A comprehensive analysis of Transcribed Ultra Conserved Regions uncovers important regulatory functions of novel non-coding transcripts in gliomas.

Myron K Gibert Jr and Roger Abounader

January 27, 2020

Contents

Corresponding author: Roger Abounader, University of Virginia, PO Box 800168, Charlottesville VA 22908, USA, Phone: (434) 982-6634, Fax: 434-243-6843 E-mail: ra6u@virginia.edu.

ABSTRACT

Transcribed Ultra-Conserved Regions (TUCRs) represent a severely understudied class of putative non-coding RNAs (ncRNAs) that are 100% conserved across multiple species. We performed the first-ever analysis of TUCRs in glioblastoma (GBM) and low-grade gliomas (LGG). We leveraged large human datasets to identify the genomic locations, chromatin accessibility, transcription, differential expression, correlation with survival, and predicted functions of all 481 TUCRs, and identified TUCRs that are relevant to glioma biology. Of these, we investigated the expression, function, and mechanism of action of the most highly upregulated intergenic TUCR, uc.110, identifying it as a new tumor enhancer. Uc.110 was highly overexpressed in GBM and LGG, where it promoted malignancy and tumor growth. Uc.110 activated the WNT pathway by upregulating the expression of membrane frizzled-related protein (MFRP), by sponging the tumor suppressor microRNA miR-544. This pioneering study shows important roles for TUCRs in gliomas and provides an extensive database and novel methods for future TUCR research.

INTRODUCTION

Transcribed Ultra-conserved Regions (TUCRs) represent 481 unique transcribed RNA molecules that are "ultraconserved" across multiple species, including in the human, mouse (100%), rat (100%), dog (98%), and chicken (95%) genomes. [1] TUCR expression has been found to be highly deregulated in some cancers. Because of their ultra-conservation and their deregulation, it is believed that TUCRs may have important regulatory roles in cancer. [2-11] About ~90% of the genome is transcribed, but only ~2 percent of the transcriptome is translated. [2] The remainder of the transcriptome is made up of non-coding elements that serve key regulatory roles. Of these elements, long non-coding RNAs (lncRNAs) serve as important regulators of malignancy and potential therapeutic targets in cancer. [2, 12-19] Due to their size and lack of known associated protein products, it has been suggested that many TUCRs may function as lncRNAs, while those within protein coding genes may represent ultraconserved regions of the coding gene without being independent transcripts themselves. For example, many homeobox (HOX) genes contain ultraconserved regions.[2] And yet, the putative existence of "ultraconserved" lncRNAs is particularly significant, as lncRNAs are typically poorly conserved as a class of molecules.[2] Very little is known about TUCRs. [2] In particular, the literature elucidating the expressions, functions, and mechanisms of action of TUCRs in glioblastoma (GBM) and low-grade glioma (LGG) is nonexistent. GBM and LGG represent over 80% of primary malignant brain tumors in humans, of which GBM is the deadliest, with a median survival of approximately 15 months. [20-28] Studying TUCRs in gliomas is therefore an untouched avenue for understanding novel tumor enhancing mechanisms and discovering new biomarkers and therapeutic targets.

In this study, we leveraged large human datasets to identify the genomic locations, chromatin accessibility, transcription, differential expression, correlation with survival, and predicted functions of all 481 TUCRs, and identified TUCRs that are relevant to glioma biology (Supplementary Figure 1A). Of these, we investigated the expression, function, and mechanism of action of the most highly upregulated intergenic TUCR, uc.110, identifying it as a new tumor enhancer. Uc.110 was highly overexpressed in GBM and LGG, where it promoted malignancy parameters and tumor growth. Uc.110 activated the WNT pathway by upregulating the expression of membrane frizzled-related protein (MFRP), by sponging the tumor suppressor microRNA miR-544. This work shows important roles for TUCRs in gliomas and provides an extensive database and novel methods for future TUCR research in any disease context.

Setup

Set parameters and install required programs

This initial section is used to set the dependent variables that are used for this analysis.

The following variables can be set: * outputdir: The output directory to save outputs from each analysis. * filterannot: Set to TUCR, lncRNA, protein_coding,antisense_RNA, or misc_RNA, and is used to focus the dataset on a specific gene category for downstream expression and survival analyses.
* tpmethod: Set to TPM or tpm. Used to determine the method for deriving absolute expression of genes. * repel: Logical vector set to TRUE or FALSE. Used to determine whether gene labels are added to downstream scatter and volcano plots.

In addition to setting the above variables, this chunk loads the color palette and installs (if required) and loads all package dependencies for this document.

```
knitr::opts_chunk$set(
eval = FALSE,
tidy = TRUE#,
#   echo = TRUE,
#   message = FALSE,
#   warning = FALSE
)
```

```
## Variables

outputdir <- "Outputs"

filterannot <- "TUCR"

makekpmplots <- TRUE

tpmethod <- "TPM"

repel <- TRUE

## Load Colors

paper_black = "#000000"

paper_gold = "#EAA304"

paper_green2 = "#20C799"

paper_yellow = "#F0E442"

paper_blue = "#017DC3"

paper_lightblue = "#58AFE0"

paper_skyblue = "#A5DFFF"

paper_orange = "#D55E00"

paper_purple = "#8C004C"

paper_gray = "#CAC4C4"

paper_red = "#C30223"

paper_darkpink = "#DA7C8C"
```

```r
paper_pink = "#FFCCD5"

paper_green = "#54BB44"

paper_red2 = "#CA465C"

paper_turq = "#20C799"

## Load Programs

if (!require("jsonlite")) install.packages("jsonlite")
library("jsonlite")

if (!require("tidyjson")) install.packages("tidyjson")
library("tidyjson")

if (!require("tidyverse")) install.packages("tidyverse")
library("tidyverse")

if (!require("ggplot2")) install.packages("ggplot2")
library("ggplot2")

if (!require("ggforce")) install.packages("ggforce")
library("ggforce")

if (!require("ggbreak")) install.packages("ggbreak")
library("ggbreak")

if (!require("ggrepel")) install.packages("ggrepel")
library("ggrepel")

if (!require("ggfortify")) install.packages("ggfortify")
library("ggfortify")

if (!require("ggdendro")) install.packages("ggdendro")
library("ggdendro")

if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")
if (!requireNamespace("DESeq2", quietly = TRUE)) BiocManager::install("DESeq2")
library("DESeq2")

if (!require("survival")) install.packages("survival")
library("survival")

if (!require("broom")) install.packages("broom")
library("broom")

if (!requireNamespace("limma", quietly = TRUE)) BiocManager::install("limma")
library("limma")

if (!require("Tmisc")) install.packages("Tmisc")
library("Tmisc")
```

```r
if (!require("survminer")) install.packages("survminer")
library("survminer")

if (!require("corrr")) install.packages("corrr")
library("corrr")

if (!require("here")) install.packages("here")
library("here")

if (!require("lessR")) install.packages("lessR")
library("lessR")

if (!require("splitstackshape")) install.packages("splitstackshape")
library(splitstackshape)

if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")

if (!require("org.Hs.eg.db")) BiocManager::install("org.Hs.eg.db")
library(org.Hs.eg.db)

if (!require("GO.db")) BiocManager::install("GO.db")
library(GO.db)

if (!require("limma")) BiocManager::install("limma")
library(limma)

if (!require("ggthemes")) install.packages("ggthemes", dependencies = TRUE)
library(ggthemes)

if (!require("gplots")) install.packages("gplots")
library(gplots)

if (!require("ggsignif")) install.packages("ggsignif")
library(ggsignif)

if (!require("RColorBrewer")) install.packages("RColorBrewer")
library(RColorBrewer)

if (!require("reshape2")) install.packages("reshape2")
library(reshape2)

if (!require("gridExtra")) install.packages("gridExtra")
library(gridExtra)

if (!require("dendextend")) install.packages("dendextend")
library(dendextend)

if (!require("plyr")) install.packages("plyr")
library(plyr)

if (!require("magick")) install.packages("magick")
library(magick)
```

```
if (!require("png")) install.packages("png")
library(png)

if (!require("scales")) install.packages("scales")
library(scales)

if (!require("ggsci")) install.packages("ggsci")
library(ggsci)

if (!require("stringi")) install.packages("stringi")
library(stringi)
if (!require("stringr")) install.packages("stringr")
library(stringr)

if (!require("impute")) BiocManager::install("impute")
library(impute)

if (!require("preprocessCore")) BiocManager::install("preprocessCore")
library(preprocessCore)

if (!require("Go.db")) BiocManager::install("GO.db")
library(GO.db)


if (!require("WGCNA")) install.packages("WGCNA")
library(WGCNA)

if (!require("umap")) install.packages("umap")
library(umap)

if (!require("igraph")) install.packages("igraph")
library(igraph)

if (!require("gtable")) install.packages("gtable")
library(gtable)

if (!dir.exists(outputdir)) {
    dir.create(outputdir)
}
```

RESULTS

TUCRs are encoded throughout the genome, resistant to variation, and actively transcribed.

We analyzed TUCR genomic locations published in Bejerano et al, [1] using hg38 genome coordinates lifted over from the provided hg19 coordinates. We found that some TUCRs are exonic and are contained within an exon of the "host" gene. Others are contained within an intron instead. Some TUCRs straddle a region that spans exonic and intronic regions of the host gene (exonic/intronic), and others are not contained within a known genetic element at all (intergenic) (Supplementary Figure 1B). We manually annotated each TUCR using a combination of UCSC Genome Browser tracks, [29, 30] Quinlan Laboratory's bedtools, [31, 32] and TUCR genomic locations lifted over to hg38 from hg19. [1] We updated the initial annotations by Bejerano et al, 2004, to now include 45 exonic, 231 intronic, 68 intronic/exonic, and 137 intergenic TUCRs

(Supplementary Figure 1C). Intragenic TUCRs (exonic, intronic, and exonic/intronic) tend to congregate in protein coding genes, while intergenic TUCRs likely represent independent and potentially novel transcripts. (Supplementary Figure 1D). We confirmed that TUCRs are located on all 21 numbered chromosomes and the X chromosome. There were no annotated TUCRs on chromosome 21 (chr21), the Y chromosome (chrY) or in the mitochondrial DNA (chrM) (Supplementary Figure 1E). Detailed TUCR annotation information for all 481 TUCRs is provided in the supplementary materials (Supplementary Table 1).

Since TUCRs are expected to be resistant to variation [2], we characterized the overlap of current dbSNP (build 156) single nucleotide polymorphism (SNP) annotations to the lifted over hg38 TUCR genomic coordinates. We found that TUCRs overlap with fewer SNPs than protein coding genes and non-coding RNAs, suggesting that they are more resistant to variation. Remarkably, the results that we have acquired from this same independent analysis suggests that the lncRNAs in our dataset are more resistant to variation than protein coding genes, which are more resistant than antisense RNAs, which are more resistant than small miscellaneous RNAs, which are the least resistant to single nucleotide polymorphism variation. These results provide further insight into the mutational burden of different gene categories across the genome. (Supplementary Figure 1F)

We also investigated TUCR transcription levels in comparison to transcription of known protein-coding and non-coding genes. To accomplish this, we first analyzed their spatial associations with markers for active chromatin (H3K4me3), active enhancers (H3K27ac), lncRNA transcription (RNA Pol.II) and open chromatin (ATAC-Seq). We determined the significance of the spatial relationships between these marks and TUCRs utilizing publicly available U87 CHIP- and ATAC-Seq datasets. Then, we compared the data to TUCR intervals that were randomly shuffled to create a negative control, other classes of non-coding RNAs, and TUCRs subset by genomic annotation (Supplementary Figure 2A). We found that TUCRs displayed a significant enrichment for all transcriptional activity markers over expected and compared to control. The above data show that TUCRs are distributed throughout the genome, resistant to variation, and actively transcribed in GBM and LGG.

##TUCRs are highly expressed in GBM and LGG tumors. TUCR expression has not been characterized in GBM or LGG before. We performed the first comprehensive analysis of TUCR expression in these cancers by comparing GBM (n = 166) and LGG (n = 505) tumor samples from the Cancer Genome Atlas (TCGA) [33] to their normal brain cortex counterparts in TCGA (n = 5) and the Genotype-Tissue Expression Database (GTEx, n = 255). [34] We first analyzed absolute TUCR expression, as measured by transcripts-per-kilobase million (TPM). The absolute expression, in GBM, of all TUCRs was compared to the expression of lncRNAs, coding genes, antisense RNAs, and small noncoding RNAs (< 200 nt length), and the expression of TUCRs separated by genomic annotation into exonic, intronic, exonic/intronic, and intergenic. (Supplementary Figure 3A) All gene annotations were obtained using the CHESS gene catalog, which contains most Refseq and Ensembl genes, while also including a series of understudied novel genes.[35] Highly expressed genes are visualized via heatmap (>=10 tpm, red) along with moderately (>=1 tpm, white) and lowly expressed genes (<1 tpm, blue). These analyses were repeated in LGG (Supplementary Figure 3B). The data show that intragenic TUCRs are expressed at magnitudes that are like those of protein coding genes in both GBM and LGG, while intergenic TUCRs demonstrate expression levels that are closer to those of lncRNAs.

TUCRs are deregulated in gliomas, and deregulation is associated with clinical outcomes.

We analyzed TCGA tumor data and GTEx normal brain cortex data and found that in addition to being highly expressed in gliomas, TUCRs are highly deregulated in GBM and LGG as compared to normal brain cortex. Of the 481 annotated TUCRs, we identified 87 that were upregulated and 67 that were downregulated in GBM (Figure 1A). We also identified 59 TUCRs that were upregulated and 53 TUCRs that were downregulated in LGG. (Figure 1B). Of the 154 deregulated TUCRs in GBM, 86 were also deregulated in LGG, a 56% overlap (Figure 1C). Notably, there are multiple deregulated TUCRs across all annotation categories (Supplementary Figure 4A).

We then sought to determine whether deregulation of TUCR expression correlates with patient outcomes in LGG and GBM. For each of the 481 TUCRs, we generated a Kaplan-Meier plot tracking differences in survival

for high expressing (upper quartile) and low expressing (lower quartile) tumor groups. Of the TUCRs that are expressed in GBM TCGA RNA-Seq data, only 4 were correlated with survival in a statistically significant manner in both of our workflows ($p <= 0.05$, Figure 1D, left panel). We considered that this low prevalence of survival associated TUCRs in GBM was due to the short overall survival of GBM patients (~15 months). We also studied survival differences in LGG patients, as they have a longer median survival (~84 months). Of the TUCRs that are expressed in LGG TCGA RNA-Seq data, 93 were correlated with survival in both of our workflows (Figure 1D). We have highlighted two TUCRs that represent a statistically significant correlation with good (uc.338, Figure 2E) or poor (uc.75, Figure 2F) prognosis using both methods. When separated by annotation category, intragenic TUCR deregulation has a greater association with patient outcomes than intergenic TUCR deregulation (Supplementary Figure 4B). Since IDH mutation status could potentially serve as a positive confounding variable, we separated our samples into IDH WT and MUT cohorts (Supplementary Figure 5A) and again considered TUCR expression (Supplementary Figure 5B) and survival (Supplementary Figure 5C). Aggregating samples by IDH mutation status had a minimal effect on the number of deregulated TUCRs and was more likely to diminish the likelihood that a TUCR correlates with patient outcomes, rather than enhancing it. Expression, deregulation, and survival analyses were performed on all 481 TUCRs. Detailed results for individual TUCRs can be found as supplementary materials (Supplementary Figures 6).

TUCRs are coregulated with genes that have known functions.

We predicted TUCR function by identifying coregulated genes with known functions via weighted gene co-expression network analysis (WGCNA).[36] We aggregated the 42,644 genes in our dataset into 60 colored modules based on clustered gene ontology (GO) terms. Each of these modules contains genes with known functions, such as RNA binding, cell signaling, immune response, metabolic response, etc. These modules can also be used to identify genes that associate with clinical traits, such as the tumor tissue-type (Figure 1G). The data can also be used to predict gene function for novel genes. To do this, we aggregated all 481 TUCRs into our modules. We identified TUCRs that correlate with each of the 60 modules, with some having positive correlations and others negative. For example, TUCRs that exhibit a positive correlation with the #004C54 "midnight green" module (Supplementary Figure 7) could have a promoting effect on nucleic acid binding and regulation, while those that are negatively correlated with the #F4a460 module (Supplementary Figure 8) may have a negative effect on G-protein coupled receptor and metabolic functions. Since many different TUCRs show associations with different modules, and every module has at least one TUCR that is associated with it, these results suggest that TUCRs may have a broad range of potential functions in GBM and LGG (Figure 1H). WGCNA analyses were performed on all 481 TUCRs. Detailed results for individual TUCRs can be found as supplementary materials (Supplementary Figures 6).

Figure 1.

```
if (!dir.exists(paste(outputdir, "/Figure1/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure1/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/Figure1/DESeq2Results/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure1/DESeq2Results/", sep = ""))
}
```

Methodology: Expression Analysis, Figure 1A, 1B, and 1C

Identifying differentially expressed TUCRs with DESeq2

```
## read data

gbm_countfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_mergedcounts.txt",
    sep = "")
```

```r
lgg_countfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_mergedcounts.txt",
    sep = "")

cortex_countfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_mergedcounts.txt",
    sep = "")


gbm_metadatafile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_tcga_metadata.csv",
    sep = "")

lgg_metadatafile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_tcga_metadata.csv",
    sep = "")

cortex_metadatafile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_gtex_metadata.csv",
    sep = "")


gbm_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_seqdepth_counts.csv",
    sep = "")

lgg_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_seqdepth_counts.csv",
    sep = "")

cortex_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_seqdepth_counts.csv",
    sep = "")


## Merge Data


if (!is.na(cortex_countfile)) {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    lgg_mergedcounts <- read.table(lgg_countfile, header = TRUE)

    cortex_normalcounts <- read.table(cortex_countfile, header = TRUE)

    cortex_normalcounts <- cortex_normalcounts[, 9:ncol(cortex_normalcounts)]

    gbm_mergedcounts <- cbind(gbm_mergedcounts, cortex_normalcounts) %>%
        distinct()

    lgg_mergedcounts <- cbind(lgg_mergedcounts, cortex_normalcounts) %>%
        distinct()

    rm(cortex_normalcounts)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    lgg_metadata <- read_csv(file = lgg_metadatafile)

    cortex_metadata <- read_csv(file = cortex_metadatafile)
```

```
    gbm_metadata <- rbind(gbm_metadata, cortex_metadata)

    lgg_metadata <- rbind(lgg_metadata, cortex_metadata)

    rm(cortex_metadata)

    gbm_seqdepth <- read.csv(file = gbm_seqdepthfile)

    lgg_seqdepth <- read.csv(file = lgg_seqdepthfile)

    cortex_seqdepth <- read.csv(file = cortex_seqdepthfile)

    gbm_seqdepth <- rbind(gbm_seqdepth, cortex_seqdepth)

    lgg_seqdepth <- rbind(lgg_seqdepth, cortex_seqdepth)

    rm(cortex_seqdepth)

} else {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    lgg_mergedcounts <- read.table(lgg_countfile, header = TRUE)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    lgg_metadata <- read_csv(file = lgg__metadatafile)

    gbm_seqdepth <- read.csv(file = gbm__seqdepthfile)

    lgg_seqdepth <- read.csv(file = lgg__seqdepthfile)
}

gbm_mergedcounts_deseq2 <- gbm_mergedcounts %>%
    filter(tag == "TUCR" & annot != "random")

lgg_mergedcounts_deseq2 <- lgg_mergedcounts %>%
    filter(tag == "TUCR" & annot != "random")

gbm_mergedcounts_deseq2_countsonly <- gbm_mergedcounts_deseq2[, 9:ncol(gbm_mergedcounts_deseq2)]

lgg_mergedcounts_deseq2_countsonly <- lgg_mergedcounts_deseq2[, 9:ncol(lgg_mergedcounts_deseq2)]

rownames(gbm_mergedcounts_deseq2_countsonly) <- gbm_mergedcounts_deseq2$id

rownames(lgg_mergedcounts_deseq2_countsonly) <- lgg_mergedcounts_deseq2$id

gbm_mergedcounts_deseq2_info <- gbm_mergedcounts_deseq2[, 1:8]

lgg_mergedcounts_deseq2_info <- lgg_mergedcounts_deseq2[, 1:8]

gbm_mergedcounts_deseq2_info$length <- (gbm_mergedcounts_deseq2$end - gbm_mergedcounts_deseq2$start)/10

lgg_mergedcounts_deseq2_info$length <- (lgg_mergedcounts_deseq2$end - lgg_mergedcounts_deseq2$start)/10
```

```r
dds <- DESeqDataSetFromMatrix(countData = gbm_mergedcounts_deseq2_countsonly, colData = gbm_metadata,
    design = ~dex)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)

disease <- "GBM"

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

write_csv(res, file = paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs.csv",
    sep = ""))

dds <- DESeqDataSetFromMatrix(countData = lgg_mergedcounts_deseq2_countsonly, colData = lgg_metadata,
    design = ~dex)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)


disease <- "LGG"

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}


write_csv(res, file = paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs.csv",
    sep = ""))
```

```r
### Volcano plot

volcanoplot_expression <- function (res,
```

```
                          genes = "all",
                          title = "Deregulated TUCRs in TCGA Gliomas",
                          output = "",
                          height = 7,
                          width = 14,
                          dpi = 600,
                          annot_filter = ""){
  #res <- res_TUCR
  #genes <- "all"
  #annot_filter = ""
  #genes = c("uc.110")
  #title = "Deregulated Genes"

  res <- res %>%
  mutate(dereg = ifelse(log2FoldChange >=1, "upregulated",
                        ifelse(log2FoldChange <=-1,"downregulated","unchanged")),
         deregcount = ifelse(log2FoldChange >=1 & padj <= 0.05, 1,
                        ifelse(log2FoldChange <=-1 & padj <= 0.05,-1,0))) %>%
  dplyr::filter(padj != 0 | alias == "uc.110") %>%
  dplyr::group_by(alias) %>%
  dplyr::mutate(sum = sum(deregcount,na.rm=TRUE)) %>%
  ungroup() %>%
  dplyr::mutate(deregcategory = ifelse(sum == 2, "Up-Both",
                                ifelse(sum == -2, "Down-Both",
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM","Up-GBM",
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", "Up-LGG"
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", "Down
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", "Down
                color = ifelse(sum == 2, paper_darkpink,
                                ifelse(sum == -2, paper_lightblue,
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM",paper_red,
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", paper_pi
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", paper
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", paper
                                ifelse(disease == "GBM",paper_black,paper_gray)))))),
                newdisease = ifelse(str_detect(deregcategory,"Both"),"BOTH",disease),
                diseasefilter = ifelse((disease == "LGG" & newdisease == "BOTH"),0,1)) %>%
    dplyr::filter(diseasefilter == 1)

  res_summary <- res %>%
    group_by(color) %>%
    dplyr::summarize(n = n())

if(annot_filter == ""){}else{
  res %>%
    filter(annot == annot_filter)
}

res$newdisease <- factor(res$newdisease, levels = c("GBM","LGG","BOTH"))

res <- res %>%
  dplyr::mutate(newdisease_order = ifelse(newdisease=="GBM",1,
                                   ifelse(newdisease=="LGG",2,3)))
```

```
res_110_up <- res %>%
   filter(color != paper_gray & color != paper_black & log2FoldChange > 0 | alias == "uc.110") %>%
   dplyr::arrange(desc(log2FoldChange)) %>%
   dplyr::group_by(newdisease) %>%
   dplyr::mutate(order_rank= row_number()) %>%
   mutate(maxrank = max(order_rank,na.rm=TRUE)) %>%
   ungroup() %>%
   filter(order_rank <= 5 | alias == "uc.110")

res_110_down <- res %>%
   filter(color != paper_gray & color != paper_black  & log2FoldChange < 0 | alias == "uc.110") %>%
   dplyr::arrange(log2FoldChange) %>%
   dplyr::group_by(newdisease) %>%
   dplyr::mutate(order_rank= row_number()) %>%
   mutate(maxrank = max(order_rank,na.rm=TRUE)) %>%
   ungroup() %>%
   filter(order_rank <= 5 | alias == "uc.110")

p <- ggplot(res) +
        #geom_point(aes(x=log2FoldChange, y=-log10(pvalue),col=color)) +
        #geom_hline(yintercept = 1.30,linetype = 2) +
        geom_point(aes(x=log2FoldChange,y=-log10(padj),col=color)) +
        geom_hline(yintercept = -log10(0.05),linetype = 2,size=0.5) +
        geom_vline(xintercept = -1,linetype = 2,size=0.5) +
        geom_vline(xintercept = 1,linetype = 2,size=0.5) +
        scale_color_identity(guide = "deregcategory", labels = res$deregcategory, breaks = res$color) +
        #ggtitle(title) +
        guides(color = guide_legend(override.aes = list(size=5))) +
        geom_text_repel(data= res_110_up,aes(x=log2FoldChange,y=-log10(padj),color=paper_black, label =
        geom_text_repel(data= res_110_down,aes(x=log2FoldChange,y=-log10(padj),color=paper_black, label
        labs(x = "log2 fold change of TUCR in tumors",
        #y = "-log10 adjusted p-value",
        y = "-log10 of adjusted p-value",
        color = "Legend") +
        theme(plot.title = element_text(size = rel(2.0), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_text(size=10)) +
  coord_cartesian(clip = "off") +
  facet_wrap(~reorder(newdisease,newdisease_order),scales="free",ncol=3)

p

ggsave(plot = print(p),filename = output, width = width, height = height, dpi = 600)

}
```

Figure 1A, 1B and 1C (New)

```
res_TUCR_LGG <- read_csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs.csv",
    sep = ""), col_names = TRUE)

res_TUCR_GBM <- read_csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs.csv",
    sep = ""), col_names = TRUE)

tucr_annot <- read_delim("Inputs/general_files/bedfiles/hg38.ultraconserved.bed",
    col_names = TRUE, delim = "\t")

res_TUCR_LGG_2 <- res_TUCR_LGG %>%
    separate(row, into = c("alias", "kibble"), sep = "___") %>%
    dplyr::select(-kibble) %>%
    left_join(tucr_annot, by = "alias") %>%
    filter(!is.na(chrom)) %>%
    dplyr::select(alias, annot, log2FoldChange, padj) %>%
    mutate(disease = "LGG")

res_TUCR_GBM_2 <- res_TUCR_GBM %>%
    separate(row, into = c("alias", "kibble"), sep = "___") %>%
    dplyr::select(-kibble) %>%
    left_join(tucr_annot, by = "alias") %>%
    filter(!is.na(chrom)) %>%
    dplyr::select(alias, annot, log2FoldChange, padj) %>%
    mutate(disease = "GBM")

res_TUCR <- rbind(res_TUCR_GBM_2, res_TUCR_LGG_2)

volcanoplot_expression(res_TUCR, genes = "all", output = paste(outputdir, "/Figure1/figure1abc.png",
    sep = ""), height = 3, width = 12, dpi = 600, annot = "")
```

Figure 1C (Old)

```
res_TUCR_LGG <- read_csv(paste(outputdir,"/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs.csv",sep=

res_TUCR_GBM <- read_csv(paste(outputdir,"/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs.csv",sep=

tucr_annot <- read_delim("Inputs/general_files/bedfiles/hg38.ultraconserved.bed",col_names = TRUE,delim=

res_TUCR_LGG_2 <- res_TUCR_LGG %>%
  separate(row,into=c("alias","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  left_join(tucr_annot,by="alias") %>%
  filter(!is.na(chrom)) %>%
  dplyr::select(alias,log2FoldChange,padj) %>%
  mutate(disease = "LGG")

res_TUCR_GBM_2 <- res_TUCR_GBM %>%
  separate(row,into=c("alias","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  left_join(tucr_annot,by="alias") %>%
  filter(!is.na(chrom)) %>%
```

```
  dplyr::select(alias,log2FoldChange,padj) %>%
  mutate(disease = "GBM")

res_TUCR <- rbind(res_TUCR_GBM_2,res_TUCR_LGG_2)

res_summary <- res_TUCR %>%
  mutate(dereg = ifelse(log2FoldChange >=1  & padj <= 0.05, "upregulated",
                    ifelse(log2FoldChange <=-1  & padj <= 0.05,"downregulated","unchanged")),
         deregcount = ifelse(log2FoldChange >=1  & padj <= 0.05, 1,
                    ifelse(log2FoldChange <=-1  & padj <= 0.05,-1,0))) %>%
  dplyr::group_by(alias) %>%
  dplyr::mutate(sum = sum(deregcount,na.rm=TRUE)) %>%
  ungroup() %>%
  dplyr::mutate(deregcategory = ifelse(sum == 2, "Up-Both",
                                ifelse(sum == -2, "Down-Both",
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM","Up-GBM",
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", "Up-LGG"
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", "Down
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", "Down
                newdisease = ifelse(str_detect(deregcategory,"Both"),"BOTH",disease),
                diseasefilter = ifelse((disease == "LGG" & newdisease == "BOTH"),0,1)) %>%
    dplyr::filter(diseasefilter == 1)  %>%
  filter(deregcategory != "not deregulated" & padj != 0) %>%
  dplyr::select(alias,deregcategory) %>%
  distinct() %>%
  group_by(deregcategory) %>%
  dplyr::summarize(n = n()) %>%
  mutate(dereg = ifelse(str_detect(deregcategory,"Up"),"Upregulated","Downregulated")) %>%
  group_by(dereg) %>%
  dplyr::mutate(deregcounts = sum(n),circlecounts=(n/deregcounts)*360,min = cumsum(circlecounts) - circl
  ungroup() %>%
  mutate(allsum = sum(n))
  #dplyr::mutate(deregcounts = n(),lab.ypos = (cumsum(n)-deregcounts) - 0.5*n)

p <- ggplot(res_summary, aes(x=2,y=circlecounts,fill=deregcategory))          +
  geom_bar(
    stat="identity",#colour="black",
    width = 1)                                                                +
  geom_text(
    aes(y = lab.ypos, label = n),
    color = "black",size=5)                                                    +
  coord_polar("y", start=0)                                                   +
  scale_fill_manual(values=c(paper_lightblue,paper_blue,paper_skyblue,paper_darkpink,paper_red,paper_pi
  ggtitle(paste("TUCR Genomic Location"))                                     +
  guides(colour = guide_legend(override.aes = list(size=8))) +
        theme(plot.title = element_blank(),
              axis.title = element_blank(),
              axis.text.y = element_blank(),
              axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_blank(),legend.position="none",legend.title=ele
legend.text = element_text(size = 12.5),
axis.ticks = element_blank(),
```

```
      legend.box.spacing = unit(0, "pt"),
legend.margin=margin(0,0,0,0),
strip.background = element_rect(fill="white"),
strip.text = element_blank(),
panel.spacing = unit(-2, "lines")) +
  xlim(0.5, 2.5) +
  facet_wrap(~dereg)

ggsave(file=paste(outputdir,"/Figure1/figure1c_old.png",sep=""),
       plot = print(p),
       width = 6,
       height = 4,
       dpi = 600)
```

