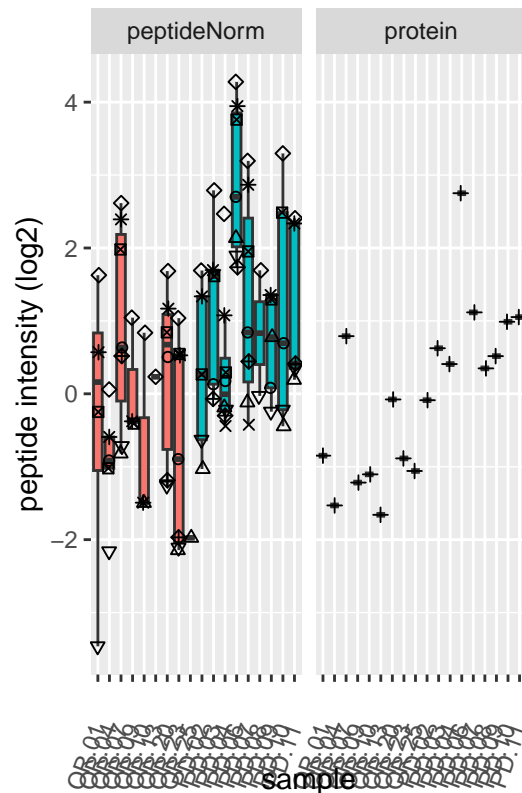




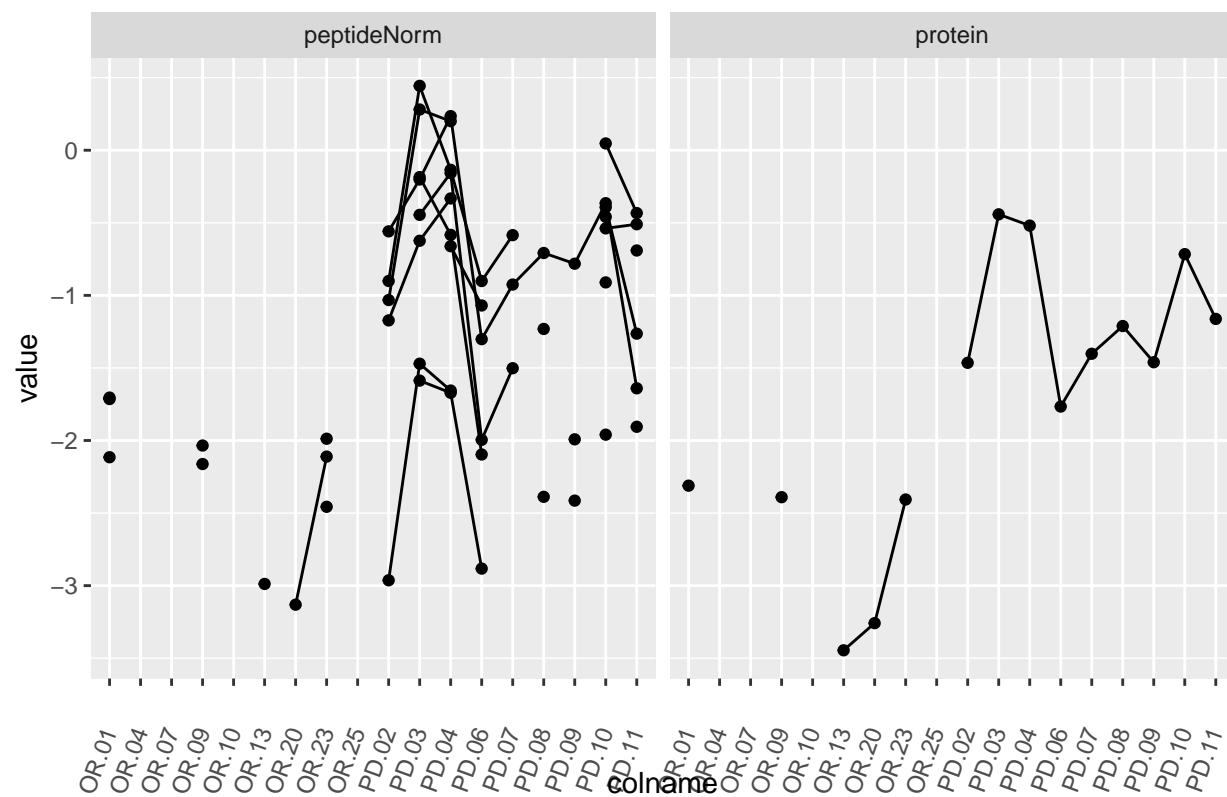
P55327



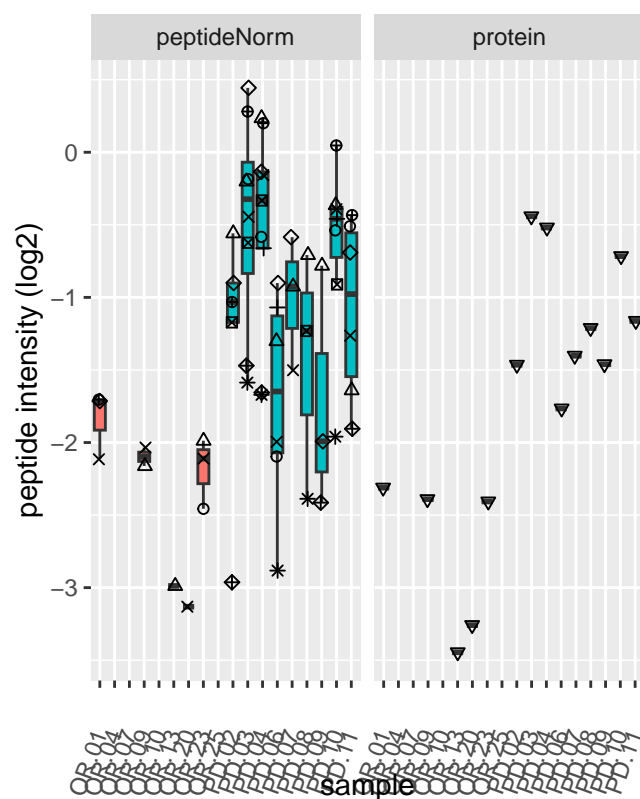
P55327



P46108



P46108



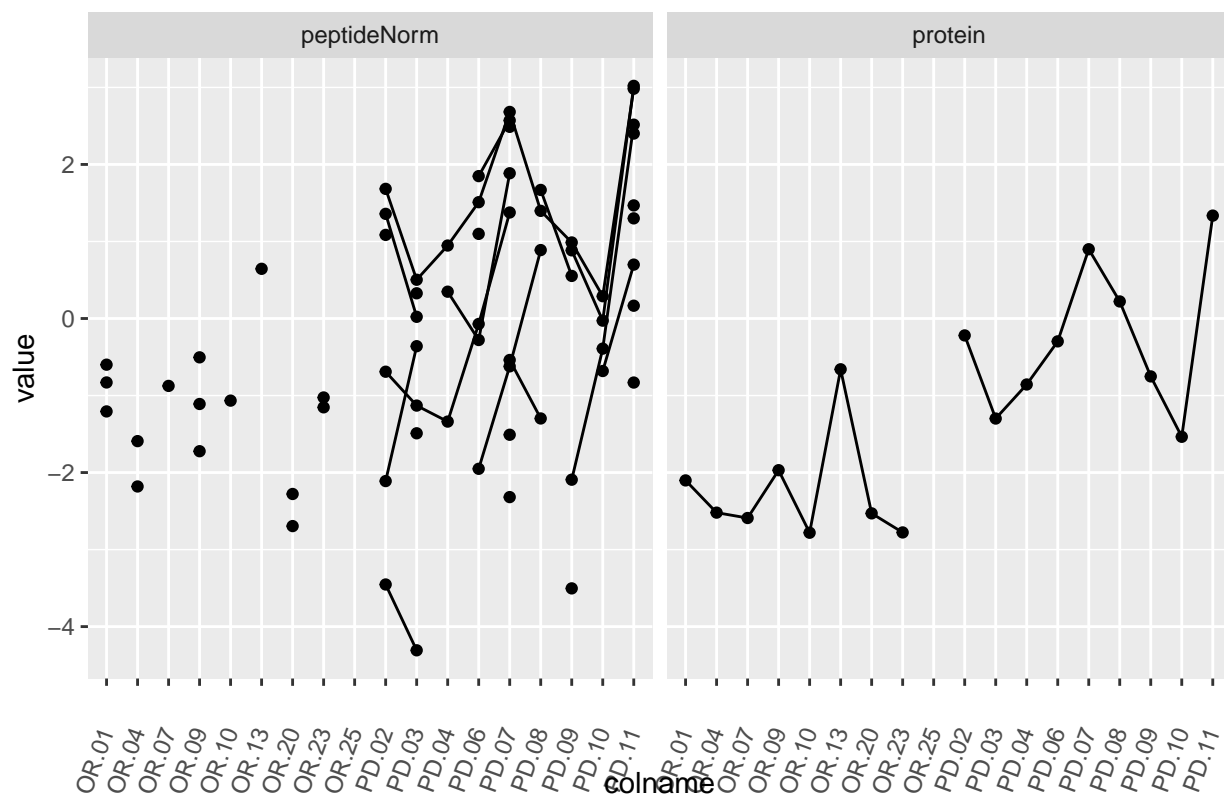
rowname

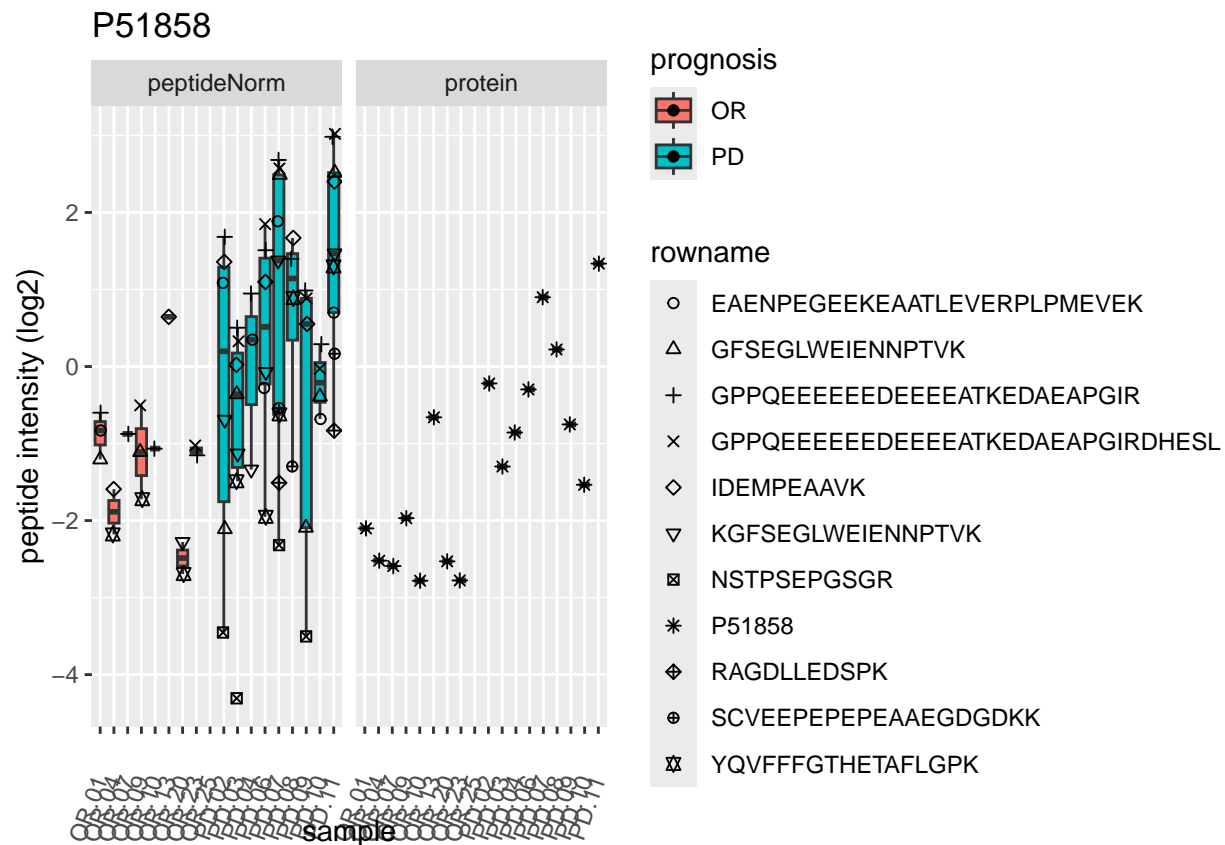
- ALFDFNGNDEEDLPFK
- △ DSSTSPGDYVLSVSENSR
- + HGVFLVR
- × IGDQEFDSLPALEFYK
- ◇ IHYLDTTTLIEPVSR
- ▽ P46108
- ⊠ QEAVALQQR
- \* QGSGVILR
- ⊞ TALAELVGELVK
- ⊕ VSHYIINSSGPRPPVPPSPAQPPPGVSPSR

