# Introduction

## Cancer

You can cite one or multiple authors. One author (Kumar *et al.*, 2017) and multiple authors (Kumar *et al.*, 2017; Zavidij *et al.*, 2020). Write in **bold** or in *italic* or in both ***bolditalic***. You can also write inline code, e.g. Seurat::RunUMAP.

## LUAD

Some information Kumar *et al.* (2017)

## Computational Tools

### Gene Set Variation Analysis

### Gene Set Enrichment Analysis

### UMAP

### PCA

## Our Analysis

### Pan Cancer

### Focused Analysis

### Related Work

# Material and Methods

## TCGA data

What kind of data do we have? ## Used Packages

show a table!

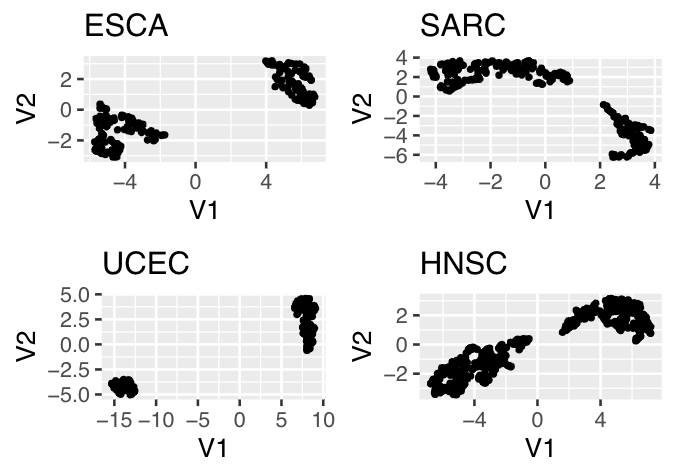
# Results

## 33 tumor types are showing disting clusters in UMAP

hello world!

## blb alsdjflaskdf umap of some tumortypes

Figure generation. You can do it with knitr or with latex formatting. This is knitr:



Title. Description

easier alternative: this is latex formatting. In Figure you can see an UMAP. (+ label your figures, equations etc and then reference with /)

# Discussion

## Immune pathways are significantly upregulated in X

# References

Kumar, SK, Rajkumar, V, Kyle, RA, Duin, M van, Sonneveld, P, Mateos, M-V, Gay, F, and Anderson, KC (2017). Multiple myeloma. Nat Rev Dis Primers 3, 17046.

Zavidij, O et al. (2020). Single-cell RNA sequencing reveals compromised immune microenvironment in precursor stages of multiple myeloma. Nat Cancer 1, 493–506.

# Appendix

## Plots

hello ## Code world

#createn einer liste mit allen patienten in dfs sortiert nach krebs  
cancers = list();cancers = vector('list',length(table(tcga\_anno$cancer\_type\_abbreviation)))  
names(cancers) = names(table(tcga\_anno$cancer\_type\_abbreviation))  
i=1  
for (i in 1:length(cancers)){  
 cancers[[i]] = tcga\_exp\_cleaned[,tcga\_anno$cancer\_type\_abbreviation == names(cancers)[i]]  
}  
#function die einen krebstypen df und genesets als input nimmt und ein df mit pvalues ausgibt  
enrichment = function(expressiondata, genesets = genesets\_ids){  
 ESmatrix = sapply(genesets, FUN = function(x){  
 ins = na.omit(match(x,rownames(expressiondata)))#indices der gene im aktuellen set  
 outs = -ins#indices der gene nicht im aktuellen set  
 #gibt einen vektor der für jeden patienten den pval für das aktuelle gene enthält  
 res = NULL  
 for (i in 1:ncol(expressiondata)){#testet für jeden patienten  
 res[i] = wilcox.test(expressiondata[ins,i],expressiondata[outs,i],'two.sided')$p.value  
 }  
 return(res)  
 })  
 row.names(ESmatrix) = colnames(expressiondata); return(ESmatrix)  
}  
pvalueslist = lapply(cancers, enrichment)#für die tests für jeden krebstypen durch

get\_top10pathways\_from\_pvalues = function(df\_p\_values, length\_genesets) {  
   
 require(ggplot2)  
   
 results <- list()  
   
 df\_p\_values\_log10 <- -log10(as.data.frame(df\_p\_values))  
   
 mean\_pathway <- as.data.frame(apply(df\_p\_values\_log10, 1, mean))  
 rownames(mean\_pathway) <- rownames(df\_p\_values\_log10)  
   
 ordered\_score <- mean\_pathway[order(-mean\_pathway[ ,1]), 1]  
 top\_10 <- data.frame(ordered\_score[1:10])  
 colnames(top\_10) <- "mean\_pathway"  
   
 ordered\_names <- order(-mean\_pathway[ ,1])  
 top\_10\_names <- ordered\_names[1:10]  
 top\_10$pathway\_names <- row.names(mean\_pathway)[top\_10\_names]  
   
 results[[1]] <- top\_10  
   
 results[[2]] <- ggplot(data = top\_10, aes(x = mean\_pathway, y = reorder(pathway\_names, mean\_pathway)))+  
 geom\_bar(stat = "identity")+  
 coord\_cartesian(xlim =c(3, 3.75))+  
 labs(title = names(df\_p\_values),  
 x = "mean p-value pathway",  
 y = "pathway name")  
   
 pathway\_size <- order(-mean\_pathway[ ,1])  
 top\_10\_size <- pathway\_size[1:10]  
 top\_10$pathway\_size <- length\_genesets[top\_10\_size]  
   
 results[[3]] <- ggplot(data = top\_10, aes(x = mean\_pathway, y = reorder(pathway\_names,  
 mean\_pathway)))+  
 geom\_point(aes(size = pathway\_size))+  
 labs(title = names(df\_p\_values),  
 x = "mean p-value pathway",  
 y = "pathway name")  
   
 return(results)  
}