Methodology: Survival Analysis, Figures 1D, 1E, and 1F

Completing survival analysis for TUCRs

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 5

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
```

```
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
```

```
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")
```

```r
# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {

    km_countdata <- z_rna
    colnames(km_countdata) <- metadata$id[t_index]
    posdata$median <- rowMedians(survcounts)

    # km_countdata2 <- km_countdata[,complete.cases(clinical2)]

    probs <- c(0.25, 0.5, 0.75)
    q <- rowQuantiles(km_countdata, probs = probs)
    posdata$n25 <- q[, 1]
    posdata$n75 <- q[, 3]
    posdata <- posdata %>%
        dplyr::select(TUCR = alias.x, median, n25, n75)
    km_TUCRs <- cbind(posdata, km_countdata)

    km_TUCRs <- km_TUCRs %>%
        gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
        mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
        distinct %>%
        dplyr::select(TUCR, median, id, group) %>%
        left_join(clinical2, by = "id") %>%
        dplyr::filter(median != 0)

    TUCRids <- as.character(posdata$TUCR)
    i <- 1

    ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
        "pvalue", "method"))))

    for (i in 1:length(TUCRids)) {
        print(i)
```

```
        print(TUCRids[i])

        skip_to_next <- FALSE
        TUCRsurv <- as.character(TUCRids[i])
        if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
            dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
        }

        # TUCR <- 'uc.1'
        kmdata <- km_TUCRs %>%
            dplyr::filter(TUCR == TUCRsurv) %>%
            dplyr::select(TUCR, id, group, time, status)
        fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
            skip_to_next <<- TRUE
        })
        p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
            risk.table = TRUE), error = function(e) {
            skip_to_next <<- TRUE
        })
        # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
        # TRUE,risk.table = TRUE)

        if (skip_to_next == TRUE) {
            rbinder <- cbind(as.character(TUCRsurv), NA, NA)
            ptable[i, ] <- rbinder
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}
```

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 6

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")
```

```
n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}
```

```r
if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
```

```
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {

    km_countdata <- z_rna
    colnames(km_countdata) <- metadata$id[t_index]
    posdata$median <- rowMedians(survcounts)
```

```r
# km_countdata2 <- km_countdata[,complete.cases(clinical2)]

probs <- c(0.25, 0.5, 0.75)
q <- rowQuantiles(km_countdata, probs = probs)
posdata$n25 <- q[, 1]
posdata$n75 <- q[, 3]
posdata <- posdata %>%
    dplyr::select(TUCR = alias.x, median, n25, n75)
km_TUCRs <- cbind(posdata, km_countdata)

km_TUCRs <- km_TUCRs %>%
    gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
    mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
    distinct %>%
    dplyr::select(TUCR, median, id, group) %>%
    left_join(clinical2, by = "id") %>%
    dplyr::filter(median != 0)

TUCRids <- as.character(posdata$TUCR)
i <- 1

ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
    "pvalue", "method"))))

for (i in 1:length(TUCRids)) {
    print(i)
    print(TUCRids[i])

    skip_to_next <- FALSE
    TUCRsurv <- as.character(TUCRids[i])
    if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
        dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
    }

    # TUCR <- 'uc.1'
    kmdata <- km_TUCRs %>%
        dplyr::filter(TUCR == TUCRsurv) %>%
        dplyr::select(TUCR, id, group, time, status)
    fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
        skip_to_next <<- TRUE
    })
    p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
        risk.table = TRUE), error = function(e) {
        skip_to_next <<- TRUE
    })
    # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
    # TRUE,risk.table = TRUE)

    if (skip_to_next == TRUE) {
        rbinder <- cbind(as.character(TUCRsurv), NA, NA)
        ptable[i, ] <- rbinder
    } else {
        grid.draw.ggsurvplot <- function(x) {
```

```
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}
```

Figure 1D

Writing a script to generate a volcano plot for survival

```
### Volcano plot

volcanosurv <- function (res,genes = "all",title = "TUCRs correlated with survival in gliomas", output =
  #res <- res_surv
  #genes = c("uc.110","uc.62")
  #title = paste("TUCR correlation with patient survival in ",disease,sep="")
  #output = paste(outputdir,"/intergenic_tucr_results_volcanosurv.png",sep="")
  i <- 1
  vres <- res %>%
    filter(abs(estimate) <=1)
  vres <- vres %>% mutate(gene="",Survival="",color="")
  for (i in 1:length(vres$id)){
  ifelse(is.na(vres$pvalue[i]),vres$pvalue[i] <- 1,vres$pvalue[i] <- vres$pvalue[i])

  ifelse(genes!="all",ifelse(!is.na(match(vres$id[i],genes)),vres$gene[i] <- as.character(vres$id[i]),

  ifelse((vres$pvalue[i] <0.05 & vres$p.value[i] <0.05 & vres$estimate[i] < -0),{vres$Survival[i] <- "Si
  ifelse((vres$pvalue[i] <0.05 & vres$p.value[i] <0.05 & vres$estimate[i] > 0),{vres$Survival[i] <- "Sig
  ifelse((vres$pvalue[i] >0.05 & vres$p.value[i] <0.05 & vres$estimate[i] < -0),{vres$Survival[i] <- "Si
  ifelse((vres$pvalue[i] >0.05 & vres$p.value[i] <0.05 & vres$estimate[i] > 0),{vres$Survival[i] <- "Sig
  ifelse((vres$pvalue[i] <0.05 & vres$p.value[i] >0.05),{vres$Survival[i] <- "Significant (KM)";vres$col
  ifelse({vres$Survival[i] <- "Not significant";vres$color[i] <- "lightgray";}))))))}
  #{vres$Survival[i] <- "Not significant";vres$color[i] <- "lightgray";}))}

  vres <- vres %>%
  arrange(desc(Survival))


  ## plot
if(repel==TRUE){p <- ggplot(vres) +
        geom_point(aes(x=estimate, y=-log10(as.numeric(p.value)),col=color)) +
        scale_color_identity(guide = "legend", labels = vres$Survival, breaks = vres$color) +
        ggtitle(title) +
```

```
        labs(x = "Cox Estimated Proportional Hazard",
        y = "-log10 P-Value (CH)",
        color = "Legend")+
        theme(plot.title = element_blank(),
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=10)) +
        geom_text_repel(aes(x=estimate, y=-log10(as.numeric(p.value)),label = gene),force=50) +
        facet_wrap(~disease.y)
}else{
  p <- ggplot(vres) +
        geom_point(aes(x=estimate, y=-log10(as.numeric(p.value)),col=color)) +
    scale_color_identity(guide = "legend", labels = vres$Survival, breaks = vres$color) +
        + ggtitle(title) +
        labs(x = "Cox Estimated Proportional Hazard",
        y = "-log10 P-Value (CH)",
        color = "Legend") +
        theme(plot.title = element_blank(),
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=10))}
ggsave(output, width = width, height = height, dpi = dpi) +
        facet_wrap(~disease.y)
}
```

```
genes <- ""

lgg_surv_TUCR <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_survival_coxph_allTUCRs
    sep = ""), header = TRUE) %>%
    mutate(disease = "LGG")

lgg_ptable <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_survival_kpm_allTUCRs.csv
    sep = ""), header = TRUE) %>%
    dplyr::select(id = TUCR, pvalue, method) %>%
    mutate(disease = "LGG")

gbm_surv_TUCR <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_survival_coxph_allTUCRs
    sep = ""), header = TRUE) %>%
    mutate(disease = "GBM")

gbm_ptable <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_survival_kpm_allTUCRs.csv
    sep = ""), header = TRUE) %>%
    dplyr::select(id = TUCR, pvalue, method) %>%
    mutate(disease = "GBM")

surv_TUCR <- rbind(gbm_surv_TUCR, lgg_surv_TUCR)
```

```
ptable <- rbind(gbm_ptable, lgg_ptable)

colnames(ptable) <- c("alias.x", "pvalue", "method", "disease")

res_surv <- inner_join(ptable, surv_TUCR, by = "alias.x")

res_surv_sum <- res_surv

res_surv_sum <- res_surv_sum %>%
    mutate(gene = "", Survival = "", color = "")

annot_unique <- as.character(unique(res_surv_sum$annot))

annotation <- "All"

res_surv_sum2 <- res_surv_sum

for (i in 1:length(res_surv_sum2$alias)) {
    # print(as.character(res_surv_sum2$alias.x[i]))
    ifelse(is.na(res_surv_sum2$pvalue[i]), res_surv_sum2$pvalue[i] <- 1, res_surv_sum2$pvalue[i] <- res

    ifelse(genes != "all", ifelse(!is.na(match(res_surv_sum2$alias[i], genes)), res_surv_sum2$gene[i] <-
        ""), res_surv_sum2$gene[i] <- res_surv_sum2$alias[i])

    ifelse((res_surv_sum2$pvalue[i] < 0.05 & res_surv_sum2$p.value[i] < 0.05 & res_surv_sum2$estimate[i]
        0), {
        res_surv_sum2$Survival[i] <- "Significant (Both, Good Prognosis)"
        res_surv_sum2$color[i] <- print(paper_green)
    }, ifelse((res_surv_sum2$pvalue[i] < 0.05 & res_surv_sum2$p.value[i] < 0.05 &
        res_surv_sum2$estimate[i] > 0), {
        res_surv_sum2$Survival[i] <- "Significant (Both, Poor Prognosis)"
        res_surv_sum2$color[i] <- print(paper_red)
    }, ifelse((res_surv_sum2$pvalue[i] > 0.05 & res_surv_sum2$p.value[i] < 0.05 &
        res_surv_sum2$estimate[i] < 0), {
        res_surv_sum2$Survival[i] <- "Significant (CH, Good Prognosis)"
        res_surv_sum2$color[i] <- print(paper_green)
    }, ifelse((res_surv_sum2$pvalue[i] > 0.05 & res_surv_sum2$p.value[i] < 0.05 &
        res_surv_sum2$estimate[i] > 0), {
        res_surv_sum2$Survival[i] <- "Significant (CH, Poor Prognosis)"
        res_surv_sum2$color[i] <- print(paper_red)
    }, ifelse((res_surv_sum2$pvalue[i] < 0.05 & res_surv_sum2$p.value[i] > 0.05),
        {
            res_surv_sum2$Survival[i] <- "Significant (KM)"
            res_surv_sum2$color[i] <- "black"
        }, {
            res_surv_sum2$Survival[i] <- "Not significant"
            res_surv_sum2$color[i] <- "lightgray"
        })))))
}

# res_surv_sum4 <- res_surv_sum2 %>% group_by(Survival) %>%
# dplyr::summarise(count = n(),color) %>% distinct() %>%
# arrange(desc(Survival))
```

```
# write.csv(res_surv_sum4,file=paste(outputdir,'/SurvivalAnalysis/SummaryFigures/BarGraphs/',disease,'_

# p <- ggplot(res_surv_sum4, aes(x=Survival,y=count,fill=color)) +
# geom_bar(stat='identity', width = 0.7) + scale_fill_identity(guide =
# 'legend', labels = res_surv_sum2$Survival, breaks = res_surv_sum2$color) +
# labs(y = '# of TUCRs', fill = 'Legend') + theme( plot.title =
# element_text(size=rel(1.5), face='bold',hjust = 0.5), axis.title =
# element_text(size = rel(1.25), face='bold'), panel.grid.major =
# element_blank(), panel.grid.minor = element_blank(), panel.background =
# element_blank(), axis.line = element_line(colour = 'black'), axis.text.x =
# element_blank())

# ggsave(file=paste(outputdir,'/SurvivalAnalysis/SummaryFigures/BarGraphs/',disease,'_',annotation,'_tu
# = print(p), width = 5, height = 2.5, dpi = 600)

res_surv_sum3 <- res_surv_sum

volcanosurv(res_surv_sum3, genes = "", output = paste(outputdir, "/Figure1/Figure1d.png",
    sep = ""))
```

Figure 1E

Completing survival analysis for TUCRs

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 6

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)
```

```
    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}
```

```
posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)
```

```r
clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {

    km_countdata <- z_rna
    colnames(km_countdata) <- metadata$id[t_index]
    posdata$median <- rowMedians(survcounts)

    # km_countdata2 <- km_countdata[,complete.cases(clinical2)]

    probs <- c(0.25, 0.5, 0.75)
    q <- rowQuantiles(km_countdata, probs = probs)
    posdata$n25 <- q[, 1]
    posdata$n75 <- q[, 3]
    posdata <- posdata %>%
        dplyr::select(TUCR = alias.x, median, n25, n75)
    km_TUCRs <- cbind(posdata, km_countdata)

    km_TUCRs <- km_TUCRs %>%
        gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
        mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
        distinct %>%
        dplyr::select(TUCR, median, id, group) %>%
        left_join(clinical2, by = "id") %>%
        dplyr::filter(median != 0)

    TUCRids <- as.character(posdata$TUCR)
    i <- 1
```

```
    ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
        "pvalue", "method"))))

    for (i in 1:length(TUCRids)) {
        print(i)
        print(TUCRids[i])

        skip_to_next <- FALSE
        TUCRsurv <- as.character(TUCRids[i])
        if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
            dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
        }

        # TUCR <- 'uc.1'
        kmdata <- km_TUCRs %>%
            dplyr::filter(TUCR == TUCRsurv) %>%
            dplyr::select(TUCR, id, group, time, status)
        fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
            skip_to_next <<- TRUE
        })
        p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
            risk.table = TRUE), error = function(e) {
            skip_to_next <<- TRUE
        })
        # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
        # TRUE,risk.table = TRUE)

        if (skip_to_next == TRUE) {
            rbinder <- cbind(as.character(TUCRsurv), NA, NA)
            ptable[i, ] <- rbinder
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}

TUCRsurv <- "uc.132"
kmdata <- km_TUCRs %>%
    dplyr::filter(TUCR == TUCRsurv) %>%
    dplyr::select(TUCR, id, group, time, status)
```

```
fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
    skip_to_next <<- TRUE
})
p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE, risk.table = TRUE),
    error = function(e) {
        skip_to_next <<- TRUE
    })
# p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval = TRUE,risk.table =
# TRUE)

if (skip_to_next == TRUE) {
    rbinder <- cbind(as.character(TUCRsurv), NA, NA)
    ptable[i, ] <- rbinder
} else {
    grid.draw.ggsurvplot <- function(x) {
        survminer:::print.ggsurvplot(x, newpage = FALSE)
    }
    ggsave(file = paste(outputdir, "/Figure1/figure1e.png", sep = ""), device = "png",
        plot = p)
}
```

Figure 1F

Completing survival analysis for TUCRs

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 6

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
```

```
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
```

```
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)
```

```r
clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {

    km_countdata <- z_rna
    colnames(km_countdata) <- metadata$id[t_index]
    posdata$median <- rowMedians(survcounts)

    # km_countdata2 <- km_countdata[,complete.cases(clinical2)]

    probs <- c(0.25, 0.5, 0.75)
    q <- rowQuantiles(km_countdata, probs = probs)
    posdata$n25 <- q[, 1]
    posdata$n75 <- q[, 3]
    posdata <- posdata %>%
        dplyr::select(TUCR = alias.x, median, n25, n75)
    km_TUCRs <- cbind(posdata, km_countdata)

    km_TUCRs <- km_TUCRs %>%
        gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
        mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
        distinct %>%
        dplyr::select(TUCR, median, id, group) %>%
        left_join(clinical2, by = "id") %>%
        dplyr::filter(median != 0)
```

```
    TUCRids <- as.character(posdata$TUCR)
    i <- 1

    ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
        "pvalue", "method"))))

    for (i in 1:length(TUCRids)) {
        print(i)
        print(TUCRids[i])

        skip_to_next <- FALSE
        TUCRsurv <- as.character(TUCRids[i])
        if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
            dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
        }

        # TUCR <- 'uc.1'
        kmdata <- km_TUCRs %>%
            dplyr::filter(TUCR == TUCRsurv) %>%
            dplyr::select(TUCR, id, group, time, status)
        fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
            skip_to_next <<- TRUE
        })
        p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
            risk.table = TRUE), error = function(e) {
            skip_to_next <<- TRUE
        })
        # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
        # TRUE,risk.table = TRUE)

        if (skip_to_next == TRUE) {
            rbinder <- cbind(as.character(TUCRsurv), NA, NA)
            ptable[i, ] <- rbinder
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}

TUCRsurv <- "uc.75"
```

```
kmdata <- km_TUCRs %>%
    dplyr::filter(TUCR == TUCRsurv) %>%
    dplyr::select(TUCR, id, group, time, status)
fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
    skip_to_next <<- TRUE
})
p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE, risk.table = TRUE),
    error = function(e) {
        skip_to_next <<- TRUE
    })
# p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval = TRUE,risk.table =
# TRUE)

if (skip_to_next == TRUE) {
    rbinder <- cbind(as.character(TUCRsurv), NA, NA)
    ptable[i, ] <- rbinder
} else {
    grid.draw.ggsurvplot <- function(x) {
        survminer:::print.ggsurvplot(x, newpage = FALSE)
    }
    ggsave(file = paste(outputdir, "/Figure1/figure1f.png", sep = ""), device = "png",
        plot = p)
}
```

Methodology: WGCNA, Figure 1G and 1H

```
disease <- "GBM"

normal <- "cortex"

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
```

```
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

tpmcounts <- mergedcounts %>%
    mutate(length = end.x - start.x)

tpmcounts.info <- tpmcounts %>%
    dplyr::select(chrom = chrom.x, start = start.x, end = end.x, strand = strand.x,
        id, alias = alias.x, tag = tag.x, annot = annot.x, length)
```

```
tpmcounts <- tpmcounts %>%
    dplyr::select(-chrom.x, -start.x, -end.x, -id, -strand.x, -tag.x, -annot.x, -alias.x,
        -length)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1e+06

tpm <- function(counts, len, dep) {
    # x <- tpmcounts/genelength x2 <- t(t(x)/(seqdepth2))
    x <- counts/len
    return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts, genelength, seqdepth2)

tpm.df <- cbind(tpmcounts.info, tpm.df)

mergedcounts2 <- tpm.df %>%
    filter(annot != "random")

rm(tpm.df, tpmcounts, tpmcounts.info, seqdepth)

datExpr0 <- as.data.frame(t(mergedcounts2[, -c(1:9)]))
names(datExpr0) <- mergedcounts2$id
rownames(datExpr0) <- names(mergedcounts2)[-c(1:9)]
```

```
gsg = goodSamplesGenes(datExpr0, verbose = 3)
gsg$allOK
```

```
if (!gsg$allOK) {
    # Optionally, print the gene and sample names that were removed:
    if (sum(!gsg$goodGenes) > 0)
        printFlush(paste("Removing genes:", paste(names(datExpr0)[!gsg$goodGenes],
            collapse = ", ")))
    if (sum(!gsg$goodSamples) > 0)
        printFlush(paste("Removing samples:", paste(rownames(datExpr0)[!gsg$goodSamples],
            collapse = ", ")))
    # Remove the offending genes and samples from the data:
    datExpr0 = datExpr0[gsg$goodSamples, gsg$goodGenes]
}
```

```
sampleTree = hclust(dist(datExpr0), method = "average")
# Plot the sample tree: Open a graphic output window of size 12 by 9 inches The
# user should change the dimensions if the window is too large or too small.
sizeGrWindow(120, 120)
pdf(file = "sampleClustering.pdf", width = 12, height = 9)
par(cex = 0.6)
par(mar = c(0, 4, 2, 0))
plot(sampleTree, main = "Sample clustering to detect outliers", sub = "", xlab = "",
    cex.lab = 1.5, cex.axis = 1.5, cex.main = 2)
```

```
# Determine cluster under the line
#clust = cutreeStatic(sampleTree, cutHeight = 15, minSize = 10)
#table(clust)
# clust 1 contains the samples we want to keep.
#keepSamples = (clust==1)
#datExpr = datExpr0[keepSamples, ]
datExpr = datExpr0[]
newcolnames <- colnames(datExpr)
newrownames <- row.names(datExpr)
datExpr <- matrix(as.numeric(unlist(datExpr)),    # Convert to numeric matrix
                  ncol = ncol(datExpr))
colnames(datExpr) <- newcolnames
row.names(datExpr) <- newrownames
nGenes = ncol(datExpr)
nSamples = nrow(datExpr)
```

The variable datExpr now contains the expression data ready for network analysis.

We now read in the trait data and match the samples for which they were measured to the expression samples.

```
tucrcounts <- mergedcounts %>%
  # Filter(alias.x == "LINC00643" | alias.x == "SOX21-AS1" | tag.x == "TUCR" & annot.x != "random") %>%
  filter(tag.x == "TUCR" & annot.x != "random") %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-strand.x,-annot.x,-tag.x,-id)

rownames(tucrcounts) <- tucrcounts$alias.x

tucrcolumns <- colnames(tucrcounts[,2:length(colnames(tucrcounts))])

tucrcounts <- tucrcounts[,-1]

tucrcounts <- t(tucrcounts)

tucrcounts <- cbind(tucrcolumns,tucrcounts)

colnames(tucrcounts)[1] <- "survid"

tucrcounts <- as.data.frame(tucrcounts)

traitData <- metadata %>%
  left_join(tucrcounts,by="survid")

# remove columns that hold information we do not need.
allTraits = traitData[, -c(1,3)]

# Form a data frame analogous to expression data that will hold the clinical traits.

tcgaSamples = rownames(datExpr)
traitRows = match(tcgaSamples, allTraits$survid)
datTraits = allTraits
newrows <- allTraits[traitRows,2]
rownames(datTraits) <- as.character(datTraits$survid)
datTraits2 <- datTraits %>%
  mutate(dex2 = ifelse(dex == "tumor",1,0),p53status2 = ifelse(p53status == "WT",0,1)) %>%
```

```
  dplyr::select(-survid,-dex,-p53status)

newcolnames <- colnames(datTraits2)

datTraits2 <- matrix(as.numeric(unlist(datTraits2)),    # Convert to numeric matrix
                  ncol = ncol(datTraits2))
rownames(datTraits2) <- as.character(datTraits$survid)
colnames(datTraits2) <- newcolnames

is.numeric(datTraits2)

collectGarbage()
```

```
save(datExpr, datTraits2, file = "./Inputs/general_files/wgcnafiles/TUCR-01-dataInput.RData")
```

```
# The following setting is important, do not omit.
options(stringsAsFactors = FALSE)
# Allow multi-threading within WGCNA. This helps speed up certain calculations.
# At present this call is necessary for the code to work.  Any error here may
# be ignored but you may want to update WGCNA if you see one.  Caution: skip
# this line if you run RStudio or other third-party R environments.  See note
# above. enableWGCNAThreads() Load the data saved in the first part
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-01-dataInput.RData")
# The variable lnames contains the names of loaded variables.
lnames
```

```
# Choose a set of soft-thresholding powers
powers = c(c(1:10), seq(from = 12, to = 20, by = 2))
# Call the network topology analysis function
sft = pickSoftThreshold(datExpr, powerVector = powers, verbose = 5)
# Plot the results:
sizeGrWindow(9, 5)
par(mfrow = c(1, 2))
cex1 = 0.9
# scale-free topology fit index as a function of the soft-thresholding power
{
    plot(sft$fitIndices[, 1], -sign(sft$fitIndices[, 3]) * sft$fitIndices[, 2], xlab = "Soft Threshold
        ylab = "Scale Free Topology Model Fit,signed R^2", type = "n", main = paste("Scale independence
    text(sft$fitIndices[, 1], -sign(sft$fitIndices[, 3]) * sft$fitIndices[, 2], labels = powers,
        cex = cex1, col = "red")
    # this line corresponds to using an R^2 cut-off of h
    abline(h = 0.9, col = "red")
}
# Mean connectivity as a function of the soft-thresholding power
{
    plot(sft$fitIndices[, 1], sft$fitIndices[, 5], xlab = "Soft Threshold (power)",
        ylab = "Mean Connectivity", type = "n", main = paste("Mean connectivity"))
    text(sft$fitIndices[, 1], sft$fitIndices[, 5], labels = powers, cex = cex1, col = "red")
}
```

```
# net2 = blockwiseModules(datExpr, power = 8,maxBlockSize=15000, TOMType =
# 'unsigned', minModuleSize = 30, reassignThreshold = 0.05, mergeCutHeight =
# 0.25, deepSplit = 2,randomSeed = 20240219, numericLabels = TRUE,
# pamRespectsDendro = FALSE, saveTOMs = TRUE, saveTOMFileBase = 'TUCRTOM',
# verbose = 3)
```

```
net = blockwiseModules(datExpr, power = 8, maxBlockSize = 15000, TOMType = "unsigned",
    minModuleSize = 30, reassignThreshold = 0.05, mergeCutHeight = 0.4, deepSplit = 2,
    randomSeed = 20240219, numericLabels = TRUE, pamRespectsDendro = FALSE, saveTOMs = TRUE,
    saveTOMFileBase = "./Inputs/general_files/wgcnafiles/TUCRTOM", verbose = 3)

# net3 = blockwiseModules(datExpr, power = 8,maxBlockSize=15000, TOMType =
# 'unsigned', minModuleSize = 60, reassignThreshold = 0.05, mergeCutHeight =
# 0.25, deepSplit = 2,randomSeed = 20240219, numericLabels = TRUE,
# pamRespectsDendro = FALSE, saveTOMs = TRUE, saveTOMFileBase = 'TUCRTOM',
# verbose = 3)

# Save(net,net2,net3,file='nets.Rdata')
```

Figure 1G

colors are from https://mokole.com/palette.html

```
load(file = "./Inputs/general_files/wgcnafiles/nets.Rdata")

if (!dir.exists(paste(outputdir, "/Figure1/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure1/", sep = ""))
}

# open a graphics window
sizeGrWindow(12, 9)
# Convert labels to colors for plotting

moduleLabels = net$colors
MEs = net$MEs

## getcolors

n <- length(MEs)
dat_brewer <- read.csv("colorcodelist.csv", header = F)

col_vector = as.character(dat_brewer$V2)

pie(rep(1, n), col = col_vector[2:n])


# moduleColors = labels2colors(net$colors,colorSeq=col_vector[1:n])
moduleColors = col_vector[moduleLabels + 1]
genetree = net$dendrograms[[1]]


# Plot the dendrogram and the module colors underneath

i <- 1
for (i in 1:length(net$dendrograms)) {
    print(i)
    png(file = paste(outputdir, "/Figure1/figure1g_", i, ".png", sep = ""))
    plotDendroAndColors(dendro = net$dendrograms[[i]], colors = moduleColors[net$blockGenes[[i]]],
        "Module colors", dendroLabels = FALSE, hang = 0.03, addGuide = TRUE, guideHang = 0.05)
```

```
    dev.off()

}

save(net, MEs, moduleLabels, moduleColors, genetree, file = "./Inputs/general_files/wgcnafiles/TUCR-02-
```

```
# Load the WGCNA package
library(WGCNA)
# The following setting is important, do not omit.
options(stringsAsFactors = FALSE)
# Load the expression and trait data saved in the first part
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-01-dataInput.RData")
# The variable lnames contains the names of loaded variables.
lnames
# Load network data saved in the second part.
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-02-networkConstruction-auto.RData")
lnames
```

filter out modules that are particularly large.

```
# Define numbers of genes and samples
nGenes = ncol(datExpr)
nSamples = nrow(datExpr)
# Recalculate MEs with color labels
MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes

datExpr2 = as.data.frame(datExpr)
MEs = orderMEs(MEs0)
datTraits2 <- as.data.frame(datTraits2)
moduleTraitCor = cor(MEs, datTraits2, use = "p")
moduleTraitPvalue = corPvalueStudent(moduleTraitCor, nSamples)
```

```
geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))
MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

allmodules <- as.data.frame(t(MMPvalue))

allmodules2 <- allmodules %>%
    dplyr::mutate(Module = row.names(allmodules)) %>%
    gather(key = "Gene", value = "pvalue", -Module) %>%
    group_by(Gene) %>%
    mutate(maxmodule = min(pvalue, na.rm = TRUE)) %>%
    filter(pvalue == maxmodule) %>%
    group_by(Module) %>%
    dplyr::summarize(n = n())

allmodules3 <- allmodules2 %>%
    mutate(q1 = quantile(allmodules2$n, 0.25), q3 = quantile(allmodules2$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# repeat{

n_count <- length(allmodules3$n)
```

```
allmodules3 <- allmodules3 %>%
    mutate(q1 = quantile(allmodules3$n, 0.25), q3 = quantile(allmodules3$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# if(n_count == length(allmodules3$n)) break;

# }

allmodules_check <- as.data.frame(as.character(allmodules3$Module)) %>%
    dplyr::mutate(checker = TRUE)

colnames(allmodules_check) <- c("Module", "checker")

boxplot(allmodules3$n, ylab = "n")

moduleTraitCor2 <- as.data.frame(moduleTraitCor) %>%
    mutate(Module = row.names(moduleTraitCor)) %>%
    left_join(allmodules_check, by = "Module") %>%
    filter(checker == TRUE)

datTraits <- datTraits2

geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))

match_modules <- match(as.character(allmodules_check$Module), colnames(geneModuleMembership))

geneModuleMembership2 <- geneModuleMembership[, match_modules]

modNames = substring(names(geneModuleMembership2), 3)

MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

match_modules <- match(as.character(allmodules_check$Module), colnames(MMPvalue))

MMPvalue2 = MMPvalue[, match_modules]

names(geneModuleMembership2) = paste("MM", modNames, sep = "")
names(MMPvalue2) = paste("p.MM", modNames, sep = "")

# Save.image(file='./Inputs/general_files/wgcnafiles/allprelims.Rdata')
```

