

## Figure\_2

Kaitlyn Lagattuta

2024-11-18

```
suppressPackageStartupMessages(source("utils.R"))
```

```
## Warning: package 'tidyr' was built under R version 4.3.2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

## regularized Canonical Correlation Analysis results

```
xt = readRDS("data/CRtrtest_061324/CR_xtest.rds")
yt = readRDS("data/CRtrtest_061324/CR_ytest.rds")
print(load("data/CRtrtest_061324/mnsds1250.20_091324.RData"))
```

```
## [1] "mns_x" "sds_x" "mns_y" "sds_y"
```

```
xt = scale_variables(xt, mns_x, sds_x)
yt = scale_variables(yt, mns_y, sds_y)
print(load("data/CCA_0.01_1e-05_CDR2Bfix0901_tcrfilt0607_hPCscombat_combatren_ridge_hPC20_train_full_CDI
```

```
## [1] "res"      "cor"      "cor_test" "obsx"     "obsy"
```

```
xscores = as.matrix(xt) %*% as.matrix(res$loadings$X)
yscores = as.matrix(yt) %*% as.matrix(res$loadings$Y)
cor_test = sapply(1:10, function(x) cor(xscores[,x], yscores[,x]))
mat = readRDS("data/cormat_cca_perms_090924.rds")
perm_mins = sapply(1:6, function(x) min(mat[,x]))
perm_maxs = sapply(1:6, function(x) max(mat[,x]))
```

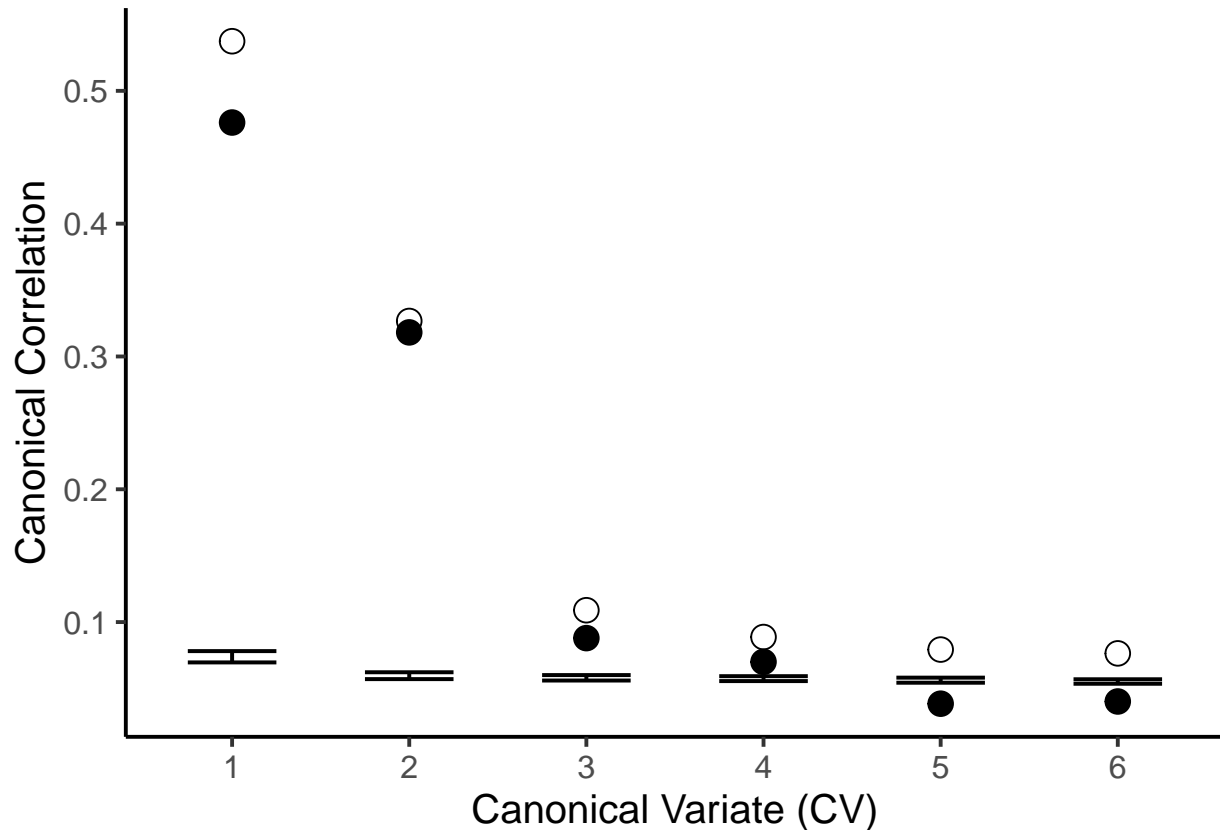
```
tp = data.frame(CV = rep(seq(1:6), 2), cor=c(res$cor[1:6], cor_test[1:6]), tt = c(rep("train", 6), rep("test", 6)))
g = ggplot()
g = g + geom_errorbar(aes(x=factor(tp$CV[tp$tt=="test"]), ymin=perm_mins[1:6], ymax=perm_maxs[1:6]), width=0.5)
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```

```

g = g + geom_point(aes(x=factor(tp$CV), y=tp$cor, shape=tp$tt), size=4, show.legend = FALSE) + theme_cl
g = g + ylab("Canonical Correlation") + scale_shape_manual(values=c(19,1))
g = g + xlab("Canonical Variate (CV)")
g