prognosis

- OR
- PD

P51858



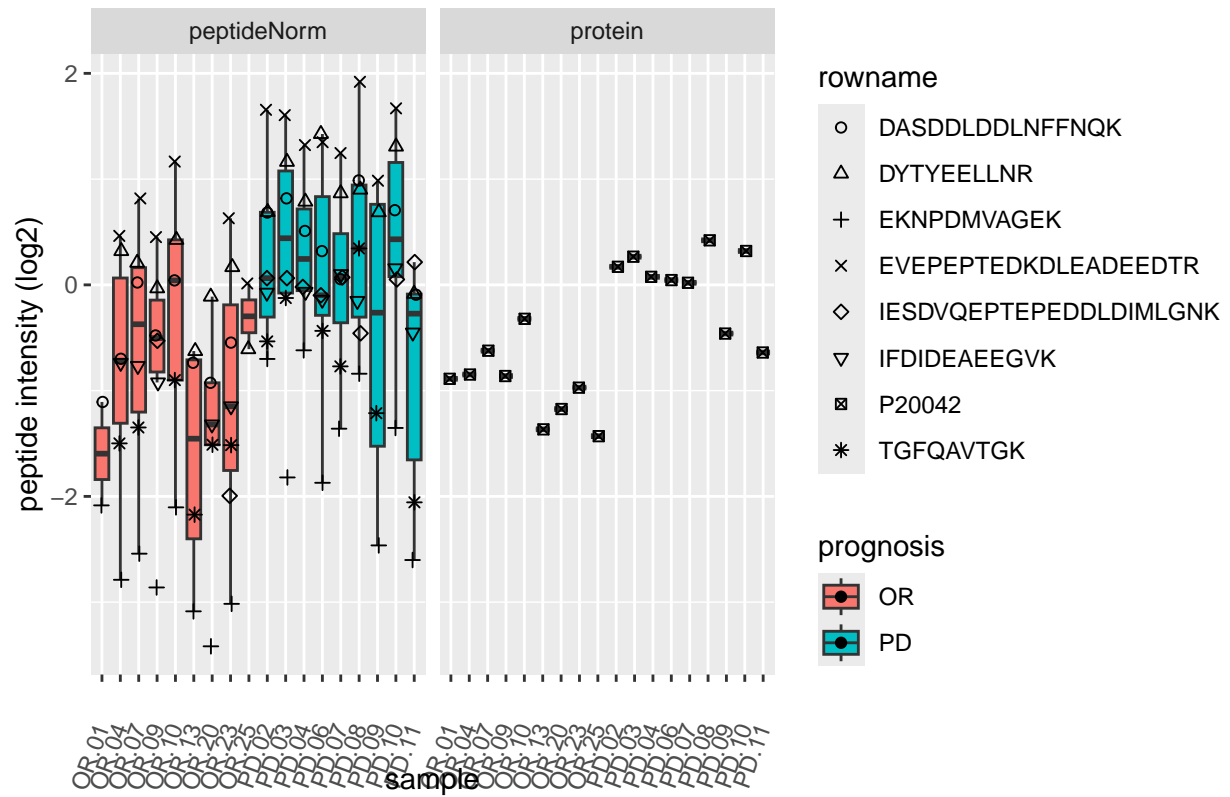




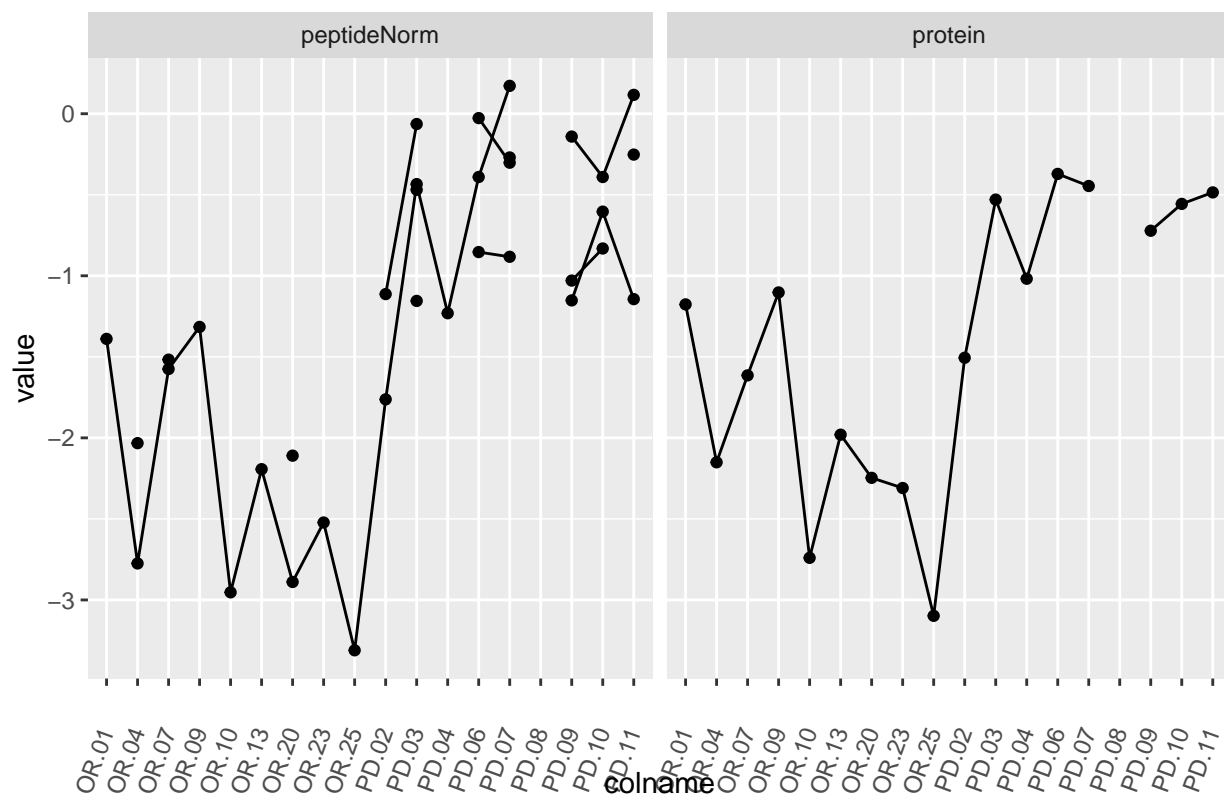
P20042



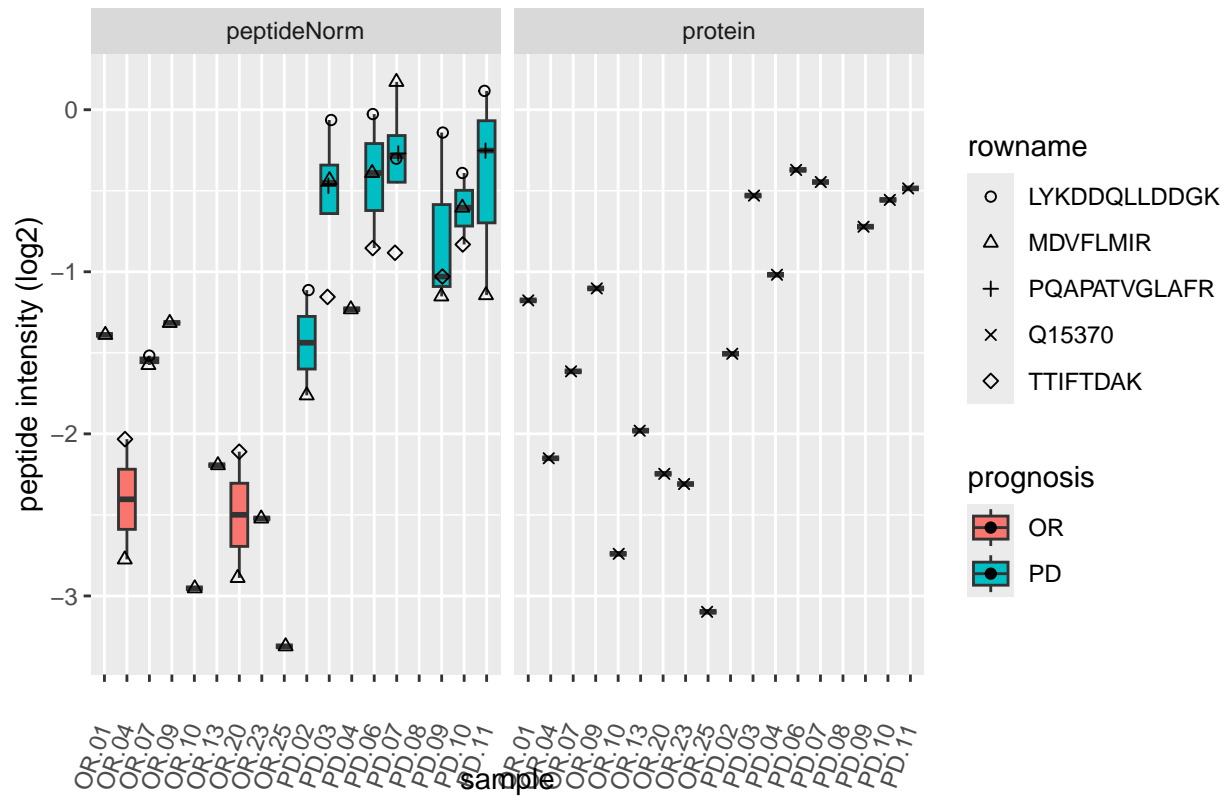
P20042



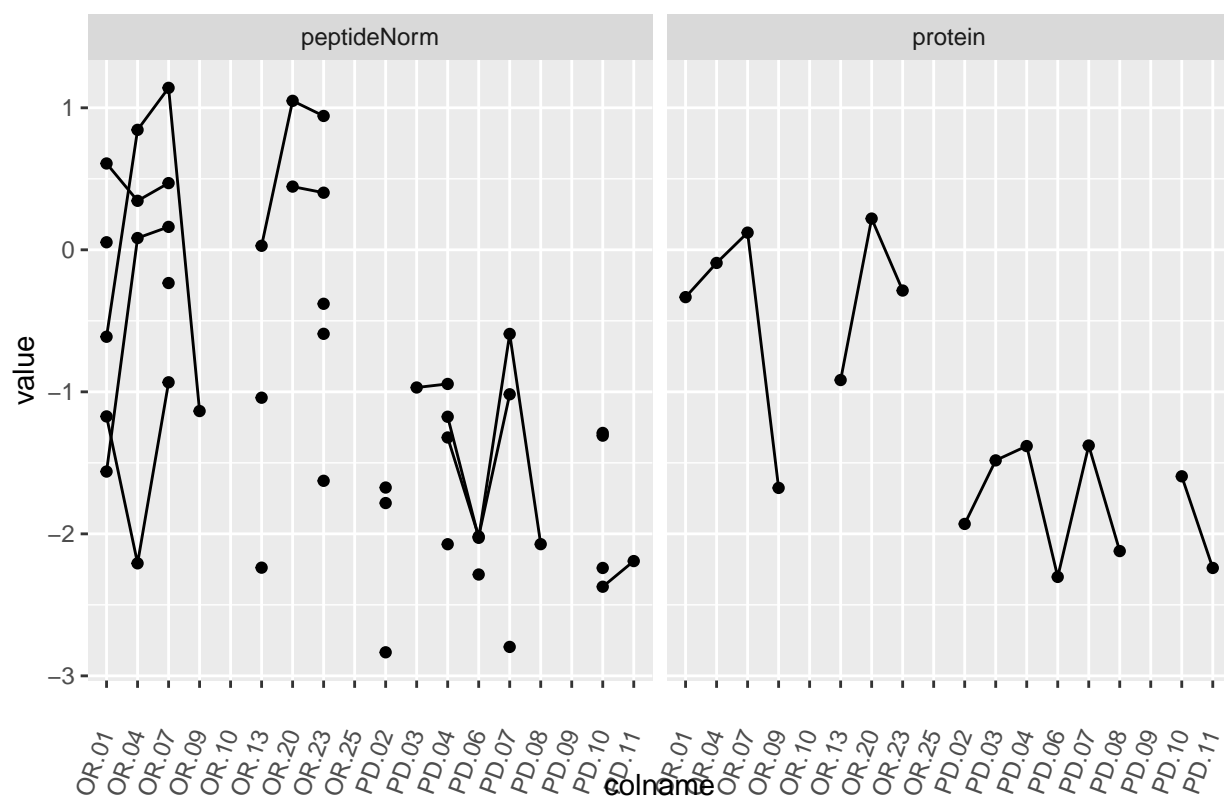
Q15370



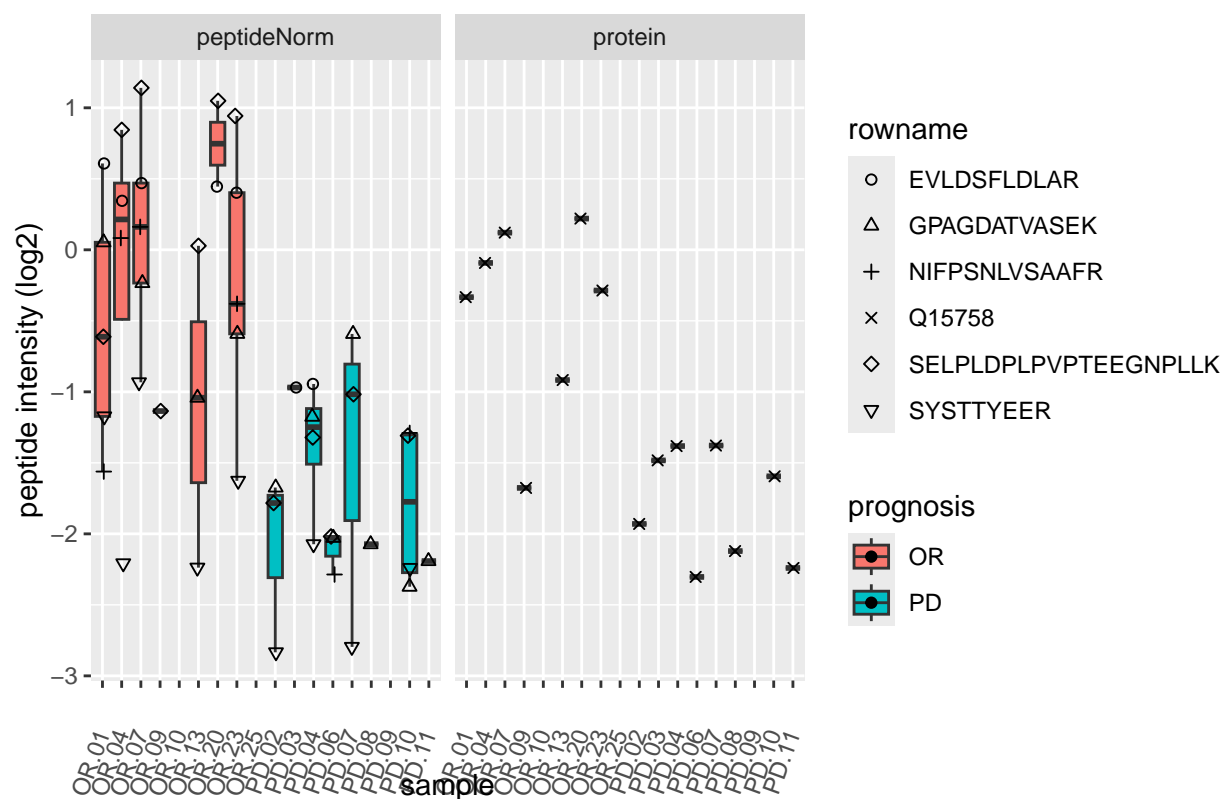
Q15370



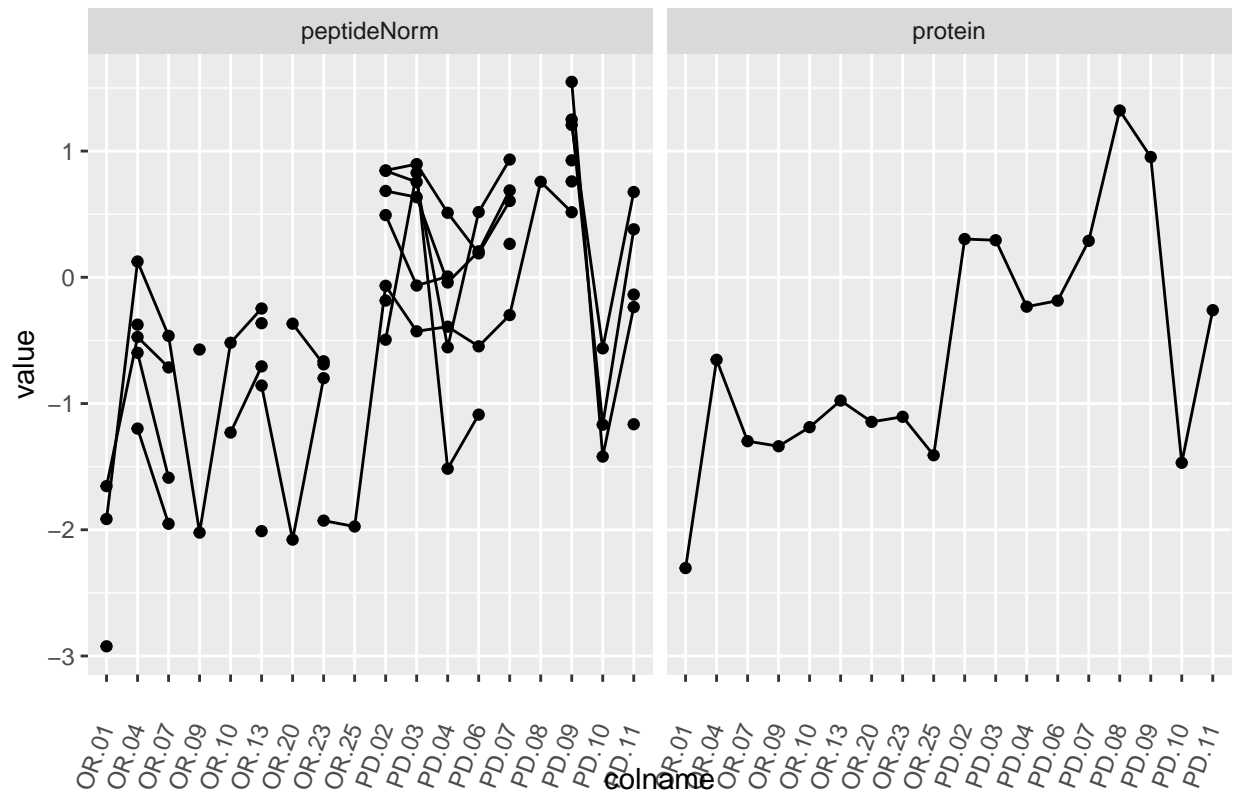
Q15758



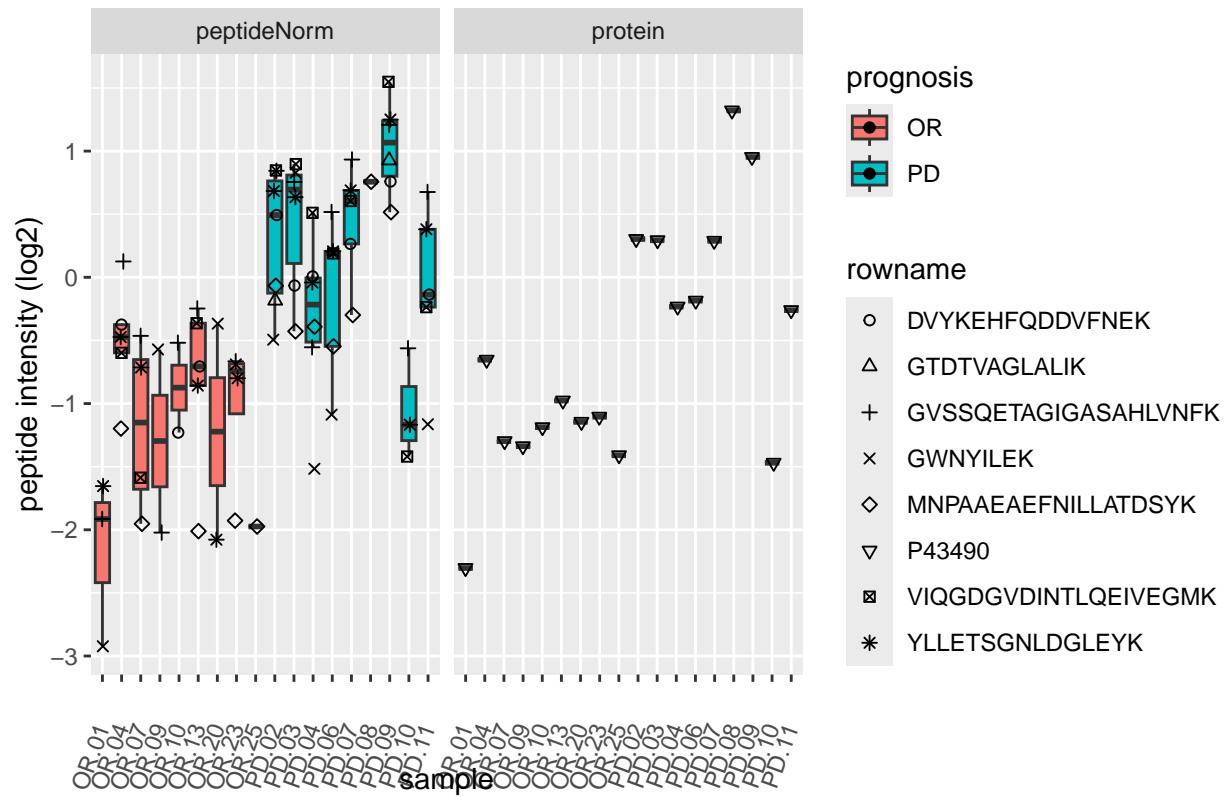
Q15758



P43490

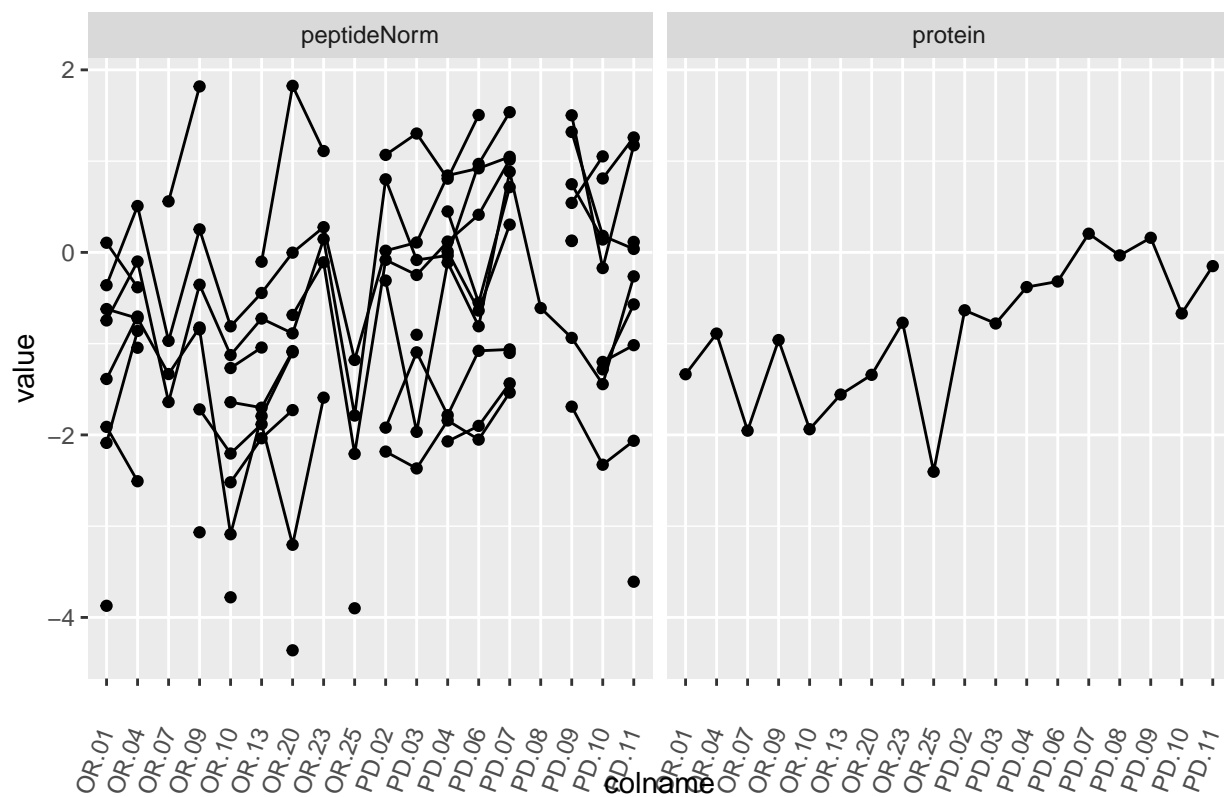


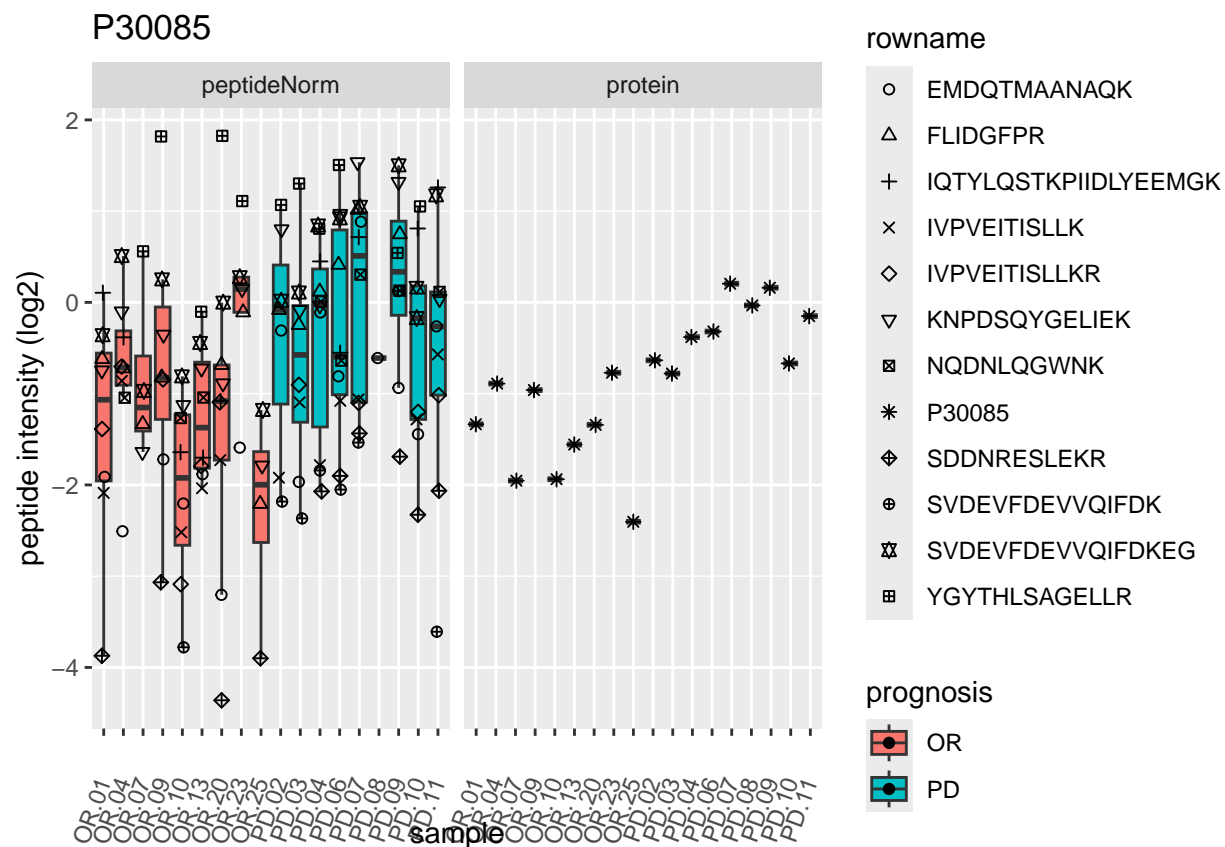
P43490



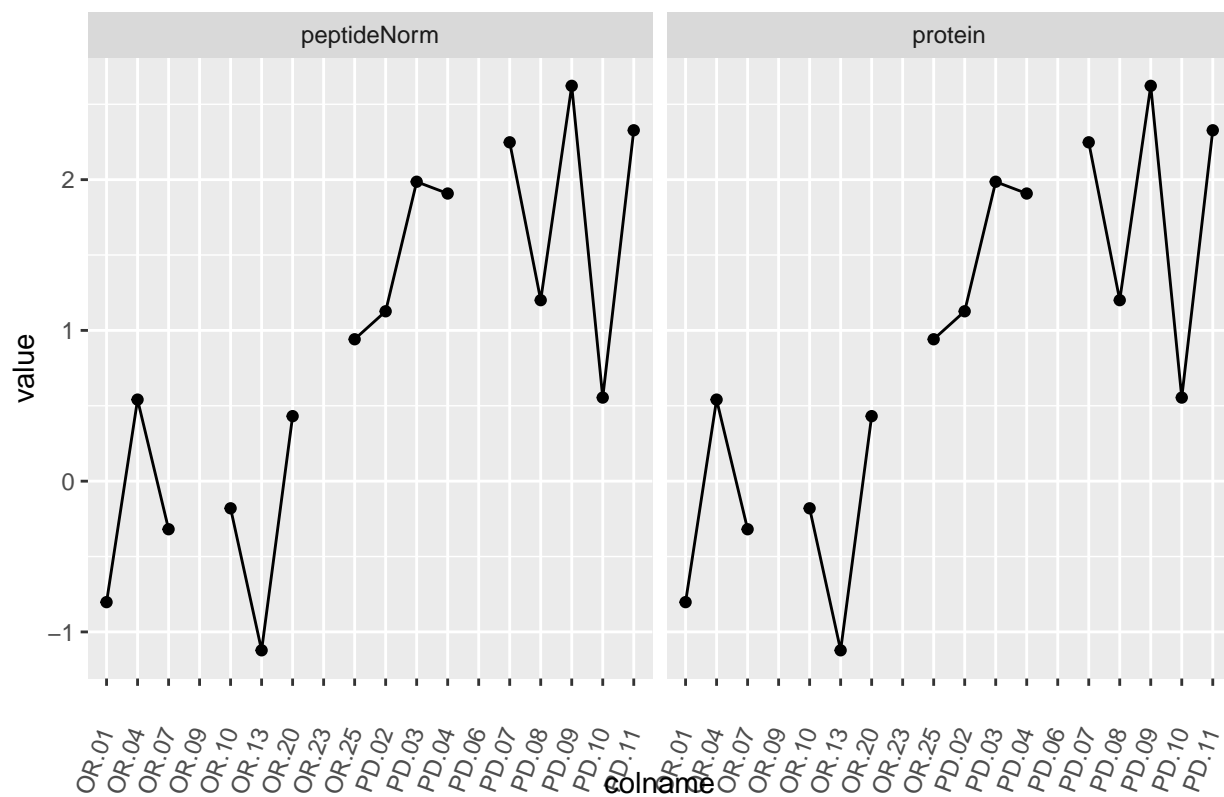


P30085

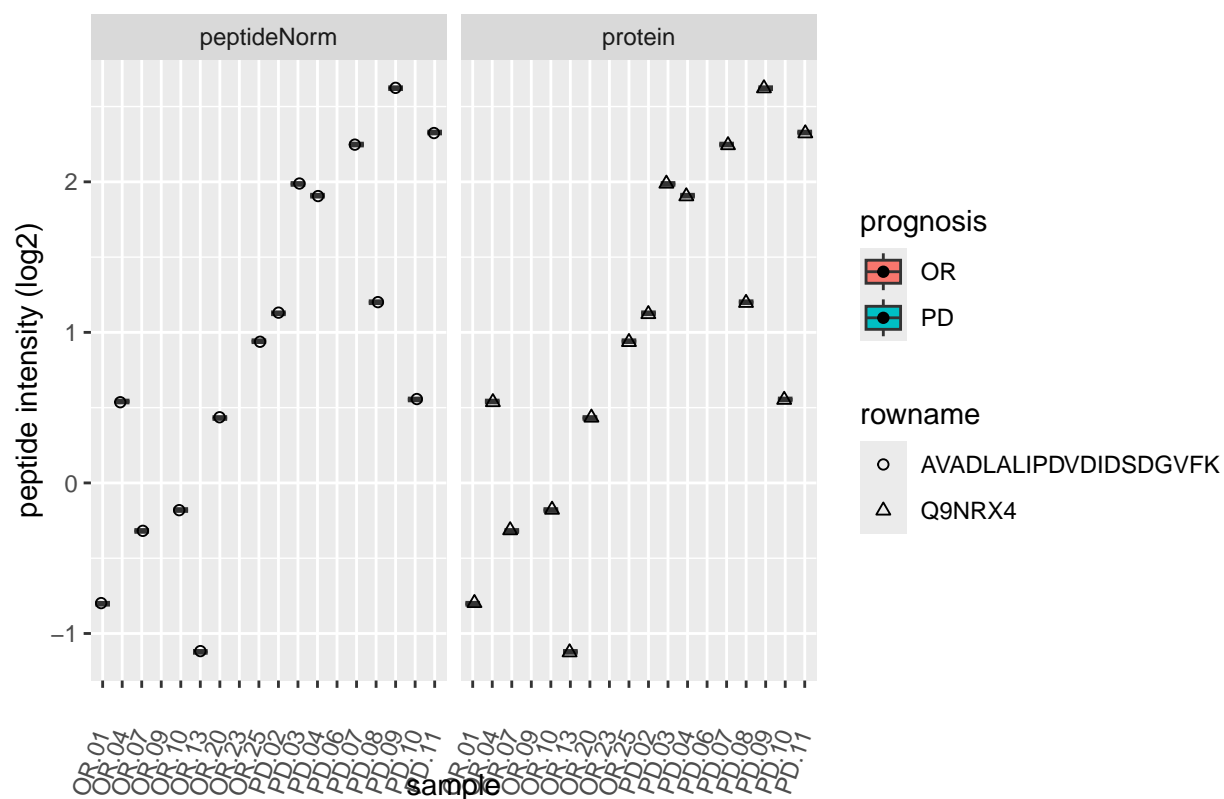




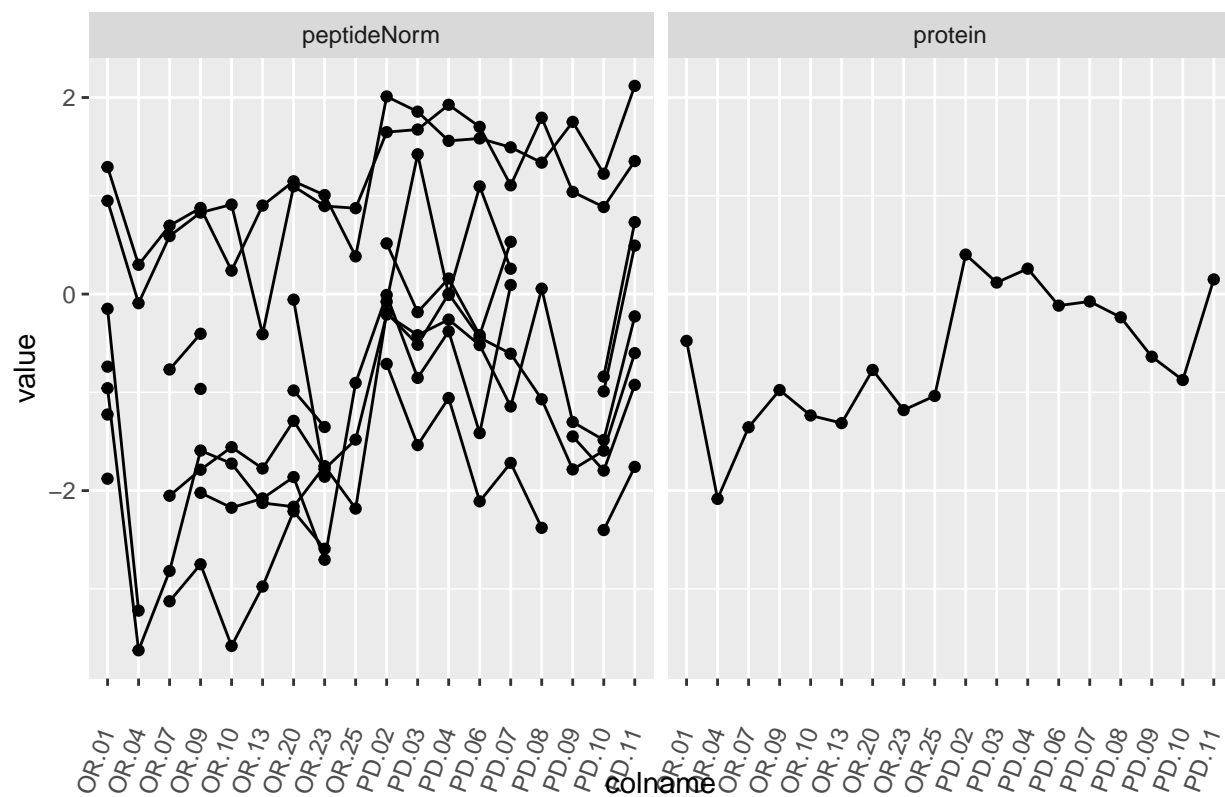
Q9NRX4

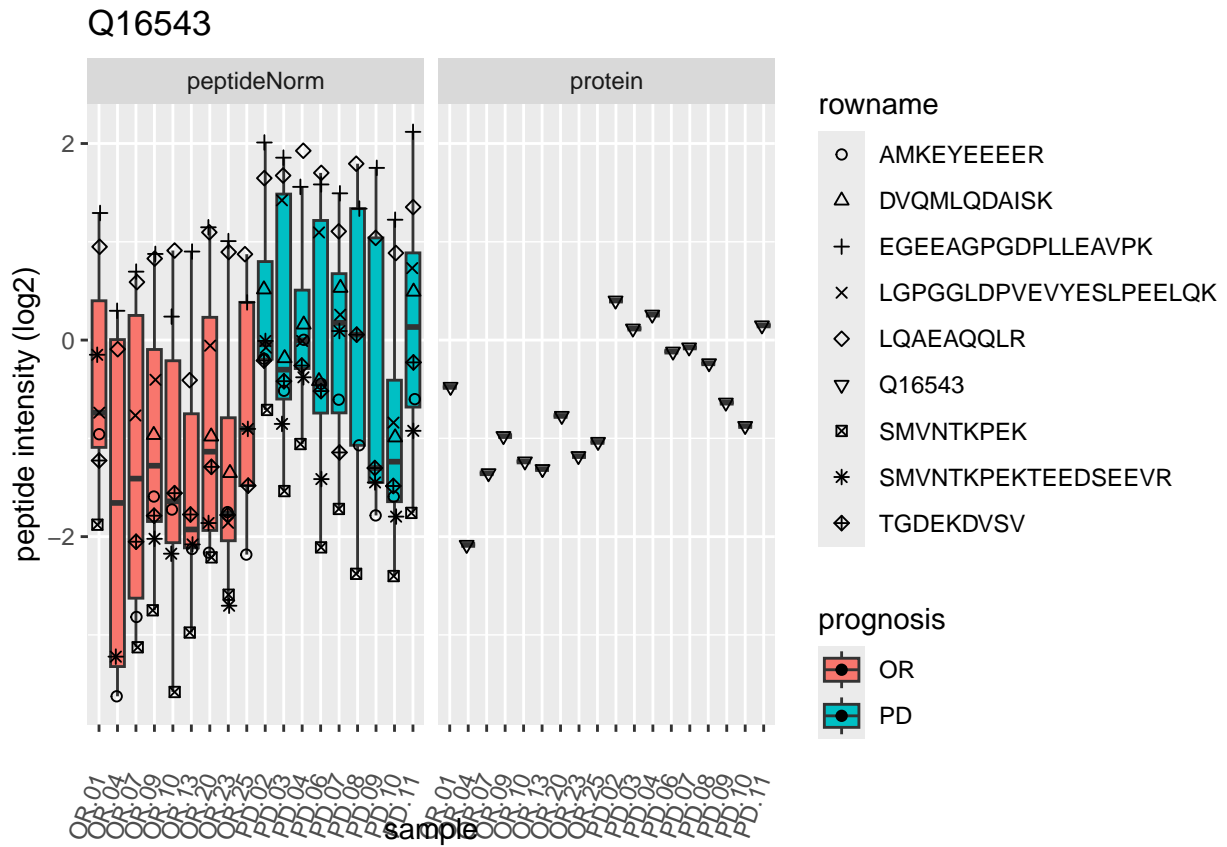


# Q9NRX4

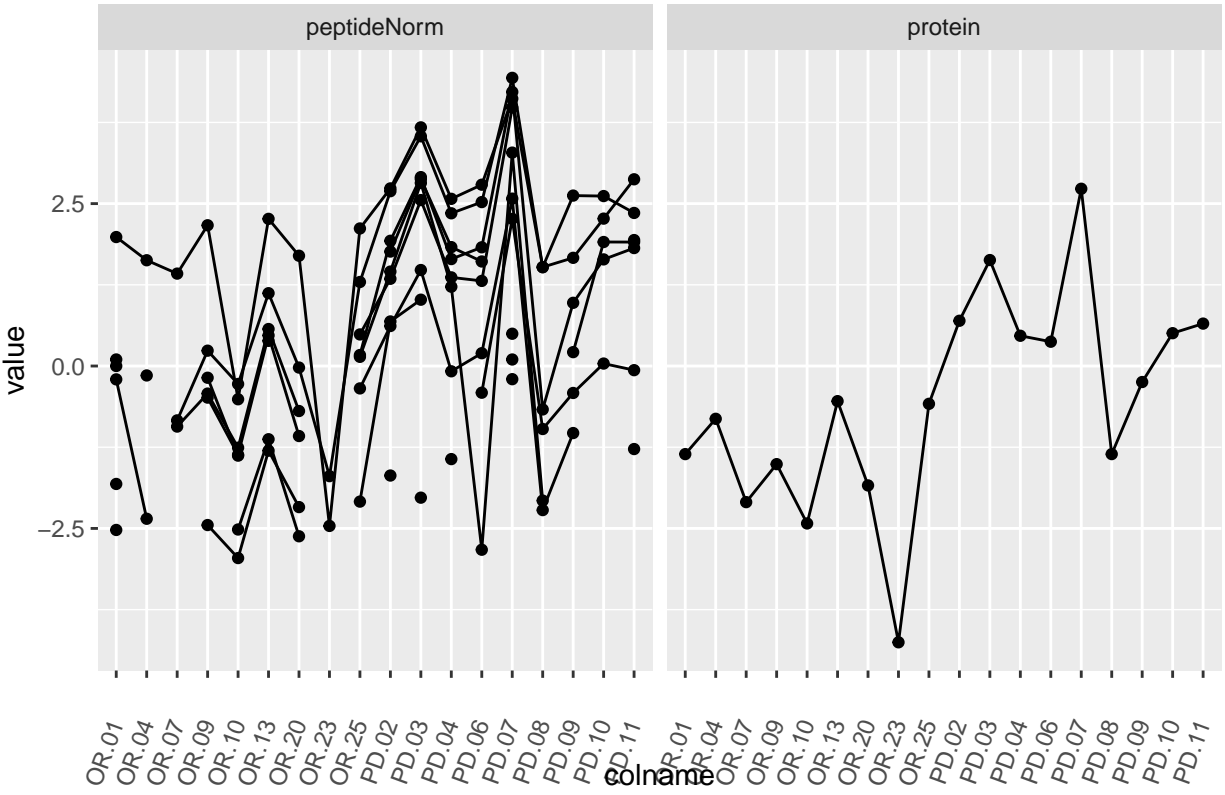


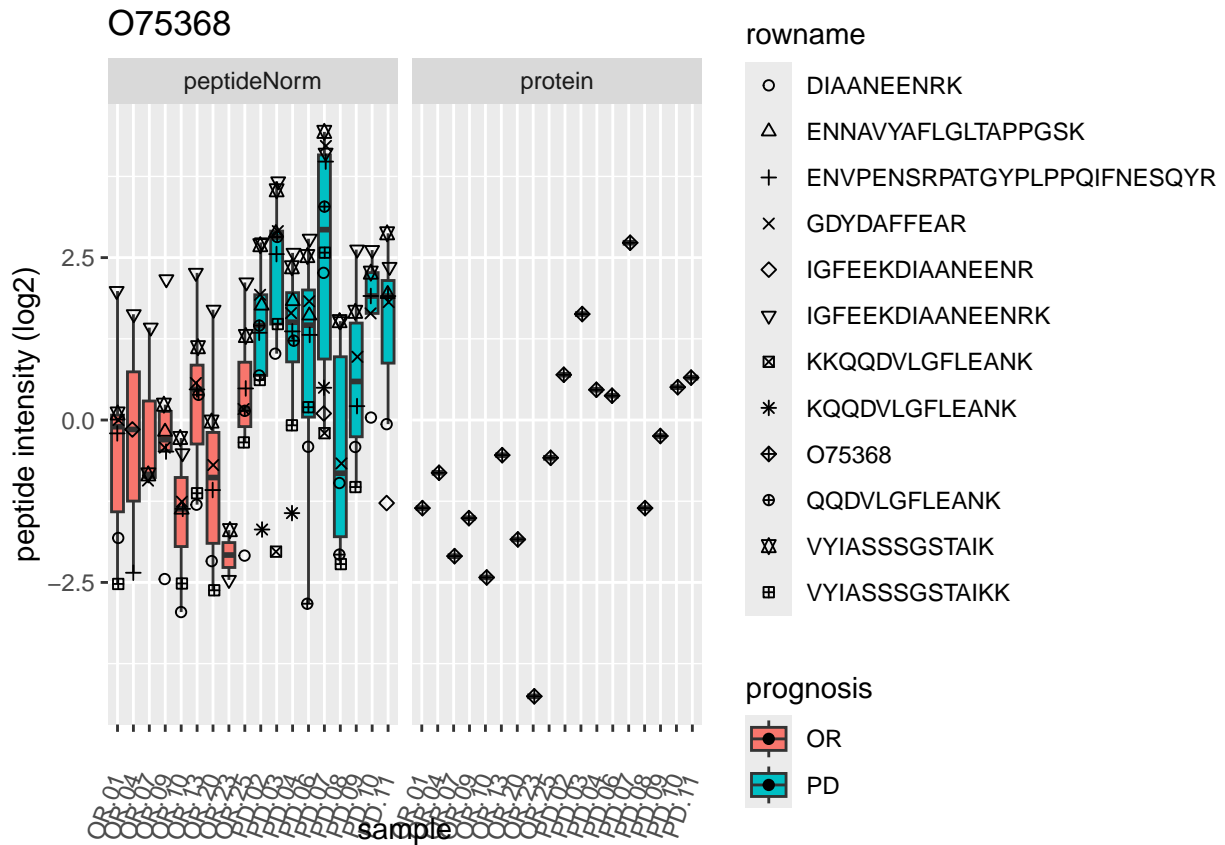
Q16543





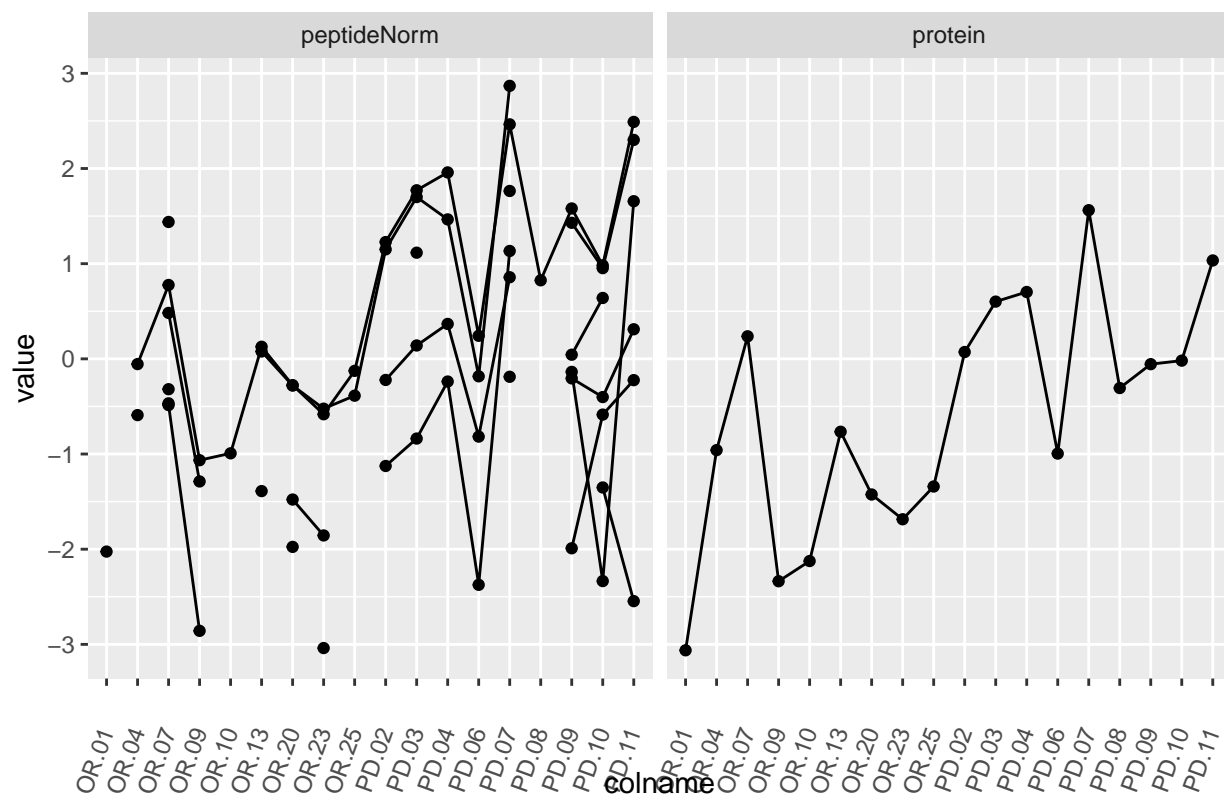
O75368



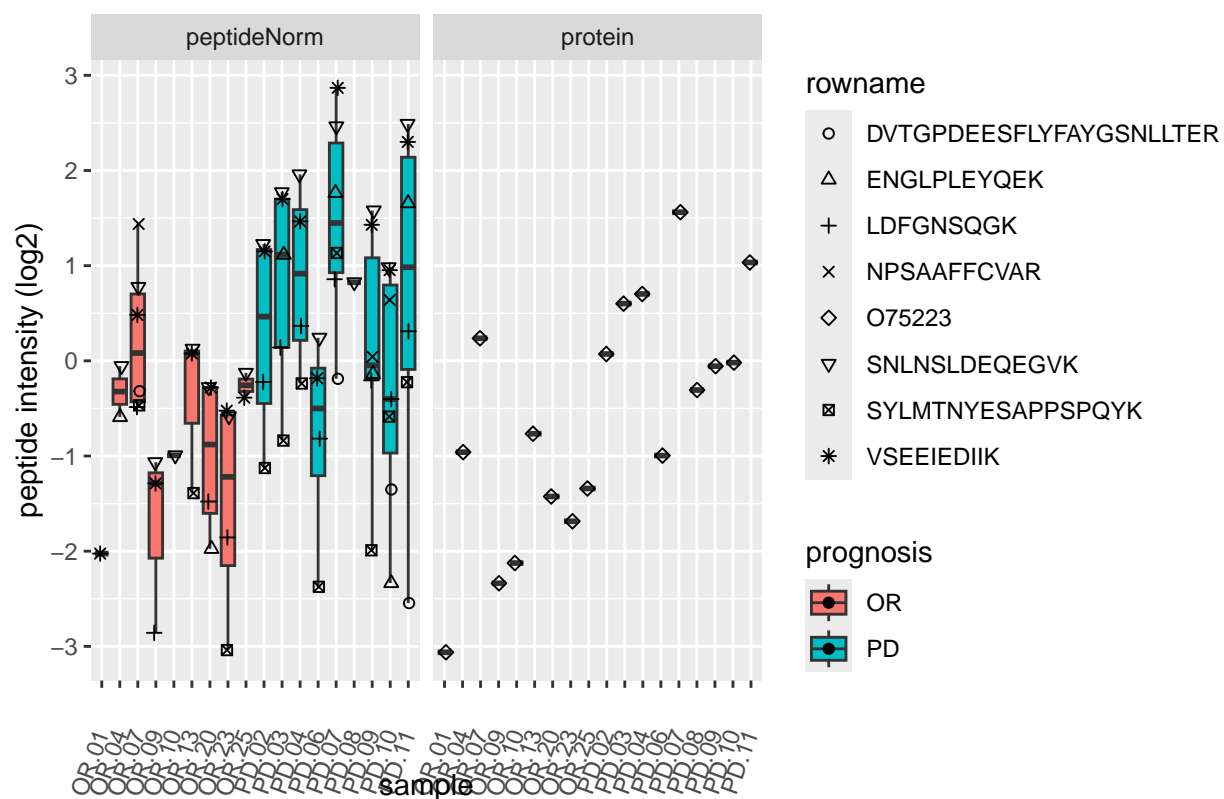




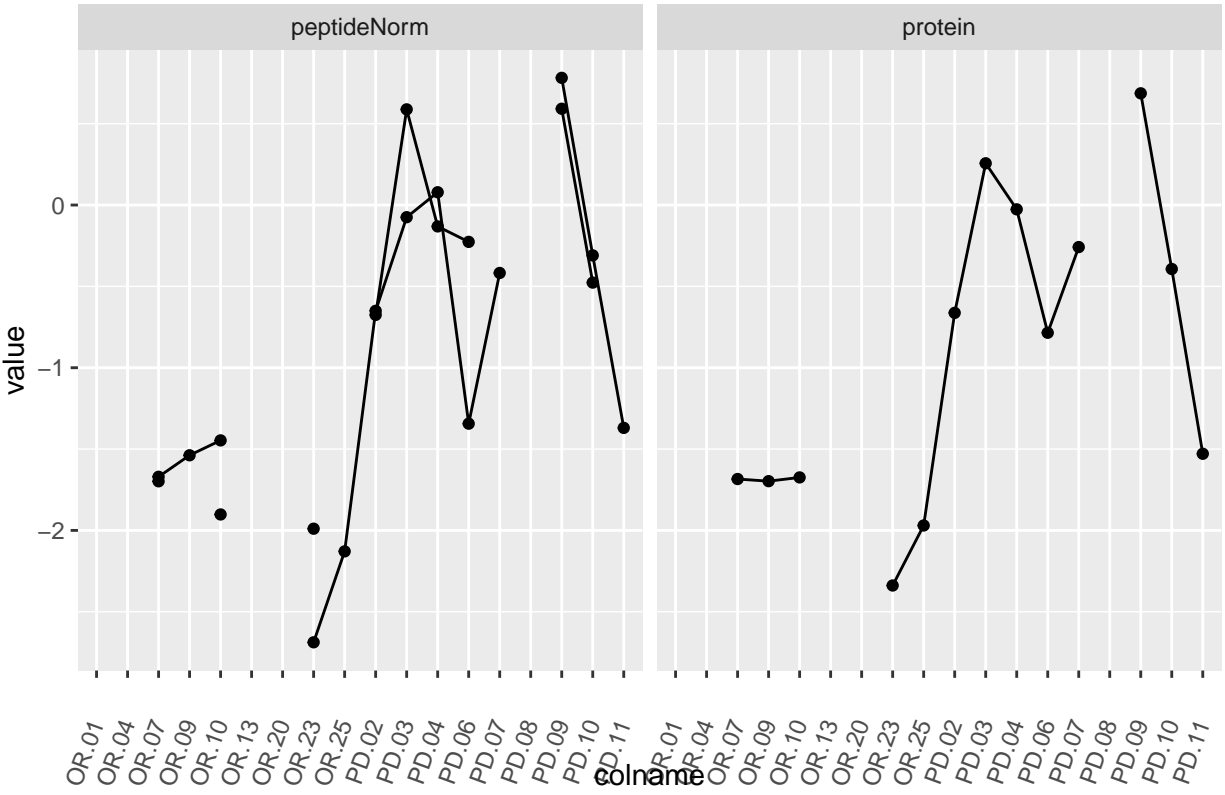
O75223



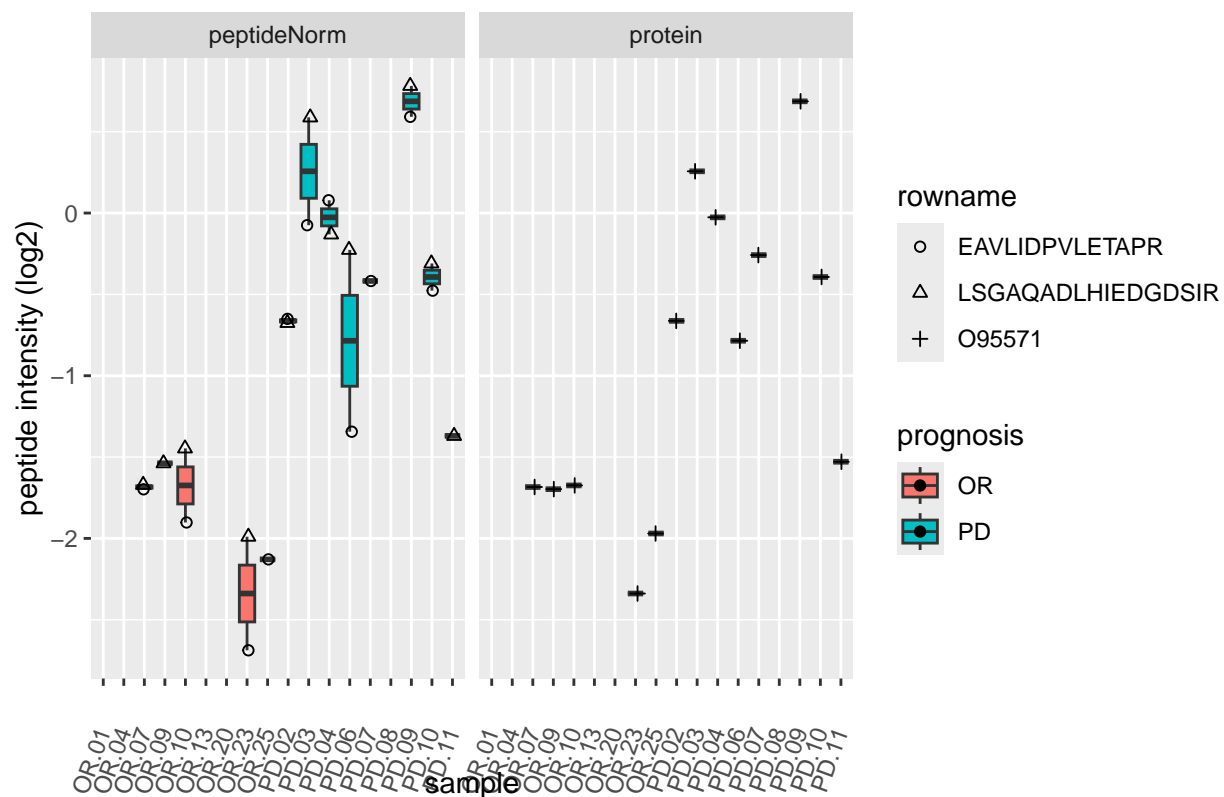
O75223



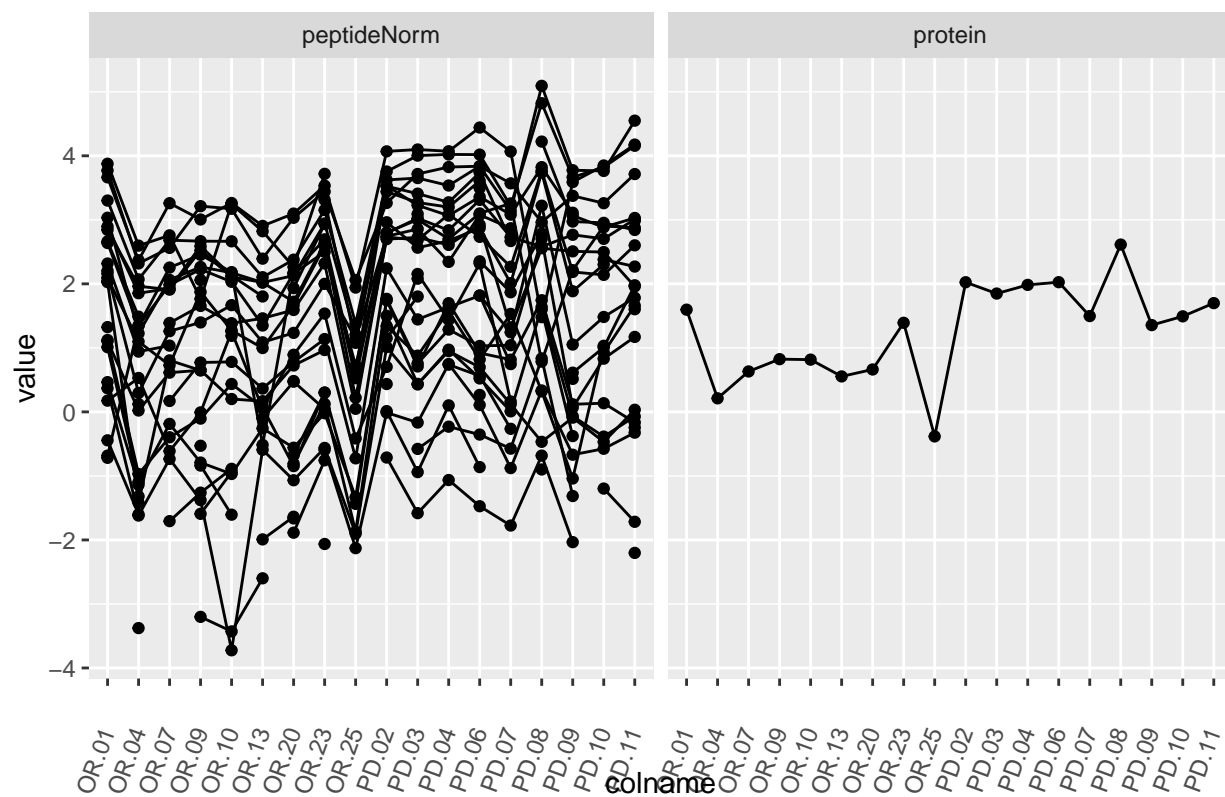
O95571

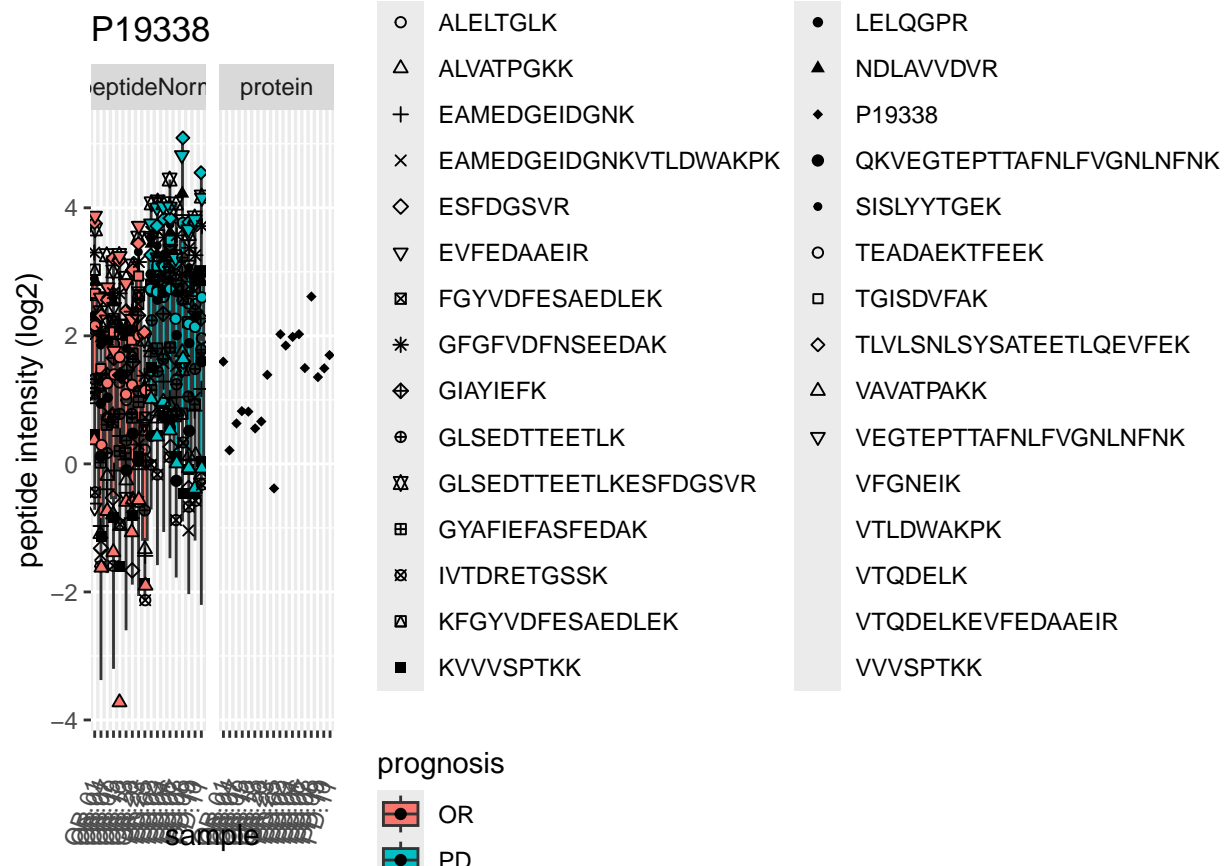


O95571

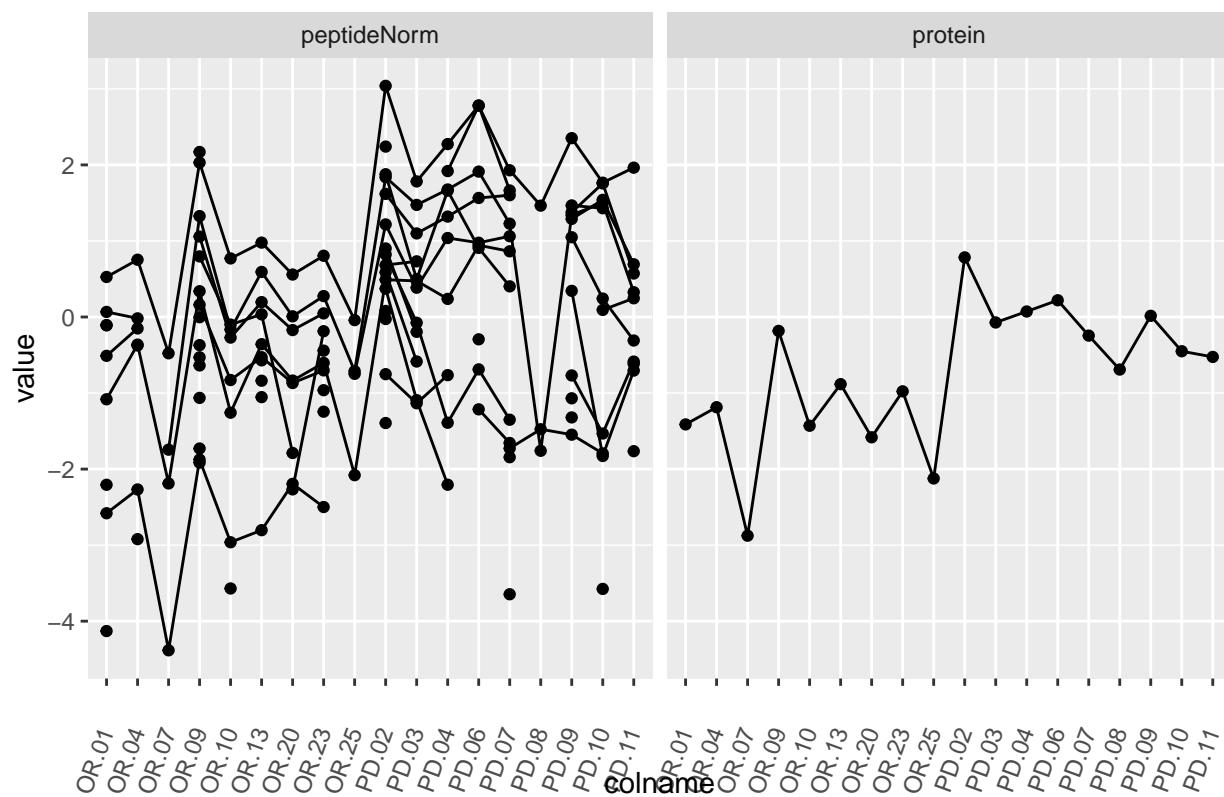


P19338

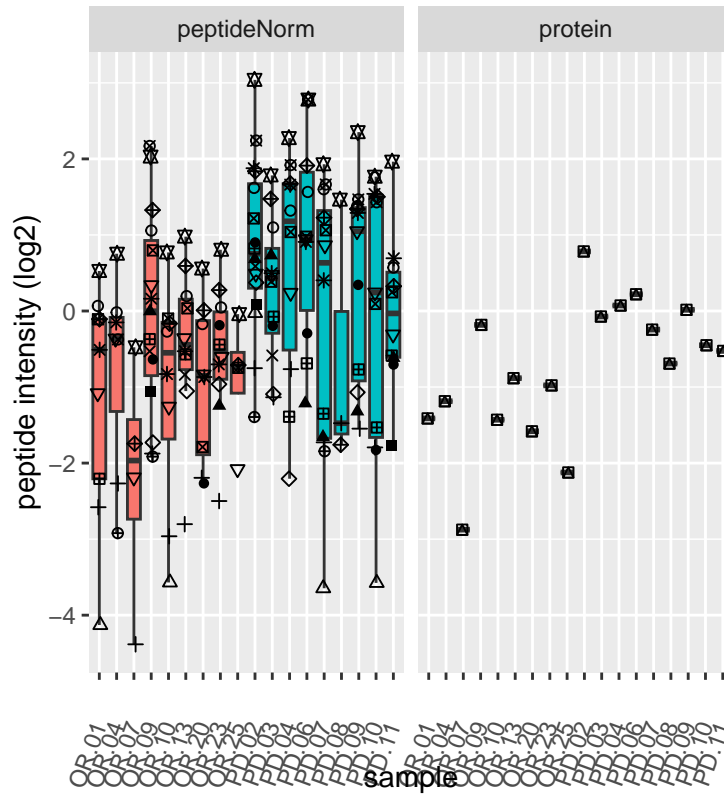




P22307



P22307



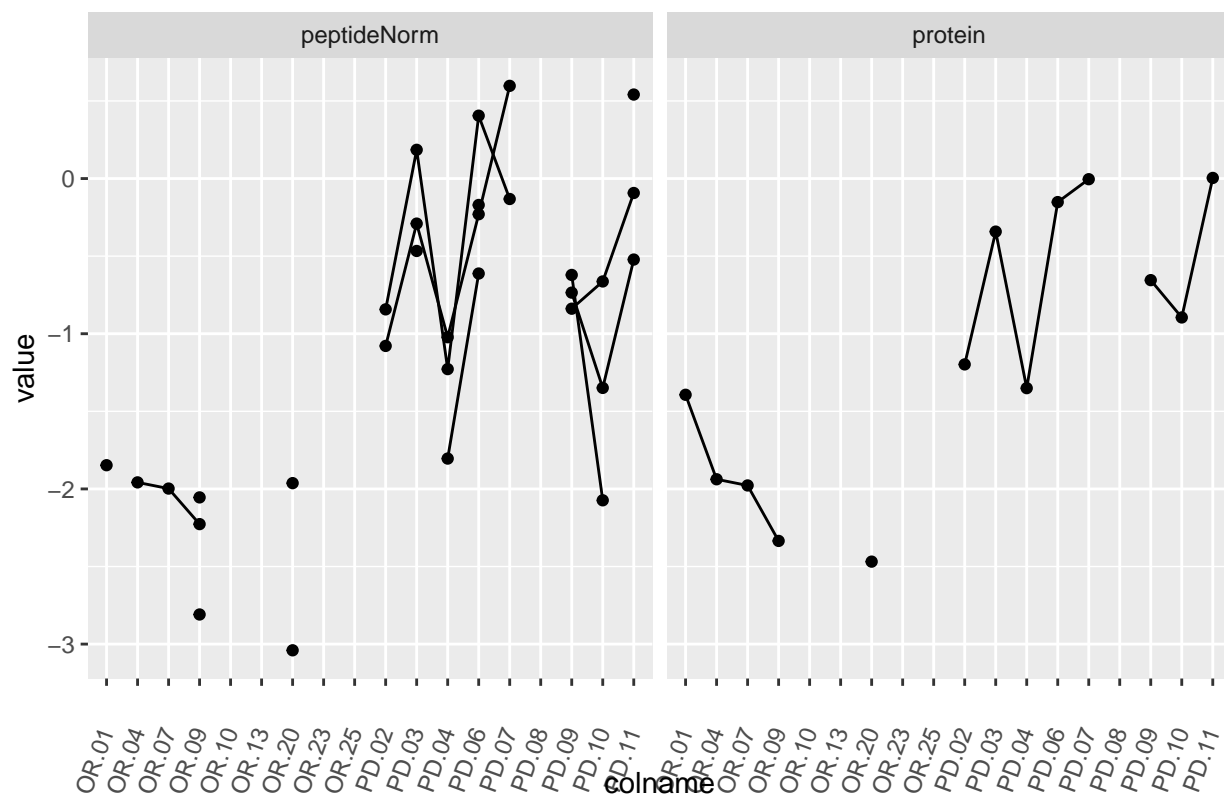
PD

rowname

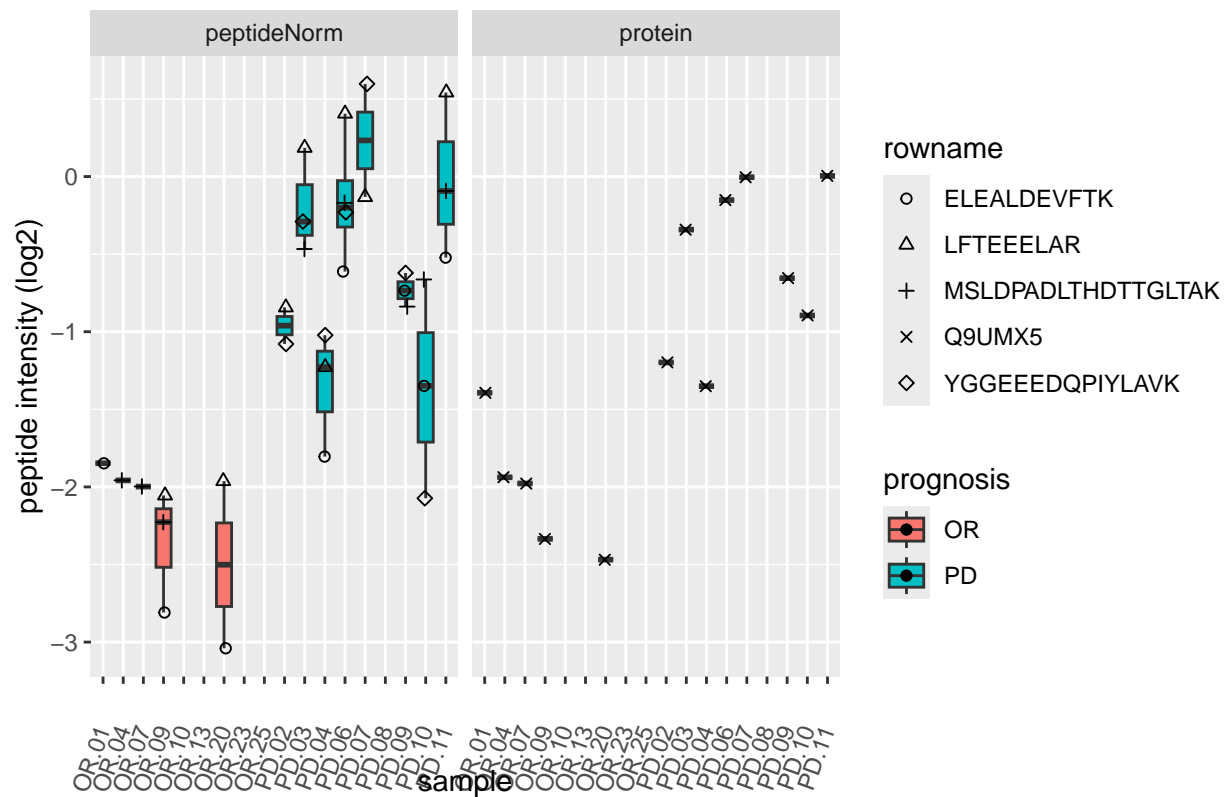
- EATWVVDVK
- △ FVKPGAENSR
- + GSVLPNSDKK
- × HSVNNPYSQFQDEYSLDEVMSK
- ◇ HVDLLINK
- ▽ IGGIFAFK
- ▣ ITGNMGLAMK
- \* KLEEEGEQFVK
- ⊕ KLEEEGEQFVKK
- ⊗ LEEEGEQFVK
- ⊠ LQNLQLQPGNAK
- ▤ MGFPEAASSFR
- ⊞ MNPQSAFFQGK
- ▥ P22307
- VFVVGVGMTK
- WVINPSGGIISK



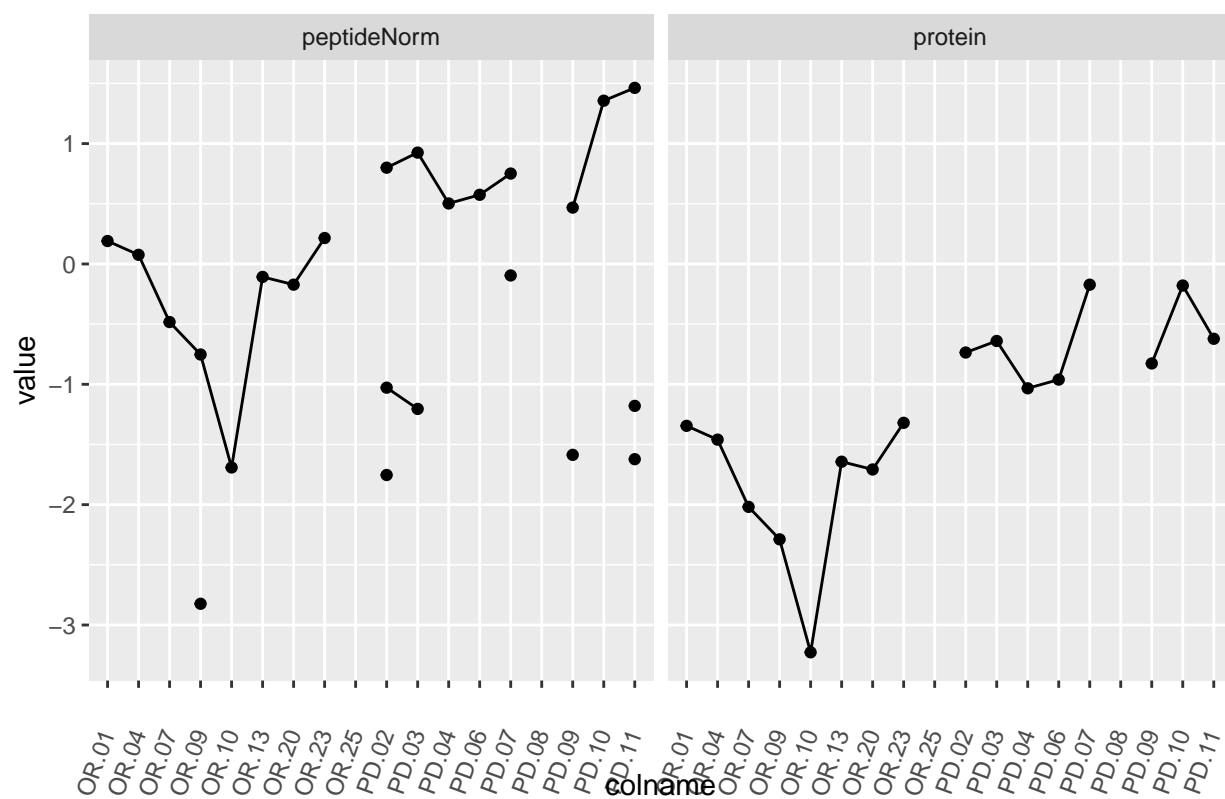
Q9UMX5



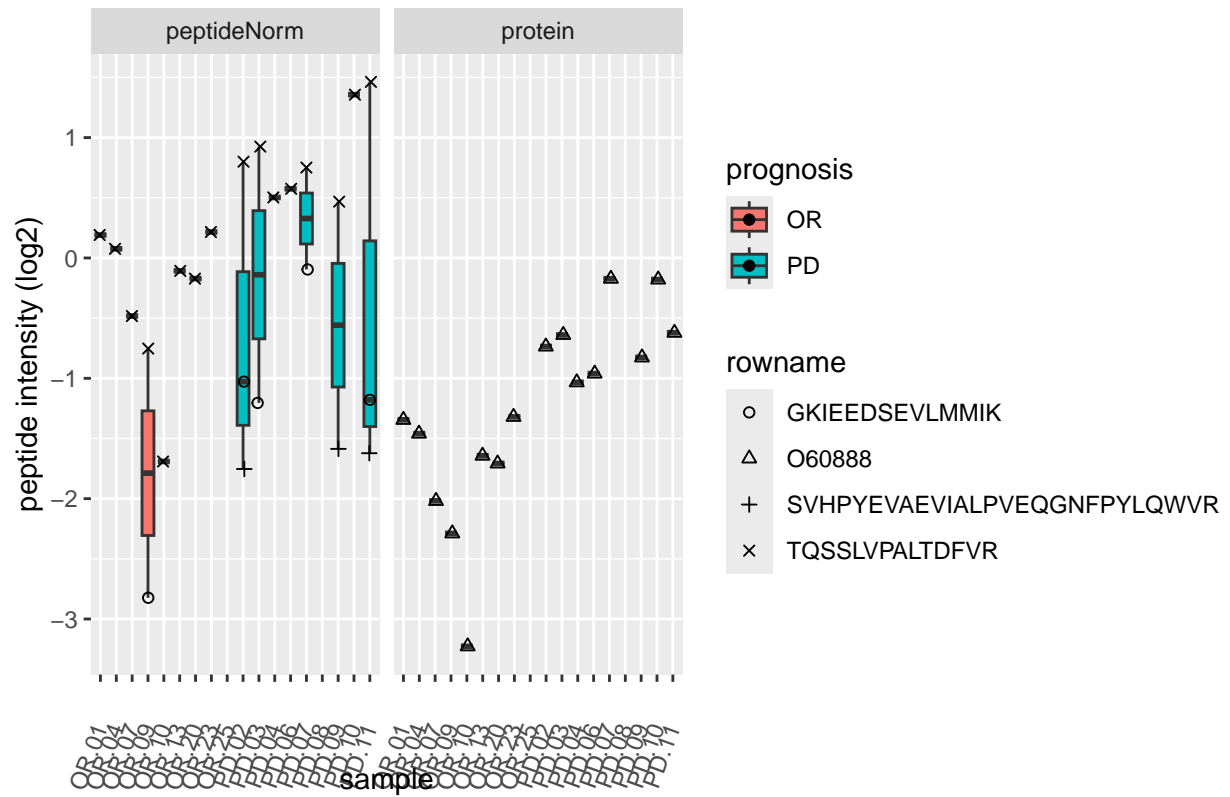
Q9UMX5



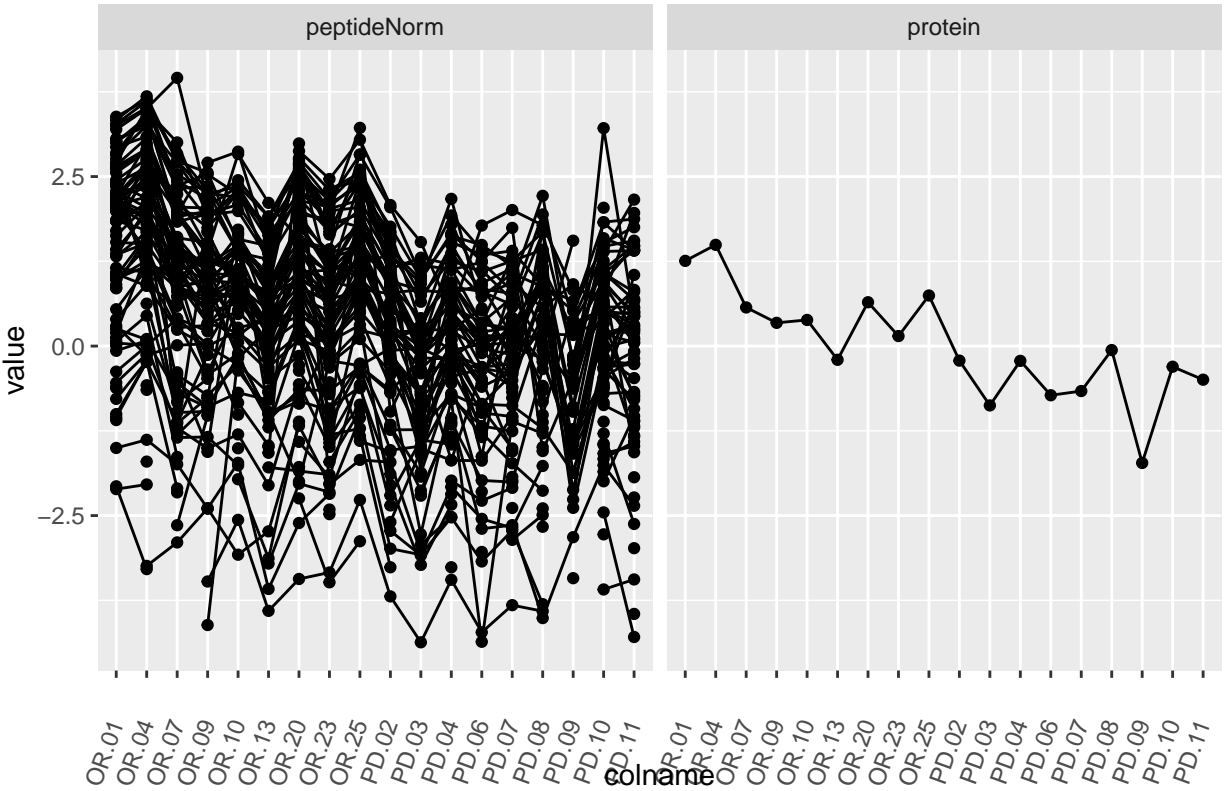
O60888



O60888

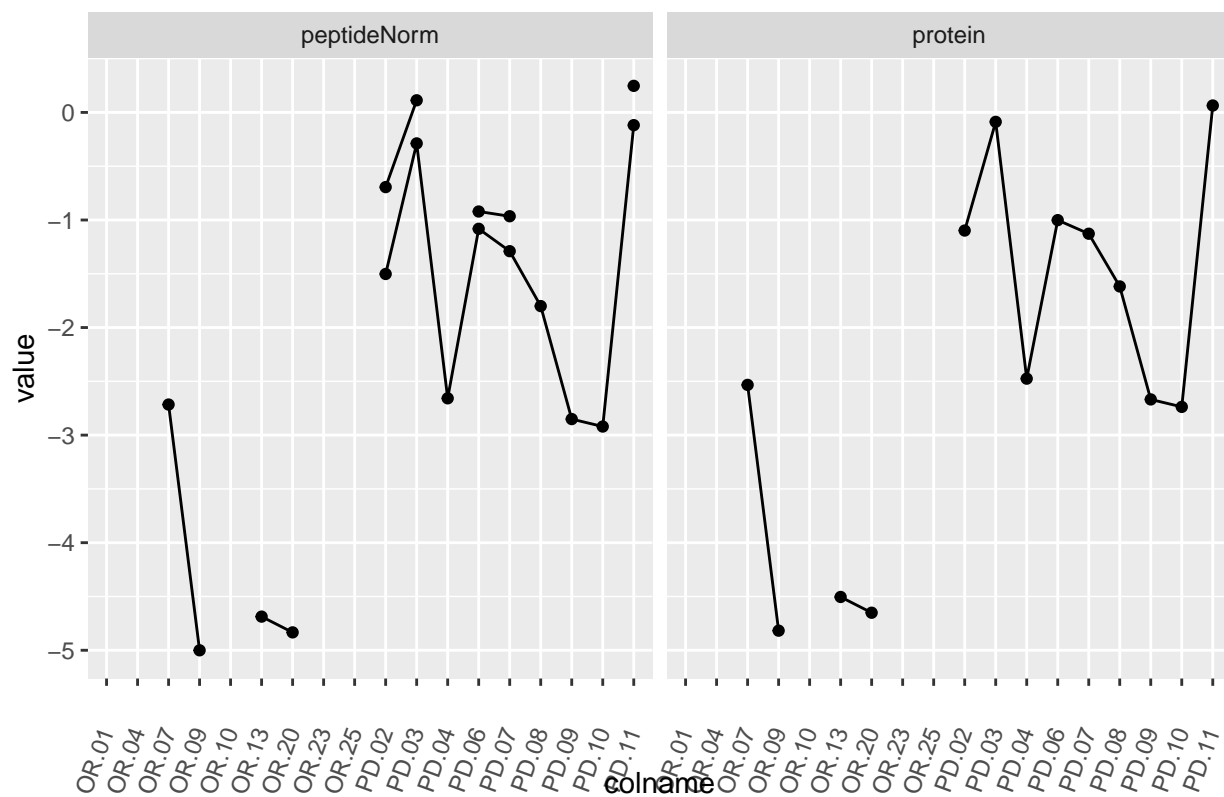


Q14980

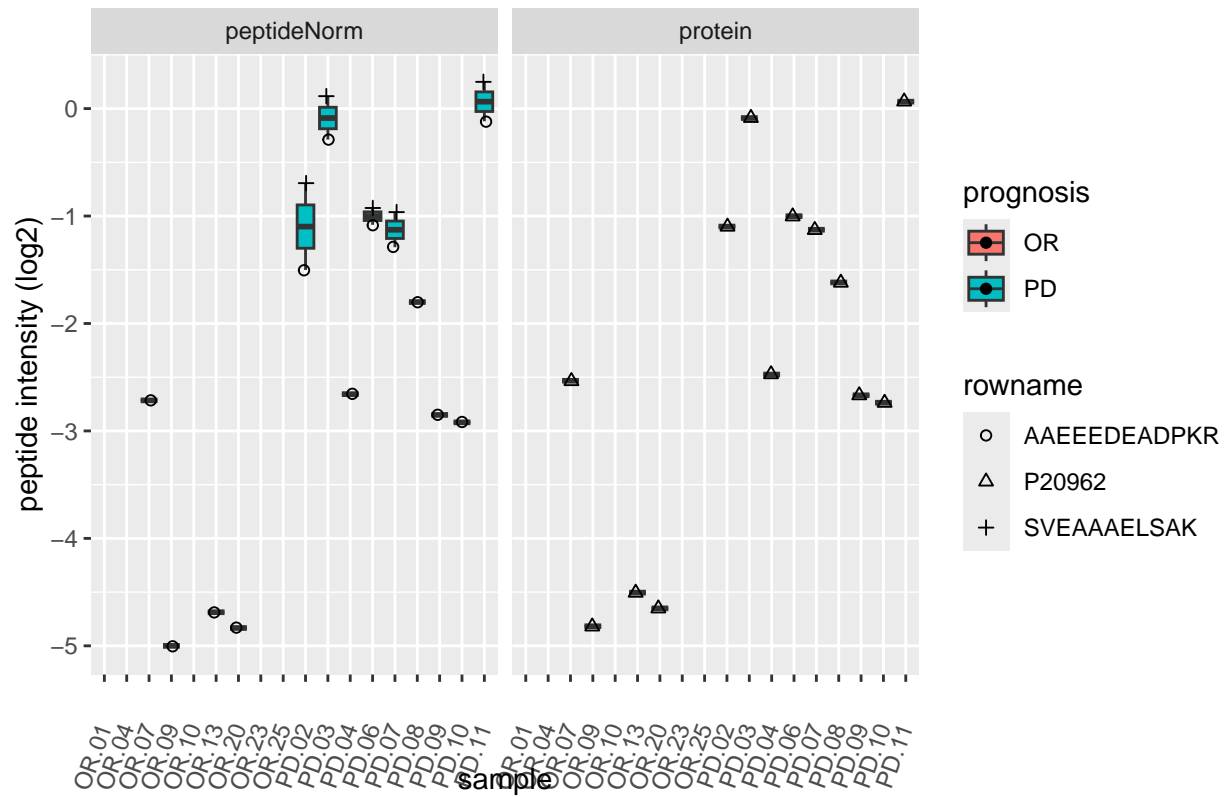


	8	IRQVEQLEGGGR	9	LEQALFAGNGAR	9	QLEQILF
	·	HREELEQSK	9	LPPKVESLESLYFTPIPAR	K	QFCSTG
IR	(	IATTTASAATAAAIGATPR	:	LQAQLNELQAQLSQK	L	QFLEVE
	)	IHGTEEGQQILK	:	LQNALNEQR	M	QLEALE
YAGR	·	INQLSEENGDLSEFK	<	LQQLGEAHQAETEVLR	N	QPEWLE
SEAAGR	+	IQAELAVILK	=	LQQLGEAHQAETEVLR	O	QQEQAI
IQEQASQGLR	·	KHPSSPECLVSAQK	>	LSQLEEHLSQLQDNPPQEK	P	QQLSSL
ESECEQLVK	-	KINQLSEENGDLSEFK	?	LTAQVASLTSELTTLNATIQQDQELAGLK	Q	QQNELA
IMR	·	KLDVEEPDSANSSFYSTR	@	LTAQVEQLEVFQR	R	QQNQEI
	/	KNSLISSLEEEVSILNR	A	LVMAESEK	S	RSQAGV
	0	KQQNQELQEQLR	B	MTMLLLYHSTMSSK	T	SAPASQ
	1	KVEELQACVETAR	C	NSLISSLEEEVSILNR	U	SLEAQV
STQALVSELLPAK	2	LADDLSTLQEK	D	PSLSLGTITDEEMK	V	SLVEQH
	3	LALLNEK	E	Q14980	W	SNRDEL
EDLENFLQK	4	LDFVCSFLQK	F	QAQLAQTLQQEQASQGLR	X	SNRDEL
LK	5	LEILQQQLQVANEAR	G	QDHAQQLATAAEER	Y	SQAPLE
YAEKR	6	LGHELQQAGLK	H	QEAATLAANNTQLQAR	Z	SQAPLE

P20962

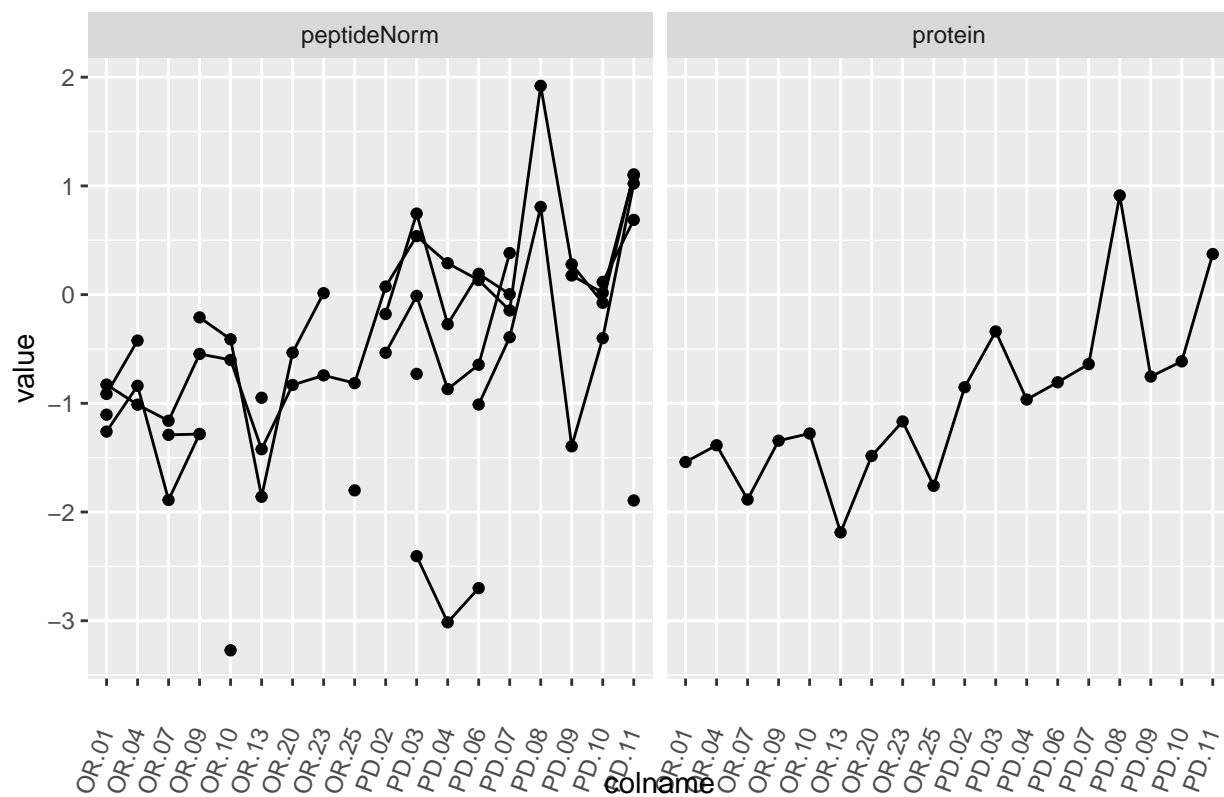


P20962

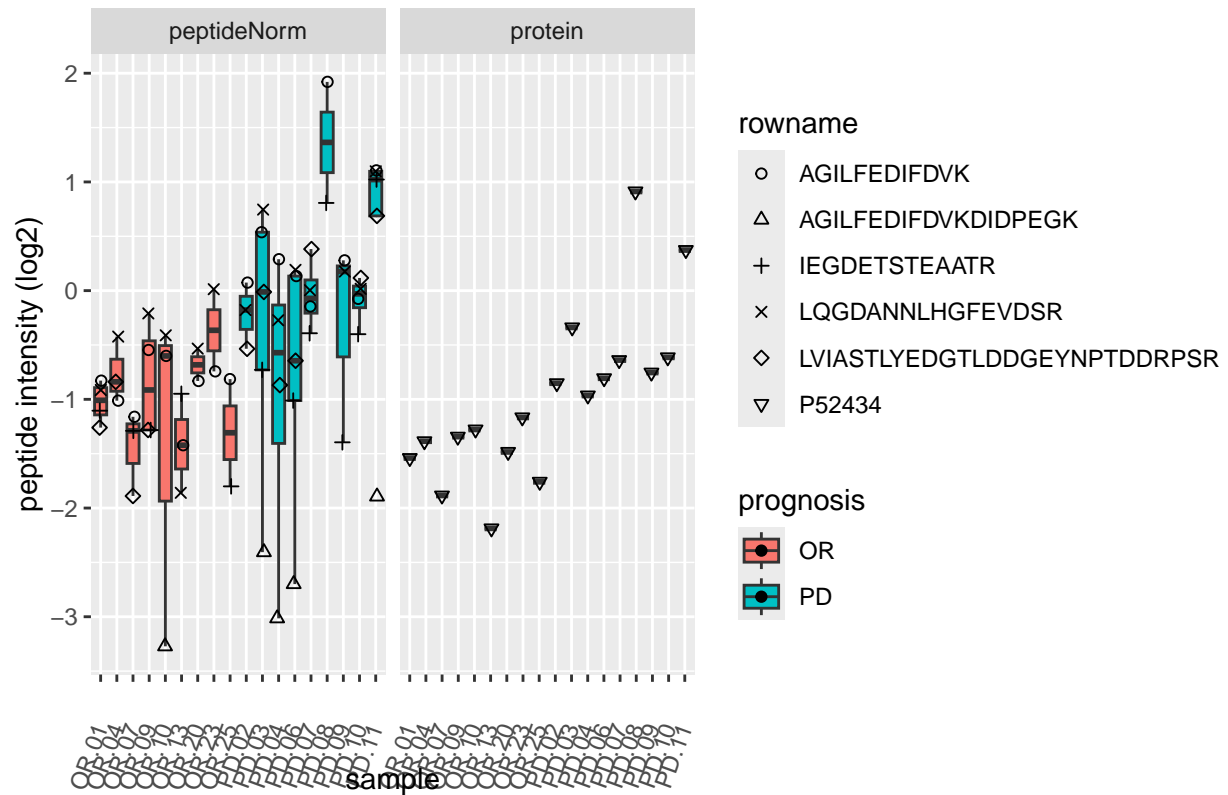




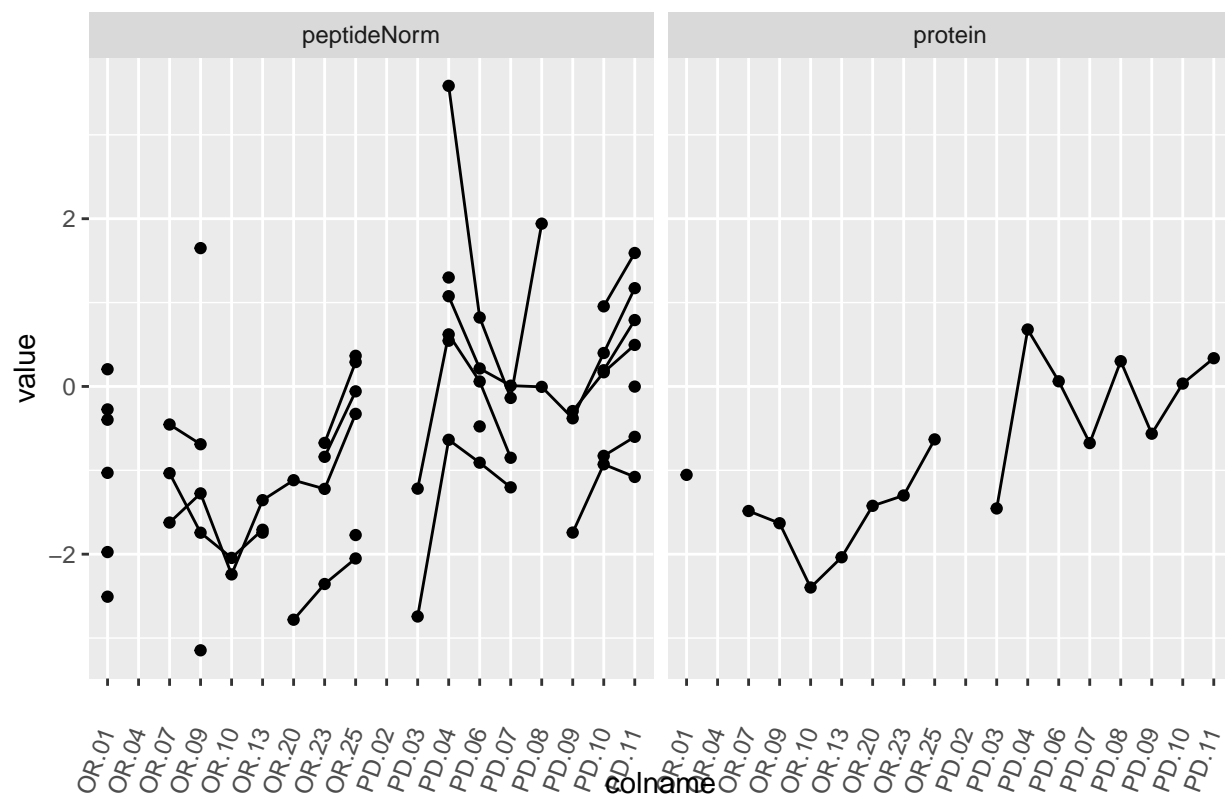
P52434



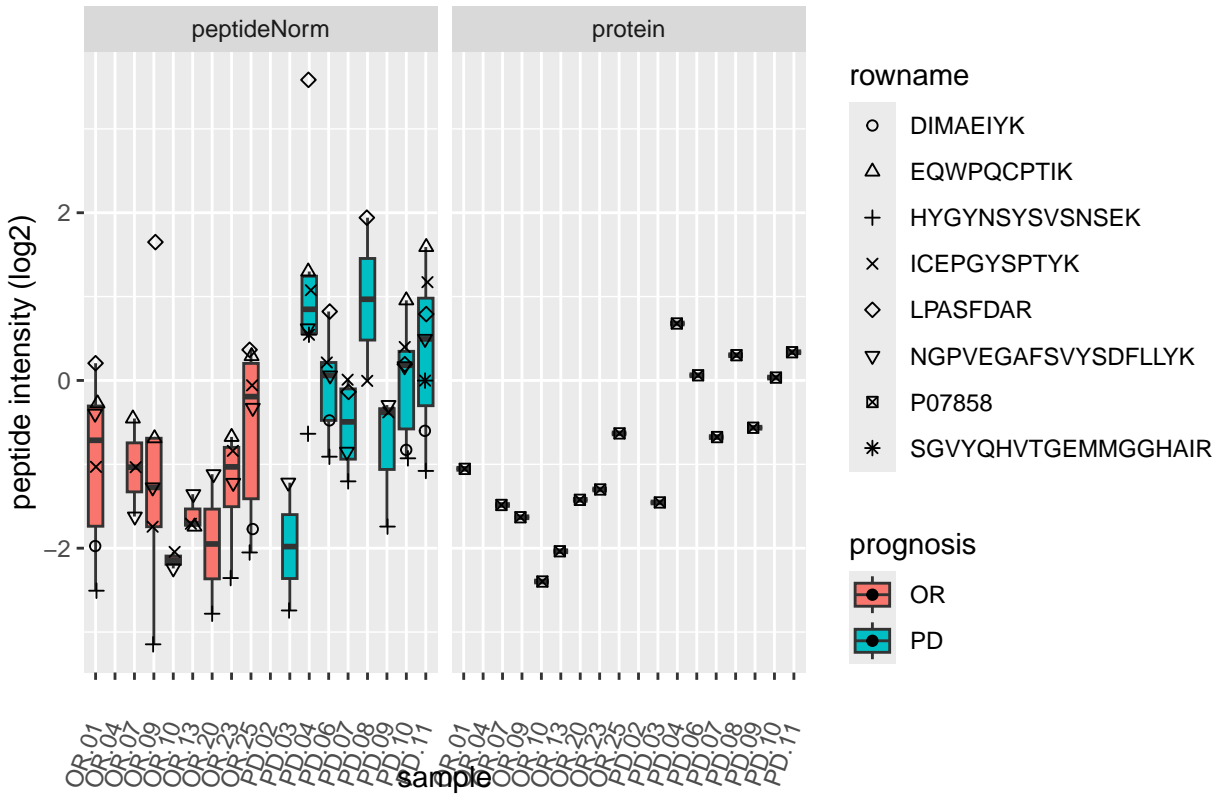
P52434



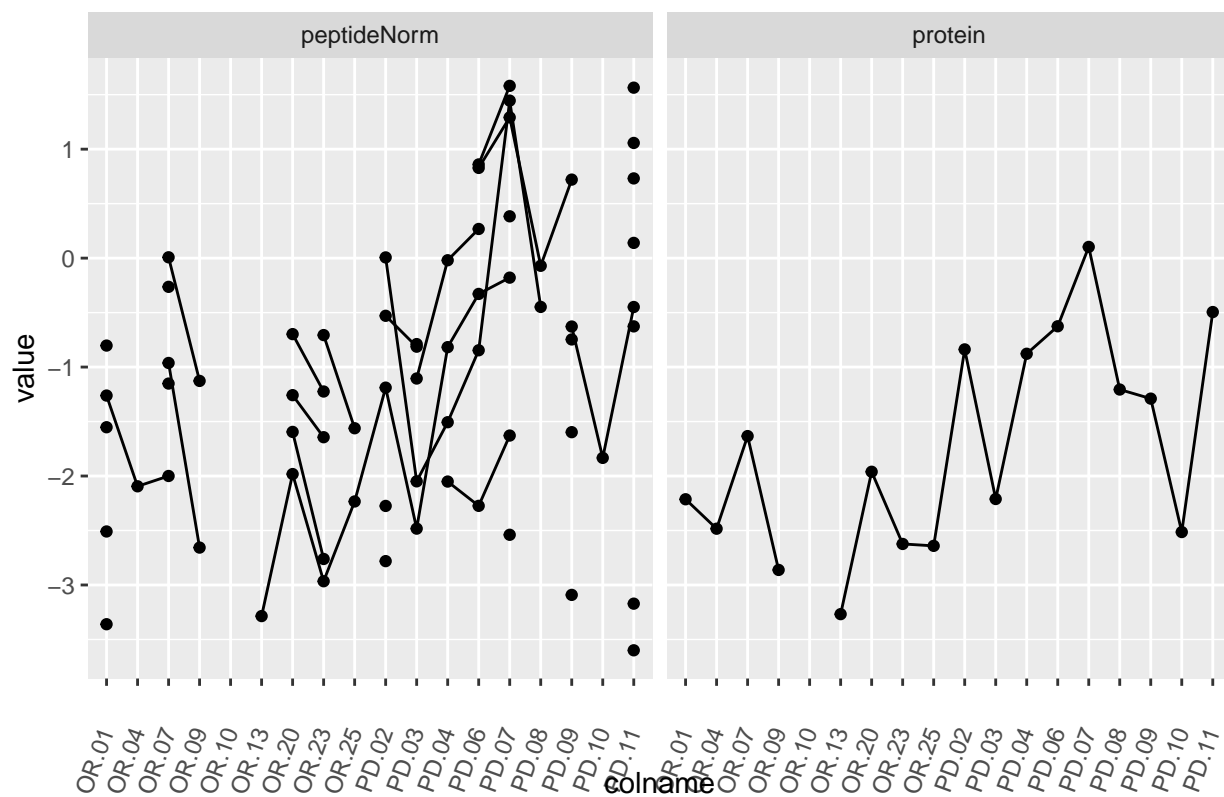
P07858



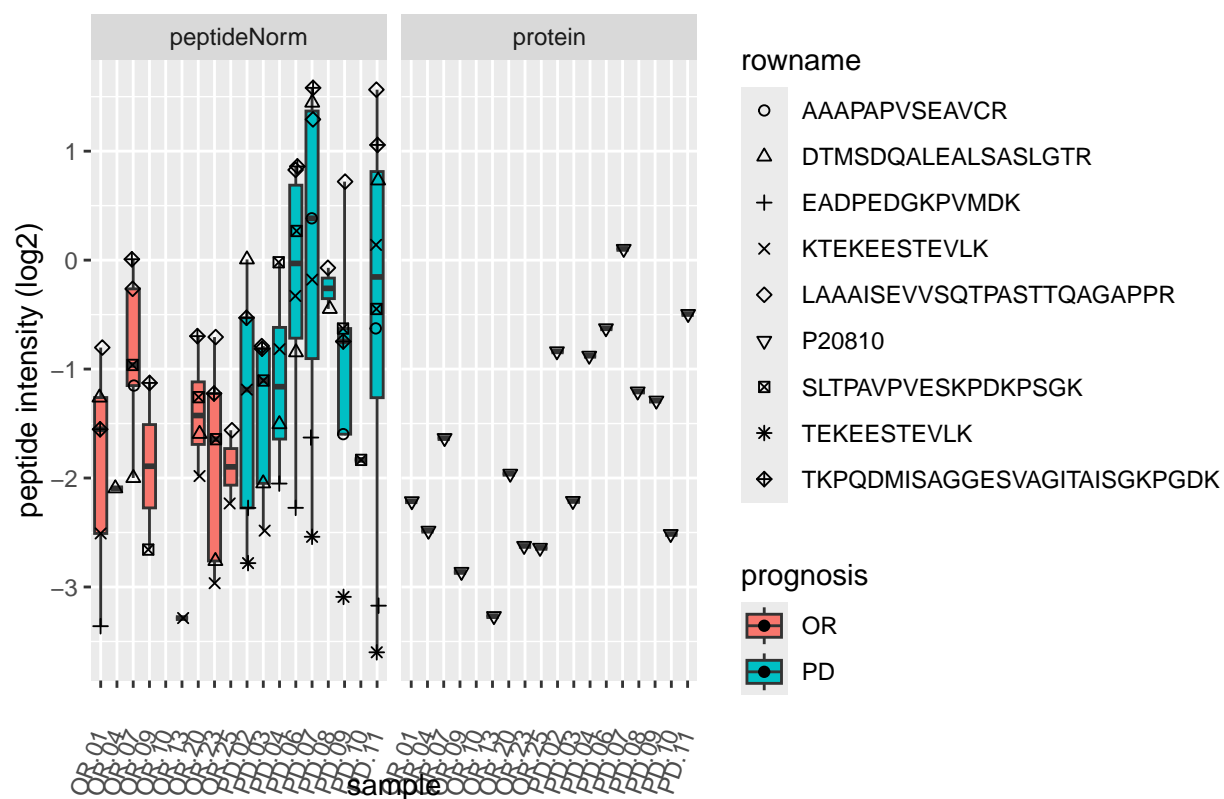
P07858



P20810



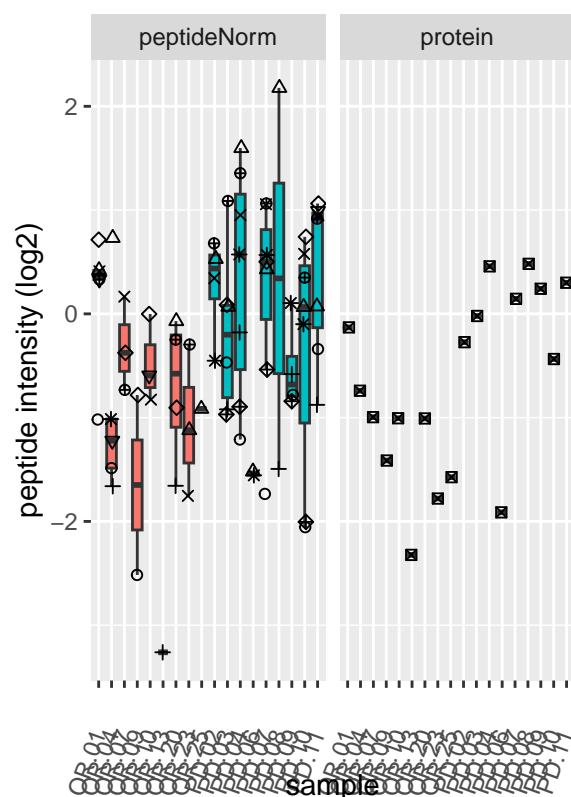
P20810



P22061



P22061



rowname

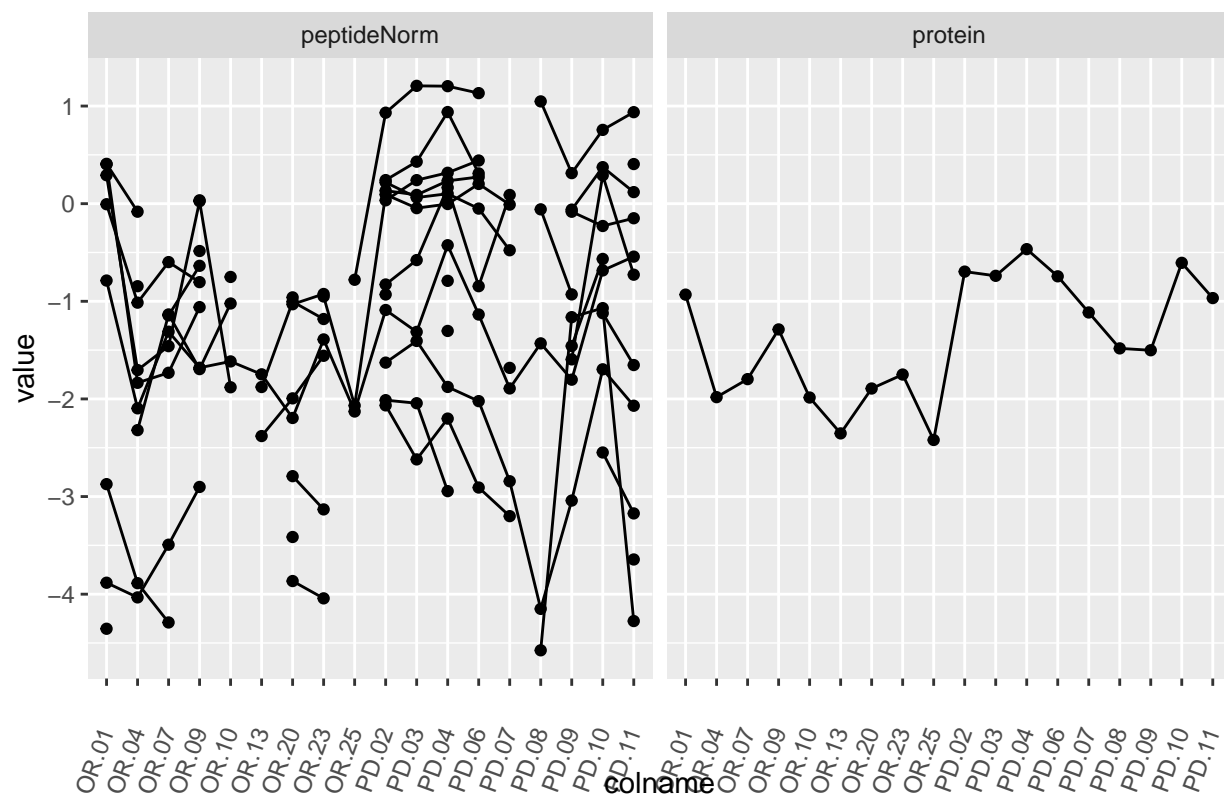
- ALDVGSGSGILTACFAR
- △ ELVDDSVNNVR
- + KDDPTLLSSGR
- × LILPVGPAAGNQMLEQYDK
- ◇ LILPVGPAAGNQMLEQYDKLQDGSIK
- ▽ MGYAEEAPYDAIHVGAAAPVVPQALIDQLKPGGR
- ⊠ P22061
- \* SGGASHSELIHNLR
- ⬠ VFEVMLATDR
- ⊕ VQLVVG DGR