Figure 1H

```
load(file="./Inputs/general_files/wgcnafiles/allprelims.Rdata")

datTraits2 <- datTraits2 %>%
  dplyr::select(-disease)

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
```

```
  # Filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

hc.Traits <- hclust(dist(t(datTraits2)))

clust_order_traits <- hc.Traits$order

clust_positions <- as.data.frame(cbind(as.character(colnames(datTraits)[clust_order_traits]),as.numeric

colnames(clust_positions) <- c("trait","clust_order")

geneTraitSignificance <- as.data.frame(cor(datExpr,datTraits2, use = "p")) %>%
  dplyr::select(-dex2,-p53status2)

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules,-Module,-checker,-p53status2,-dex2) %>%
  # Filter(is.numeric(cor)) %>%
  # Filter(trait == trait_id) %>%
  dplyr::group_by(trait) %>%
  arrange(desc(cor)) %>%
  ungroup() %>%
  dplyr::group_by(modules) %>%
  dplyr::mutate(sumnumber = sum(cor,na.rm=T),
         n = n(),
         weight = sumnumber/n) %>%
  ungroup() %>%
  arrange(desc(weight))

traitpositions <- moduleTraitCor_trait_id %>%
  dplyr::select(modules) %>%
  distinct() %>%
  dplyr::mutate(position = row_number())

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  left_join(traitpositions,by="modules") %>%
  left_join(clust_positions,by="trait")

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  mutate(clust_position_order = as.numeric(moduleTraitCor_trait_id$clust_order))

p <- ggplot(data = moduleTraitCor_trait_id,mapping = aes(x = reorder(modules,position),y = reorder(trai
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("TUCRs") +
  xlab("Modules") +
  labs(fill = "cor") +
```

```
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(moduleTraitCor_trait_id$modules)),
    x = moduleTraitCor_trait_id$position,
    y = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/Figure1/figure1h.png",sep=""), width = 7, height = 3, dpi = 600)
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))
}

#####GO-TERM ANALYSIS

if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")

if (!require('GO.db')) BiocManager::install('GO.db')
library(GO.db)

#if (!require('AnnotationDBI')) BiocManager::install('AnnotationDBI')
#library(AnnotationDBI)

if (!require('org.Hs.eg.db')) BiocManager::install('org.Hs.eg.db')
library(org.Hs.eg.db)

if (!require('limma')) BiocManager::install('limma')
library(limma)

i <- 1
```

```
genelist <- geneTraitSignificance %>%
  mutate(genenames = row.names(geneTraitSignificance))

for(i in 1:length(unique(moduleTraitCor_trait_id$modules))){

  print(i)
  print(unique(moduleTraitCor_trait_id$modules)[i])

  module <- unique(moduleTraitCor_trait_id$modules)[i]
  #module <- "yellowgreen"
  column = match(module, modNames);
  moduleGenes = moduleColors==module;

  genelist2 <- as.data.frame(genelist[moduleGenes,])

  genelist2 <- genelist2 %>%
    separate(genenames,into=c("Alias","kibble"),sep="___")

  symbols <- as.character(genelist2$Alias)

  EntrezIDs <- mapIds(org.Hs.eg.db, symbols, 'ENTREZID', 'SYMBOL')
  allgenes <- cbind(symbols,EntrezIDs)
  allgenes <- allgenes[complete.cases(allgenes),]

  g <- goana(EntrezIDs)

  g_bp <- g %>%
    filter(Ont == "BP")
  topGO_bp <- topGO(g_bp) %>%
    mutate(log10_p = -log10(P.DE),module=module,color="red") %>%
    arrange(desc(log10_p))

  g_mf <- g %>%
    filter(Ont == "MF")
  topGO_mf <- topGO(g_mf) %>%
    mutate(log10_p = -log10(P.DE),module=module,color="blue") %>%
    arrange(desc(log10_p))

  topGO_all <- rbind(topGO_bp,topGO_mf) %>%
                       arrange(as.numeric(log10_p)) %>%
                       dplyr::mutate(roworder = row_number()) %>%
                       mutate(newTerm = substr(Term,1,80))

  p <- ggplot(topGO_all,aes(x=reorder(newTerm,roworder),y=as.numeric(log10_p),fill=module,color=color))
    geom_bar(stat="identity") + coord_flip() + scale_fill_identity() + scale_color_identity() + facet_w
    ggtitle(module) +
    labs(x="Go Term", y = "-log10(p-value)") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
```

```
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold",angle=0,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank(),
        strip.text.x = element_text(size = rel(2.2)))
  p

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",module,"_all_bar.png",sep=

if(module == "#004C54"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8a.png",sep=""),heigh

if(module == "#FFC0CB"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8b.png",sep=""),heigh

if(module == "#0000FF"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9a.png",sep=""),heigh

if(module == "#008080"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9b.png",sep=""),heigh

}
```

```
####Trait Heatmaps

figureorder <- 7

#moduleTraitCor2 <- moduleTraitCor2 %>%
#  dplyr::select(-disease)

h <- 2
```

```
for(h in 2:482){
skip_to_next <<- FALSE
print(h)
trait_id <- colnames(moduleTraitCor2)[h]
#trait_id <-"uc.15"

print(trait_id)

if(!dir.exists(paste(outputdir,"/TUCR_Database/",trait_id,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",trait_id,sep=""))
}

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
  filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

geneTraitSignificance = as.data.frame(cor(datExpr,datTraits, use = "p"))

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  dplyr::mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules) %>%
  dplyr::filter(trait == trait_id) %>%
  dplyr::arrange(cor) %>%
  dplyr::mutate(position = row_number())

traitmodules <- geneModuleMembership2 %>%
  dplyr::mutate(rownamer = row.names(geneModuleMembership2)) %>%
  separate(rownamer,into=c("rownamer","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  dplyr::filter(rownamer==trait_id)

traitmodules <- t(traitmodules)

traitmodules <- as.data.frame(cbind(row.names(traitmodules),traitmodules))

colnames(traitmodules) <- c("Module","ModuleMembership")

traitpvalues <- MMPvalue2 %>%
  dplyr::mutate(rownamer = row.names(MMPvalue2)) %>%
  separate(rownamer,into=c("rownamer","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  dplyr::filter(rownamer==trait_id)

traitpvalues <- t(traitpvalues)

traitpvalues <- as.data.frame(cbind(paste("MM",str_remove(row.names(traitpvalues),"p.MM"),sep=""),traitp
```

```
colnames(traitpvalues) <- c("Module","MMpvalue")
traitmodules <- traitmodules %>%
  left_join(allmodules_check, by= "Module") %>%
  left_join(traitpvalues,"Module") %>%
  # Filter(checker == TRUE) %>%
  dplyr::mutate(moduleColors = str_remove(Module,"MM")) %>%
  dplyr::filter(moduleColors != "rownamer")

df_correlations <- data.frame(matrix(ncol=2,nrow=0, dimnames=list(NULL, c("module","cor"))))

i <- 1

for(i in 1:length(unique(traitmodules$moduleColors))){

module = as.character(unique(traitmodules$moduleColors)[i])
#module = "red"
column = match(module, modNames)
moduleGenes = traitmodules$moduleColors==module;
correlation <- cor(abs(geneModuleMembership[moduleGenes, column]),
                   abs(geneTraitSignificance[moduleGenes, 1]))

rbinder <- c(module,correlation)
df_correlations <- rbind(df_correlations,rbinder)


}

colnames(df_correlations) <- c("module","cor")

df_correlations <- as.data.frame(df_correlations) %>%
  mutate(Module = paste("MM",module,sep=""))

traitmodules2 <- traitmodules %>%
  left_join(df_correlations,by = "Module") %>%
  dplyr::select(module,cor,ModuleMembership,MMpvalue) %>%
  dplyr::mutate(RankModule = percent_rank(1-as.numeric(ModuleMembership)),Rankcor = percent_rank(cor),M
  dplyr::group_by(module) %>%
  dplyr::mutate(totalscore = sum(as.numeric(RankModule),as.numeric(Rankcor),na.rm=TRUE)) %>%
  dplyr::ungroup() %>%
  dplyr::arrange(as.numeric(cor)) %>%
  dplyr::mutate(position = row_number(),cor = ifelse(MMpvalue2 >= 0.05,NA,cor))


p <- ggplot(data = traitmodules2,mapping = aes(x = as.character(trait_id),y = reorder(module,position),
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("modules") +
  xlab(trait_id) +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
```

```
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(traitmodules2$module)),
    y = traitmodules2$position,
    x = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/TUCR_Database/",trait_id,"/",figureorder,"_",trait_id,"_wgcna_modulecor

if(!dir.exists(paste(outputdir,"/Figure6_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure6_Supplementary/",sep=""))
}

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure7_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure7_Supplementary/",sep=""))
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure2/",sep=""))){
  dir.create(paste(outputdir,"/Figure2/",sep=""))
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2f.png",sep=""), width = 3, height = 7, dpi = 600)
}

####Trait Correlation Plots

traitmodules3 <- traitmodules2 %>%
  filter(as.numeric(MMpvalue) <= 0.05)



i <- 1
```

```
module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_6_",trait_id,"_",module,".png",sep

if(!dir.exists(paste(outputdir,"/Figure6_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure6_Supplementary/",sep=""))
}

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure7_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure7_Supplementary/",sep=""))
}

if(trait_id == "uc.15"){
```

```
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure2/",sep=""))){
  dir.create(paste(outputdir,"/Figure2/",sep=""))
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2f.png",sep=""), width = 3, height = 7, dpi = 600)
}

i <- 2

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_5_",trait_id,"_",module,".png",sep
```

```
if(!dir.exists(paste(outputdir,"/Figure6_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure6_Supplementary/",sep=""))
}

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure7_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure7_Supplementary/",sep=""))
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure2/",sep=""))){
  dir.create(paste(outputdir,"/Figure2/",sep=""))
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2f.png",sep=""), width = 3, height = 7, dpi = 600)
}

i <- 3

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
```

```
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_4_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_4.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_4.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_4.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
```

```
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_1_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_1.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_1.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_1.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)-1

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
```

```
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_2_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_2.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_2.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_2.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)-2

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
```

```
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_3_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_3.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_3.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_3.png",sep=""), width = 5, height = 5, dpi = 600)
}


}
```

Figure 1i

Identifying differentially expressed TUCRs with DESeq2

```
## read data

gbm_countfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_mergedcounts.txt",
    sep = "")

lgg_countfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_mergedcounts.txt",
    sep = "")

cortex_countfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_mergedcounts.txt",
    sep = "")


gbm_metadatafile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_tcga_metadata.csv",
    sep = "")

lgg_metadatafile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_tcga_metadata.csv",
    sep = "")

cortex_metadatafile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_gtex_metadata.csv",
```

```
    sep = "")


gbm_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_seqdepth_counts.csv",
    sep = "")

lgg_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_seqdepth_counts.csv",
    sep = "")

cortex_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_seqdepth_counts.csv",
    sep = "")


## Merge Data


if (!is.na(cortex_countfile)) {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    lgg_mergedcounts <- read.table(lgg_countfile, header = TRUE)

    cortex_normalcounts <- read.table(cortex_countfile, header = TRUE)

    cortex_normalcounts <- cortex_normalcounts[, 9:ncol(cortex_normalcounts)]

    gbm_mergedcounts <- cbind(gbm_mergedcounts, cortex_normalcounts) %>%
        distinct()

    lgg_mergedcounts <- cbind(lgg_mergedcounts, cortex_normalcounts) %>%
        distinct()

    rm(cortex_normalcounts)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    lgg_metadata <- read_csv(file = lgg_metadatafile)

    cortex_metadata <- read_csv(file = cortex_metadatafile)

    gbm_metadata <- rbind(gbm_metadata, cortex_metadata)

    lgg_metadata <- rbind(lgg_metadata, cortex_metadata)

    rm(cortex_metadata)

    gbm_seqdepth <- read.csv(file = gbm_seqdepthfile)

    lgg_seqdepth <- read.csv(file = lgg_seqdepthfile)

    cortex_seqdepth <- read.csv(file = cortex_seqdepthfile)

    gbm_seqdepth <- rbind(gbm_seqdepth, cortex_seqdepth)
```

```
    lgg_seqdepth <- rbind(lgg_seqdepth, cortex_seqdepth)

    rm(cortex_seqdepth)

} else {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    lgg_mergedcounts <- read.table(lgg_countfile, header = TRUE)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    lgg_metadata <- read_csv(file = lgg__metadatafile)

    gbm_seqdepth <- read.csv(file = gbm__seqdepthfile)

    lgg_seqdepth <- read.csv(file = lgg__seqdepthfile)
}

gbm_mergedcounts_deseq2 <- gbm_mergedcounts %>%
    filter(tag == "TUCR" & annot != "random")

lgg_mergedcounts_deseq2 <- lgg_mergedcounts %>%
    filter(tag == "TUCR" & annot != "random")

gbm_mergedcounts_deseq2_countsonly <- gbm_mergedcounts_deseq2[, 9:ncol(gbm_mergedcounts_deseq2)]

lgg_mergedcounts_deseq2_countsonly <- lgg_mergedcounts_deseq2[, 9:ncol(lgg_mergedcounts_deseq2)]

rownames(gbm_mergedcounts_deseq2_countsonly) <- gbm_mergedcounts_deseq2$id

rownames(lgg_mergedcounts_deseq2_countsonly) <- lgg_mergedcounts_deseq2$id

gbm_mergedcounts_deseq2_info <- gbm_mergedcounts_deseq2[, 1:8]

lgg_mergedcounts_deseq2_info <- lgg_mergedcounts_deseq2[, 1:8]

gbm_mergedcounts_deseq2_info$length <- (gbm_mergedcounts_deseq2$end - gbm_mergedcounts_deseq2$start)/10

lgg_mergedcounts_deseq2_info$length <- (lgg_mergedcounts_deseq2$end - lgg_mergedcounts_deseq2$start)/10

dds <- DESeqDataSetFromMatrix(countData = gbm_mergedcounts_deseq2_countsonly, colData = gbm_metadata,
    design = ~dex)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)

write_csv(res, file = paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs.csv",
    sep = ""))

dds <- DESeqDataSetFromMatrix(countData = lgg_mergedcounts_deseq2_countsonly, colData = lgg_metadata,
```

```
    design = ~dex)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)

write_csv(res, file = paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs.csv",
    sep = ""))
```

```
### Volcano plot

volcanoplot_expression <- function (res,
                          genes = "all",
                          title = "Deregulated TUCRs in TCGA Gliomas",
                          output = "",
                          height = 7,
                          width = 14,
                          dpi = 600,
                          annot_filter = ""){
  #res <- res_TUCR
  #genes <- "all"
  #annot_filter = ""
  #genes = c("uc.110")
  #title = "Deregulated Genes"

  res <- res %>%
  mutate(dereg = ifelse(log2FoldChange >=1, "upregulated",
                    ifelse(log2FoldChange <=-1,"downregulated","unchanged")),
         deregcount = ifelse(log2FoldChange >=1 & padj <= 0.05, 1,
                    ifelse(log2FoldChange <=-1 & padj <= 0.05,-1,0))) %>%
  dplyr::filter(padj != 0 | alias == "uc.110") %>%
  dplyr::group_by(alias) %>%
  dplyr::mutate(sum = sum(deregcount,na.rm=TRUE)) %>%
  ungroup() %>%
  dplyr::mutate(deregcategory = ifelse(sum == 2, "Up-Both",
                                ifelse(sum == -2, "Down-Both",
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM","Up-GBM",
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", "Up-LGG"
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", "Down
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", "Down
                color = ifelse(sum == 2, paper_darkpink,
                                ifelse(sum == -2, paper_lightblue,
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM",paper_red,
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", paper_pi
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", paper
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", paper
                                ifelse(disease == "GBM",paper_black,paper_gray)))))))),
                newdisease = ifelse(str_detect(deregcategory,"Both"),"BOTH",disease),
                diseasefilter = ifelse((disease == "LGG" & newdisease == "BOTH"),0,1))

  res_summary <- res %>%
    group_by(color) %>%
```

```
    dplyr::summarize(n = n())

if(annot_filter == ""){}else{
  res %>%
    filter(annot == annot_filter)
}


res_110_up <- res %>%
   filter(color != paper_gray & color != paper_black & log2FoldChange > 0 | alias == "uc.110") %>%
   dplyr::arrange(desc(log2FoldChange)) %>%
   dplyr::group_by(disease) %>%
   dplyr::mutate(order_rank= row_number()) %>%
   mutate(maxrank = max(order_rank,na.rm=TRUE)) %>%
   ungroup() %>%
   filter(order_rank <= 5 | alias == "uc.110")

res_110_down <- res %>%
   filter(color != paper_gray & color != paper_black  & log2FoldChange < 0 | alias == "uc.110") %>%
   dplyr::arrange(log2FoldChange) %>%
   dplyr::group_by(disease) %>%
   dplyr::mutate(order_rank= row_number()) %>%
   mutate(maxrank = max(order_rank,na.rm=TRUE)) %>%
   ungroup() %>%
   filter(order_rank <= 5 | alias == "uc.110")

p <- ggplot(res) +
        #geom_point(aes(x=log2FoldChange, y=-log10(pvalue),col=color)) +
        #geom_hline(yintercept = 1.30,linetype = 2) +
        geom_point(aes(x=log2FoldChange,y=-log10(padj),col=color)) +
        geom_hline(yintercept = -log10(0.05),linetype = 2,size=0.5) +
        geom_vline(xintercept = -1,linetype = 2,size=0.5) +
        geom_vline(xintercept = 1,linetype = 2,size=0.5) +
        scale_color_identity(guide = "deregcategory", labels = res$deregcategory, breaks = res$color) +
        #ggtitle(title) +
        guides(color = guide_legend(override.aes = list(size=5))) +
        geom_text_repel(data= res_110_up,aes(x=log2FoldChange,y=-log10(padj),color=paper_black, label =
        geom_text_repel(data= res_110_down,aes(x=log2FoldChange,y=-log10(padj),color=paper_black, label
        labs(x = "log2 fold change of TUCR in tumors",
        #y = "-log10 adjusted p-value",
        y = "-log10 of adjusted p-value",
        color = "Legend") +
        theme(plot.title = element_text(size = rel(2.0), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_text(size=10)) +
  coord_cartesian(clip = "off") +
  facet_wrap(~disease,scales="free",ncol=1)
```

```
p

ggsave(plot = print(p),filename = output, width = width, height = height, dpi = 600)

}
```

```
res_TUCR_LGG <- read_csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs.csv",
    sep = ""), col_names = TRUE)

res_TUCR_GBM <- read_csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs.csv",
    sep = ""), col_names = TRUE)

tucr_annot <- read_delim("Inputs/general_files/bedfiles/hg38.ultraconserved.bed",
    col_names = TRUE, delim = "\t")

res_TUCR_LGG_2 <- res_TUCR_LGG %>%
    separate(row, into = c("alias", "kibble"), sep = "___") %>%
    dplyr::select(-kibble) %>%
    left_join(tucr_annot, by = "alias") %>%
    filter(!is.na(chrom)) %>%
    dplyr::select(alias, annot, log2FoldChange, padj) %>%
    mutate(disease = "LGG")

res_TUCR_GBM_2 <- res_TUCR_GBM %>%
    separate(row, into = c("alias", "kibble"), sep = "___") %>%
    dplyr::select(-kibble) %>%
    left_join(tucr_annot, by = "alias") %>%
    filter(!is.na(chrom)) %>%
    dplyr::select(alias, annot, log2FoldChange, padj) %>%
    mutate(disease = "GBM")

res_TUCR <- rbind(res_TUCR_GBM_2, res_TUCR_LGG_2) %>%
    filter(annot == "intergenic")

volcanoplot_expression(res_TUCR, genes = "all", output = paste(outputdir, "/Figure1/figure1i.png",
    sep = ""), height = 5, width = 5, dpi = 600, annot = "")
```

TUCR, uc.110, is highly upregulated in gliomas and is predicted to bind nucleic acids.

The expression and deregulation of intergenic TUCRs is of particular interest as they may represent novel lncRNAs due to their similar expression levels, genomic location, and lack of coding potential.[2] These TUCRs are also easier to study experimentally; they are often thousands of kilobases (kb) from the nearest protein-coding gene and likely function in a manner that is independent of a "host gene". Because of this, we focused on intergenic TUCRs for our experimental studies. Of the deregulated intergenic TUCRs in GBM and in LGG (Figure 1i), we found that uc.110 is the most upregulated as compared to normal brain; 30-fold in GBM and 61.4-fold in LGG (Figure 2A). It has near binary expression; it is rarely expressed at all in normal brain but is very highly expressed in GBM and LGG (Figure 2B). Due to its high expression, we hypothesized that this TUCR is functioning as a tumor enhancer.

Since many TUCRs exist as a part of a larger transcript [2], we first determined the sequence of the uc.110 full transcript. We utilized machine learning and de novo transcript reassembly using TCGA and GTEx RNA-seq data to reconstruct RNA-Seq transcripts in the absence of a reference genome (Supplementary Figure 9A). [35] We identified a 2,158 nucleotide (nt) long RNA molecule that contains the 243 nucleotide

(nt) uc.110 ultraconserved sequence (Figure 2C) as a novel transcript. We confirmed the existence of this transcript experimentally using PCR amplifications and sequencing (Supplementary Figure 9B).

After identifying the sequence for the full uc.110 transcript (Supplementary Figure 9C) and using qPCR to confirm its independence from its "host" gene, GBX2 (Supplementary Figure 9D-9E), as previously published by Mestdagh et al, we utilized our WGCNA workflow to identify genes and modules (Figure 2D) that are significant to this transcript. Of note, one of the top modules for uc.110 by module association is the #004C54 module, which represents genes that are involved in nucleic acid and protein binding (Supplementary Figure 7). This is a published function for some TUCRs. [2] Genes that are members of these modules are positively coregulated with uc.110 (Figure 2E). Based on these findings, we hypothesized that uc.110 may be operating as a tumor enhancing RNA-binding molecule. We also performed similar analyses for all 481 TUCRs to identify potential functional roles for each TUCR in gliomas. Examples of an tumor enhancing TUCR (uc.2, Supplementary Figures 6-2) and a tumor suppressor TUCR (uc.15, Supplementary Figure 6-15) are depicted in this manuscript, while the rest of the 481 TUCRs are provided as supplementary materials (Supplementary Figures 6)

#uc.110 has tumor enhancing effects in GBM. To determine the function of uc.110 in GBM, we first used qPCR to investigate the expression of uc.110 using RNA in our banked tumor samples compared to normal brain cortex and cell lines compared to normal human astrocytes. We independently confirmed the results from our TCGA analysis showing uc.110 is highly upregulated in GBM tumors (Figure 3A, 3B). We then designed two siRNAs that target separate regions on the uc.110 RNA, one that begins at nucleotide 96/243 (si-uc.110-1) and one that begins at nucleotide 195/243 (si-uc.110-2), as well as a scrambled control (si-SCR) (Supplementary Figure 10A). We generated stable A172 and U251 GBM cell lines that express uc.110 (LV-uc.110) or the empty expression vector (LV-pCDH). We subjected these cell lines to siRNA transfection and assessed the effects on cell counting, survival and invasion assays (Supplementary Figure 10B). We used qPCR to show that uc.110 is generally, though not uniformly, upregulated in GBM cells (Supplementary Figure 10C,10D,10E). Based on these data, we prioritized the use of cell lines that overexpress uc.110 (A172, U251) for knockdown experiments, and cells that express low levels of uc.110 (U87, GSC-28) for overexpression experiments. We confirmed that siRNA targeting of uc.110 lead to knockdown of uc.110 expression in A172 and U251 cells. (Figure 3C) We also confirmed that LV-uc.110 overexpresses uc.110, and that this overexpression rescues uc.110 bioavailability in A172 and U251 cells (Figure 3C).

Next, we performed cell counting assays [20, 37-39] to determine the effects of uc.110 knockdown and rescue on cell accumulation. When we reduced uc.110 expression, we reduced cell accumulation in A172 and U251 cells (Figure 3D). When we rescue uc.110 bioavailability, the cell accumulation phenotype is restored in A172 and U251 cells (Figure 3E). We then used AlamarBlue [40, 41] to measure cell viability. When we reduce uc.110 expression, A172 and U251 cell viability is reduced. We were able to rescue this phenotype by increasing uc.110 bioavailability (Figure 3F). We observed a similar phenotype in a glioma stem cell line that overexpresses uc.110 (GSC-34, Figure 3G).

We then investigated the invasive potential of uc.110 using a transwell invasion assay. [42-44] Knockdown of uc.110 reduced A172 and U251 cell invasion through a collagen IV matrix (Figure 3H). When uc.110 bioavailability was increased, a partial recovery of the phenotype is observed (Supplementary Figure 10F). Lastly, we overexpressed uc.110 in U87 and GSC-28 cells (Figure 3i) and determined that this leads to increased cell accumulation compared to the empty vector after 7 days (Figure 3J) in U87 and GSC-28 cells. These data show that uc.110 has tumor enhancing effects in GBM cells and stem cells.

Figure 2.

```
dir.create(paste(outputdir, "/Figure2/", sep = ""))
```

After determining that uc.110 displays a tumor enhancer phenotype in vitro, we sought to determine whether this effect is recapitulated in vivo. U251 GBM cells were transfected with si-uc.110-1 or si-uc.110-2. After 2 days, these cells were implanted into immunodeficient mice using intracranial injection (Supplementary Figure 11A). [37, 38, 45, 46] Tumor growth was monitored by MRI and mouse survival was observed over a period of 70 days. Mice that were xenografted with U251 cells that were transfected with si-uc.110-1 and

si-uc.110-2 expression developed smaller tumors, as depicted, and quantified by MRI (Figure 4A, 4B). The mice that received si-uc.110 also displayed better overall survival than mice that received scrambled control siRNA cells (Figure 4C). a

Figure 2A

Generate FC Plots for each TUCR (GBM)

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 1

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)
```

```
  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))
}

if(!is.na(filterannot)){
  mergedcounts_trimmed <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random")
}

posdata <- mergedcounts_trimmed[,1:8] %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

mergedcounts_trimmed <- mergedcounts_trimmed[,9:length(colnames(mergedcounts_trimmed))]

match_colnames <- match(as.character(colnames(mergedcounts_trimmed)),as.character(metadata$survid))

mergedcounts_trimmed <- as.matrix(mergedcounts_trimmed)
rownames(mergedcounts_trimmed) <- posdata$alias

dds <-
  DESeqDataSetFromMatrix(countData = mergedcounts_trimmed,
                                colData = metadata,
                                design = ~dex)

dds <-
  DESeq(dds)

res <-
  results(dds, tidy=TRUE)

res <-
  as_tibble(res)
```

```r
colnames(res)[1] <- "alias"

#variablename <- which(as.character(colnames(tablename) %in% as.character(row.names(table2name))))

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x){
  cond <- factor(ifelse(seq(1,dim(x)[2],1) %in% t_index, 1,  0))
  d <- model.matrix(~1+cond)
  x <- t(apply(x,1,as.numeric))
  ex <- voom(x,d,plot=F)
  return(ex$E)
}

count_vm <- vm(mergedcounts_trimmed)

colnames(count_vm) <- colnames(mergedcounts_trimmed)

#if(is.sequential(match_colnames)){
#  print("metadata rows match countfile columns")
#  colnames(count_vm) <- metadata$id
#}else{
#  "metadata rows do not match countfile columns"
#}

scal <- function(x,y){
  median_n <- rowMedians(y)  # mean of normal
  sd_n <- apply(y,1,sd)  # SD of normal
  # z score as (value - mean normal)/SD normal
  res <- matrix(nrow=nrow(x), ncol=ncol(x))
  colnames(res) <- colnames(x)
  rownames(res) <- rownames(x)
  for(i in 1:dim(x)[1]){
    for(j in 1:dim(x)[2]){
      res[i,j] <- (x[i,j]-median_n[i])/sd_n[i]
    }
  }
  return(res)
}

z_rna <- scal(count_vm,count_vm[,n_index])

z_rna2 <- cbind(posdata,z_rna)

write.csv(as.data.frame(z_rna2),file=paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",diseas

fc_counts <- cbind(posdata,z_rna) %>%
  dplyr::select(-chrom,-start,-end,-strand)

fc.dotplot <- fc_counts %>%
  gather(key = "Sample", value = "ZScore",-id,-alias,-tag,-annot) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
```

```
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

fc.dotplot <- fc.dotplot %>%
  left_join(res,by="alias") %>%
  mutate(DESeq2 = ifelse(dex=="normal",0,log2FoldChange)) %>%
  mutate(fill = ifelse(DESeq2 >= 1,paper_red,
                ifelse(DESeq2 <= -1,paper_green,paper_gray))) %>%
  dplyr::select(-Sample,-Key,-id.y,-string,-log2FoldChange) %>%
  filter(!is.na(dex))

i <- 1

for(i in 1:length(unique(fc.dotplot$id.x))){

filterdotplot <- as.character(unique(fc.dotplot$id.x[i]))

print(i)

fc.dotplot2 <- fc.dotplot %>%
  filter(id.x == filterdotplot)


print(fc.dotplot2$alias[1])
TUCRname <- fc.dotplot2$alias[1]

if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}

if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_FC_


padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()
```

```
p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_F


if(file.exists(paste(outputdir,"/Figure2/figure2a.png",sep = ""))){}else{

fc.dotplot2 <- fc.dotplot %>%
  filter(str_detect(id.x,"uc.110"))


print("uc.110")
TUCRname <- "uc.110"

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
```

```
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=paste(outputdir,"/Figure2/figure2a.png",sep = ""),height=7,width=5)}
```

Figure 2B

Generate FC Plots for each TUCR (LGG)

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 2

# read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

# Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()
```

```
  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))
}

if(!is.na(filterannot)){
  mergedcounts_trimmed <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random")
}

posdata <- mergedcounts_trimmed[,1:8] %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

mergedcounts_trimmed <- mergedcounts_trimmed[,9:length(colnames(mergedcounts_trimmed))]

match_colnames <- match(as.character(colnames(mergedcounts_trimmed)),as.character(metadata$survid))

mergedcounts_trimmed <- as.matrix(mergedcounts_trimmed)
```

```
rownames(mergedcounts_trimmed) <- posdata$alias

dds <-
  DESeqDataSetFromMatrix(countData = mergedcounts_trimmed,
                                colData = metadata,
                                design = ~dex)

dds <-
  DESeq(dds)

res <-
  results(dds, tidy=TRUE)

res <-
  as_tibble(res)

colnames(res)[1] <- "alias"

#variablename <- which(as.character(colnames(tablename) %in% as.character(row.names(table2name))))

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x){
  cond <- factor(ifelse(seq(1,dim(x)[2],1) %in% t_index, 1,  0))
  d <- model.matrix(~1+cond)
  x <- t(apply(x,1,as.numeric))
  ex <- voom(x,d,plot=F)
  return(ex$E)
}

count_vm <- vm(mergedcounts_trimmed)

colnames(count_vm) <- colnames(mergedcounts_trimmed)

#if(is.sequential(match_colnames)){
#  print("metadata rows match countfile columns")
#  colnames(count_vm) <- metadata$id
#}else{
#  "metadata rows do not match countfile columns"
#}

scal <- function(x,y){
  median_n <- rowMedians(y)  # mean of normal
  sd_n <- apply(y,1,sd)  # SD of normal
  # z score as (value - mean normal)/SD normal
  res <- matrix(nrow=nrow(x), ncol=ncol(x))
  colnames(res) <- colnames(x)
  rownames(res) <- rownames(x)
  for(i in 1:dim(x)[1]){
    for(j in 1:dim(x)[2]){
      res[i,j] <- (x[i,j]-median_n[i])/sd_n[i]
    }
```

```
  }
  return(res)
}

z_rna <- scal(count_vm,count_vm[,n_index])

z_rna2 <- cbind(posdata,z_rna)

write.csv(as.data.frame(z_rna2),file=paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",diseas

fc_counts <- cbind(posdata,z_rna) %>%
  dplyr::select(-chrom,-start,-end,-strand)

fc.dotplot <- fc_counts %>%
  gather(key = "Sample", value = "ZScore",-id,-alias,-tag,-annot) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

fc.dotplot <- fc.dotplot %>%
  left_join(res,by="alias") %>%
  mutate(DESeq2 = ifelse(dex=="normal",0,log2FoldChange)) %>%
  mutate(fill = ifelse(DESeq2 >= 1,paper_red,
                ifelse(DESeq2 <= -1,paper_green,paper_gray))) %>%
  dplyr::select(-Sample,-Key,-id.y,-string,-log2FoldChange) %>%
  filter(!is.na(dex))

i <- 1

for(i in 1:length(unique(fc.dotplot$id.x))){

filterdotplot <- as.character(unique(fc.dotplot$id.x[i]))

print(i)

fc.dotplot2 <- fc.dotplot %>%
  filter(id.x == filterdotplot)


print(fc.dotplot2$alias[1])
TUCRname <- fc.dotplot2$alias[1]

if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}

if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_FC
```

```
padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_F


if(file.exists(paste(outputdir,"/Figure2/figure2b.png",sep = ""))){}else{

fc.dotplot2 <- fc.dotplot %>%
  filter(str_detect(id.x,"uc.110"))


print("uc.110")
TUCRname <- "uc.110"

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)
```

```
padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=paste(outputdir,"/Figure2/figure2b.png",sep = ""),height=7,width=5)}
```

