#Pt15 pre-Tx MLR defined alloreactivity

This part of R script is published previously, and is available in the GitHub repository at <https://github.com/Aleksobrad/Fu-J-et-al.-LGVHR-manuscript>

#Pt15 post-Tx MLR defined alloreactivity

|  |  |
| --- | --- |
| Abbreviation | Full name |
| D4U | PreTx\_Donor\_CD4\_unstim |
| D4L | PreTx\_Donor\_CD4\_CFSElow |
| D8U | PreTx\_Donor\_CD8\_unstim |
| D8L | PreTx\_Donor\_CD8\_CFSElow |
| R4U | PreTx\_Recipient\_CD4\_unstim |
| R4L | PreTx\_Recipient\_CD4\_CFSElow |
| R8U | PreTx\_Recipient\_CD8\_unstim |
| R8L | PreTx\_Recipient\_CD8\_CFSElow |
| R'4U | PostTx\_Recipient\_CD4\_unstim |
| R'4L | PostTx\_Recipient\_CD4\_CFSElow |
| R'8U | PostTx\_Recipient\_CD8\_unstim |
| R'8L | PostTx\_Recipient\_CD8\_CFSElow |

data <- read.delim("P:/CCTI\_USERS/Jianing Fu/Adaptive samples and analysis/Pt15 MVTx/Pt15 08072019.tsv")

source('P:/CCTI\_USERS/Jianing Fu/Adaptive samples and analysis/! summary of R codes/alloreactivity\_printtofile 09132017.R')

names(data)

[1] "Rearrangement" "Sum..Templates." "Present.In"

[4] "Pt15MVTx.Bx.POD17" "Pt15MVTx.Bx.POD27" "Pt15MVTx.Bx.POD55"

[7] "Pt15MVTx.GVH.D4L" "Pt15MVTx.GVH.D8L" "Pt15MVTx.GVH.R4L"

[10] "Pt15MVTx.GVH.R8L" "Pt15MVTx.PBMCs.POD11" "Pt15MVTx.PBMCs.POD143.Donor\_T"

[13] "Pt15MVTx.PBMCs.POD143.Recipient\_T" "Pt15MVTx.PBMCs.POD19" "Pt15MVTx.PBMCs.POD26"

[16] "Pt15MVTx.pre.Tx.donor\_MLN" "Pt15MVTx.unstim.D4" "Pt15MVTx.unstim.D8"

[19] "Pt15MVTx.unstim.R4" "Pt15MVTx.unstim.R8" "Pt15MVTx\_Bx\_POD237"

[22] "Pt15MVTx\_PBMC\_POD255" "Pt15\_MVTx\_Bx\_POD455" "Pt15\_MVTx\_Bx\_Tx\_colon\_POD1018"

[25] "Pt15\_MVTx\_Bx\_duodenum\_POD1018" "Pt15\_MVTx\_Bx\_ileum\_POD1018" "Pt15\_MVTx\_Bx\_native\_colon\_POD1018"

[28] "Pt15\_MVTx\_Bx\_stomach\_POD1018" "Pt15\_MVTx\_PBMC\_POD1070" "Pt15\_MVTx\_PBMC\_POD83"

[31] "Pt15\_MVTx\_PBMC\_POD83\_MLR\_D3L" "Pt15\_MVTx\_POD1336\_PBMC" "Pt15\_MVTx\_POD1336\_Tx\_colon\_Bx"

[34] "Pt15\_MVTx\_POD1336\_duodenum\_Bx" "Pt15\_MVTx\_POD1336\_ileum\_Bx\_ACR" "Pt15\_MVTx\_POD1336\_native\_colon\_Bx"

[37] "Pt15\_MVTx\_POD1336\_stomach\_Bx" "Pt15\_MVTx\_POD319\_PBMC\_MLR\_R4L\_D\_SP" "Pt15\_MVTx\_POD319\_PBMC\_MLR\_R4U"