```



```

prot = readRDS("data/combat_protnorm_incNKT.rds")
exp = readRDS("data/combat_expnorm_incNKT.rds")

sref = readRDS("data/sref_combat_full_authTpplusNKT_20hPCs_tcrfilt0607_nvargenes200_sampTH1.instTH0.5.p
exp.v = exp[rownames(exp) %in% sref$vargenes$symbol,]

print(load("data/CCA_0.01_1e-05_CDR2Bfix0901_tcrfilt0607_hPCscombat_combatren_ridge_hPC20_train_full_CDI

## [1] "res"      "cor"      "cor_test" "obsx"    "obsy"

ccares = res
cells = intersect(rownames(ccares$variates$X), colnames(exp.v))
exp.v = exp.v[,as.character(cells)]
prot = prot[,as.character(cells)]
CB.variatesX = ccares$variates$X[as.character(cells),]
CB.variatesY = ccares$variates$Y[as.character(cells),]
ncv=4
ord = c("AB_TCR_Va7_2", "AB_CD161", "AB_KLRG1_MAF", "AB_CD8", "AB_CD4", "FOXP3", "RTKN2", "IKZF2", "AB
R_tp = matrix(ncol=length(ord), nrow=ncv)

```

```

P = matrix(ncol=length(ord), nrow=ncv)
for (i in 1:ncv){
  for (j in 1:length(ord)){
    if (grepl("^AB_",ord[j])){
      test = cor.test(prot[which(rownames(prot)==ord[j]),], CB.variatesY[,i])
    } else {
      test = cor.test(exp.v[which(rownames(exp.v)==ord[j]),], CB.variatesY[,i])
    }
    R_tp[i,j] = test$estimate
    P[i,j] = test$p.value
  }
  print(i)
}