Figure 2C

Generate tpm Box Plots for each TUCR (GBM)

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 3

# read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv
```

```
t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

# Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))
```

```
}

#if(makeRPKboxplots == TRUE){

if(!is.na(filterannot)){
  tpmcounts <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random") %>%
  mutate(length = end.x - start.x)
}else{
tpmcounts <- mergedcounts %>%
  mutate(length = end.x - start.x)}

tpmcounts.info <- tpmcounts %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

tpmcounts <- tpmcounts %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-id,-strand.x,-tag.x,-annot.x,-alias.x)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1000000

tpm <- function(counts,len,dep){
  #x <- tpmcounts/genelength
  #x2 <- t(t(x)/(seqdepth2))
  x <- counts/len
  return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts,genelength,seqdepth2)

tpm.df2 <- cbind(tpmcounts.info,tpm.df)

tpm.median <- as.data.frame(tpm.df)
tpm.median <- tpm.median %>%
  dplyr::summarize(median_tpm = rowMedians(tpm.df)) %>%
  dplyr::mutate(countif = ifelse(median_tpm >=1,1,0),proportion = sum(countif,na.rm = TRUE)/n())
tpm.median <- cbind(tpmcounts.info,tpm.median)
tpm.median <- tpm.median %>% dplyr::select(id,median_tpm,countif,proportion)

proportion.df <- round(as.numeric(tpm.median$proportion[1]),3)*100

write.csv(tpm.df2,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_tpms_allTUCRs.c

tpm.median.write <- tpm.median[,-3]

tpm.median.write <- tpm.median.write[,-3]

tpm.median.write <- tpmcounts.info %>%
  left_join(tpm.median.write,by="id")
```

```
write.csv(tpm.median.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_median

tpm.dotplot <- tpm.df2[,9:ncol(tpm.df2)]
TUCRids <- as.character(tpm.df2$id)
annot <- as.character(tpm.df2$annot)
length <- as.character(tpm.df2$length)
tpm.dotplot <- cbind(TUCRids,annot,length,tpm.dotplot)
tpm.dotplot <- tpm.dotplot %>%
  gather(key = "Sample", value = "TPM",-TUCRids,-annot,-length) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

tpm.dotplot <- tpm.dotplot[complete.cases(tpm.dotplot),]

median(tpm.dotplot$TPM,na.rm=TRUE)

i <- 1
for(i in 1:length(TUCRids)){
#TUCR <- "uc.110"
TUCR <- tpm.df2$id[i]
TUCRname <- as.character(tpm.df2$alias[i])
print(i)
print(TUCRname)
if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}
if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tp
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(TUCRids == TUCR)

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
```

```
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                      "Transcripts per kilobase million (TPM)",
                      "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tpm
}}

TUCR <- "uc.110"
TUCRname <- "uc.110"
print(i)
print(TUCRname)
if(file.exists(paste(outputdir,"/Figure2/figure2c.png",sep = ""))){}else{
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(str_detect(TUCRids,TUCRname))

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                      "Transcripts per kilobase million (TPM)",
                      "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/Figure2/figure2c.png",sep = ""),height=7,width=5)

}
```

Figure 2D

Generate tpm Box Plots for each TUCR

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 4
```

```r
# read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

# Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))
}

#if(makeRPKboxplots == TRUE){

if(!is.na(filterannot)){
  tpmcounts <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random") %>%
  mutate(length = end.x - start.x)
}else{
tpmcounts <- mergedcounts %>%
  mutate(length = end.x - start.x)}

tpmcounts.info <- tpmcounts %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

tpmcounts <- tpmcounts %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-id,-strand.x,-tag.x,-annot.x,-alias.x)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1000000

tpm <- function(counts,len,dep){
  #x <- tpmcounts/genelength
  #x2 <- t(t(x)/(seqdepth2))
  x <- counts/len
  return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts,genelength,seqdepth2)

tpm.df2 <- cbind(tpmcounts.info,tpm.df)

tpm.median <- as.data.frame(tpm.df)
tpm.median <- tpm.median %>%
  dplyr::summarize(median_tpm = rowMedians(tpm.df)) %>%
  dplyr::mutate(countif = ifelse(median_tpm >=1,1,0),proportion = sum(countif,na.rm = TRUE)/n())
tpm.median <- cbind(tpmcounts.info,tpm.median)
tpm.median <- tpm.median %>% dplyr::select(id,median_tpm,countif,proportion)

proportion.df <- round(as.numeric(tpm.median$proportion[1]),3)*100
```

```
tpm.df.write <- cbind(tpmcounts.info,tpm.df2)

write.csv(tpm.df.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_tpms_allTU

tpm.median.write <- tpm.median[,-3]

tpm.median.write <- tpm.median.write[,-3]

tpm.median.write <- tpmcounts.info %>%
  left_join(tpm.median.write,by="id")

write.csv(tpm.median.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_median

tpm.dotplot <- tpm.df2[,9:ncol(tpm.df2)]
TUCRids <- as.character(tpm.df2$id)
annot <- as.character(tpm.df2$annot)
length <- as.character(tpm.df2$length)
tpm.dotplot <- cbind(TUCRids,annot,length,tpm.dotplot)
tpm.dotplot <- tpm.dotplot %>%
  gather(key = "Sample", value = "TPM",-TUCRids,-annot,-length) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

tpm.dotplot <- tpm.dotplot[complete.cases(tpm.dotplot),]

median(tpm.dotplot$TPM,na.rm=TRUE)

i <- 1
for(i in 1:length(TUCRids)){
#TUCR <- "uc.110"
TUCR <- tpm.df2$id[i]
TUCRname <- as.character(tpm.df2$alias[i])
print(i)
print(TUCRname)
if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}
if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tp
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(TUCRids == TUCR)

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
```

```
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                     "Transcripts per kilobase million (TPM)",
                     "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tpm
}}

TUCR <- "uc.110"
TUCRname <- "uc.110"
print(i)
print(TUCRname)
if(file.exists(paste(outputdir,"/Figure2/figure2d.png",sep = ""))){}else{
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(str_detect(TUCRids,TUCRname))

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                     "Transcripts per kilobase million (TPM)",
                     "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/Figure2/figure2d.png",sep = ""),height=7,width=5)
```

```
}
```

Figure 2E

```
Figure2_copy <- list.files("./Inputs/Figure2/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure2/", "Figure2e.png", sep = ""))) {
    file.copy(paste0("./Inputs/Figure2/pregenerated_figures/", Figure2_copy, sep = ""),
        paste("./", outputdir, "/Figure2/", sep = ""))
}
```

Figure 2F and 2G

```
# Load the WGCNA package
library(WGCNA)
# The following setting is important, do not omit.
options(stringsAsFactors = FALSE)
# Load the expression and trait data saved in the first part
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-01-dataInput.RData")
# The variable lnames contains the names of loaded variables.
lnames
# Load network data saved in the second part.
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-02-networkConstruction-auto.RData")
lnames
```

filter out modules that are particularly large.

```
# Define numbers of genes and samples
nGenes = ncol(datExpr)
nSamples = nrow(datExpr)
# Recalculate MEs with color labels
MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes

datExpr2 = as.data.frame(datExpr)
MEs = orderMEs(MEs0)
datTraits2 <- as.data.frame(datTraits2)
moduleTraitCor = cor(MEs, datTraits2, use = "p")
moduleTraitPvalue = corPvalueStudent(moduleTraitCor, nSamples)
```

```
geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))
MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

allmodules <- as.data.frame(t(MMPvalue))

allmodules2 <- allmodules %>%
    dplyr::mutate(Module = row.names(allmodules)) %>%
    gather(key = "Gene", value = "pvalue", -Module) %>%
    group_by(Gene) %>%
    mutate(maxmodule = min(pvalue, na.rm = TRUE)) %>%
    filter(pvalue == maxmodule) %>%
    group_by(Module) %>%
    dplyr::summarize(n = n())

allmodules3 <- allmodules2 %>%
```

```
    mutate(q1 = quantile(allmodules2$n, 0.25), q3 = quantile(allmodules2$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# repeat{

n_count <- length(allmodules3$n)

allmodules3 <- allmodules3 %>%
    mutate(q1 = quantile(allmodules3$n, 0.25), q3 = quantile(allmodules3$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# if(n_count == length(allmodules3$n)) break;

# }

allmodules_check <- as.data.frame(as.character(allmodules3$Module)) %>%
    dplyr::mutate(checker = TRUE)

colnames(allmodules_check) <- c("Module", "checker")

boxplot(allmodules3$n, ylab = "n")

moduleTraitCor2 <- as.data.frame(moduleTraitCor) %>%
    mutate(Module = row.names(moduleTraitCor)) %>%
    left_join(allmodules_check, by = "Module") %>%
    filter(checker == TRUE)

datTraits <- datTraits2

geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))

match_modules <- match(as.character(allmodules_check$Module), colnames(geneModuleMembership))

geneModuleMembership2 <- geneModuleMembership[, match_modules]

modNames = substring(names(geneModuleMembership2), 3)

MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

match_modules <- match(as.character(allmodules_check$Module), colnames(MMPvalue))

MMPvalue2 = MMPvalue[, match_modules]

names(geneModuleMembership2) = paste("MM", modNames, sep = "")
names(MMPvalue2) = paste("p.MM", modNames, sep = "")

# Save.image(file='./Inputs/general_files/wgcnafiles/allprelims.Rdata')
```

```
load(file="./Inputs/general_files/wgcnafiles/allprelims.Rdata")
```

```r
datTraits2 <- datTraits2 %>%
  dplyr::select(-disease)

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
  # Filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

hc.Traits <- hclust(dist(t(datTraits2)))

clust_order_traits <- hc.Traits$order

clust_positions <- as.data.frame(cbind(as.character(colnames(datTraits)[clust_order_traits]),as.numeric

colnames(clust_positions) <- c("trait","clust_order")

geneTraitSignificance <- as.data.frame(cor(datExpr,datTraits2, use = "p")) %>%
  dplyr::select(-dex2,-p53status2)

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules,-Module,-checker,-p53status2,-dex2) %>%
  # Filter(is.numeric(cor)) %>%
  # Filter(trait == trait_id) %>%
  dplyr::group_by(trait) %>%
  arrange(desc(cor)) %>%
  ungroup() %>%
  dplyr::group_by(modules) %>%
  dplyr::mutate(sumnumber = sum(cor,na.rm=T),
         n = n(),
         weight = sumnumber/n) %>%
  ungroup() %>%
  arrange(desc(weight))

traitpositions <- moduleTraitCor_trait_id %>%
  dplyr::select(modules) %>%
  distinct() %>%
  dplyr::mutate(position = row_number())

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  left_join(traitpositions,by="modules") %>%
  left_join(clust_positions,by="trait")

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  mutate(clust_position_order = as.numeric(moduleTraitCor_trait_id$clust_order))

p <- ggplot(data = moduleTraitCor_trait_id,mapping = aes(x = reorder(modules,position),y = reorder(trai
```

```
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("TUCRs") +
  xlab("Modules") +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(moduleTraitCor_trait_id$modules)),
    x = moduleTraitCor_trait_id$position,
    y = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/Figure1/figure1h.png",sep=""), width = 7, height = 3, dpi = 600)
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))
}

#####GO-TERM ANALYSIS

if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")

if (!require('GO.db')) BiocManager::install('GO.db')
library(GO.db)

#if (!require('AnnotationDBI')) BiocManager::install('AnnotationDBI')
#library(AnnotationDBI)

if (!require('org.Hs.eg.db')) BiocManager::install('org.Hs.eg.db')
library(org.Hs.eg.db)
```

```
if (!require('limma')) BiocManager::install('limma')
library(limma)

i <- 1

genelist <- geneTraitSignificance %>%
  mutate(genenames = row.names(geneTraitSignificance))

for(i in 1:length(unique(moduleTraitCor_trait_id$modules))){

  print(i)
  print(unique(moduleTraitCor_trait_id$modules)[i])

  module <- unique(moduleTraitCor_trait_id$modules)[i]
  #module <- "yellowgreen"
  column = match(module, modNames);
  moduleGenes = moduleColors==module;

  genelist2 <- as.data.frame(genelist[moduleGenes,])

  genelist2 <- genelist2 %>%
    separate(genenames,into=c("Alias","kibble"),sep="___")

  symbols <- as.character(genelist2$Alias)

  EntrezIDs <- mapIds(org.Hs.eg.db, symbols, 'ENTREZID', 'SYMBOL')
  allgenes <- cbind(symbols,EntrezIDs)
  allgenes <- allgenes[complete.cases(allgenes),]

  g <- goana(EntrezIDs)

  g_bp <- g %>%
    filter(Ont == "BP")
  topGO_bp <- topGO(g_bp) %>%
    mutate(log10_p = -log10(P.DE),module=module,color="red") %>%
    arrange(desc(log10_p))

  g_mf <- g %>%
    filter(Ont == "MF")
  topGO_mf <- topGO(g_mf) %>%
    mutate(log10_p = -log10(P.DE),module=module,color="blue") %>%
    arrange(desc(log10_p))

  topGO_all <- rbind(topGO_bp,topGO_mf) %>%
                       arrange(as.numeric(log10_p)) %>%
                       dplyr::mutate(roworder = row_number()) %>%
                       mutate(newTerm = substr(Term,1,80))

  p <- ggplot(topGO_all,aes(x=reorder(newTerm,roworder),y=as.numeric(log10_p),fill=module,color=color))
    geom_bar(stat="identity") + coord_flip() + scale_fill_identity() + scale_color_identity() + facet_w
    ggtitle(module) +
    labs(x="Go Term", y = "-log10(p-value)") +
  theme(panel.grid.major = element_blank(),
```

```
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold",angle=0,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank(),
        strip.text.x = element_text(size = rel(2.2)))
  p

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",module,"_all_bar.png",sep=

if(module == "#004C54"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8a.png",sep=""),heigh

if(module == "#FFC0CB"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8b.png",sep=""),heigh

if(module == "#0000FF"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9a.png",sep=""),heigh

if(module == "#008080"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9b.png",sep=""),heigh

}
```

```
####Trait Heatmaps

figureorder <- 7
```

```
#moduleTraitCor2 <- moduleTraitCor2 %>%
#  dplyr::select(-disease)

h <- 2

for(h in 2:482){
skip_to_next <<- FALSE
print(h)
trait_id <- colnames(moduleTraitCor2)[h]
#trait_id <-"uc.15"

print(trait_id)

if(!dir.exists(paste(outputdir,"/TUCR_Database/",trait_id,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",trait_id,sep=""))
}

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
  filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

geneTraitSignificance = as.data.frame(cor(datExpr,datTraits, use = "p"))

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  dplyr::mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules) %>%
  dplyr::filter(trait == trait_id) %>%
  dplyr::arrange(cor) %>%
  dplyr::mutate(position = row_number())

traitmodules <- geneModuleMembership2 %>%
  dplyr::mutate(rownamer = row.names(geneModuleMembership2)) %>%
  separate(rownamer,into=c("rownamer","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  dplyr::filter(rownamer==trait_id)

traitmodules <- t(traitmodules)

traitmodules <- as.data.frame(cbind(row.names(traitmodules),traitmodules))

colnames(traitmodules) <- c("Module","ModuleMembership")

traitpvalues <- MMPvalue2 %>%
  dplyr::mutate(rownamer = row.names(MMPvalue2)) %>%
  separate(rownamer,into=c("rownamer","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
```

```
  dplyr::filter(rownamer==trait_id)

traitpvalues <- t(traitpvalues)

traitpvalues <- as.data.frame(cbind(paste("MM",str_remove(row.names(traitpvalues),"p.MM"),sep=""),traitp

colnames(traitpvalues) <- c("Module","MMpvalue")
traitmodules <- traitmodules %>%
  left_join(allmodules_check, by= "Module") %>%
  left_join(traitpvalues,"Module") %>%
  # Filter(checker == TRUE) %>%
  dplyr::mutate(moduleColors = str_remove(Module,"MM")) %>%
  dplyr::filter(moduleColors != "rownamer")

df_correlations <- data.frame(matrix(ncol=2,nrow=0, dimnames=list(NULL, c("module","cor"))))

i <- 1

for(i in 1:length(unique(traitmodules$moduleColors))){

module = as.character(unique(traitmodules$moduleColors)[i])
#module = "red"
column = match(module, modNames)
moduleGenes = traitmodules$moduleColors==module;
correlation <- cor(abs(geneModuleMembership[moduleGenes, column]),
                   abs(geneTraitSignificance[moduleGenes, 1]))

rbinder <- c(module,correlation)
df_correlations <- rbind(df_correlations,rbinder)


}

colnames(df_correlations) <- c("module","cor")

df_correlations <- as.data.frame(df_correlations) %>%
  mutate(Module = paste("MM",module,sep=""))

traitmodules2 <- traitmodules %>%
  left_join(df_correlations,by = "Module") %>%
  dplyr::select(module,cor,ModuleMembership,MMpvalue) %>%
  dplyr::mutate(RankModule = percent_rank(1-as.numeric(ModuleMembership)),Rankcor = percent_rank(cor),M
  dplyr::group_by(module) %>%
  dplyr::mutate(totalscore = sum(as.numeric(RankModule),as.numeric(Rankcor),na.rm=TRUE)) %>%
  dplyr::ungroup() %>%
  dplyr::arrange(as.numeric(cor)) %>%
  dplyr::mutate(position = row_number(),cor = ifelse(MMpvalue2 >= 0.05,NA,cor))


p <- ggplot(data = traitmodules2,mapping = aes(x = as.character(trait_id),y = reorder(module,position),
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("modules") +
```

```
  xlab(trait_id) +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(traitmodules2$module)),
    y = traitmodules2$position,
    x = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/TUCR_Database/",trait_id,"/",figureorder,"_",trait_id,"_wgcna_modulecor

if(!dir.exists(paste(outputdir,"/Figure6_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure6_Supplementary/",sep=""))
}

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure7_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure7_Supplementary/",sep=""))
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure2/",sep=""))){
  dir.create(paste(outputdir,"/Figure2/",sep=""))
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2f.png",sep=""), width = 3, height = 7, dpi = 600)
}

####Trait Correlation Plots
```

```
traitmodules3 <- traitmodules2 %>%
  filter(as.numeric(MMpvalue) <= 0.05)



i <- 1

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_6_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_6.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_6.png",sep=""), width = 5
```

```
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_6.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- 2

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_5_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_5.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
```

```
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_5.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_5.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- 3

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_4_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_4.png",sep=""), width = 5
}
```

```
if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_4.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_4.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_1_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_1.png",sep=""), width = 5
}
```

```
if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_1.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_1.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)-1

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_2_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_2.png",sep=""), width = 5
}
```

```
if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_2.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_2.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)-2

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_3_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_3.png",sep=""), width = 5
}
```

```
if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_3.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_3.png",sep=""), width = 5, height = 5, dpi = 600)
}


}
```

Figure 3

```
dir.create(paste(outputdir, "/Figure3/", sep = ""))
```

Figure 3E and 3H (Images)

```
Figure3_copy <- list.files("./Inputs/Figure3/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure3/", "Figure3e_image", sep = ""))) {
    file.copy(paste0("./Inputs/Figure3/pregenerated_figures/", Figure3_copy, sep = ""),
        paste("./", outputdir, "/Figure3/", sep = ""))
}
```

Figure 3A

```
data <- read_csv("./Inputs/Figure3/Figure3a_input.csv")

p <- ggplot(data,aes(x=reorder(Sample,Order), y=as.numeric(FoldChange),fill=Sample)) +
        geom_bar(stat="identity"
                 #,color="black"
                   ) +
        geom_errorbar(aes(ymax=FoldChange+stderr, ymin=FoldChange-stderr, width=.2)) +
        geom_text(aes(y = FoldChange, label=pvalue,vjust=-0.25,hjust=0.5),color = "#000000",size=10) +
        scale_y_continuous(limits = c(-0.25,7),breaks= scales::pretty_breaks(n=9)) +
        #geom_hline(yintercept = log10(5),linetype = 2,size=1.5) +
        #geom_vline(xintercept = -0.65,linetype = 2,size=1.5) +
        #geom_vline(xintercept = 0.65,linetype = 2,size=1.5) +
        #scale_color_identity(guide = "legend", labels = cellfracdata$Fraction, breaks = cellfracdata$F
        labs(y = "log2(uc.110 FoldChange)",
        x = "GT = GBM Tumor, NBC = Normal Brain Cortex") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_blank()) +
  coord_cartesian(clip = "off")

p
```

```
ggsave(file=paste(outputdir,"/Figure3/Figure3a.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
       dpi = 600)
```

Figure 3B

uc.110 expression in GBM Tumors (summarized)

```
data <- read_csv("./Inputs/Figure3/Figure3b_input.csv")

p <- ggplot(data,aes(x=dex, y=FC,fill=Fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = data, aes(x=dex, y=FC),binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.3
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_text(size = rel(2.5), face="bold"),
        axis.text.x = element_text(size = rel(2.5), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
  labs(y="log2FoldChange", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

p

ggsave(file=paste(outputdir,"/Figure3/Figure3b.png",sep=""),
       plot = print(p),
       height = 7,
       width = 5,
       dpi = 600)
```

Figure 3C

```
data <- read_csv("./Inputs/Figure3/Figure3c_input.csv") %>%
  mutate(fillcolor = paste(Gene,Sample,sep=" "))

p <- ggplot(data,aes(x=reorder(Gene,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-0.25,hjust=0.5),color = "#000000",size=10,position = p
        scale_fill_manual(values = c(paper_red,paper_green,paper_blue,paper_gold,paper_turq,paper_purple
        labs(y = "Relative uc.110 expression",
        x = "") +
```

```
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2),face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.2)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank(),) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure3/Figure3c.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
       dpi = 600)
```

Figure 3D

```
data <- read_csv("./Inputs/Figure3/Figure3d_input.csv") %>%
  group_by(Day,siRNA,Sample,Color) %>%
  dplyr::summarise(mean = mean(Count,na.rm=TRUE),n=n(),stddev= sd(Count,na.rm=TRUE),stderr=stddev/n) %>%
  mutate(pvalue = ifelse(Day == 7 & siRNA != "si-SCR" & Sample == "A172","*",
                  ifelse(Day == 5 & siRNA != "si-SCR" & Sample == "U251","*","")))

p <- ggplot(data,aes(x=Day,y=mean,color=siRNA)) +
  geom_line(size=1.2) +
  geom_point(size=3) +
  geom_text(aes(x=Day,y=mean, label=pvalue,vjust=-0.25,hjust=0.5),color = "#000000",size=10) +
  geom_errorbar(aes(ymax=mean+stderr, ymin=mean-stderr, width=.2)) +
  scale_color_manual(values = c(paper_red,paper_green,paper_blue)) +
  labs(y = "Number of Cells",
       x = "Days post transfection") +
  theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.2)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~Sample)

p

ggsave(file=paste(outputdir,"/Figure3/Figure3d.png",sep=""),
```

```
       plot = print(p),
       height = 5,
       width = 10,
       dpi = 600)
```

Figure 3E

```
data <- read_csv("./Inputs/Figure3/Figure3e_input.csv") %>%
  mutate(fillcolor = paste(Gene,Sample,sep=" "))

p <- ggplot(data,aes(x=reorder(Gene,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-0.25,hjust=0.5),color = "#000000",size=10,position = p
        scale_fill_manual(values = c(paper_red,paper_green,paper_blue,paper_gold,paper_turq,paper_purple
        labs(y = "Relative accumulated cells",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure3/Figure3e.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
       dpi = 600)
```

Figure 3F

```
data <- read_csv("./Inputs/Figure3/Figure3f_input.csv") %>%
  mutate(fillcolor = paste(Gene,Sample,sep=" "))

p <- ggplot(data,aes(x=reorder(Gene,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-0.25,hjust=0.5),color = "#000000",size=10,position = p
        scale_fill_manual(values = c(paper_red,paper_green,paper_blue,paper_gold,paper_turq,paper_purple
        labs(y = "Relative AlamarBlue Signal",
        x = "") +
        #ylim(-5, 10) +
```

```
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.2)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure3/Figure3f.png",sep=""),
       plot = print(p),
       height = 6,
       width = 7,
       dpi = 600)
```

Figure 3G

```
data <- read_csv("./Inputs/Figure3/Figure3g_input.csv") %>%
  mutate(fillcolor = paste(Gene,Sample,sep=" "))

p <- ggplot(data,aes(x=reorder(Sample,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-1.5,hjust=0.5),color = "#000000",size=10,position = po
        scale_fill_manual(values = c(paper_red,paper_green,paper_blue,paper_gold,paper_turq,paper_purple
        labs(y = "Relative AlamarBlue Signal",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure3/Figure3g.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
```

```
       dpi = 600)
```

Figure 3H

```
data <- read_csv("./Inputs/Figure3/Figure3h_input.csv")

p <- ggplot(data,aes(x=reorder(Sample,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-1.5,hjust=0.5),color = "#000000",size=10,position = po
        scale_fill_manual(values = c(paper_red,paper_green,paper_blue,paper_gold,paper_turq,paper_purple
        labs(y = "Relative invading cells",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure3/Figure3h.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
       dpi = 600)
```

Figure 3i

```
data <- read_csv("Inputs/Figure3/Figure3i_input.csv")

p <- ggplot(data,aes(x=reorder(Gene,order), y=FC,fill=Gene)) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-3.0,hjust=0.5),color = "#000000",size=10,position = po
        scale_fill_manual(values = c(paper_gold,paper_purple)) +
        labs(y = "Relative uc.110 expression",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
```

```
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank(),) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine,scales="free")


p

ggsave(file=paste("Outputs/Figure3/Figure3i.png",sep=""),
       plot = print(p),
       height = 12,
       width = 6,
       dpi = 600)
```

Figure 3j

```
data <- read_csv("./Inputs/Figure3/Figure3j_input.csv")

p <- ggplot(data,aes(x=reorder(Gene,order), y=FC,fill=Gene)) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-1.5,hjust=0.5),color = "#000000",size=10,position = po
        scale_fill_manual(values = c(paper_gold,paper_purple)) +
        labs(y = "Relative accumulated cells",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank(),) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure3/Figure3j.png",sep=""),
       plot = print(p),
       height = 8,
       width = 4,
       dpi = 600)
```

Figure 4

```
dir.create(paste(outputdir, "/Figure4/", sep = ""))
```

Figure 4A-4C

```
Figure4_copy <- list.files("./Inputs/Figure4/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure4/", "Figure4abc.png", sep = ""))) {
    file.copy(paste0("./Inputs/Figure4/pregenerated_figures/", Figure4_copy, sep = ""),
        paste("./", outputdir, "/Figure4/", sep = ""))
}
```

uc.110 regulates the expression of the Wnt pathway member, Membrane Frizzled Related Protein (MFRP).

LncRNAs can have various functions that depend on their subcellular localization. Nuclear lncRNAs are usually involved in transcriptional regulation, while cytosolic lncRNAs are usually involved in translational and spatial regulation. [2] We fractioned four GBM cell lines (A172, U251, U87, U1242) into nuclear and cytosolic fractions. When compared to nuclear (U44, U48) and cytosolic (GADPH, PPIA) controls, uc.110 appears to be localized to both the nucleus (mainly in U87, U251, and U1242 cells), and the cytoplasm (mainly in A172 cells) (Supplementary Figure 10G). We then performed RNA-Seq on A172 cells that had been transfected with si-SCR, si-uc.110-1, or si-uc.110-2 for 48 hrs. and found several genes that are deregulated when uc.110 expression is downregulated (Figure 5A). To identify genes that are particularly related to uc.110 function, we focused on genes that demonstrated coregulation with uc.110 in our WGCNA analysis (Figure 2E). Of particular interest was the membrane frizzled related protein, also known as MFRP. [47, 48] MFRP serves as a shuttle for the Wnt-ligand, and functions as an activator of the Wnt-signaling pathway. This gene was the only gene in our analysis that correlated with uc.110 expression, was upregulated in GBM tumors, and downregulated when uc.110 is knocked down in A172 cells, suggesting MFRP coregulation with uc.110. (Figure 5B).

uc.110 sponges the tumor suppressor microRNA miR-544 to increase the bioavailability of MFRP and WNT activity in GBM.

One common lncRNA mechanism of action is as a miRNA sponge, acting as a binding competitor for various miRNAs and therefore increasing the bioavailability of those miRNAs' targets. [2, 49, 52-53] Based on the WGCNA data that we generated above, we hypothesized that uc.110 may function by sponging miRNAs away from MFRP transcripts, as their expression relationship is consistent with such an interaction. We hypothesized that a tumor suppressor miRNA can successfully target and suppress MFRP in the normal brain (Supplementary Figure 13A). This leads to downstream activation of Wnt target genes involved in biological processes such as cell accumulation, invasion, and stem cell differentiation (Supplementary Figure 13B). We further hypothesized that in glioma tumors, uc.110 is activated and acts as a binding competitor for this miRNA (Supplementary Figure 13C), increasing the bioavailability of MFRP and increasing Wnt pathway signaling (Supplementary Figure 13D). To identify candidate miRNAs that are consistent with the afore mentioned hypothesis, we screened public databases and published literature for GBM tumor suppressor miRNAs that are predicted to bind to both uc.110 and MFRP. The only miRNA that fulfilled these criteria was miR-544. We first investigated the functional effects of miR-544 in GBM cells. Transfection of miR-544 into U251, A172, and T98G GBM cell lines reduced cell accumulation after 5 days (Figure 5C). Expression of both uc.110 and MFRP was statistically significantly reduced when A172 cells transfected with miR-544 or si-uc.110 (Figure 5D). U251 cells demonstrated a similar reduction in expression for both genes, but to a statistically insignificant degree. To further test the hypotheses, we asked if miR-544 targets both uc.110 and MFRP, and if this binding affects Wnt signaling. To determine whether MFRP and uc.110 are direct targets of miR-544, we constructed luciferase reporter vectors by inserting the uc.110 ultraconserved region and MFRP 3'UTR downstream of hRluc followed by Synthetic Poly(A) using psiCHECK-2 backbone vector (Promega) (Figure 6A,6B). We first measured target binding by transfecting the reporter constructs followed by transfection with miR-544 or miR-SCR (control) in GBM cells. Ectopic expression of miR-544 significantly decreased luciferase activity compared to miR-SCR (Figure 6D, left panel and figure 6E, left panel). These binding sites for miR-544 were predicted via computational algorithms and validated via

sequencing. We then mutated the binding sites for MFRP and uc.110 (Supplementary Figure 12, Figure 6C) and assessed signal strength again. The data showed that luciferase activity was not significantly altered in mutant-reporter-vectors transfected cells (Figure 6D, right panel and Figure 6E, right panel), indicating that miR-544 binds to both uc.110 and MFRP in GBM cells, and that this binding is lost when the miRNA binding sites are mutated.