[40] "Pt15\_MVTx\_POD319\_PBMC\_MLR\_R8L\_D\_SP" "Pt15\_MVTx\_POD319\_PBMC\_MLR\_R8U"

|  |  |
| --- | --- |
| Sample name (Pt: patient) | updated rename |
| Pt15MVTx-GVH-D4L | Pt15\_Pre\_Donor\_CD4\_CFSElo |
| Pt15MVTx-unstim-D4 spleen | Pt15\_Pre\_Donor\_CD4\_unstim |
| Pt15MVTx-GVH-D8L | Pt15\_Pre\_Donor\_CD8\_CFSElo |
| Pt15MVTx-unstim-D8 spleen | Pt15\_Pre\_Donor\_CD8\_unstim |
| Pt15MVTx-Bx-POD17 | Pt15\_Post\_ileum\_day0017 |
| Pt15MVTx-Bx-POD27 | Pt15\_Post\_ileum\_day0027 |
| Pt15MVTx-Bx-POD55 | Pt15\_Post\_ileum\_day0055 |
| Pt15MVTx-Bx-POD237 | Pt15\_Post\_ileum\_day0237 |
| Pt15MVTx-Bx-POD319 | Pt15\_Post\_ileum\_day0319 |
| Pt15MVTx-Bx-POD455 | Pt15\_Post\_ileum\_day0455 |
| Pt15\_MVTx\_Bx\_ileum\_POD1018 | Pt15\_Post\_ileum\_day1018 |
| Pt15MVTx-PBMCs-POD11 | Pt15\_Post\_PBMC\_day0011 |
| Pt15MVTx-PBMCs-POD19 | Pt15\_Post\_PBMC\_day0019 |
| Pt15MVTx-PBMCs-POD26 | Pt15\_Post\_PBMC\_day0026 |
| Pt15\_MVTx\_PBMC\_POD83 | Pt15\_Post\_PBMC\_day0083 |
| Pt15MVTx-PBMCs-POD255 | Pt15\_Post\_PBMC\_day0255 |
| Pt15MVTx-PBMCs-POD1070 | Pt15\_Post\_PBMC\_day1070 |
| Pt15MVTx-HVG-R4L | Pt15\_Pre\_Recipient\_CD4\_CFSElo |
| Pt15MVTx-unstim-R4 | Pt15\_Pre\_Recipient\_CD4\_unstim |
| Pt15MVTx-HVG-R8L | Pt15\_Pre\_Recipient\_CD8\_CFSElo |
| Pt15MVTx-unstim-R8 | Pt15\_Pre\_Recipient\_CD8\_unstim |

# resolve CD4 and CD8 ambiguous clones first (ratio=5), then followed by resolve donor and recipient ambiguous clones.

#D4U vs D8U

normalize(data[,c(17,18)])

ratio=5

c1indices=which(data[,17]>0 & data[,18]>0 & data[,17]>ratio\*data[,18])

c2indices=which(data[,17]>0 & data[,18]>0 & data[,18]>ratio\*data[,17])

ambiindices=which(data[,17]>0 & data[,18]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

data[c1indices,18]=0

data[c2indices,17]=0

data[ambiindices,c(17,18)]=0

#D4L vs D8L

normalize(data[,c(7,8)])

ratio=5

c1indices=which(data[,7]>0 & data[,8]>0 & data[,7]>ratio\*data[,8])

c2indices=which(data[,7]>0 & data[,8]>0 & data[,8]>ratio\*data[,7])

ambiindices=which(data[,7]>0 & data[,8]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

data[c1indices,8]=0

data[c2indices,7]=0

data[ambiindices,c(7,8)]=0

#R4U vs R8U

normalize(data[,c(19,20)])

ratio=5

c1indices=which(data[,19]>0 & data[,20]>0 & data[,19]>ratio\*data[,20])

c2indices=which(data[,19]>0 & data[,20]>0 & data[,20]>ratio\*data[,19])

ambiindices=which(data[,19]>0 & data[,20]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

data[c1indices,20]=0

data[c2indices,19]=0

data[ambiindices,c(19,20)]=0

#R4L vs R8L

normalize(data[,c(9,10)])

ratio=5

c1indices=which(data[,9]>0 & data[,10]>0 & data[,9]>ratio\*data[,10])

c2indices=which(data[,9]>0 & data[,10]>0 & data[,10]>ratio\*data[,9])