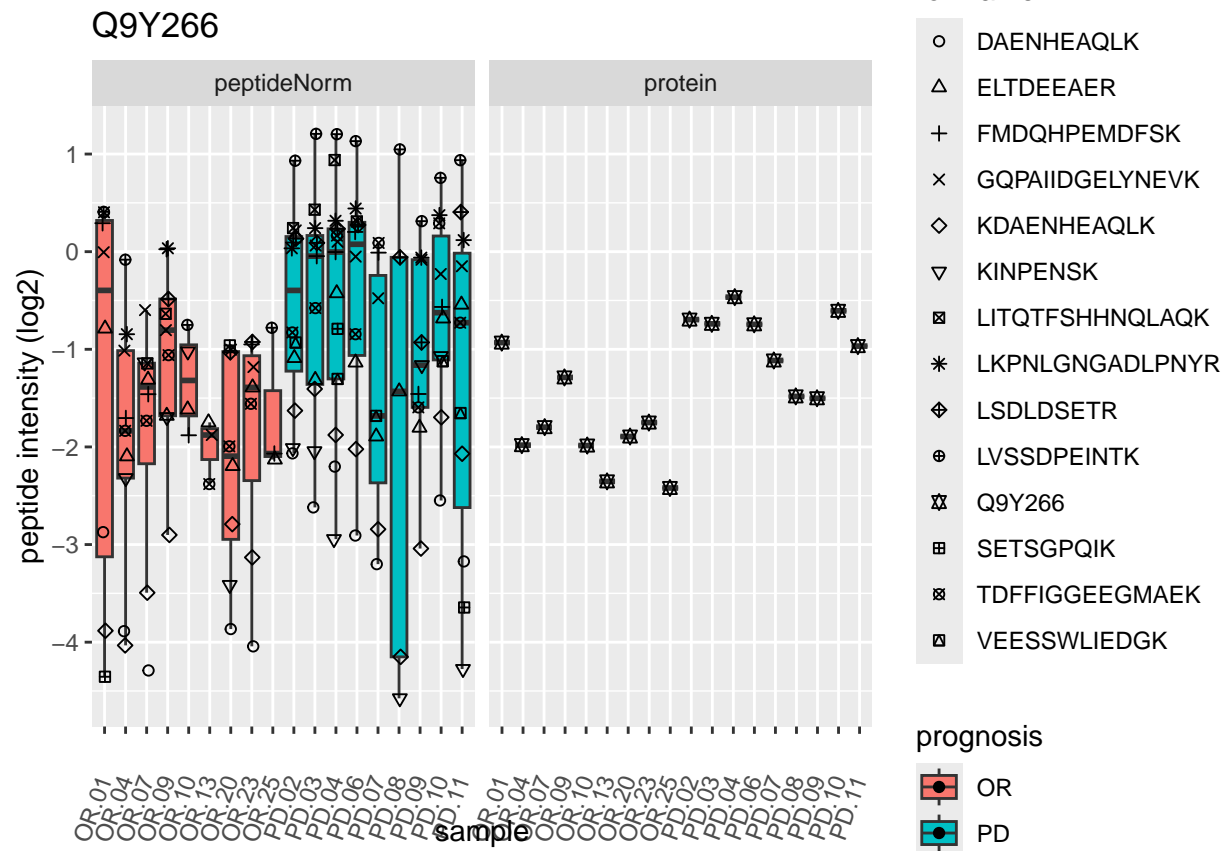
prognosis

- OR
- PD

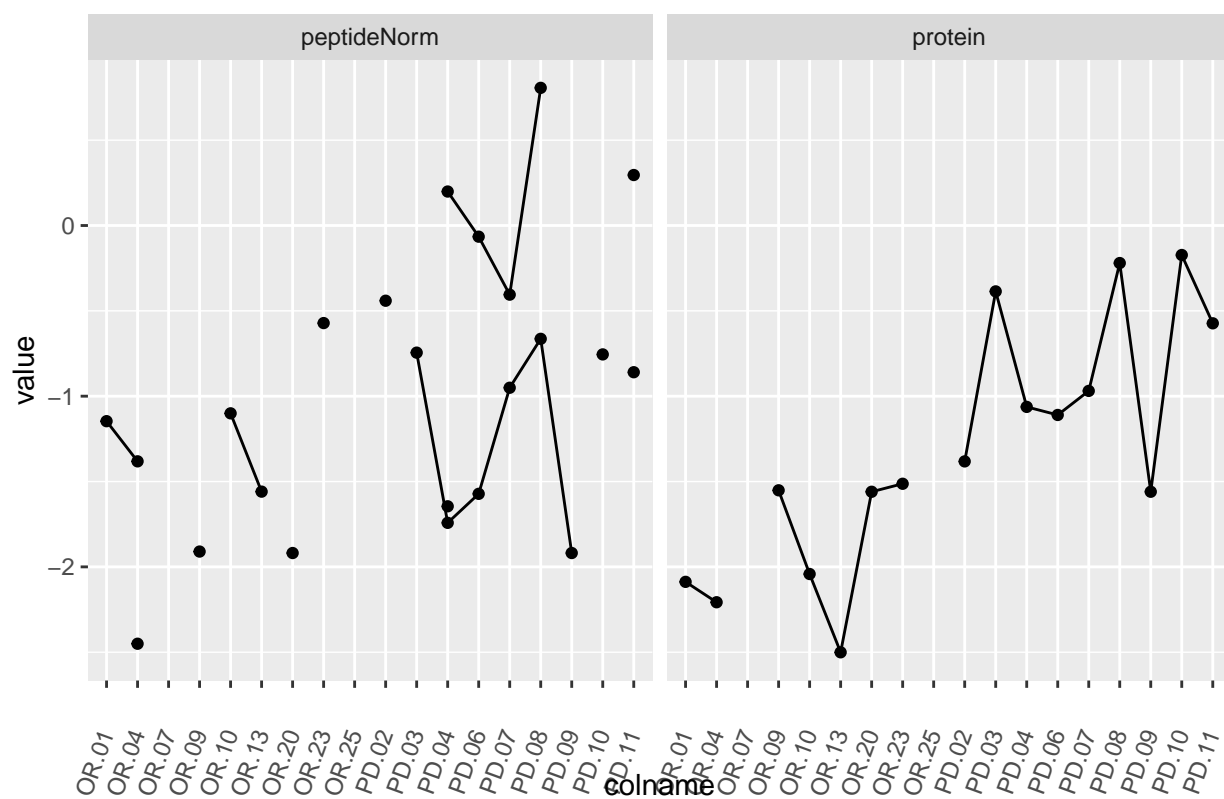


Q9Y266

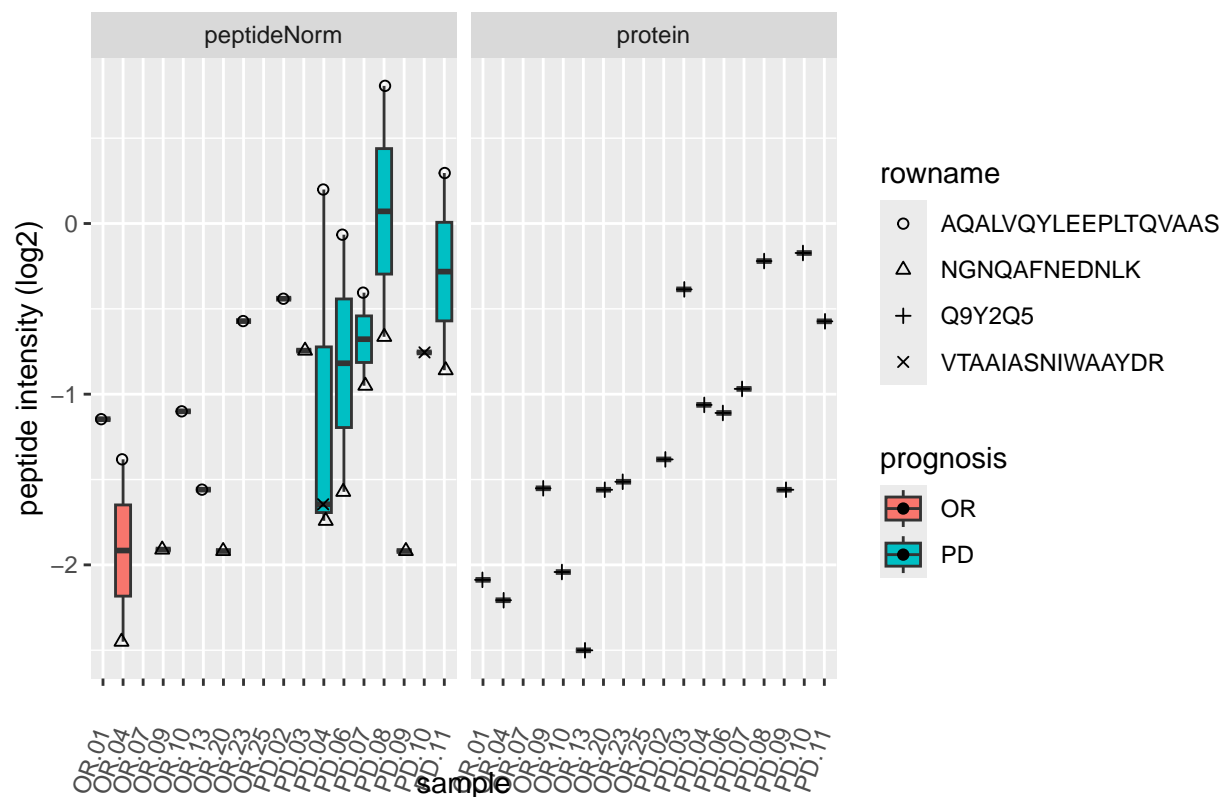




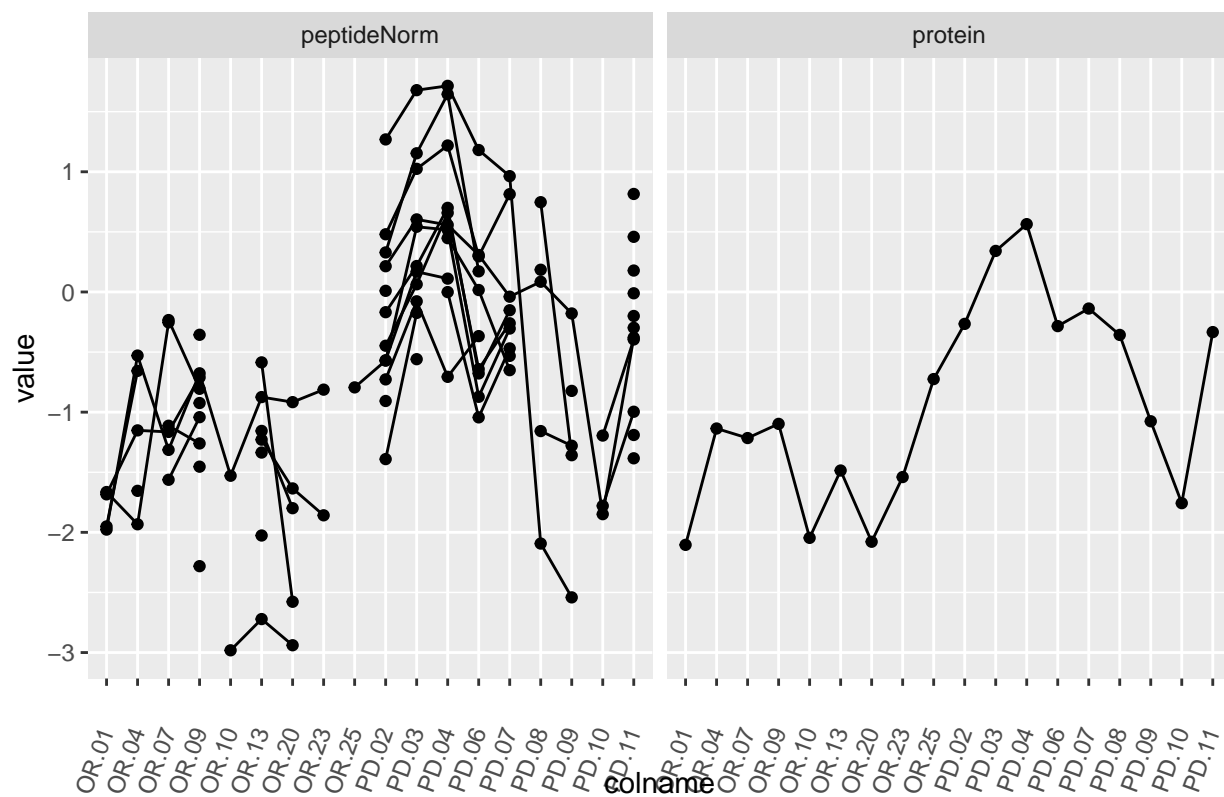
Q9Y2Q5

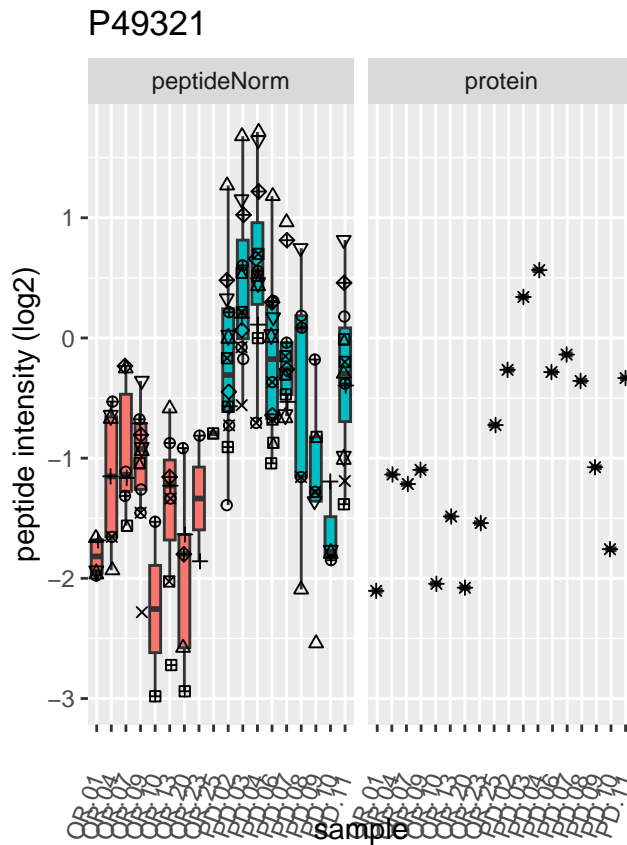


Q9Y2Q5



P49321





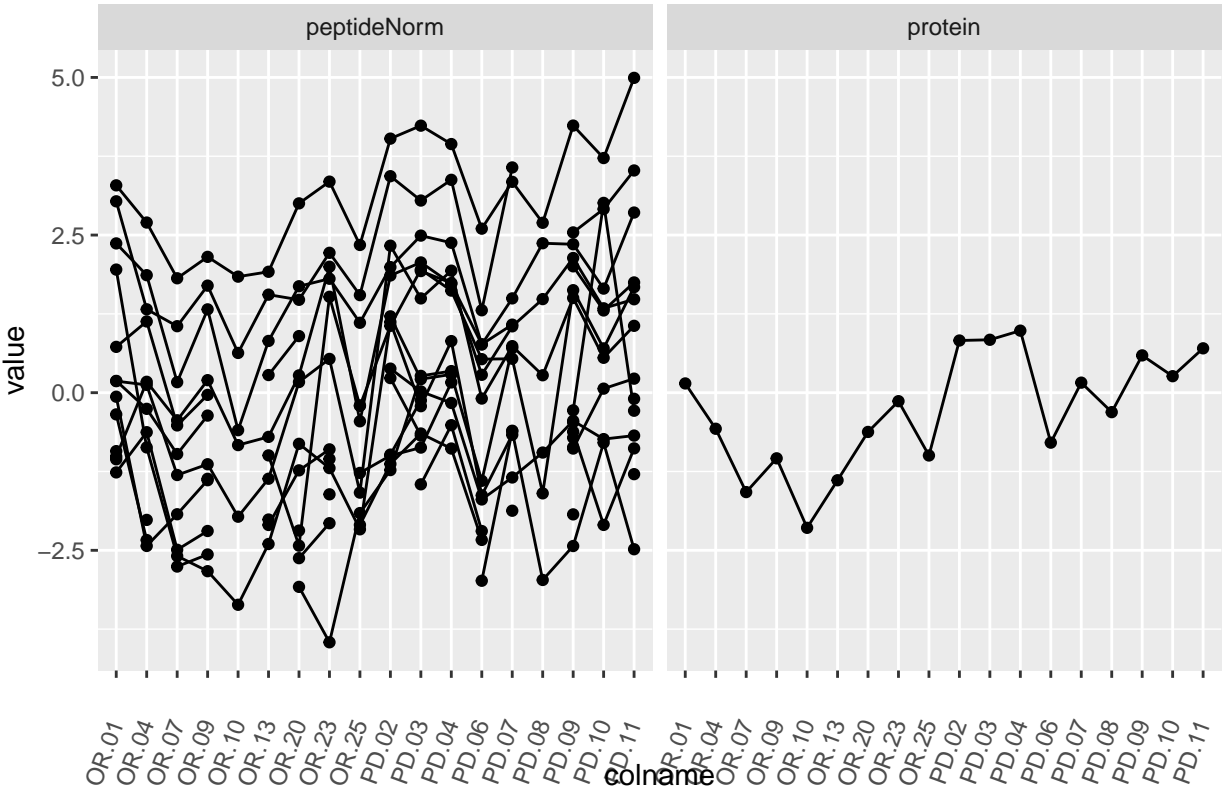
peptide

- DGA VNGPSVVG DQTPIEPQTSIER
- △ EAQLYAAQAHLK
- + EIEELKELLPEIR
- × EQVYDAMGEKEEAK
- ◇ EVSEEQPVVTLEK
- ▽ GGAAPEGPNEAEVTSGKPEQEVPDAAEEK
- ▣ LSVEESEAAGDGVDTK
- \* P49321
- ◊ QGTAVEVEAESLDPTVKPVDVGGDEPEEK
- ⊕ SGNVAELALK
- ⊗ SIEVIENR
- ⊞ SLAKPETDKEQDSEMEK
- ⊠ SLLELAR
- ⊡ VDLTLDWLTETSEEAK

prognosis

- OR
- PD

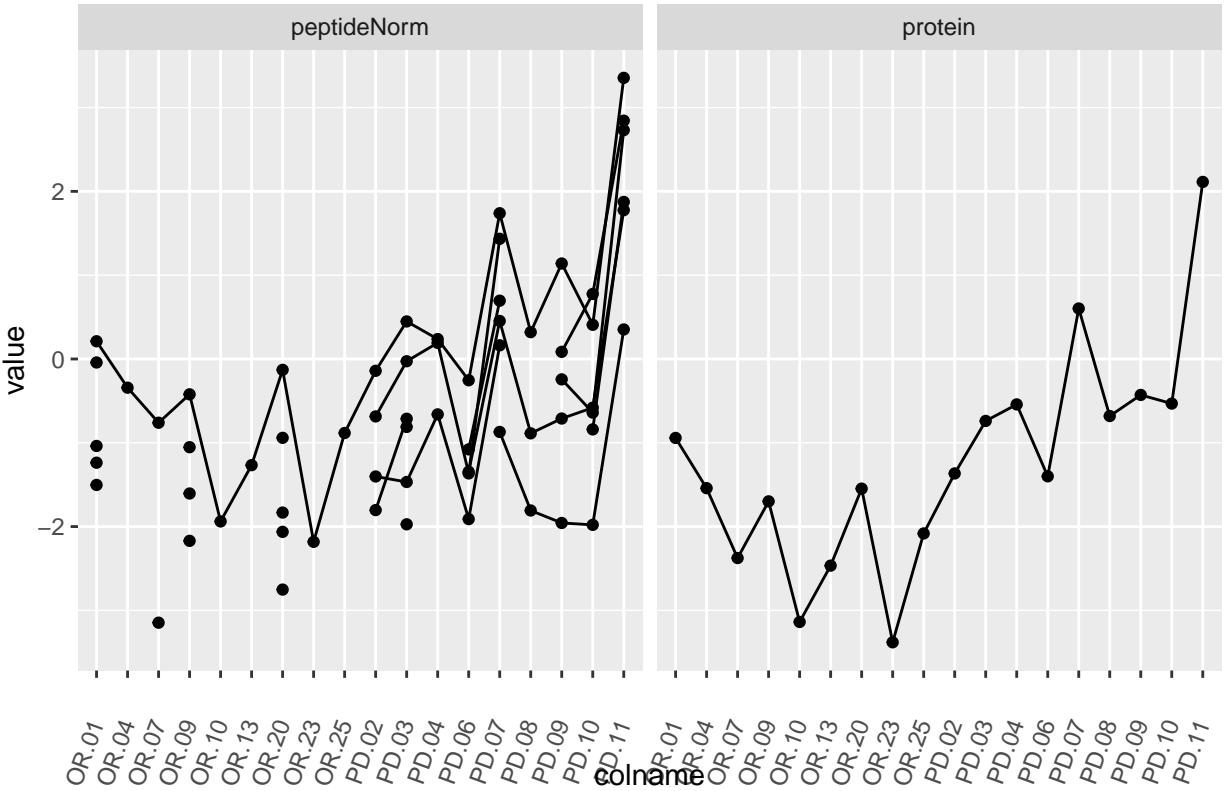
Q99497



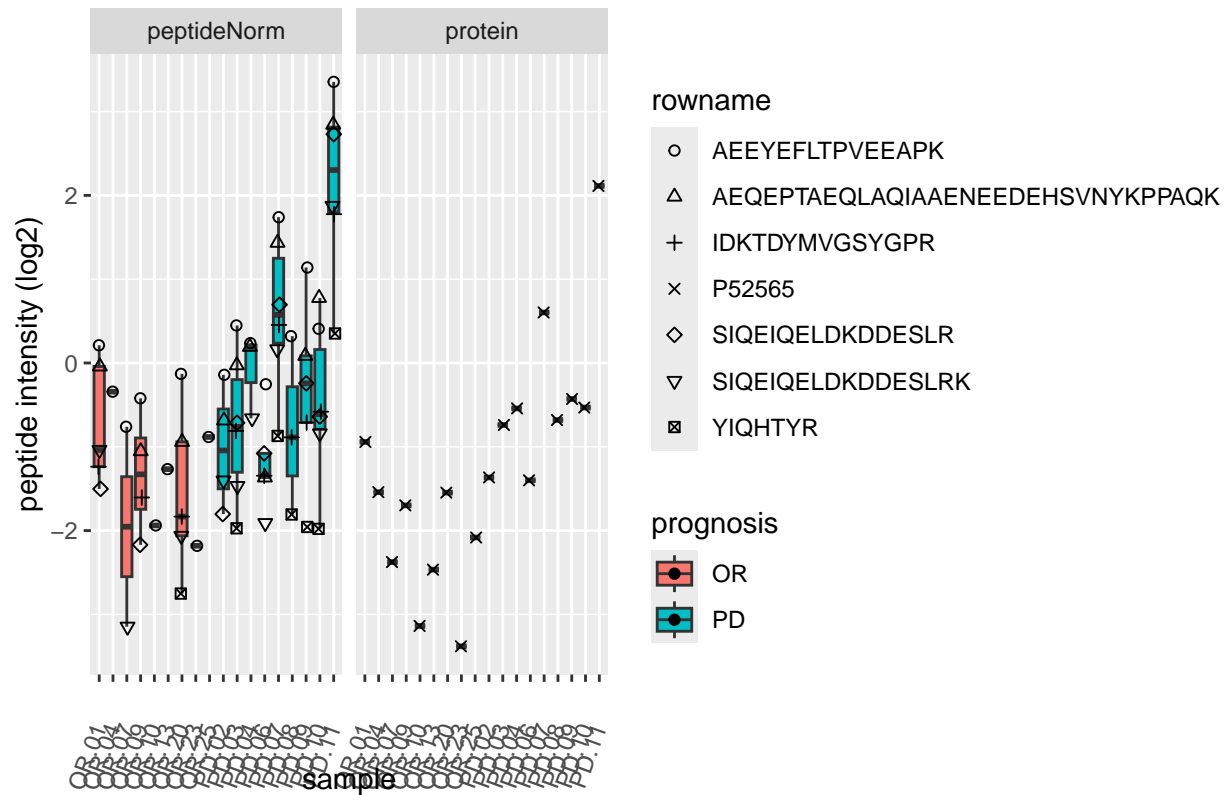




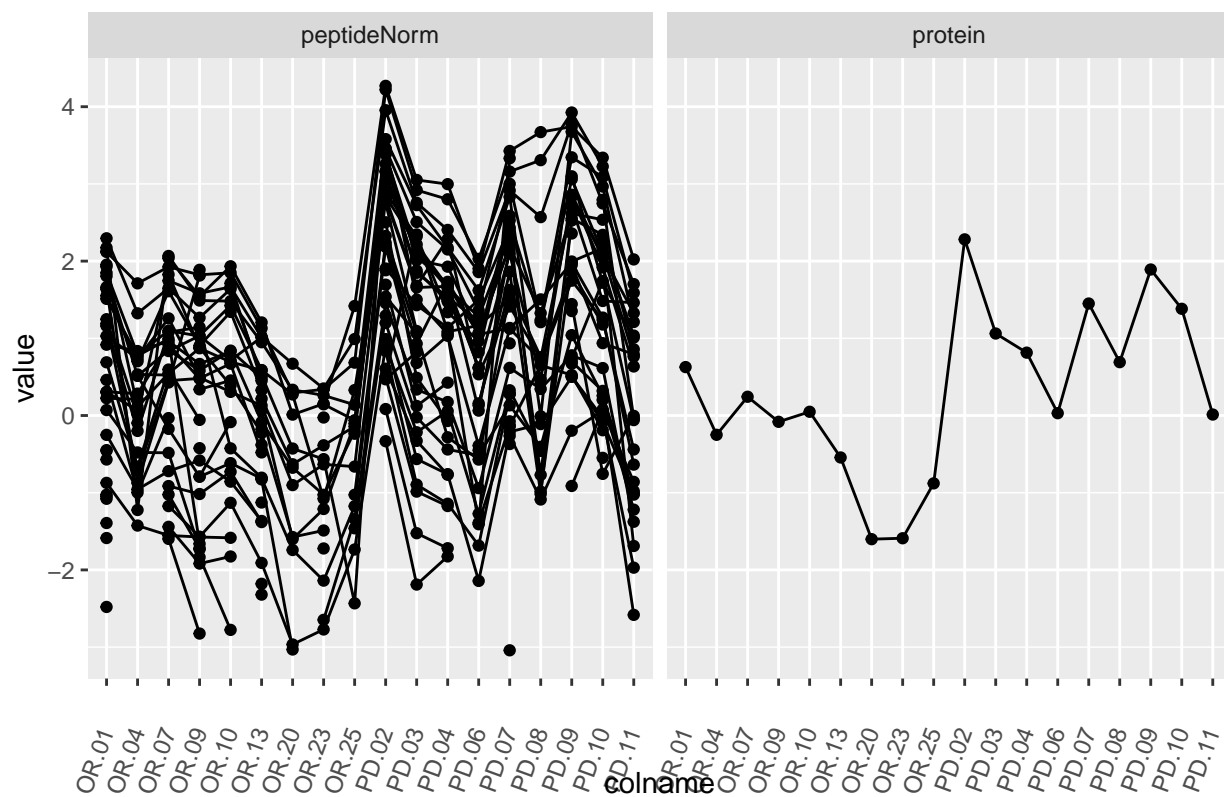
P52565

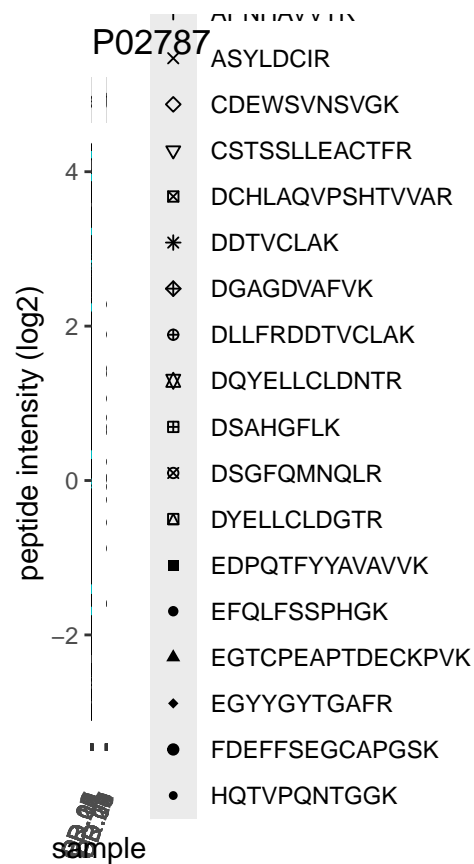


P52565



P02787



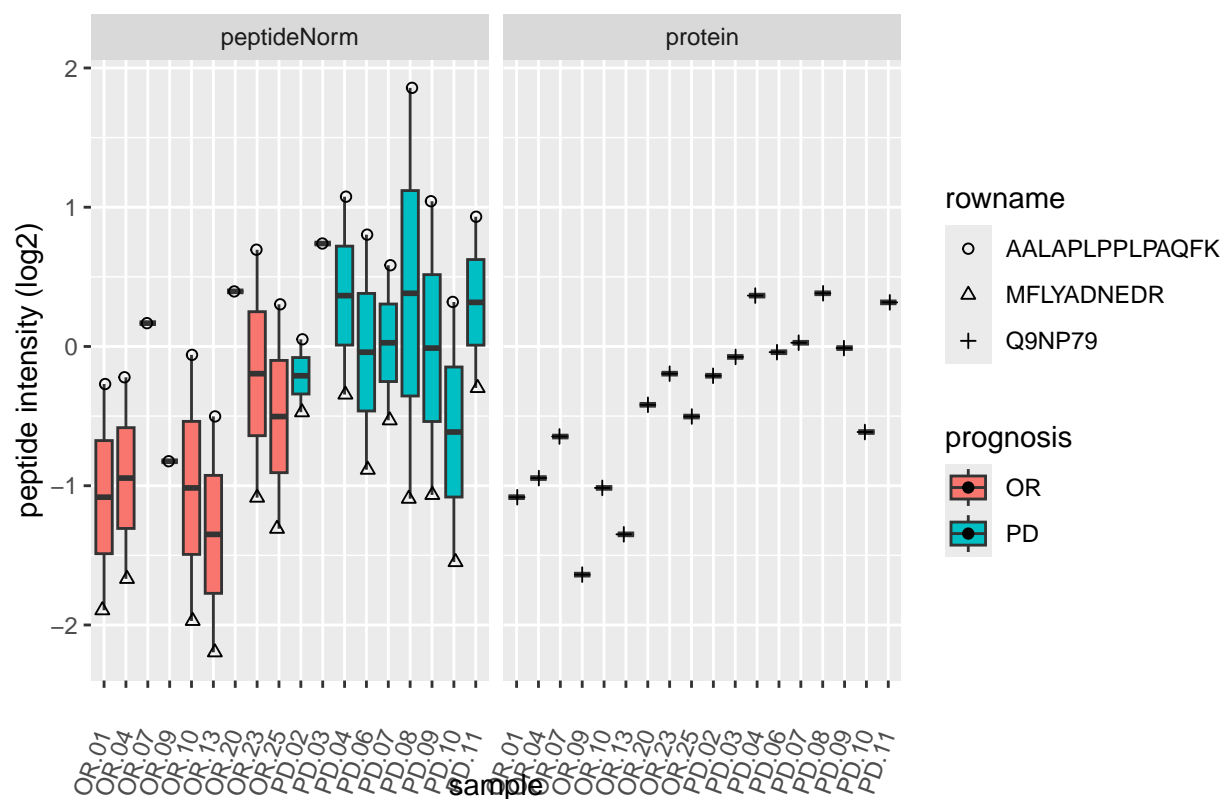


▽	IMINGLEADAMOLEDQSTVTAIR
△	KASYLDCIR
▽	KPVDEYKDCHLAQVPSHTVVAR
	KPVEEYANCHLAR
	LCMGSGNLNCEPNNK
	LKCDEWSVNSVGK
	MYLGYEYVTAIR
	NLNEKDYELLCLDGTR
	NPDPWAK
	P02787
!	SAGWNIPIGLLYCDLPEPR
"	SASDLTWDNLK
#	SKEFQLFSSPHGK
\$	SVIPSDGPSVACVK
%	TAGWNIPMGLLYNK
&	WCALSHHER
'	WCAVSEHEATK
(	YLGEYEVK