Figure 5.

```
dir.create(paste(outputdir, "/Figure5/", sep = ""))
```

Figure 5A and 5B

```
res_A172 <- read_csv(paste("./Inputs/Figure5/Figure5a_input.csv", sep = ""))

miR544 <- read_delim("./Inputs/Figure5/TargetScan8.0__miR-544a-5p.predicted_targets.txt",
    col_names = TRUE, delim = "\t")

res_miR544 <- res_A172 %>%
    inner_join(miR544, by = "Target gene") %>%
    # left_join(cancermine,by='Target gene') %>%
    # Filter(abs(log2FoldChange)>=0.65) %>% dplyr::select(`Target
    # gene`,log2FoldChange,role) %>%
dplyr::select(`Target gene`, log2FoldChange, padj) %>%
    mutate(mirna = "miR-544")

if (!dir.exists(paste(outputdir, "/Supplementary_Table3/", sep = ""))) {
    dir.create(paste(outputdir, "/Supplementary_Table3/", sep = ""))
}

## read data

gbm_countfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_mergedcounts.txt",
    sep = "")

lgg_countfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_mergedcounts.txt",
    sep = "")

cortex_countfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_mergedcounts.txt",
    sep = "")


gbm_metadatafile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_tcga_metadata.csv",
    sep = "")

lgg_metadatafile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_tcga_metadata.csv",
    sep = "")

cortex_metadatafile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_gtex_metadata.csv",
    sep = "")


gbm_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_seqdepth_counts.csv",
    sep = "")
```

```
lgg_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_seqdepth_counts.csv",
    sep = "")

cortex_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_seqdepth_counts.csv",
    sep = "")


## Merge Data


if (!is.na(cortex_countfile)) {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    cortex_normalcounts <- read.table(cortex_countfile, header = TRUE)

    cortex_normalcounts <- cortex_normalcounts[, 9:ncol(cortex_normalcounts)]

    gbm_mergedcounts <- cbind(gbm_mergedcounts, cortex_normalcounts) %>%
        distinct()

    rm(cortex_normalcounts)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    cortex_metadata <- read_csv(file = cortex_metadatafile)

    gbm_metadata <- rbind(gbm_metadata, cortex_metadata)

    rm(cortex_metadata)

    gbm_seqdepth <- read.csv(file = gbm_seqdepthfile)

    cortex_seqdepth <- read.csv(file = cortex_seqdepthfile)

    gbm_seqdepth <- rbind(gbm_seqdepth, cortex_seqdepth)

    rm(cortex_seqdepth)

} else {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    gbm_seqdepth <- read.csv(file = gbm__seqdepthfile)
}

gbm_mergedcounts_deseq2 <- gbm_mergedcounts

gbm_mergedcounts_deseq2_countsonly <- gbm_mergedcounts_deseq2[, 9:ncol(gbm_mergedcounts_deseq2)]

rownames(gbm_mergedcounts_deseq2_countsonly) <- gbm_mergedcounts_deseq2$id

gbm_mergedcounts_deseq2_info <- gbm_mergedcounts_deseq2[, 1:8]
```

```
gbm_mergedcounts_deseq2_info$length <- (gbm_mergedcounts_deseq2$end - gbm_mergedcounts_deseq2$start)/10

dds <- DESeqDataSetFromMatrix(countData = gbm_mergedcounts_deseq2_countsonly, colData = gbm_metadata,
    design = ~dex)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)

res_combine <- res %>%
    separate(row, into = c("Target gene", "kibble"), sep = "___") %>%
    inner_join(res_miR544, by = "Target gene") %>%
    write_csv(paste(outputdir, "/Supplementary_Table3/supplementary_table3.csv",
        sep = ""))
```

Find likely uc.110-miR regulated genes

```
### Volcano plot

volcanoplot_mirna <- function (res,res2,
                          title = "Deregulated and coregulated uc.110 genes are predicted to be miRNA ta
                          output = "",
                          output2 = "",
                          height = 10,
                          width = 15,
                          dpi = 600){
  #res <- uc.110_miRNAs
  #genes <- "all"
  #genes = c("uc.110")
  #title = "Deregulated Genes"
  #res <- res %>%
  #  mutate(filter1 = abs(log2FoldChange)>=0.65,filter2 = pvalue <=0.05) %>%
  #  mutate(Legend = ifelse(filter1 == TRUE & filter2 == TRUE, ">1.6-Fold & <0.05 FDR",
  #                  ifelse(filter1 == TRUE, ">1.6-Fold","Not deregulated"))) %>%
  #  mutate(color = ifelse(Legend==">1.6-Fold & <0.05 FDR","red",
  #                 ifelse(Legend==">1.6-Fold","blue",
  #                 ifelse(Legend=="Not deregulated","black","gray"
  #                        ))),
  #         order = ifelse(Legend==">1.6-Fold & <0.05 FDR",1,
  #                 ifelse(Legend==">1.6-Fold",2,
  #                 ifelse(Legend=="Not deregulated","black",3
  #                        )))) %>%
  #  arrange(order)

  res <- res_combine %>%
    filter(!is.na(log2FoldChange.y)) %>%
    mutate(color = ifelse(abs(log2FoldChange.x) >= 1 & abs(log2FoldChange.y) >= 1,"#9b00d2",
                   ifelse(abs(log2FoldChange.x) >= 1,"pink",
                   ifelse(abs(log2FoldChange.y) >= 1,"#b9d4ed","lightgray"
                          ))),
           order = ifelse(abs(log2FoldChange.x) >= 1 & abs(log2FoldChange.y) >= 1,1,
                   ifelse(abs(log2FoldChange.x) >= 1,2,
```

```
                  ifelse(abs(log2FoldChange.y),3,4
                         ))),
          newname = ifelse(`Target gene` == "MFRP","MFRP",""),
          legend = ifelse(abs(log2FoldChange.x) >= 1 & abs(log2FoldChange.y) >= 1,">= 2-FC Both",
                  ifelse(abs(log2FoldChange.x) >= 1,">= 2-FC A172s only",
                  ifelse(abs(log2FoldChange.y) >= 1,">= 2-FC TCGA only","Neither"
                         ))) %>%
    arrange(order)

#p <- ggplot(res,aes(x=TUC110_rank, y=GBM_rank,label=label,color=color)) +
p <- ggplot(res,aes(x=log2FoldChange.x, y=log2FoldChange.y)) +
        geom_point(data = res,aes(x=log2FoldChange.x, y=log2FoldChange.y,color=color,size=20)) +
        #geom_hline(yintercept = log10(5),linetype = 2,size=1.5) +
        #geom_vline(xintercept = -0.65,linetype = 2,size=1.5) +
        #geom_vline(xintercept = 0.65,linetype = 2,size=1.5) +
        #scale_y_continuous(breaks = scales::pretty_breaks(n = 5)) +
        scale_color_identity(guide = "legend", breaks = res$color, labels = res$legend) +
        ggtitle("") +
        guides(colour = guide_legend(override.aes = list(size=2.5))) +
        labs(x = "Fold Change (TCGA GBM tumor v normal brain cortex)",
        #y = "-log10 adjusted p-value",
        y = "Fold Change (A172 si-uc.110 v si-SCR)") +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(2.0), face="bold"),
              axis.text = element_text(size = rel(2.2), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=20)) +
  geom_label_repel(data = res,aes(x=log2FoldChange.x, y=log2FoldChange.y,label=newname),force=300,stat =

p

ggsave(output, width = 10, height = 6, dpi = dpi)

  res2 <- res2 %>%
    filter(log2FoldChange >= -5) %>%
    mutate(color = ifelse(abs(log2FoldChange) >= 1 & pvalue <= 0.05,"#9b00d2",
                  ifelse(abs(log2FoldChange) >= 1,"pink",
                  ifelse(pvalue <= 0.05,"#b9d4ed","lightgray"
                         ))),
          order = ifelse(abs(log2FoldChange) >= 1 & pvalue <= 0.05,1,
                  ifelse(abs(log2FoldChange) >= 1,2,
                  ifelse(pvalue <= 0.05,3,4
                         ))),
          newname = ifelse(`Target gene` == "MFRP","MFRP",
                           ifelse(`Target gene` == "LINC01038","LINC01038",
                           ifelse(`Target gene` == "LINC01068","LINC01068",
                           ifelse(`Target gene` == "LINC01643","LINC01643",
                           ifelse(`Target gene` == "ST18","ST18",""))))),
          legend = ifelse(abs(log2FoldChange) >= 1 & pvalue <= 0.05,">= 2 FC & 0.05 FDR",
                  ifelse(abs(log2FoldChange) >= 1,">= 2 FC",
                  ifelse(pvalue <= 0.05,"0.05 FDR","Neither"
                         ))) %>%
```

```
    filter(!is.na(legend)) %>%
    arrange(order)

res3 <- res2 %>%
  filter(`Target gene` == "MFRP")

p <- ggplot(res2,aes(x=log2FoldChange, y=-log10(pvalue))) +
        geom_point(data = res2,aes(x=log2FoldChange, y=-log10(pvalue),color=color)) +
        scale_x_continuous(breaks=seq(-5,5,1)) +
        scale_y_continuous(breaks=seq(0,15,3)) +
        scale_color_identity(guide = "legend", breaks = res2$color, labels = res2$legend) +
        ggtitle("") +
        guides(colour = guide_legend(override.aes = list(size=2.5))) +
        geom_hline(yintercept = 1.30,linetype = 2) +
        geom_vline(xintercept = -1,linetype = 2) +
        geom_vline(xintercept = 1,linetype = 2) +
        labs(x = "Fold Change (A172 si-uc.110 vs si-SCR)",
        #y = "-log10 adjusted p-value",
        y = "-log10(padj)") +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(2.2), face="bold"),
              axis.text = element_text(size = rel(2.2), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.2))) +
  geom_label_repel(data = res3,aes(x=log2FoldChange, y=-log10(pvalue),label=newname),force=100,stat = "u

p

ggsave(output2, width = 12, height = 8, dpi = dpi)
}

volcanoplot_mirna(res_combine,res_A172,output=paste(outputdir,"/Figure5/figure5b.png",sep=""),output2=pa
```

Figure 5C

```
data <- read_csv("./Inputs/Figure5/accum_miRNA2.csv")

p <- ggplot(data,aes(x=reorder(Sample,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-1.5,hjust=0.5),color = "#000000",size=10,position = po
        #scale_fill_manual(values = c("# F8766D","#0CB702","#619CFF","#D39200","#00C19F","#DB72FB")) +
        scale_fill_manual(values = c(paper_red,paper_yellow)) +
        labs(y = "Relative accumulated cells",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
```

```
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure5/Figure5c.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
       dpi = 600)
```

Figure 5D

```
data <- read_csv("./Inputs/Figure5/qPCR_miRNA3.csv")

p <- ggplot(data,aes(x=reorder(Sample,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC+stderr+0.01, label=pvalue,vjust=-1,hjust=0.5),color = "#000000",size=10,pos
        scale_fill_manual(values = c(paper_red,paper_blue,paper_yellow)) +
        labs(y = "Relative gene expression",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(2), face="bold"),
              axis.text.y = element_text(size = rel(2.2)),
              axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine+Gene,ncol=4)


p

ggsave(file=paste(outputdir,"/Figure5/Figure5d.png",sep=""),
       plot = print(p),
       height = 6,
       width = 9,
       dpi = 600)
```

Lastly, we asked if uc.110 expression alters Wnt pathway activity. To answer this question, we studied one of the most established downstream targets of Wnt-signaling, the T cell factor/lymphoid enhancer factor family (TCF/LEF). When Wnt-signaling is activated, TCF/LEF is produced downstream and activates Wnt-signaling target genes. Therefore, TCF/LEF activity can be used as a proxy for pathway activity and can be measured with a TCF/LEF luciferase reporter assay. (Figure 6F). The activity of this reporter

can be regulated by either directly reducing Wnt bioavailability with miR-544 or indirectly by targeting uc.110 with siRNA. If upstream Wnt signaling is reduced, the luciferase construct will bind fewer activators and exhibit decreased signal. Likewise, we would expect that overexpression of uc.110 would rescue the bioavailability of MFRP and consequently also downstream activation of the TCF/LEF construct. We found that transfection of A172 and U251 cells with si-uc.110 and miR-544 reduced reporter activity in A172 (Figure 6G) and U251 (Figure 6H) cells, and that this effect can be rescued via uc.110 overexpression. These data taken in conjunction provide strong support for a miRNA sponge model for the uc.110 tumor enhancer. Altogether, the above data demonstrate an important role for uc.110 in regulating the Wnt pathway in GBM by sponging the Wnt inhibitory miRNA miR-544 (model shown in Supplementary Figure 13).

Figure 6.

```
dir.create(paste(outputdir, "/Figure6/", sep = ""))
```

Figure 6A, 6B, 6C, and 6F

```
Figure6_copy <- list.files("./Inputs/Figure6/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure6/", "Figure6ab.png", sep = ""))) {
    file.copy(paste0("./Inputs/Figure6/pregenerated_figures/", Figure6_copy, sep = ""),
        paste("./", outputdir, "/Figure6/", sep = ""))
}
```

Figure 6D and 6E

```
data <- read_csv("./Inputs/Figure6/figure6de_input.csv")

data$CellLine <- factor(data$CellLine, levels = c("A172 WT","A172 MUT","U251 WT","U251 MUT"))

p <- ggplot(data,aes(x=reorder(Sample,Order),y=MEAN,fill=reorder(Sample,Order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=MEAN+SE, ymin=MEAN-SE, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = MEAN, label= SIG,vjust=-1.5,hjust=0.5),color = "#000000",size=10,position = po
        scale_fill_manual(values = c(paper_red,paper_blue,paper_red2,paper_yellow)) +
        labs(y = "Relative uc.110/LUC reporter expression",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine,ncol=4)


p

ggsave(file=paste(outputdir,"/Figure6/Figure6de.png",sep=""),
```

```
       plot=print(p),
       height = 6,
       width = 8,
       dpi = 600)
```

Figure 6G and 6H

```
data <- read_csv("Inputs/Figure6/figure6gh_input.csv") %>%
  dplyr::select(CellLine,Sample,MEAN,SE,SIG,Order)

p <- ggplot(data,aes(x=reorder(Sample,Order), y=MEAN,fill=reorder(Sample,Order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=MEAN+SE, ymin=MEAN-SE, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = MEAN, label= SIG,vjust=-1.5,hjust=0.5),color = "#000000",size=10,position = po
        scale_fill_manual(values = c(paper_red,paper_blue,paper_yellow,paper_red,paper_blue,paper_yellou
        labs(y = "Relative TCF/LUC reporter expression",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure6/Figure6gh.png",sep=""),
       plot=print(p),
       height = 7,
       width = 9,
       dpi = 600)
```

#DISCUSSION
This study investigated Transcribed Ultraconserved Regions (TUCRs), a set that might contain long non-coding RNA sequences that are fully conserved across human, mouse, and rat genomes. These TUCRs are distinct due to their exceptional conservation, which often signifies functional importance. Despite their potential significance, TUCRs have been minimally explored, especially in relation to cancer. For example, while a PubMed search of TUCR, UCR, or "ultraconserved" And Cancer reveals more than 69 publications. Other classes of RNA, such as MicroRNAs (miRNAs) contain over 12,000 publications in cancer. Even single protein coding genes, such as TP53/p53, contain over 24,000 publications in cancer. [2] Of note, the findings of this study represent the first of their kind on TUCRs in gliomas. They contribute critical new insights into an uncharted area of glioma biology, while also providing a novel framework for studying TUCRs in other cancers and other diseases, where they are also understudied.

We confirmed that TUCRs are located across the genome, resistant to variation, and actively transcribed. We manually annotated each as either exonic, intronic, exonic/intronic, or intergenic. We identified distinct signatures for intergenic and intragenic (exonic, intronic, exonic/intronic) RNAs. Intragenic TUCRs are

expressed at a level that is most like coding genes, while intergenic TUCRs more closely resemble lncRNAs. We then performed the first analysis of TUCR expression in gliomas and found that the majority of TUCRs are deregulated >= 2-fold in GBM and LGG, with a 56% overlap. This shows that TUCRs are not only expressed, but also frequently dysregulated in gliomas compared to normal brain tissue. This is critical, as their high degree of conservation and dysregulation suggests that they may serve critical biological functions. We then extended our analysis to TUCR correlation with patient survival. In GBM, the extremely short survival times (15 months) limit the detection of significant correlations. However, patients with LGG live substantially longer (84 months), and therefore more TUCRs are associated with patient outcomes in this disease, suggesting a potential impact on glioma patients' prognoses and indicating possible novel biomarkers. Another facet of our research involved predicting the functions and mechanisms of action of TUCRs in gliomas. We studied this for the first time in gliomas WGCNA workflows to cluster TUCRs and provide functional predictions based on shared functions between coregulated genes. This approach identifies a wide range of potential functions for TUCRs, encompassing activities such as nucleic acid binding regulation, stem cell differentiation, organ development, immune response, and cell signaling.

We found intergenic TUCRs to be of notable interest because they resemble lncRNAs but are much more highly conserved and experience less sequence variation. Notably, these TUCRs do not overlap with known genes, suggesting they might represent novel lncRNAs. Of these TUCRs, uc.110 is the most upregulated in both GBM and LGG. Knocking down uc.110 reduces cancer cell characteristics in vitro and in vivo and improves survival in mouse models. On the other hand, increasing uc.110 expression increases malignancy in cells that do not express it, further indicating its potential tumor enhancing role. We explored uc.110's function via WGCNA, revealing its membership in modules associated with tumor enhancing nucleic acid binding. We integrated these data with transcriptome deregulation data (RNA-Seq) post-uc.110 knockdown, revealing a close relationship between uc.110 and the oncogenic membrane frizzled-related protein (MFRP). This protein is involved in activating the Wnt-signaling pathway, impacting cell proliferation, invasion, migration, and stem cell differentiation. From these data we hypothesized that uc.110 might sponge tumor suppressor miRNAs from MFRP, enhancing Wnt signaling activation. Accordingly, we demonstrated that one mechanism of action for the uc.110 tumor enhancer is as a miRNA sponge for miR-544, therefore increasing the bioavailability of MFRP. This is a novel interaction between all three genes, as while the miRNA sponge model is well established for TUCRs, the published literature on the role of miR-544 and its effects on MFRP alone is non-existent, and likewise there is no published literature considering uc.110's role in potentially mediating this interaction as a sponge. We also note that this is a likely cytosolic function for uc.110, as although miRNAs may target nuclear genes during cell division, the sponging model is generally considered to be a cytosolic function. Per the same WGCNA analysis, it is entirely possible that uc.110 has a separate nuclear factor, perhaps in transcription factor binding (Figure 3, Supplementary Figure 7). This is a role that we anticipate exploring in future experiments.

In conclusion, our results suggest that TUCRs are an important class of regulatory RNAs. They are more highly conserved than typical genes and more resistant to variation, which suggests biological importance. They are perturbed in gliomas, and this perturbation is associated with clinical outcomes. Our predicted functions reveal that TUCRs are widely involved in cancer-related biological processes. Some TUCRs previously thought to be intergenic may represent previously undiscovered genes. Our findings also identify and characterize uc.110 as a new tumor enhancer and likely oncogene in gliomas. Each of the experiments performed in our study represents the first of its kind in gliomas. We have developed, adapted, and presented novel methods for studying TUCRs that can be used in other cancers and other diseases, where TUCRs remain very understudied. These methods and the data derived from them represent a "TUCR database" that will serve the scientific community in future TUCR studies in gliomas and other diseases, where they remain unstudied or understudied.

SUPPLEMENTARY DATA

Supplementary Figure 1.

Supplementary Figure 1A and 1B

Supplementary Figure 1A is a manually generated flowchart describing the experimental workflow for this project. It is copied from the input director to the output directory with no changes.

Supplementary Figure 1B is a manually generated cartoon using Microsoft Powerpoint 2016. It is copied from the input directory to the output directory with no changes.

```
if (!dir.exists(paste(outputdir, "/Figure1_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure1_Supplementary/", sep = ""))
}

Figure1_supplementary_copy <- list.files("./Inputs/Figure1_Supplementary/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure1_Supplementary/", "supplementary_figure_1a.png",
    sep = ""))) {
    file.copy(paste0("./Inputs/Figure1_Supplementary/pregenerated_figures/", Figure1_supplementary_copy
        sep = ""), paste("./", outputdir, "/Figure1_Supplementary/", sep = ""))
}
```

Supplementary Figure 1C

TUCRs were then manually annotated, creating the "hg38.ultraconserved.bed" file provided in this repository. As previously mentioned, we have identified 45 exonic, 231 intronic, 68 intronic/exonic, and 137 intergenic TUCRs. Figure 1c was generated using GGPlot2.

The following code chunk reads in the "hg38.ultraconserved.bed" file and compiles the relevant information into a proportion ring chart as depicted in Figure 1c, using GGPlot2. The results are placed in the output directory as defined by the variable outputdir, in the subfolder "Supplementary Figure 1."

```
tucrjoiner <- read.table("Inputs/general_files/bedfiles/hg38.ultraconserved.bed",header=TRUE)

supplementary_figure1c <- tucrjoiner %>%
  dplyr::select(alias,annot) %>%
  distinct() %>%
  group_by(annot) %>%
  dplyr::summarise(count = n()) %>%
  mutate(percent = count/sum(count)) %>%
  arrange(desc(count)) %>%
#Determine label positions
  mutate(lab.ypos = cumsum(count) - 0.5*count)

# Set factor levels
names(supplementary_figure1c)[names(supplementary_figure1c) == 'annot'] <-
  'Annotation'

#Create Chart
supplementary_figure1c <- ggplot(supplementary_figure1c, aes(x=2,y=count,fill=Annotation))          +
  geom_bar(
    stat="identity",#colour="black",
    width = 1)                                                                +
  geom_text(
    aes(y = lab.ypos, label = count),
```

```
    color = "black",size=5)                                                    +
  coord_polar("y", start=0)                                                    +
  scale_fill_manual(values=c(paper_red,paper_purple,paper_green,paper_blue))  +
  ggtitle(paste("TUCR Genomic Location"))                                      +
  guides(colour = guide_legend(override.aes = list(size=10))) +
        theme(plot.title = element_blank(),
              axis.title = element_blank(),
              axis.text.y = element_blank(),
              axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_blank(),legend.position="none",legend.title=elem
legend.text = element_text(size = 15),
axis.ticks = element_blank(),
      legend.box.spacing = unit(0, "pt"),
legend.margin=margin(0,0,0,0)) +
  xlim(0.5, 2.5)

# Save Chart to output directory
ggsave(file=paste(outputdir,"/Figure1_Supplementary/","supplementary_figure_1c.png",sep=""),
       plot = print(supplementary_figure1c),
       dpi = 600,
       height = 4.75,
       width = 4.75)
```

Generate BED Files for each annotation category

```
if (!dir.exists(paste(outputdir, "/BEDFiles_TUCR_annotations/", sep = ""))) {
    dir.create(paste(outputdir, "/BEDFiles_TUCR_annotations/", sep = ""))
}

tucr_annot <- read.table("Inputs/general_files/bedfiles/hg38.ultraconserved.bed",
    header = TRUE)
write_delim(tucr_annot, file = paste(outputdir, "/BEDFiles_TUCR_annotations/alltucrs_hg38.ultraconserve
    sep = ""), delim = "\t")

annotations <- unique(as.character(tucr_annot$annot))

for (i in 1:length(annotations)) {
    filter_TUCR <- annotations[i]
    tucr_annot2 <- tucr_annot %>%
        filter(annot == filter_TUCR) %>%
        write_delim(file = paste(outputdir, "/BEDFiles_TUCR_annotations/", filter_TUCR,
            "_hg38.ultraconserved.bed", sep = ""), delim = "\t")
}
```

Supplementary Figure 1D

```
expected_values <- read_delim("Inputs/general_files/bedfiles/allgenes.bed",delim="\t",col_names=FALSE) %
  filter(X7 != "TUCR") %>%
  group_by(X8) %>%
  dplyr::summarise(count = n()) %>%
  mutate(percent = count/sum(count)) %>%
  arrange(desc(count)) %>%
#Determine label positions
```

```
  mutate(lab.ypos = cumsum(count) - 0.5*count,Annotation = X8,test="expected") %>%
  dplyr::select(Annotation,percent,test)

tucrjoiner <- read.table("Inputs/Figure1_supplementary/TUCRhostgenes.bed",header=TRUE)

observed_values <- tucrjoiner %>%
  filter(tag_host != "TUCR") %>%
  group_by(annot_host) %>%
  dplyr::summarise(count = n()) %>%
  mutate(percent = count/sum(count)) %>%
  arrange(desc(count)) %>%
#Determine label positions
  mutate(lab.ypos = cumsum(count) - 0.5*count,test="observed")

# Set factor levels
names(observed_values)[names(observed_values) == 'annot_host'] <-
  'Annotation'

observed_values <- observed_values %>%
  dplyr::select(Annotation,percent,test)

supplementary_figure1e <- rbind(observed_values,expected_values) %>%
  filter(Annotation != "misc_RNA") %>%
  arrange(desc(test))

supplementary_figure1e$test <- factor(supplementary_figure1e$test,      # Reordering group factor level
                          levels = c("expected","observed"))

supplementary_figure1e <-   ggplot(supplementary_figure1e, aes(x=Annotation,y=as.numeric(percent),fill=
  geom_bar(stat="identity",
           #color = "black",
           width=0.7,position="dodge") +

  scale_fill_manual(values = c(paper_red2,paper_turq)) +
  #scale_y_continuous(expand=expansion(mult=c(0,0.15))) +
  #geom_text(
  #  aes(label=round(value,2)),
  #  vjust=1.6,
  #  color="black",
  #  size=rel(4))                                                    #+
  #ggtitle(paste("TUCRs are enriched for \n RNA Pol.II, H3K4me3,"))  +
  xlab("Host Gene Annotation")                                        +
  ylab("Proportion")                                          +
  #ggtitle(annot) +
  theme(
    #plot.title = element_text(
    #  size=rel(1.5),
    #  face="bold",hjust = 0.5),
    #plot.title = element_text(
    #  size = rel(1.5),hjust=0.5,
    #  face="bold"),
    strip.background = element_rect(fill="white"),
    plot.title = element_text(size=rel(3.0), face="bold",hjust = 0.5),
```

```
    axis.title = element_text(size = rel(2.0), face="bold"),
    axis.text = element_text(size = rel(2.0)),
    axis.text.x = element_text(angle=45,vjust=0.5),
    strip.text = element_text(size = rel(3.0), face="bold"),
    legend.title = element_blank(),
    legend.position = "top",
    legend.text = element_text(size = rel(2.0)),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    panel.background = element_blank(),
    axis.line = element_line(colour = "black"))

# Save Chart to output directory
ggsave(file=paste(outputdir,"/Figure1_Supplementary/","supplementary_figure_1d.png",sep=""),
       plot = print(supplementary_figure1e),
       dpi = 600,
       height = 7,
       width = 6)
```

Supplementary Figure 1e

```
# Read in Data
tucr_annot <- read.table("Inputs/general_files/bedfiles/hg38.ultraconserved.bed",
    header = TRUE) %>%
    dplyr::select(chrom, start, end, alias, annot) %>%
    mutate(chromosome = str_remove(chrom, "chr"), chromnum = ifelse(chromosome ==
        "X", 23, ifelse(chromosome == "Y", 24, ifelse(chromosome == "M", 25, as.numeric(chromosome)))))
    dplyr::select(chrom, start, end, alias, annot, chromnum)

hg38_chromsizes <- read.table("Inputs/Figure1_supplementary/hg38_chromsizes.txt") %>%
    dplyr::select(chrom = V1, chromend = V3, chromnum = V4)

hg38_subtract <- read.table("Inputs/Figure1_supplementary/hg38_subtract.txt") %>%
    mutate(row = "interval", annot = "not conserved") %>%
    dplyr::select(chrom = V1, start = V2, end = V3, alias = row, annot, chromnum = V4)

tucr_locations <- rbind(tucr_annot, hg38_subtract) %>%
    dplyr::group_by(chrom) %>%
    mutate(chromend = max(as.numeric(end), na.rm = TRUE)) %>%
    ungroup() %>%
    arrange(chromnum, start)

chrom_order <- c("chr1", "chr2", "chr3", "chr4", "chr5", "chr6", "chr7", "chr8",
    "chr9", "chr10", "chr11", "chr12", "chr13", "chr14", "chr15", "chr16", "chr17",
    "chr18", "chr19", "chr20", "chr21", "chr22", "chrX", "chrY", "chrM")
chrom_key <- setNames(object = as.character(c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,
    12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25)), nm = chrom_order)
chrom_key2 <- rev(chrom_key)
chrom_order <- factor(x = chrom_order, levels = rev(chrom_order))

tucr_annot[["annot"]] <- factor(x = tucr_annot[["annot"]], levels = c("exonic", "intergenic",
    "exonic_intronic", "intronic"))
```

```
tucr_annot <- tucr_annot %>%
    mutate(test = ifelse(annot == "intronic", paper_blue, ifelse(annot == "exonic",
        paper_red, ifelse(annot == "intergenic", paper_green, paper_purple))))

supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
supplementary_figure1d <- ggplot() + # base rectangles for the chroms, with numeric value for each chro
geom_rect(data = hg38_chromsizes, aes(xmin = as.numeric(26 - chromnum) - 0.3, xmax = as.numeric(26 -
    chromnum) + 0.3, ymax = as.numeric(chromend), ymin = 0), color = "black", fill = "lightgray") +
    geom_rect(data = tucr_annot, aes(xmin = as.numeric(26 - chromnum) - 0.3, xmax = as.numeric(26 -
        chromnum) + 0.3, ymax = as.numeric(end), ymin = as.numeric(start), color = test,
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees f
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees =
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees t
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees +
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees s
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees +
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees s
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees +
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees #
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees r
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees t
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees p
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees 9
        fill = test)) + scale_fill_identity() + scale_color_identity() + # rotate the plot 90 degrees d
coord_flip() + # black & white color theme  coord_flip() + # black & white
coord_flip() + # black & white color theme  color theme
theme(plot.title = element_blank(), text = element_text(size = 20), axis.title = element_text(size = rel
    face = "bold"), axis.text.y = element_text(colour = "black", size = rel(1.5)),
    axis.text.x = element_text(colour = "black", size = rel(1.5)), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), panel.background = element_blank(), legend.position = "none",
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
```

```
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
    legend.title = element_blank(), legend.text = element_text(size = 12.5)) + # give the appearance of
scale_x_discrete(name = "chromosome", limits = rev(names(chrom_key))) + scale_y_continuous(name = "regi
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
    labels = comma, position = "right") + # add bands for centromeres scale_fill_manual(values = group.
ggtitle("TUCR Genomic Locations")
# supress scientific notation on the y-axis

ggsave(file = paste(outputdir, "/Figure1_Supplementary/", "supplementary_figure_1e.png",
    sep = ""), plot = print(supplementary_figure1d), height = 10, width = 16, dpi = 600)

### uncolored version

# p2 <- ggplot() + base rectangles for the chroms, with numeric value for each
# chrom on the x-axis\ geom_rect(data = hg38_chromsizes, aes(xmin =
# as.numeric(chromnum) - 0.3, xmax = as.numeric(chromnum) + 0.3, ymax =
# as.numeric(chromend), ymin = 0),color='black',fill='lightgray') +
# geom_rect(data = tucr_annot, aes(xmin = as.numeric(chromnum) - 0.3, xmax =
# as.numeric(chromnum) + 0.3, ymax = as.numeric(end), ymin =
# as.numeric(start)),color='black',fill='black') +

# rotate the plot 90 degrees coord_flip() + black & white color theme
# theme(text = element_text(size = 20), axis.text.x = element_text(colour =
# 'black',size = 10 ), axis.text.y = element_text(size = rel(2),colour =
# 'black'), axis.title = element_text(size = rel(2), face='bold'),
# panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
# panel.background = element_blank()) + guides(colour = 'black',fill =
```

```
# guide_legend(override.aes = list(size=5))) + give the appearance of a
# discrete axis with chrom labels scale_x_discrete(name = 'chromosome', limits
# = names(chrom_key)) + scale_y_continuous(name = 'region (bp)',label_comma) +
# add bands for centromeres scale_fill_manual(values = group.colors) +
# ggtitle('TUCR Genomic Locations') supress scientific notation on the y-axis


# ggsave(file=paste(outputdir,'/',disease,'_tucr_locations.png',sep=''), plot =
# print(p2),height=10,width=15, dpi = 600)
```

Supplementary Figure 1F

Bejerano et al, 2004, reported that TUCRs are resistant to variation. We sought to update our understanding of TUCR variation using the latest resources. The following code chunk using bash scripting to download and decompress the most recent database for known SNPs (as of 2021) by chromosome.

```
mkdir Inputs/figure_specific_files/Figure1_supplementary/SNPs/

cd Inputs/figure_specific_files/Figure1_supplementary/SNPs/

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_1.bed.gz
gunzip bed_chr_1.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_10.bed.gz
gunzip bed_chr_10.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_11.bed.gz
gunzip bed_chr_11.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_13.bed.gz
gunzip bed_chr_13.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_12.bed.gz
gunzip bed_chr_12.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_14.bed.gz
gunzip bed_chr_14.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_15.bed.gz
gunzip bed_chr_15.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_16.bed.gz
gunzip bed_chr_16.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_17.bed.gz
gunzip bed_chr_17.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_18.bed.gz
gunzip bed_chr_18.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_19.bed.gz
gunzip bed_chr_19.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_2.bed.gz
gunzip bed_chr_2.bed.gz
```

```
wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_20.bed.gz
gunzip bed_chr_20.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_21.bed.gz
gunzip bed_chr_21.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_22.bed.gz
gunzip bed_chr_22.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_3.bed.gz
gunzip bed_chr_3.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_4.bed.gz
gunzip bed_chr_4.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_5.bed.gz
gunzip bed_chr_5.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_6.bed.gz
gunzip bed_chr_6.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_7.bed.gz
gunzip bed_chr_7.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_8.bed.gz
gunzip bed_chr_8.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_9.bed.gz
gunzip bed_chr_9.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_X.bed.gz
gunzip bed_chr_X.bed.gz

wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b151_GRCh38p7/BED/bed_chr_Y.bed.gz
gunzip bed_chr_Y.bed.gz
```

After downloading and unzipping the SNP genomic location files, we then use bedtools to identify and report all SNPs that overlap with TUCR bases/nucleotides.