ambiindices=which(data[,9]>0 & data[,10]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

data[c1indices,10]=0

data[c2indices,9]=0

data[ambiindices,c(9,10)]=0

rownames(data)=data[,1]

x=data

#remove donor/recipient ambiguous clones, either real shared clones, or ambiguous raised by CFSE-MLR sorting error

ambiguous=which((x[,17]>0 | x[,18]>0)&(x[,19]>0 | x[,20]>0))

ambiguous=union(ambiguous,which((x[,17]>0 | x[,18]>0)&(x[,9]>0 | x[,10]>0)))

ambiguous=union(ambiguous,which((x[,19]>0 | x[,20]>0)&(x[,7]>0 | x[,8]>0)))

x=x[setdiff(1:nrow(x),ambiguous),]

#additionally remove stim vs unstim ambiguous within recipient preTx (R4L, R4U, R8L, R8U) or donor preTx (D4L, D4U, D8L, D8U) samples; which can be considered as additional step of optimization for CD4 and CD8 sorting error.

#R4L vs R8U

normalize(x[,c(9,20)])

ratio=2

c1indices=which(x[,9]>0 & x[,20]>0 & x[,9]>ratio\*x[,20])

c2indices=which(x[,9]>0 & x[,20]>0 & x[,20]>ratio\*x[,9])

ambiindices=which(x[,9]>0 & x[,20]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

x[c1indices,20]=0

x[c2indices,9]=0

x[ambiindices,c(9,20)]=0

#R8L vs R4U

normalize(x[,c(10,19)])

ratio=2

c1indices=which(x[,10]>0 & x[,19]>0 & x[,10]>ratio\*x[,19])

c2indices=which(x[,10]>0 & x[,19]>0 & x[,19]>ratio\*x[,10])

ambiindices=which(x[,10]>0 & x[,19]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

x[c1indices,19]=0

x[c2indices,10]=0

x[ambiindices,c(10,19)]=0

#D4L vs D8U

normalize(x[,c(7,18)])

ratio=2

c1indices=which(x[,7]>0 & x[,18]>0 & x[,7]>ratio\*x[,18])

c2indices=which(x[,7]>0 & x[,18]>0 & x[,18]>ratio\*x[,7])

ambiindices=which(x[,7]>0 & x[,18]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

x[c1indices,18]=0

x[c2indices,7]=0

x[ambiindices,c(7,18)]=0

#D8L vs D4U

normalize(x[,c(8,17)])

ratio=2

c1indices=which(x[,8]>0 & x[,17]>0 & x[,8]>ratio\*x[,17])

c2indices=which(x[,8]>0 & x[,17]>0 & x[,17]>ratio\*x[,8])

ambiindices=which(x[,8]>0 & x[,17]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

x[c1indices,17]=0

x[c2indices,8]=0

x[ambiindices,c(8,17)]=0

#resolve ambiguous raised by post-Tx unstim and CFSE-MLR

#post-Tx R4U vs R8U

normalize(x[,c(39,41)])

ratio=5

c1indices=which(x[,39]>0 & x[,41]>0 & x[,39]>ratio\*x[,41])

c2indices=which(x[,39]>0 & x[,41]>0 & x[,41]>ratio\*x[,39])

ambiindices=which(x[,39]>0 & x[,41]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

x[c1indices,41]=0

x[c2indices,39]=0

x[ambiindices,c(39,41)]=0

#post-Tx R4L vs R8L

normalize(x[,c(38,40)])

ratio=5

c1indices=which(x[,38]>0 & x[,40]>0 & x[,38]>ratio\*x[,40])

c2indices=which(x[,38]>0 & x[,40]>0 & x[,40]>ratio\*x[,38])

ambiindices=which(x[,38]>0 & x[,40]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

x[c1indices,40]=0

x[c2indices,38]=0

x[ambiindices,c(38,40)]=0

#remove donor/recipient ambiguous clones, either real shared clones, or ambiguous raised by CFSE-MLR sorting error

ambiguous=which((x[,17]>0 | x[,18]>0)&(x[,39]>0 | x[,41]>0))

ambiguous=union(ambiguous,which((x[,17]>0 | x[,18]>0)&(x[,38]>0 | x[,40]>0)))

ambiguous=union(ambiguous,which((x[,39]>0 | x[,41]>0)&(x[,7]>0 | x[,8]>0)))

x=x[setdiff(1:nrow(x),ambiguous),]

#additionally remove stim vs unstim ambiguous within recipient postTx (R’4L, R’4U, R’8L, R’8U); which can be considered as additional step of optimization for CD4 and CD8 sorting error.

