```
#----Preliminaries----

# READ ME
#Format your data in excel similar to the supplied
Sample Data file (.xlsx) with samples as rows and
measured variables as columns
#Column A: "Sample" in A1; the rest of your A cells
should have your sample labels; label background
samples "Background" in the end rows
#Column B: your variable of interest to be regressed
against, the response variable; B1 should have the
name of the response variable (i.e. Iba1);
  #  the rest of the B cells have measured values of
the variable
#Column C and onward: the first row should have the
label, the following rows should have measured values
of predictor variables (i.e. cytokines or phospho-
proteins)

rm(list=ls()) #Remove all variables from current
environment



#User Input   (Change this so it fits for your data)
dataFileName="MyData.xlsx"  #the name of your data
file, which should be saved in the same folder as this
R file
SwapAxes=FALSE  #If you want to swap LV1 and LV2, make
this TRUE, otherwise FALSE


#Set working directory and load needed libraries
setwd(dirname(rstudioapi::getActiveDocumentContext()
$path)) #Set working directory to the folder where the
code file resides
if(!"pacman" %in% installed.packages())
{install.packages("pacman")}
pacman::p_load(tidyverse,rio,gplots,matrixStats,readxl
```

```
#subtract background from predictor variables
dataPredictorVariables=dataPredictorVariablesIn
for (i in 1:ncol(dataPredictorVariables))
{
  dataPredictorVariables[,i]=dataPredictorVariables[,i
mean((dataBackground[,i]))  #subtract each column of
data (each cytokine) by the mean of that cytokine's
background values
}
dataPredictorVariables[dataPredictorVariables<0]=0;
#if value is below background, set to zero
rownames(dataPredictorVariables)=dataInALL$Sample[1:nu

dataPredictorVariablesZ=apply(dataPredictorVariables,
2,scale); #z-score the data
dataResponseVariableZ=apply(dataResponseVariable,
2,scale); #z-score response variable (for use in
heatmap)
rownames(dataPredictorVariablesZ)=rownames(dataPredict



#----PLS----
#Remove constant columns and then run the PLS
indConstantColumns=colSds(dataPredictorVariablesZ)==0
plsOut=plsRglm(dataY=dataResponseVariable,dataX=dataPr
indConstantColumns],2,modele="pls", scaleX = TRUE)

P=varimax(plsOut$pp) #Use varimax to rotate the LVs

#Return the loadings with missing values set to 0
P1=matrix(0,ncol = 1,
nrow=dim(dataPredictorVariablesZ)[2])
P2=matrix(0,ncol = 1,
nrow=dim(dataPredictorVariablesZ)[2])
P1[!indConstantColumns]=P$loadings[,1]
P2[!indConstantColumns]=P$loadings[,2]
rownames(P1)=colnames(dataPredictorVariablesZ)
rownames(P2)=colnames(dataPredictorVariablesZ)
```

```
        text = element_text(size=20),
        panel.border=element_blank(),
        plot.title = element_text(hjust = 0.5),
        axis.text = element_text(color = "black"),
        axis.ticks=element_line(color="black")
        )+
  labs(color = "")
dev.off()

#LV1 versus Response Variable graph to determine
acceptability of regression
LV1Regression=data.frame(T1,(dataResponseVariable))
colnames(LV1Regression)=c("T1","ResponseVariable")