```

```

## [1] 1
## [1] 2
## [1] 3
## [1] 4

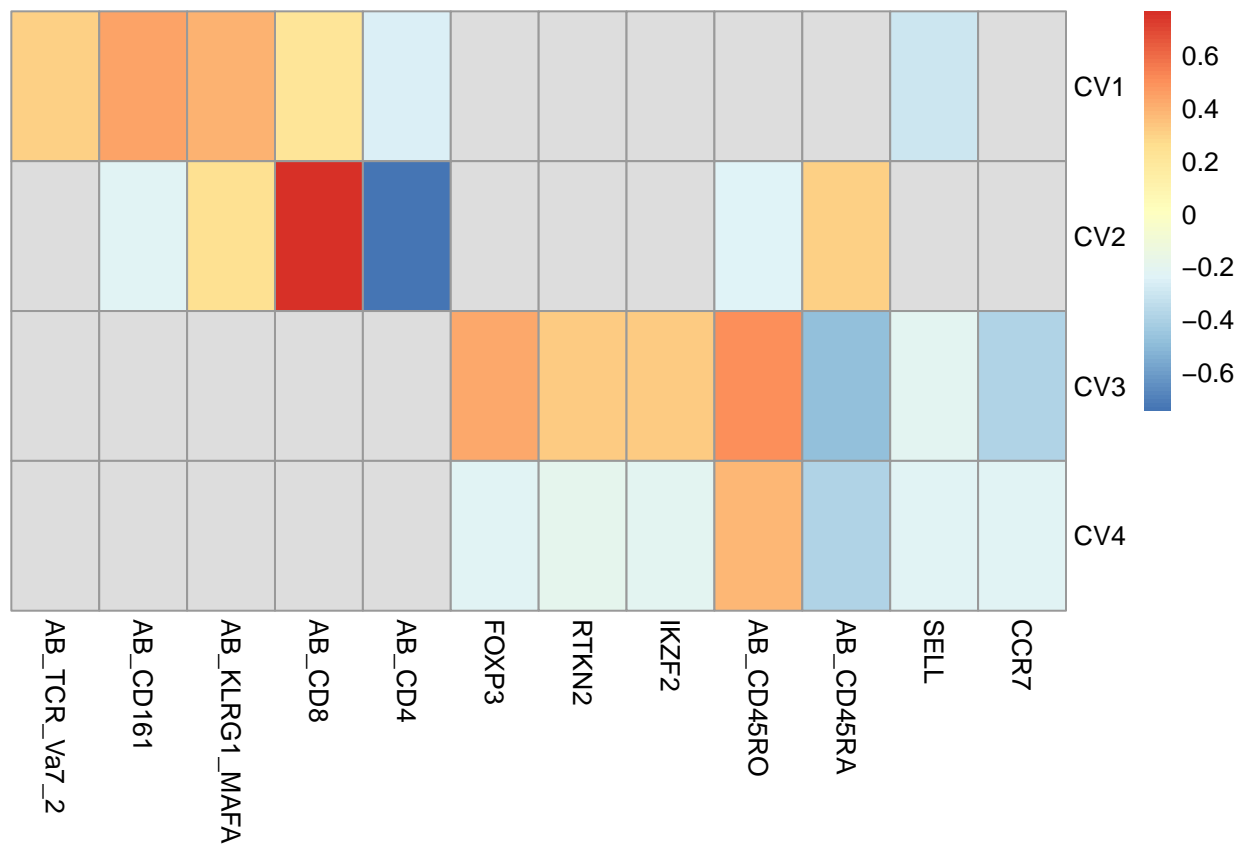
```

```

colnames(R_tp) = ord
rownames(R_tp) = paste("CV", seq(1,4), sep="")
R_tp[P>(0.05/(nrow(exp.v)+nrow(prot)))] <- NA
R_tp[abs(R_tp)<0.2] <- NA
R_tp = -R_tp
R_tp[2,] = -R_tp[2,]

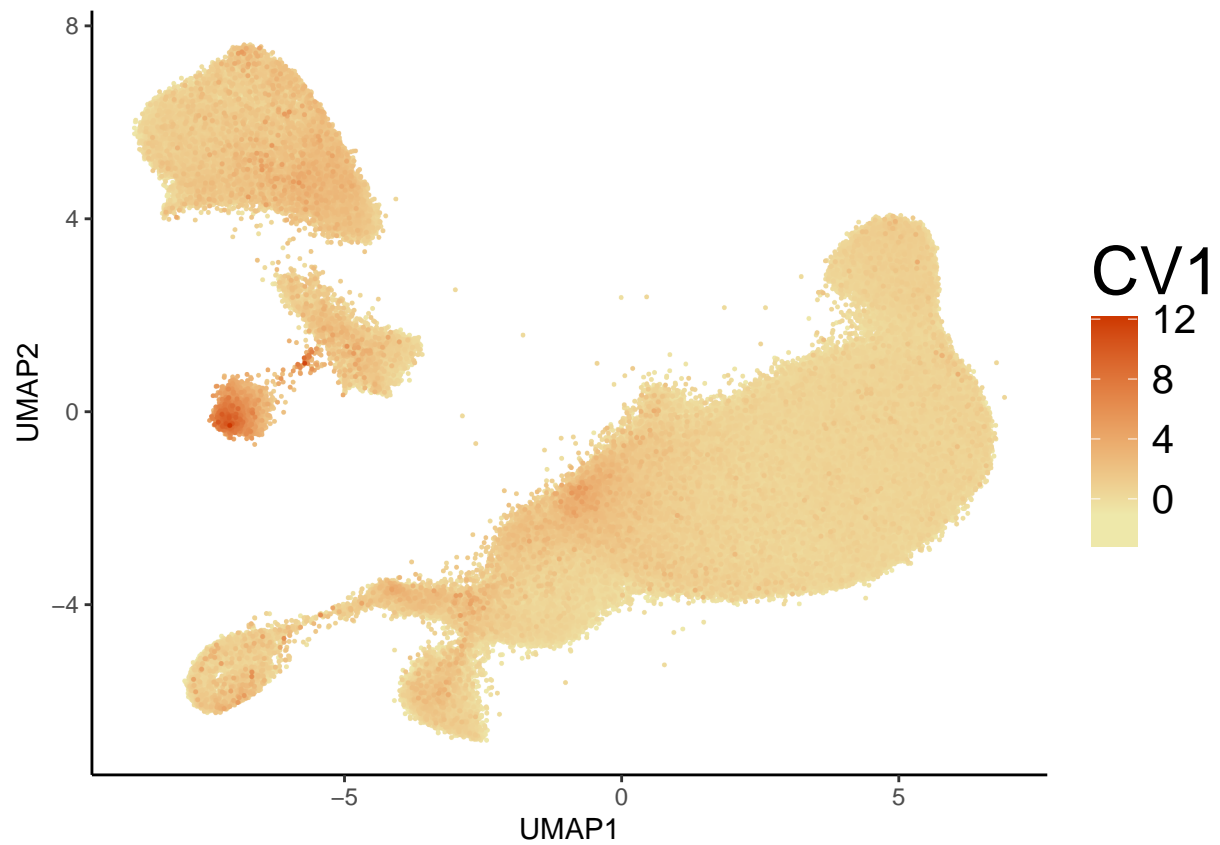
pheatmap(R_tp, cluster_rows = FALSE, cluster_cols=FALSE, heatmap_legend_param = list(
  legend_direction = "horizontal"))