#R4L vs R8U

normalize(x[,c(38,41)])

ratio=2

c1indices=which(x[,38]>0 & x[,41]>0 & x[,38]>ratio\*x[,41])

c2indices=which(x[,38]>0 & x[,41]>0 & x[,41]>ratio\*x[,38])

ambiindices=which(x[,38]>0 & x[,41]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

x[c1indices,41]=0

x[c2indices,38]=0

x[ambiindices,c(38,41)]=0

#R8L vs R4U

normalize(x[,c(40,39)])

ratio=2

c1indices=which(x[,40]>0 & x[,39]>0 & x[,40]>ratio\*x[,39])

c2indices=which(x[,40]>0 & x[,39]>0 & x[,39]>ratio\*x[,40])

ambiindices=which(x[,40]>0 & x[,39]>0)

ambiindices=setdiff(setdiff(ambiindices,c1indices),c2indices)

x[c1indices,39]=0

x[c2indices,40]=0

x[ambiindices,c(40,39)]=0

write.table(x,file ="P:/CCTI\_USERS/Jianing Fu/Adaptive samples and analysis/Pt15 MVTx/Pt15 08072019 resolve ambiguous updated with post-Tx MLR.tsv",quote=F,row.names=F,col.names=T, sep="\t")

# add "fold=2" below, use 2 fold expansion (default 5 fold)

# add "freq=0.00002" below, use freq=0.00002 for template counts; use default freq=0.00001 for read counts

#post-Tx POD319 HVG direction

# For CD4 HvG, cd4.HVG=x[,c(unstim(R4U),stim(R4L))]

# For CD8 HvG, cd8.HVG=x[,c(unstim(R8U),stim(R8L))]

cd4.HVG= x[,c(39,38)]

cd8.HVG= x[,c(41,40)]

allo.HVG=listAlloreactive(cd4.HVG,cd8.HVG, fold=2, freq=0.00002)

length(allo.HVG[[1]])

[1] 176

length(allo.HVG[[2]])

[1] 55

write.table(allo.HVG[[1]],file ="P:/CCTI\_USERS/Jianing Fu/Adaptive samples and analysis/Pt15 MVTx/Pt15 CD4 post-Tx POD319 HVGlist.txt",quote=F,row.names=F,col.names=F, sep="\t")

write.table(allo.HVG[[2]],file ="P:/CCTI\_USERS/Jianing Fu/Adaptive samples and analysis/Pt15 MVTx/Pt15 CD8 post-Tx POD319 HVGlist.txt",quote=F,row.names=F,col.names=F, sep="\t")

rCD4mappable=rownames(x[(x[,39]+x[,38])>0,])

rCD8mappable=rownames(x[(x[,41]+x[,40])>0,])

CD4nonHVG=setdiff(rCD4mappable,allo.HVG[[1]])

CD8nonHVG=setdiff(rCD8mappable,allo.HVG[[2]])

write.table(CD4nonHVG,file ="P:/CCTI\_USERS/Jianing Fu/Adaptive samples and analysis/Pt15 MVTx/Pt15 CD4nonHVG rmappable list post-Tx POD319.txt",quote=F,row.names=F,col.names=F, sep="\t")

write.table(CD8nonHVG,file ="P:/CCTI\_USERS/Jianing Fu/Adaptive samples and analysis/Pt15 MVTx/Pt15 CD8nonHVG rmappable list post-Tx POD319.txt",quote=F,row.names=F,col.names=F, sep="\t")

length(rCD4mappable)

[1] 25436

length(rCD8mappable)

[1] 8065

length(CD4nonHVG)

[1] 25260

length(CD8nonHVG)