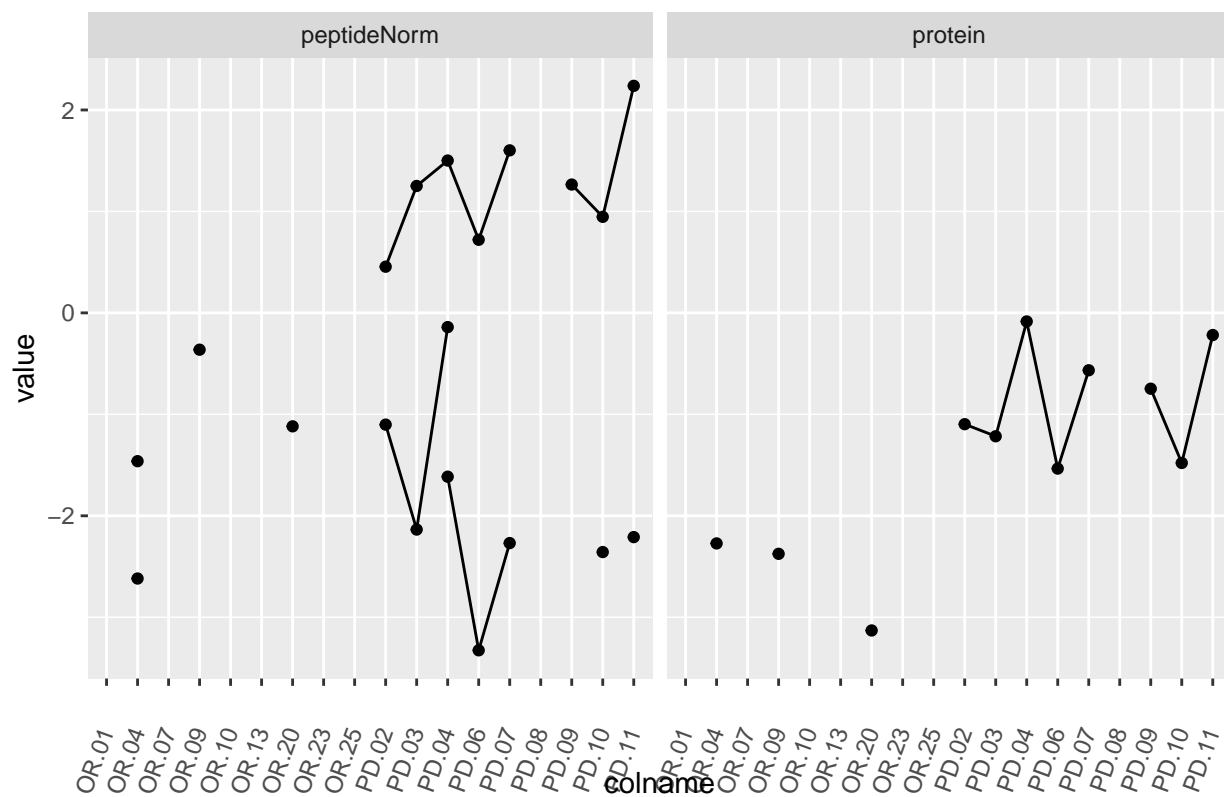
# Q9NP79



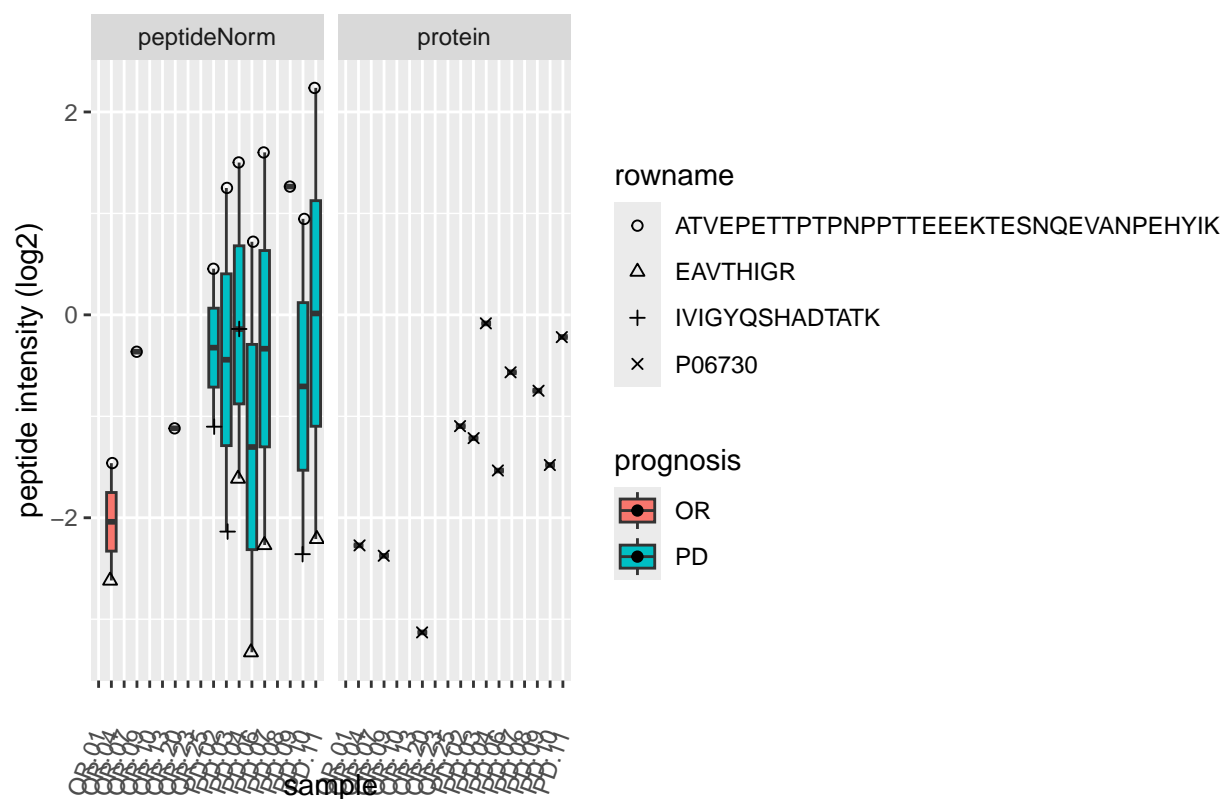
# Q9NP79



P06730

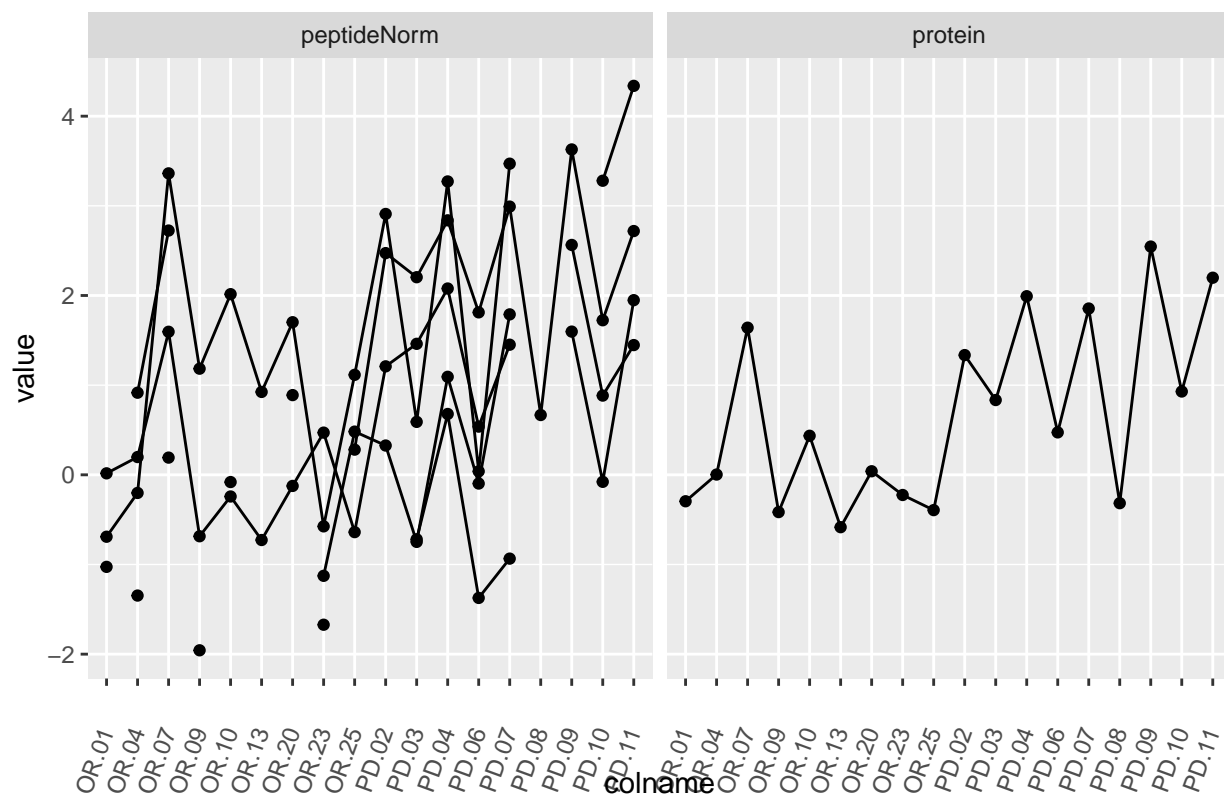


P06730

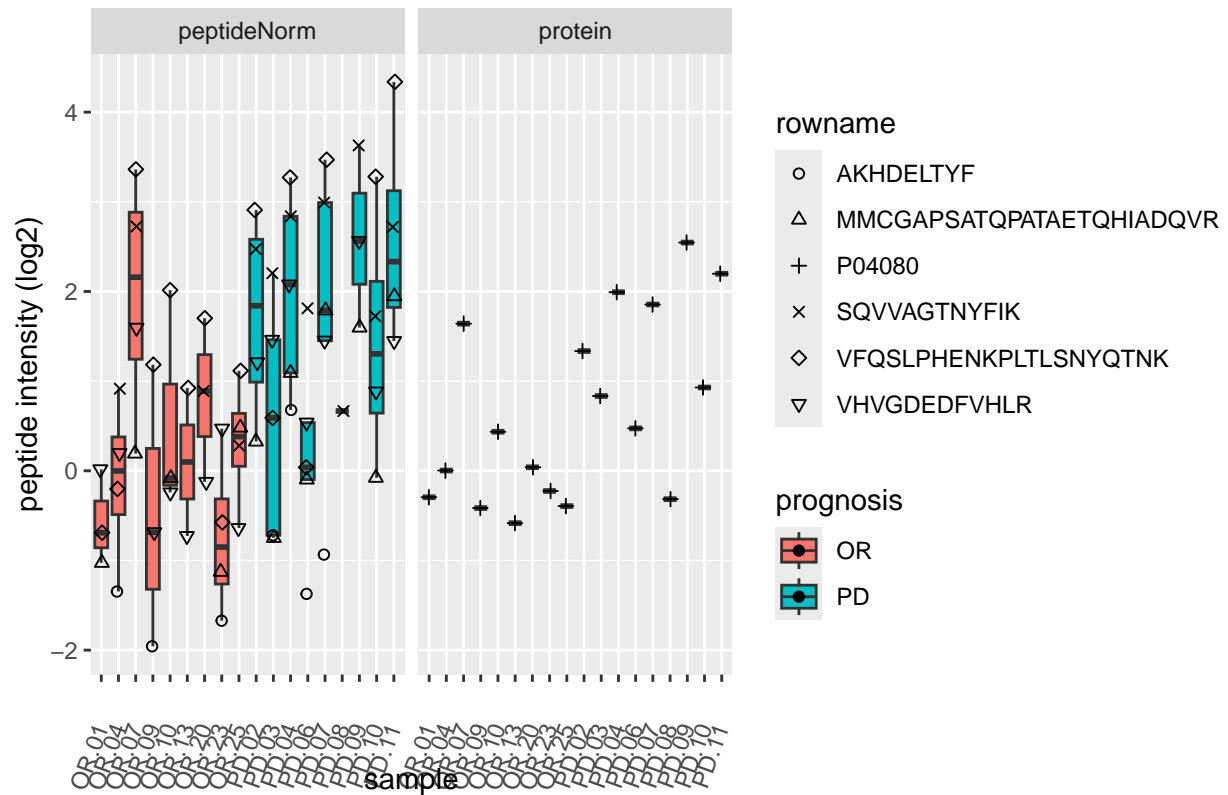




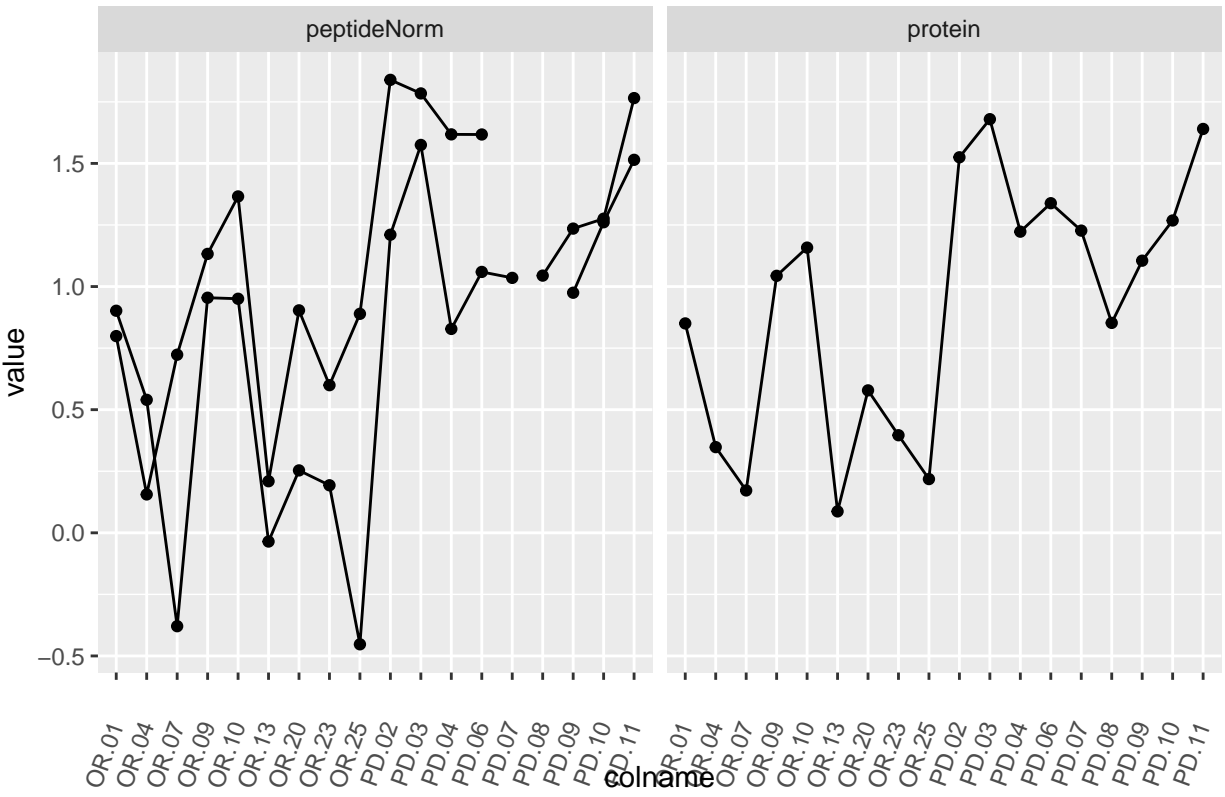
P04080



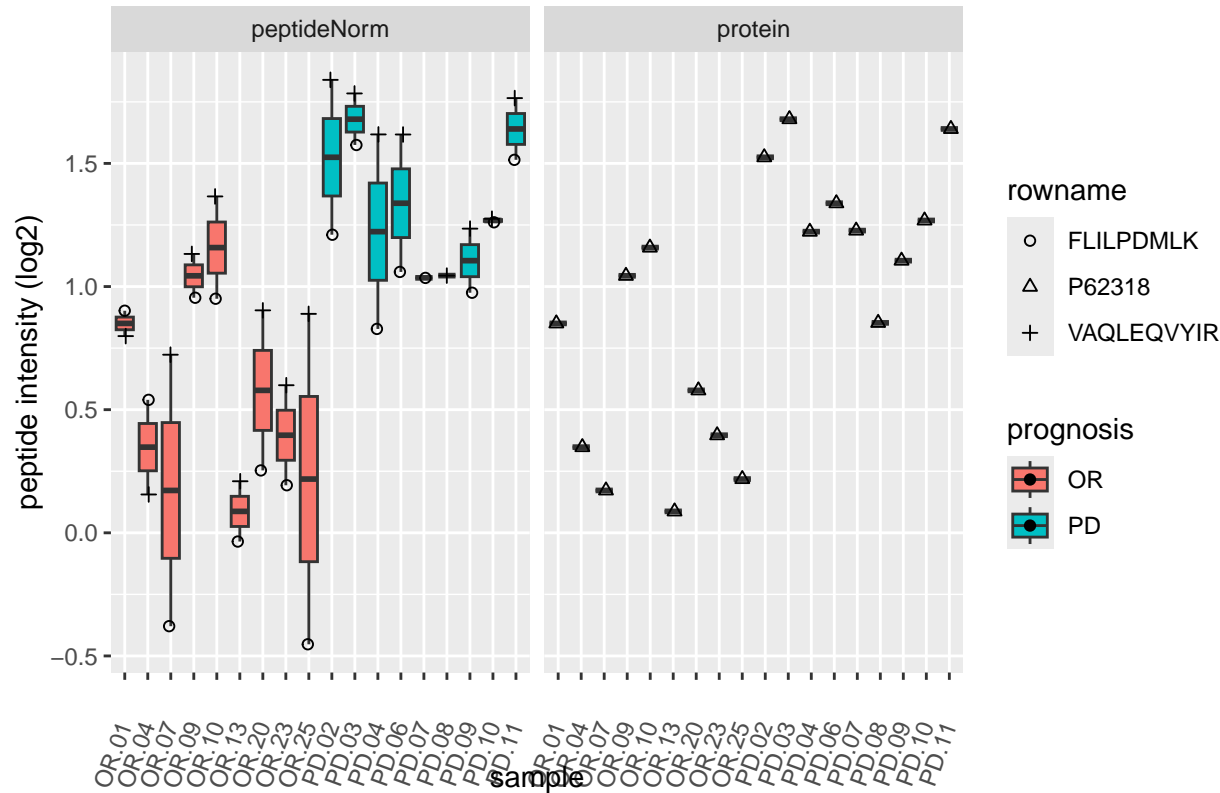
P04080



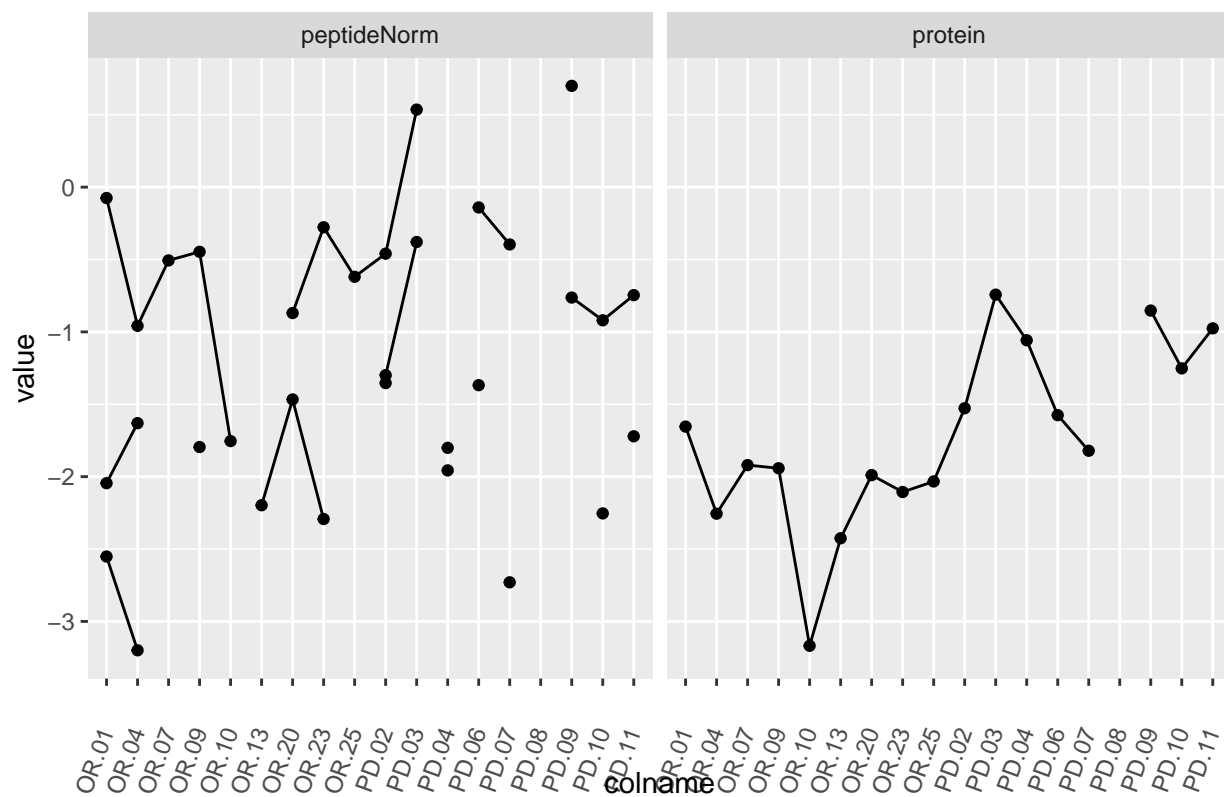
P62318

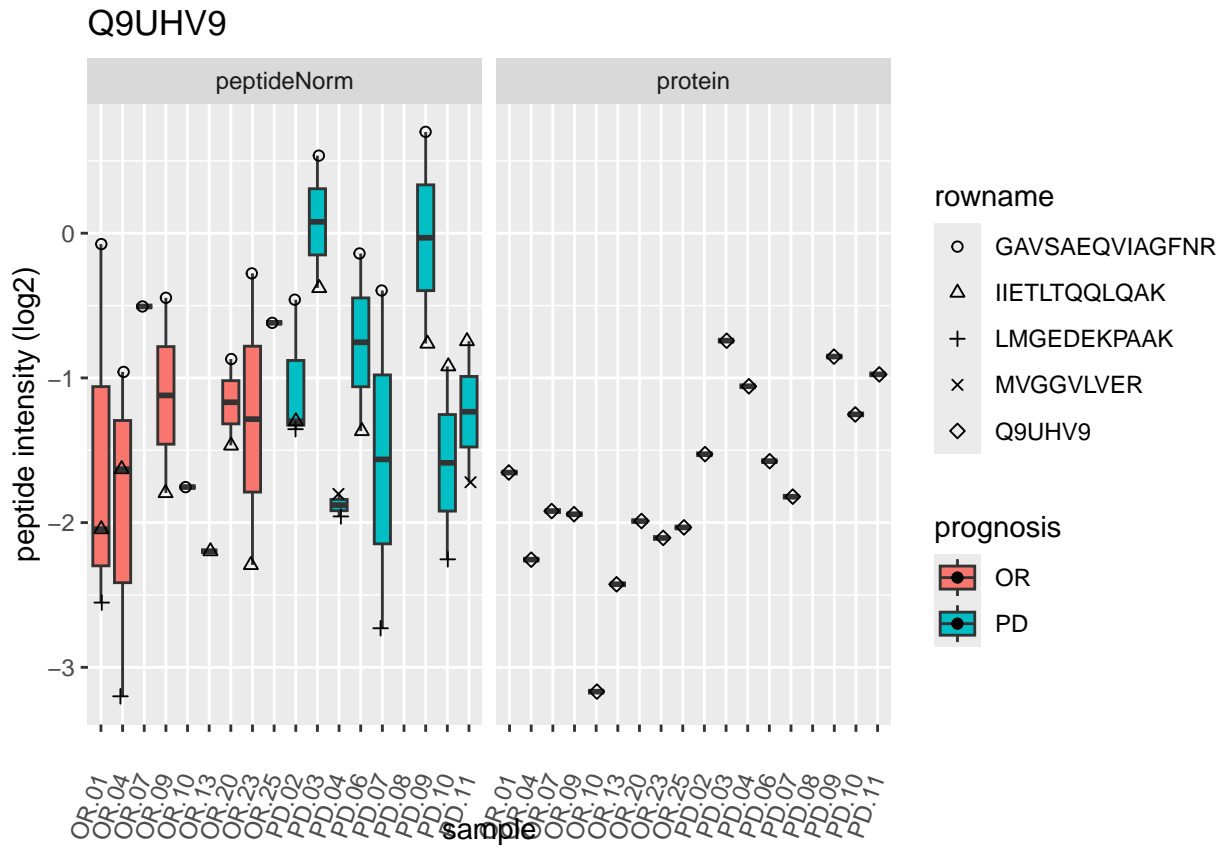


P62318

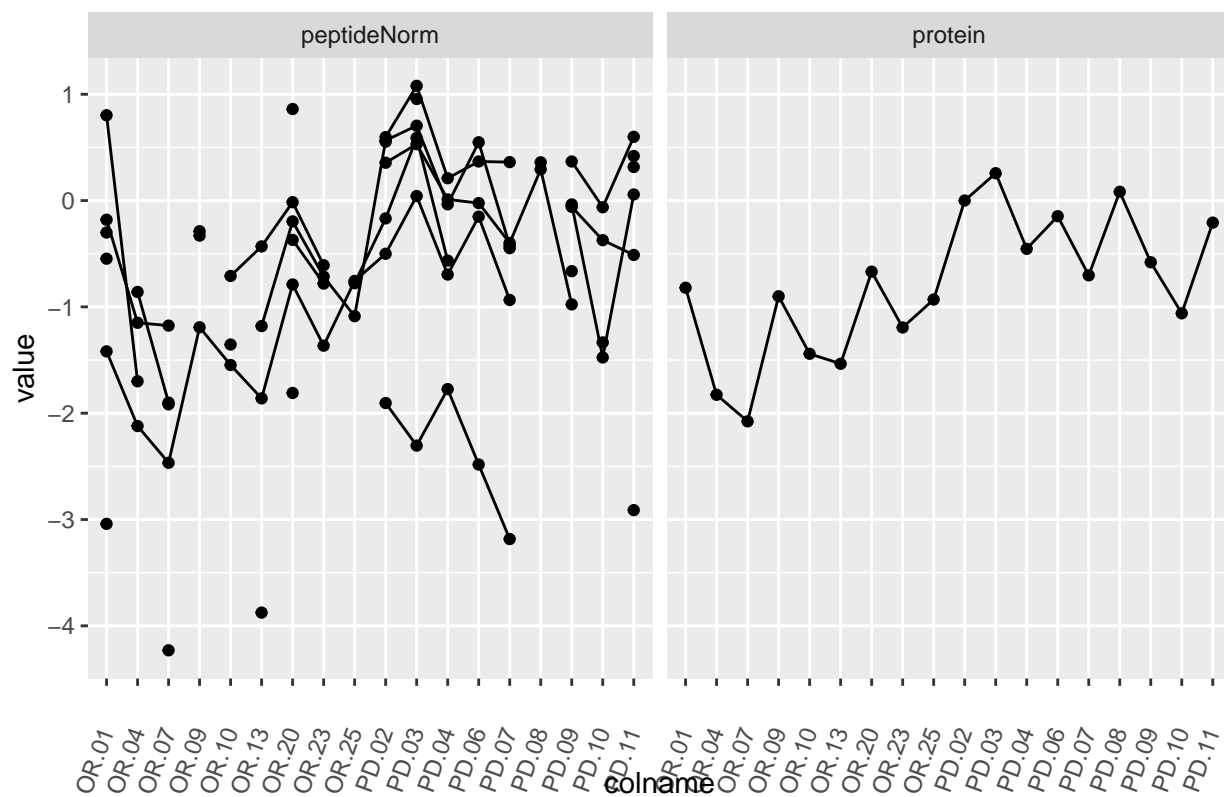


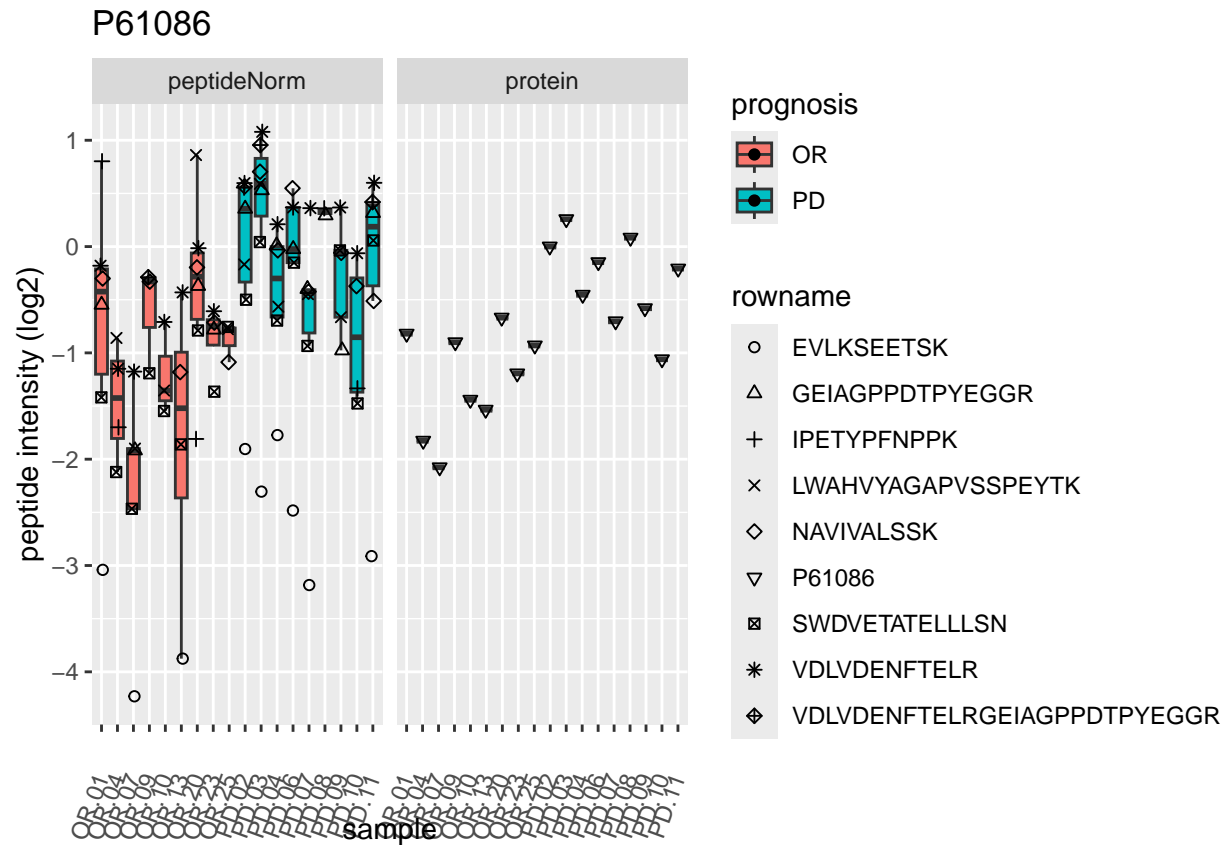
Q9UHV9





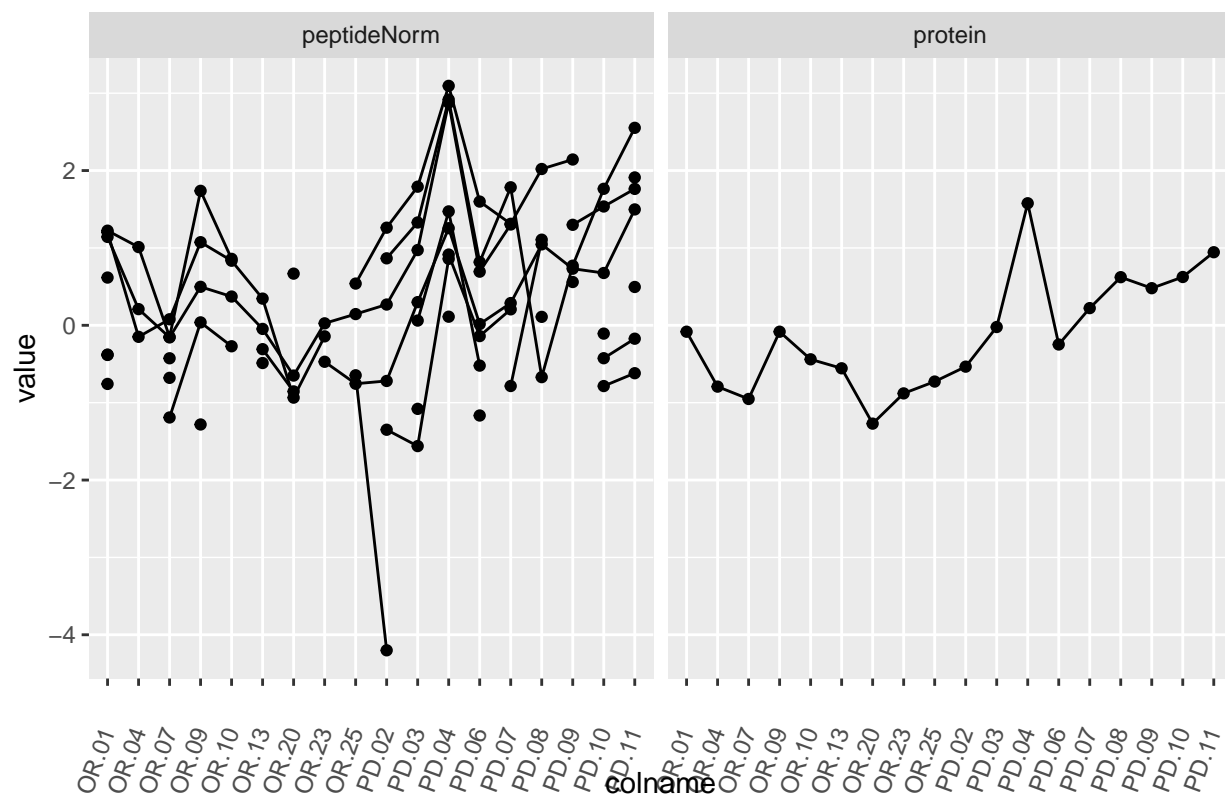
P61086

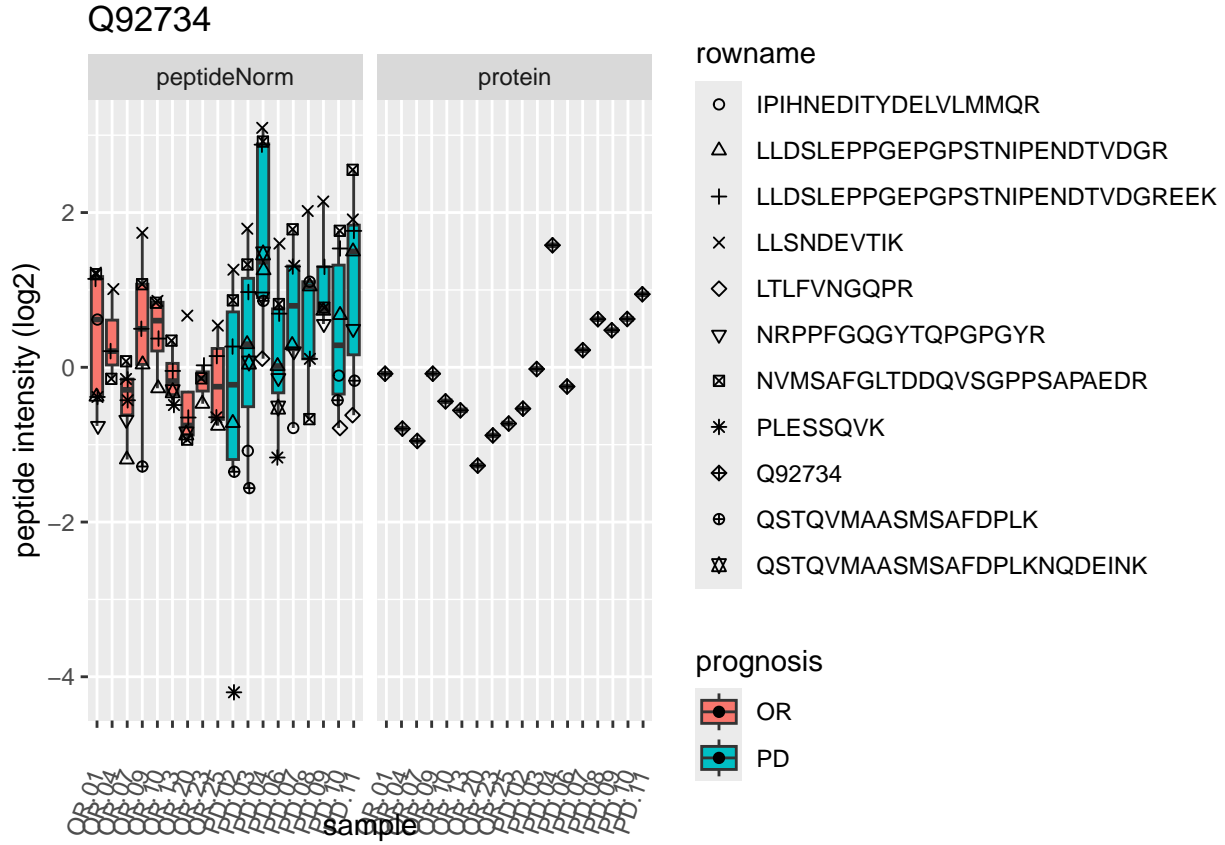




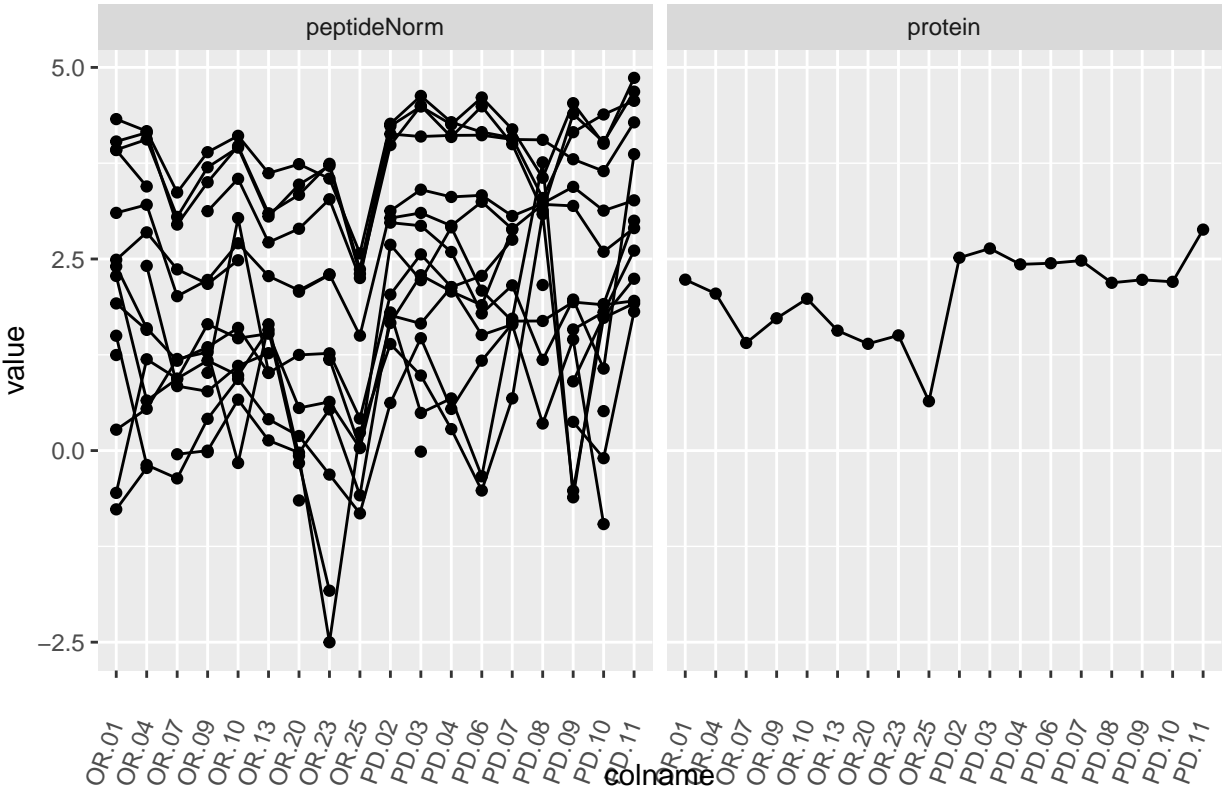


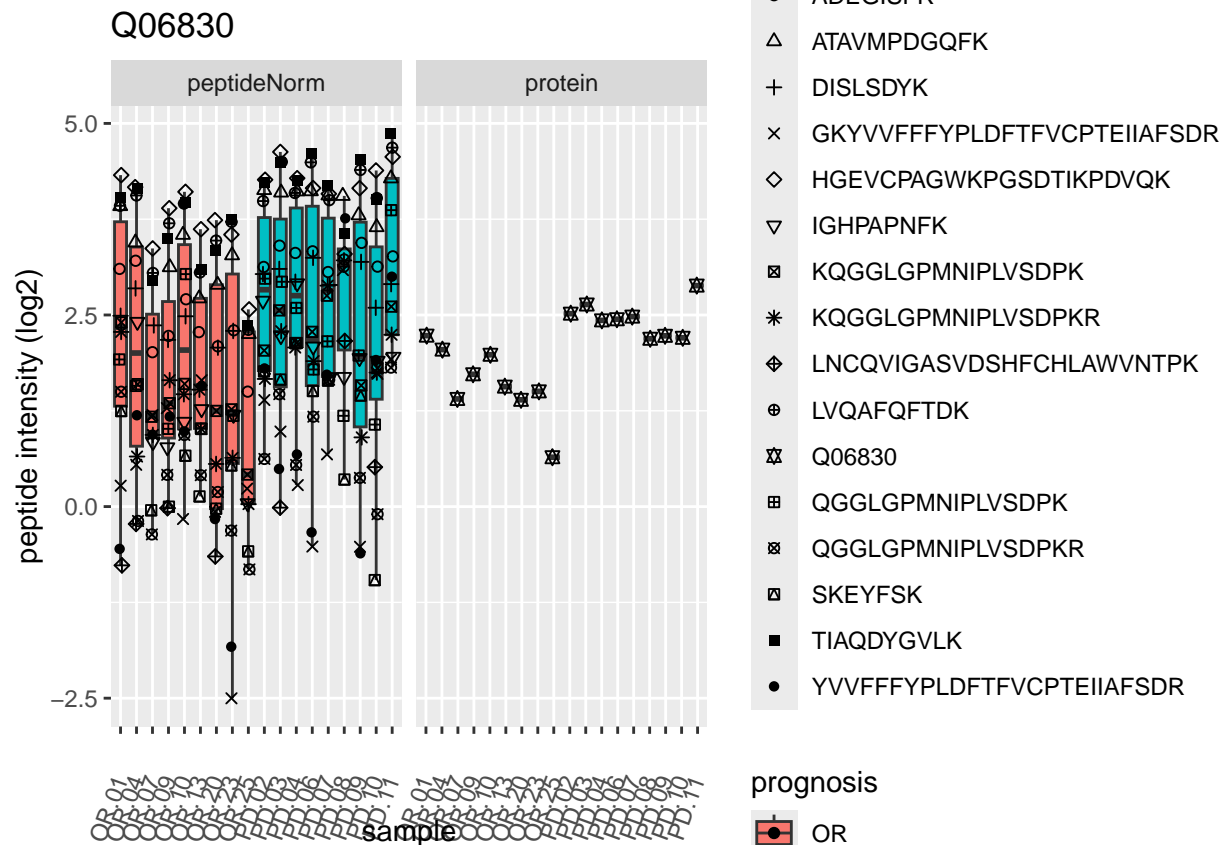
Q92734



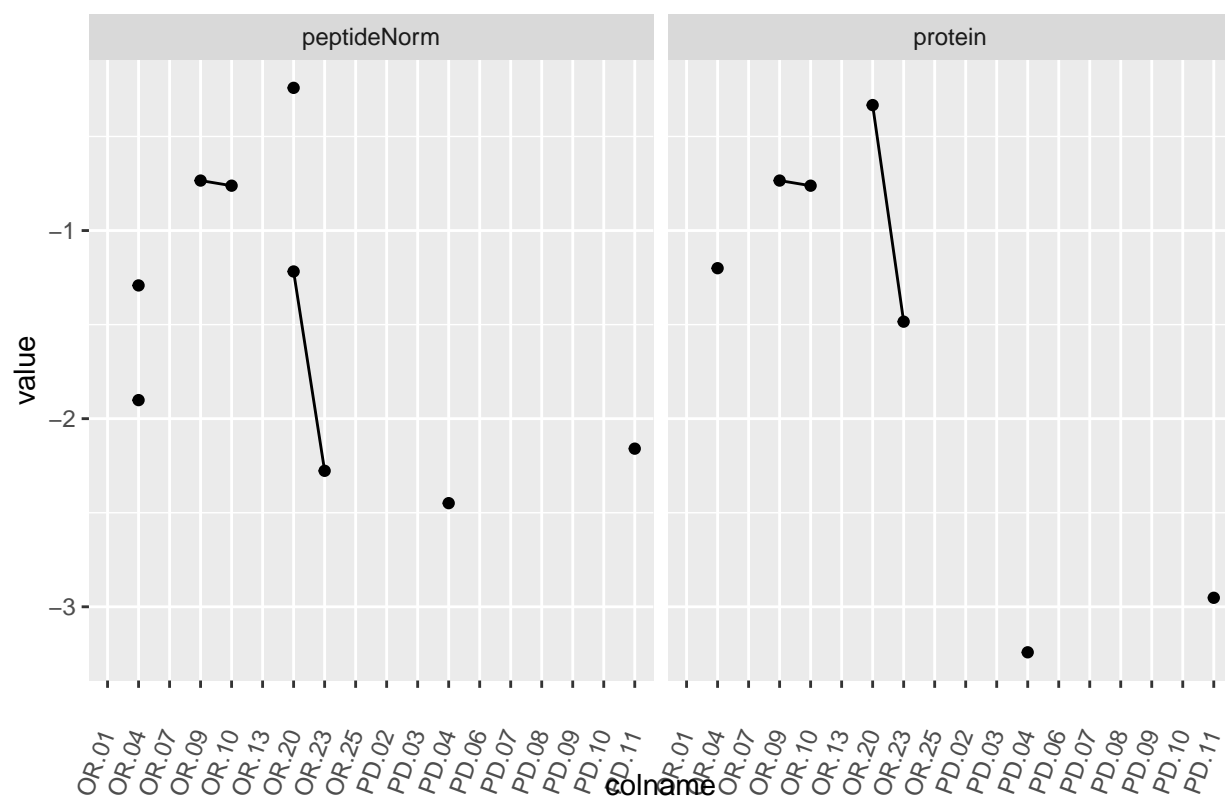


Q06830

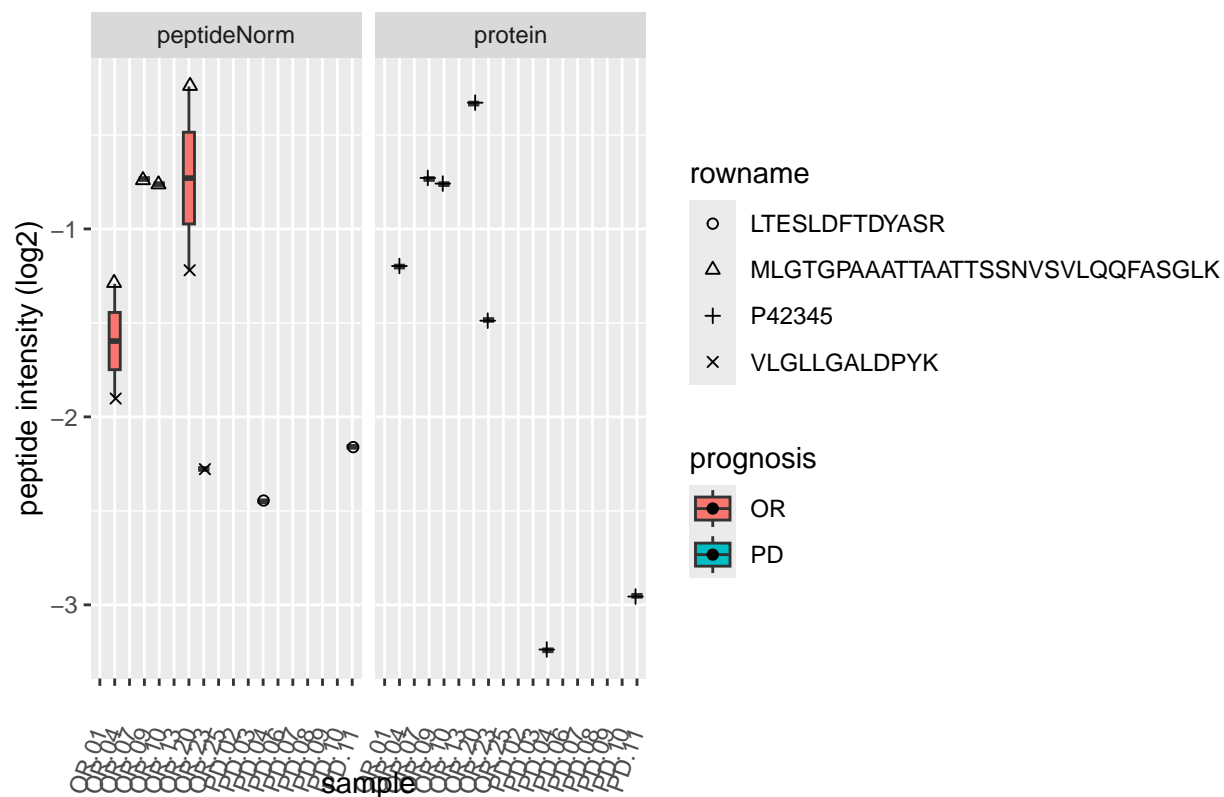




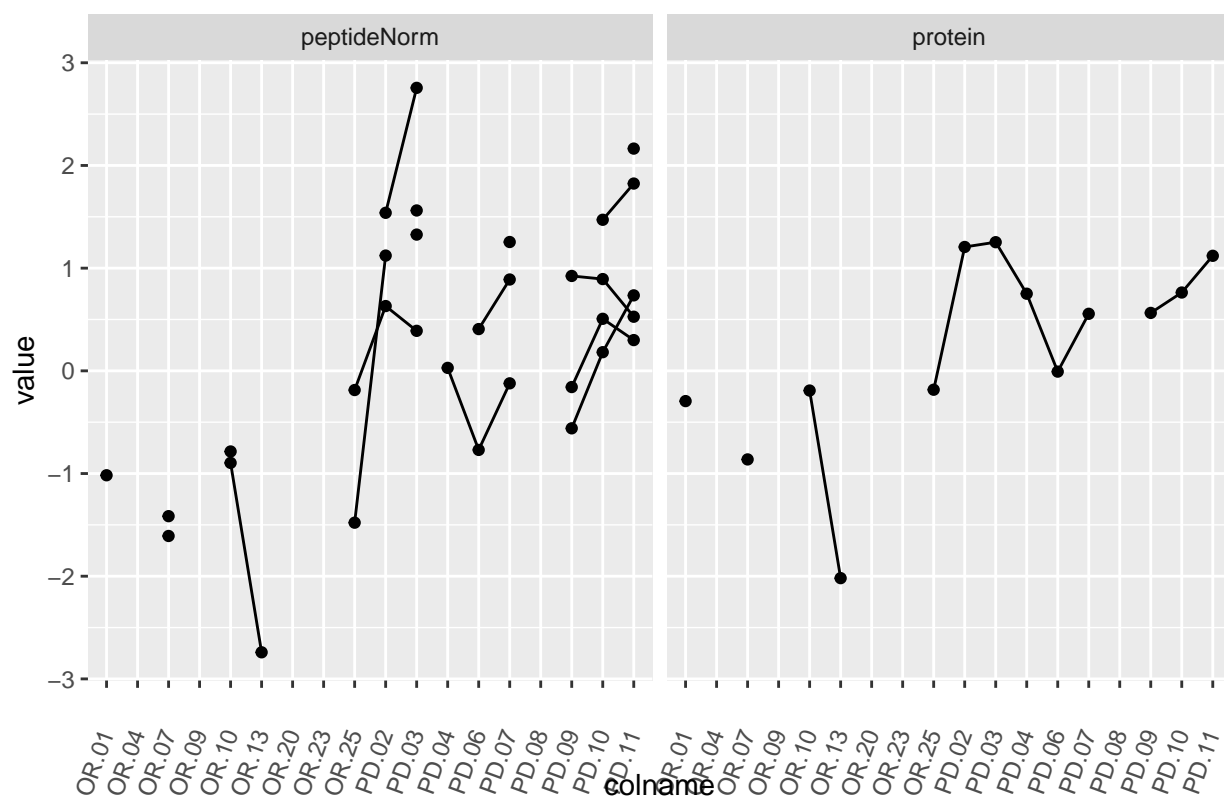
P42345



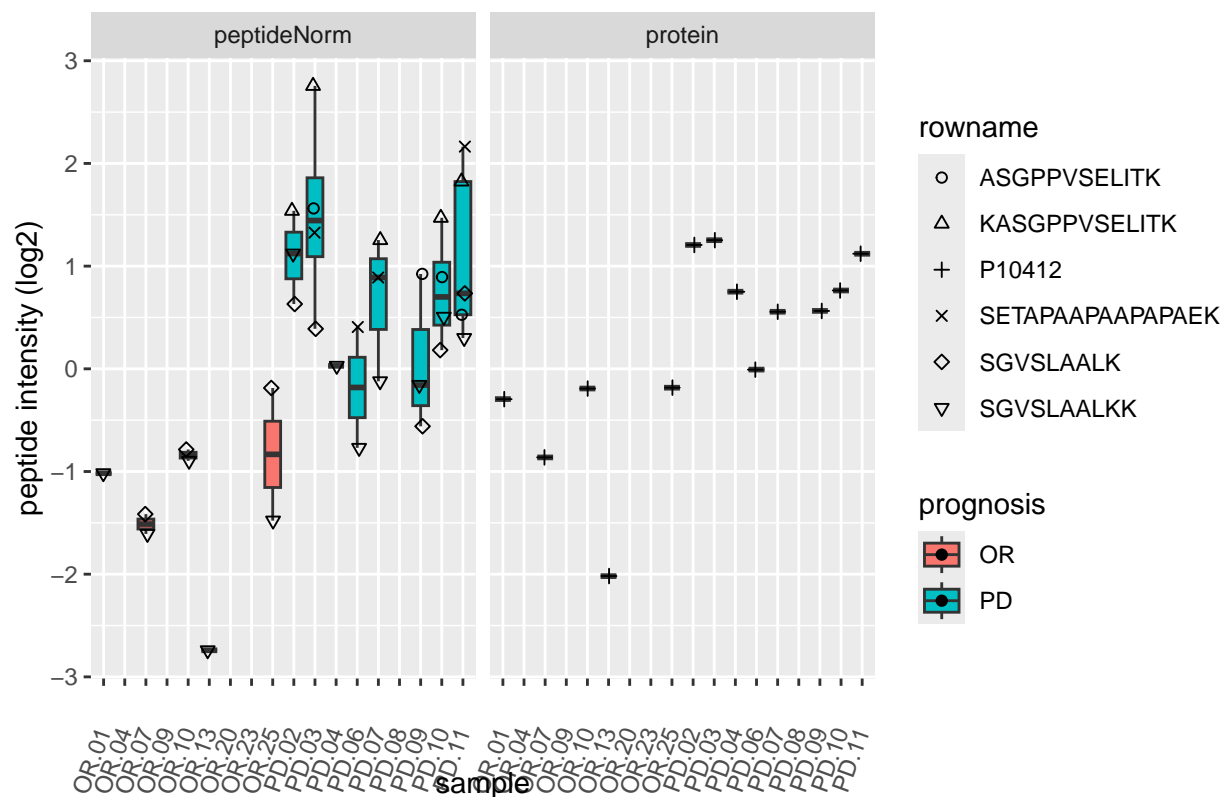
P42345



P10412

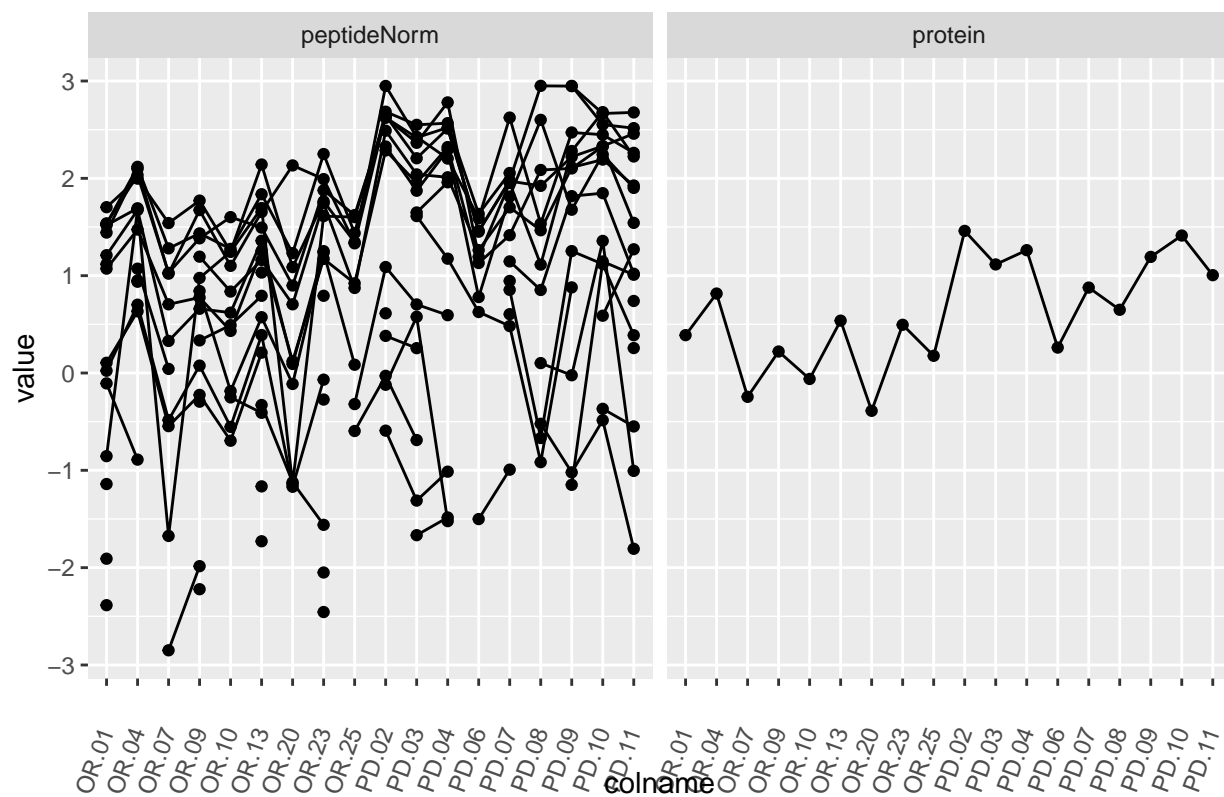


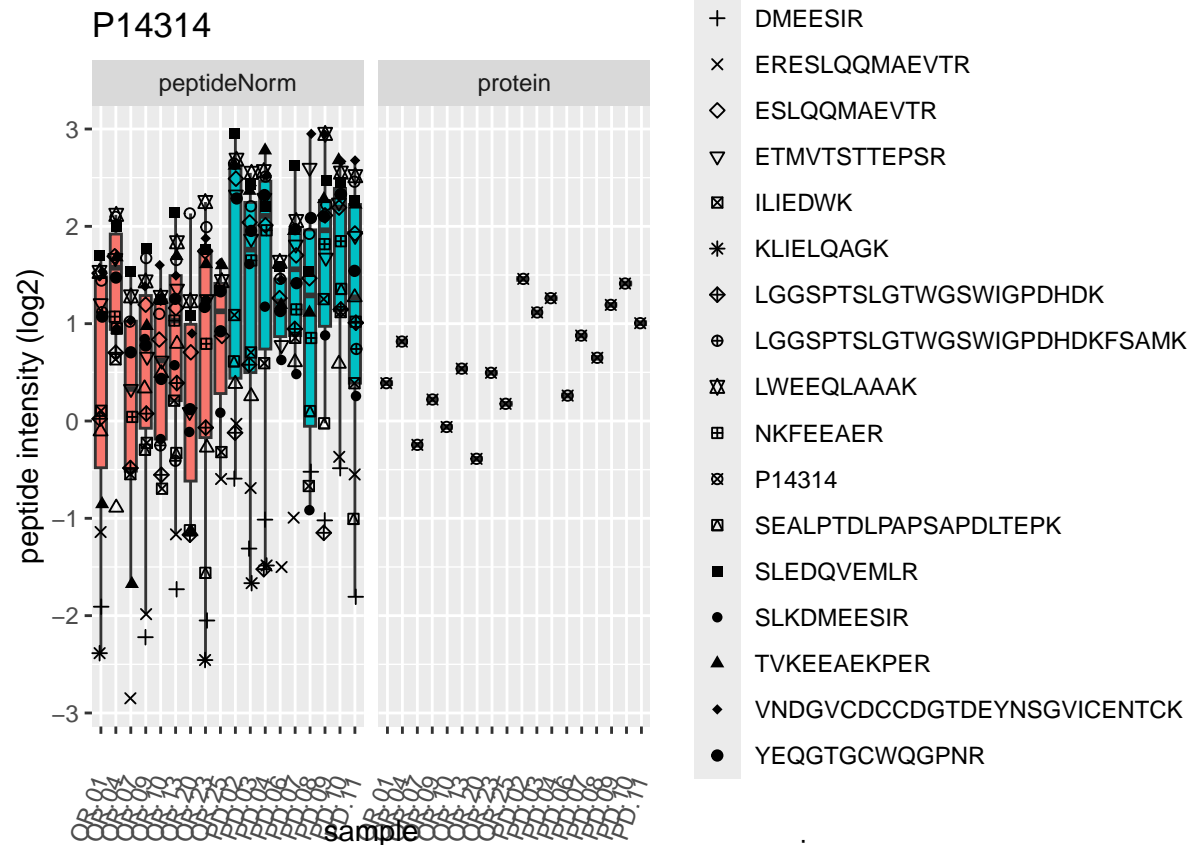
P10412





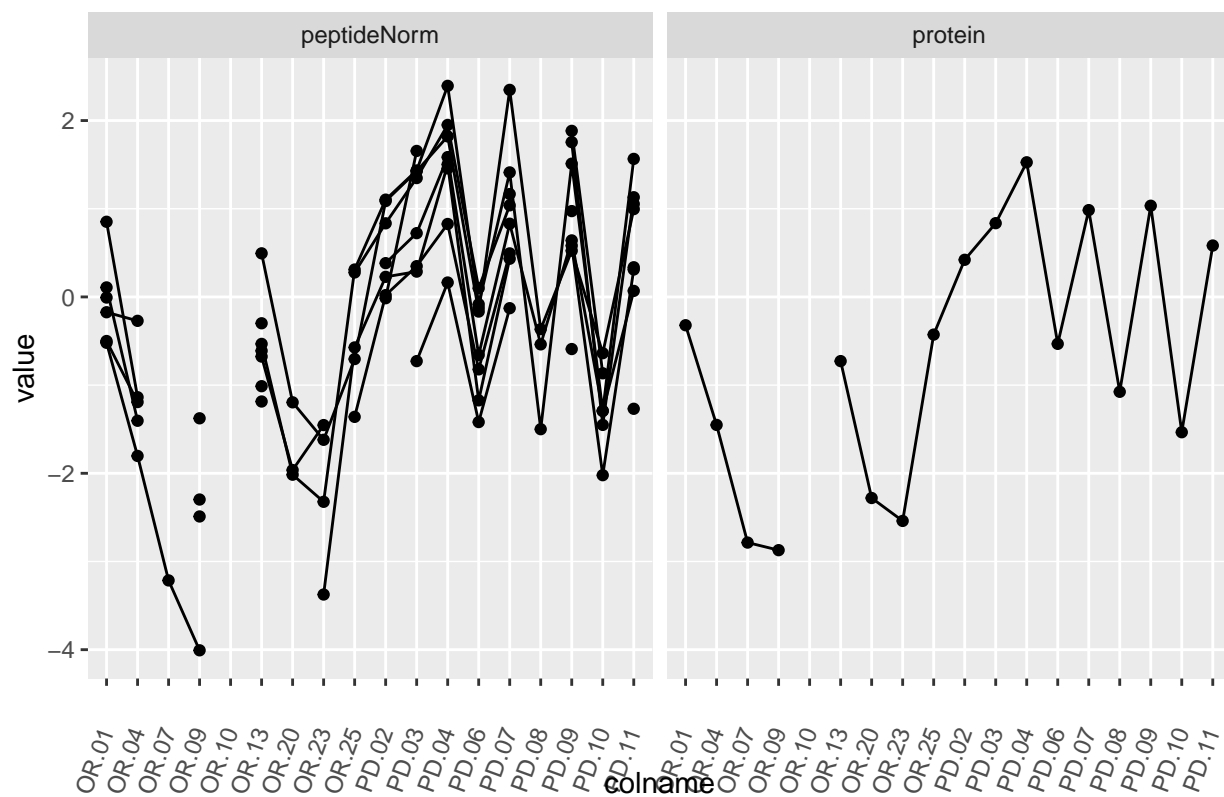
P14314



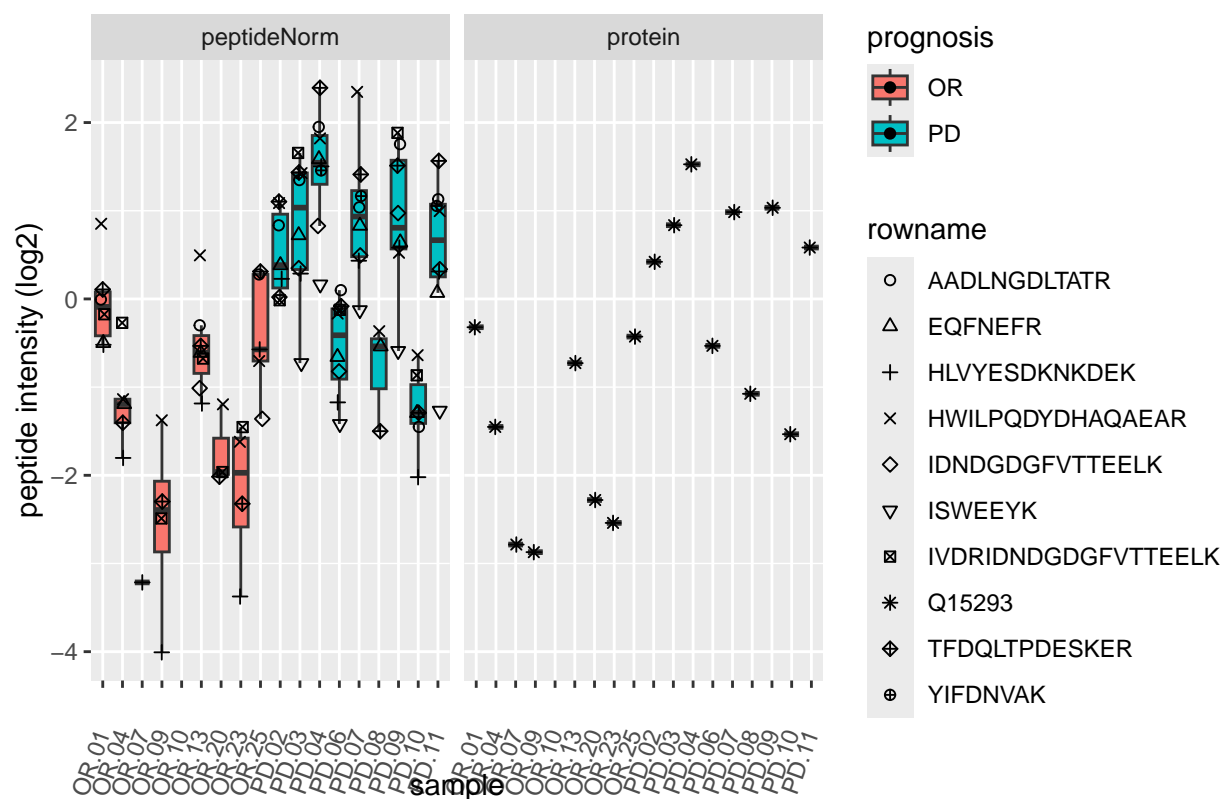


prognosis

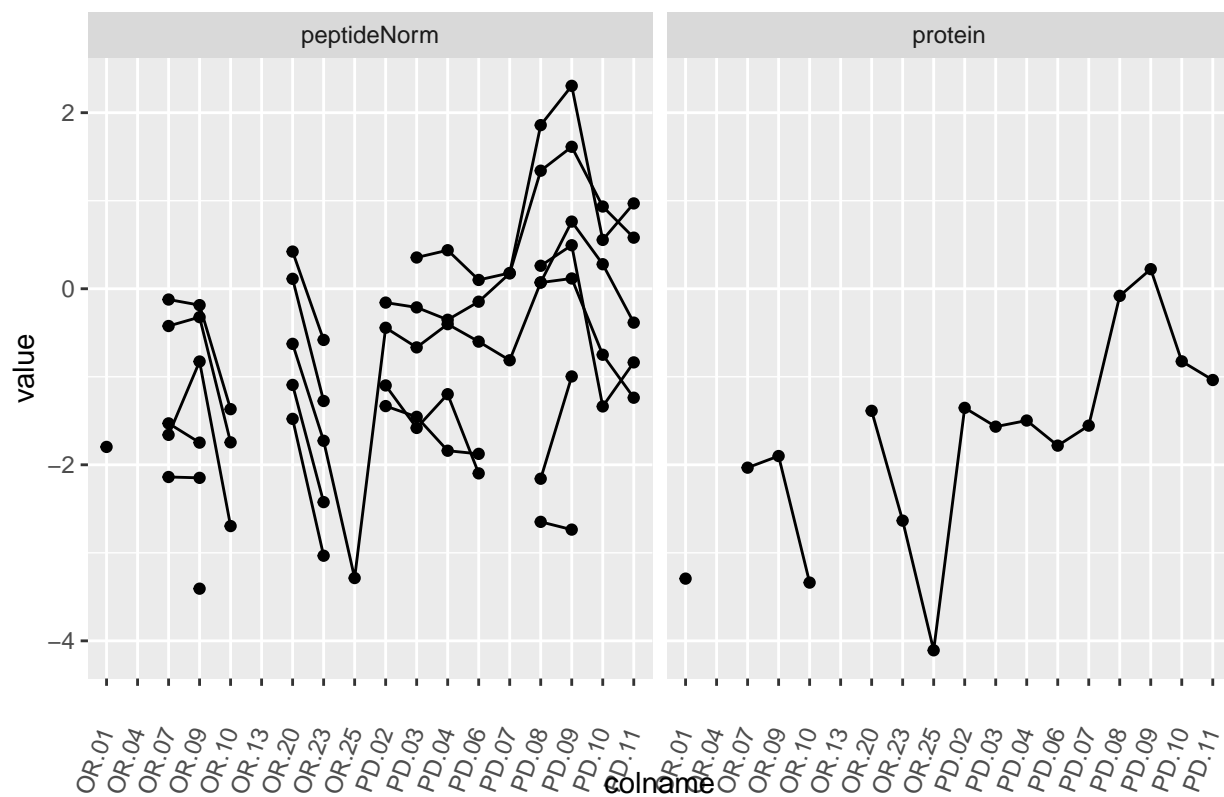
Q15293



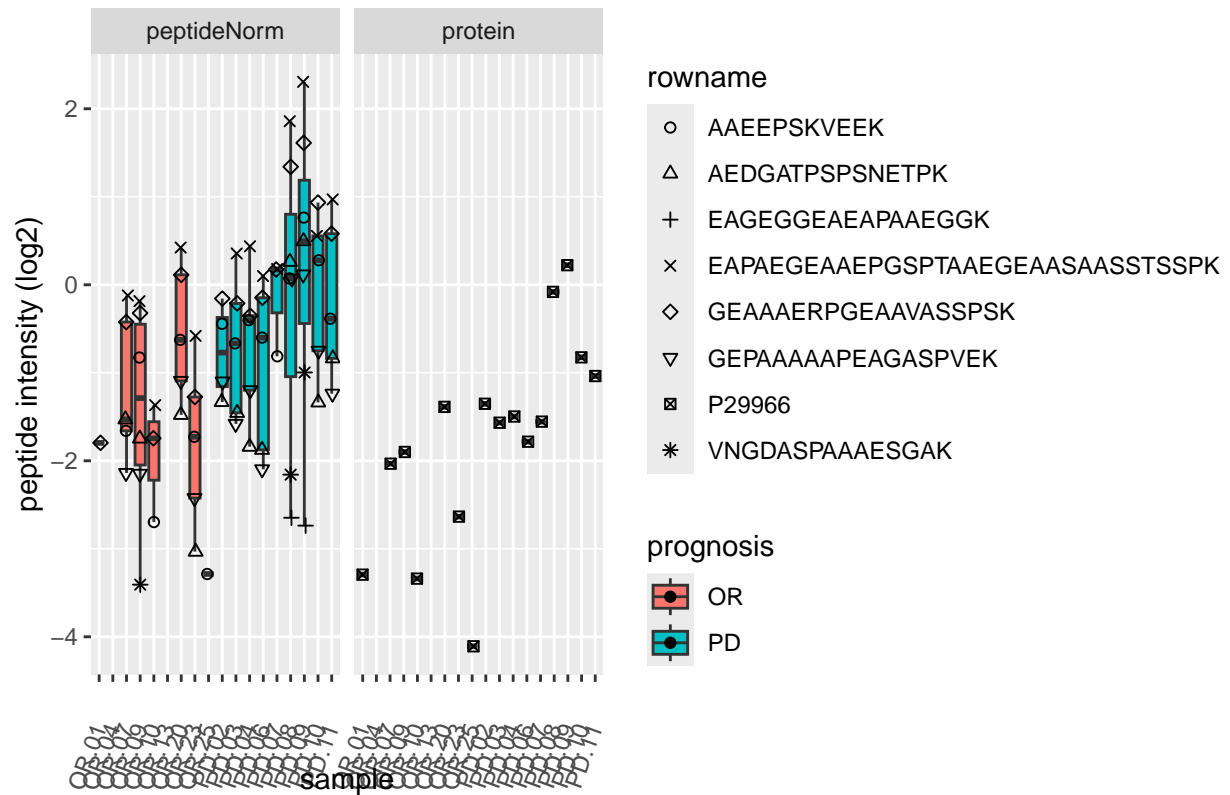
Q15293



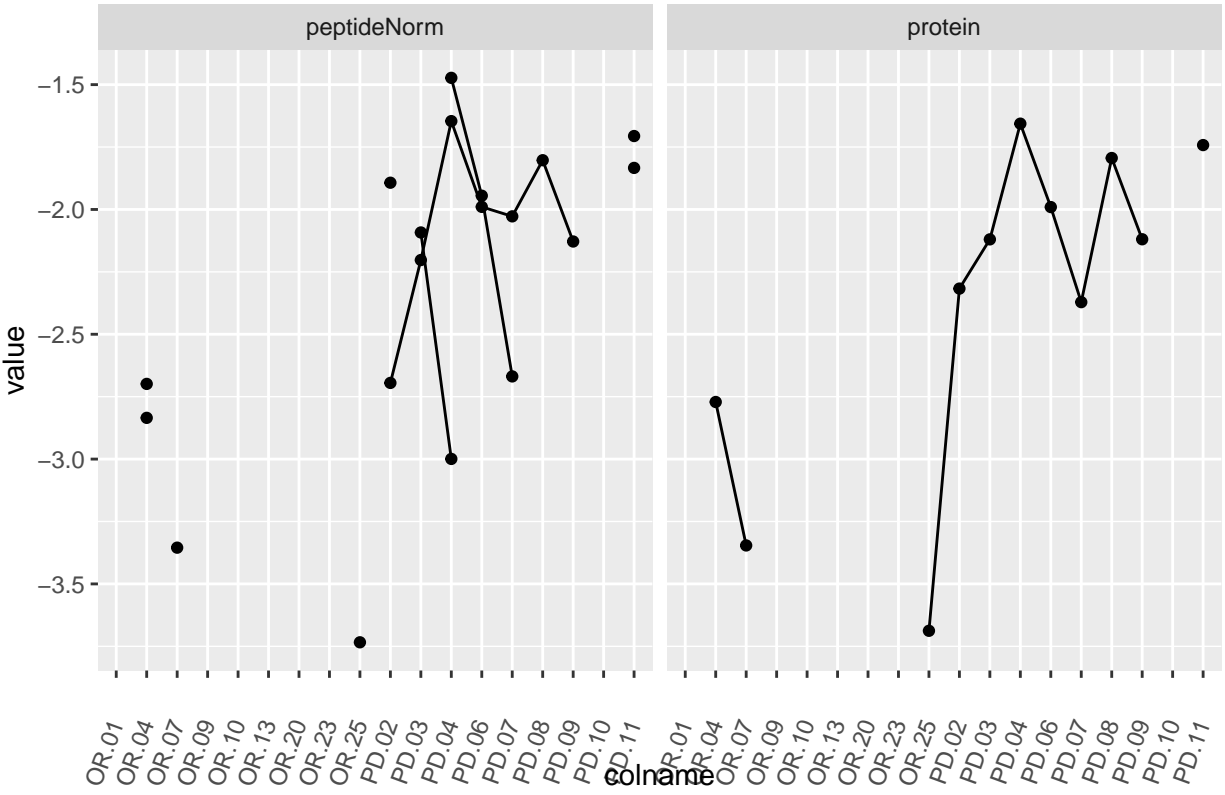