```



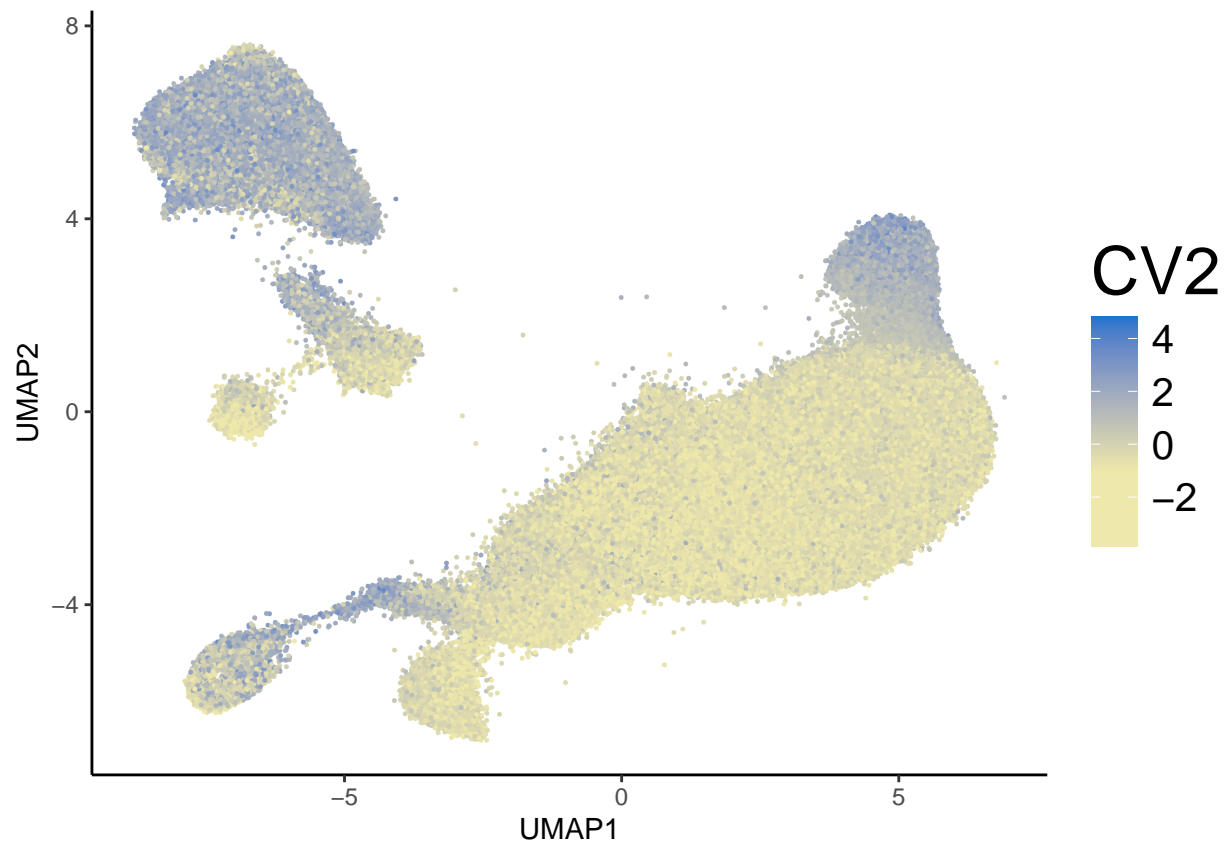
```
ren_mapped_file = "data/ren_mappedtocombat_full_authTplusNKT_500g_20hPCs_tcrfilt0607_nvargenes200_theta
get_ccascore_umap(sref, ren_mapped_file, ccares, 1, mp=-1, rev=TRUE, order=TRUE)
```

```
## Joining with 'by = join_by(cell)'
## Joining with 'by = join_by(cell)'
```