The following chunk contains a for-loop that does this for all bed files in the directory created during the previous step for TUCRs. These were then compared to results for protein coding genes, long non-coding RNAs, antisense RNAs, and miscellaneous RNAs to determine comparable resistance to variation.

```
for input in BEDFiles/*bed
do
masterfilename=$(echo "${input##*/}" | awk -F ".bed" '{print $1}')
echo $masterfilename
for bed in SNPBeds/bed_chr*
do

name=$(echo "${bed##*/}" | awk -F ".bed" '{print $1}')
echo $name

intersectBed -a $input -b $bed -wa -wb > $name.$masterfilename.SNPs.bed
```

```
done

cat *.$masterfilename.SNPs.bed > merged.$masterfilename.SNPs.bed

rm $name.*
done

### ALL TUCRs ###
for bed in SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name

intersectBed -a BEDFiles/hg38.ultraConserved.bed -b $bed -wa -wb > $name.TUCR.SNPs.bed

done

cat SNPBeds/*.TUCR.SNPs.bed > merged_TUCR_SNPs.bed

rm SNPBeds/*TUCR.SNPs.bed

### Exonic TUCRs ###
for bed in SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name

intersectBed -a BEDFiles/exonic_hg38.ultraConserved.bed -b $bed -wa -wb > $name.exonicTUCR.SNPs.bed

done

cat SNPBeds/*.exonicTUCR.SNPs.bed > merged_exonicTUCR_SNPs.bed

rm SNPBeds/*exonicTUCR.SNPs.bed

### Intronic TUCRs ###
for bed in SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name

intersectBed -a BEDFiles/intronic_hg38.ultraConserved.bed -b $bed -wa -wb > $name.intronicTUCR.SNPs.bed

done

cat SNPBeds/*.intronicTUCR.SNPs.bed > merged_intronicTUCR_SNPs.bed

rm SNPBeds/*intronicTUCR.SNPs.bed

### Exonic_Intronic TUCRs ###
```

```
for bed in SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name

intersectBed -a BEDFiles/exonic_intronic_hg38.ultraConserved.bed -b $bed -wa -wb > $name.exonicintronic

done

cat SNPBeds/*.exonicintronicTUCR.SNPs.bed > merged_exonicintronicTUCR_SNPs.bed

rm SNPBeds/*exonicintronicTUCR.SNPs.bed

### Intergenic TUCRs ###

for bed in SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name

intersectBed -a BEDFiles/intergenic_hg38.ultraConserved.bed -b $bed -wa -wb > $name.intergenicTUCR.SNPs

done

cat SNPBeds/*.intergenicTUCR.SNPs.bed > merged_intergenicTUCR_SNPs.bed

rm SNPBeds/*intergenicTUCR.SNPs.bed

### coding genes ###

for bed in SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name

intersectBed -a CHESScoding.bed -b $bed -wa -wb > $name.TUCR.SNPs.bed

done

cat SNPBeds/*coding.SNPs.bed > merged_coding_SNPs.bed

rm SNPBeds/*coding.SNPs.bed

### lncRNAs ###

for bed in SNPs/SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name
```

```
intersectBed -a CHESSlncRNA.bed -b $bed -wa -wb > $name.lncRNA.SNPs.bed

done

cat *lncRNA.SNPs.bed >  SNPs/merged_lncRNA_SNPs.bed

rm *lncRNA.SNPs.bed
rm *CHESSlncRNA.bed

### antisense RNAs ###

for bed in SNPs/SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name

intersectBed -a CHESSantisense.bed -b $bed -wa -wb > $name.antisense.SNPs.bed

done

cat *antisense.SNPs.bed >  SNPs/merged_antisense_SNPs.bed

rm *antisense.SNPs.bed
rm CHESSantisense.bed

### miscRNAs ###

for bed in SNPs/SNPBeds/bed_chr*
do

name=$(echo "${bed%.*}")
echo $name

intersectBed -a CHESSmisc.bed -b $bed -wa -wb > $name.misc.SNPs.bed

done

cat *misc.SNPs.bed >  Inputs/figure_specific_files/Figure1_supplementary/SNPs/merged_misc_SNPs.bed

rm *misc.SNPs.bed
rm CHESSmisc.bed
```

This file counts the number of rows in each file. Each row represents a single SNP, thus providing a count of all overlapping SNPs by annotation category.

```
wc -l merged_TUCR_SNPs.bed

wc -l merged_exonicTUCR_SNPs.bed

wc -l merged_intronicTUCR_SNPs.bed

wc -l merged_exonicintronicTUCR_SNPs.bed
```

```
wc -l merged_intergenicTUCR_SNPs.bed

wc -l merged_coding_SNPs.bed

wc -l merged_lncRNA_SNPs.bed

wc -l merged_antisense_SNPs.bed

wc -l merged_misc_SNPs.bed
```

```
# tucr_SNP <- read.csv('Inputs/Figure1_supplementary/tucr_SNPs.csv')

# p <- ggplot(tucr_SNP,
# aes(x=reorder(annot,order),y=proportion*100,fill=reorder(annot,order))) +
# geom_bar(stat='identity', width = 0.7 ,color = 'black' ) +
# geom_text(aes(label=round(proportion*100,3)), vjust=1.6, color='black',
# size=5) +
# scale_fill_manual(values=c(paper_purple,paper_yellow,paper_blue,paper_green,paper_orange))
# + ggtitle(paste('TUCRs are less susceptible to variation than other genes'))
# + xlab('gene annotation') + ylab('Percent variant nucleotides (SNPs)') +
# theme( plot.title = element_text(size=rel(1.5), face='bold',hjust = 0.5),
# axis.title = element_text(size = rel(1.5), face='bold'), axis.text.x =
# element_text(size = rel(1.5),angle = 90), axis.text.y = element_text(size =
# rel(1.5)), legend.title = element_blank(), legend.position = 'none',
# panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
# panel.background = element_blank(), axis.line = element_line(colour =
# 'black'))

# ggsave(file=paste(outputdir,'/Figure1_Supplementary/','supplementary_figure_1f.png',sep=''),
# plot = print(p), width = 4, height = 5, dpi = 600)
```

Supplementary Figure 2.

Supplementary Figure 2A

Plotting fisher test results. See Chromatin Landscaping methods for additional code.

```
if(!dir.exists(paste(outputdir,"/Figure2_supplementary/",sep=""))){dir.create(paste(outputdir,"/Figure2_

filenames <- list.files("./Inputs/Figure2_Supplementary/fisher_test_results/")

#header <- data.frame(matrix(ncol = 5, nrow = 0))
header <- data.frame(matrix(ncol = 6, nrow = 0))

i <- 1

for(i in 1:length(filenames)){

print(i)

filename_n <- filenames[i]

print(filename_n)
```

```
annot <- str_match(filename_n,"\\s*(.*?)\\s*_genelist")[2]

chip <- str_match(filename_n, "[.]\\s*(.*?)\\s*.fisher")[2]

fisherdata <- readLines(paste("./Inputs/Figure2_Supplementary/fisher_test_results/",filename_n,sep=""))

dat <- as.data.frame(rbind(fisherdata[9],fisherdata[10])) %>%
  mutate(left = as.numeric(stri_extract_first_regex(V1, "[0-9]+")),
         right = as.numeric(stri_extract_last_regex(V1, "[0-9]+"))) %>%
  dplyr::select(-V1)

rownames(dat) <- c("top","bottom")

cont <- chisq.test(dat)$expected

test <- fisher.test(dat)

pvalue <- ifelse(round(test$p.value,20)<=0.05,"*","")

rbinder <- cbind(filename_n,annot,chip,round(cont[1,1],0),dat[1,1],pvalue)

header <- rbind(header,rbinder)}

colnames(header) <- c("filename","annot","chip","expected","observed","pvalue")

chromatindat <- header %>%
  mutate(order = ifelse(annot == "allTUCR",1,
         ifelse(annot == "exonic",2,
         ifelse(annot == "exonic-intronic",3,
         ifelse(annot == "intronic",4,
         ifelse(annot == "intergenic",5,
         ifelse(annot == "coding",6,
         ifelse(annot == "lncRNA",7,
         ifelse(annot == "antisense_RNA",8,
         ifelse(annot == "misc_RNA",9,
         ifelse(annot == "randomTUCR",10,NA)))))))))) %>%
  gather(key = "test", value = "value",-filename,-annot,-chip,-pvalue,-order) %>%
  mutate(pvalue = ifelse(test == "observed",pvalue,""),
         fillcolor = ifelse(test == "expected",paper_red2,paper_turq)) %>%
  arrange(order,test)

chromatindat$chip <- factor(chromatindat$chip,      # Reordering group factor levels
                            levels = c("pol2", "h3k4me3","ATAC","enhancers","h3k27ac"))

chromatindat$test <- factor(chromatindat$test,      # Reordering group factor levels
                            levels = c("expected","observed"))

chromatindat$fillcolor <- factor(chromatindat$fillcolor,      # Reordering group factor levels
                            levels = c(paper_red2,paper_turq))

chromatindat$annot <- factor(chromatindat$annot,      # Reordering group factor levels
                            levels = reorder(unique(chromatindat$annot),unique(chromatindat$order)))
```

```
chromatindat <- chromatindat  %>%
  mutate(label_positions = as.numeric(value)+(as.numeric(value)*.05))


  p <- ggplot(chromatindat, aes(x=chip,y=as.numeric(value),fill=test))   +
  geom_bar(stat="identity",
           #color = "black",
           width=0.7,position="dodge") +
  #geom_errorbar(aes(ymax=as.numeric(as.character(errorbar)), ymin=as.numeric(as.character(foldchange))),
  #              position=position_dodge(.9)) +
  geom_text(data=chromatindat,aes(x=chip,y=as.numeric(label_positions),label = pvalue), position = posi
  #scale_fill_identity(guide = 'legend', breaks = levels(chromatindat$test)) +
  #scale_fill_brewer(palette = "Spectral") +
  scale_fill_manual(values = c(paper_red2,paper_turq)) +
  #scale_y_continuous(expand=expansion(mult=c(0,0.15))) +
  #geom_text(
  #  aes(label=round(value,2)),
  #  vjust=1.6,
  #  color="black",
  #  size=rel(4))                                                    #+
  #ggtitle(paste("TUCRs are enriched for \n RNA Pol.II, H3K4me3,"))  +
  xlab("Spatial Relationship")                                        +
  ylab("Number of overlaps")                                         +
  #ggtitle(annot) +
  theme(
    #plot.title = element_text(
    #  size=rel(1.5),
    #  face="bold",hjust = 0.5),
    #plot.title = element_text(
    #  size = rel(1.5),hjust=0.5,
    #  face="bold"),
    strip.background = element_rect(fill="white"),
    plot.title = element_text(size=rel(3.0), face="bold",hjust = 0.5),
    axis.title = element_text(size = rel(3.0), face="bold"),
    axis.text = element_text(size = rel(2.0),angle = -90),
    strip.text = element_text(size = rel(3.0), face="bold"),
    legend.title = element_blank(),
    legend.position = "top",
    legend.text = element_text(size = rel(4.0)),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    panel.background = element_blank(),
    axis.line = element_line(colour = "black")) +
    facet_wrap(~annot, scales = "free",ncol=5)

ggsave(
    file=paste(outputdir,"/Figure2_Supplementary/supplementary_figure2.png",sep=""),
    plot = print(p),
    width = 25,
    height = 15,
    dpi = 600)
```

Supplementary Figure 3.

```
if (!dir.exists(paste(outputdir, "/Figure3_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure3_Supplementary/", sep = ""))
}
```

Supplementary Figure 3A

```
if(!dir.exists(paste(outputdir,"/Figure3_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure3_Supplementary/",sep=""))
}

disease <- "GBM"

normal <- "cortex"

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)
```

```
  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

#if(!is.na(filterannot)){
#  tpmcounts <- mergedcounts %>%
#  filter(tag.x == filterannot) %>%
#  mutate(length = end.x - start.x)
#}else{
tpmcounts <- mergedcounts %>%
  mutate(length = end.x - start.x)#}

tpmcounts.info <- tpmcounts %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

tpmcounts <- tpmcounts %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-id,-strand.x,-tag.x,-annot.x,-alias.x)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1000000

tpm <- function(counts,len,dep){
  #x <- tpmcounts/genelength
  #x2 <- t(t(x)/(seqdepth2))
  x <- counts/len
  return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts,genelength,seqdepth2)

tpm.df2 <- cbind(tpmcounts.info,tpm.df)

tpm.median <- as.data.frame(tpm.df)
tpm.median <- tpm.median %>%
  dplyr::summarize(median_tpm = rowMedians(tpm.df)) %>%
  dplyr::mutate(countif = ifelse(median_tpm >=1,1,0),proportion = sum(countif,na.rm = TRUE)/n())
tpm.median <- cbind(tpmcounts.info,tpm.median)
tpm.median <- tpm.median %>% dplyr::select(id,median_tpm,countif,proportion)
```

```
proportion.df <- round(as.numeric(tpm.median$proportion[1]),3)*100

#write.csv(tpm.df2,paste(outputdir,"/TUCR_Database/SummaryTables/gbm_tpms_allTUCRs.csv",sep=""))

write.csv(tpm.median,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_median_tpm_al

heatmap.df <- as.data.frame(tpm.df) %>%
  dplyr::select(-length)
geneids <- tpm.df2 %>%
  dplyr::select(id)
annotids <- tpm.df2 %>%
  dplyr::select(annot)
#means <- apply(heatmap.df,1,mean)
means <- apply(heatmap.df,1,median)

heatmap.df2 <- as.data.frame(apply(heatmap.df,1:2,function(x) {ifelse(x>=10,10,x)}))
heatmap.df2 <- cbind(geneids,means,annotids,heatmap.df2)

heatmap.df2 <- heatmap.df2 %>%
  gather(key = "Sample", value = "tpm",-id,-means,-annot)

heatmap.df2$annot <- factor(heatmap.df2$annot,      # Reordering group factor levels
                            levels = c("protein_coding","lncRNA","antisense_RNA","misc_RNA","exonic","exoni

p <- ggplot(data = heatmap.df2,mapping = aes(x = Sample,y = reorder(id,means), fill = tpm)) +
  geom_tile() +
  #geom_boxplot() +
  scale_fill_gradientn(colours = c(paper_blue, "white",paper_red2), values = c(0,0.1,1)) +
  #ggtitle(paste("All TUCRs (",proportion.df,"%)",sep="")) + xlab(paste("Samples (n = ",ncol(heatmap.df
  ylab(paste("genes (n = ",nrow(heatmap.df),")",sep="")) +
  theme(plot.title = element_blank(),
        panel.grid = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        strip.text = element_text(size=rel(1.5))) +
  facet_wrap(~annot,scales="free",ncol=9)



ggsave(p,file=paste(outputdir,"/Figure3_Supplementary/supplementary_figure3a.png",sep=""), width = 20, h

tpm.TUCRs2 <- tpm.df2

#write.csv(tpm.TUCRs2,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"_tpm_TUCRs2.csv",sep=""))
```

Supplementary Figure 3B

```
if(!dir.exists(paste(outputdir,"/Figure3_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure3_Supplementary/",sep=""))
}

disease <- "LGG"
```

```
normal <- "cortex"

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
```

```
}

#if(!is.na(filterannot)){
#  tpmcounts <- mergedcounts %>%
#  filter(tag.x == filterannot) %>%
#  mutate(length = end.x - start.x)
#}else{
tpmcounts <- mergedcounts %>%
  mutate(length = end.x - start.x)#}

tpmcounts.info <- tpmcounts %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

tpmcounts <- tpmcounts %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-id,-strand.x,-tag.x,-annot.x,-alias.x)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1000000

tpm <- function(counts,len,dep){
  #x <- tpmcounts/genelength
  #x2 <- t(t(x)/(seqdepth2))
  x <- counts/len
  return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts,genelength,seqdepth2)

tpm.df2 <- cbind(tpmcounts.info,tpm.df)

tpm.median <- as.data.frame(tpm.df)
tpm.median <- tpm.median %>%
  dplyr::summarize(median_tpm = rowMedians(tpm.df)) %>%
  dplyr::mutate(countif = ifelse(median_tpm >=1,1,0),proportion = sum(countif,na.rm = TRUE)/n())
tpm.median <- cbind(tpmcounts.info,tpm.median)
tpm.median <- tpm.median %>% dplyr::select(id,median_tpm,countif,proportion)

proportion.df <- round(as.numeric(tpm.median$proportion[1]),3)*100

#write.csv(tpm.df2,paste(outputdir,"/TUCR_Database/SummaryTables/gbm_tpms_allTUCRs.csv",sep=""))

write.csv(tpm.median,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_median_tpm_a

heatmap.df <- as.data.frame(tpm.df) %>%
  dplyr::select(-length)
geneids <- tpm.df2 %>%
  dplyr::select(id)
annotids <- tpm.df2 %>%
  dplyr::select(annot)
#means <- apply(heatmap.df,1,mean)
```

```
means <- apply(heatmap.df,1,median)

heatmap.df2 <- as.data.frame(apply(heatmap.df,1:2,function(x) {ifelse(x>=10,10,x)}))
heatmap.df2 <- cbind(geneids,means,annotids,heatmap.df2)

heatmap.df2 <- heatmap.df2 %>%
  gather(key = "Sample", value = "tpm",-id,-means,-annot)

heatmap.df2$annot <- factor(heatmap.df2$annot,      # Reordering group factor levels
                            levels = c("protein_coding","lncRNA","antisense_RNA","misc_RNA","exonic","exon

p <- ggplot(data = heatmap.df2,mapping = aes(x = Sample,y = reorder(id,means), fill = tpm)) +
  geom_tile() +
  #geom_boxplot() +
  scale_fill_gradientn(colours = c(paper_blue, "white",paper_red2), values = c(0,0.1,1)) +
  #ggtitle(paste("All TUCRs (",proportion.df,"%)",sep="")) + xlab(paste("Samples (n = ",ncol(heatmap.df
  ylab(paste("genes (n = ",nrow(heatmap.df),")",sep="")) +
  theme(plot.title = element_blank(),
        panel.grid = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        strip.text = element_text(size=rel(1.5))) +
  facet_wrap(~annot,scales="free",ncol=9)



ggsave(p,file=paste(outputdir,"/Figure3_Supplementary/supplementary_figure3b.png",sep=""), width = 20, h
```

Supplementary Figure 4

```
if (!dir.exists(paste(outputdir, "/Figure4_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure4_Supplementary/", sep = ""))
}
```

Supplementary Figure 4A

Identifying differentially expressed TUCRs with DESeq2

```
## read data

gbm_countfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_mergedcounts.txt",
    sep = "")

lgg_countfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_mergedcounts.txt",
    sep = "")

cortex_countfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_mergedcounts.txt",
    sep = "")


gbm_metadatafile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_tcga_metadata.csv",
    sep = "")
```

```
lgg_metadatafile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_tcga_metadata.csv",
    sep = "")

cortex_metadatafile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_gtex_metadata.csv",
    sep = "")


gbm_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_seqdepth_counts.csv",
    sep = "")

lgg_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_seqdepth_counts.csv",
    sep = "")

cortex_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_seqdepth_counts.csv",
    sep = "")


## Merge Data


if (!is.na(cortex_countfile)) {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    lgg_mergedcounts <- read.table(lgg_countfile, header = TRUE)

    cortex_normalcounts <- read.table(cortex_countfile, header = TRUE)

    cortex_normalcounts <- cortex_normalcounts[, 9:ncol(cortex_normalcounts)]

    gbm_mergedcounts <- cbind(gbm_mergedcounts, cortex_normalcounts) %>%
        distinct()

    lgg_mergedcounts <- cbind(lgg_mergedcounts, cortex_normalcounts) %>%
        distinct()

    rm(cortex_normalcounts)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    lgg_metadata <- read_csv(file = lgg_metadatafile)

    cortex_metadata <- read_csv(file = cortex_metadatafile)

    gbm_metadata <- rbind(gbm_metadata, cortex_metadata)

    lgg_metadata <- rbind(lgg_metadata, cortex_metadata)

    rm(cortex_metadata)

    gbm_seqdepth <- read.csv(file = gbm_seqdepthfile)

    lgg_seqdepth <- read.csv(file = lgg_seqdepthfile)
```

```
    cortex_seqdepth <- read.csv(file = cortex_seqdepthfile)

    gbm_seqdepth <- rbind(gbm_seqdepth, cortex_seqdepth)

    lgg_seqdepth <- rbind(lgg_seqdepth, cortex_seqdepth)

    rm(cortex_seqdepth)

} else {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    lgg_mergedcounts <- read.table(lgg_countfile, header = TRUE)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    lgg_metadata <- read_csv(file = lgg__metadatafile)

    gbm_seqdepth <- read.csv(file = gbm__seqdepthfile)

    lgg_seqdepth <- read.csv(file = lgg__seqdepthfile)
}

gbm_mergedcounts_deseq2 <- gbm_mergedcounts %>%
    filter(tag == "TUCR" & annot != "random")

lgg_mergedcounts_deseq2 <- lgg_mergedcounts %>%
    filter(tag == "TUCR" & annot != "random")

gbm_mergedcounts_deseq2_countsonly <- gbm_mergedcounts_deseq2[, 9:ncol(gbm_mergedcounts_deseq2)]

lgg_mergedcounts_deseq2_countsonly <- lgg_mergedcounts_deseq2[, 9:ncol(lgg_mergedcounts_deseq2)]

rownames(gbm_mergedcounts_deseq2_countsonly) <- gbm_mergedcounts_deseq2$id

rownames(lgg_mergedcounts_deseq2_countsonly) <- lgg_mergedcounts_deseq2$id

gbm_mergedcounts_deseq2_info <- gbm_mergedcounts_deseq2[, 1:8]

lgg_mergedcounts_deseq2_info <- lgg_mergedcounts_deseq2[, 1:8]

gbm_mergedcounts_deseq2_info$length <- (gbm_mergedcounts_deseq2$end - gbm_mergedcounts_deseq2$start)/10

lgg_mergedcounts_deseq2_info$length <- (lgg_mergedcounts_deseq2$end - lgg_mergedcounts_deseq2$start)/10

dds <- DESeqDataSetFromMatrix(countData = gbm_mergedcounts_deseq2_countsonly, colData = gbm_metadata,
    design = ~dex)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)
```

```
write_csv(res, file = paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs.csv",
    sep = ""))

dds <- DESeqDataSetFromMatrix(countData = lgg_mergedcounts_deseq2_countsonly, colData = lgg_metadata,
    design = ~dex)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)

write_csv(res, file = paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs.csv",
    sep = ""))
```

```
### Volcano plot

volcanoplot_expression <- function (res,
                          genes = "all",
                          title = "Deregulated TUCRs in TCGA Gliomas",
                          output = "",
                          height = 7,
                          width = 14,
                          dpi = 600,
                          annot_filter = ""){
  #res <- res_TUCR
  #genes <- "all"
  #annot_filter = ""
  #genes = c("uc.110")
  #title = "Deregulated Genes"

  res <- res %>%
  mutate(dereg = ifelse(log2FoldChange >=1, "upregulated",
                    ifelse(log2FoldChange <=-1,"downregulated","unchanged")),
         deregcount = ifelse(log2FoldChange >=1 & padj <= 0.05, 1,
                    ifelse(log2FoldChange <=-1 & padj <= 0.05,-1,0))) %>%
  dplyr::filter(padj != 0 | alias == "uc.110") %>%
  dplyr::group_by(alias) %>%
  dplyr::mutate(sum = sum(deregcount,na.rm=TRUE)) %>%
  ungroup() %>%
  dplyr::mutate(deregcategory = ifelse(sum == 2, "Up-Both",
                                ifelse(sum == -2, "Down-Both",
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM","Up-GBM",
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", "Up-LGG"
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", "Down
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", "Down
                color = ifelse(sum == 2, paper_darkpink,
                                ifelse(sum == -2, paper_lightblue,
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM",paper_red,
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", paper_pi
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", paper
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", paper
                                ifelse(disease == "GBM",paper_black,paper_gray)))))),
                newdisease = ifelse(str_detect(deregcategory,"Both"),"BOTH",disease),
```

```
                diseasefilter = ifelse((disease == "LGG" & newdisease == "BOTH"),0,1)) %>%
    dplyr::filter(diseasefilter == 1)

  res_summary <- res %>%
    group_by(color) %>%
    dplyr::summarize(n = n())

if(annot_filter == ""){}else{
  res %>%
    filter(annot == annot_filter)
}


res_110_up <- res %>%
   filter(color != paper_gray & color != paper_black & log2FoldChange > 0 | alias == "uc.110") %>%
   dplyr::arrange(desc(log2FoldChange)) %>%
   dplyr::group_by(disease) %>%
   dplyr::mutate(order_rank= row_number()) %>%
   mutate(maxrank = max(order_rank,na.rm=TRUE)) %>%
   ungroup() %>%
   filter(order_rank <= 5 | alias == "uc.110")

res_110_down <- res %>%
   filter(color != paper_gray & color != paper_black  & log2FoldChange < 0 | alias == "uc.110") %>%
   dplyr::arrange(log2FoldChange) %>%
   dplyr::group_by(disease) %>%
   dplyr::mutate(order_rank= row_number()) %>%
   mutate(maxrank = max(order_rank,na.rm=TRUE)) %>%
   ungroup() %>%
   filter(order_rank <= 5 | alias == "uc.110")

p <- ggplot(res) +
        #geom_point(aes(x=log2FoldChange, y=-log10(pvalue),col=color)) +
        #geom_hline(yintercept = 1.30,linetype = 2) +
        geom_point(aes(x=log2FoldChange,y=-log10(padj),col=color)) +
        geom_hline(yintercept = -log10(0.05),linetype = 2,size=0.5) +
        geom_vline(xintercept = -1,linetype = 2,size=0.5) +
        geom_vline(xintercept = 1,linetype = 2,size=0.5) +
        scale_color_identity(guide = "deregcategory", labels = res$deregcategory, breaks = res$color) +
        #ggtitle(title) +
        guides(color = guide_legend(override.aes = list(size=5))) +
        geom_text_repel(data= res_110_up,aes(x=log2FoldChange,y=-log10(padj),color=paper_black, label =
        geom_text_repel(data= res_110_down,aes(x=log2FoldChange,y=-log10(padj),color=paper_black, label
        labs(x = "log2 fold change of TUCR in tumors",
        #y = "-log10 adjusted p-value",
        y = "-log10 of adjusted p-value",
        color = "Legend") +
        theme(plot.title = element_text(size = rel(2.0), hjust = 0.5, face="bold",margin=margin(0,0,0,0
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
```

```
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_text(size=10)) +
  coord_cartesian(clip = "off") +
  facet_wrap(~newdisease+annot,scales="free",ncol=4)

p

ggsave(plot = print(p),filename = output, width = width, height = height, dpi = 600)

}
```

```
res_TUCR_LGG <- read_csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs.csv",
    sep = ""), col_names = TRUE)

res_TUCR_GBM <- read_csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs.csv",
    sep = ""), col_names = TRUE)

tucr_annot <- read_delim("Inputs/general_files/bedfiles/hg38.ultraconserved.bed",
    col_names = TRUE, delim = "\t")

res_TUCR_LGG_2 <- res_TUCR_LGG %>%
    separate(row, into = c("alias", "kibble"), sep = "___") %>%
    dplyr::select(-kibble) %>%
    left_join(tucr_annot, by = "alias") %>%
    filter(!is.na(chrom)) %>%
    dplyr::select(alias, annot, log2FoldChange, padj) %>%
    mutate(disease = "LGG")

res_TUCR_GBM_2 <- res_TUCR_GBM %>%
    separate(row, into = c("alias", "kibble"), sep = "___") %>%
    dplyr::select(-kibble) %>%
    left_join(tucr_annot, by = "alias") %>%
    filter(!is.na(chrom)) %>%
    dplyr::select(alias, annot, log2FoldChange, padj) %>%
    mutate(disease = "GBM")

res_TUCR <- rbind(res_TUCR_GBM_2, res_TUCR_LGG_2)

volcanoplot_expression(res_TUCR, genes = "all", output = paste(outputdir, "/Figure4_supplementary/supple
    sep = ""), height = 9, width = 16, dpi = 600, annot = "")
```