P29966



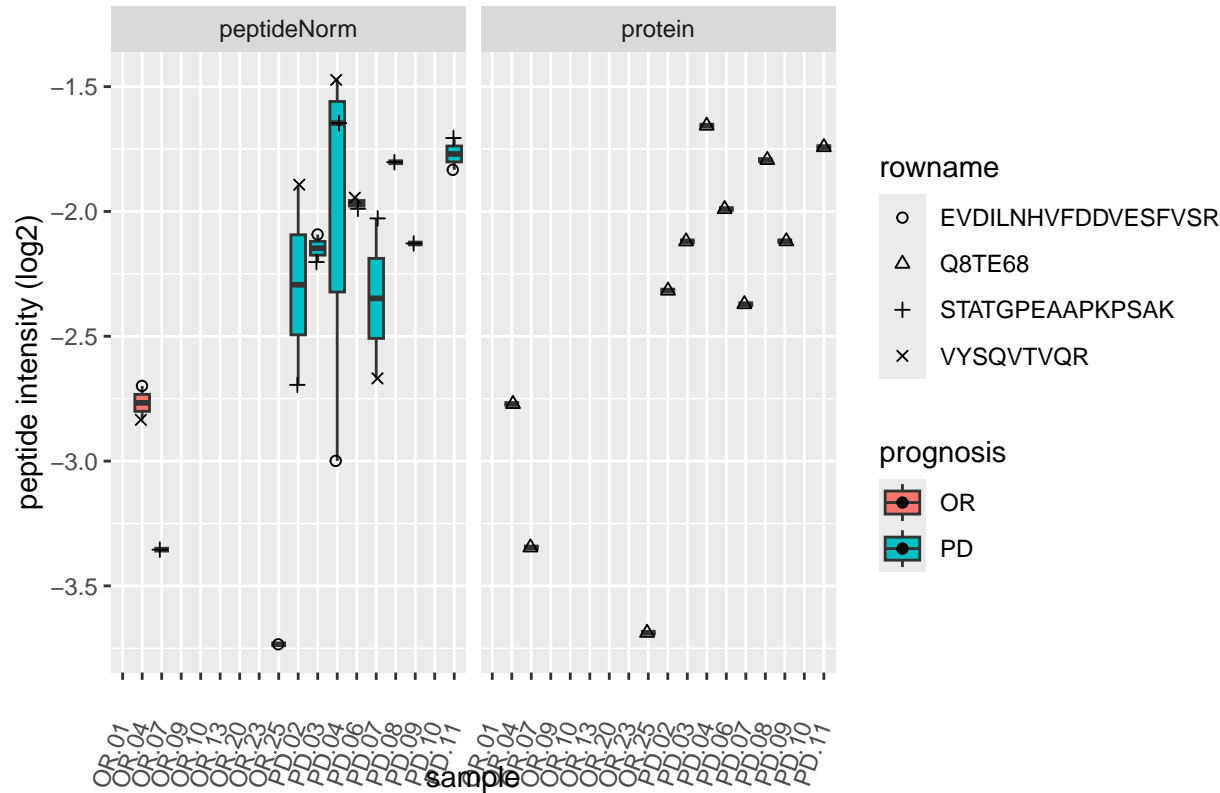
P29966



Q8TE68

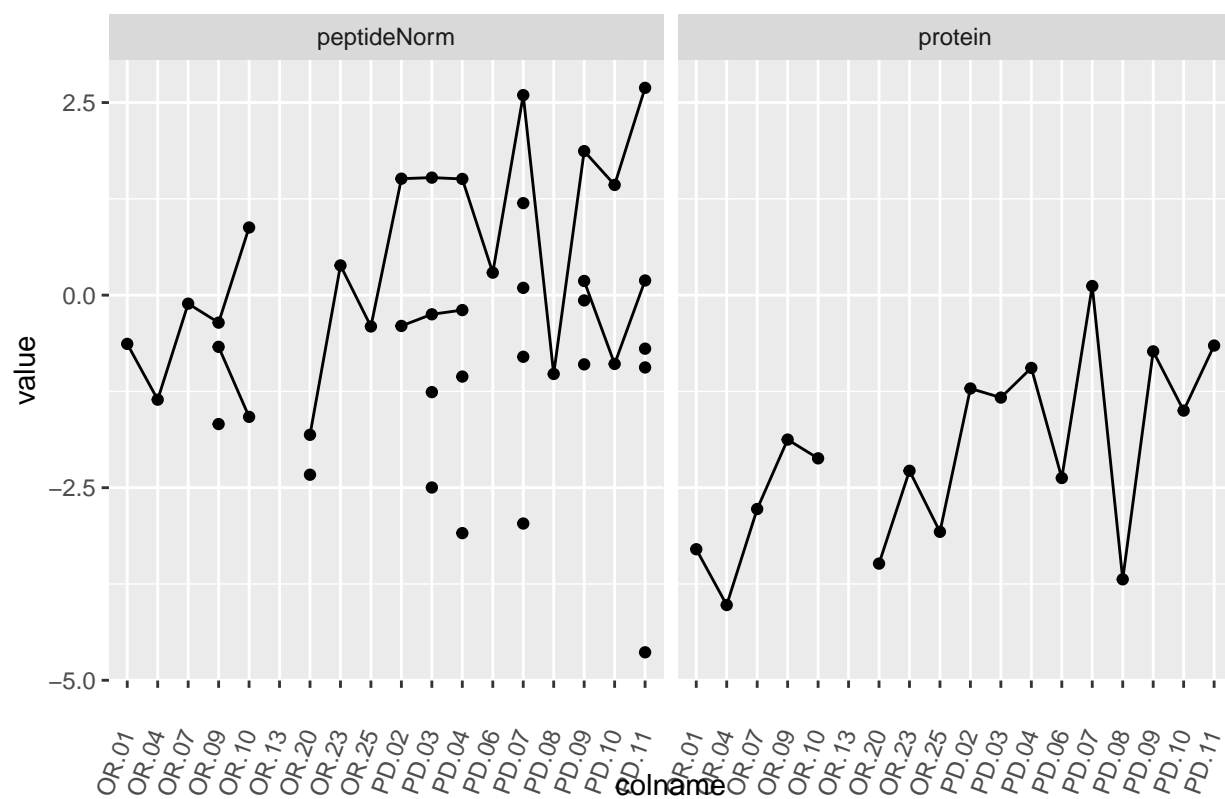


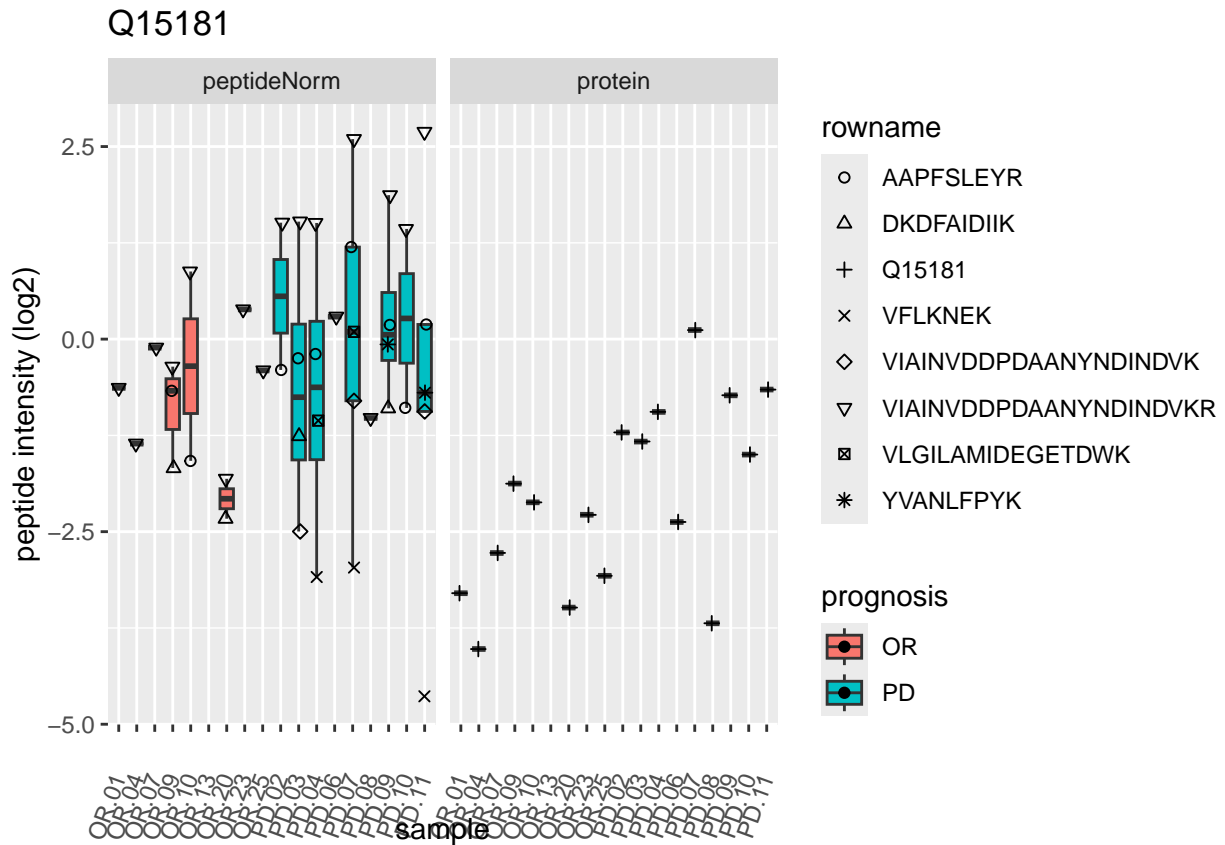
Q8TE68



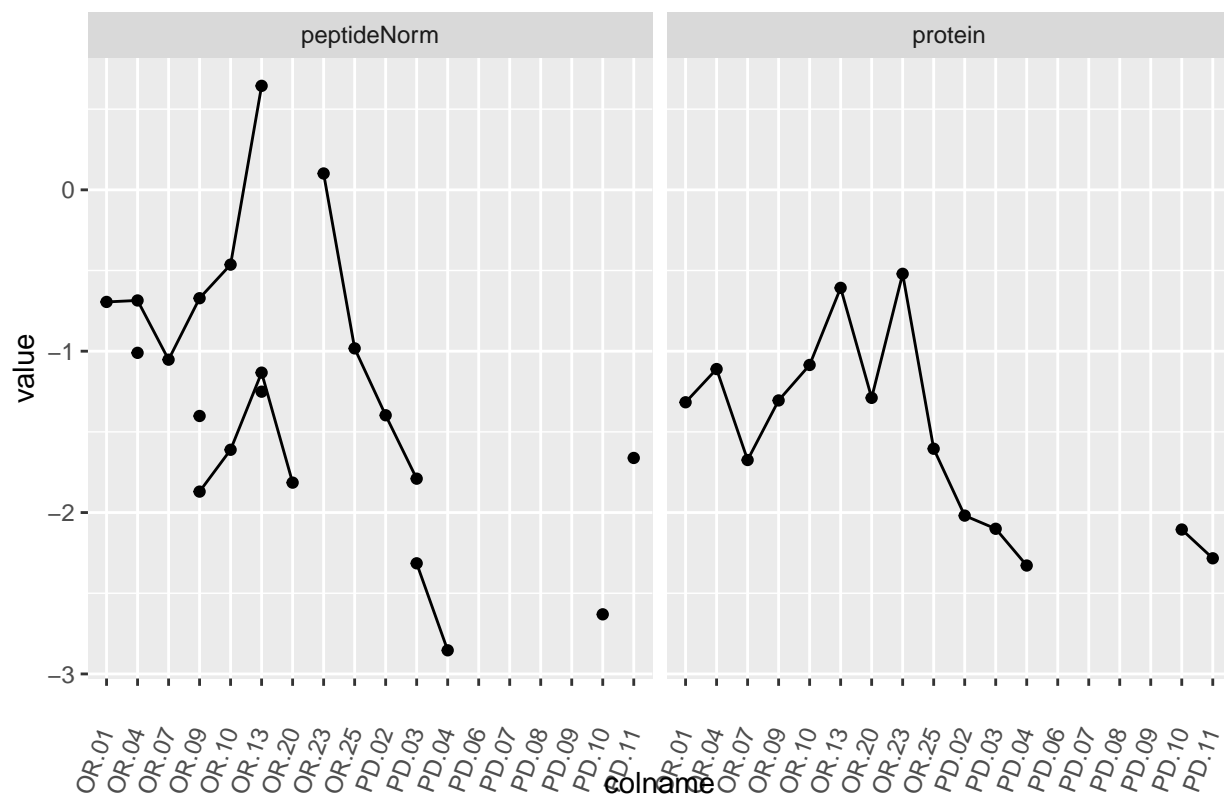


Q15181

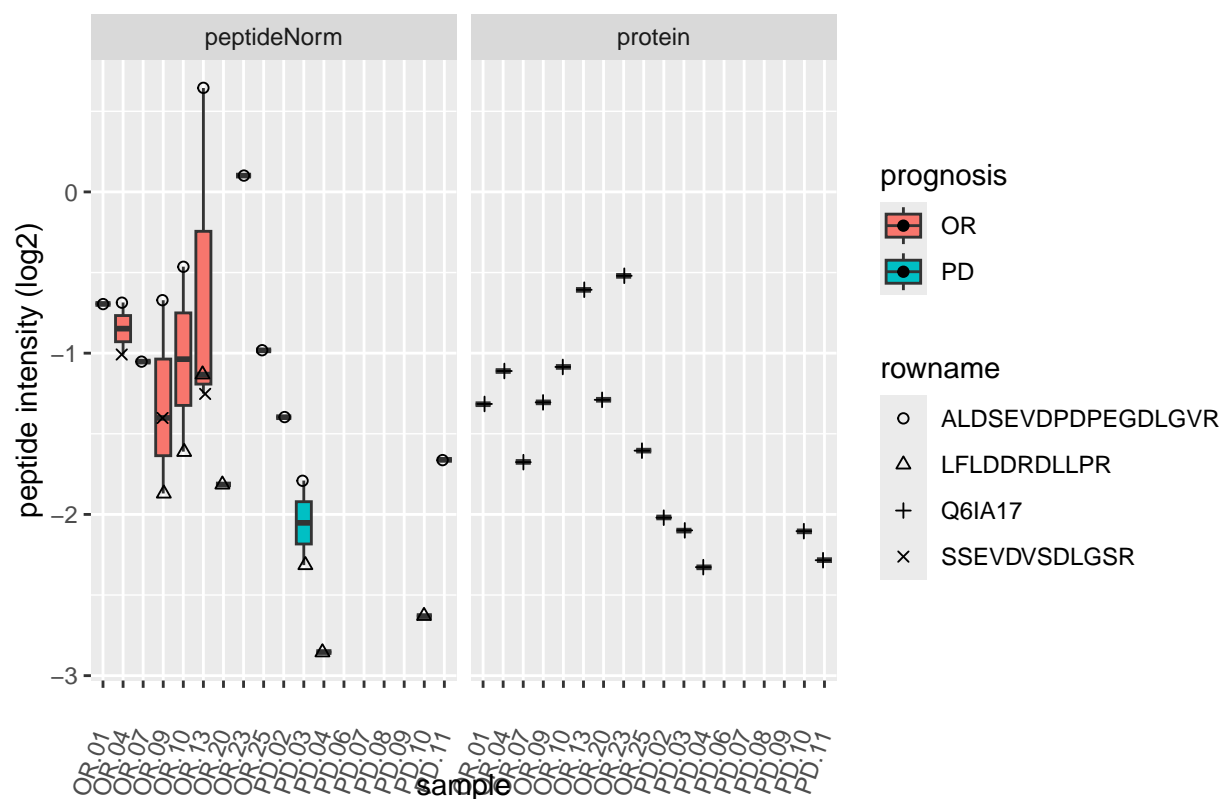




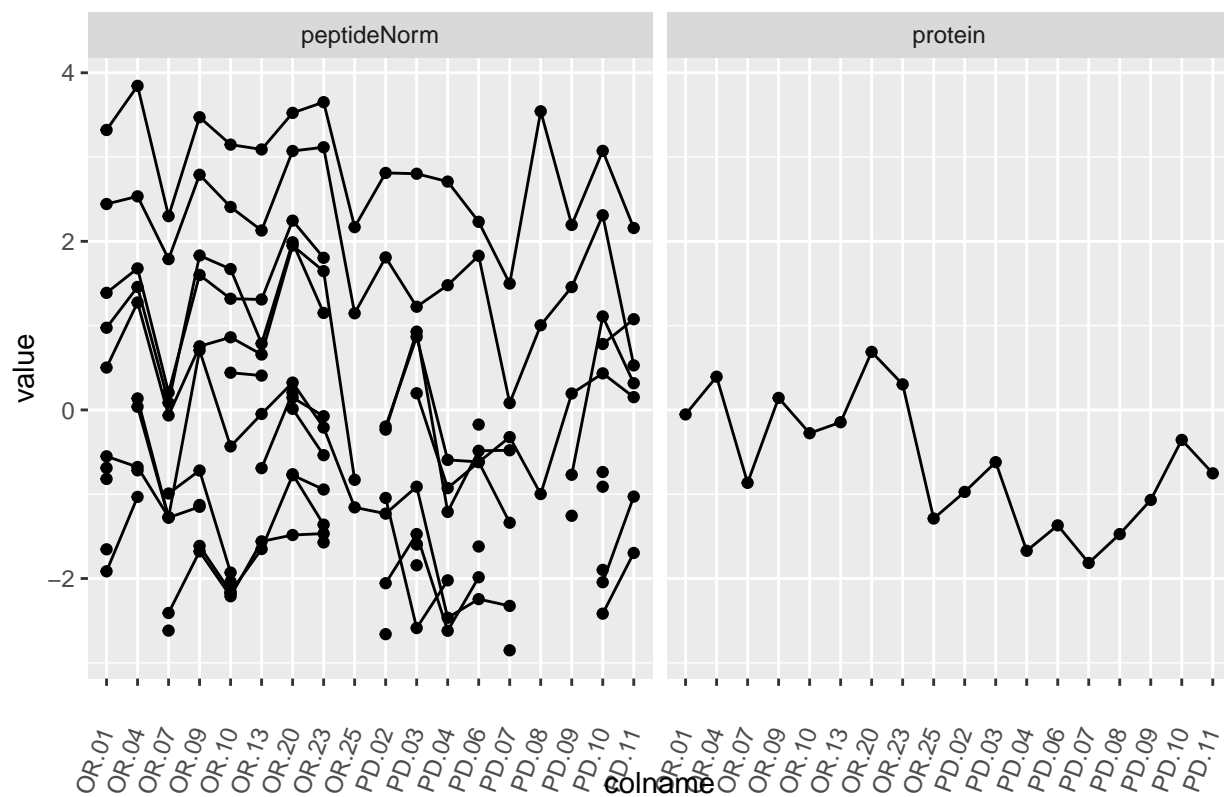
# Q6IA17

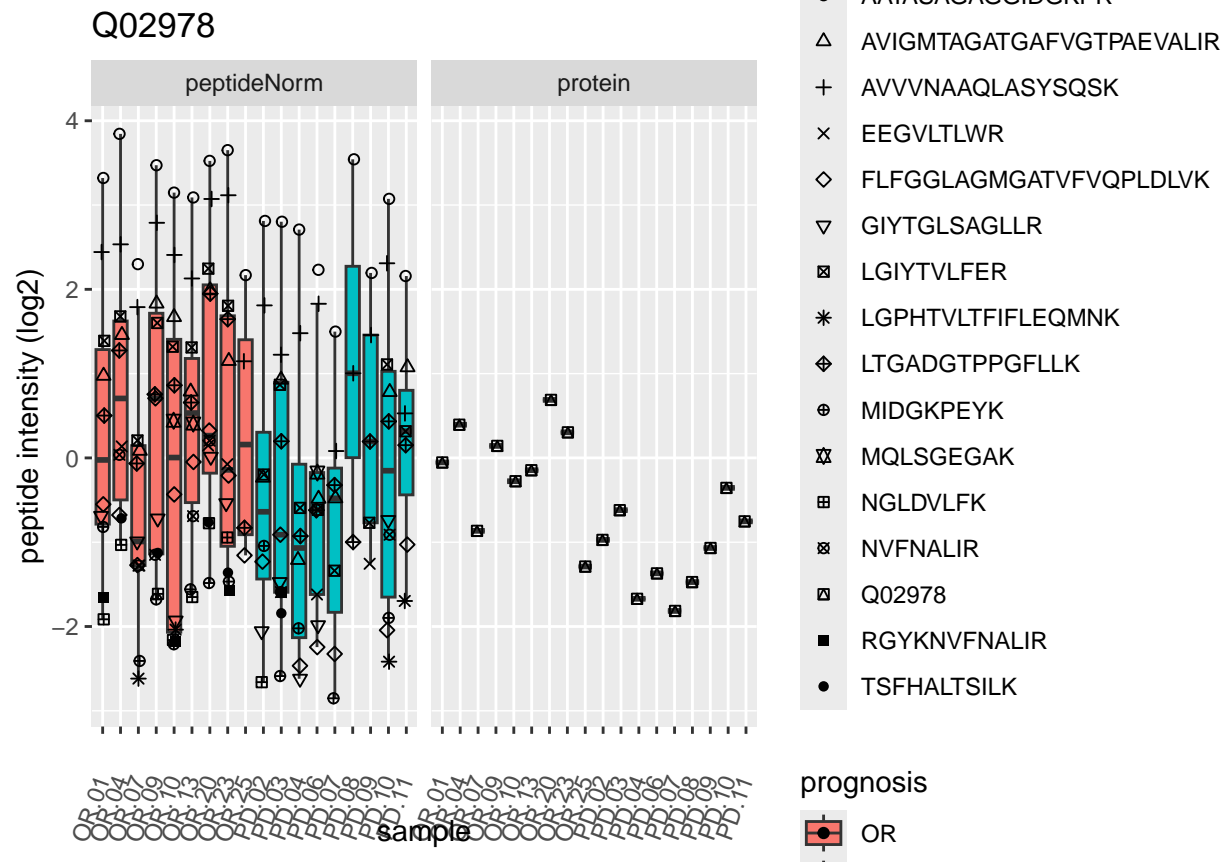


# Q6IA17

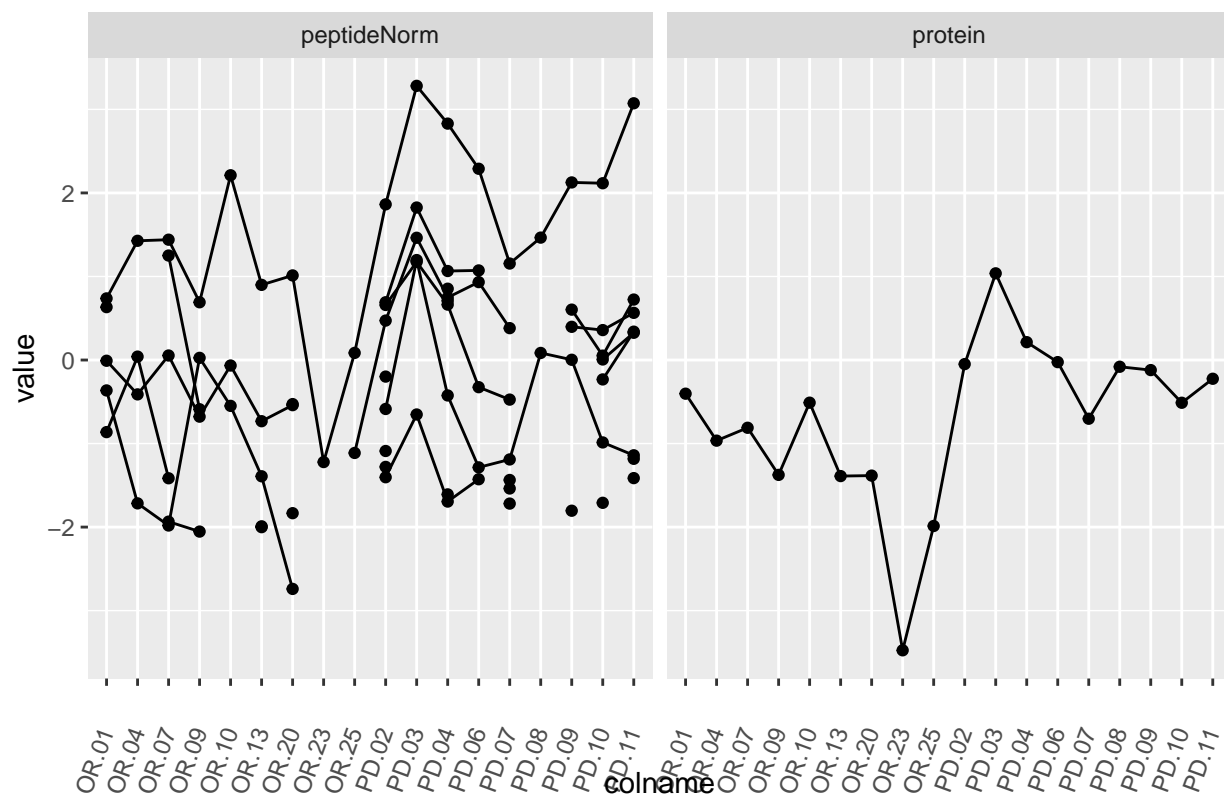


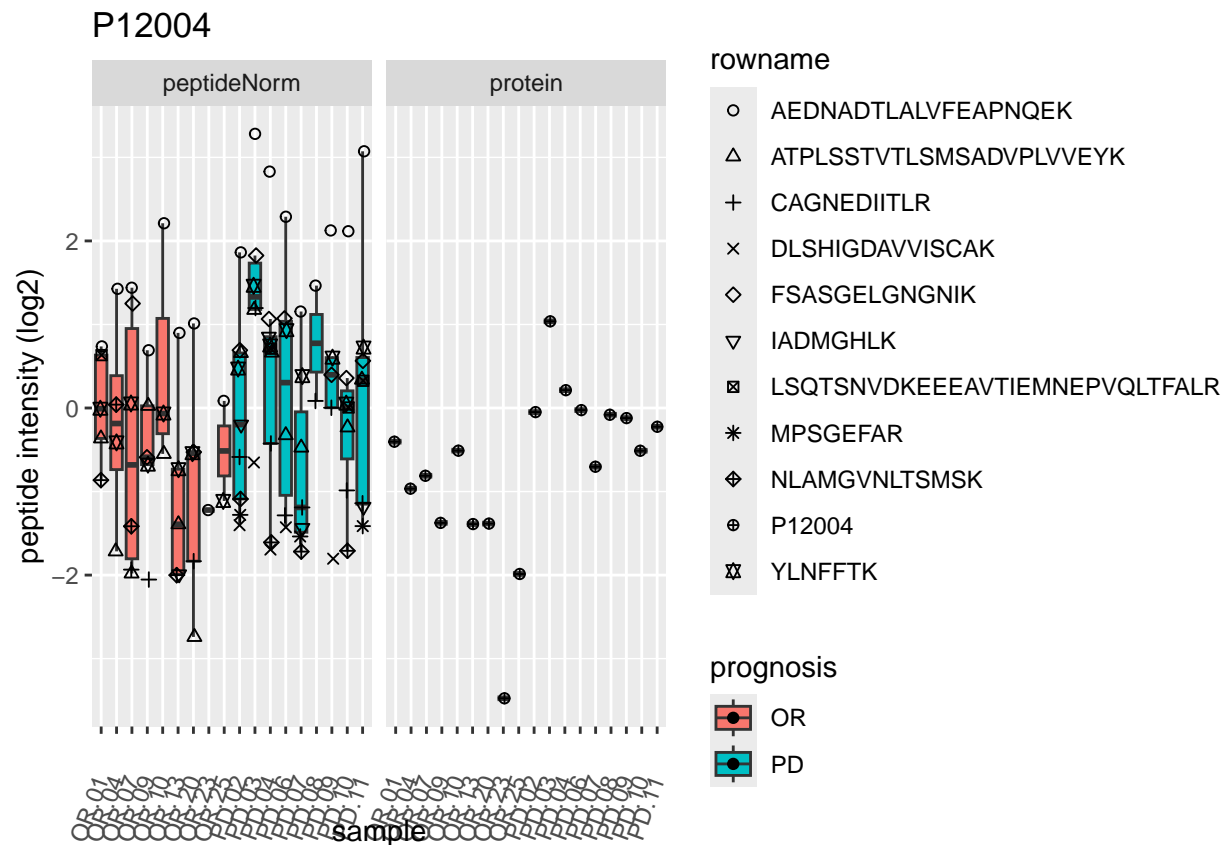
Q02978





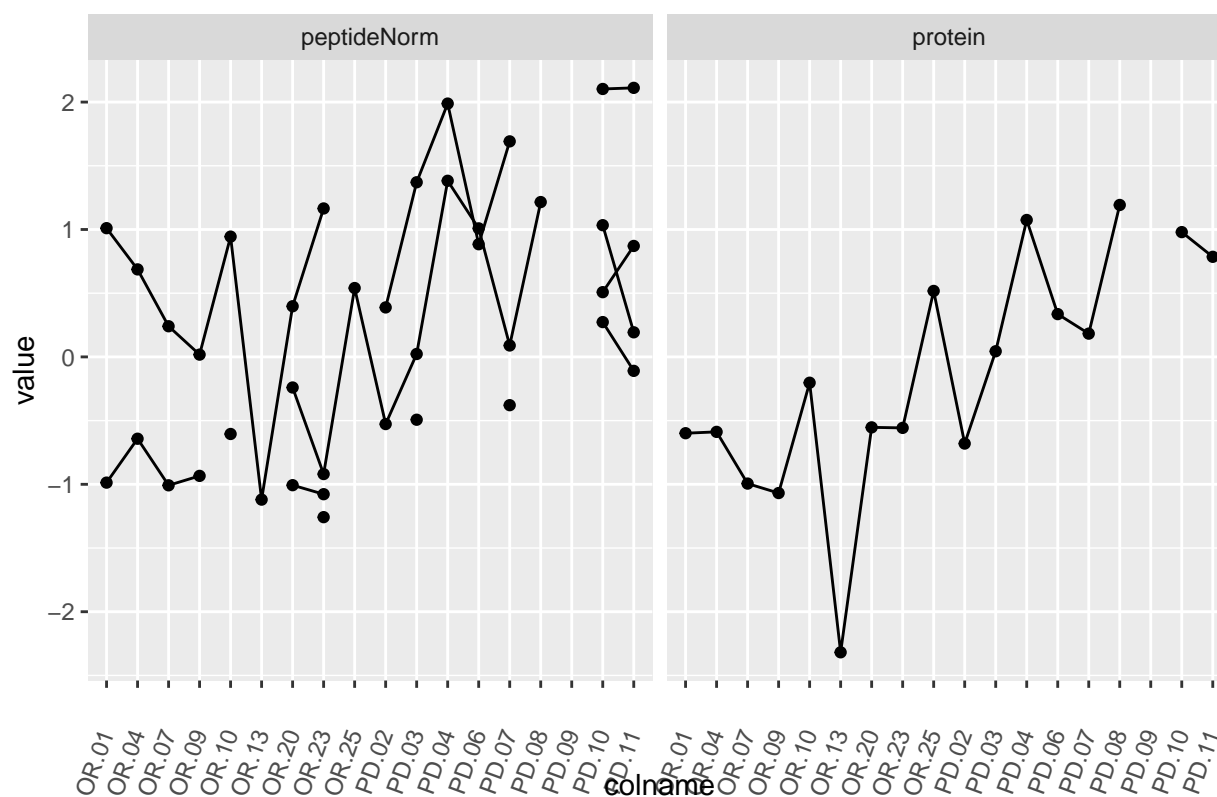
P12004



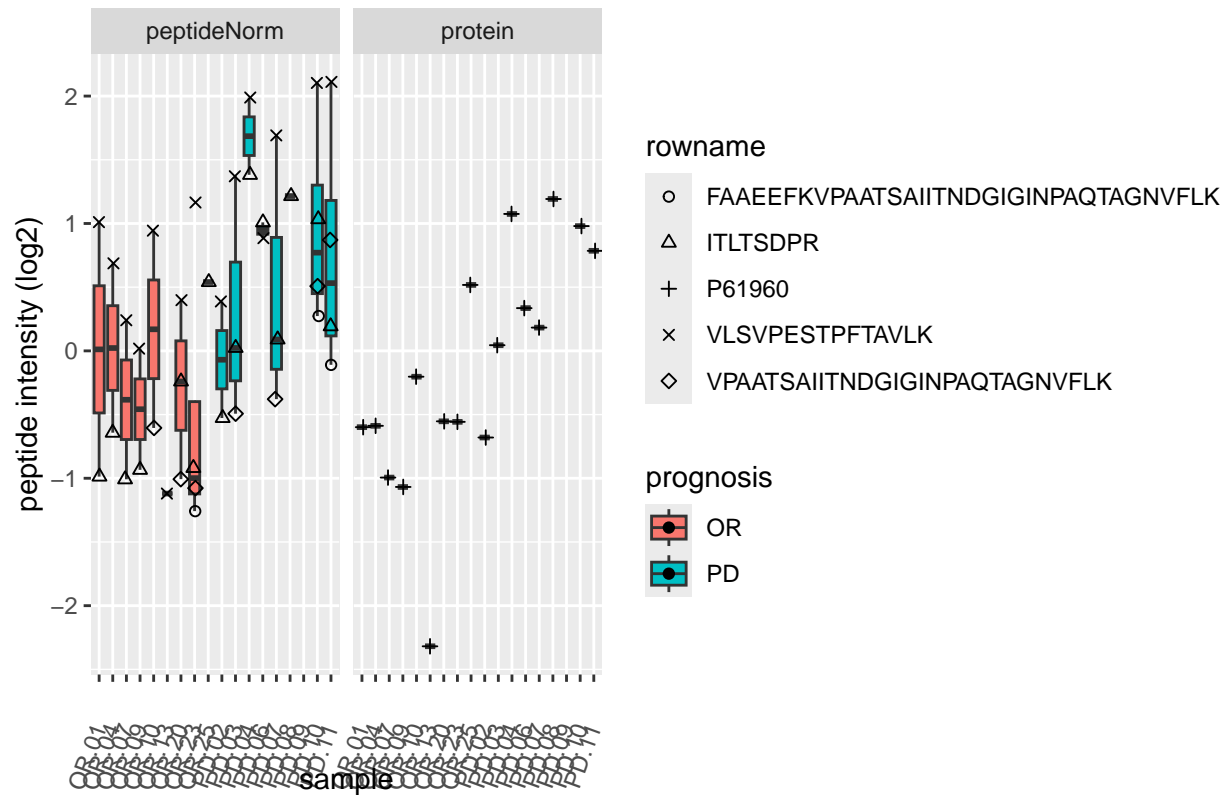




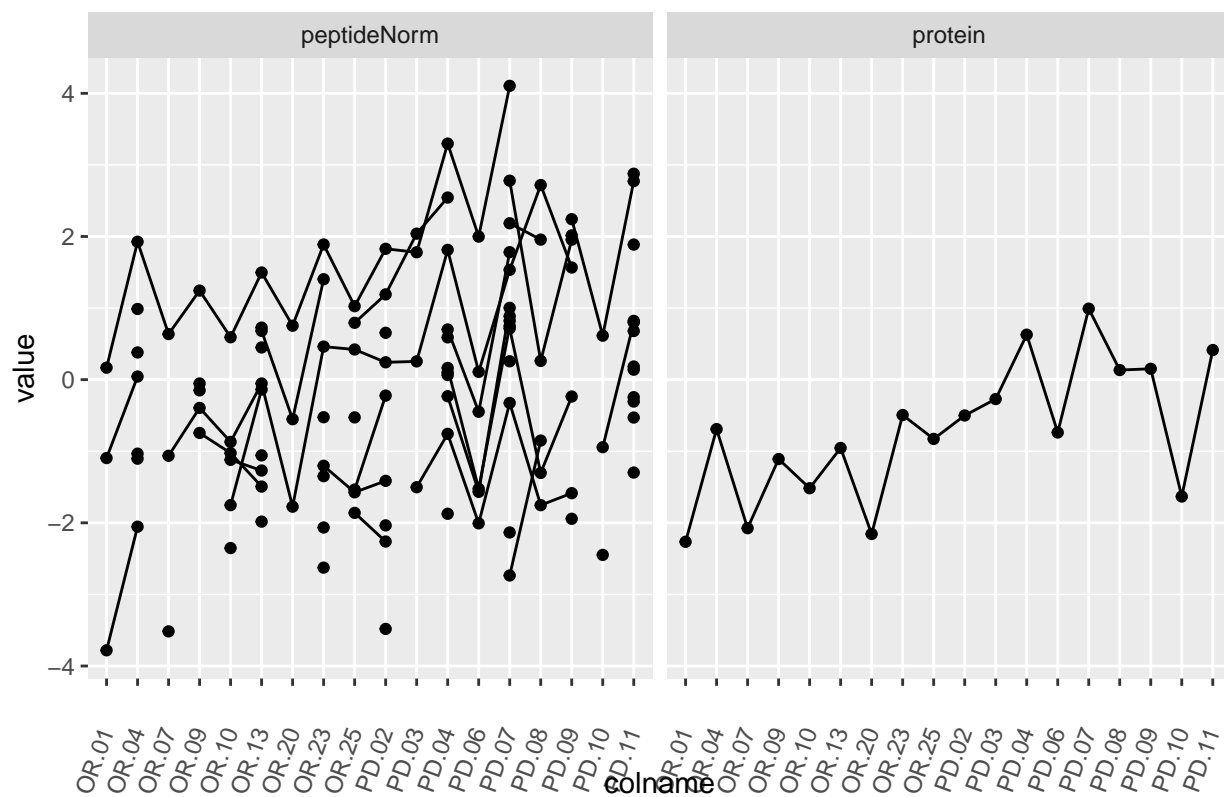
P61960

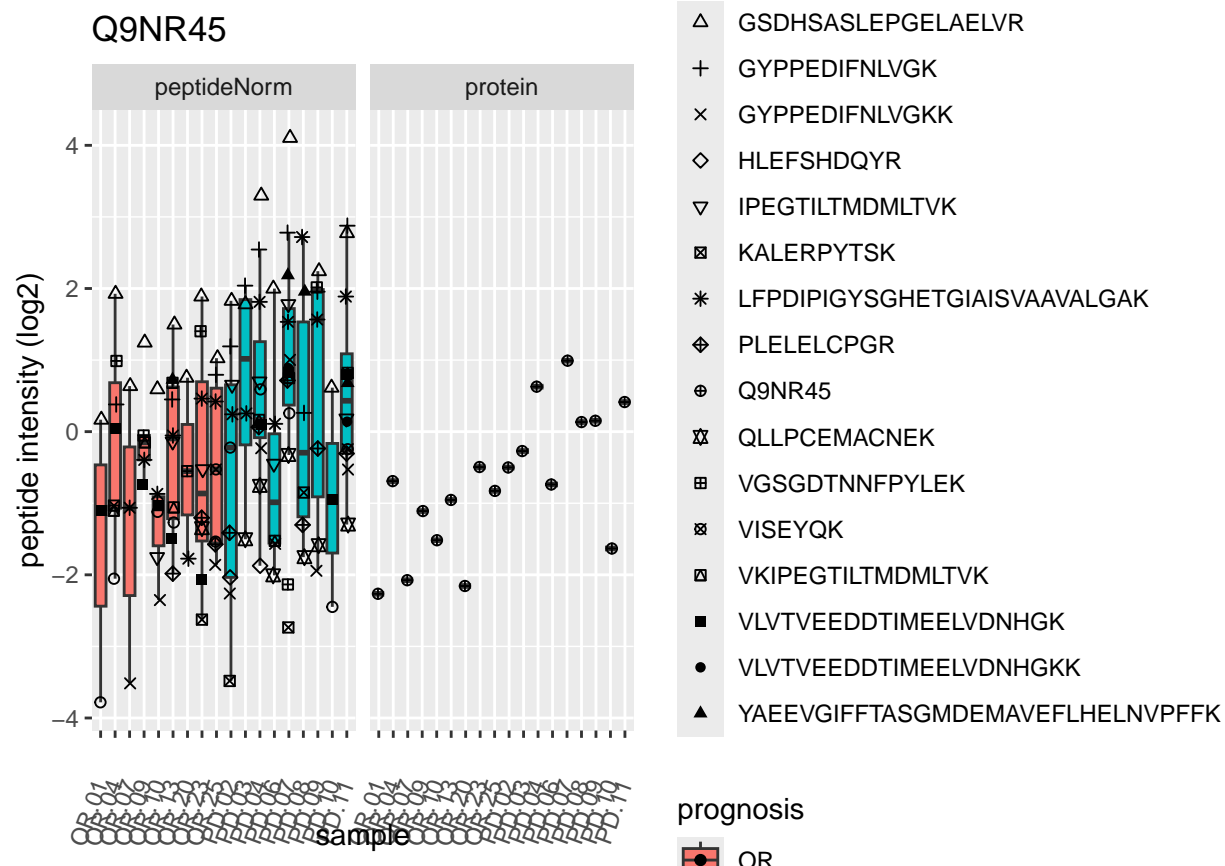


P61960

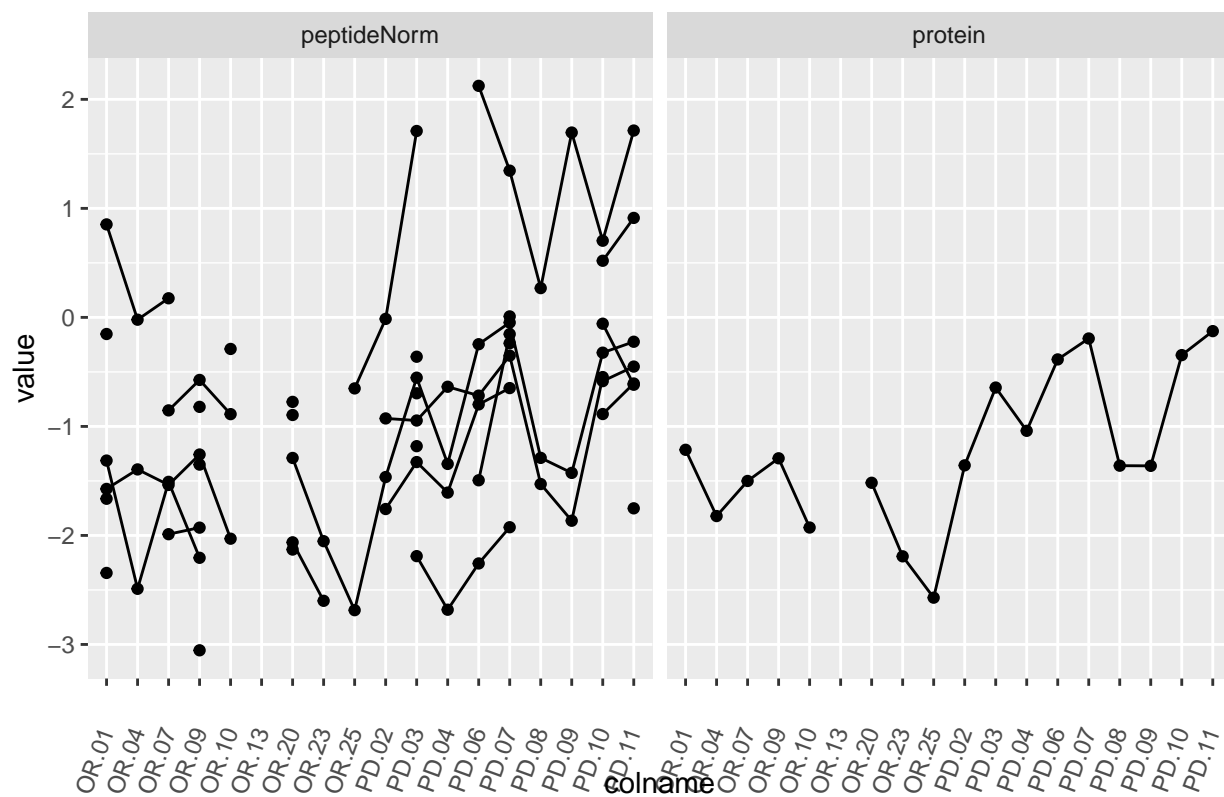


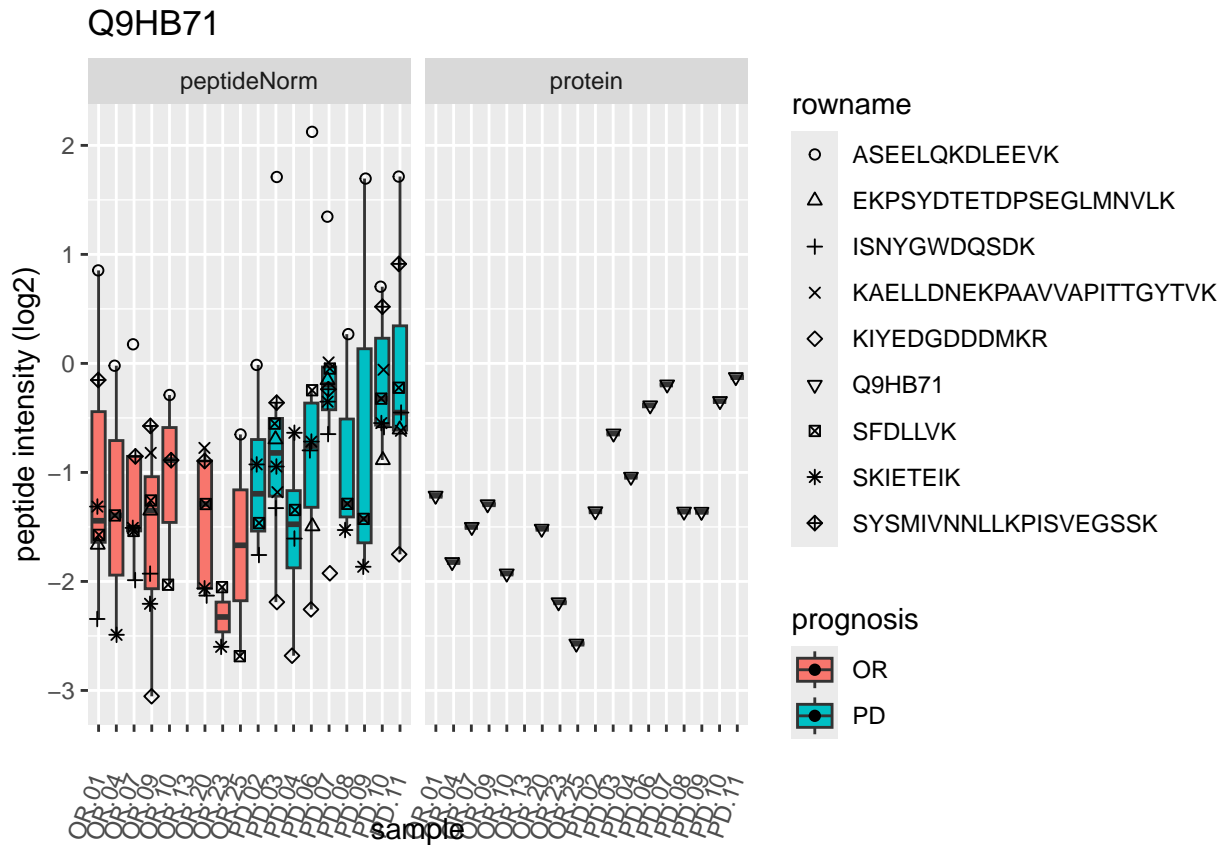
# Q9NR45



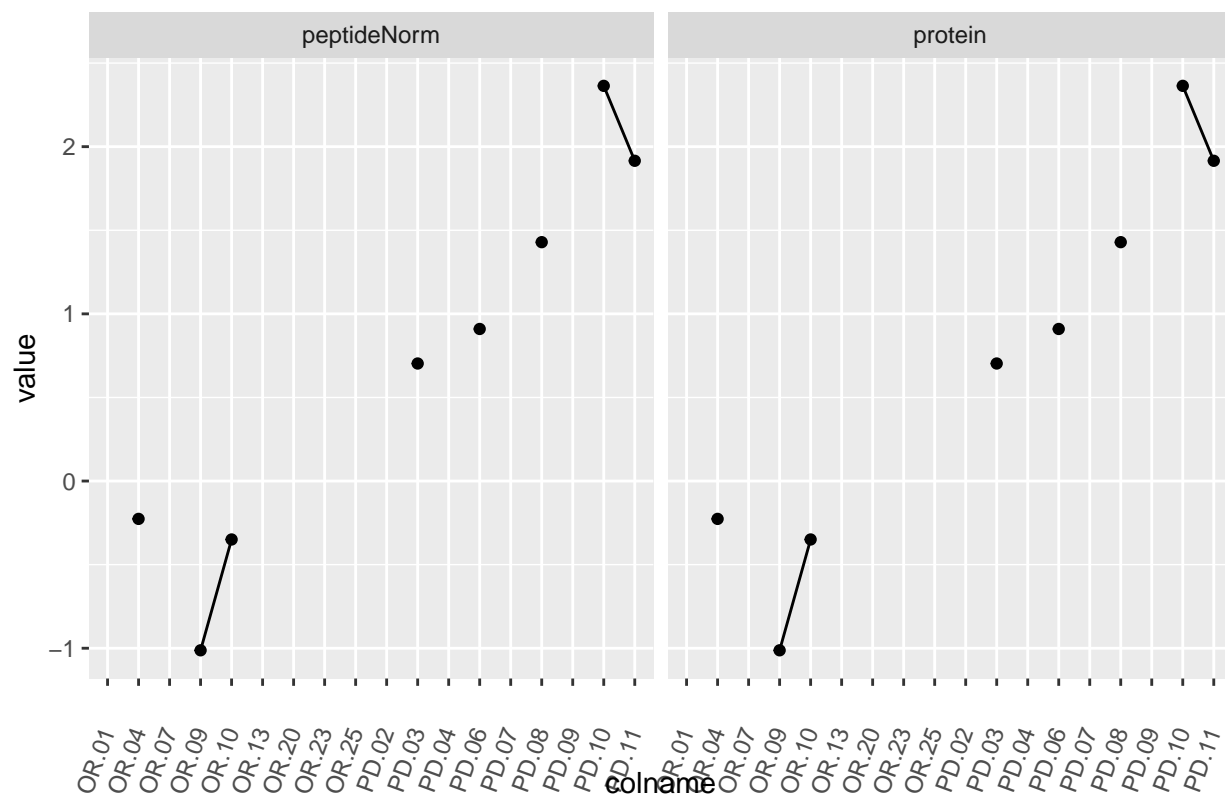


Q9HB71

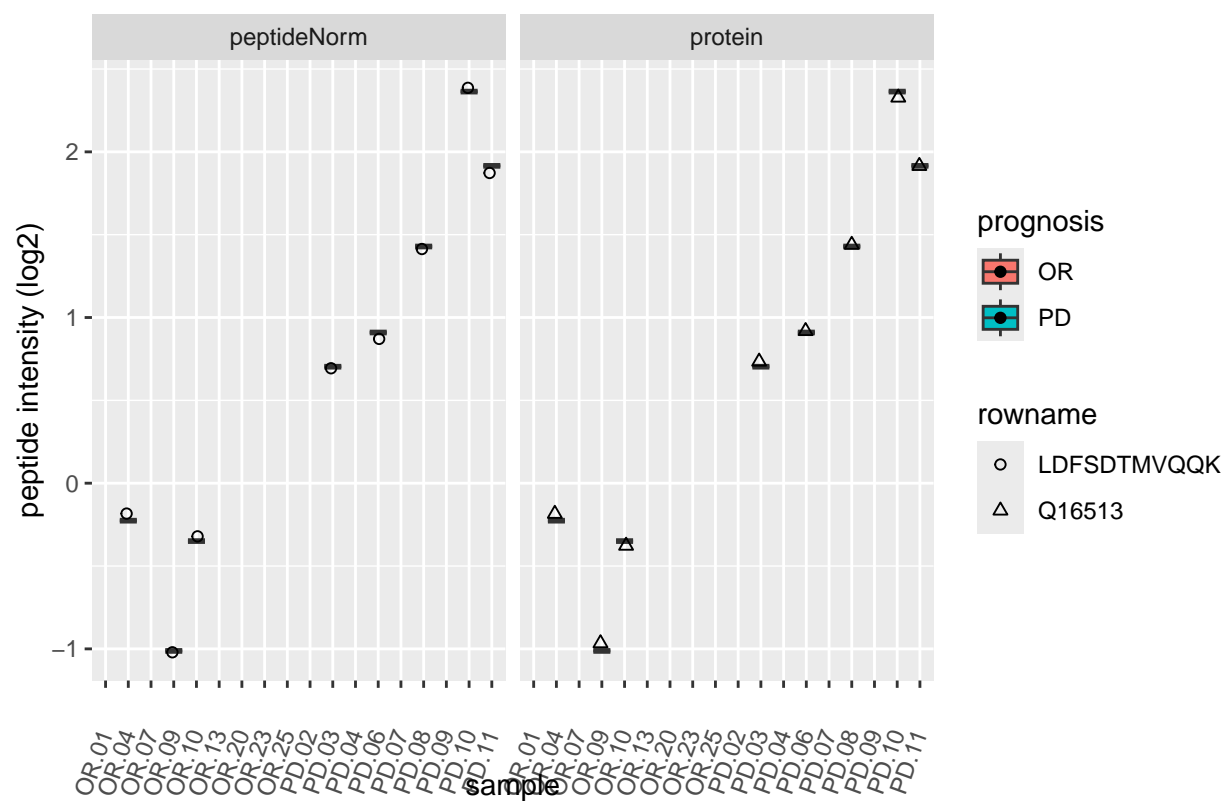




Q16513

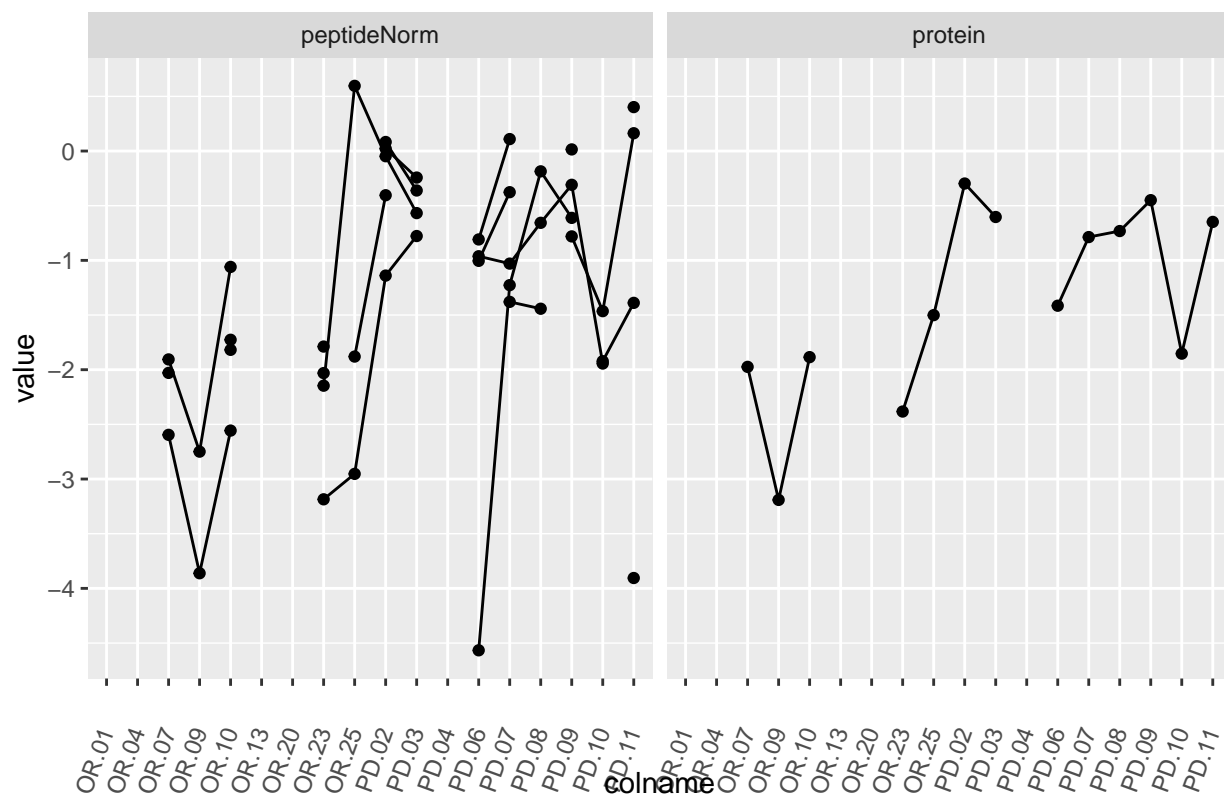


Q16513

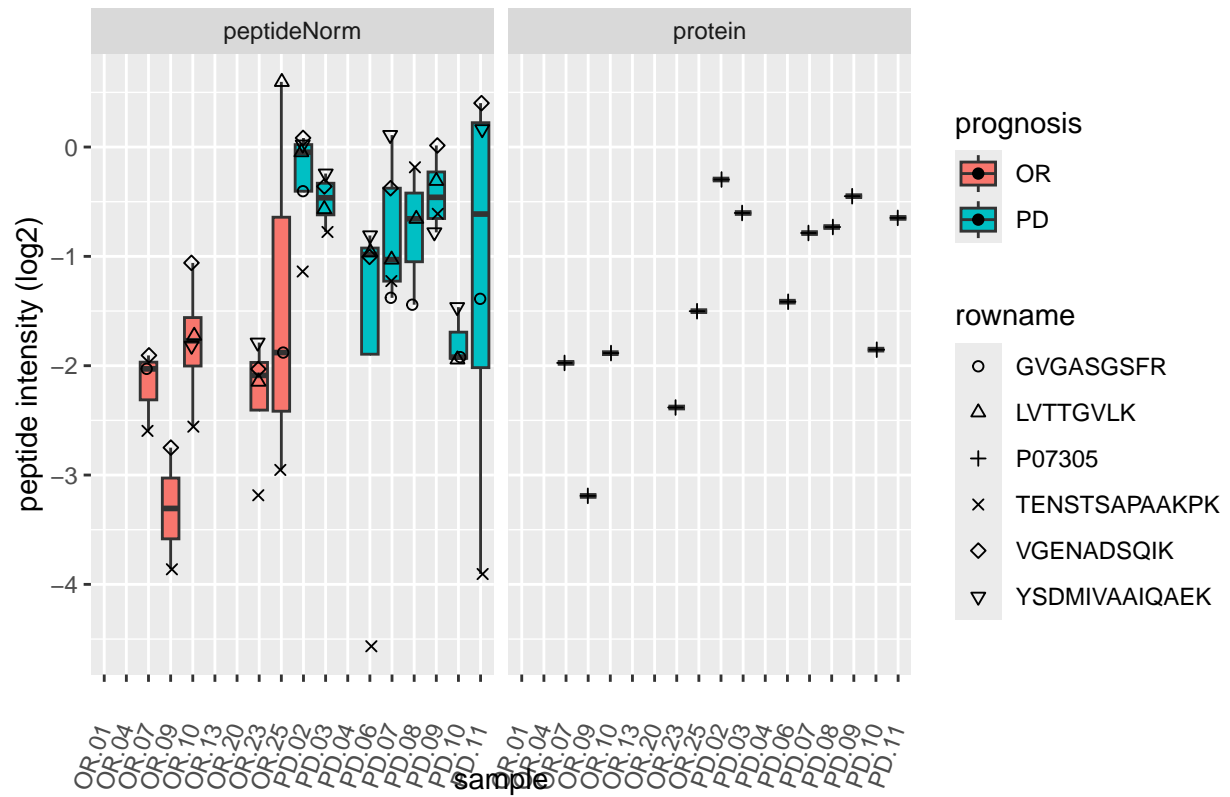




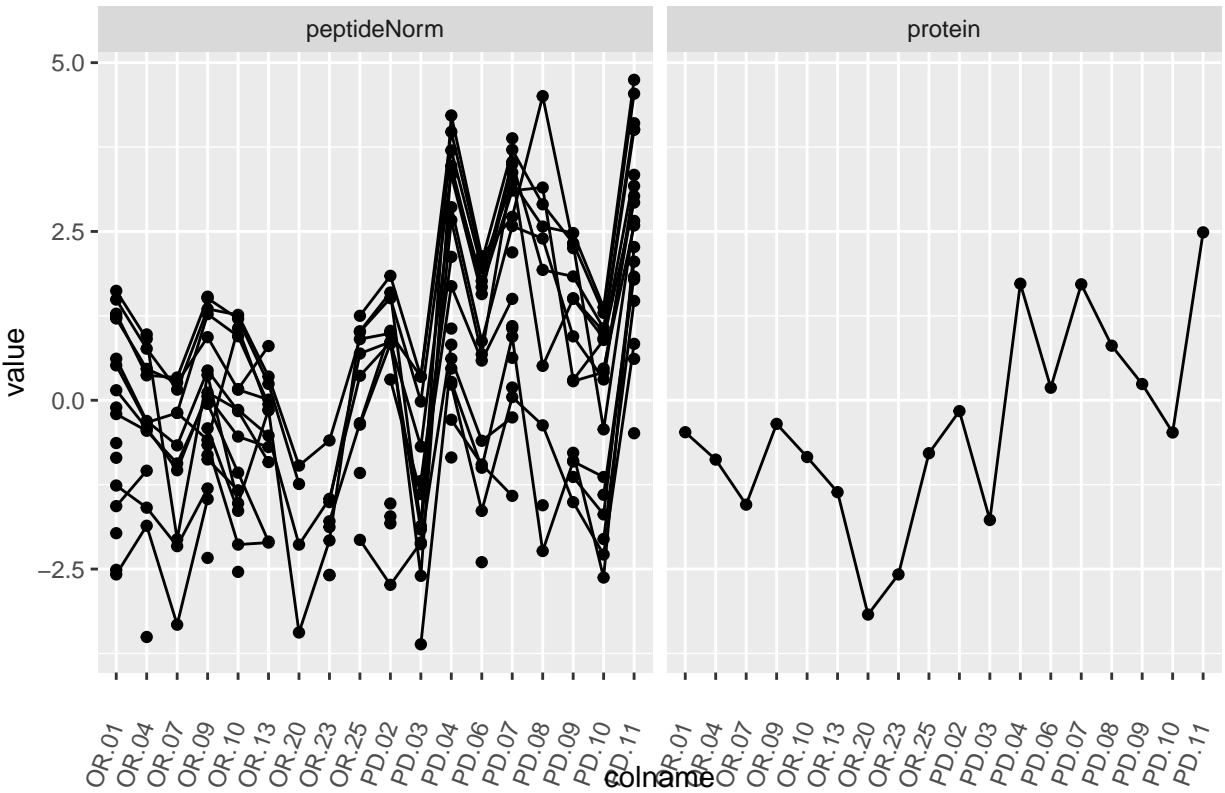
P07305

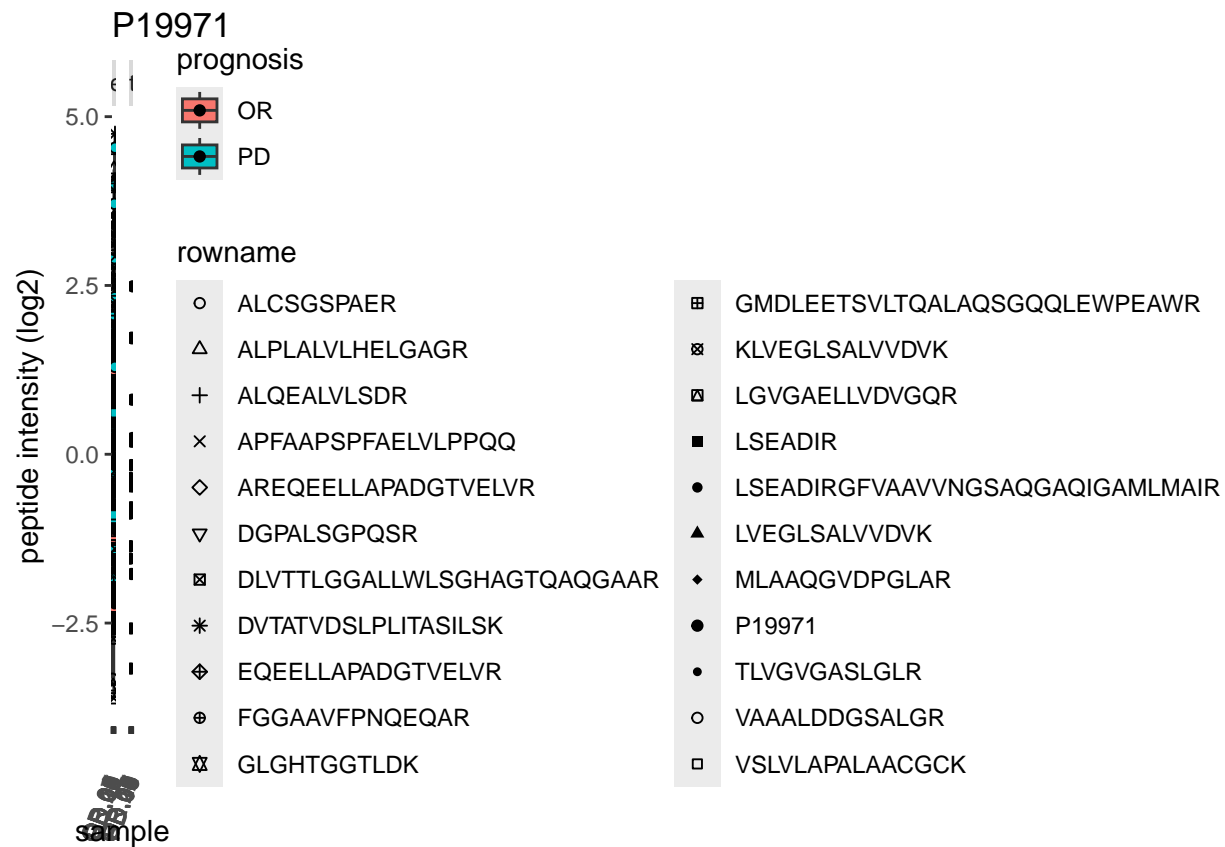


P07305

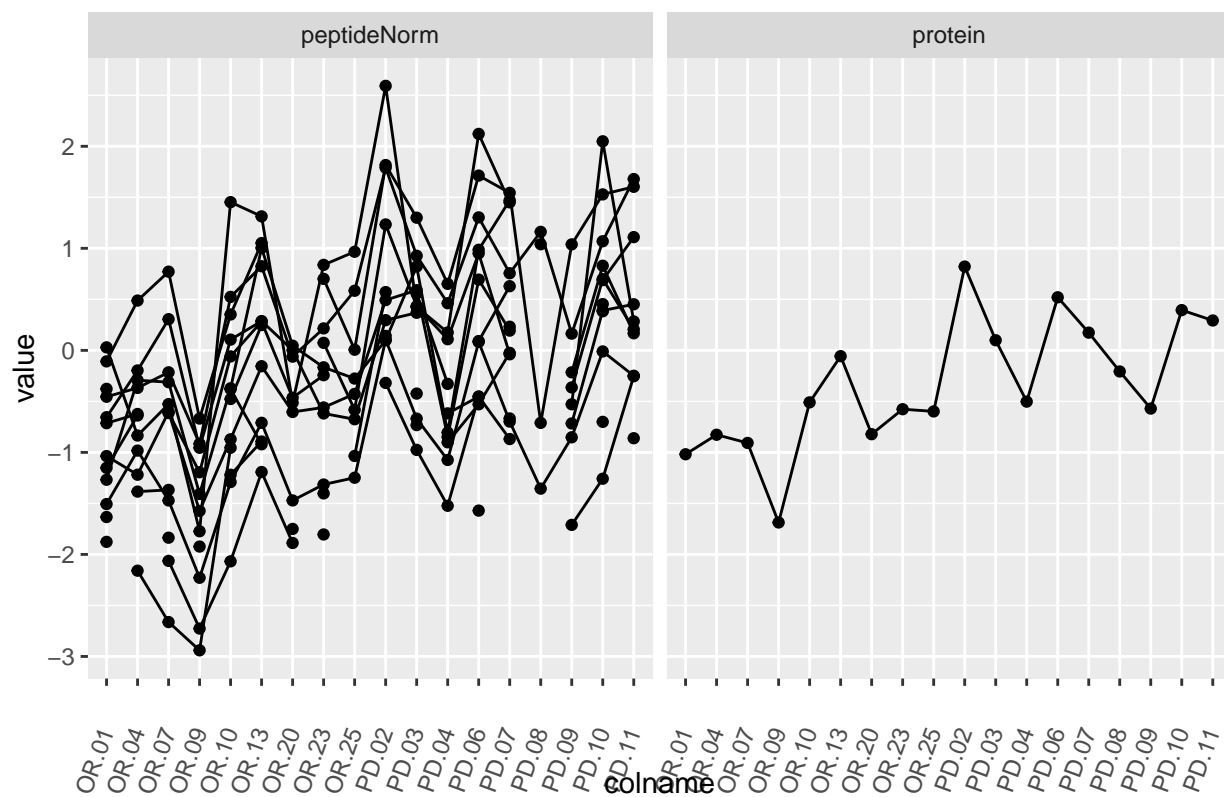


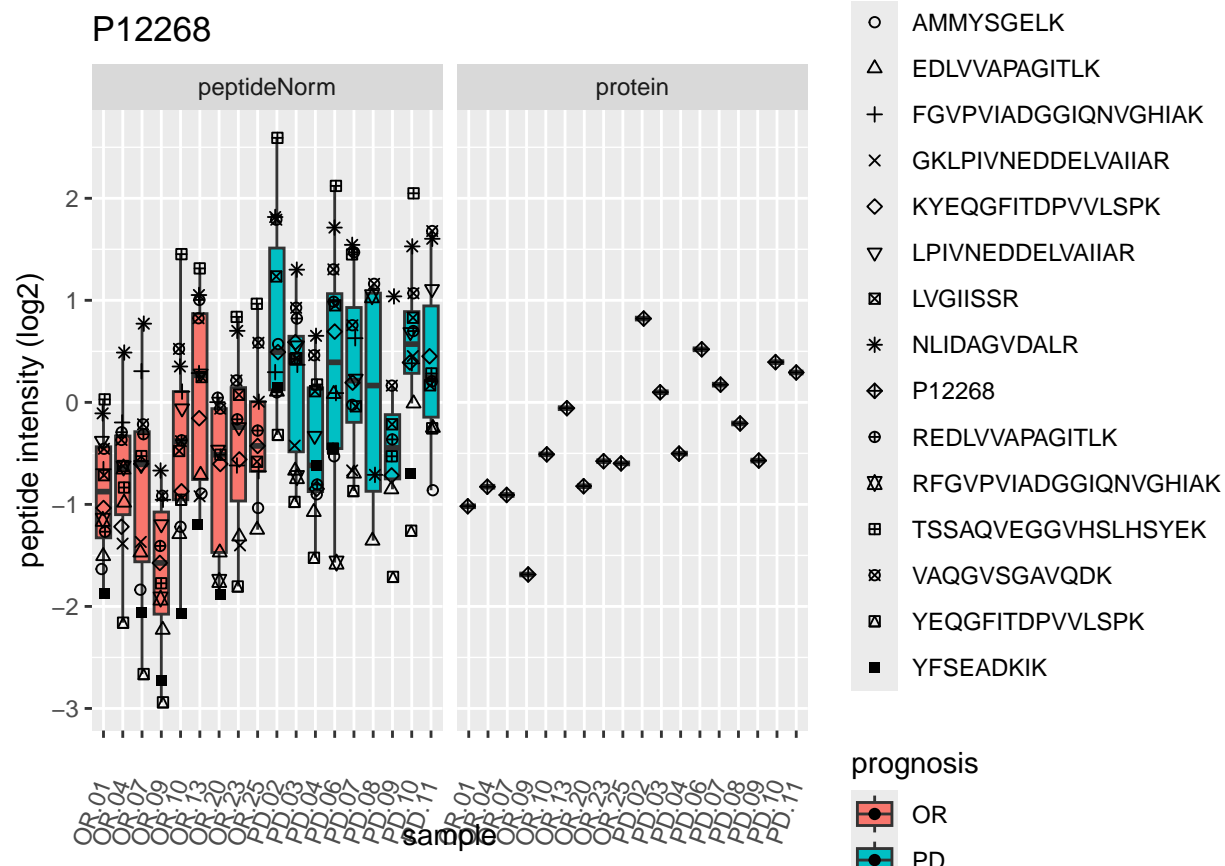
P19971



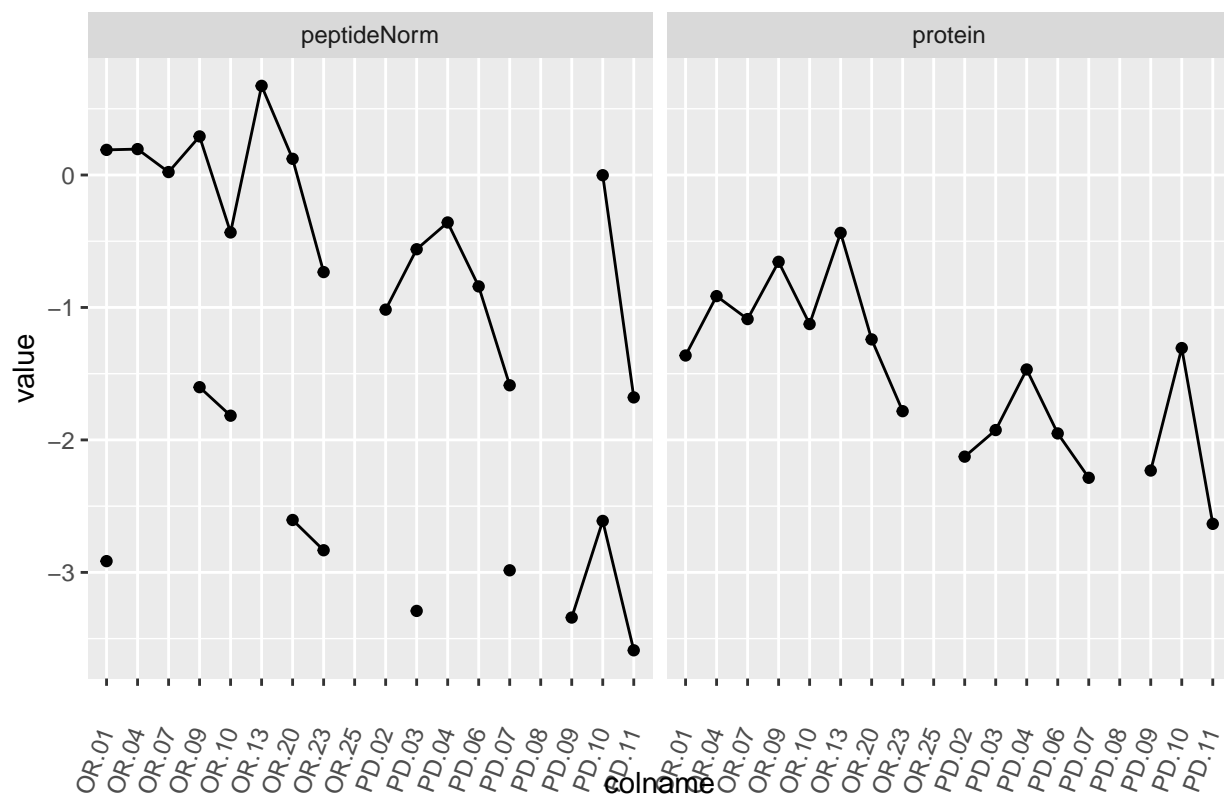


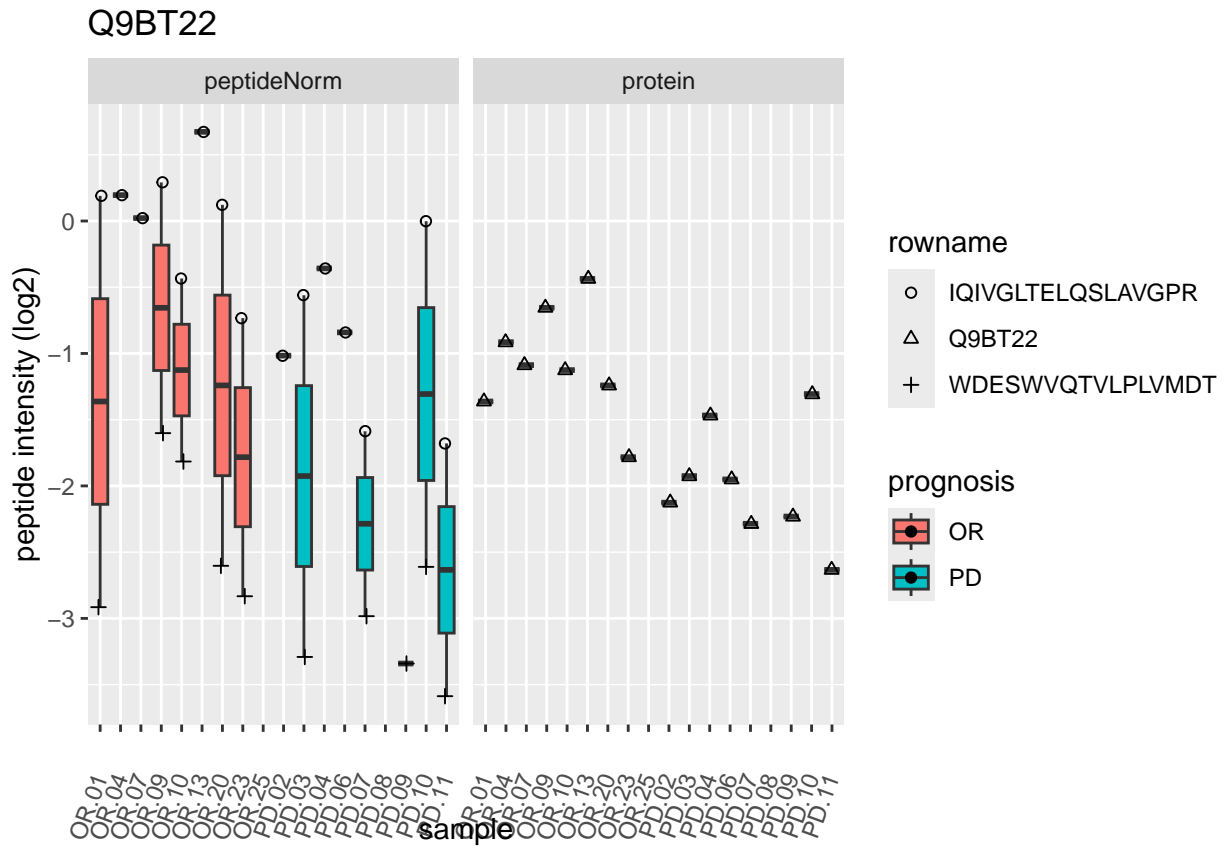