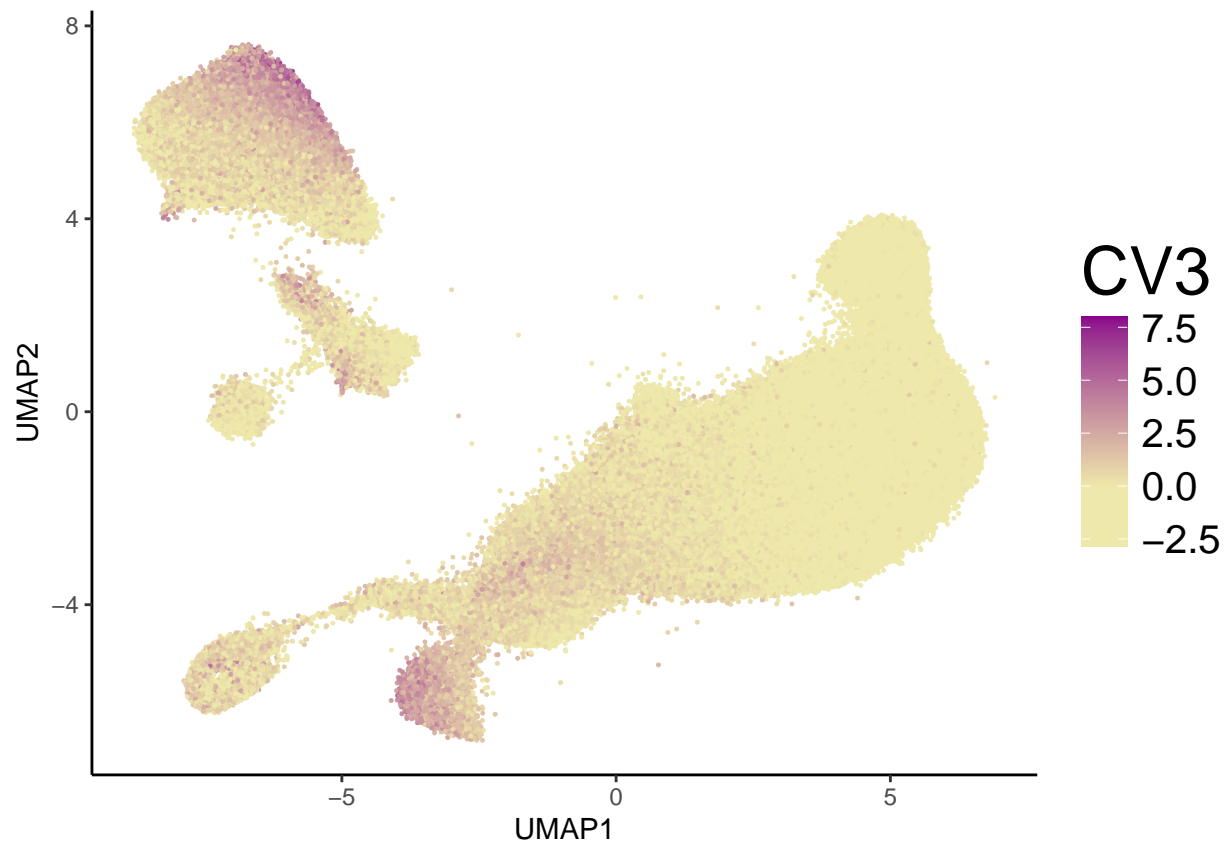
```
get_ccascore_umap(sref, ren_mapped_file, ccares, 2, mp=-1)
```

```
## Joining with 'by = join_by(cell)'  
## Joining with 'by = join_by(cell)'
```