[1] 8010

rmappable=rownames(x[(x[,39]+x[,38]+x[,41]+x[,40])>0,])

length(rmappable)

[1] 33501

write.table(rmappable,file ="P:/CCTI\_USERS/Jianing Fu/Adaptive samples and analysis/Pt15 MVTx/Pt15 recipient mappable list post-Tx POD319.txt",quote=F,row.names=F,col.names=F, sep="\t")

length(rCD4mappable)+length(rCD8mappable)

[1] 33501

length(intersect(rCD4mappable,rCD8mappable))

[1] 0

#Pt15 integrate pre- and post-Tx MLRs

#load resolve ambiguous tsv

Pt15\_new <- read.delim("~/Desktop/10x TRM 2021/pre post MLR analysis/Pt15 08072019 resolve ambiguous updated with post-Tx MLR.tsv")

colnames(Pt15\_new)

[1] "Rearrangement" "Sum..Templates." "Present.In"

[4] "Pt15MVTx.Bx.POD17" "Pt15MVTx.Bx.POD27" "Pt15MVTx.Bx.POD55"

[7] "Pt15MVTx.GVH.D4L" "Pt15MVTx.GVH.D8L" "Pt15MVTx.HVG.R4L"

[10] "Pt15MVTx.HVG.R8L" "Pt15MVTx.PBMCs.POD11" "Pt15MVTx.PBMCs.POD143.Donor\_T"

[13] "Pt15MVTx.PBMCs.POD143.Recipient\_T" "Pt15MVTx.PBMCs.POD19" "Pt15MVTx.PBMCs.POD26"

[16] "Pt15MVTx.pre.Tx.donor\_MLN" "Pt15MVTx.unstim.D4" "Pt15MVTx.unstim.D8"

[19] "Pt15MVTx.unstim.R4" "Pt15MVTx.unstim.R8" "Pt15MVTx\_Bx\_POD237"

[22] "Pt15MVTx\_PBMC\_POD255" "Pt15\_MVTx\_Bx\_POD455" "Pt15\_MVTx\_Bx\_Tx\_colon\_POD1018"

[25] "Pt15\_MVTx\_Bx\_duodenum\_POD1018" "Pt15\_MVTx\_Bx\_ileum\_POD1018" "Pt15\_MVTx\_Bx\_native\_colon\_POD1018"

[28] "Pt15\_MVTx\_Bx\_stomach\_POD1018" "Pt15\_MVTx\_PBMC\_POD1070" "Pt15\_MVTx\_PBMC\_POD83"

[31] "Pt15\_MVTx\_PBMC\_POD83\_MLR\_D3L" "Pt15\_MVTx\_POD1336\_PBMC" "Pt15\_MVTx\_POD1336\_Tx\_colon\_Bx"

[34] "Pt15\_MVTx\_POD1336\_duodenum\_Bx" "Pt15\_MVTx\_POD1336\_ileum\_Bx\_ACR" "Pt15\_MVTx\_POD1336\_native\_colon\_Bx"

[37] "Pt15\_MVTx\_POD1336\_stomach\_Bx" "Pt15\_MVTx\_POD319\_PBMC\_MLR\_R4L\_D\_SP" "Pt15\_MVTx\_POD319\_PBMC\_MLR\_R4U"

[40] "Pt15\_MVTx\_POD319\_PBMC\_MLR\_R8L\_D\_SP" "Pt15\_MVTx\_POD319\_PBMC\_MLR\_R8U"