P12268





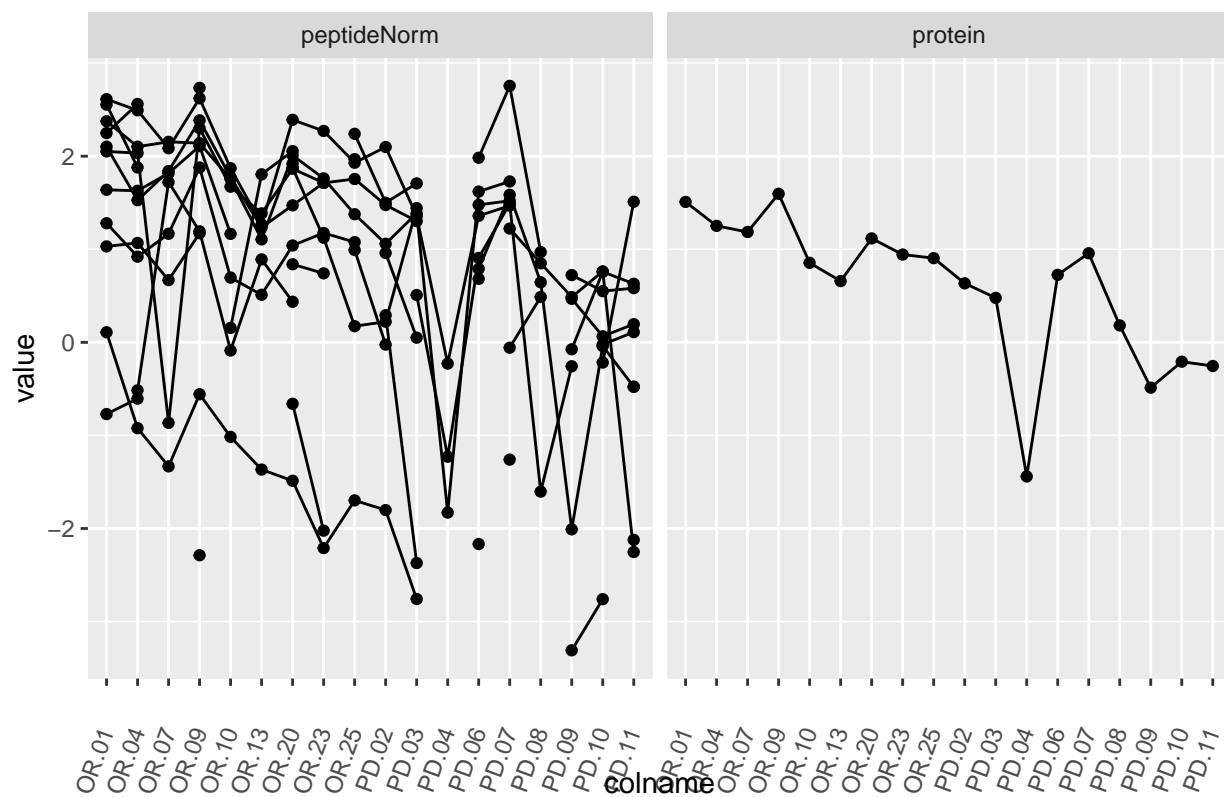
# Q9BT22

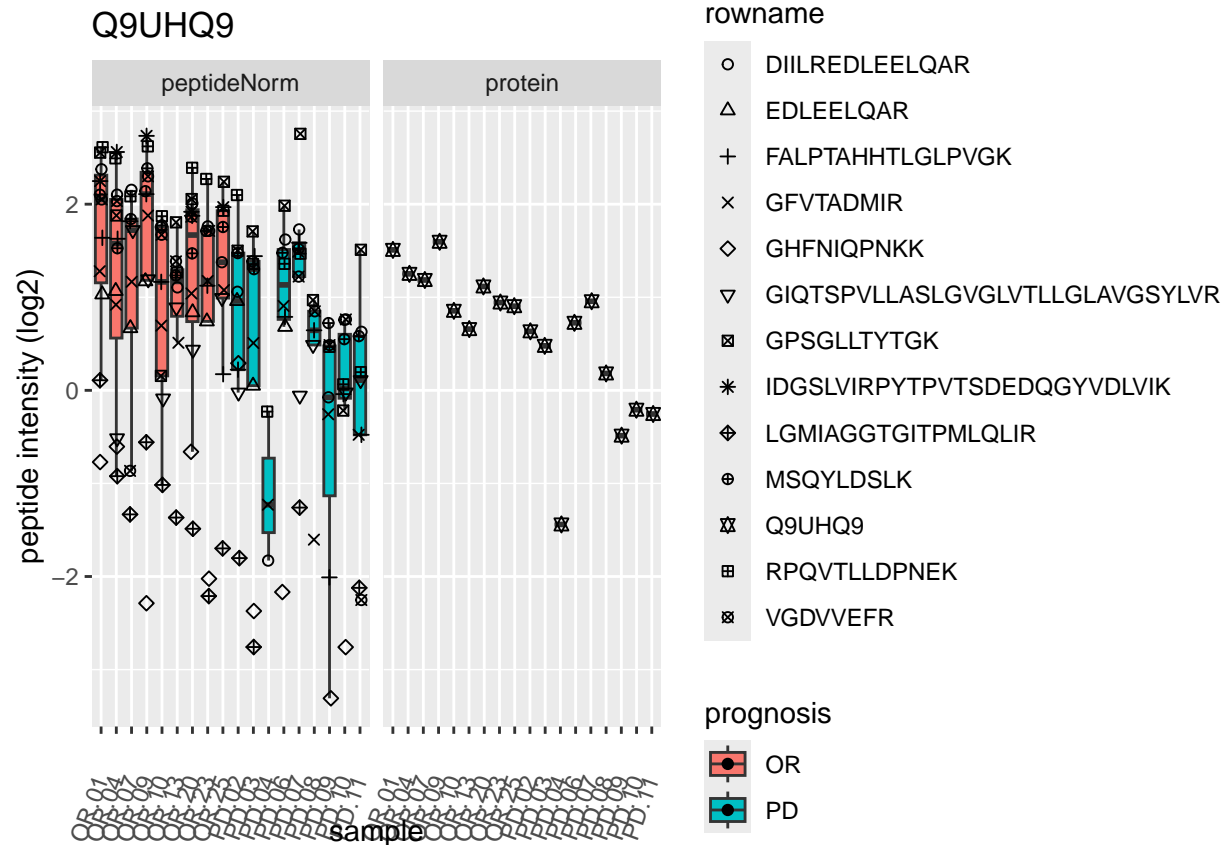




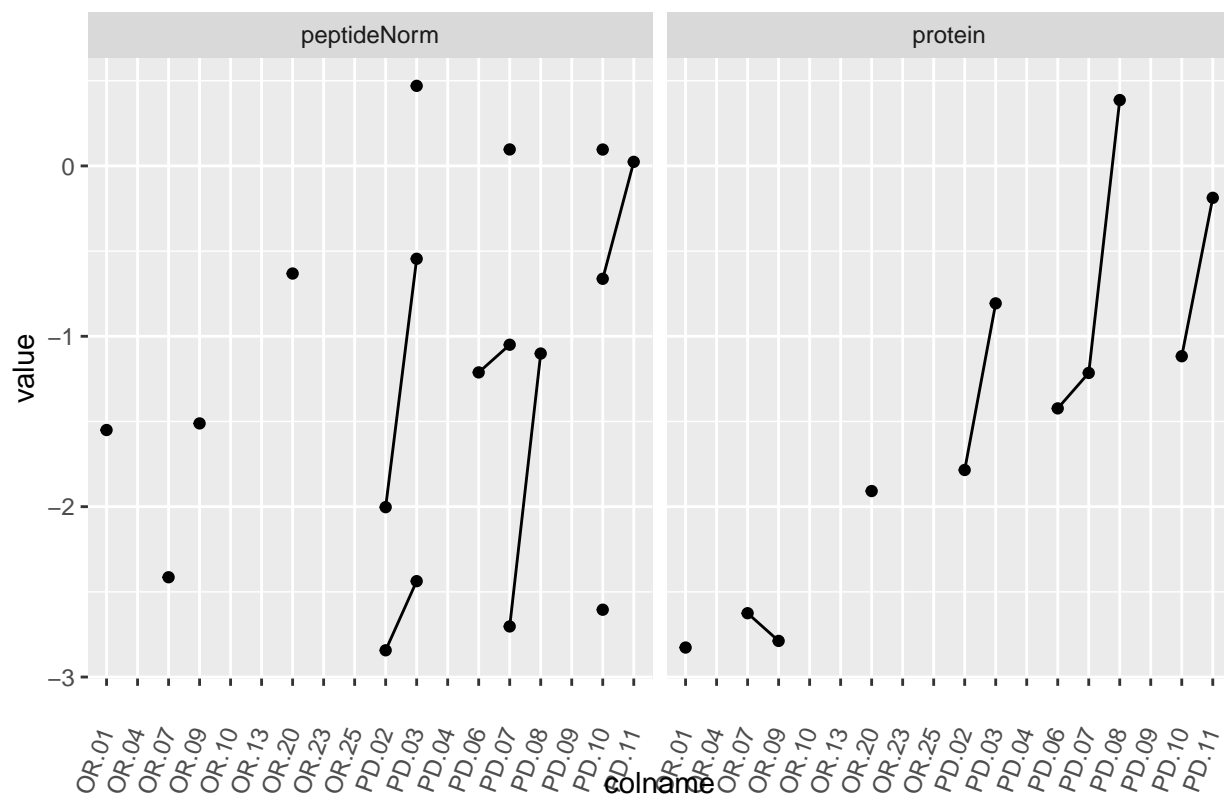


Q9UHQ9

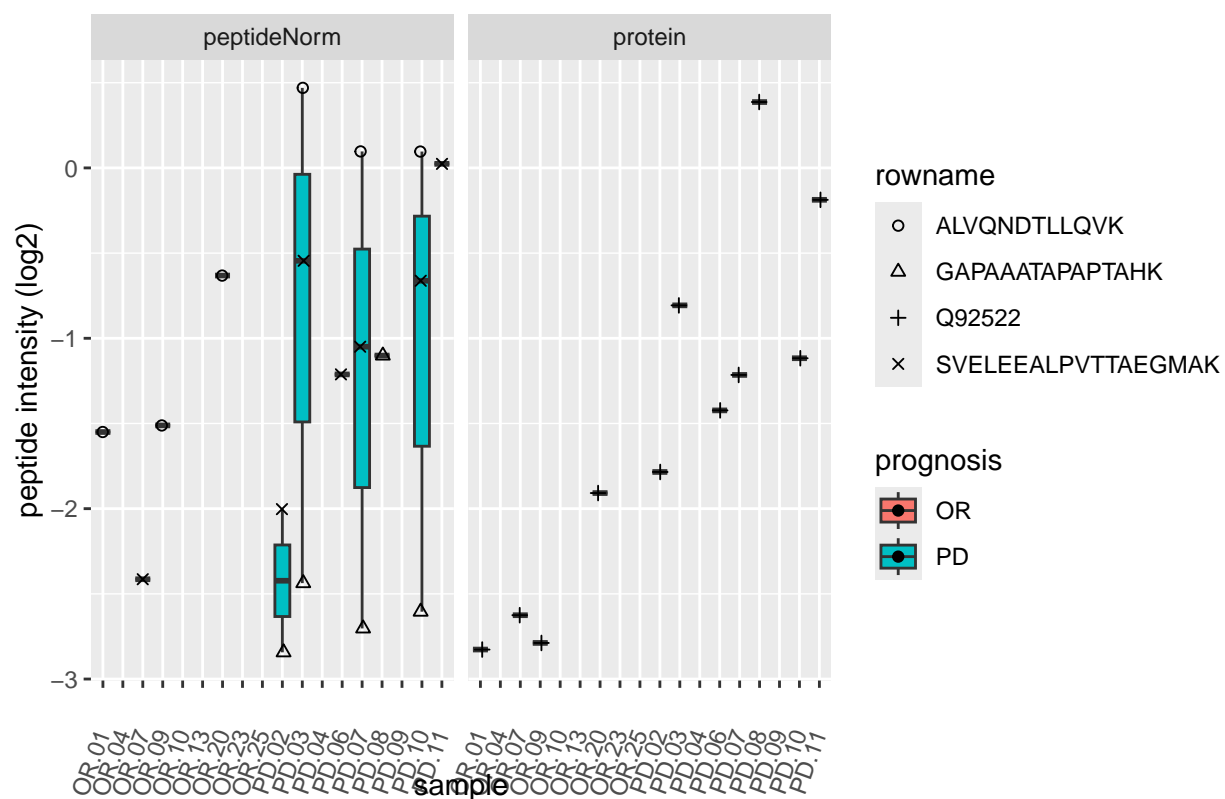




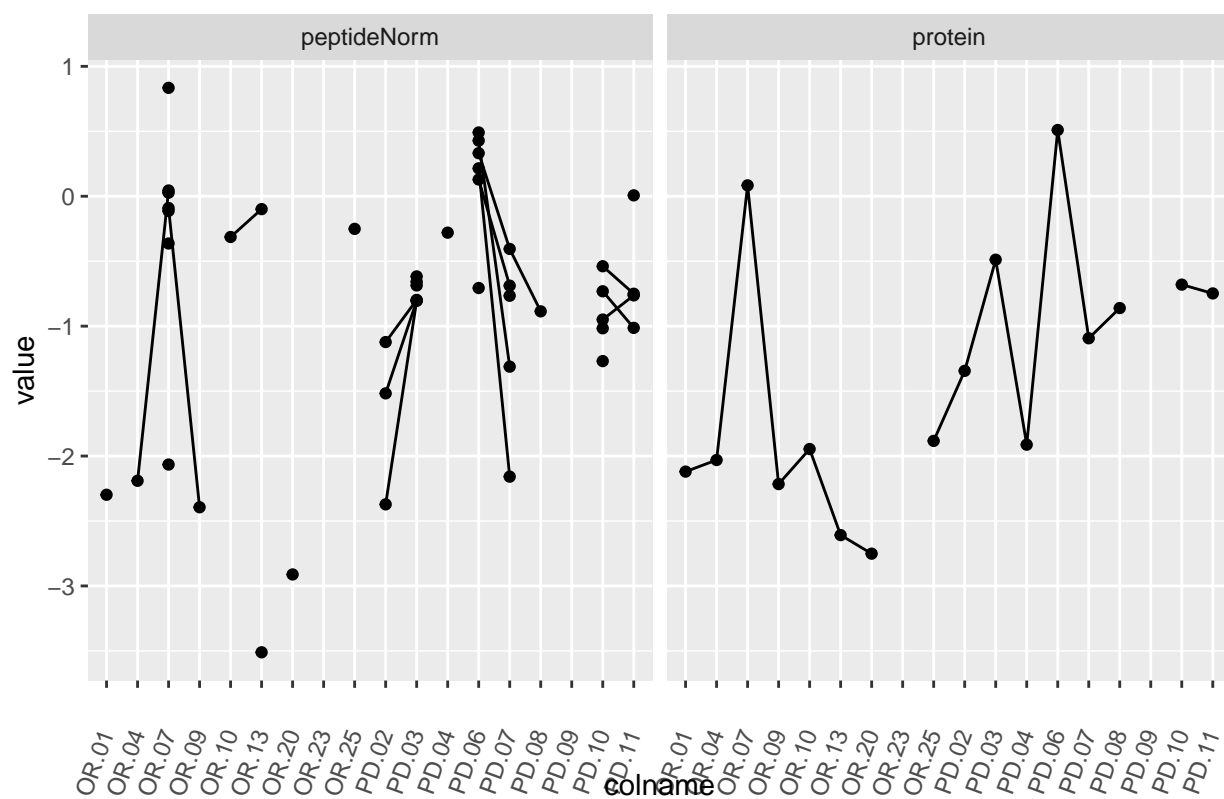
Q92522



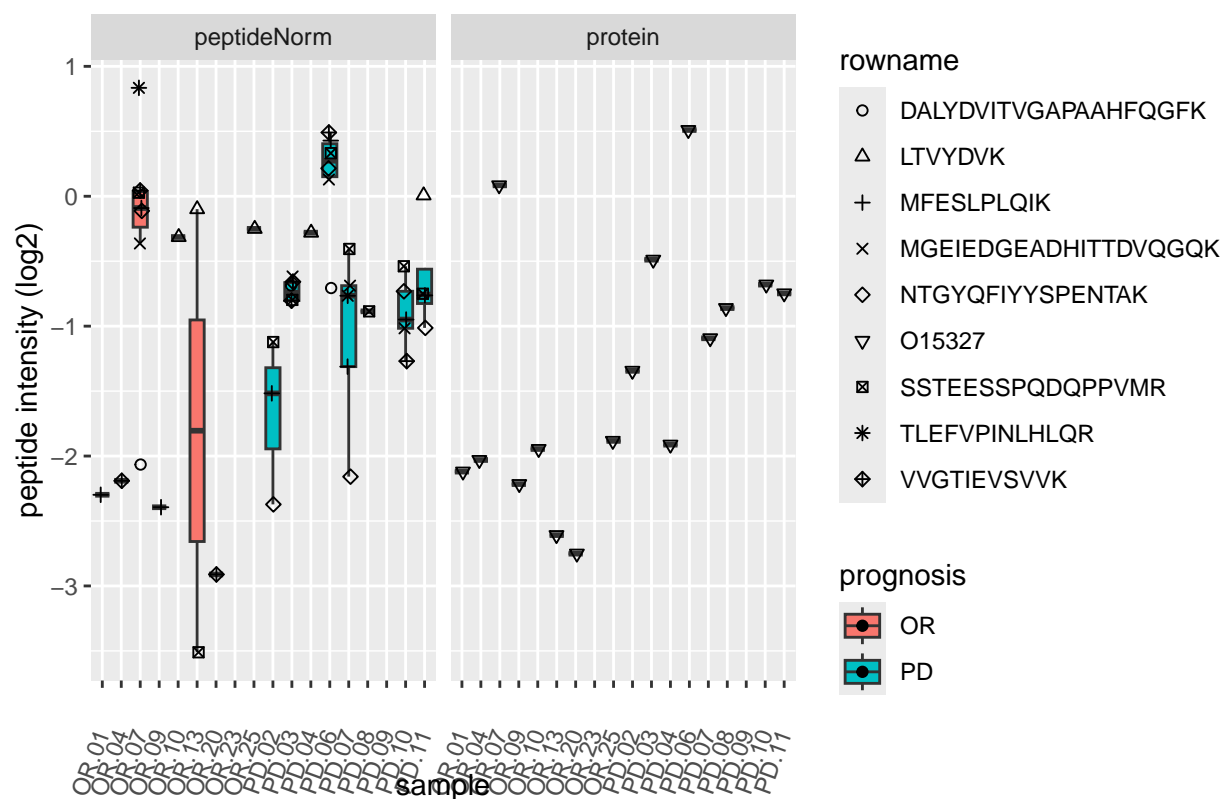
Q92522



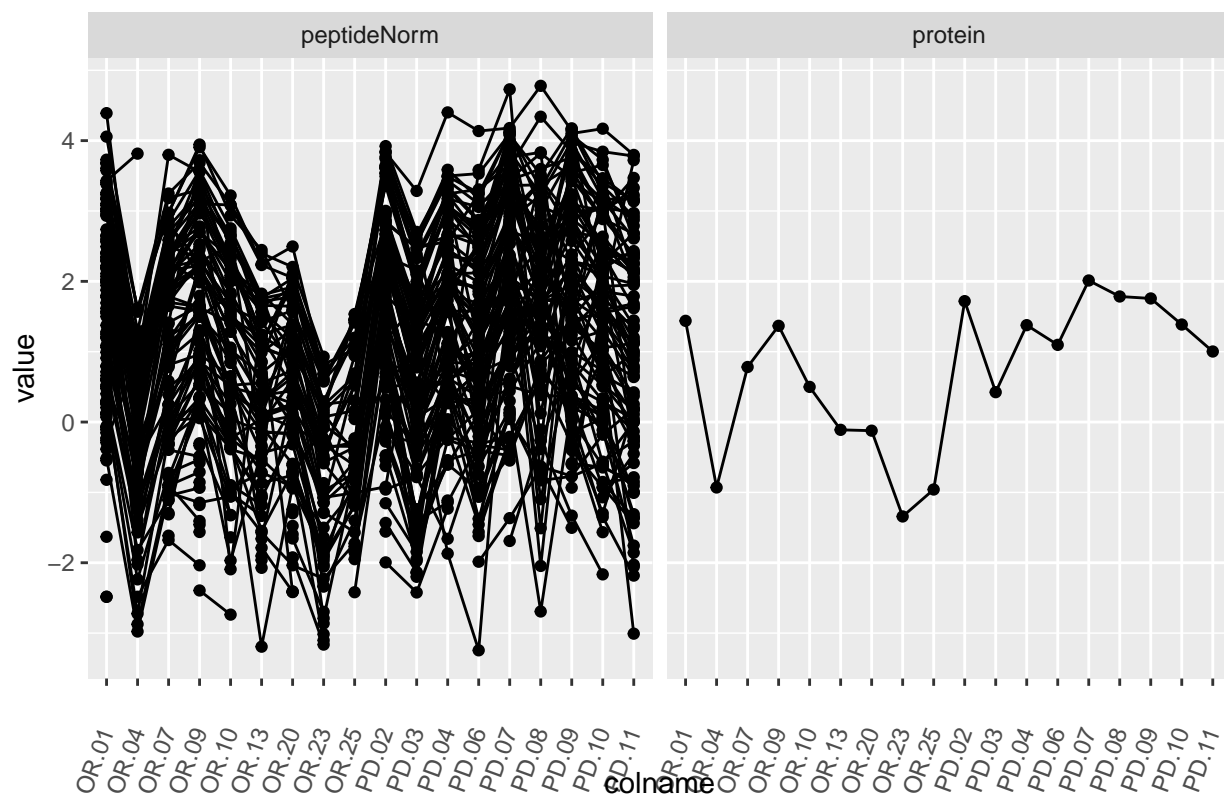
O15327



O15327



P21333



/K

TYGGHQVPGSPFK

IVTITWGGQNIGR

IVDPNVDEHSVMTYLSQFPK

PFPLEAVAPTKPSK

PSVQPPLR

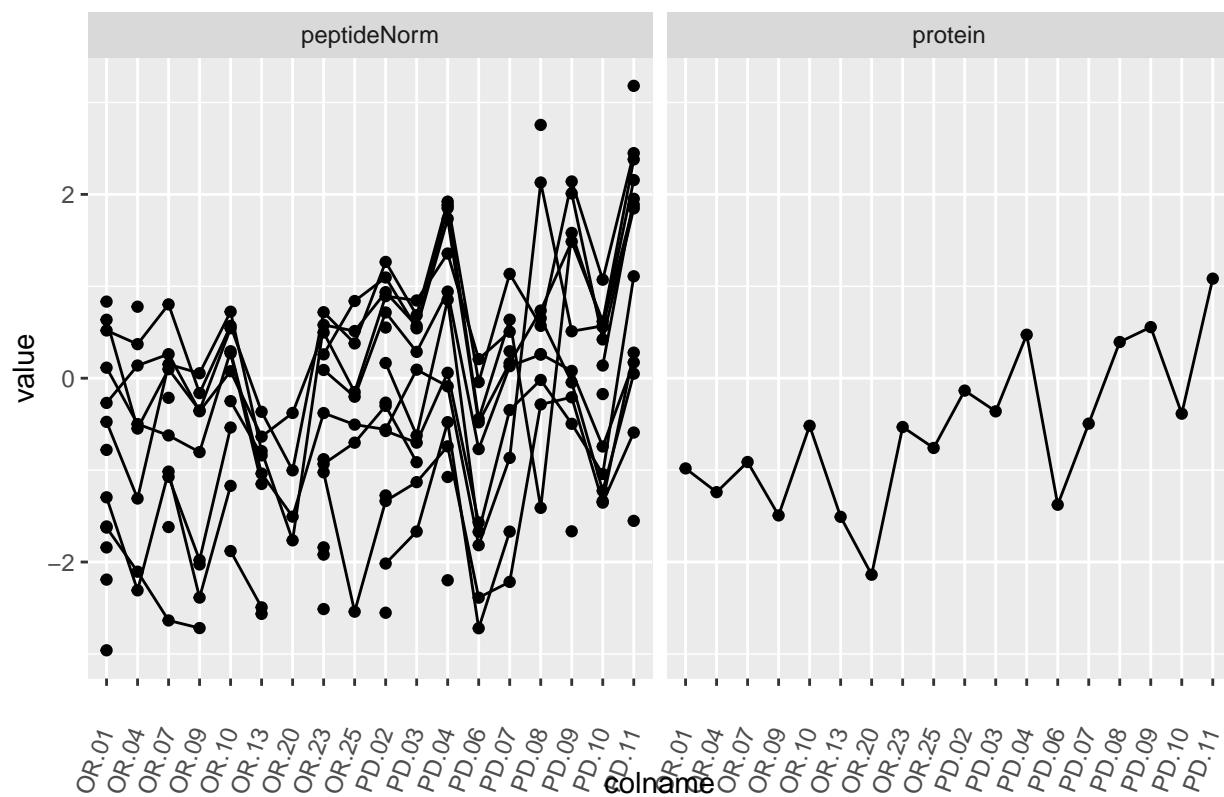
DAR

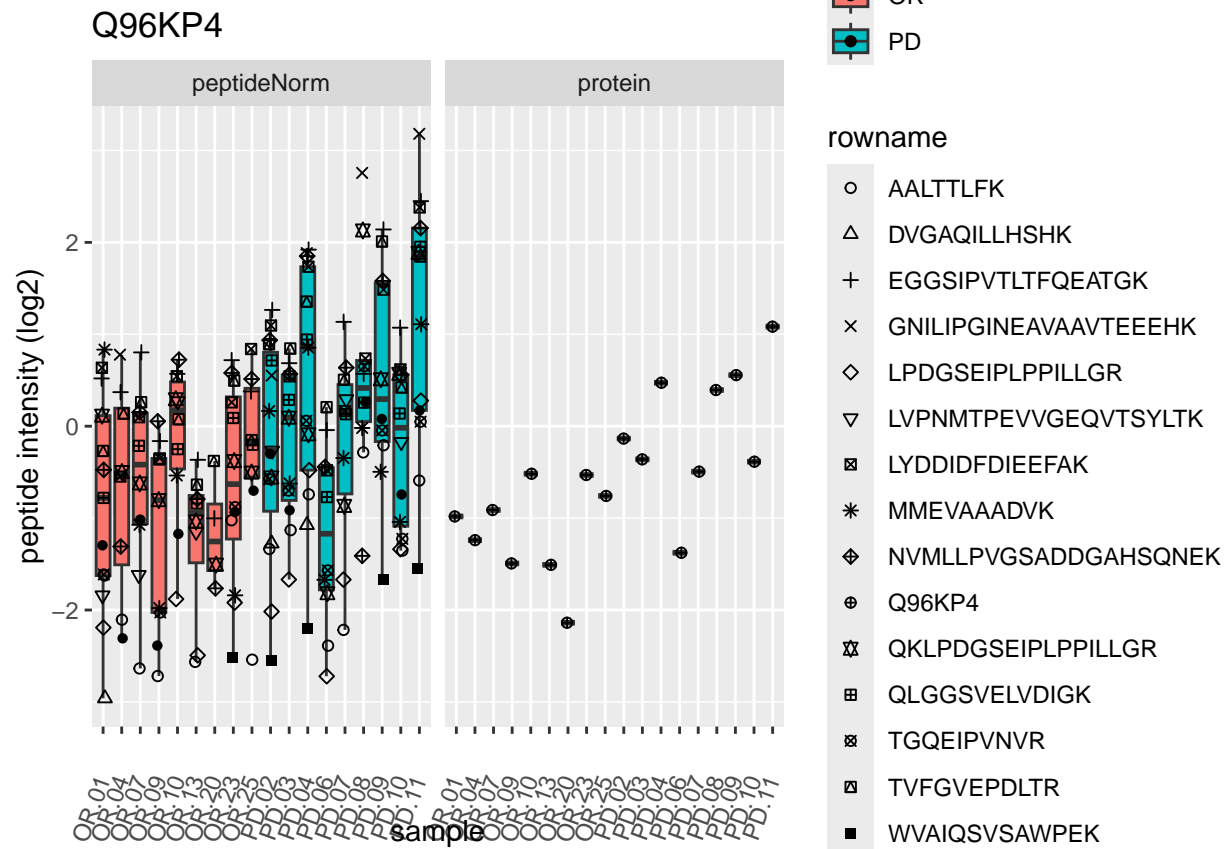
) GAGSYTIMVLFADQATPTSPIR  
\* GAGTGGLGLAVEGPSEAK  
+ GKLDVQFSGLTK  
, GLVEPVDVVDNADGTQTVNYVPSR  
- GQHVPGPSPFQFTVGPLGEGGAHK  
• GTVEPQLEAR  
/ HTAMVSWGGVSIPNSPFR  
0 IANLQTDLSGDLR  
1 IECDDKGDGSCDVR  
2 IPEISIQDMTAQVTSPSGK  
3 IVGPSGAAPVCK  
4 KGEITGEVR  
5 KTHIQDNHDGTYTVAYVPDVTGR  
6 LDVQFSGLTK  
7 LIALLEVLSQKK  
8 LLGWIQNKLPQLPITNFSR  
9 LQVEPAVDTSQVQCYGPGIEGQGVFR

< LVSNHSLHETSSVFVDSLTK  
= LYSVSYLLK  
> MDCQECPEGYR  
? NDNDTFTVK  
@ NGHVGISFVPK  
A NGQHVASSPIPVVISQSEIGDASI  
B P21333  
C PGAPLRPK  
D RAEFTVETR  
E RAPSVANVGSHCDLSLK  
F RLTVSSLQESGLK  
G SADFFVEAIGDDVGTGFSVEGF  
H SAGQGEVLVYVEDPAGHQEEAK  
I SPFEVYVDK  
J SPFSVAVSPSLDLSK  
K SPYTVTVGQACNPSACR  
L TFSVWYVPEVTGTHK