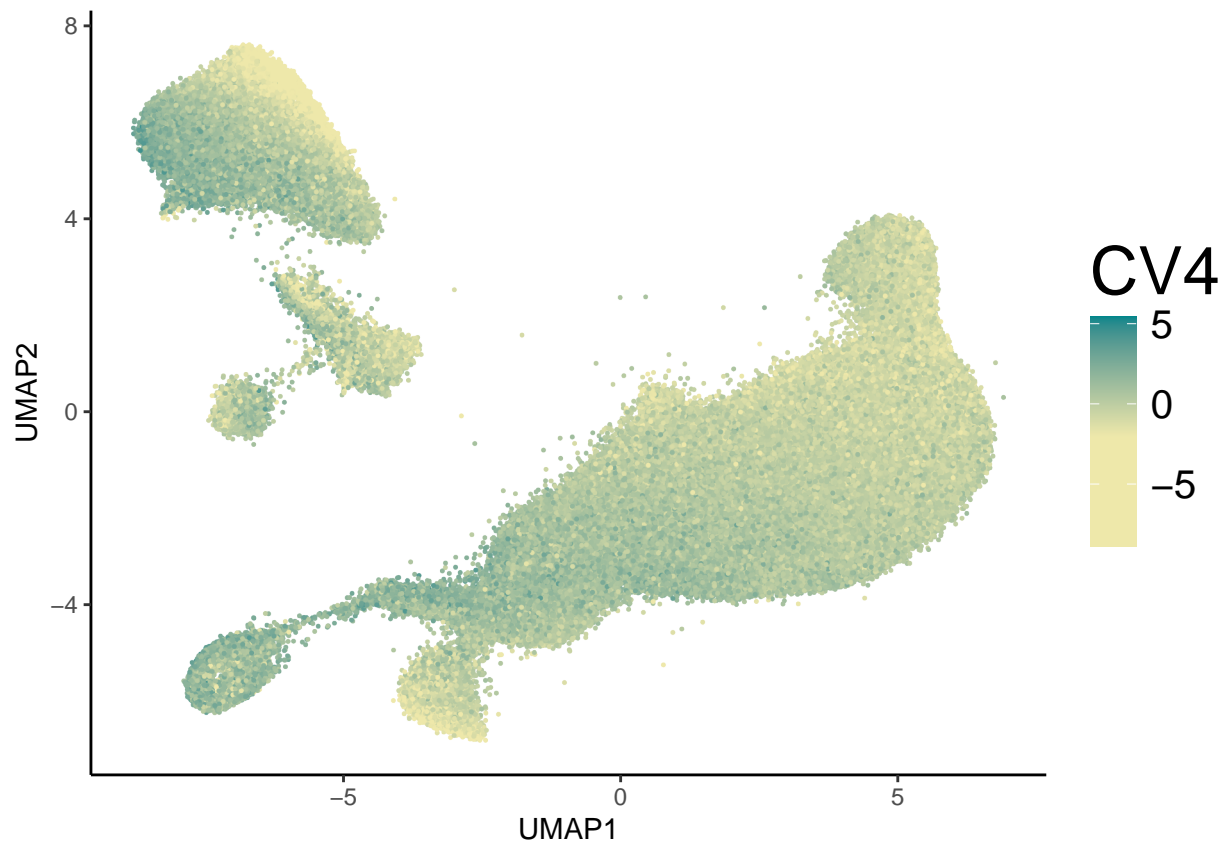
```
get_ccascore_umap(sref, ren_mapped_file, ccares, 3, mp=0, rev=TRUE)
```

```
## Joining with 'by = join_by(cell)'  
## Joining with 'by = join_by(cell)'
```



```
get_ccascore_umap(sref, ren_mapped_file, ccares, 4, mp=-2, rev=TRUE)
```

```
## Joining with 'by = join_by(cell)'  
## Joining with 'by = join_by(cell)'
```