|  |  |
| --- | --- |
| Sample name (Pt: patient) | updated rename |
| Pt15MVTx-GVH-D4L | Pt15\_Pre\_Donor\_CD4\_CFSElo |
| Pt15MVTx-unstim-D4 spleen | Pt15\_Pre\_Donor\_CD4\_unstim |
| Pt15MVTx-GVH-D8L | Pt15\_Pre\_Donor\_CD8\_CFSElo |
| Pt15MVTx-unstim-D8 spleen | Pt15\_Pre\_Donor\_CD8\_unstim |
| Pt15MVTx-Bx-POD17 | Pt15\_Post\_ileum\_day0017 |
| Pt15MVTx-Bx-POD27 | Pt15\_Post\_ileum\_day0027 |
| Pt15MVTx-Bx-POD55 | Pt15\_Post\_ileum\_day0055 |
| Pt15MVTx-Bx-POD237 | Pt15\_Post\_ileum\_day0237 |
| Pt15MVTx-Bx-POD319 | Pt15\_Post\_ileum\_day0319 |
| Pt15MVTx-Bx-POD455 | Pt15\_Post\_ileum\_day0455 |
| Pt15\_MVTx\_Bx\_ileum\_POD1018 | Pt15\_Post\_ileum\_day1018 |
| Pt15MVTx-PBMCs-POD11 | Pt15\_Post\_PBMC\_day0011 |
| Pt15MVTx-PBMCs-POD19 | Pt15\_Post\_PBMC\_day0019 |
| Pt15MVTx-PBMCs-POD26 | Pt15\_Post\_PBMC\_day0026 |
| Pt15\_MVTx\_PBMC\_POD83 | Pt15\_Post\_PBMC\_day0083 |
| Pt15MVTx-PBMCs-POD255 | Pt15\_Post\_PBMC\_day0255 |
| Pt15MVTx-PBMCs-POD1070 | Pt15\_Post\_PBMC\_day1070 |
| Pt15MVTx-HVG-R4L | Pt15\_Pre\_Recipient\_CD4\_CFSElo |
| Pt15MVTx-unstim-R4 | Pt15\_Pre\_Recipient\_CD4\_unstim |
| Pt15MVTx-HVG-R8L | Pt15\_Pre\_Recipient\_CD8\_CFSElo |
| Pt15MVTx-unstim-R8 | Pt15\_Pre\_Recipient\_CD8\_unstim |

#unstim, stim

rcd4 <- Pt15\_new[,c(19,9)]

rcd4 <- normalize(rcd4)

rCD4HVG <- rownames(rcd4[rcd4[,2]>0.00002 & rcd4[,2] > rcd4[,1]\*2,])

rCD4NonHVG <- setdiff(rownames(Pt15\_new[Pt15\_new[,19]>0|Pt15\_new[,9]>0,]), rCD4HVG)

rcd8 <- Pt15\_new[,c(20,10)]

rcd8 <- normalize(rcd8)

rCD8HVG <- rownames(rcd8[rcd8[,2]>0.00002 & rcd8[,2] > rcd8[,1]\*2,])

rCD8NonHVG <- setdiff(rownames(Pt15\_new[Pt15\_new[,20]>0|Pt15\_new[,10]>0,]), rCD8HVG)

dcd4 <- Pt15\_new[,c(17,7)]

dcd4 <- normalize(dcd4)

rCD4GVH <- rownames(dcd4[dcd4[,2]>0.00002 & dcd4[,2] > dcd4[,1]\*2,])

rCD4NonGVH <- setdiff(rownames(Pt15\_new[Pt15\_new[,17]>0|Pt15\_new[,7]>0,]), rCD4GVH)

dcd8 <- Pt15\_new[,c(18,8)]

dcd8 <- normalize(dcd8)

rCD8GVH <- rownames(dcd8[dcd8[,2]>0.00002 & dcd8[,2] > dcd8[,1]\*2,])

rCD8NonGVH <- setdiff(rownames(Pt15\_new[Pt15\_new[,18]>0|Pt15\_new[,8]>0,]), rCD8GVH)

r2cd4 <- Pt15\_new[,c(39,38)]

r2cd4 <- normalize(r2cd4)

r2CD4HVG <- rownames(r2cd4[r2cd4[,2]>0.00002 & r2cd4[,2] > r2cd4[,1]\*2,])

r2CD4NonHVG <- setdiff(rownames(Pt15\_new[Pt15\_new[,39]>0|Pt15\_new[,38]>0,]), r2CD4HVG)

r2cd8 <- Pt15\_new[,c(41,40)]

r2cd8 <- normalize(r2cd8)

r2CD8HVG <- rownames(r2cd8[r2cd8[,2]>0.00002 & r2cd8[,2] > r2cd8[,1]\*2,])

r2CD8NonHVG <- setdiff(rownames(Pt15\_new[Pt15\_new[,41]>0|Pt15\_new[,40]>0,]), r2CD8HVG)