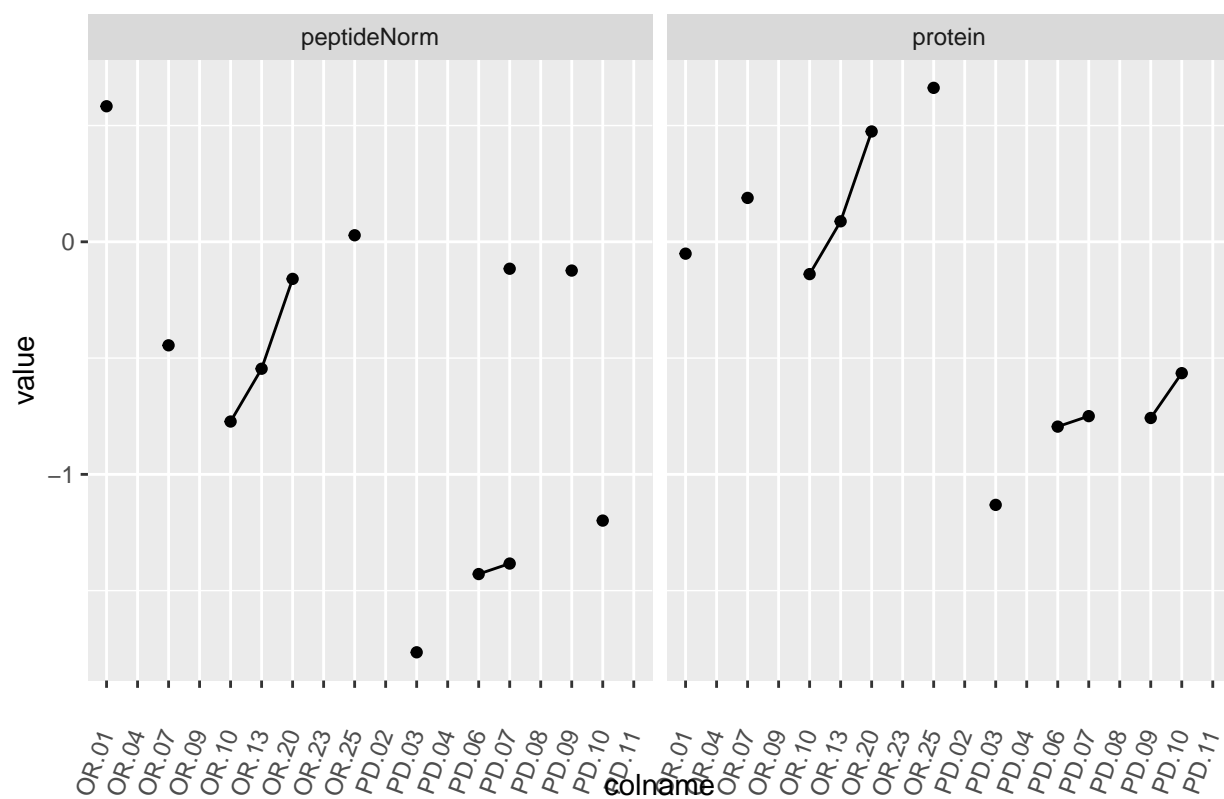


Q96KP4

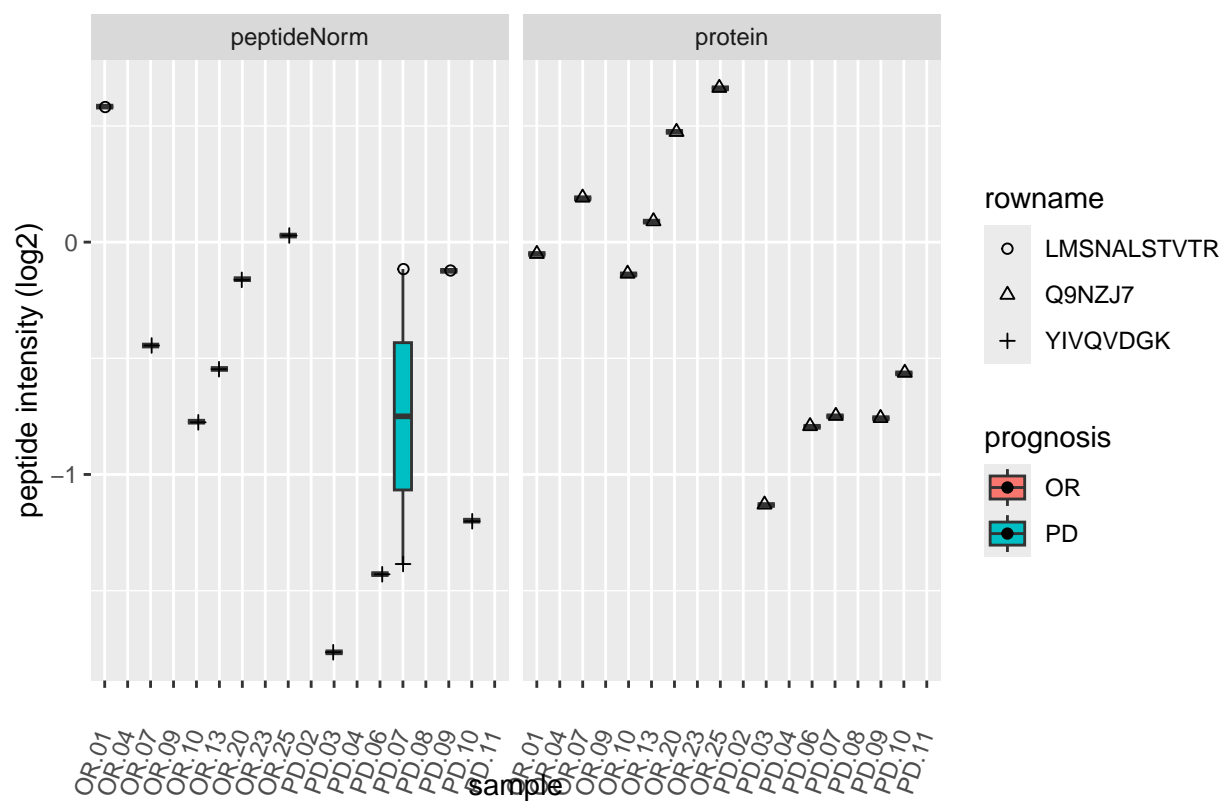




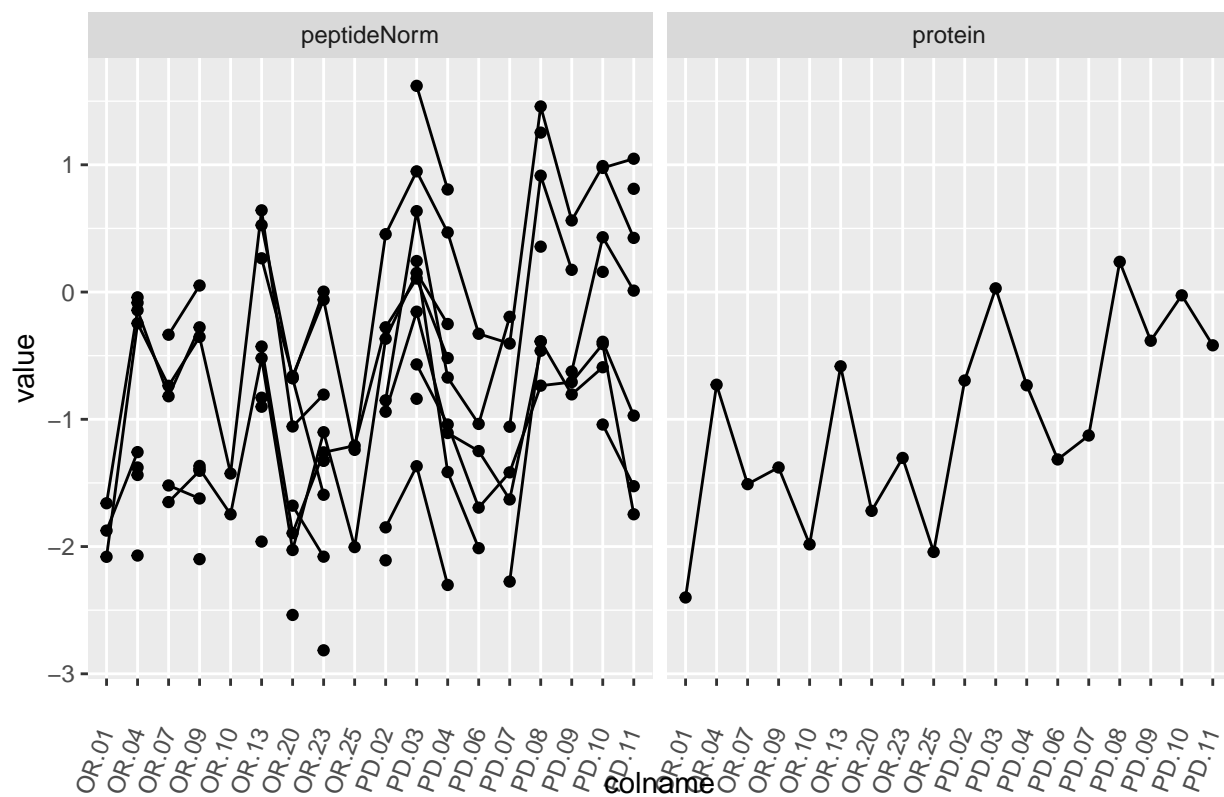
Q9NZJ7

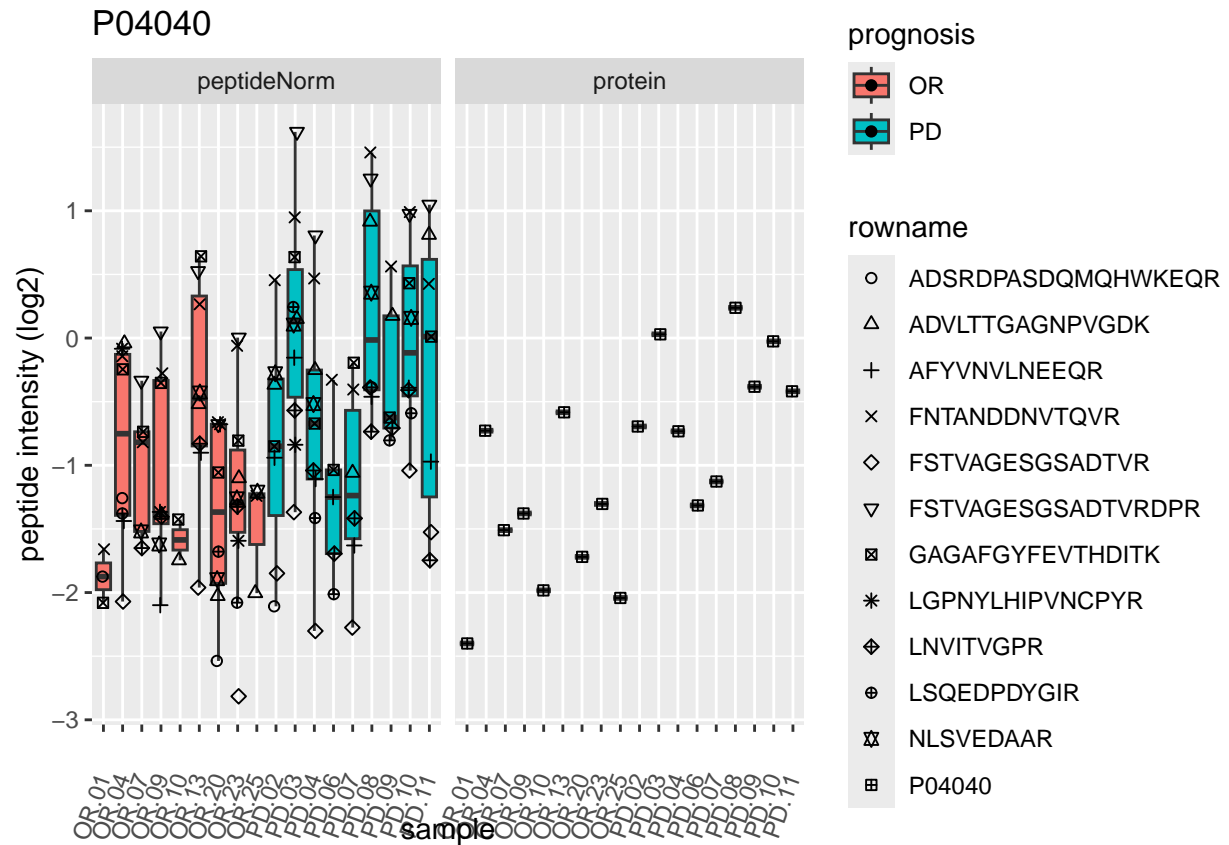


Q9NZJ7

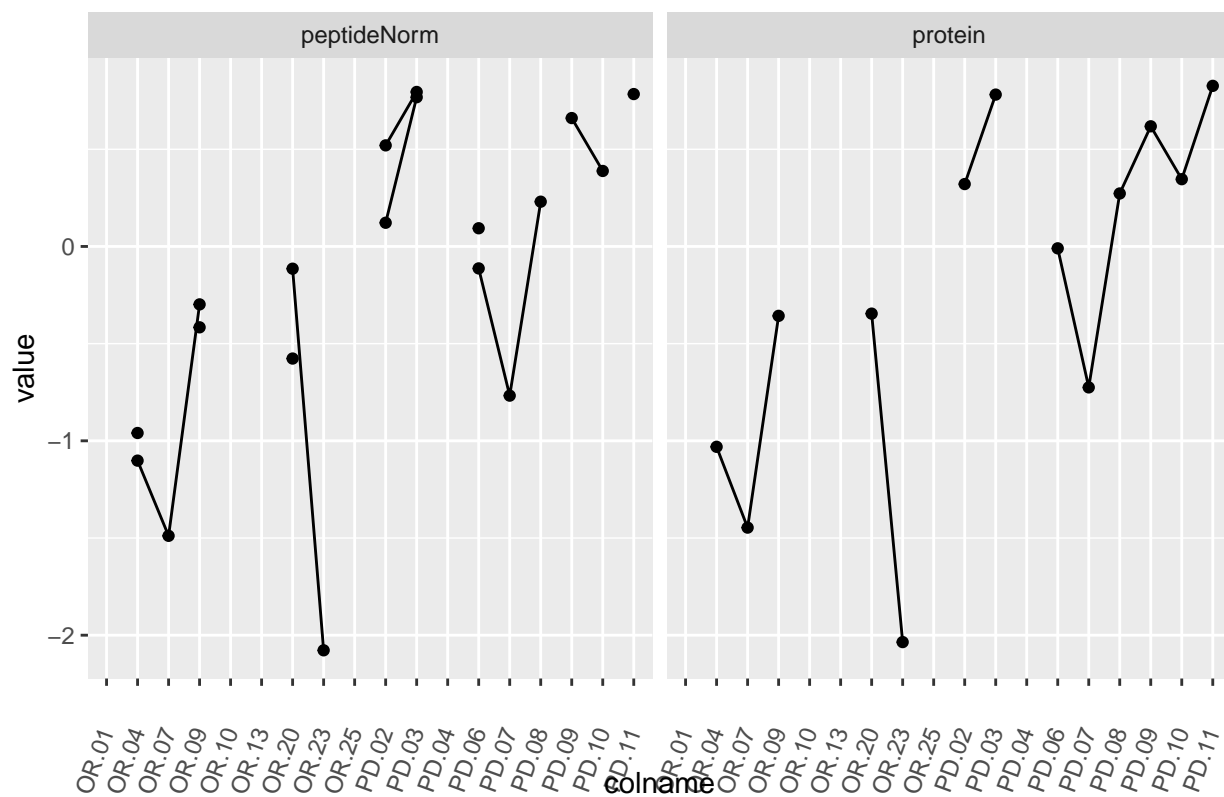


P04040

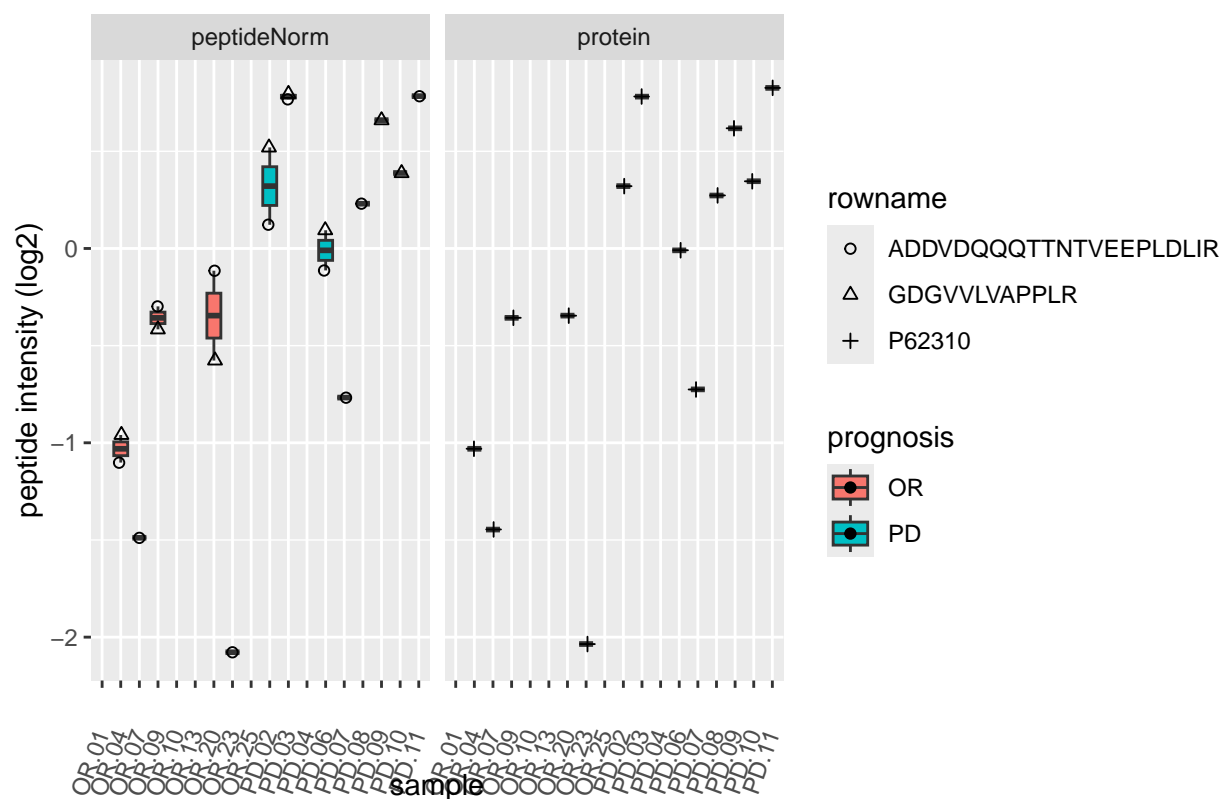




P62310

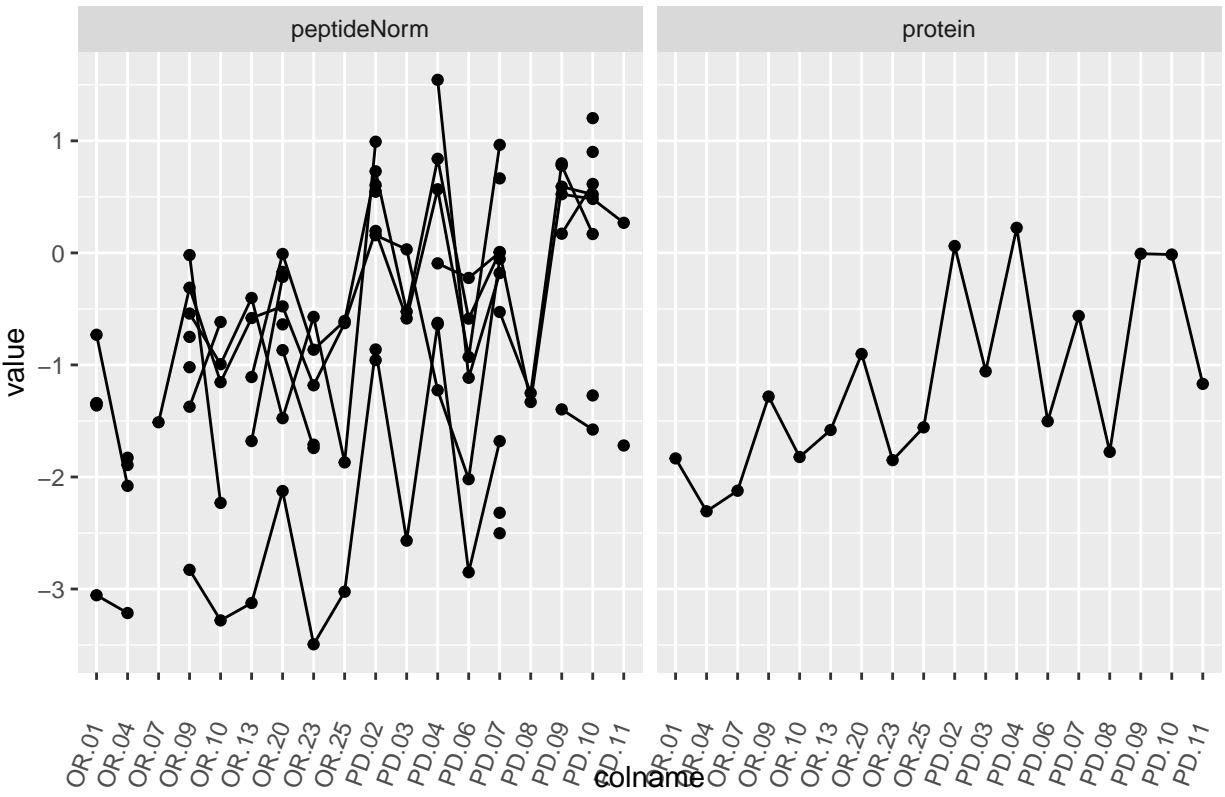


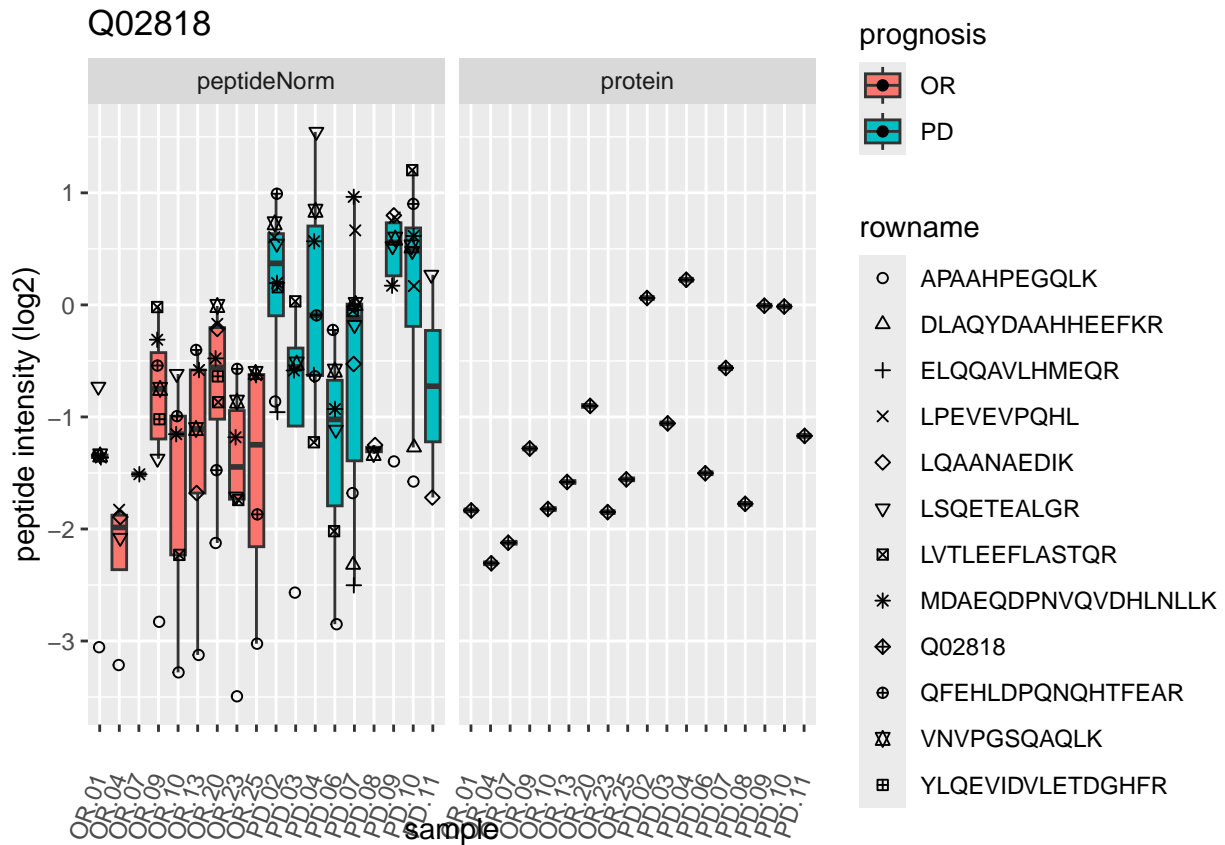
P62310



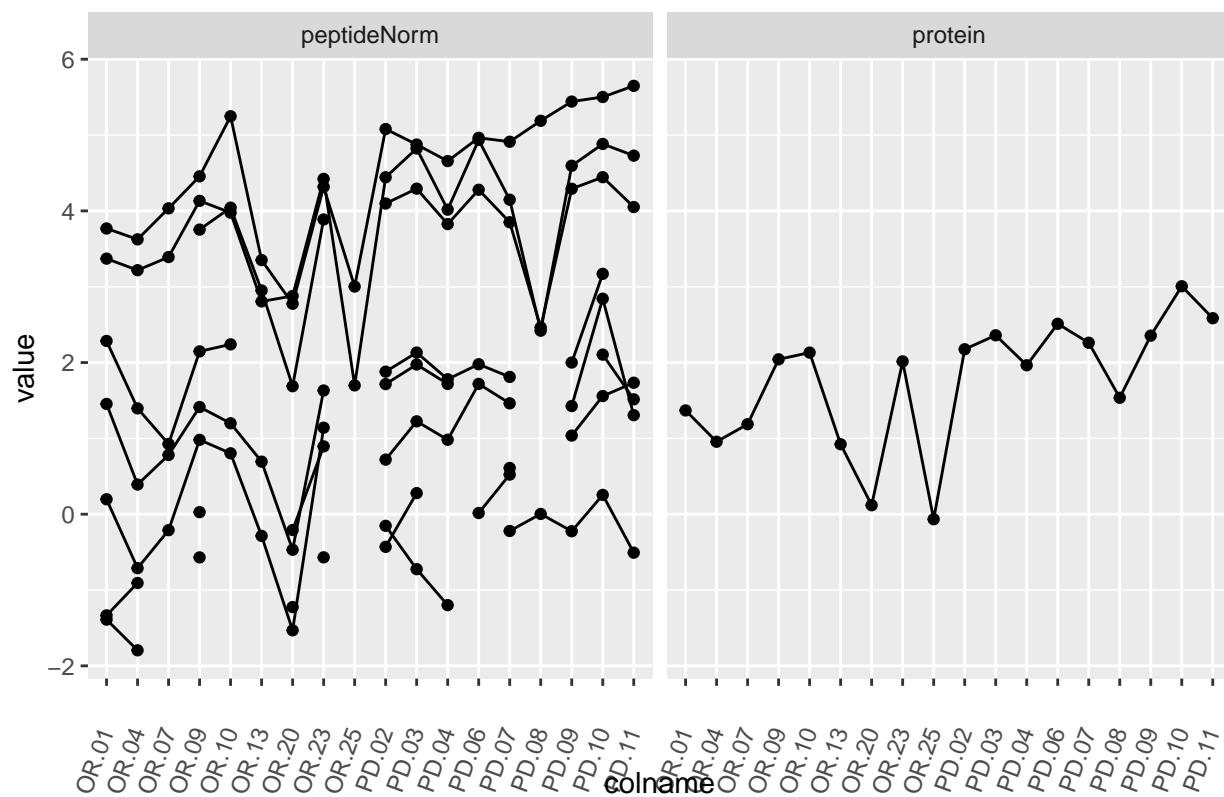


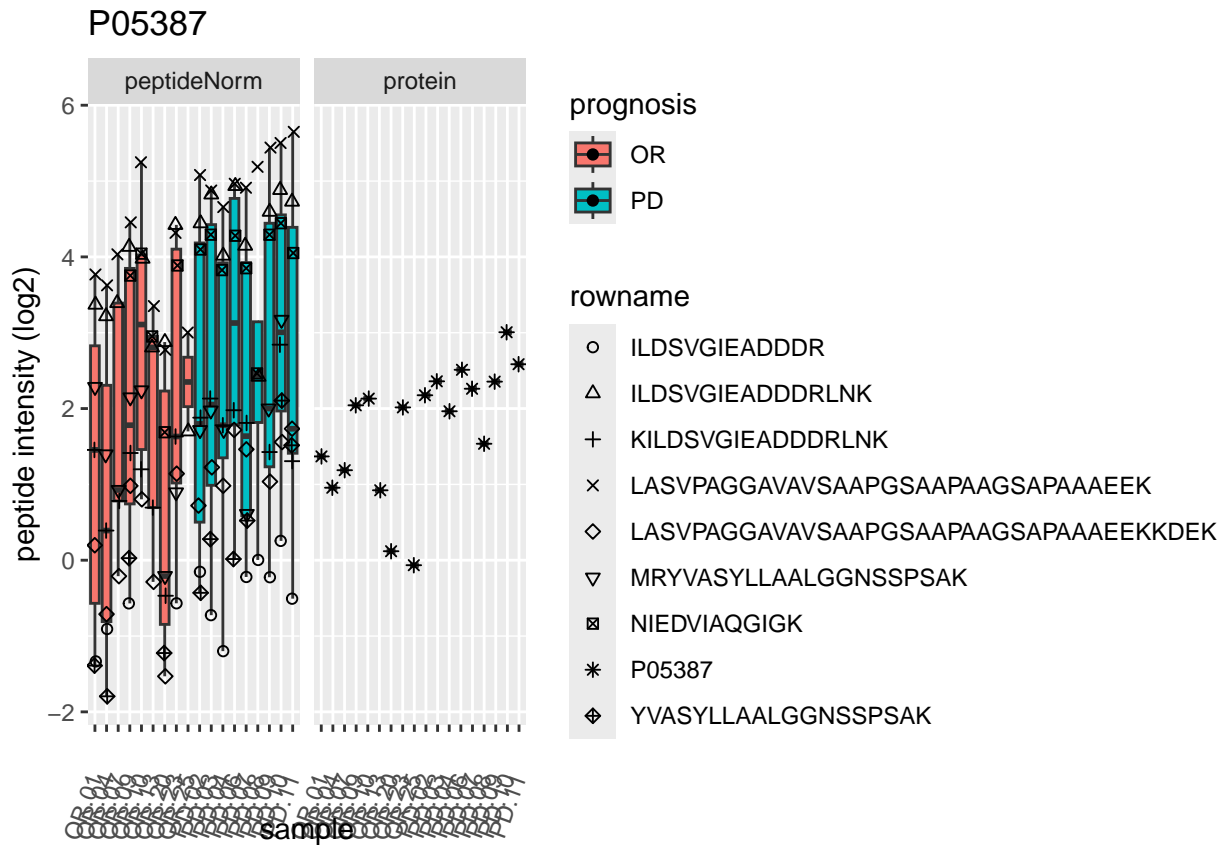
Q02818



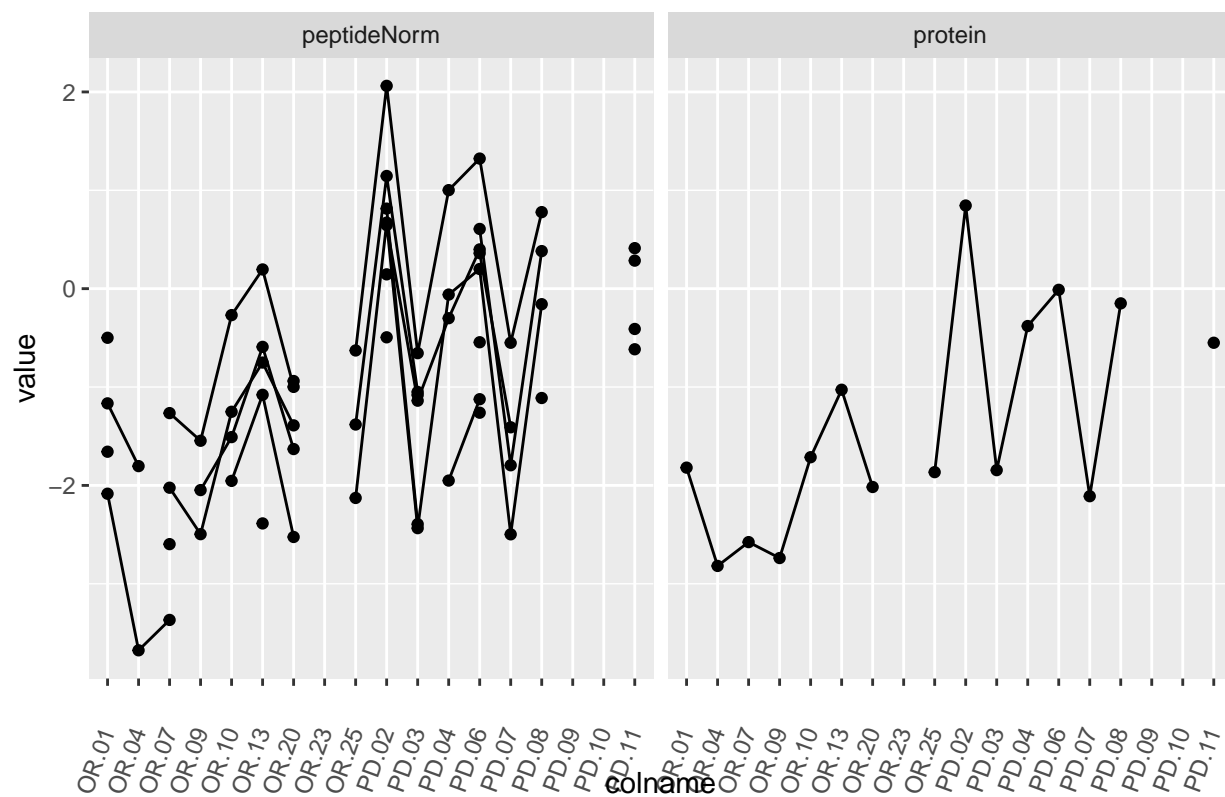


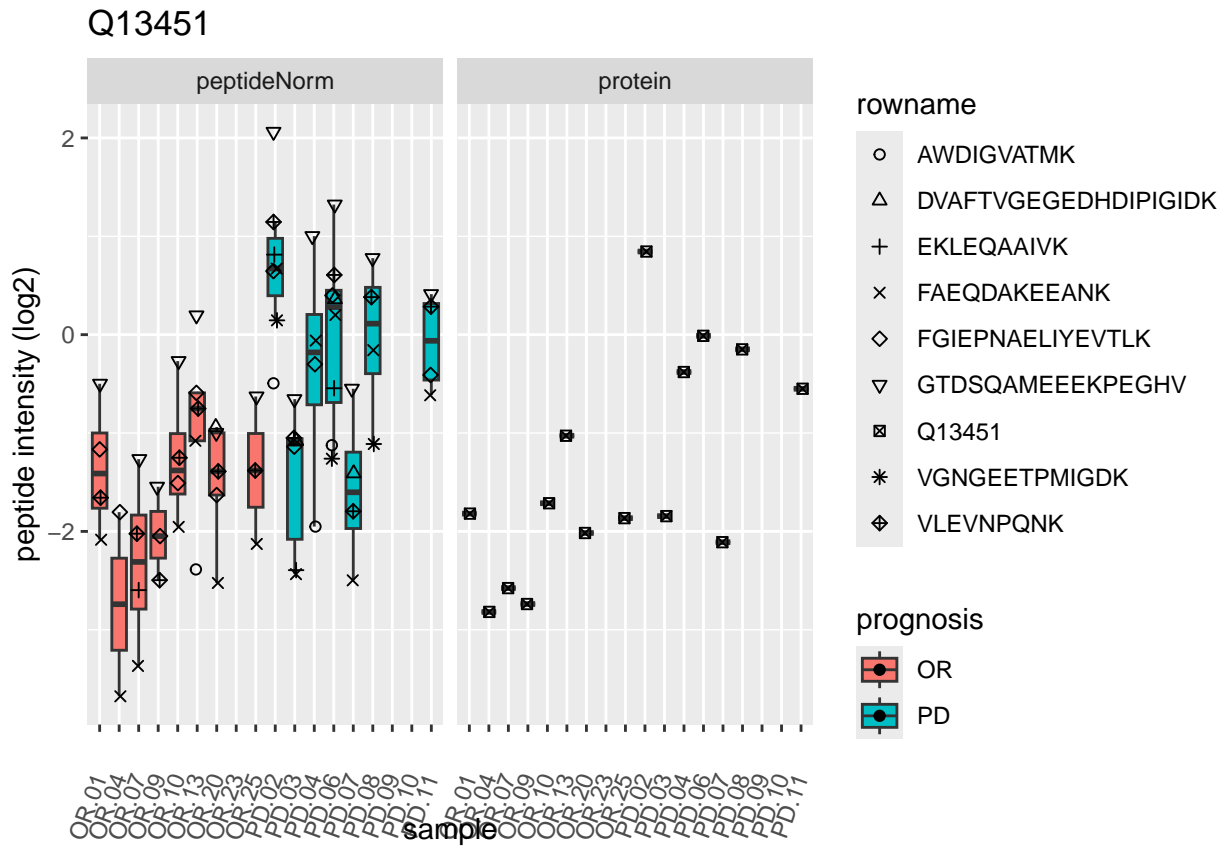
P05387



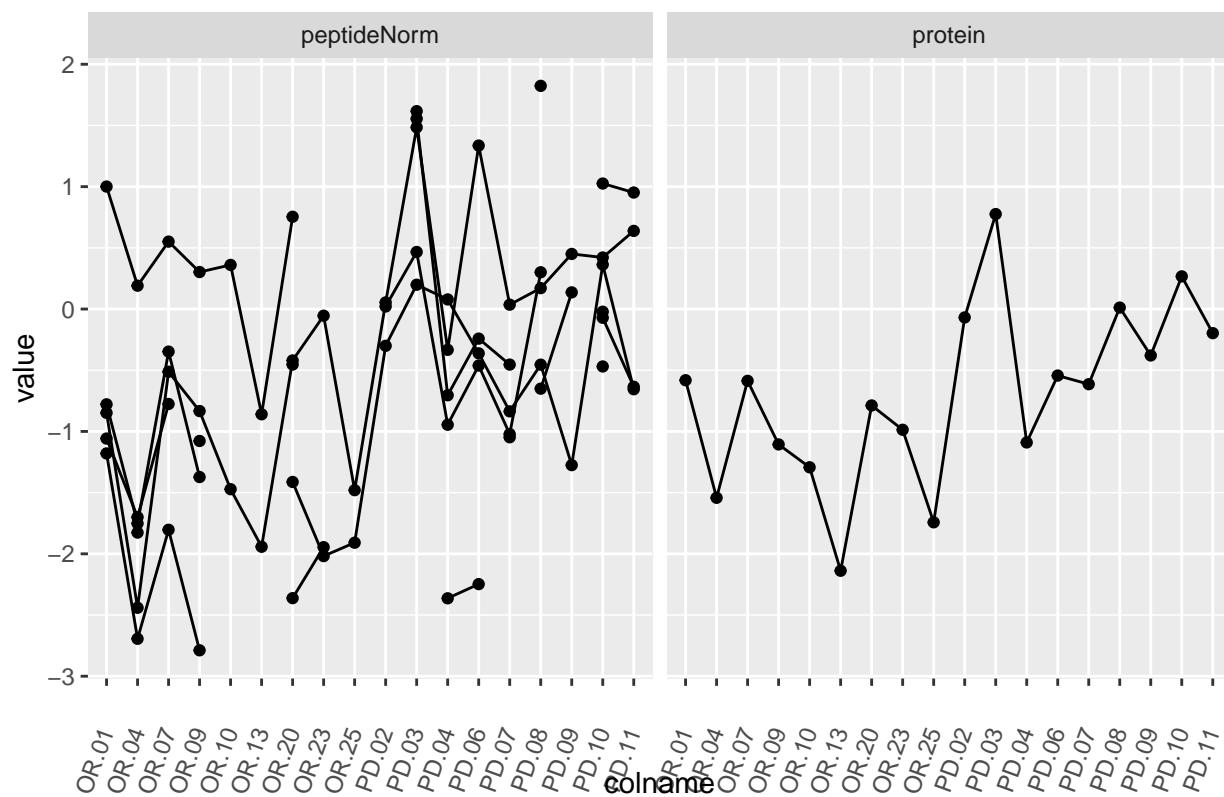


Q13451

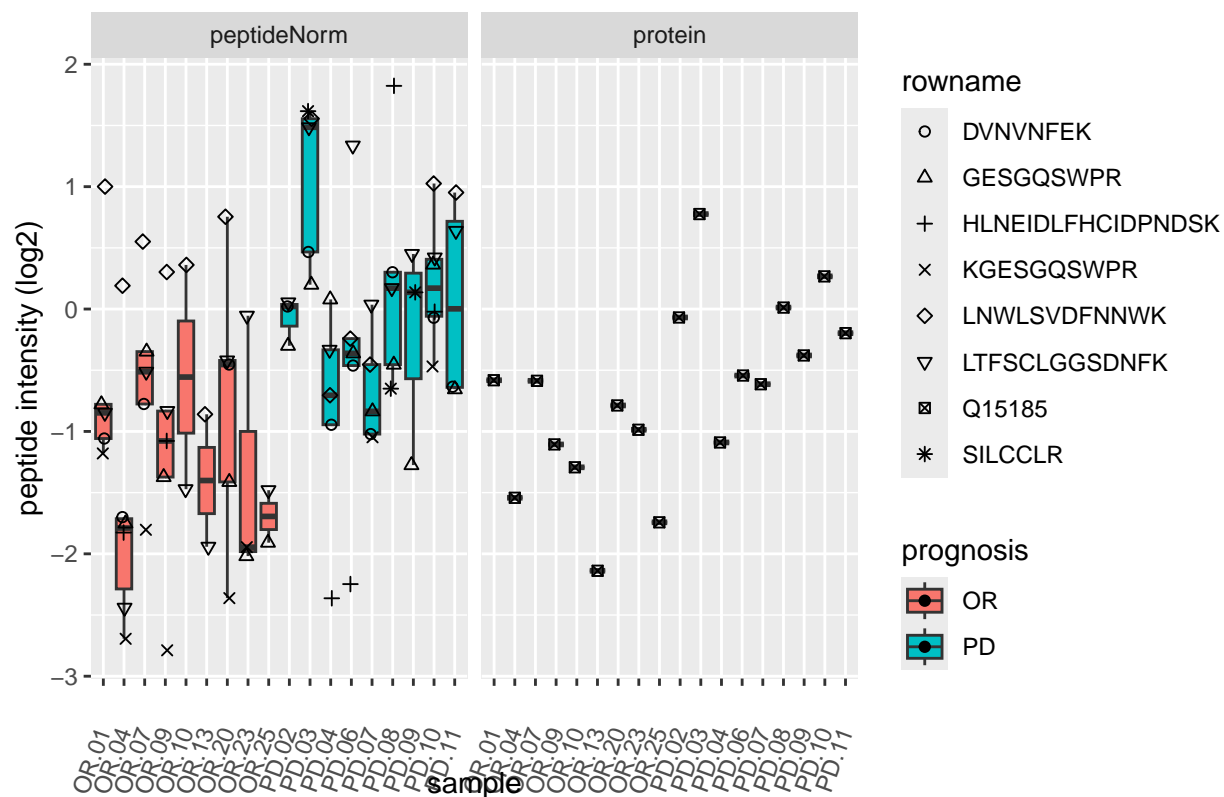




Q15185

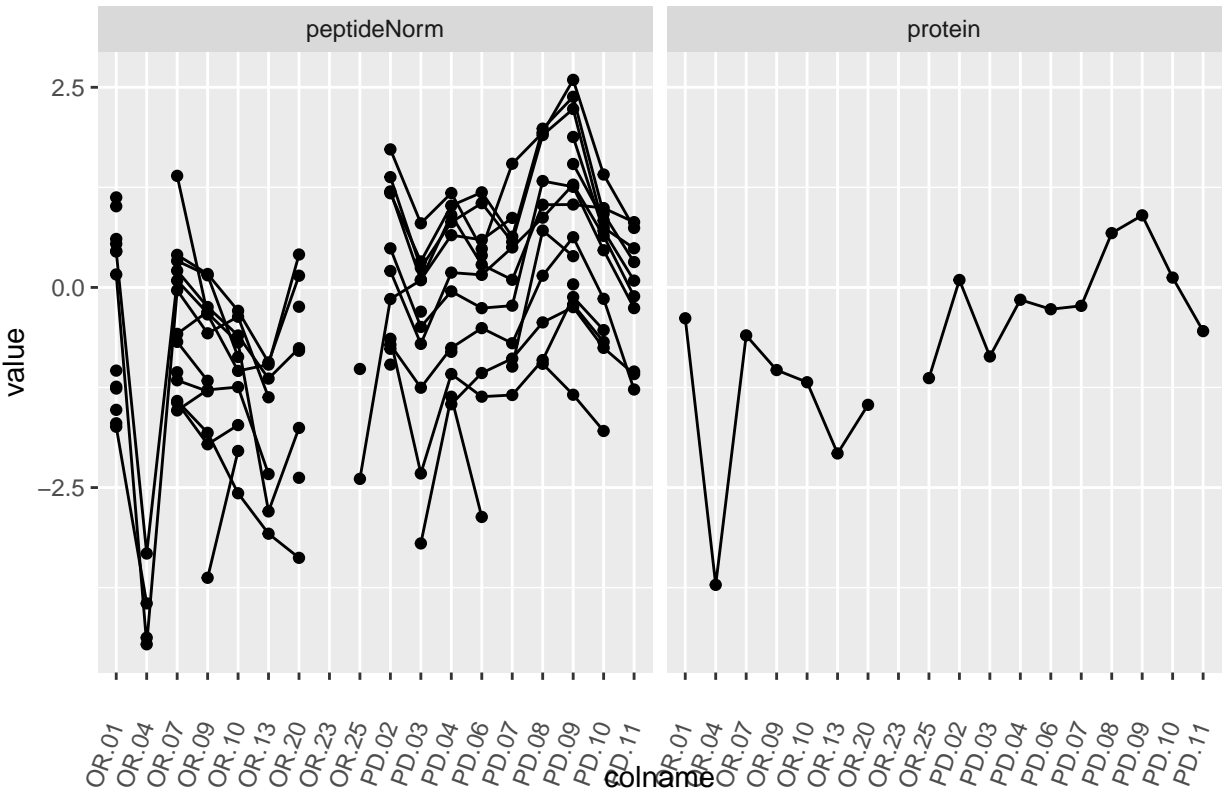


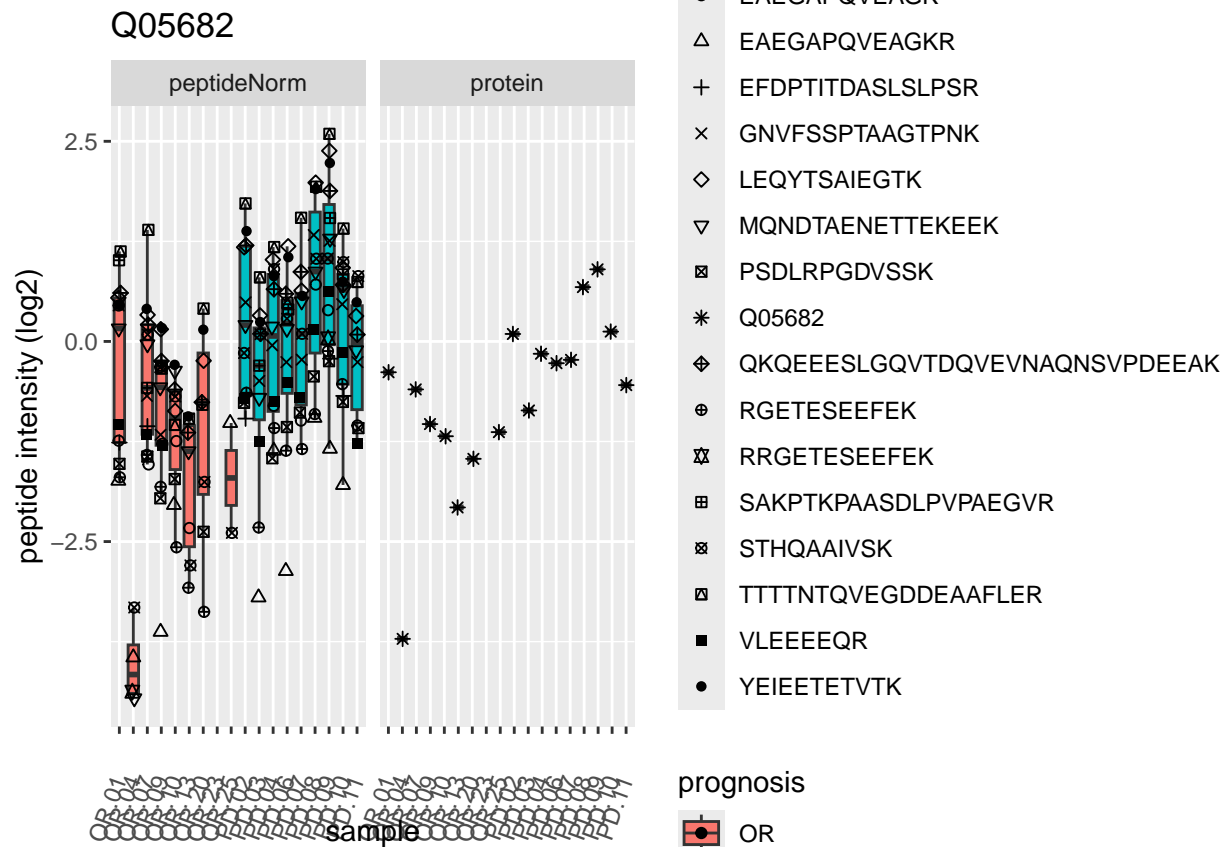
Q15185



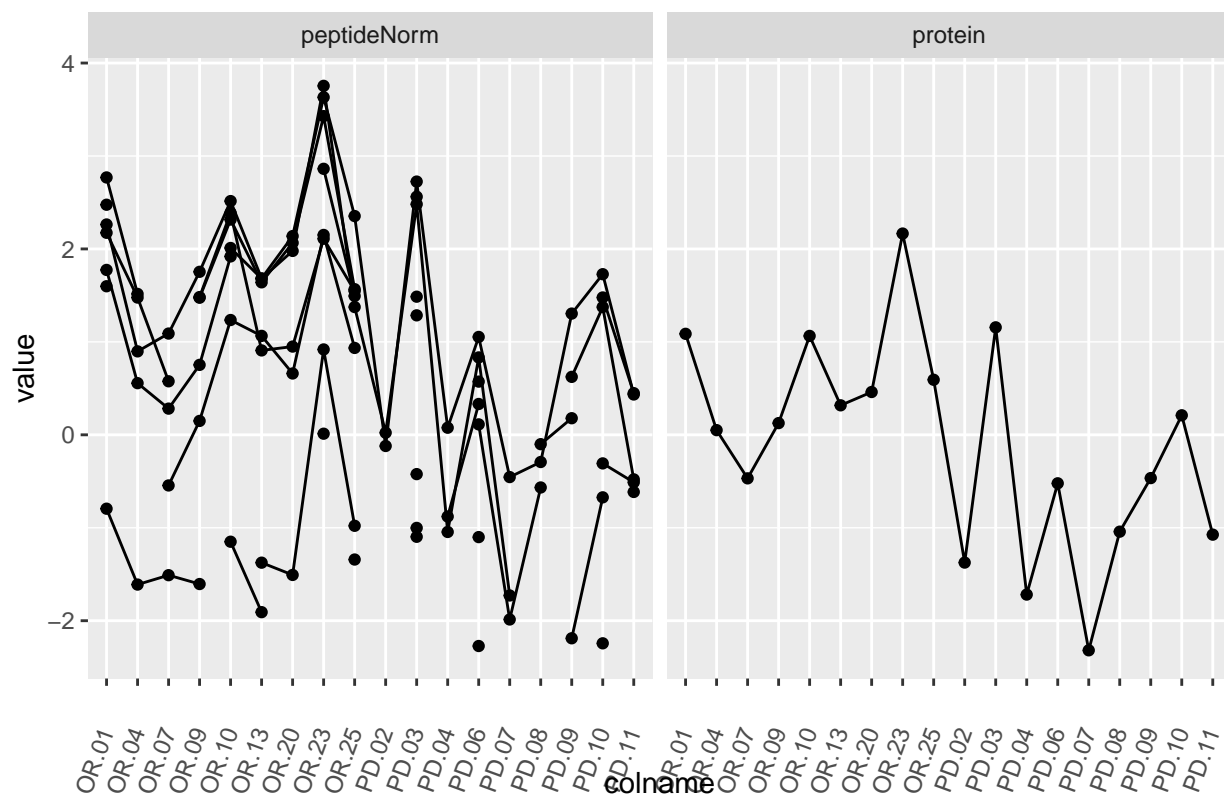


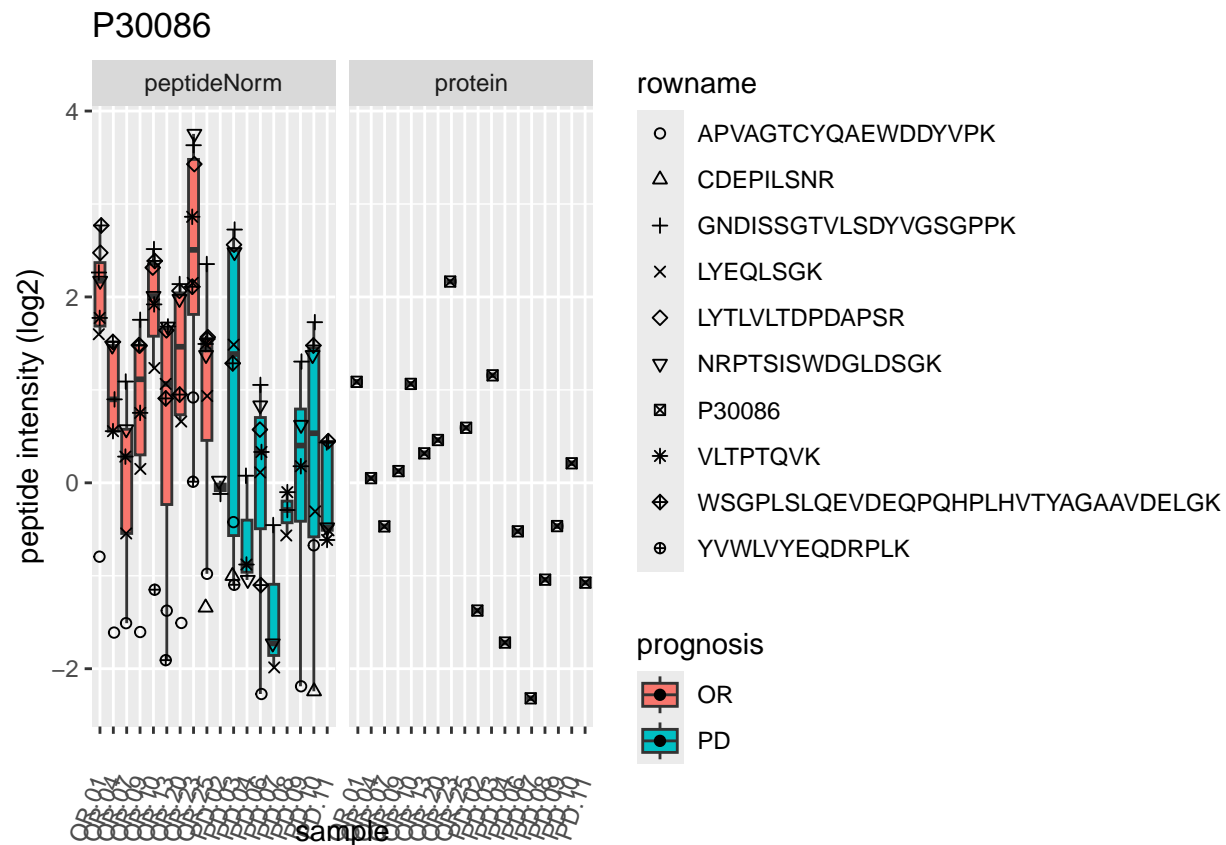
Q05682



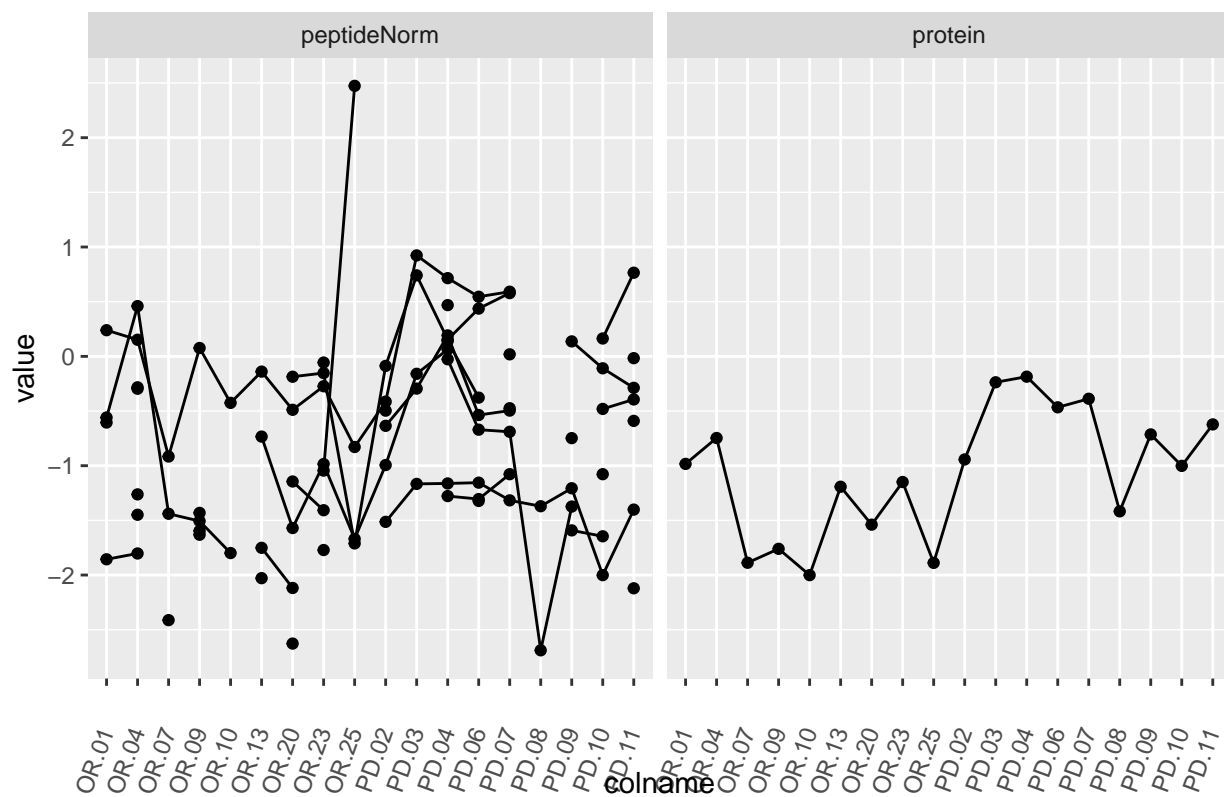


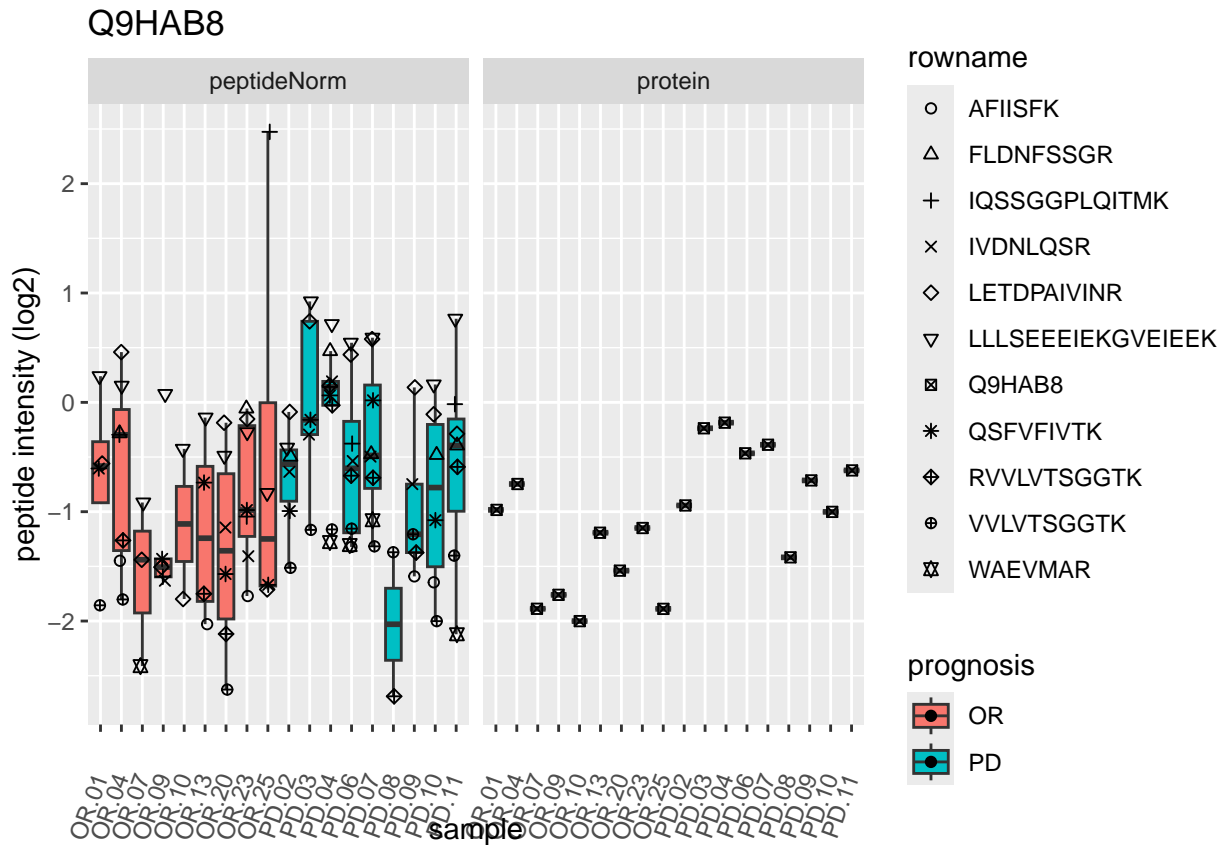
P30086



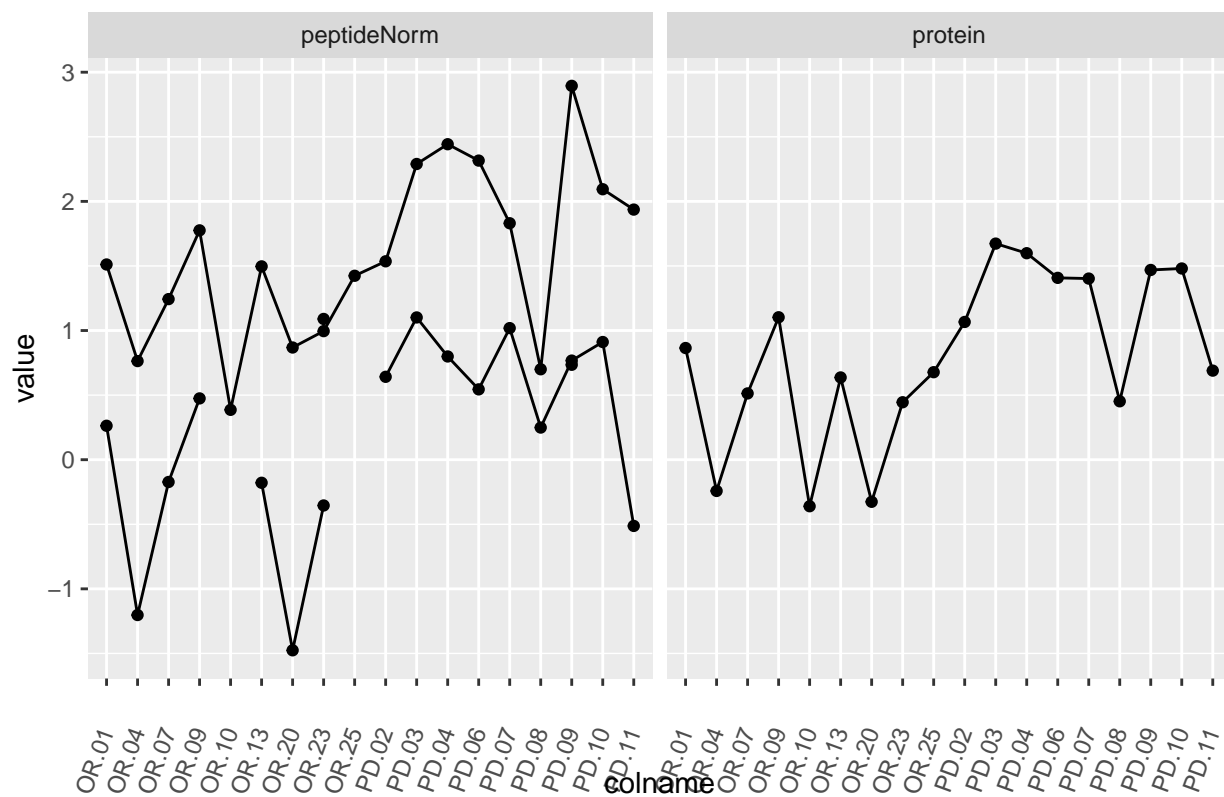


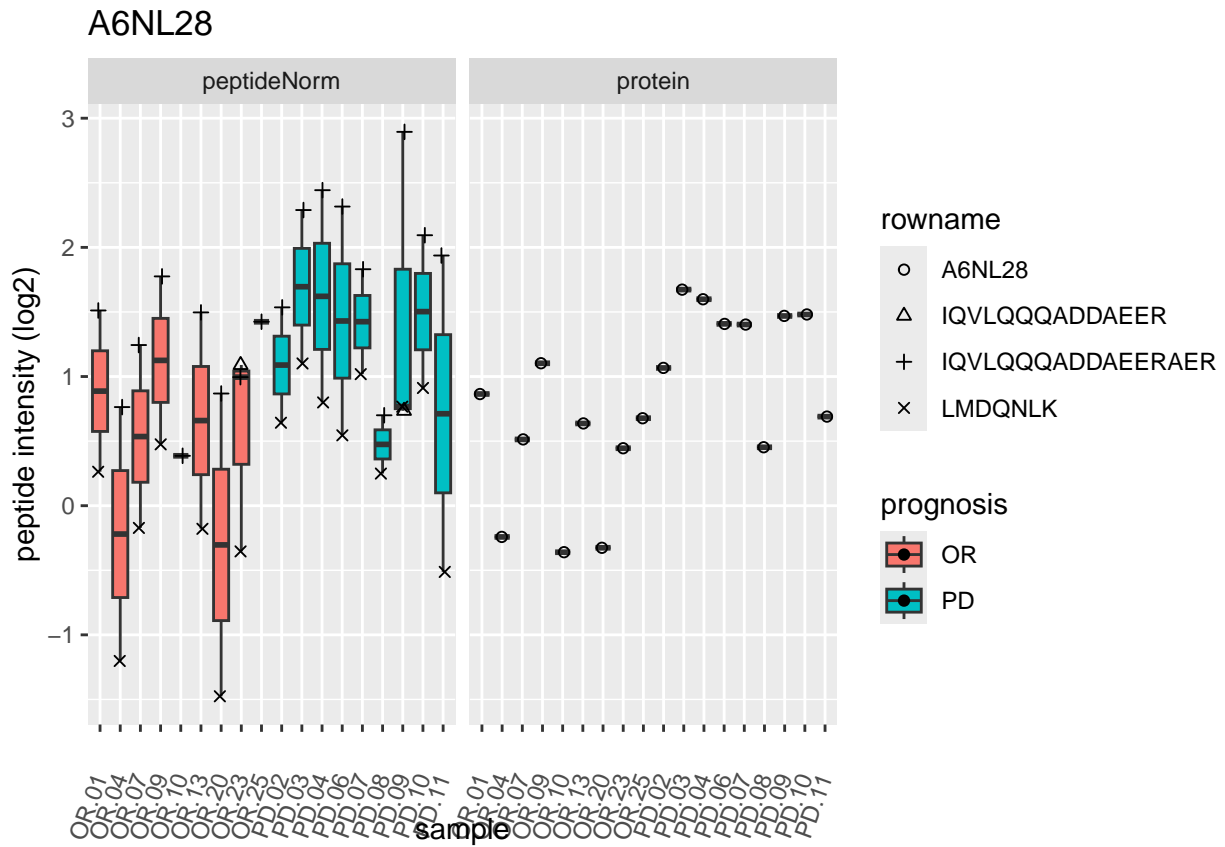
# Q9HAB8





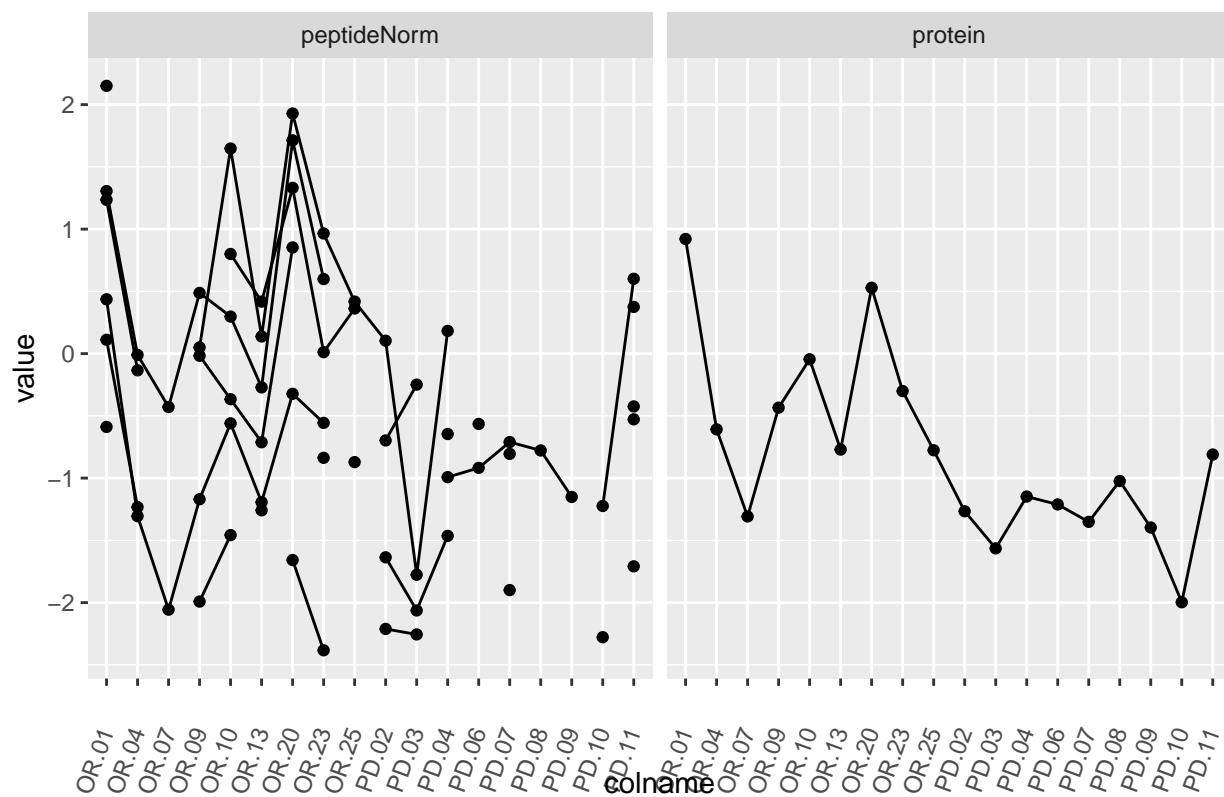
# A6NL28



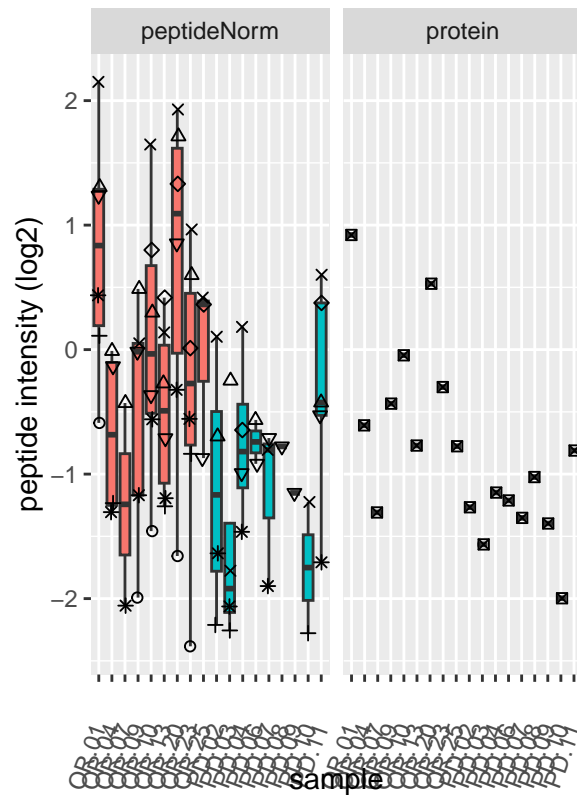




Q9H936



Q9H936



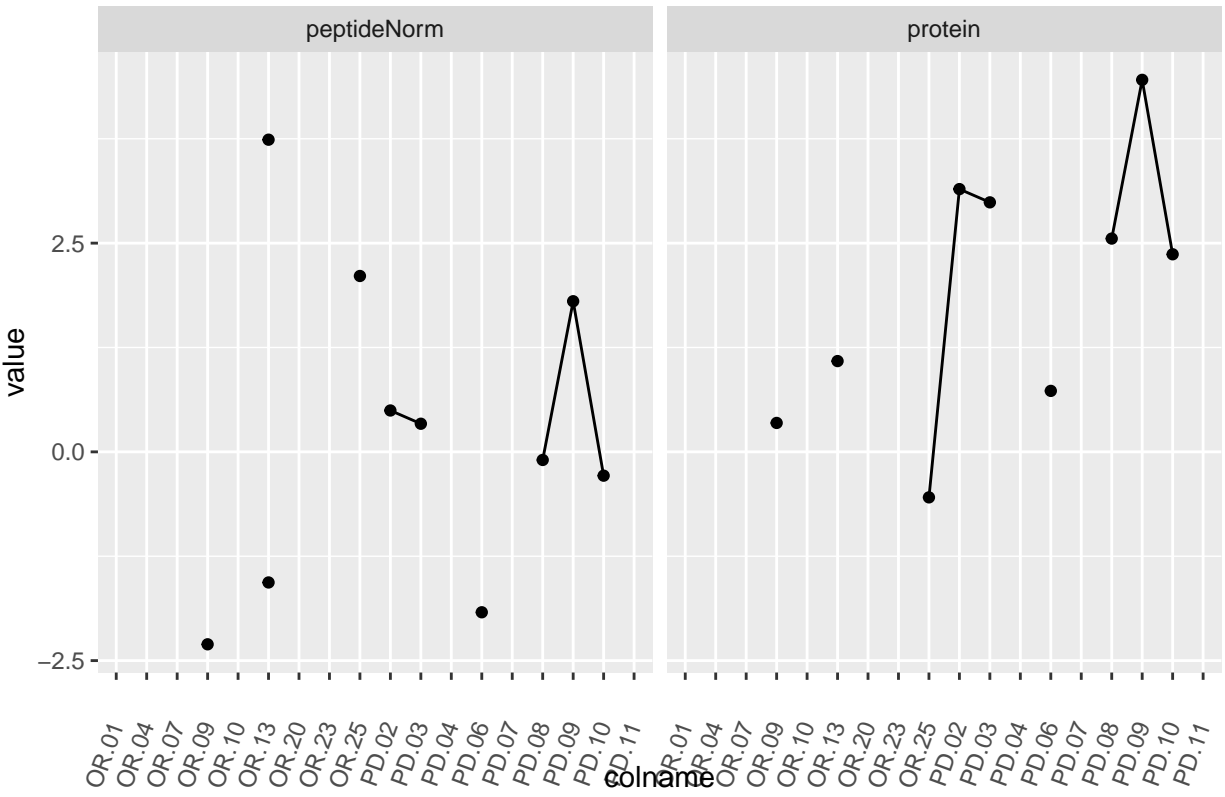
rowname

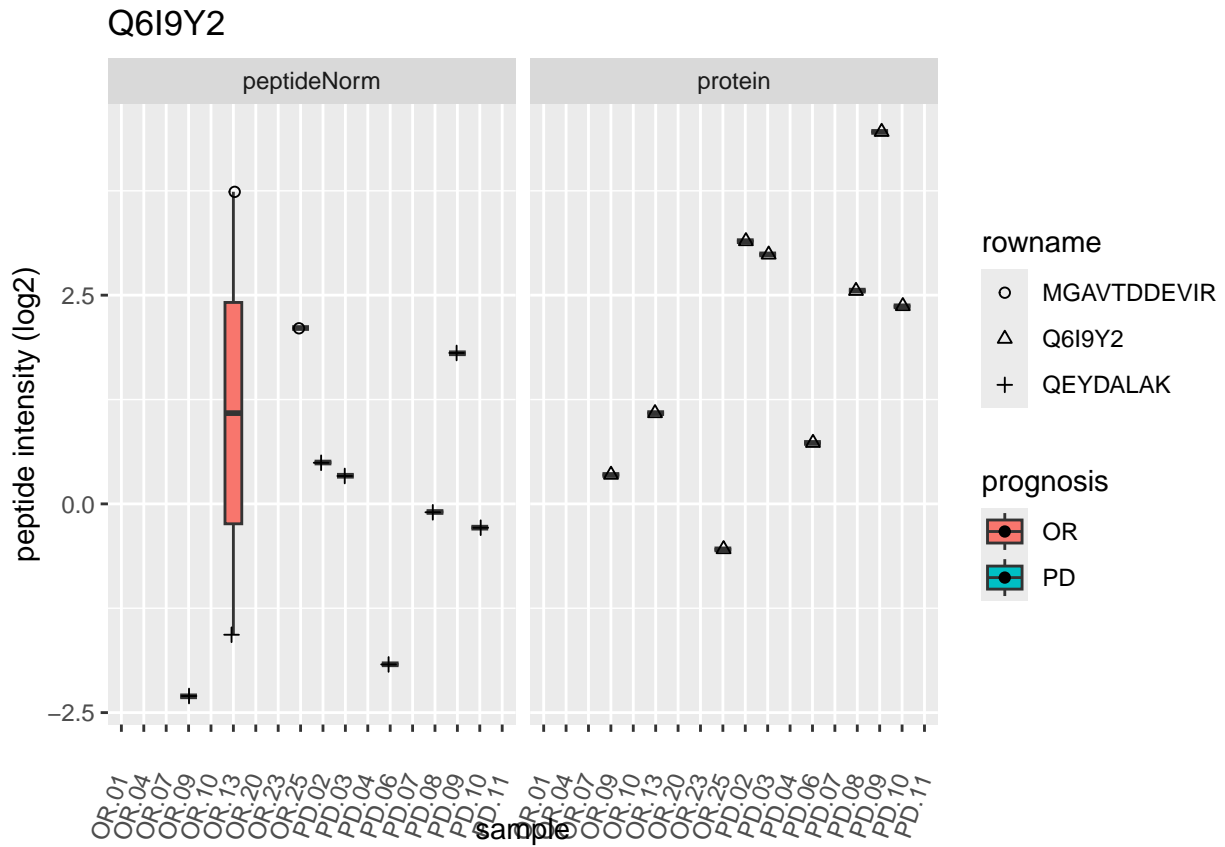
- DVPFSVVYFPLFANLNQLGRPASEEK
- △ GAAVNLTTLVTPEK
- + GLGATLLR
- × ILAAQGQLSAQGGAQPSVEAPAAPRPTATQLTR
- ◇ KILAAQGQLSAQGGAQPSVEAPAAPRPTATQLTR
- ▽ LAANDFFR
- ⊠ Q9H936
- \* SEGYFGMYR

prognosis

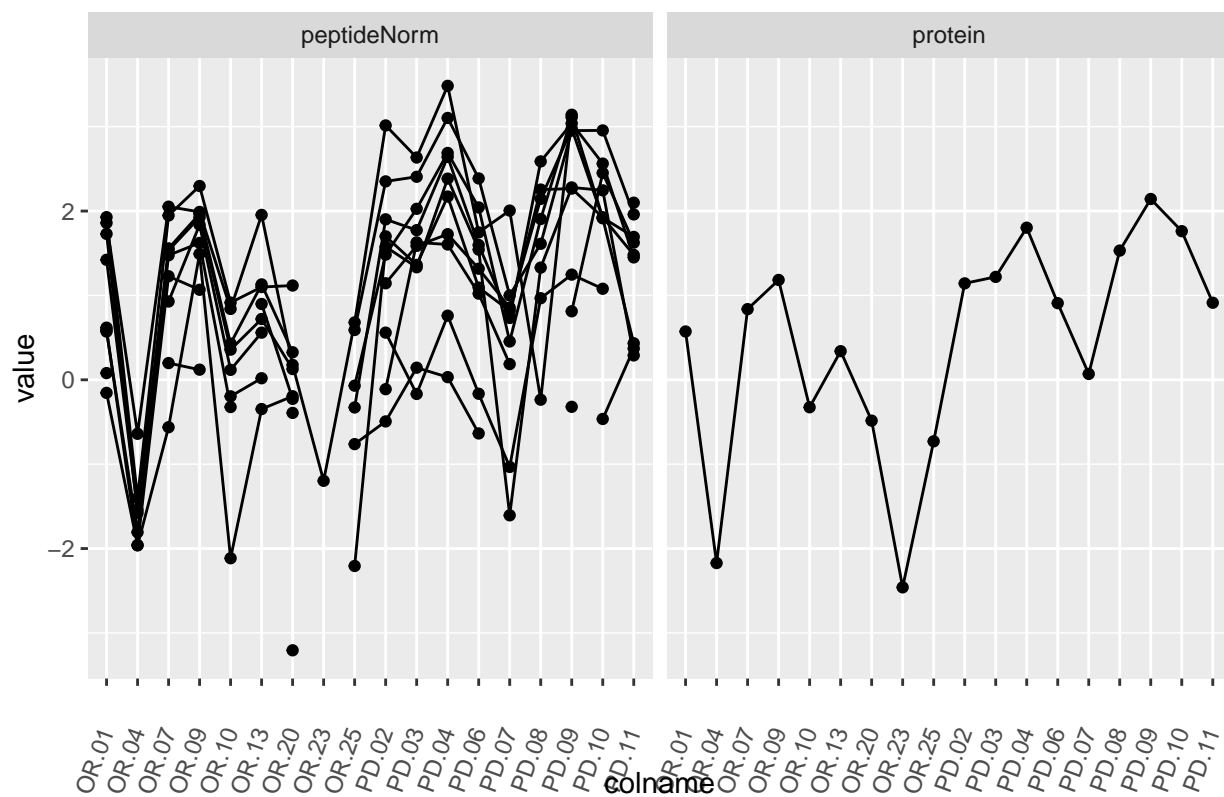
- OR
- PD

Q6I9Y2

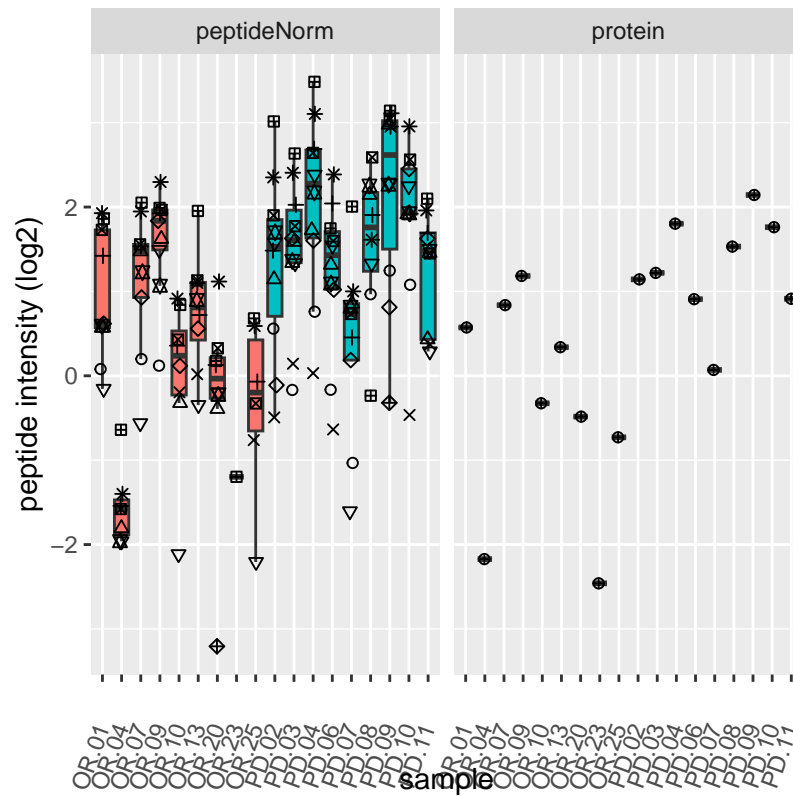




P67936



P67936



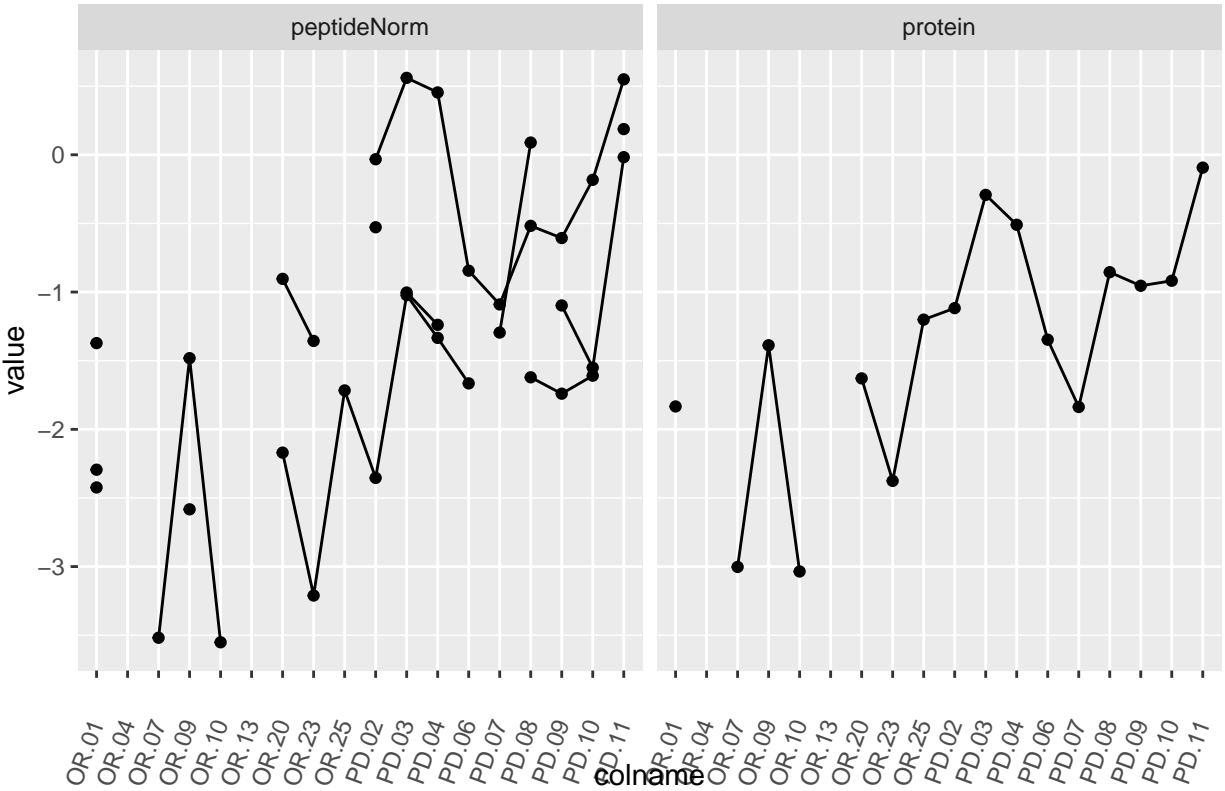
rowname

- AEGDVAALNR
- △ AGLNSLEAVK
- + AGLNSLEAVKR
- × CGDLEELKKNVTNNLK
- ◇ EENVGLHQTLDQTLNELNCI
- ▽ EKAEGDVAALNR
- ⊠ IQALQQQADEAEDR
- \* KIQUALQQQADEAEDR
- ⊕ KLVILEGELERAEER
- ⊗ P67936
- ⊠ TIDDEEEK
- ⊠ YSEKEDKYEEEEIK