linmodel=lm(ResponseVariable~T1,LV1Regression)
RSQ=summary(linmodel)$r.squared

#Bar plot the loadings
P1Sort=sort(t(P1))/max(abs(P1))
indP1=sort(P1, index.return=TRUE)$ix
pdf("LV1vsResponseVariable.pdf",
width=4,height=4,useDingbats=FALSE, pointsize = 18)
ggplot(LV1Regression, aes(x=T1, y=ResponseVariable)) +
  geom_smooth(method='lm',se=FALSE,color='red',linetyp
+
  geom_point(size=4)+
  sc+
  xlab("Scores on LV1")+
  ylab("ResponseVariable")+
  ggtitle("PLSR")+
  xlim(1.03*min(T1),1.03*max(T1))+
  ylim(0.97*min(dataResponseVariable),
1.03*max(dataResponseVariable))+
  annotate("text",size=5, x=0.8*max(T1)+0.2*min(T1),
y=min(dataResponseVariable), label=bquote({R^2}
[PLS]==.(round(RSQ,digits=2))))+
  theme(panel.background = element_rect(fill =
'white'),
        plot.title=element_text(hjust=0.5),
        text = element_text(size=20),
```

```
        names.arg = rownames(P2)[indP2],
        col = "lightblue",
        horiz = FALSE,
        las=2,
        ylim=c(-1,1))
dev.off()


#Sort data based on Response Variable values (used in
displaying heatmap)
ResponseVariableSort=sort(dataResponseVariableZ,
decreasing=FALSE, index.return=TRUE)
indSort=ResponseVariableSort$ix
dataPredictorVariablesZ_sort=dataPredictorVariablesZ[i
dataResponseVariableZ_sort=dataResponseVariableZ[indSo

#Heatmap with rows sorted by response variable and
columns sorted by LV1
dataHeatmap_P1sort=data.frame(dataPredictorVariablesZ_
ColLabels=colnames(dataHeatmap_P1sort)
pdf("Heatmap_LV1sort.pdf", width=7,height=5,pointsize
= 10)
heatmap3(dataHeatmap_P1sort,
         col=barColors,
breaks=breakBarColors,legendfun=function()showLegend(l
         Rowv=NA, Colv=NA,  scale="none",
cexCol=1,cexRow=1.1, margins=c(20,2),
         highlightCell=data.frame(rep(1:dim(dataPredic
[1],each=(dim(dataPredictorVariables)[2]+1)),rep(1:
(dim(dataPredictorVariables)
[2]+1),times=dim(dataPredictorVariables)[1]),'black'),
         labCol=ColLabels
)
dev.off()


#----LOOCV----

LOOCV_runs=200
```

```
P1_LOOCV=colMeans(Load_LV1_LOOCV)
P2_LOOCV=colMeans(Load_LV2_LOOCV)
stdevLV1_load=colSds(Load_LV1_LOOCV)/
max(abs(P1_LOOCV))
stdevLV2_load=colSds(Load_LV2_LOOCV)/
max(abs(P2_LOOCV))



#Bar plot the loadings with LOOCV error bars
P1Sort_LOOCV=sort(t(P1_LOOCV))/max(abs(P1_LOOCV))
indP1_LOOCV=sort(P1_LOOCV, index.return=TRUE)$ix
ColLabels=colnames(dataPredictorVariables)
[indP1_LOOCV]
pdf("LV1_Loadings_SD.pdf", width=10,height=5,pointsize
= 18)
barCenters <- barplot(P1Sort_LOOCV,
        main = "Signals in LV1",
        xlab = "",
        ylab = "",
        names.arg = ColLabels,
        col = "lightblue",
        horiz = FALSE,
        las=2,
        ylim=c(-1.49,1.49))
segments(barCenters,P1Sort_LOOCV-
stdevLV1_load[indP1_LOOCV],barCenters,P1Sort_LOOCV+std
arrows(barCenters,P1Sort_LOOCV-
stdevLV1_load[indP1_LOOCV],barCenters,P1Sort_LOOCV+std
dev.off()

P2Sort_LOOCV=sort(t(P2_LOOCV))/max(abs(P2_LOOCV))
indP2_LOOCV=sort(P2_LOOCV, index.return=TRUE)$ix
ColLabels=colnames(dataPredictorVariables)
[indP2_LOOCV]
pdf("LV2_Loadings_SD.pdf", width=10,height=5,pointsize
= 18)
barCenters <- barplot(P2Sort_LOOCV,
        main = "Signals in LV2",
```