CD4HVG <- setdiff(rCD4HVG,c(rCD4NonHVG,rCD4GVH,rCD4NonGVH,rCD8HVG,rCD8NonHVG,rCD8GVH,rCD8NonGVH,r2CD8HVG,r2CD8NonHVG))

CD4NonHVG <- setdiff(rCD4NonHVG,c(rCD4HVG,rCD4GVH,rCD4NonGVH,rCD8HVG,rCD8NonHVG,rCD8GVH,rCD8NonGVH,r2CD8HVG,r2CD8NonHVG))

CD8HVG <- setdiff(rCD8HVG,c(rCD4NonHVG,rCD4GVH,rCD4NonGVH,rCD4HVG,rCD8NonHVG,rCD8GVH,rCD8NonGVH,r2CD4HVG,r2CD4NonHVG))

CD8NonHVG <- setdiff(rCD8NonHVG,c(rCD4HVG,rCD4GVH,rCD4NonGVH,rCD8HVG,rCD4NonHVG,rCD8GVH,rCD8NonGVH,r2CD4HVG,r2CD4NonHVG))

unmappable <- setdiff(rownames(Pt15\_new), c(rCD4HVG,rCD4NonHVG,rCD4GVH,rCD4NonGVH,rCD8HVG,rCD8NonHVG,rCD8GVH,rCD8NonGVH))

CD4H2VG <- setdiff(r2CD4HVG,c(r2CD4NonHVG,r2CD8HVG,r2CD8NonHVG, rCD8HVG,rCD8NonHVG))

CD4NonH2VG <- setdiff(r2CD4NonHVG,c(r2CD4HVG,r2CD8HVG,r2CD8NonHVG, rCD8HVG,rCD8NonHVG))

CD8H2VG <- setdiff(r2CD8HVG,c(r2CD4NonHVG,r2CD4HVG,r2CD8NonHVG, rCD4HVG,rCD4NonHVG))

CD8NonH2VG <- setdiff(r2CD8NonHVG,c(r2CD4HVG,r2CD8HVG,r2CD4NonHVG, rCD4HVG,rCD4NonHVG))

unmappable2 <- setdiff(rownames(Pt15\_new), c(r2CD4HVG,r2CD4NonHVG,r2CD8HVG,r2CD8NonHVG))

CD4presistentHVG <- intersect(CD4HVG, CD4H2VG)

CD4acquiredHVG <- intersect(CD4NonHVG, CD4H2VG)

CD4denovoHVG <- intersect(unmappable, CD4H2VG)

CD4tolerantHVG <- intersect(CD4HVG, CD4NonH2VG)

CD4persistentNonHVG <- intersect(CD4NonHVG, CD4NonH2VG)

CD4denovoNonHVG <- intersect(unmappable, CD4NonH2VG)

CD4missingHVG <- intersect(CD4HVG, unmappable2)

CD4missingNonHVG <- intersect(CD4NonHVG, unmappable2)

Unmappable <- intersect(unmappable, unmappable2)

CD8presistentHVG <- intersect(CD8HVG, CD8H2VG)

CD8acquiredHVG <- intersect(CD8NonHVG, CD8H2VG)

CD8denovoHVG <- intersect(unmappable, CD8H2VG)

CD8tolerantHVG <- intersect(CD8HVG, CD8NonH2VG)

CD8persistentNonHVG <- intersect(CD8NonHVG, CD8NonH2VG)

CD8denovoNonHVG <- intersect(unmappable, CD8NonH2VG)

CD8missingHVG <- intersect(CD8HVG, unmappable2)