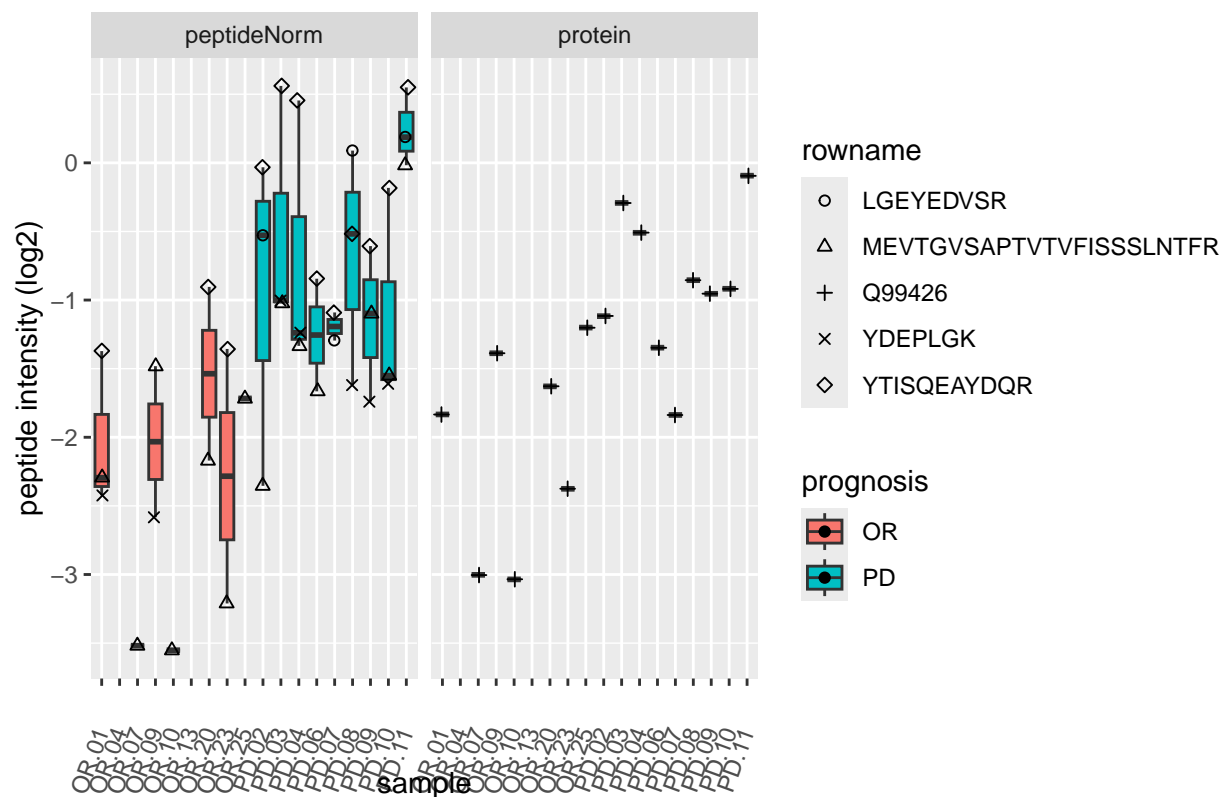
prognosis

- OR
- PD

Q99426

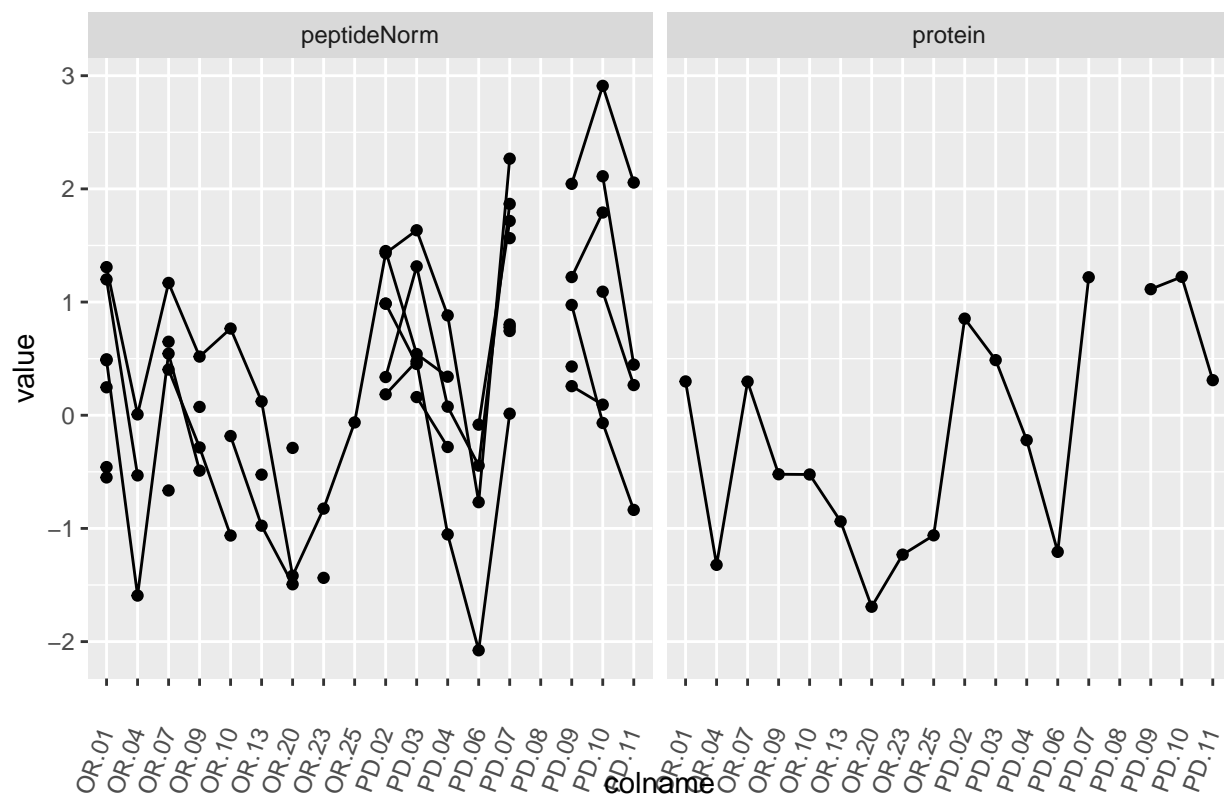


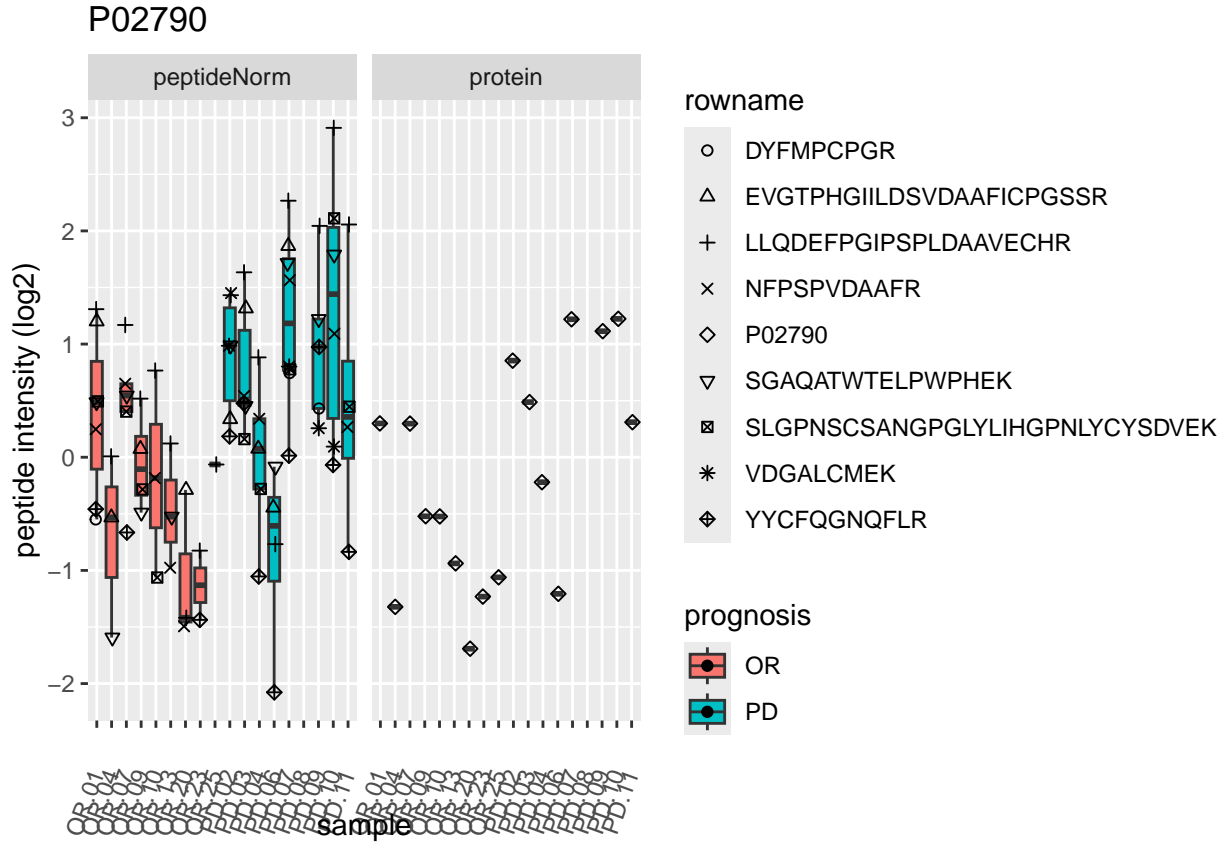
Q99426



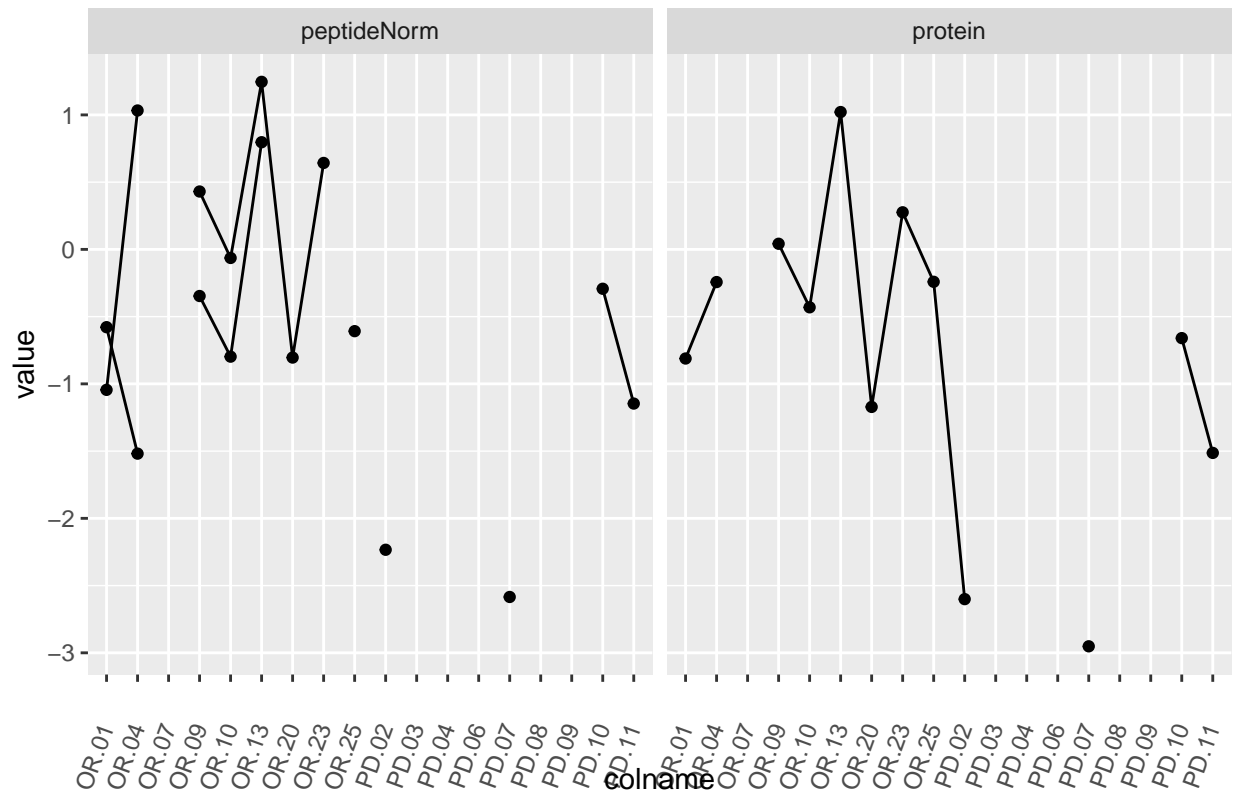


P02790

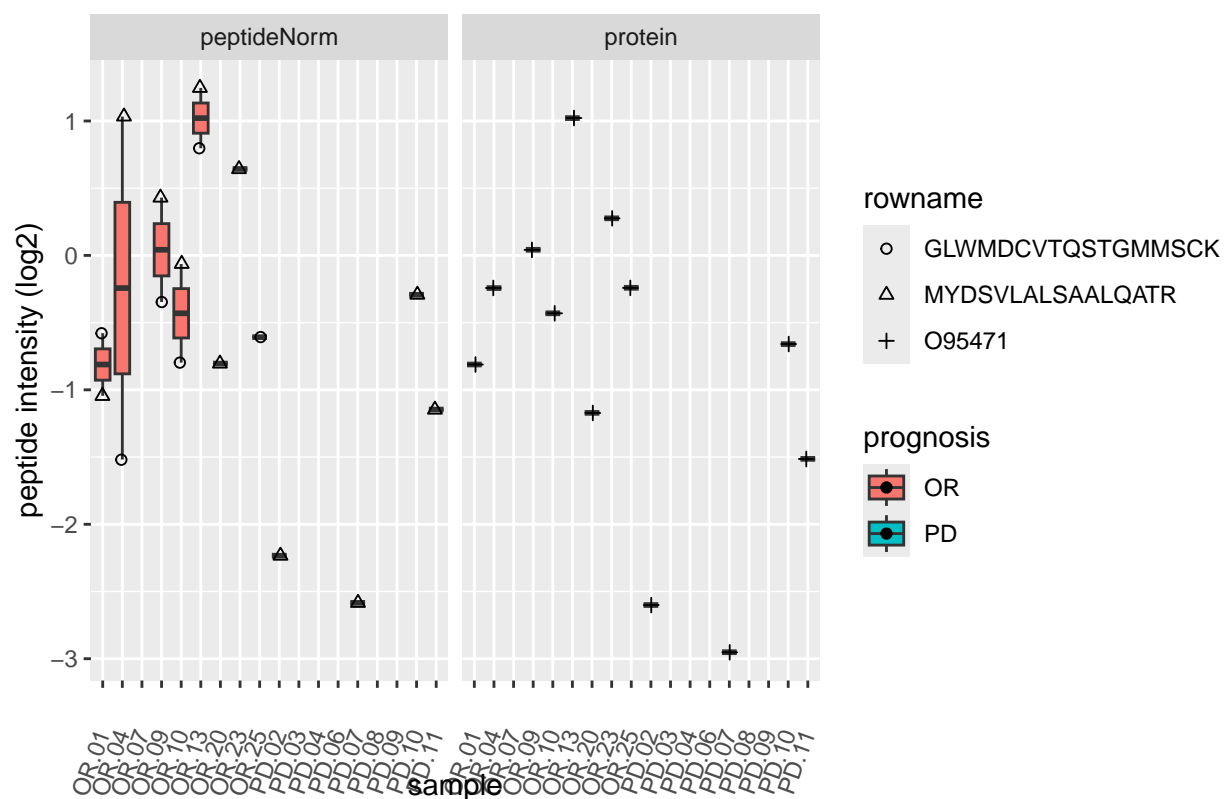




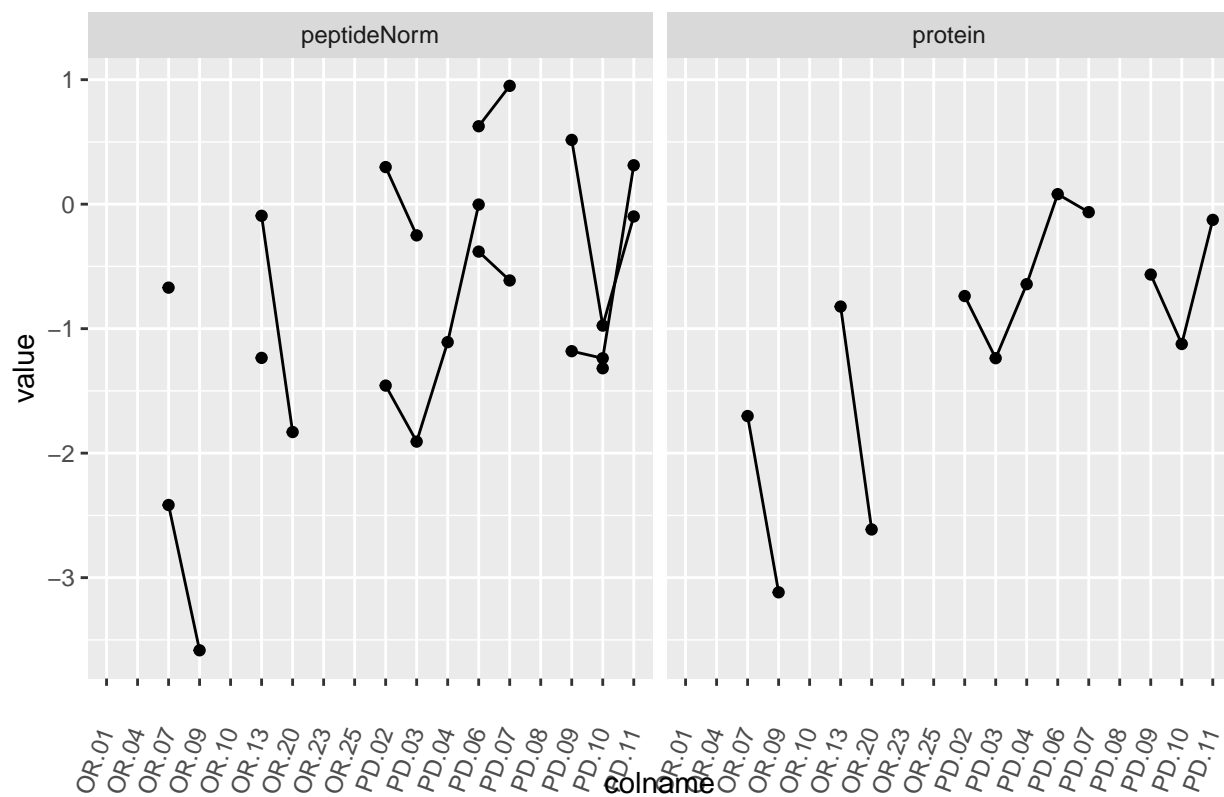
O95471

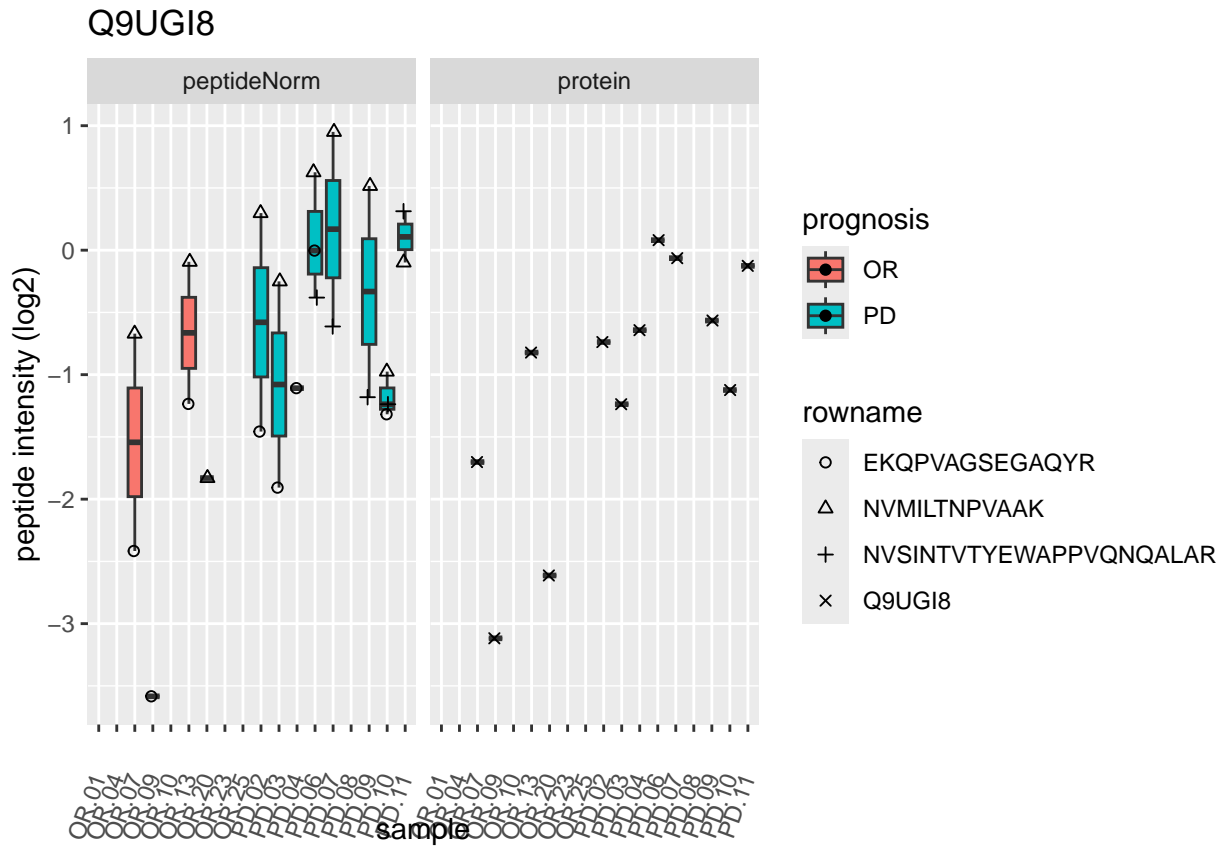


O95471

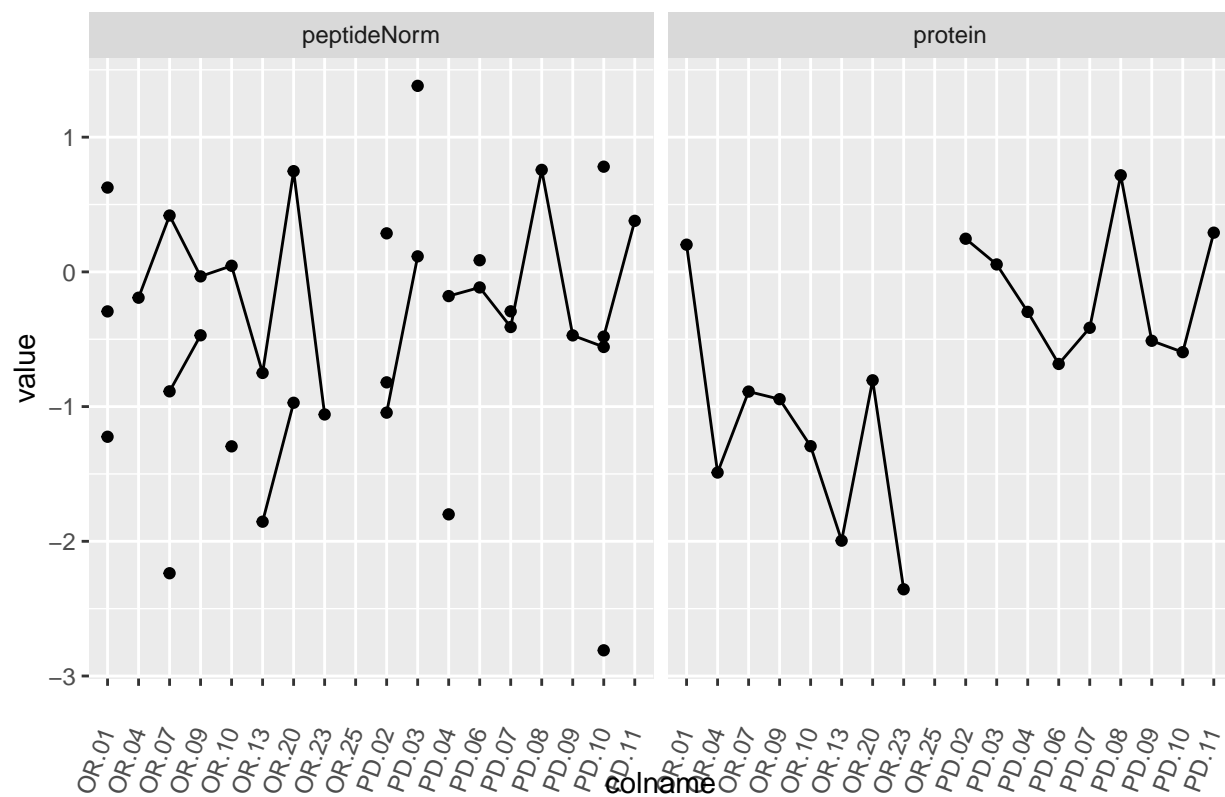


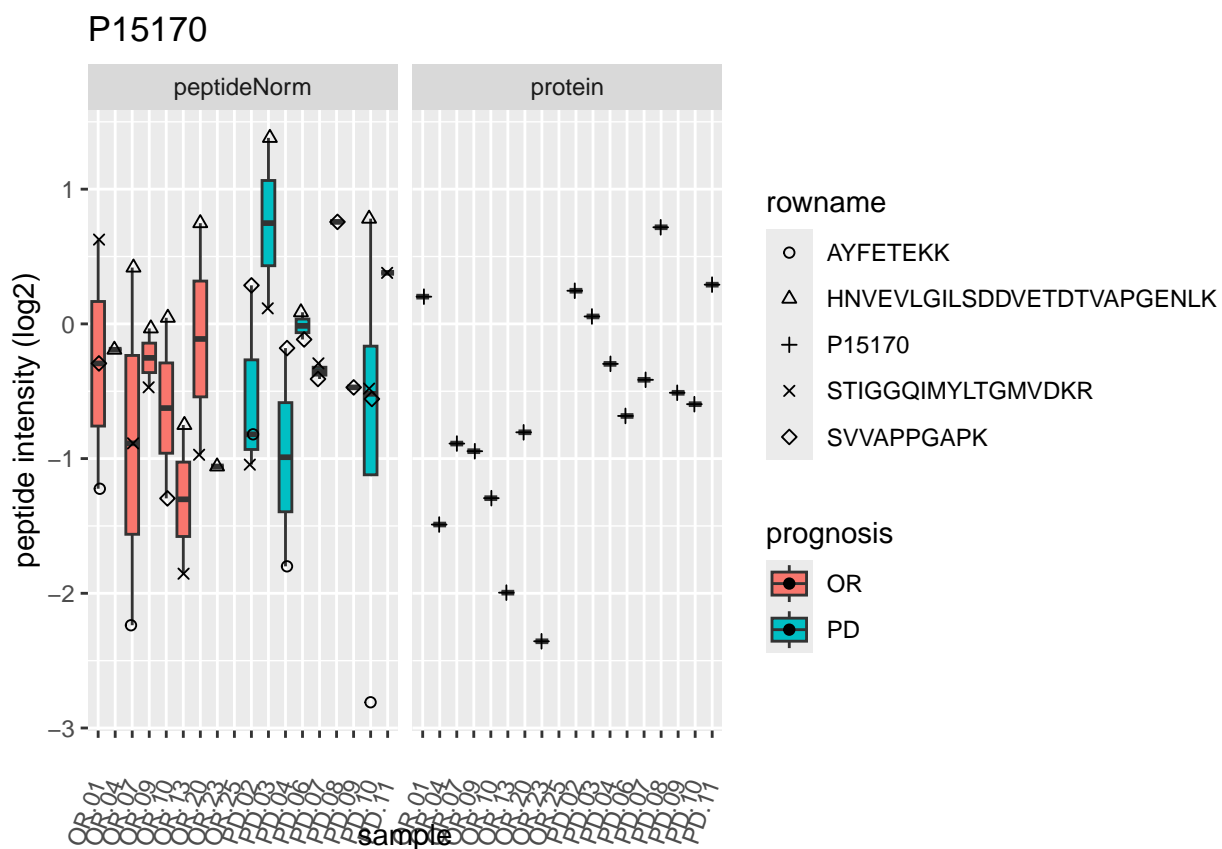
Q9UGI8





P15170



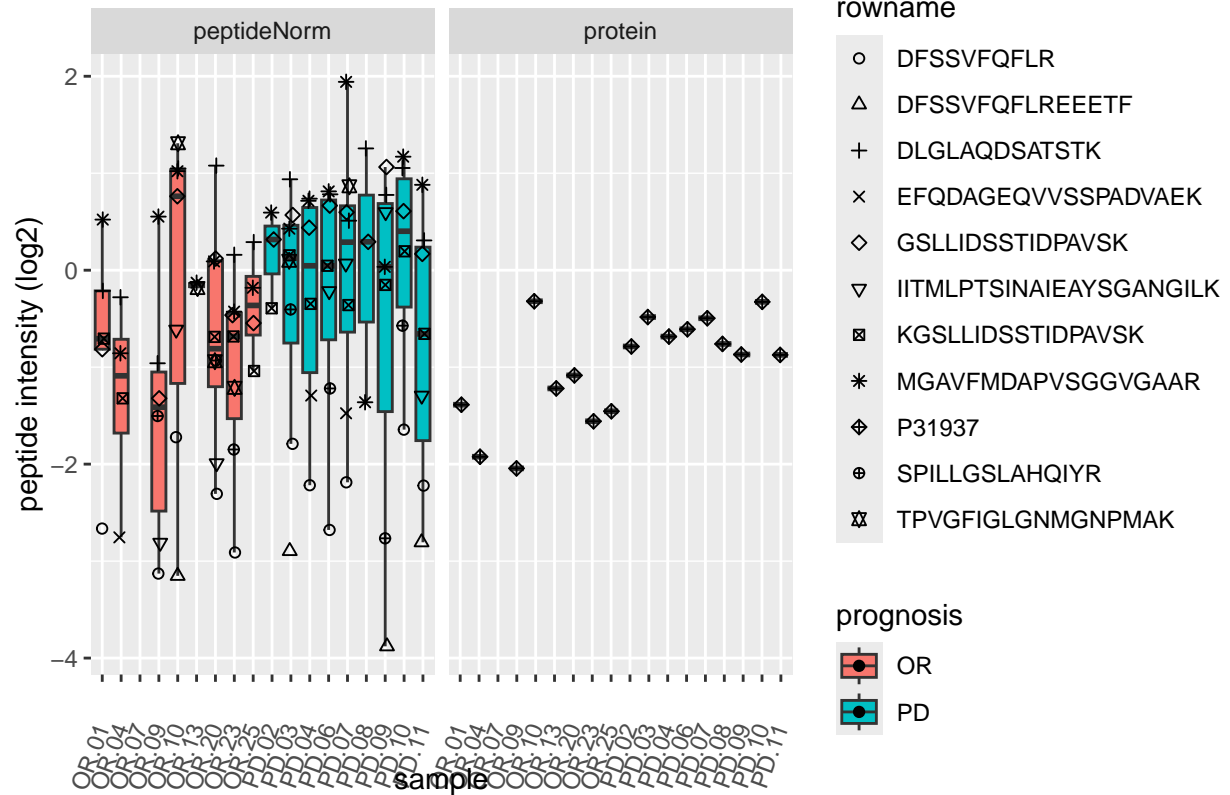




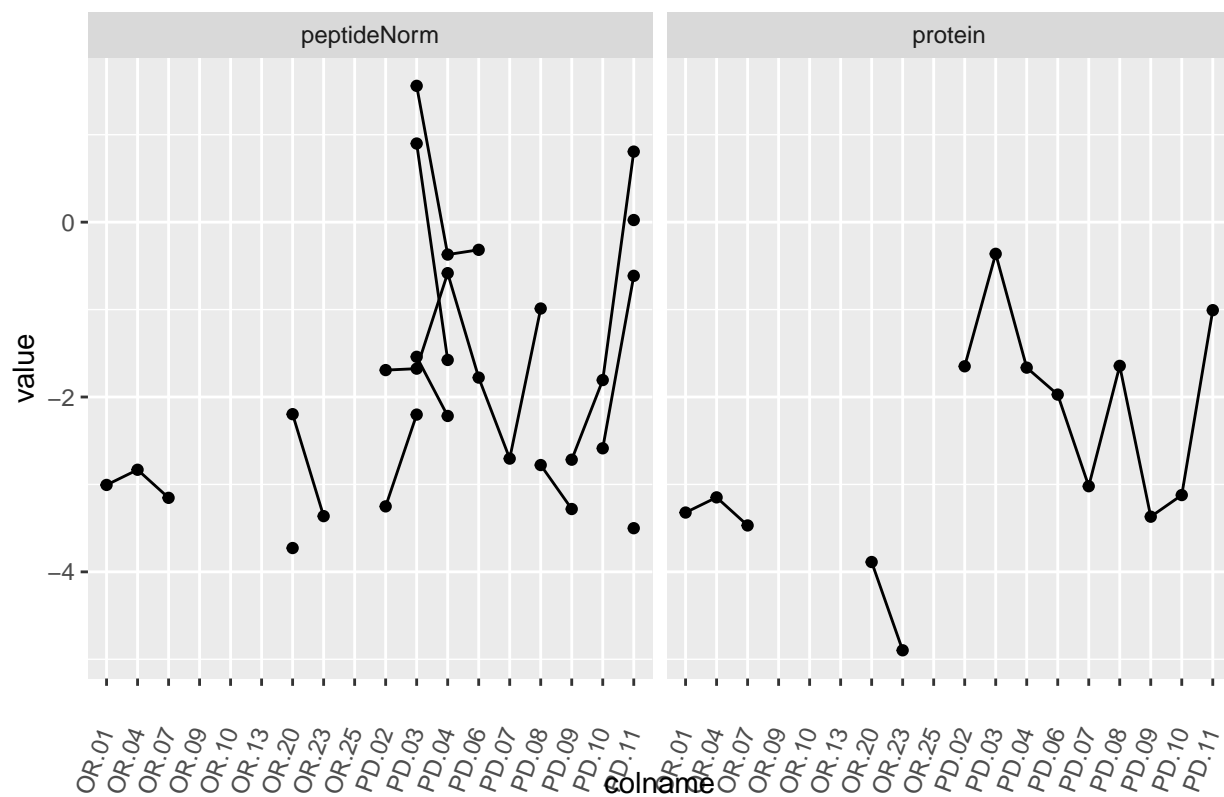
P31937



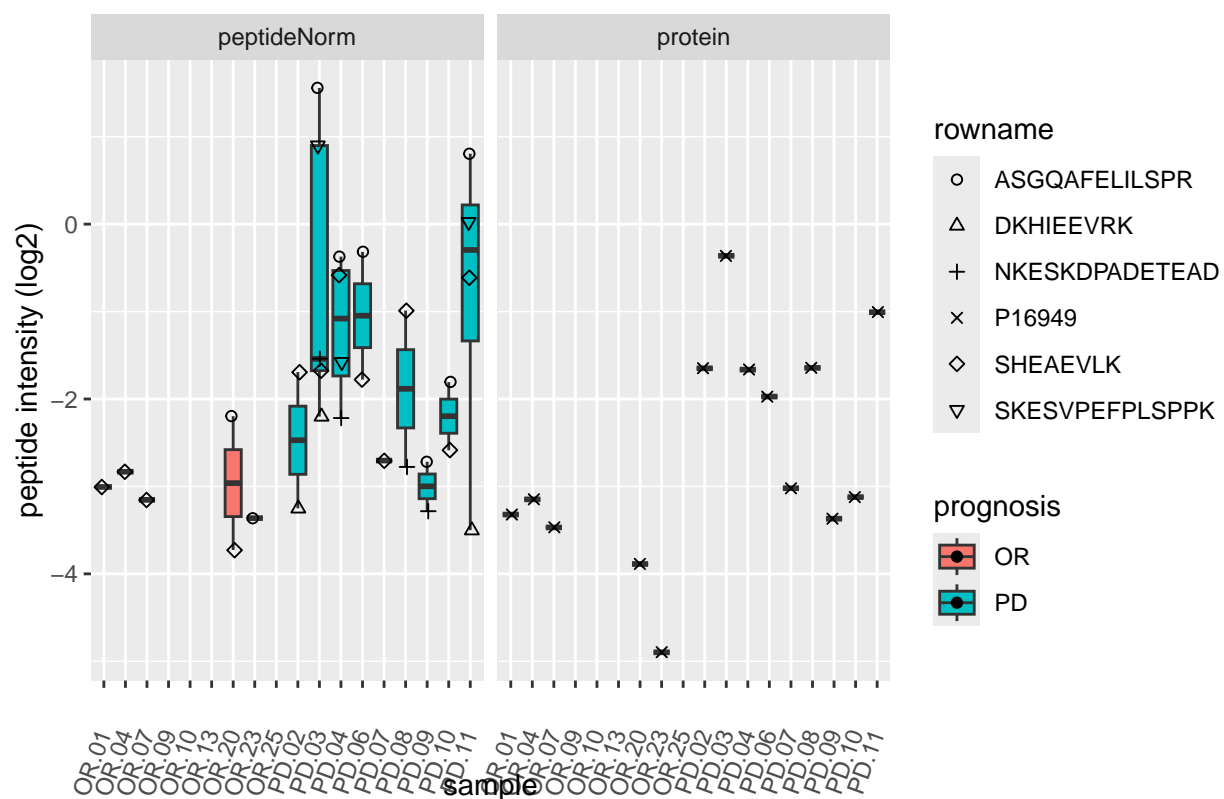
P31937



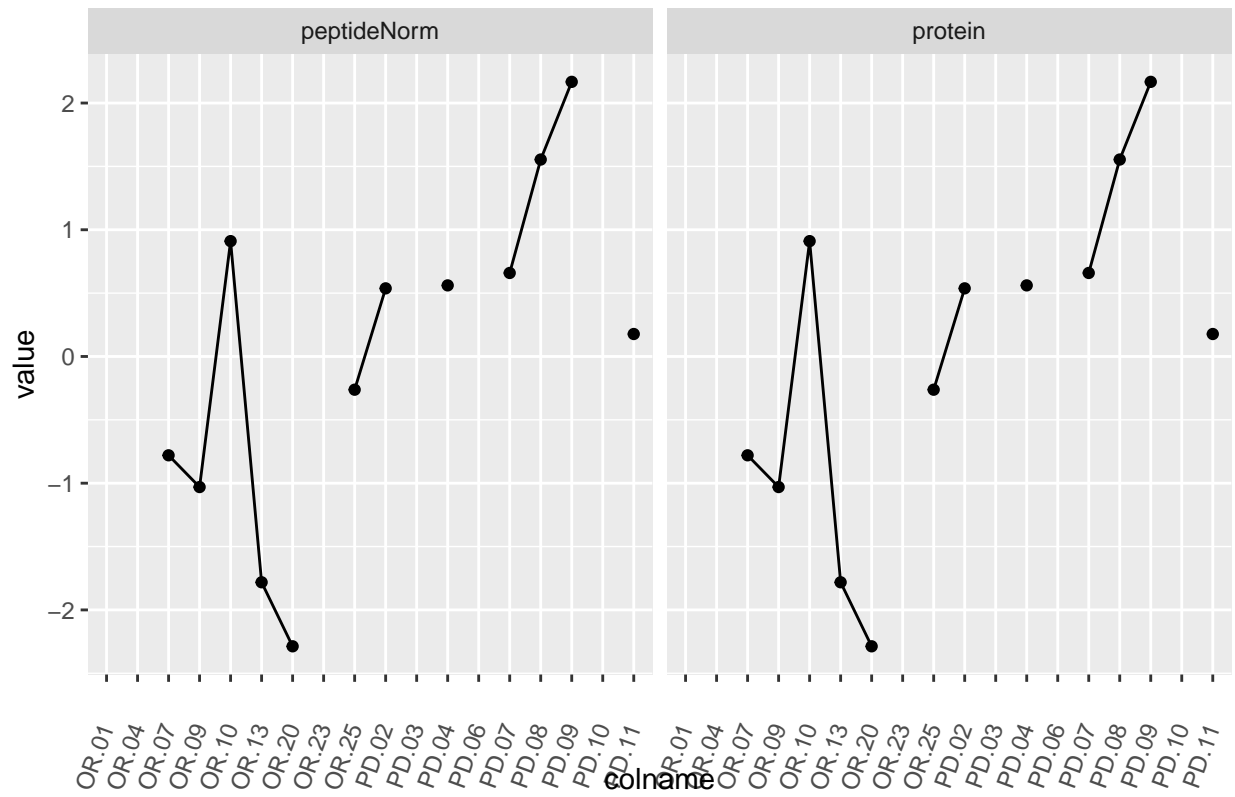
P16949



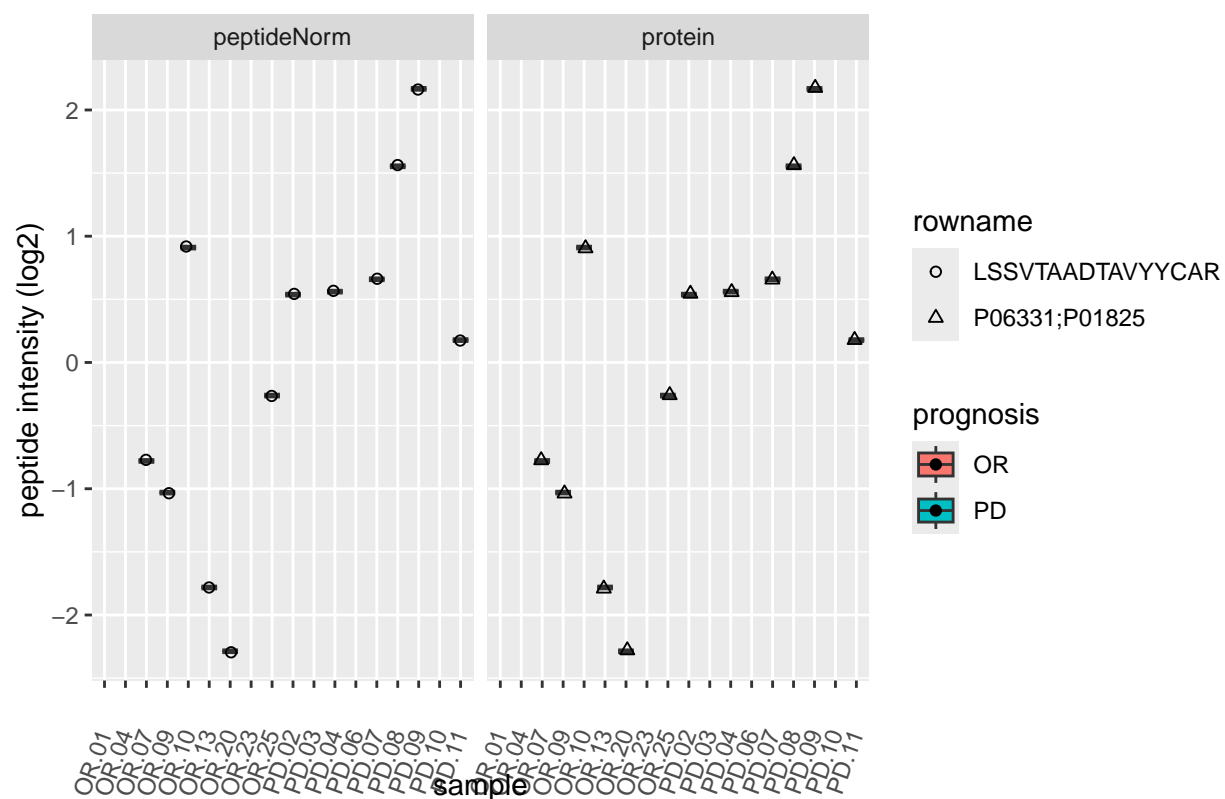
P16949



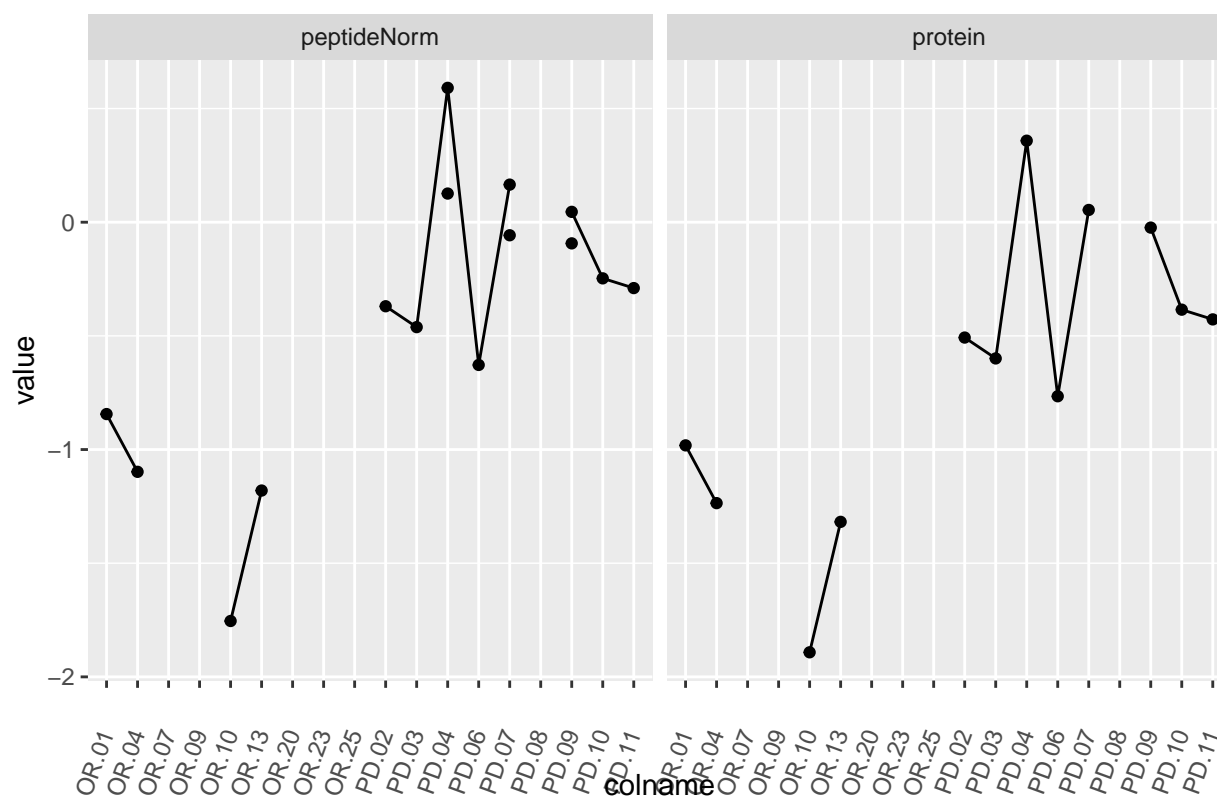
P06331;P01825



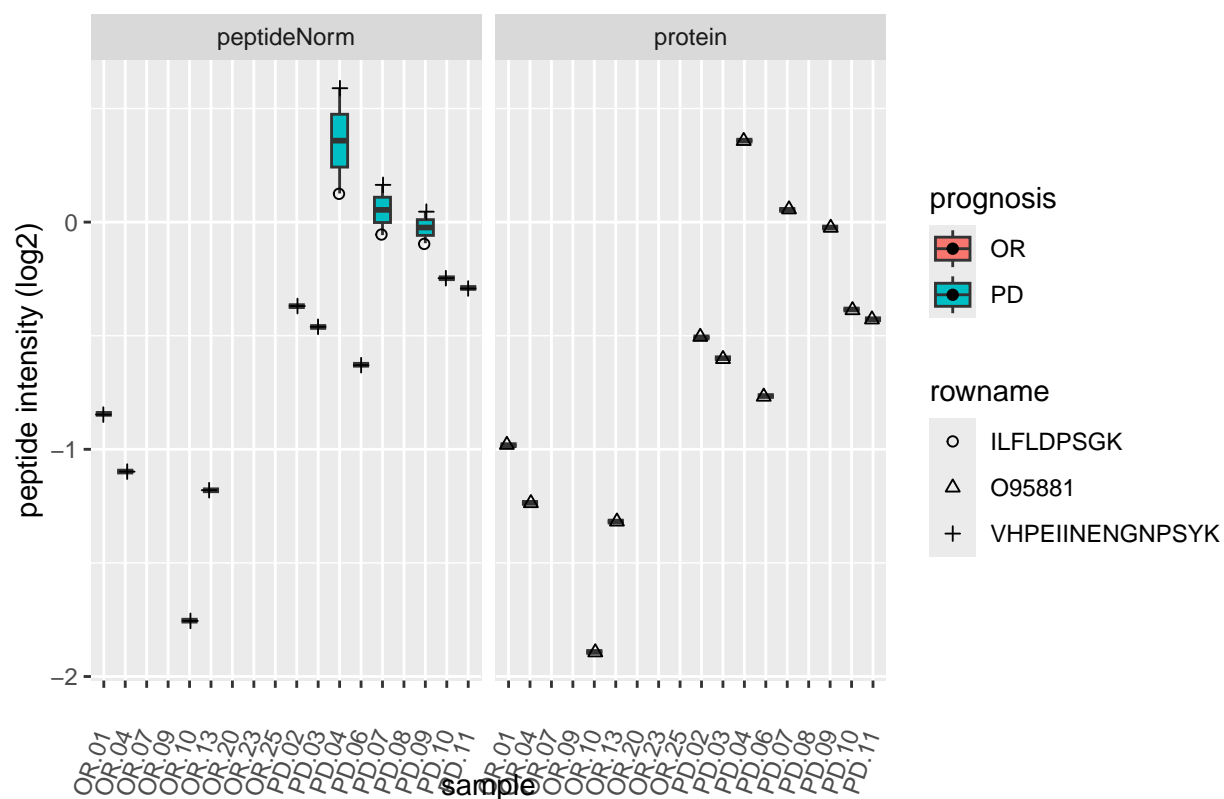
P06331;P01825



O95881

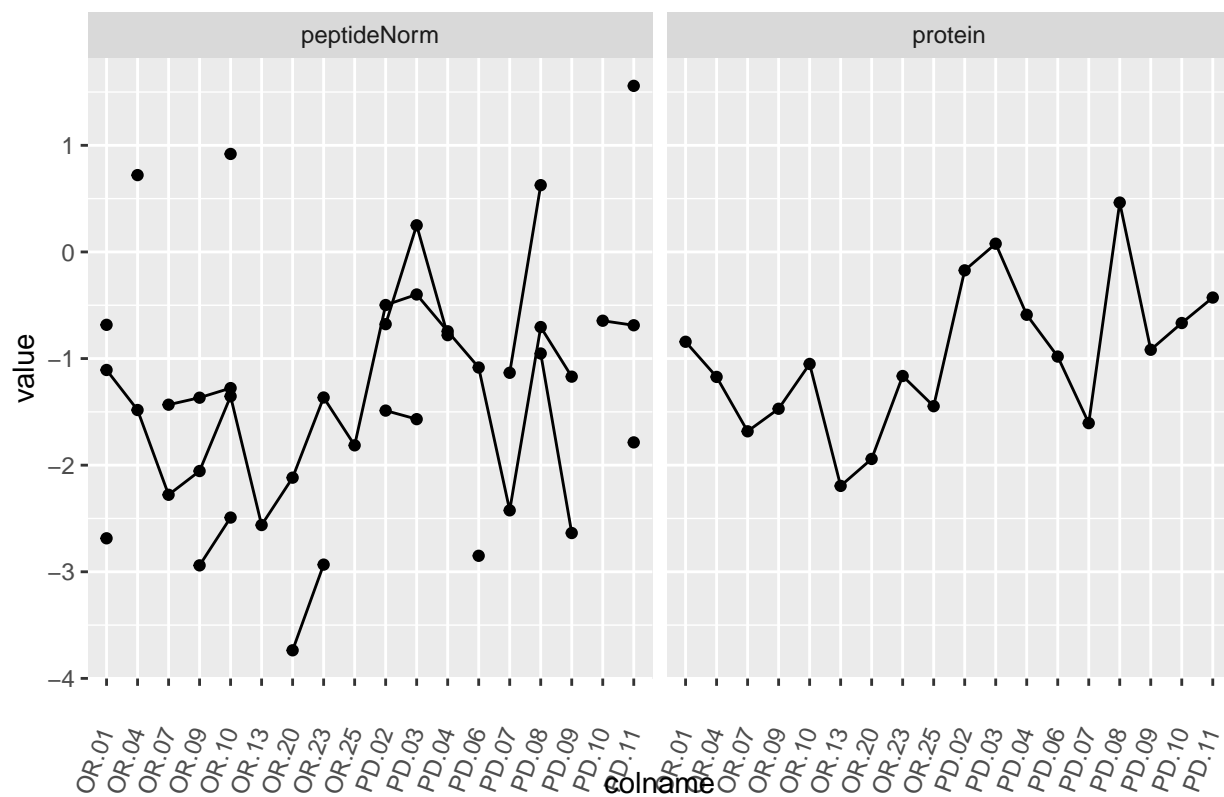


O95881

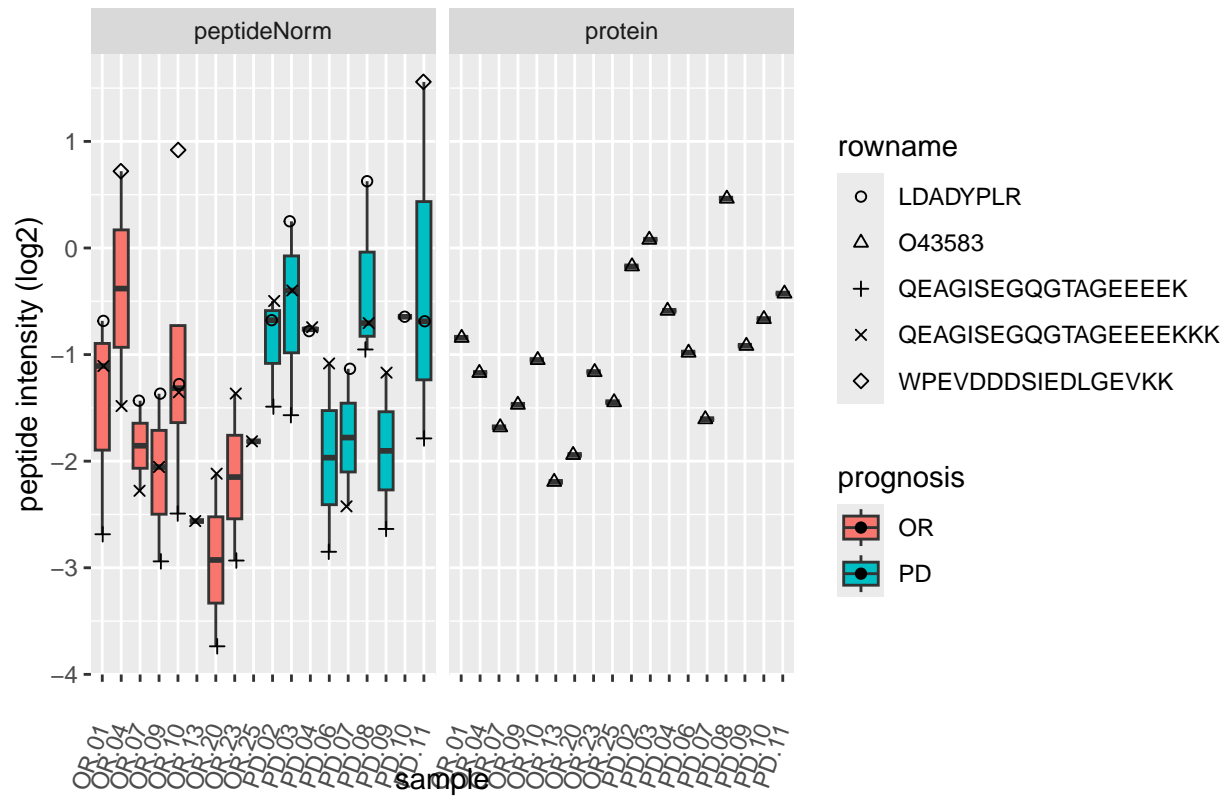




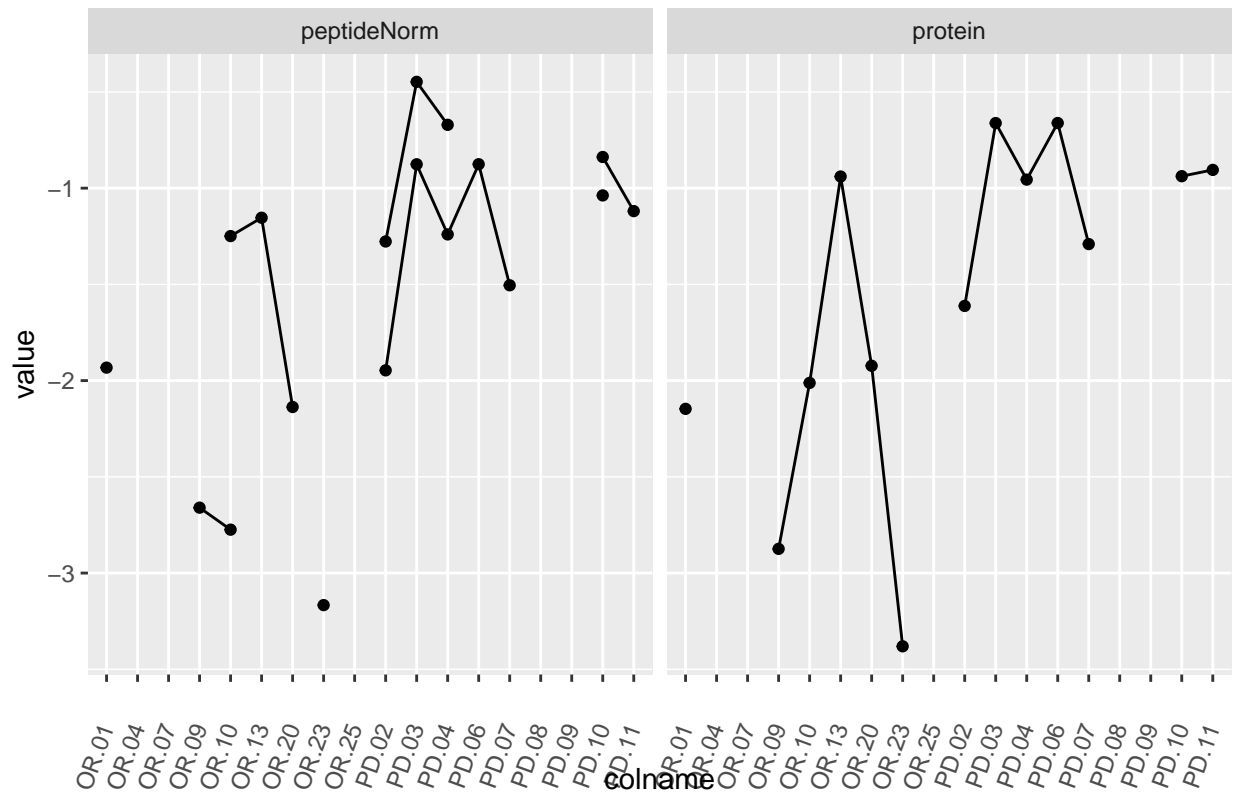
O43583



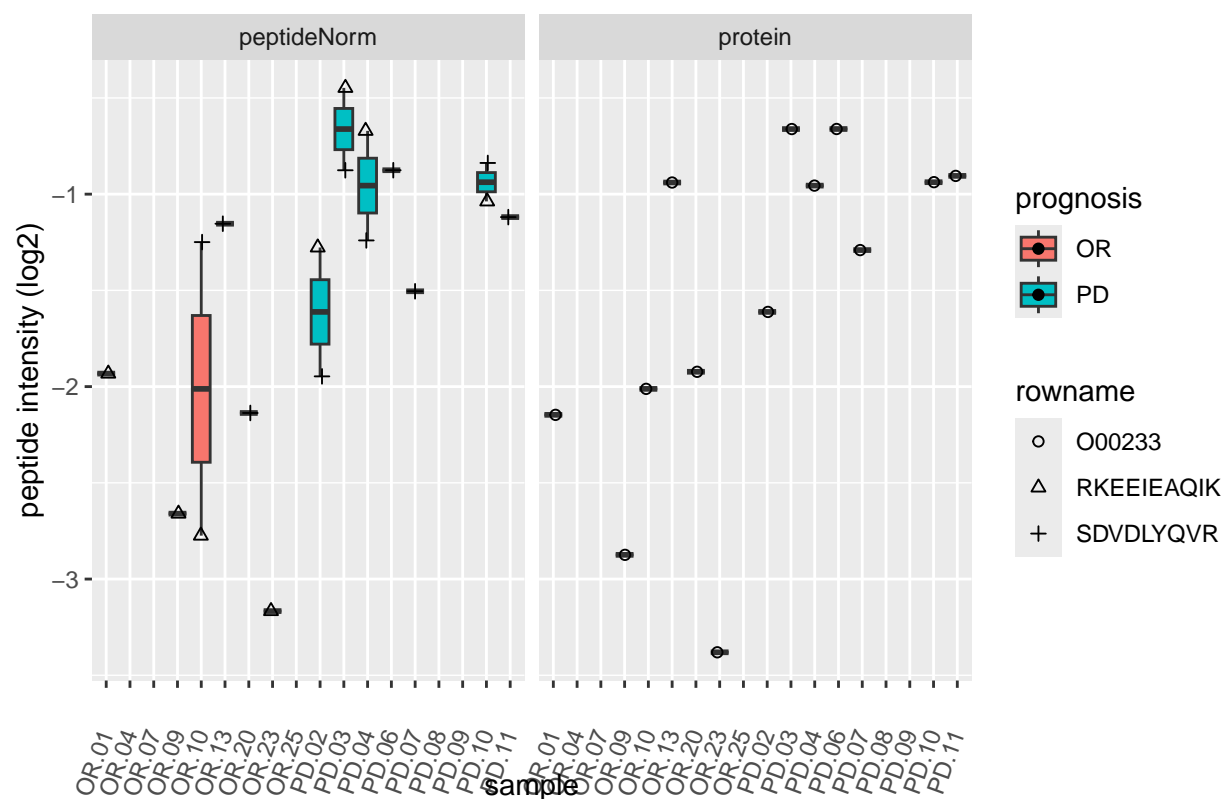
O43583



O00233

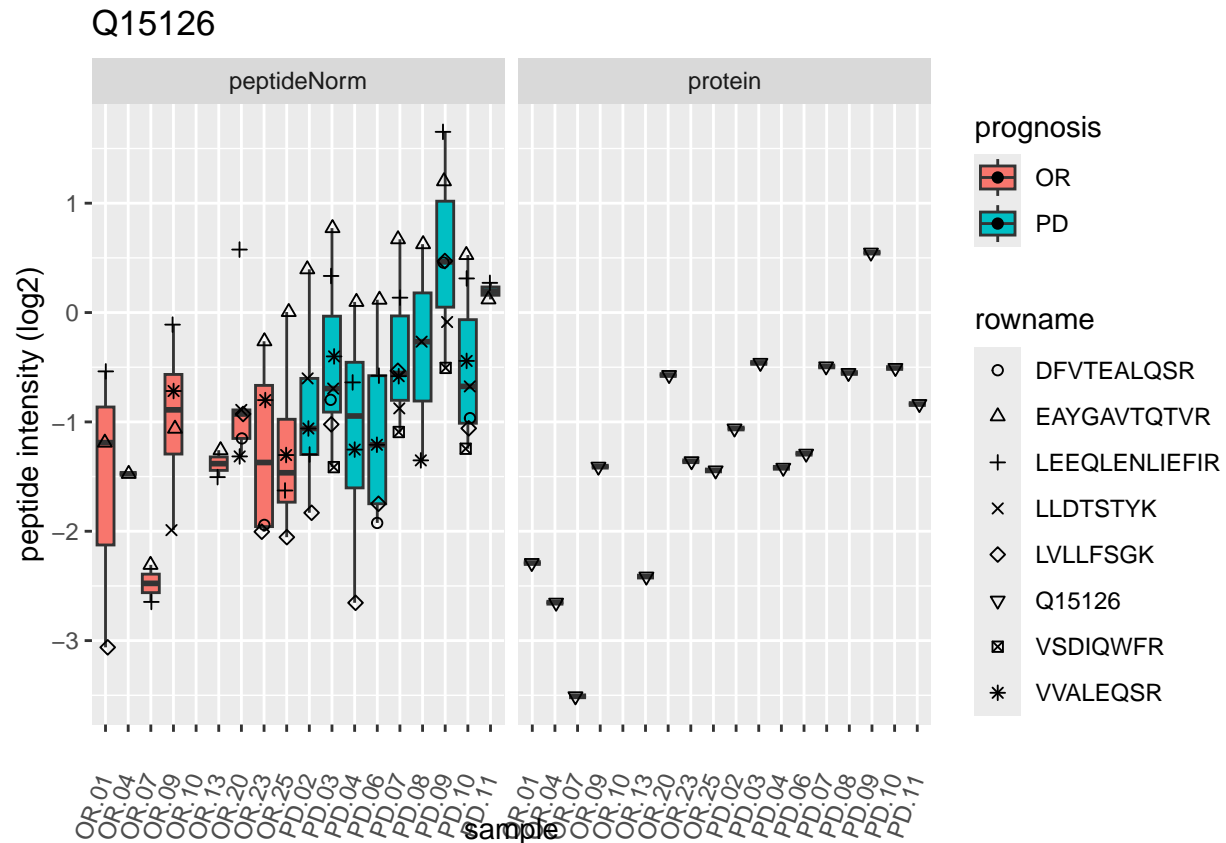


O00233



Q15126





## 5 Session Info

With respect to reproducibility, it is highly recommended to include a session info in your script so that readers of your output can see your particular setup of R.

```
sessionInfo()
```

```
## R version 4.4.0 RC (2024-04-16 r86468)
## Platform: aarch64-apple-darwin20
## Running under: macOS Big Sur 11.6
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRlapack.dylib; LAPACK v
##
## locale:
## [1] C/UTF-8/C/C/C/C
##
## time zone: Europe/Brussels
## tzcode source: internal
##
## attached base packages:
## [1] stats4 stats graphics grDevices utils datasets methods
## [8] base
##
## other attached packages:
```

```

## [1] ExploreModelMatrix_1.16.0    plotly_4.10.4
## [3] msqrob2_1.12.0                QFeatures_1.14.2
## [5] MultiAssayExperiment_1.30.3   SummarizedExperiment_1.34.0
## [7] Biobase_2.64.0                GenomicRanges_1.56.1
## [9] GenomeInfoDb_1.40.1          IRanges_2.38.1
## [11] S4Vectors_0.42.1             BiocGenerics_0.50.0
## [13] MatrixGenerics_1.16.0        matrixStats_1.4.1
## [15] limma_3.60.6                 lubridate_1.9.3
## [17] forcats_1.0.0                stringr_1.5.1
## [19] dplyr_1.1.4                  purrr_1.0.2
## [21] readr_2.1.5                  tidyr_1.3.1
## [23] tibble_3.2.1                 ggplot2_3.5.1
## [25] tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
## [1] rlang_1.1.4                  magrittr_2.0.3              shinydashboard_0.7.2
## [4] clue_0.3-65                  compiler_4.4.0              vctrs_0.6.5
## [7] reshape2_1.4.4              ProtGenerics_1.36.0        pkgconfig_2.0.3
## [10] crayon_1.5.3                fastmap_1.2.0              XVector_0.44.0
## [13] fontawesome_0.5.2           labeling_0.4.3             utf8_1.2.4
## [16] promises_1.3.0              rmarkdown_2.28             tzdb_0.4.0
## [19] UCSC.utils_1.0.0            nloptr_2.1.1              xfun_0.47
## [22] zlibbioc_1.50.0             cachem_1.1.0              jsonlite_1.8.9
## [25] later_1.3.2                 highr_0.11                 DelayedArray_0.30.1
## [28] BiocParallel_1.38.0         parallel_4.4.0            cluster_2.1.6
## [31] R6_2.5.1                    bslib_0.8.0               stringi_1.8.4
## [34] boot_1.3-31                  jquerylib_0.1.4            Rcpp_1.0.13
## [37] knitr_1.48                   BiocBaseUtils_1.6.0        httpuv_1.6.15
## [40] Matrix_1.7-0                splines_4.4.0             igraph_2.0.3
## [43] timechange_0.3.0            tidyselect_1.2.1          abind_1.4-8
## [46] yaml_2.3.10                  codetools_0.2-20          lattice_0.22-6
## [49] plyr_1.8.9                   shiny_1.9.1               withr_3.0.1
## [52] evaluate_1.0.0              pillar_1.9.0              DT_0.33
## [55] shinyjs_2.1.0               generics_0.1.3             hms_1.1.3
## [58] munsell_0.5.1               scales_1.3.0              minqa_1.2.8
## [61] xtable_1.8-4                glue_1.8.0                 lazyeval_0.2.2
## [64] tools_4.4.0                 data.table_1.16.0          lme4_1.1-35.5
## [67] cowplot_1.1.3               grid_4.4.0                 MsCoreUtils_1.16.1
## [70] colorspace_2.1-1            nlme_3.1-166              GenomeInfoDbData_1.2.12
## [73] cli_3.6.3                    fansi_1.0.6               S4Arrays_1.4.1
## [76] viridisLite_0.4.2           AnnotationFilter_1.28.0    gtable_0.3.5
## [79] rintrojs_0.3.4              sass_0.4.9                 digest_0.6.37
## [82] SparseArray_1.4.8           htmlwidgets_1.6.4          farver_2.1.2
## [85] htmltools_0.5.8.1           lifecycle_1.0.4           httr_1.4.7
## [88] mime_0.12                    statmod_1.5.0             MASS_7.3-61

```