Supplementary Figure 4B

Completing survival analysis for TUCRs

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 5

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
```

```
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
```

```
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
```

```
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {

    km_countdata <- z_rna
    colnames(km_countdata) <- metadata$id[t_index]
```

```
posdata$median <- rowMedians(survcounts)

# km_countdata2 <- km_countdata[,complete.cases(clinical2)]

probs <- c(0.25, 0.5, 0.75)
q <- rowQuantiles(km_countdata, probs = probs)
posdata$n25 <- q[, 1]
posdata$n75 <- q[, 3]
posdata <- posdata %>%
    dplyr::select(TUCR = alias.x, median, n25, n75)
km_TUCRs <- cbind(posdata, km_countdata)

km_TUCRs <- km_TUCRs %>%
    gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
    mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
    distinct %>%
    dplyr::select(TUCR, median, id, group) %>%
    left_join(clinical2, by = "id") %>%
    dplyr::filter(median != 0)

TUCRids <- as.character(posdata$TUCR)
i <- 1

ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
    "pvalue", "method"))))

for (i in 1:length(TUCRids)) {
    print(i)
    print(TUCRids[i])

    skip_to_next <- FALSE
    TUCRsurv <- as.character(TUCRids[i])
    if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
        dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
    }

    # TUCR <- 'uc.1'
    kmdata <- km_TUCRs %>%
        dplyr::filter(TUCR == TUCRsurv) %>%
        dplyr::select(TUCR, id, group, time, status)
    fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
        skip_to_next <<- TRUE
    })
    p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
        risk.table = TRUE), error = function(e) {
        skip_to_next <<- TRUE
    })
    # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
    # TRUE,risk.table = TRUE)

    if (skip_to_next == TRUE) {
        rbinder <- cbind(as.character(TUCRsurv), NA, NA)
        ptable[i, ] <- rbinder
```

```
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}
```

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 6

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
```

```
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
```

```
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
```

```
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {

    km_countdata <- z_rna
    colnames(km_countdata) <- metadata$id[t_index]
    posdata$median <- rowMedians(survcounts)

    # km_countdata2 <- km_countdata[,complete.cases(clinical2)]

    probs <- c(0.25, 0.5, 0.75)
    q <- rowQuantiles(km_countdata, probs = probs)
    posdata$n25 <- q[, 1]
    posdata$n75 <- q[, 3]
    posdata <- posdata %>%
        dplyr::select(TUCR = alias.x, median, n25, n75)
    km_TUCRs <- cbind(posdata, km_countdata)

    km_TUCRs <- km_TUCRs %>%
        gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
        mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
        distinct %>%
        dplyr::select(TUCR, median, id, group) %>%
        left_join(clinical2, by = "id") %>%
        dplyr::filter(median != 0)

    TUCRids <- as.character(posdata$TUCR)
    i <- 1

    ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
        "pvalue", "method"))))
```

```
    for (i in 1:length(TUCRids)) {
        print(i)
        print(TUCRids[i])

        skip_to_next <- FALSE
        TUCRsurv <- as.character(TUCRids[i])
        if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
            dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
        }

        # TUCR <- 'uc.1'
        kmdata <- km_TUCRs %>%
            dplyr::filter(TUCR == TUCRsurv) %>%
            dplyr::select(TUCR, id, group, time, status)
        fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
            skip_to_next <<- TRUE
        })
        p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
            risk.table = TRUE), error = function(e) {
            skip_to_next <<- TRUE
        })
        # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
        # TRUE,risk.table = TRUE)

        if (skip_to_next == TRUE) {
            rbinder <- cbind(as.character(TUCRsurv), NA, NA)
            ptable[i, ] <- rbinder
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}
```

Writing a script to generate a volcano plot for survival

```
### Volcano plot

volcanosurv <- function (res,genes = "all",title = "TUCRs correlated with survival in gliomas", output =
  #res <- res_surv
  #genes = c("uc.110","uc.62")
  #title = paste("TUCR correlation with patient survival in ",disease,sep="")
```

```
  #output = paste(outputdir,"/intergenic_tucr_results_volcanosuru.png",sep="")
  i <- 1
  vres <- res %>%
    filter(abs(estimate) <=1)
  vres <- vres %>% mutate(gene="",Survival="",color="")
  for (i in 1:length(vres$id)){
  ifelse(is.na(vres$pvalue[i]),vres$pvalue[i] <- 1,vres$pvalue[i] <- vres$pvalue[i])

  ifelse(genes!="all",ifelse(!is.na(match(vres$id[i],genes)),vres$gene[i] <- as.character(vres$id[i]),

  ifelse((vres$pvalue[i] <0.05 & vres$p.value[i] <0.05 & vres$estimate[i] < -0),{vres$Survival[i] <- "Si
  ifelse((vres$pvalue[i] <0.05 & vres$p.value[i] <0.05 & vres$estimate[i] > 0),{vres$Survival[i] <- "Sig
  ifelse((vres$pvalue[i] >0.05 & vres$p.value[i] <0.05 & vres$estimate[i] < -0),{vres$Survival[i] <- "Si
  ifelse((vres$pvalue[i] >0.05 & vres$p.value[i] <0.05 & vres$estimate[i] > 0),{vres$Survival[i] <- "Sig
  ifelse((vres$pvalue[i] <0.05 & vres$p.value[i] >0.05),{vres$Survival[i] <- "Significant (KM)";vres$col
  ifelse({vres$Survival[i] <- "Not significant";vres$color[i] <- "lightgray";})))))}
  #{vres$Survival[i] <- "Not significant";vres$color[i] <- "lightgray";}))}

  vres <- vres %>%
  arrange(desc(Survival))


  ## plot
if(repel==TRUE){p <- ggplot(vres) +
        geom_point(aes(x=estimate, y=-log10(as.numeric(p.value)),col=color)) +
        scale_color_identity(guide = "legend", labels = vres$Survival, breaks = vres$color) +
        ggtitle(title) +
        labs(x = "Cox Estimated Proportional Hazard",
        y = "-log10 P-Value (CH)",
        color = "Legend")+
        theme(plot.title = element_blank(),
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=10)) +
        geom_text_repel(aes(x=estimate, y=-log10(as.numeric(p.value)),label = gene),force=50) +
        facet_wrap(~disease.y+annot.x,scales="free",ncol=4)
}else{
  p <- ggplot(vres) +
        geom_point(aes(x=estimate, y=-log10(as.numeric(p.value)),col=color)) +
    scale_color_identity(guide = "legend", labels = vres$Survival, breaks = vres$color) +
        + ggtitle(title) +
        labs(x = "Cox Estimated Proportional Hazard",
        y = "-log10 P-Value (CH)",
        color = "Legend") +
        theme(plot.title = element_blank(),
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
```

```
            #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=10))}
ggsave(output, width = width, height = height, dpi = dpi) +
        facet_wrap(~disease.y+annot.x,scales="free",ncol=4)
}
```

```
genes <- ""

lgg_surv_TUCR <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_survival_coxph_allTUCRs
    sep = ""), header = TRUE) %>%
    mutate(disease = "LGG")

lgg_ptable <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_survival_kpm_allTUCRs.csv
    sep = ""), header = TRUE) %>%
    dplyr::select(id = TUCR, pvalue, method) %>%
    mutate(disease = "LGG")

gbm_surv_TUCR <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_survival_coxph_allTUCRs
    sep = ""), header = TRUE) %>%
    mutate(disease = "GBM")

gbm_ptable <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_survival_kpm_allTUCRs.csv
    sep = ""), header = TRUE) %>%
    dplyr::select(id = TUCR, pvalue, method) %>%
    mutate(disease = "GBM")

surv_TUCR <- rbind(gbm_surv_TUCR, lgg_surv_TUCR)

ptable <- rbind(gbm_ptable, lgg_ptable)

colnames(ptable) <- c("alias.x", "pvalue", "method", "disease")

res_surv <- inner_join(ptable, surv_TUCR, by = "alias.x")

res_surv_sum <- res_surv

res_surv_sum <- res_surv_sum %>%
    mutate(gene = "", Survival = "", color = "")

annot_unique <- as.character(unique(res_surv_sum$annot))

annotation <- "All"

res_surv_sum2 <- res_surv_sum

for (i in 1:length(res_surv_sum2$alias)) {
    # print(as.character(res_surv_sum2$alias.x[i]))
    ifelse(is.na(res_surv_sum2$pvalue[i]), res_surv_sum2$pvalue[i] <- 1, res_surv_sum2$pvalue[i] <- res

    ifelse(genes != "all", ifelse(!is.na(match(res_surv_sum2$alias[i], genes)), res_surv_sum2$gene[i] <
        ""), res_surv_sum2$gene[i] <- res_surv_sum2$alias[i])

    ifelse((res_surv_sum2$pvalue[i] < 0.05 & res_surv_sum2$p.value[i] < 0.05 & res_surv_sum2$estimate[i
```

```
        0), {
        res_surv_sum2$Survival[i] <- "Significant (Both, Good Prognosis)"
        res_surv_sum2$color[i] <- print(paper_green)
    }, ifelse((res_surv_sum2$pvalue[i] < 0.05 & res_surv_sum2$p.value[i] < 0.05 &
        res_surv_sum2$estimate[i] > 0), {
        res_surv_sum2$Survival[i] <- "Significant (Both, Poor Prognosis)"
        res_surv_sum2$color[i] <- print(paper_red)
    }, ifelse((res_surv_sum2$pvalue[i] > 0.05 & res_surv_sum2$p.value[i] < 0.05 &
        res_surv_sum2$estimate[i] < 0), {
        res_surv_sum2$Survival[i] <- "Significant (CH, Good Prognosis)"
        res_surv_sum2$color[i] <- print(paper_green)
    }, ifelse((res_surv_sum2$pvalue[i] > 0.05 & res_surv_sum2$p.value[i] < 0.05 &
        res_surv_sum2$estimate[i] > 0), {
        res_surv_sum2$Survival[i] <- "Significant (CH, Poor Prognosis)"
        res_surv_sum2$color[i] <- print(paper_red)
    }, ifelse((res_surv_sum2$pvalue[i] < 0.05 & res_surv_sum2$p.value[i] > 0.05),
        {
            res_surv_sum2$Survival[i] <- "Significant (KM)"
            res_surv_sum2$color[i] <- "black"
        }, {
            res_surv_sum2$Survival[i] <- "Not significant"
            res_surv_sum2$color[i] <- "lightgray"
        })))))
}

# res_surv_sum4 <- res_surv_sum2 %>% group_by(Survival) %>%
# dplyr::summarise(count = n(),color) %>% distinct() %>%
# arrange(desc(Survival))

# write.csv(res_surv_sum4,file=paste(outputdir,'/SurvivalAnalysis/SummaryFigures/BarGraphs/',disease,'_

# p <- ggplot(res_surv_sum4, aes(x=Survival,y=count,fill=color)) +
# geom_bar(stat='identity', width = 0.7) + scale_fill_identity(guide =
# 'legend', labels = res_surv_sum2$Survival, breaks = res_surv_sum2$color) +
# labs(y = '# of TUCRs', fill = 'Legend') + theme( plot.title =
# element_text(size=rel(1.5), face='bold',hjust = 0.5), axis.title =
# element_text(size = rel(1.25), face='bold'), panel.grid.major =
# element_blank(), panel.grid.minor = element_blank(), panel.background =
# element_blank(), axis.line = element_line(colour = 'black'), axis.text.x =
# element_blank())

# ggsave(file=paste(outputdir,'/SurvivalAnalysis/SummaryFigures/BarGraphs/',disease,'_',annotation,'_tu
# = print(p), width = 5, height = 2.5, dpi = 600)

res_surv_sum3 <- res_surv_sum

volcanosurv(res_surv_sum3, genes = "", output = paste(outputdir, "/Figure4_Supplementary/supplementary_
    sep = ""))
```

Supplementary Figure 5

```
if (!dir.exists(paste(outputdir, "/Figure5_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure5_Supplementary/", sep = ""))
```

```
}
```

Supplementary Figure 5A

```
Figure_copy <- list.files("./Inputs/Figure5_Supplementary/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure5_Supplementary/", "supplementary_figure5a.png",
    sep = ""))) {
    file.copy(paste0("./Inputs/Figure5_Supplementary/pregenerated_figures/", Figure_copy,
        sep = ""), paste("./", outputdir, "/Figure5_Supplementary/", sep = ""))
}
```

Supplementary Figure 5B

```
## read data

gbm_countfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_mergedcounts.txt",
    sep = "")

lgg_countfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_mergedcounts.txt",
    sep = "")

cortex_countfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_mergedcounts.txt",
    sep = "")


gbm_metadatafile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_tcga_metadata.csv",
    sep = "")

lgg_metadatafile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_tcga_metadata.csv",
    sep = "")

cortex_metadatafile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_gtex_metadata.csv",
    sep = "")


gbm_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/GBM/GBM_seqdepth_counts.csv",
    sep = "")

lgg_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/LGG/LGG_seqdepth_counts.csv",
    sep = "")

cortex_seqdepthfile <- paste("Inputs/general_files/sequencingfiles/cortex/cortex_seqdepth_counts.csv",
    sep = "")


## Merge Data


if (!is.na(cortex_countfile)) {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    lgg_mergedcounts <- read.table(lgg_countfile, header = TRUE)
```

```
    cortex_normalcounts <- read.table(cortex_countfile, header = TRUE)

    cortex_normalcounts <- cortex_normalcounts[, 9:ncol(cortex_normalcounts)]

    gbm_mergedcounts <- cbind(gbm_mergedcounts, cortex_normalcounts) %>%
        distinct()

    lgg_mergedcounts <- cbind(lgg_mergedcounts, cortex_normalcounts) %>%
        distinct()

    rm(cortex_normalcounts)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    lgg_metadata <- read_csv(file = lgg_metadatafile)

    cortex_metadata <- read_csv(file = cortex_metadatafile)

    gbm_metadata <- rbind(gbm_metadata, cortex_metadata)

    lgg_metadata <- rbind(lgg_metadata, cortex_metadata)

    rm(cortex_metadata)

    gbm_seqdepth <- read.csv(file = gbm_seqdepthfile)

    lgg_seqdepth <- read.csv(file = lgg_seqdepthfile)

    cortex_seqdepth <- read.csv(file = cortex_seqdepthfile)

    gbm_seqdepth <- rbind(gbm_seqdepth, cortex_seqdepth)

    lgg_seqdepth <- rbind(lgg_seqdepth, cortex_seqdepth)

    rm(cortex_seqdepth)

} else {
    gbm_mergedcounts <- read.table(gbm_countfile, header = TRUE)

    lgg_mergedcounts <- read.table(lgg_countfile, header = TRUE)

    gbm_metadata <- read_csv(file = gbm_metadatafile)

    lgg_metadata <- read_csv(file = lgg__metadatafile)

    gbm_seqdepth <- read.csv(file = gbm__seqdepthfile)

    lgg_seqdepth <- read.csv(file = lgg__seqdepthfile)
}

gbm_mergedcounts_deseq2 <- gbm_mergedcounts %>%
    filter(tag == "TUCR" & annot != "random")
```

```
lgg_mergedcounts_deseq2 <- lgg_mergedcounts %>%
    filter(tag == "TUCR" & annot != "random")

gbm_mergedcounts_deseq2_countsonly <- gbm_mergedcounts_deseq2[, 9:ncol(gbm_mergedcounts_deseq2)]

lgg_mergedcounts_deseq2_countsonly <- lgg_mergedcounts_deseq2[, 9:ncol(lgg_mergedcounts_deseq2)]

rownames(gbm_mergedcounts_deseq2_countsonly) <- gbm_mergedcounts_deseq2$id

rownames(lgg_mergedcounts_deseq2_countsonly) <- lgg_mergedcounts_deseq2$id

gbm_mergedcounts_deseq2_info <- gbm_mergedcounts_deseq2[, 1:8]

lgg_mergedcounts_deseq2_info <- lgg_mergedcounts_deseq2[, 1:8]

gbm_mergedcounts_deseq2_info$length <- (gbm_mergedcounts_deseq2$end - gbm_mergedcounts_deseq2$start)/10

lgg_mergedcounts_deseq2_info$length <- (lgg_mergedcounts_deseq2$end - lgg_mergedcounts_deseq2$start)/10

dds <- DESeqDataSetFromMatrix(countData = gbm_mergedcounts_deseq2_countsonly, colData = gbm_metadata,
    design = ~IDH1status)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)

write_csv(res, file = paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs_byIDH1statu
    sep = ""))

dds <- DESeqDataSetFromMatrix(countData = lgg_mergedcounts_deseq2_countsonly, colData = lgg_metadata,
    design = ~IDH1status)

dds <- DESeq(dds)

res <- results(dds, tidy = TRUE)

res <- as_tibble(res)

write_csv(res, file = paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs_byIDH1statu
    sep = ""))
```

```
### Volcano plot

volcanoplot_expression <- function (res,
                          genes = "all",
                          title = "Deregulated TUCRs in TCGA Gliomas",
                          output = "",
                          height = 7,
                          width = 14,
                          dpi = 600,
                          annot_filter = ""){
  #res <- res_TUCR
```

```
  #genes <- "all"
  #annot_filter = ""
  #genes = c("uc.110")
  #title = "Deregulated Genes"

  res <- res %>%
  mutate(dereg = ifelse(log2FoldChange >=1, "upregulated",
                    ifelse(log2FoldChange <=-1,"downregulated","unchanged")),
         deregcount = ifelse(log2FoldChange >=1 & padj <= 0.05, 1,
                    ifelse(log2FoldChange <=-1 & padj <= 0.05,-1,0))) %>%
  dplyr::filter(padj != 0 | alias == "uc.110") %>%
  dplyr::group_by(alias) %>%
  dplyr::mutate(sum = sum(deregcount,na.rm=TRUE)) %>%
  ungroup() %>%
  dplyr::mutate(deregcategory = ifelse(sum == 2, "Up-Both",
                                ifelse(sum == -2, "Down-Both",
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM","Up-GBM",
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", "Up-LGG"
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", "Down
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", "Down
                color = ifelse(sum == 2, paper_darkpink,
                                ifelse(sum == -2, paper_lightblue,
                                ifelse(sum == 1 & dereg == "upregulated" & disease == "GBM",paper_red,
                                ifelse(sum == 1  & dereg == "upregulated" &  disease == "LGG", paper_pi
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "LGG", paper
                                ifelse(sum == -1  & dereg == "downregulated" &  disease == "GBM", paper
                                ifelse(disease == "GBM",paper_black,paper_gray)))))),
                newdisease = ifelse(str_detect(deregcategory,"Both"),"BOTH",disease),
                diseasefilter = ifelse((disease == "LGG" & newdisease == "BOTH"),0,1))

  res_summary <- res %>%
    group_by(color) %>%
    dplyr::summarize(n = n())

if(annot_filter == ""){}else{
  res %>%
    filter(annot == annot_filter)
}


res_110_up <- res %>%
   filter(color != paper_gray & color != paper_black & log2FoldChange > 0 | alias == "uc.110") %>%
   dplyr::arrange(desc(log2FoldChange)) %>%
   dplyr::group_by(disease) %>%
   dplyr::mutate(order_rank= row_number()) %>%
   mutate(maxrank = max(order_rank,na.rm=TRUE)) %>%
   ungroup() %>%
   filter(order_rank <= 5 | alias == "uc.110")

res_110_down <- res %>%
   filter(color != paper_gray & color != paper_black  & log2FoldChange < 0 | alias == "uc.110") %>%
   dplyr::arrange(log2FoldChange) %>%
   dplyr::group_by(disease) %>%
```

```
   dplyr::mutate(order_rank= row_number()) %>%
   mutate(maxrank = max(order_rank,na.rm=TRUE)) %>%
   ungroup() %>%
   filter(order_rank <= 5 | alias == "uc.110")

p <- ggplot(res) +
        #geom_point(aes(x=log2FoldChange, y=-log10(pvalue),col=color)) +
        #geom_hline(yintercept = 1.30,linetype = 2) +
        geom_point(aes(x=log2FoldChange,y=-log10(padj),col=color)) +
        geom_hline(yintercept = -log10(0.05),linetype = 2,size=0.5) +
        geom_vline(xintercept = -1,linetype = 2,size=0.5) +
        geom_vline(xintercept = 1,linetype = 2,size=0.5) +
        scale_color_identity(guide = "deregcategory", labels = res$deregcategory, breaks = res$color) +
        #ggtitle(title) +
        guides(color = guide_legend(override.aes = list(size=5))) +
        geom_text_repel(data= res_110_up,aes(x=log2FoldChange,y=-log10(padj),color=paper_black, label =
        geom_text_repel(data= res_110_down,aes(x=log2FoldChange,y=-log10(padj),color=paper_black, label
        labs(x = "log2 fold change of TUCR in tumors",
        #y = "-log10 adjusted p-value",
        y = "-log10 of adjusted p-value",
        color = "Legend") +
        theme(plot.title = element_text(size = rel(2.0), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_text(size=10)) +
  coord_cartesian(clip = "off") +
  facet_wrap(~disease,scales="free",ncol=3)

p

ggsave(plot = print(p),filename = output, width = width, height = height, dpi = 600)

}
```

```
res_TUCR_LGG <- read_csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_DESeq2_allTUCRs_byIDH1st
    sep = ""), col_names = TRUE)

res_TUCR_GBM <- read_csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_DESeq2_allTUCRs_byIDH1st
    sep = ""), col_names = TRUE)

tucr_annot <- read_delim("Inputs/general_files/bedfiles/hg38.ultraconserved.bed",
    col_names = TRUE, delim = "\t")

res_TUCR_LGG_2 <- res_TUCR_LGG %>%
    separate(row, into = c("alias", "kibble"), sep = "___") %>%
    dplyr::select(-kibble) %>%
    left_join(tucr_annot, by = "alias") %>%
    filter(!is.na(chrom)) %>%
    dplyr::select(alias, annot, log2FoldChange, padj) %>%
    mutate(disease = "LGG")
```

```
res_TUCR_GBM_2 <- res_TUCR_GBM %>%
    separate(row, into = c("alias", "kibble"), sep = "___") %>%
    dplyr::select(-kibble) %>%
    left_join(tucr_annot, by = "alias") %>%
    filter(!is.na(chrom)) %>%
    dplyr::select(alias, annot, log2FoldChange, padj) %>%
    mutate(disease = "GBM")

res_TUCR <- rbind(res_TUCR_GBM_2, res_TUCR_LGG_2)

volcanoplot_expression(res_TUCR, genes = "all", output = paste(outputdir, "/Figure5_Supplementary/supple
    sep = ""), height = 3, width = 8, dpi = 600, annot = "")
```

Supplementary Figure 5C

```
disease <- "GBM"

normal <- "cortex"

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)
```

```
    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))
```

```
survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$IDH1status) %in% "WT")
t_index <- which(as.character(metadata$IDH1status) %in% "MUT")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
```

```
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs_byIDH1status.csv", sep = ""))
```

```
disease <- "LGG"

normal <- "cortex"

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()
```

```
    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}
```

```
match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$IDH1status) %in% "WT")
t_index <- which(as.character(metadata$IDH1status) %in% "MUT")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))
```

```
ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs_byIDH1status.csv", sep = ""))
```

Writing a script to generate a volcano plot for survival

```
### Volcano plot

volcanosurv <- function (res,genes = "all",title = "TUCRs correlated with survival in gliomas", output =
  #res <- res_surv
  #genes = c("uc.110","uc.62")
  #title = paste("TUCR correlation with patient survival in ",disease,sep="")
  #output = paste(outputdir,"/intergenic_tucr_results_volcanosurv.png",sep="")
  i <- 1
  vres <- res %>%
    filter(abs(estimate) <=1)
  vres <- vres %>% mutate(gene="",Survival="",color="")
  for (i in 1:length(vres$id)){
  ifelse(is.na(vres$pvalue[i]),vres$pvalue[i] <- 1,vres$pvalue[i] <- vres$pvalue[i])

  ifelse(genes!="all",ifelse(!is.na(match(vres$id[i],genes)),vres$gene[i] <- as.character(vres$id[i]),

  ifelse((vres$p.value[i] <0.05 & vres$estimate[i] < -0),{vres$Survival[i] <- "Significant (Good Prognos
  ifelse((vres$p.value[i] <0.05 & vres$estimate[i] > 0),{vres$Survival[i] <- "Significant (Poor Prognosi
  ifelse({vres$Survival[i] <- "Not significant";vres$color[i] <- "lightgray";}))))}
  #{vres$Survival[i] <- "Not significant";vres$color[i] <- "lightgray";}))}

  vres <- vres %>%
  arrange(desc(Survival))


  ## plot
if(repel==TRUE){p <- ggplot(vres) +
        geom_point(aes(x=estimate, y=-log10(as.numeric(p.value)),col=color)) +
        scale_color_identity(guide = "legend", labels = vres$Survival, breaks = vres$color) +
        ggtitle(title) +
        labs(x = "Cox Estimated Proportional Hazard",
        y = "-log10 P-Value (CH)",
        color = "Legend")+
```

```
        theme(plot.title = element_text(size = rel(2.0), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=10)) +
        geom_text_repel(aes(x=estimate, y=-log10(as.numeric(p.value)),label = gene),force=50) +
        facet_wrap(~disease)
}else{
  p <- ggplot(vres) +
        geom_point(aes(x=estimate, y=-log10(as.numeric(p.value)),col=color)) +
    scale_color_identity(guide = "legend", labels = vres$Survival, breaks = vres$color) +
        + ggtitle(title) +
        labs(x = "Cox Estimated Proportional Hazard",
        y = "-log10 P-Value (CH)",
        color = "Legend") +
        theme(plot.title = element_text(size = rel(2.0), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              strip.background = element_rect(fill="white"),
              axis.title = element_text(size = rel(1.4), face="bold"),
              strip.text = element_text(size = rel(1.4), face="bold"),
              axis.text = element_text(size = rel(1.0)),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=10))}
ggsave(output, width = width, height = height, dpi = dpi) +
        facet_wrap(~disease)
}
```

```
genes <- ""

lgg_surv_TUCR <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/LGG/LGG_survival_coxph_allTUCRs
    sep = ""), header = TRUE) %>%
    mutate(disease = "LGG")

gbm_surv_TUCR <- read.csv(paste(outputdir, "/TUCR_Database/SummaryTables/GBM/GBM_survival_coxph_allTUCRs
    sep = ""), header = TRUE) %>%
    mutate(disease = "GBM")

res_surv <- rbind(gbm_surv_TUCR, lgg_surv_TUCR)

res_surv_sum <- res_surv

res_surv_sum <- res_surv_sum %>%
    mutate(gene = "", Survival = "", color = "")

annot_unique <- as.character(unique(res_surv_sum$annot))

annotation <- "All"

res_surv_sum2 <- res_surv_sum

for (i in 1:length(res_surv_sum2$alias)) {
```

```
    # print(as.character(res_surv_sum2$alias.x[i]))
    ifelse(is.na(res_surv_sum2$pvalue[i]), res_surv_sum2$pvalue[i] <- 1, res_surv_sum2$pvalue[i] <- res

    ifelse(genes != "all", ifelse(!is.na(match(res_surv_sum2$alias[i], genes)), res_surv_sum2$gene[i] <-
        ""), res_surv_sum2$gene[i] <- res_surv_sum2$alias[i])

    ifelse((res_surv_sum2$pvalue[i] < 0.05 & res_surv_sum2$p.value[i] < 0.05 & res_surv_sum2$estimate[i
        0), {
        res_surv_sum2$Survival[i] <- "Significant (Both, Good Prognosis)"
        res_surv_sum2$color[i] <- print(paper_green)
    }, ifelse((res_surv_sum2$pvalue[i] < 0.05 & res_surv_sum2$p.value[i] < 0.05 &
        res_surv_sum2$estimate[i] > 0), {
        res_surv_sum2$Survival[i] <- "Significant (Both, Poor Prognosis)"
        res_surv_sum2$color[i] <- print(paper_red)
    }, {
        res_surv_sum2$Survival[i] <- "Not significant"
        res_surv_sum2$color[i] <- "lightgray"
    }))
}

# res_surv_sum4 <- res_surv_sum2 %>% group_by(Survival) %>%
# dplyr::summarise(count = n(),color) %>% distinct() %>%
# arrange(desc(Survival))

# write.csv(res_surv_sum4,file=paste(outputdir,'/SurvivalAnalysis/SummaryFigures/BarGraphs/',disease,'_

# p <- ggplot(res_surv_sum4, aes(x=Survival,y=count,fill=color)) +
# geom_bar(stat='identity', width = 0.7) + scale_fill_identity(guide =
# 'legend', labels = res_surv_sum2$Survival, breaks = res_surv_sum2$color) +
# labs(y = '# of TUCRs', fill = 'Legend') + theme( plot.title =
# element_text(size=rel(1.5), face='bold',hjust = 0.5), axis.title =
# element_text(size = rel(1.25), face='bold'), panel.grid.major =
# element_blank(), panel.grid.minor = element_blank(), panel.background =
# element_blank(), axis.line = element_line(colour = 'black'), axis.text.x =
# element_blank())

# ggsave(file=paste(outputdir,'/SurvivalAnalysis/SummaryFigures/BarGraphs/',disease,'_',annotation,'_tu
# = print(p), width = 5, height = 2.5, dpi = 600)

res_surv_sum3 <- res_surv_sum

volcanosurv(res_surv_sum3, title = "", genes = "", output = paste(outputdir, "/Figure5_Supplementary/su
    sep = ""))
```