## TCR feature contributions

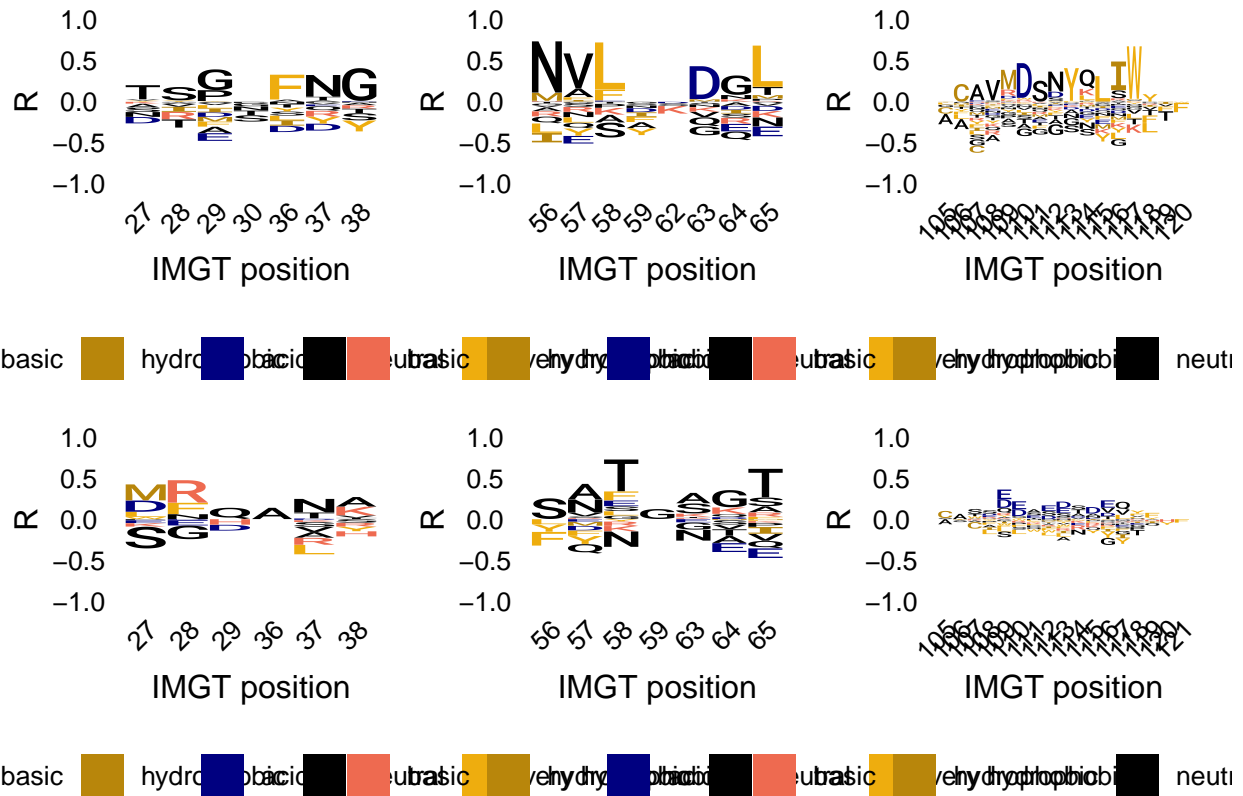
```
tcrcores = aggregate_tcr_cores_simp("data/tcrcores_linridge091824_combatren_justX")
tcrcores = tcrcores[tcrcores$p.value<0.05/nrow(tcrcores),]
```

```
TCRinnate.seqlogos = viz_all_postTCRcores(tcrcores[tcrcores$CV=="X1",], ymin=-1, ymax=1)
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