CD8missingNonHVG <- intersect(CD8NonHVG, unmappable2)

matrix1 <- as.data.frame(matrix(nrow=17,ncol=17))

rownames(matrix1) <- c("CD4presistentHVG","CD4acquiredHVG","CD4denovoHVG",

"CD4tolerantHVG","CD4persistentNonHVG","CD4denovoNonHVG",

"CD4missingHVG","CD4missingNonHVG","Unmappable",

"CD8presistentHVG","CD8acquiredHVG","CD8denovoHVG",

"CD8tolerantHVG","CD8persistentNonHVG","CD8denovoNonHVG",

"CD8missingHVG","CD8missingNonHVG") -> colnames(matrix1)

subsets <- list(CD4presistentHVG,CD4acquiredHVG,CD4denovoHVG,

CD4tolerantHVG,CD4persistentNonHVG,CD4denovoNonHVG,

CD4missingHVG,CD4missingNonHVG,Unmappable,

CD8presistentHVG,CD8acquiredHVG,CD8denovoHVG,

CD8tolerantHVG,CD8persistentNonHVG,CD8denovoNonHVG,

CD8missingHVG,CD8missingNonHVG)

for (m in 1:17){

for (n in 1:17) {

matrix1[m,n] <- length(intersect(subsets[[m]], subsets[[n]]))

}

}

write.table(matrix1, file ="~/Desktop/10x TRM 2021/pre post MLR analysis/Pt15 17 subsets uniqueclonenubmer\_JF.tsv",row.names=T,col.names=T, sep="\t")

matrix2 <- as.data.frame(matrix(nrow=17\*4,ncol=(ncol(Pt15\_new)-2)))

rownames(matrix2) <- paste0(rep(c("CD4presistentHVG","CD4acquiredHVG","CD4denovoHVG",

"CD4tolerantHVG","CD4persistentNonHVG","CD4denovoNonHVG",

"CD4missingHVG","CD4missingNonHVG","Unmappable",

"CD8presistentHVG","CD8acquiredHVG","CD8denovoHVG",

"CD8tolerantHVG","CD8persistentNonHVG","CD8denovoNonHVG",

"CD8missingHVG","CD8missingNonHVG"),4), c(rep("template#", 17), rep("template%", 17), rep("uniqueclone#", 17), rep("uniqueclone%", 17)))

colnames(matrix2) <- colnames(Pt15\_new[,4:ncol(Pt15\_new)])

length1 <- function(x){

return(length(which(x>0)))

}

#Pt15\_new\_v2 only choose defined 17 subsets in the demoninator when calculating %, which is not what we want, use total template counts instead.

Pt15\_new\_v2 <- Pt15\_new

Pt15\_new\_v2 <- Pt15\_new\_v2[rownames(Pt15\_new\_v2) %in% c(CD4presistentHVG,CD4acquiredHVG,CD4denovoHVG,

CD4tolerantHVG,CD4persistentNonHVG,CD4denovoNonHVG,

CD4missingHVG,CD4missingNonHVG,Unmappable,

CD8presistentHVG,CD8acquiredHVG,CD8denovoHVG,

CD8tolerantHVG,CD8persistentNonHVG,CD8denovoNonHVG,

CD8missingHVG,CD8missingNonHVG), ]

for (m in 1:17) {

for (n in 4:ncol(Pt15\_new\_v2)) {

matrix2[m,n-2] <- sum(Pt15\_new\_v2[subsets[[m]], n])

matrix2[m+17,n-2] <- sum(Pt15\_new\_v2[subsets[[m]], n])/sum(Pt15\_new\_v2[,n])\*100

matrix2[m+17\*2,n-2] <- length1(Pt15\_new\_v2[subsets[[m]], n])

matrix2[m+17\*3,n-2] <- length1(Pt15\_new\_v2[subsets[[m]], n])/length1(Pt15\_new\_v2[, n])\*100

}

}

write.table(matrix2, file ="~/Desktop/10x TRM 2021/pre post MLR analysis/Pt15 17 subsets number% in uniqueclone and template\_JF.tsv",row.names=T,col.names=T, sep="\t")

### matrix first row: column names need slightly adjust. calculation takes about ~10-15mins

#total template counts

apply(Pt15\_new[,c(4:41)],2,sum)

#generate data matrix

matrix<-apply(Pt15\_new[,c(4:41)],2, sum)

write.table(matrix,file ="~/Desktop/10x TRM 2021/pre post MLR analysis/Pt15 total template counts.tsv",quote=F,row.names=T,col.names=T, sep="\t")