```
TCRinnate.seqlogos
```

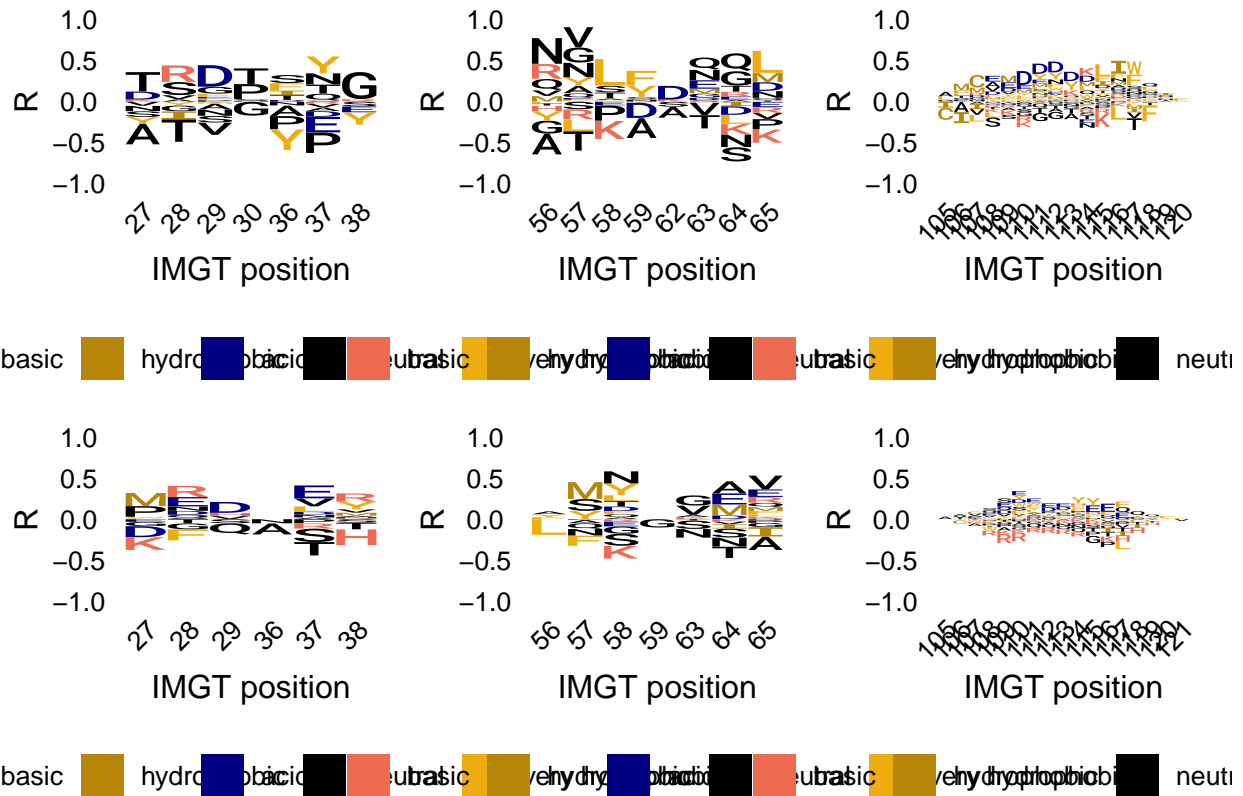




```
TCRCD8.seqlogos = viz_all_posTCRcors(tcr cors[tcr cors$CV=="X2",], ymin=-1, ymax=1)
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

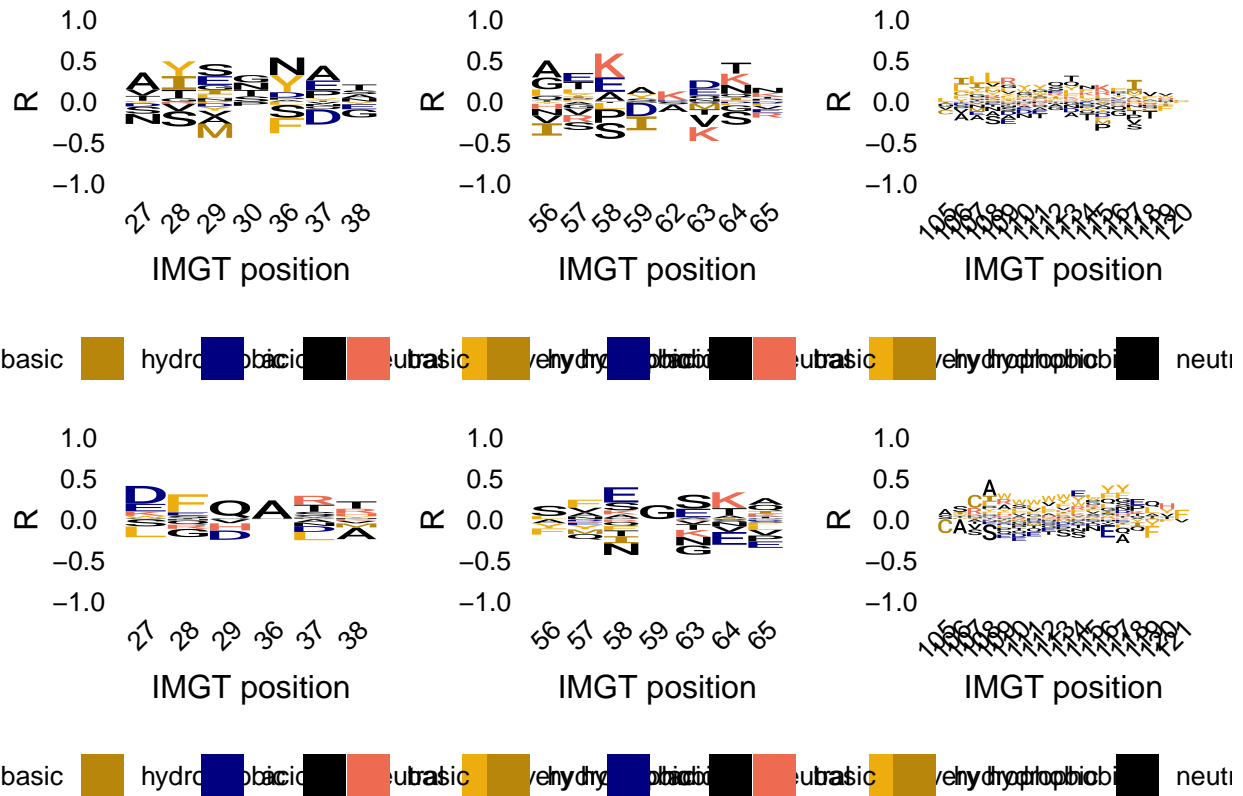
```
TCRCD8.seqlogos
```



```
TCRreg.seqlogos = viz_all_posTCRcors(tcrcores[tcrcores$CV=="X3",], ymin=-1, ymax=1)
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

```
TCRreg.seqlogos
```



```
TCRmem.seqlogos = viz_all_posTCRcors(tcrcores[tcrcores$CV=="X4",], ymin=-1, ymax=1)
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

```
TCRmem.seqlogos
```