Supplementary Figure 6

```
if (!dir.exists(paste(outputdir, "/Figure6_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure6_Supplementary/", sep = ""))
}
```

Supplementary Figure 6A

Generate FC Plots for each TUCR (GBM)

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 1

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)
```

```
    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))
}

if(!is.na(filterannot)){
  mergedcounts_trimmed <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random")
}

posdata <- mergedcounts_trimmed[,1:8] %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

mergedcounts_trimmed <- mergedcounts_trimmed[,9:length(colnames(mergedcounts_trimmed))]

match_colnames <- match(as.character(colnames(mergedcounts_trimmed)),as.character(metadata$survid))

mergedcounts_trimmed <- as.matrix(mergedcounts_trimmed)
rownames(mergedcounts_trimmed) <- posdata$alias

dds <-
  DESeqDataSetFromMatrix(countData = mergedcounts_trimmed,
                                colData = metadata,
                                design = ~dex)

dds <-
  DESeq(dds)

res <-
  results(dds, tidy=TRUE)

res <-
  as_tibble(res)

colnames(res)[1] <- "alias"

#variablename <- which(as.character(colnames(tablename) %in% as.character(row.names(table2name))))

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")
```

```
vm <- function(x){
  cond <- factor(ifelse(seq(1,dim(x)[2],1) %in% t_index, 1,  0))
  d <- model.matrix(~1+cond)
  x <- t(apply(x,1,as.numeric))
  ex <- voom(x,d,plot=F)
  return(ex$E)
}

count_vm <- vm(mergedcounts_trimmed)

colnames(count_vm) <- colnames(mergedcounts_trimmed)

#if(is.sequential(match_colnames)){
#  print("metadata rows match countfile columns")
#  colnames(count_vm) <- metadata$id
#}else{
#  "metadata rows do not match countfile columns"
#}

scal <- function(x,y){
  median_n <- rowMedians(y)  # mean of normal
  sd_n <- apply(y,1,sd)  # SD of normal
  # z score as (value - mean normal)/SD normal
  res <- matrix(nrow=nrow(x), ncol=ncol(x))
  colnames(res) <- colnames(x)
  rownames(res) <- rownames(x)
  for(i in 1:dim(x)[1]){
    for(j in 1:dim(x)[2]){
      res[i,j] <- (x[i,j]-median_n[i])/sd_n[i]
    }
  }
  return(res)
}

z_rna <- scal(count_vm,count_vm[,n_index])

z_rna2 <- cbind(posdata,z_rna)

write.csv(as.data.frame(z_rna2),file=paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",diseas

fc_counts <- cbind(posdata,z_rna) %>%
  dplyr::select(-chrom,-start,-end,-strand)

fc.dotplot <- fc_counts %>%
  gather(key = "Sample", value = "ZScore",-id,-alias,-tag,-annot) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")
```

```
fc.dotplot <- fc.dotplot %>%
  left_join(res,by="alias") %>%
  mutate(DESeq2 = ifelse(dex=="normal",0,log2FoldChange)) %>%
  mutate(fill = ifelse(DESeq2 >= 1,paper_red,
                ifelse(DESeq2 <= -1,paper_green,paper_gray))) %>%
  dplyr::select(-Sample,-Key,-id.y,-string,-log2FoldChange) %>%
  filter(!is.na(dex))

i <- 1

for(i in 1:length(unique(fc.dotplot$id.x))){

filterdotplot <- as.character(unique(fc.dotplot$id.x[i]))

print(i)

fc.dotplot2 <- fc.dotplot %>%
  filter(id.x == filterdotplot)


print(fc.dotplot2$alias[1])
TUCRname <- fc.dotplot2$alias[1]

if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}

if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_FC_

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
```

```
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_F


if(file.exists(paste(outputdir,"/Figure6_Supplementary/figure6a_supplementary.png",sep = ""))){}else{

fc.dotplot2 <- fc.dotplot %>%
  filter(str_detect(id.x,"uc.2_"))


print("uc.2")
TUCRname <- "uc.2"

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
```

```
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=paste(outputdir,"/Figure6_Supplementary/figure6a_supplementary.png",sep = ""),height=7,w
```

Supplementary Figure 6B

Generate FC Plots for each TUCR (LGG)

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 2

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)
```

```
  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))
}

if(!is.na(filterannot)){
  mergedcounts_trimmed <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random")
}

posdata <- mergedcounts_trimmed[,1:8] %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

mergedcounts_trimmed <- mergedcounts_trimmed[,9:length(colnames(mergedcounts_trimmed))]

match_colnames <- match(as.character(colnames(mergedcounts_trimmed)),as.character(metadata$survid))

mergedcounts_trimmed <- as.matrix(mergedcounts_trimmed)
rownames(mergedcounts_trimmed) <- posdata$alias

dds <-
  DESeqDataSetFromMatrix(countData = mergedcounts_trimmed,
                                colData = metadata,
                                design = ~dex)

dds <-
```

```
  DESeq(dds)

res <-
  results(dds, tidy=TRUE)

res <-
  as_tibble(res)

colnames(res)[1] <- "alias"

#variablename <- which(as.character(colnames(tablename) %in% as.character(row.names(table2name))))

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x){
  cond <- factor(ifelse(seq(1,dim(x)[2],1) %in% t_index, 1,  0))
  d <- model.matrix(~1+cond)
  x <- t(apply(x,1,as.numeric))
  ex <- voom(x,d,plot=F)
  return(ex$E)
}

count_vm <- vm(mergedcounts_trimmed)

colnames(count_vm) <- colnames(mergedcounts_trimmed)

#if(is.sequential(match_colnames)){
#  print("metadata rows match countfile columns")
#  colnames(count_vm) <- metadata$id
#}else{
#  "metadata rows do not match countfile columns"
#}

scal <- function(x,y){
  median_n <- rowMedians(y)  # mean of normal
  sd_n <- apply(y,1,sd)  # SD of normal
  # z score as (value - mean normal)/SD normal
  res <- matrix(nrow=nrow(x), ncol=ncol(x))
  colnames(res) <- colnames(x)
  rownames(res) <- rownames(x)
  for(i in 1:dim(x)[1]){
    for(j in 1:dim(x)[2]){
      res[i,j] <- (x[i,j]-median_n[i])/sd_n[i]
    }
  }
  return(res)
}

z_rna <- scal(count_vm,count_vm[,n_index])

z_rna2 <- cbind(posdata,z_rna)
```

```
write.csv(as.data.frame(z_rna2),file=paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",diseas

fc_counts <- cbind(posdata,z_rna) %>%
  dplyr::select(-chrom,-start,-end,-strand)

fc.dotplot <- fc_counts %>%
  gather(key = "Sample", value = "ZScore",-id,-alias,-tag,-annot) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

fc.dotplot <- fc.dotplot %>%
  left_join(res,by="alias") %>%
  mutate(DESeq2 = ifelse(dex=="normal",0,log2FoldChange)) %>%
  mutate(fill = ifelse(DESeq2 >= 1,paper_red,
                ifelse(DESeq2 <= -1,paper_green,paper_gray))) %>%
  dplyr::select(-Sample,-Key,-id.y,-string,-log2FoldChange) %>%
  filter(!is.na(dex))

i <- 1

for(i in 1:length(unique(fc.dotplot$id.x))){

filterdotplot <- as.character(unique(fc.dotplot$id.x[i]))

print(i)

fc.dotplot2 <- fc.dotplot %>%
  filter(id.x == filterdotplot)


print(fc.dotplot2$alias[1])
TUCRname <- fc.dotplot2$alias[1]

if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}

if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_FC_

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
```

```
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_F


if(file.exists(paste(outputdir,"/Figure6_Supplementary/figure6b_supplementary.png",sep = ""))){}else{

fc.dotplot2 <- fc.dotplot %>%
  filter(str_detect(id.x,"uc.2_"))


print("uc.2")
TUCRname <- "uc.2"

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)
```

```
fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=paste(outputdir,"/Figure6_Supplementary/figure6b_supplementary.png",sep = ""),height=7,w
```

Supplementary Figure 6C

Generate tpm Box Plots for each TUCR (GBM)

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 3

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
```

```
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))
}

#if(makeRPKboxplots == TRUE){

if(!is.na(filterannot)){
  tpmcounts <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random") %>%
```

```
  mutate(length = end.x - start.x)
}else{
tpmcounts <- mergedcounts %>%
  mutate(length = end.x - start.x)}

tpmcounts.info <- tpmcounts %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

tpmcounts <- tpmcounts %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-id,-strand.x,-tag.x,-annot.x,-alias.x)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1000000

tpm <- function(counts,len,dep){
  #x <- tpmcounts/genelength
  #x2 <- t(t(x)/(seqdepth2))
  x <- counts/len
  return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts,genelength,seqdepth2)

tpm.df2 <- cbind(tpmcounts.info,tpm.df)

tpm.median <- as.data.frame(tpm.df)
tpm.median <- tpm.median %>%
  dplyr::summarize(median_tpm = rowMedians(tpm.df)) %>%
  dplyr::mutate(countif = ifelse(median_tpm >=1,1,0),proportion = sum(countif,na.rm = TRUE)/n())
tpm.median <- cbind(tpmcounts.info,tpm.median)
tpm.median <- tpm.median %>% dplyr::select(id,median_tpm,countif,proportion)

proportion.df <- round(as.numeric(tpm.median$proportion[1]),3)*100

write.csv(tpm.df2,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_tpms_allTUCRs.c

tpm.median.write <- tpm.median[,-3]

tpm.median.write <- tpm.median.write[,-3]

tpm.median.write <- tpmcounts.info %>%
  left_join(tpm.median.write,by="id")

write.csv(tpm.median.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_median

tpm.dotplot <- tpm.df2[,9:ncol(tpm.df2)]
TUCRids <- as.character(tpm.df2$id)
annot <- as.character(tpm.df2$annot)
length <- as.character(tpm.df2$length)
tpm.dotplot <- cbind(TUCRids,annot,length,tpm.dotplot)
```

```
tpm.dotplot <- tpm.dotplot %>%
  gather(key = "Sample", value = "TPM",-TUCRids,-annot,-length) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

tpm.dotplot <- tpm.dotplot[complete.cases(tpm.dotplot),]

median(tpm.dotplot$TPM,na.rm=TRUE)

i <- 1
for(i in 1:length(TUCRids)){
#TUCR <- "uc.110"
TUCR <- tpm.df2$id[i]
TUCRname <- as.character(tpm.df2$alias[i])
print(i)
print(TUCRname)
if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}
if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tp
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(TUCRids == TUCR)

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                     "Transcripts per kilobase million (TPM)",
                     "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tpm
```

```
}}

TUCR <- "uc.2"
TUCRname <- "uc.2"
print(i)
print(TUCRname)
if(file.exists(paste(outputdir,"/Figure6_Supplementary/figure6c_supplementary.png",sep = ""))){}else{
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(str_detect(TUCRids,"uc.2_"))

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                      "Transcripts per kilobase million (TPM)",
                      "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/figure6c_supplementary.png",sep = ""),height=7,wi

}
```

Supplementary Figure 6D

Generate tpm Box Plots for each TUCR

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 4

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs
```

```
n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))
}

#if(makeRPKboxplots == TRUE){

if(!is.na(filterannot)){
  tpmcounts <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random") %>%
  mutate(length = end.x - start.x)
}else{
tpmcounts <- mergedcounts %>%
  mutate(length = end.x - start.x)}

tpmcounts.info <- tpmcounts %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

tpmcounts <- tpmcounts %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-id,-strand.x,-tag.x,-annot.x,-alias.x)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1000000

tpm <- function(counts,len,dep){
  #x <- tpmcounts/genelength
  #x2 <- t(t(x)/(seqdepth2))
  x <- counts/len
  return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts,genelength,seqdepth2)

tpm.df2 <- cbind(tpmcounts.info,tpm.df)

tpm.median <- as.data.frame(tpm.df)
tpm.median <- tpm.median %>%
  dplyr::summarize(median_tpm = rowMedians(tpm.df)) %>%
  dplyr::mutate(countif = ifelse(median_tpm >=1,1,0),proportion = sum(countif,na.rm = TRUE)/n())
tpm.median <- cbind(tpmcounts.info,tpm.median)
tpm.median <- tpm.median %>% dplyr::select(id,median_tpm,countif,proportion)

proportion.df <- round(as.numeric(tpm.median$proportion[1]),3)*100

tpm.df.write <- cbind(tpmcounts.info,tpm.df2)

write.csv(tpm.df.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_tpms_allTU

tpm.median.write <- tpm.median[,-3]

tpm.median.write <- tpm.median.write[,-3]
```

```
tpm.median.write <- tpmcounts.info %>%
  left_join(tpm.median.write,by="id")

write.csv(tpm.median.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_median_

tpm.dotplot <- tpm.df2[,9:ncol(tpm.df2)]
TUCRids <- as.character(tpm.df2$id)
annot <- as.character(tpm.df2$annot)
length <- as.character(tpm.df2$length)
tpm.dotplot <- cbind(TUCRids,annot,length,tpm.dotplot)
tpm.dotplot <- tpm.dotplot %>%
  gather(key = "Sample", value = "TPM",-TUCRids,-annot,-length) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

tpm.dotplot <- tpm.dotplot[complete.cases(tpm.dotplot),]

median(tpm.dotplot$TPM,na.rm=TRUE)

i <- 1
for(i in 1:length(TUCRids)){
#TUCR <- "uc.110"
TUCR <- tpm.df2$id[i]
TUCRname <- as.character(tpm.df2$alias[i])
print(i)
print(TUCRname)
if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}
if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tpm
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(TUCRids == TUCR)

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
```

```
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                     "Transcripts per kilobase million (TPM)",
                     "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tpm
}}

TUCR <- "uc.2"
TUCRname <- "uc.2"
print(i)
print(TUCRname)
if(file.exists(paste(outputdir,"/Figure6_Supplementary/figure6d_supplementary.png",sep = ""))){}else{
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(str_detect(TUCRids,"uc.2_"))

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                     "Transcripts per kilobase million (TPM)",
                     "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/figure6d_supplementary.png",sep = ""),height=7,wi

}
```

Supplementary Figure 6E and 6F

```
# Load the WGCNA package
library(WGCNA)
# The following setting is important, do not omit.
options(stringsAsFactors = FALSE)
```

```
# Load the expression and trait data saved in the first part
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-01-dataInput.RData")
# The variable lnames contains the names of loaded variables.
lnames
# Load network data saved in the second part.
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-02-networkConstruction-auto.RData")
lnames
```

filter out modules that are particularly large.

```
# Define numbers of genes and samples
nGenes = ncol(datExpr)
nSamples = nrow(datExpr)
# Recalculate MEs with color labels
MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes

datExpr2 = as.data.frame(datExpr)
MEs = orderMEs(MEs0)
datTraits2 <- as.data.frame(datTraits2)
moduleTraitCor = cor(MEs, datTraits2, use = "p")
moduleTraitPvalue = corPvalueStudent(moduleTraitCor, nSamples)
```

```
geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))
MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

allmodules <- as.data.frame(t(MMPvalue))

allmodules2 <- allmodules %>%
    dplyr::mutate(Module = row.names(allmodules)) %>%
    gather(key = "Gene", value = "pvalue", -Module) %>%
    group_by(Gene) %>%
    mutate(maxmodule = min(pvalue, na.rm = TRUE)) %>%
    filter(pvalue == maxmodule) %>%
    group_by(Module) %>%
    dplyr::summarize(n = n())

allmodules3 <- allmodules2 %>%
    mutate(q1 = quantile(allmodules2$n, 0.25), q3 = quantile(allmodules2$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# repeat{

n_count <- length(allmodules3$n)

allmodules3 <- allmodules3 %>%
    mutate(q1 = quantile(allmodules3$n, 0.25), q3 = quantile(allmodules3$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# if(n_count == length(allmodules3$n)) break;

# }
```

```
allmodules_check <- as.data.frame(as.character(allmodules3$Module)) %>%
    dplyr::mutate(checker = TRUE)

colnames(allmodules_check) <- c("Module", "checker")

boxplot(allmodules3$n, ylab = "n")

moduleTraitCor2 <- as.data.frame(moduleTraitCor) %>%
    mutate(Module = row.names(moduleTraitCor)) %>%
    left_join(allmodules_check, by = "Module") %>%
    filter(checker == TRUE)

datTraits <- datTraits2

geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))

match_modules <- match(as.character(allmodules_check$Module), colnames(geneModuleMembership))

geneModuleMembership2 <- geneModuleMembership[, match_modules]

modNames = substring(names(geneModuleMembership2), 3)

MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

match_modules <- match(as.character(allmodules_check$Module), colnames(MMPvalue))

MMPvalue2 = MMPvalue[, match_modules]

names(geneModuleMembership2) = paste("MM", modNames, sep = "")
names(MMPvalue2) = paste("p.MM", modNames, sep = "")

# Save.image(file='./Inputs/general_files/wgcnafiles/allprelims.Rdata')
```

```
load(file="./Inputs/general_files/wgcnafiles/allprelims.Rdata")

datTraits2 <- datTraits2 %>%
  dplyr::select(-disease)

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
  # Filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

hc.Traits <- hclust(dist(t(datTraits2)))

clust_order_traits <- hc.Traits$order

clust_positions <- as.data.frame(cbind(as.character(colnames(datTraits)[clust_order_traits]),as.numeric

colnames(clust_positions) <- c("trait","clust_order")

geneTraitSignificance <- as.data.frame(cor(datExpr,datTraits2, use = "p")) %>%
```

```
  dplyr::select(-dex2,-p53status2)

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules,-Module,-checker,-p53status2,-dex2) %>%
  # Filter(is.numeric(cor)) %>%
  # Filter(trait == trait_id) %>%
  dplyr::group_by(trait) %>%
  arrange(desc(cor)) %>%
  ungroup() %>%
  dplyr::group_by(modules) %>%
  dplyr::mutate(sumnumber = sum(cor,na.rm=T),
         n = n(),
         weight = sumnumber/n) %>%
  ungroup() %>%
  arrange(desc(weight))

traitpositions <- moduleTraitCor_trait_id %>%
  dplyr::select(modules) %>%
  distinct() %>%
  dplyr::mutate(position = row_number())

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  left_join(traitpositions,by="modules") %>%
  left_join(clust_positions,by="trait")

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  mutate(clust_position_order = as.numeric(moduleTraitCor_trait_id$clust_order))

p <- ggplot(data = moduleTraitCor_trait_id,mapping = aes(x = reorder(modules,position),y = reorder(trai
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("TUCRs") +
  xlab("Modules") +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
```

```
        annotate(
    geom = "point",
    color = c(as.character(moduleTraitCor_trait_id$modules)),
    x = moduleTraitCor_trait_id$position,
    y = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/Figure1/figure1h.png",sep=""), width = 7, height = 3, dpi = 600)
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))
}

#####GO-TERM ANALYSIS

if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")

if (!require('GO.db')) BiocManager::install('GO.db')
library(GO.db)

#if (!require('AnnotationDBI')) BiocManager::install('AnnotationDBI')
#library(AnnotationDBI)

if (!require('org.Hs.eg.db')) BiocManager::install('org.Hs.eg.db')
library(org.Hs.eg.db)

if (!require('limma')) BiocManager::install('limma')
library(limma)

i <- 1

genelist <- geneTraitSignificance %>%
  mutate(genenames = row.names(geneTraitSignificance))

for(i in 1:length(unique(moduleTraitCor_trait_id$modules))){

  print(i)
  print(unique(moduleTraitCor_trait_id$modules)[i])

  module <- unique(moduleTraitCor_trait_id$modules)[i]
  #module <- "yellowgreen"
  column = match(module, modNames);
  moduleGenes = moduleColors==module;
```

```
  genelist2 <- as.data.frame(genelist[moduleGenes,])

  genelist2 <- genelist2 %>%
    separate(genenames,into=c("Alias","kibble"),sep="___")

  symbols <- as.character(genelist2$Alias)

  EntrezIDs <- mapIds(org.Hs.eg.db, symbols, 'ENTREZID', 'SYMBOL')
  allgenes <- cbind(symbols,EntrezIDs)
  allgenes <- allgenes[complete.cases(allgenes),]

  g <- goana(EntrezIDs)

  g_bp <- g %>%
    filter(Ont == "BP")
  topGO_bp <- topGO(g_bp) %>%
    mutate(log10_p = -log10(P.DE),module=module,color="red") %>%
    arrange(desc(log10_p))

  g_mf <- g %>%
    filter(Ont == "MF")
  topGO_mf <- topGO(g_mf) %>%
    mutate(log10_p = -log10(P.DE),module=module,color="blue") %>%
    arrange(desc(log10_p))

  topGO_all <- rbind(topGO_bp,topGO_mf) %>%
                       arrange(as.numeric(log10_p)) %>%
                       dplyr::mutate(roworder = row_number()) %>%
                       mutate(newTerm = substr(Term,1,80))

  p <- ggplot(topGO_all,aes(x=reorder(newTerm,roworder),y=as.numeric(log10_p),fill=module,color=color))
    geom_bar(stat="identity") + coord_flip() + scale_fill_identity() + scale_color_identity() + facet_w
    ggtitle(module) +
    labs(x="Go Term", y = "-log10(p-value)") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold",angle=0,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank(),
        strip.text.x = element_text(size = rel(2.2)))
  p

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",module,"_all_bar.png",sep=

if(module == "#004C54"){
```

```
if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8a.png",sep=""),heigh

if(module == "#FFC0CB"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8b.png",sep=""),heigh

if(module == "#0000FF"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9a.png",sep=""),heigh

if(module == "#008080"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9b.png",sep=""),heigh

}
```

```
####Trait Heatmaps

figureorder <- 7

#moduleTraitCor2 <- moduleTraitCor2 %>%
#  dplyr::select(-disease)

h <- 2

for(h in 2:482){
skip_to_next <<- FALSE
print(h)
trait_id <- colnames(moduleTraitCor2)[h]
#trait_id <-"uc.15"

print(trait_id)

if(!dir.exists(paste(outputdir,"/TUCR_Database/",trait_id,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",trait_id,sep=""))
}

datTraits <- as.data.frame(t(datTraits2)) %>%
```

```
  mutate(rownames = row.names(t(datTraits2))) %>%
  filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

geneTraitSignificance = as.data.frame(cor(datExpr,datTraits, use = "p"))

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  dplyr::mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules) %>%
  dplyr::filter(trait == trait_id) %>%
  dplyr::arrange(cor) %>%
  dplyr::mutate(position = row_number())

traitmodules <- geneModuleMembership2 %>%
  dplyr::mutate(rownamer = row.names(geneModuleMembership2)) %>%
  separate(rownamer,into=c("rownamer","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  dplyr::filter(rownamer==trait_id)

traitmodules <- t(traitmodules)

traitmodules <- as.data.frame(cbind(row.names(traitmodules),traitmodules))

colnames(traitmodules) <- c("Module","ModuleMembership")

traitpvalues <- MMPvalue2 %>%
  dplyr::mutate(rownamer = row.names(MMPvalue2)) %>%
  separate(rownamer,into=c("rownamer","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  dplyr::filter(rownamer==trait_id)

traitpvalues <- t(traitpvalues)

traitpvalues <- as.data.frame(cbind(paste("MM",str_remove(row.names(traitpvalues),"p.MM"),sep=""),trait

colnames(traitpvalues) <- c("Module","MMpvalue")
traitmodules <- traitmodules %>%
  left_join(allmodules_check, by= "Module") %>%
  left_join(traitpvalues,"Module") %>%
  # Filter(checker == TRUE) %>%
  dplyr::mutate(moduleColors = str_remove(Module,"MM")) %>%
  dplyr::filter(moduleColors != "rownamer")

df_correlations <- data.frame(matrix(ncol=2,nrow=0, dimnames=list(NULL, c("module","cor"))))

i <- 1
```

```
for(i in 1:length(unique(traitmodules$moduleColors))){

module = as.character(unique(traitmodules$moduleColors)[i])
#module = "red"
column = match(module, modNames)
moduleGenes = traitmodules$moduleColors==module;
correlation <- cor(abs(geneModuleMembership[moduleGenes, column]),
                   abs(geneTraitSignificance[moduleGenes, 1]))

rbinder <- c(module,correlation)
df_correlations <- rbind(df_correlations,rbinder)


}

colnames(df_correlations) <- c("module","cor")

df_correlations <- as.data.frame(df_correlations) %>%
  mutate(Module = paste("MM",module,sep=""))

traitmodules2 <- traitmodules %>%
  left_join(df_correlations,by = "Module") %>%
  dplyr::select(module,cor,ModuleMembership,MMpvalue) %>%
  dplyr::mutate(RankModule = percent_rank(1-as.numeric(ModuleMembership)),Rankcor = percent_rank(cor),M
  dplyr::group_by(module) %>%
  dplyr::mutate(totalscore = sum(as.numeric(RankModule),as.numeric(Rankcor),na.rm=TRUE)) %>%
  dplyr::ungroup() %>%
  dplyr::arrange(as.numeric(cor)) %>%
  dplyr::mutate(position = row_number(),cor = ifelse(MMpvalue2 >= 0.05,NA,cor))


p <- ggplot(data = traitmodules2,mapping = aes(x = as.character(trait_id),y = reorder(module,position),
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("modules") +
  xlab(trait_id) +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(traitmodules2$module)),
```

```
    y = traitmodules2$position,
    x = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/TUCR_Database/",trait_id,"/",figureorder,"_",trait_id,"_wgcna_modulecor

if(!dir.exists(paste(outputdir,"/Figure6_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure6_Supplementary/",sep=""))
}

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure7_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure7_Supplementary/",sep=""))
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure2/",sep=""))){
  dir.create(paste(outputdir,"/Figure2/",sep=""))
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2f.png",sep=""), width = 3, height = 7, dpi = 600)
}

####Trait Correlation Plots

traitmodules3 <- traitmodules2 %>%
  filter(as.numeric(MMpvalue) <= 0.05)



i <- 1

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));
```

```
xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_6_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_6.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_6.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_6.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- 2

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));
```

```
xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_5_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_5.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_5.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_5.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- 3

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));
```

```
xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_4_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_4.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_4.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_4.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));
```

```
xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_1_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_1.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_1.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_1.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)-1

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));
```

```
xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_2_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_2.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_2.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_2.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)-2

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));
```

```
xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_3_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_3.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_3.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_3.png",sep=""), width = 5, height = 5, dpi = 600)
}


}
```

Supplementary Figure 6G

Completing survival analysis for TUCRs

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 6
```

```
## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)
```

```
    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
```

```
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {
```

```
km_countdata <- z_rna
colnames(km_countdata) <- metadata$id[t_index]
posdata$median <- rowMedians(survcounts)

# km_countdata2 <- km_countdata[,complete.cases(clinical2)]

probs <- c(0.25, 0.5, 0.75)
q <- rowQuantiles(km_countdata, probs = probs)
posdata$n25 <- q[, 1]
posdata$n75 <- q[, 3]
posdata <- posdata %>%
    dplyr::select(TUCR = alias.x, median, n25, n75)
km_TUCRs <- cbind(posdata, km_countdata)

km_TUCRs <- km_TUCRs %>%
    gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
    mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
    distinct %>%
    dplyr::select(TUCR, median, id, group) %>%
    left_join(clinical2, by = "id") %>%
    dplyr::filter(median != 0)

TUCRids <- as.character(posdata$TUCR)
i <- 1

ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
    "pvalue", "method"))))

for (i in 1:length(TUCRids)) {
    print(i)
    print(TUCRids[i])

    skip_to_next <- FALSE
    TUCRsurv <- as.character(TUCRids[i])
    if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
        dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
    }

    # TUCR <- 'uc.1'
    kmdata <- km_TUCRs %>%
        dplyr::filter(TUCR == TUCRsurv) %>%
        dplyr::select(TUCR, id, group, time, status)
    fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
        skip_to_next <<- TRUE
    })
    p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
        risk.table = TRUE), error = function(e) {
        skip_to_next <<- TRUE
    })
    # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
    # TRUE,risk.table = TRUE)

    if (skip_to_next == TRUE) {
```

```
            rbinder <- cbind(as.character(TUCRsurv), NA, NA)
            ptable[i, ] <- rbinder
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}

TUCRsurv <- "uc.2"
kmdata <- km_TUCRs %>%
    dplyr::filter(TUCR == TUCRsurv) %>%
    dplyr::select(TUCR, id, group, time, status)
fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
    skip_to_next <<- TRUE
})
p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE, risk.table = TRUE),
    error = function(e) {
        skip_to_next <<- TRUE
    })
# p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval = TRUE,risk.table =
# TRUE)

if (skip_to_next == TRUE) {
    rbinder <- cbind(as.character(TUCRsurv), NA, NA)
    ptable[i, ] <- rbinder
} else {
    grid.draw.ggsurvplot <- function(x) {
        survminer:::print.ggsurvplot(x, newpage = FALSE)
    }
    ggsave(file = paste(outputdir, "/Figure6_Supplementary/figure6g_supplementary.png",
        sep = ""), device = "png", plot = p)
}
```

Supplementary Figure 6H

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 6
```

```r
## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)

    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)
```

```
    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
```

```
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {
```

```
km_countdata <- z_rna
colnames(km_countdata) <- metadata$id[t_index]
posdata$median <- rowMedians(survcounts)

# km_countdata2 <- km_countdata[,complete.cases(clinical2)]

probs <- c(0.25, 0.5, 0.75)
q <- rowQuantiles(km_countdata, probs = probs)
posdata$n25 <- q[, 1]
posdata$n75 <- q[, 3]
posdata <- posdata %>%
    dplyr::select(TUCR = alias.x, median, n25, n75)
km_TUCRs <- cbind(posdata, km_countdata)

km_TUCRs <- km_TUCRs %>%
    gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
    mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
    distinct %>%
    dplyr::select(TUCR, median, id, group) %>%
    left_join(clinical2, by = "id") %>%
    dplyr::filter(median != 0)

TUCRids <- as.character(posdata$TUCR)
i <- 1

ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
    "pvalue", "method"))))

for (i in 1:length(TUCRids)) {
    print(i)
    print(TUCRids[i])

    skip_to_next <- FALSE
    TUCRsurv <- as.character(TUCRids[i])
    if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
        dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
    }

    # TUCR <- 'uc.1'
    kmdata <- km_TUCRs %>%
        dplyr::filter(TUCR == TUCRsurv) %>%
        dplyr::select(TUCR, id, group, time, status)
    fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
        skip_to_next <<- TRUE
    })
    p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
        risk.table = TRUE), error = function(e) {
        skip_to_next <<- TRUE
    })
    # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
    # TRUE,risk.table = TRUE)

    if (skip_to_next == TRUE) {
```

```
            rbinder <- cbind(as.character(TUCRsurv), NA, NA)
            ptable[i, ] <- rbinder
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}

TUCRsurv <- "uc.2"
kmdata <- km_TUCRs %>%
    dplyr::filter(TUCR == TUCRsurv) %>%
    dplyr::select(TUCR, id, group, time, status)
fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
    skip_to_next <<- TRUE
})
p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE, risk.table = TRUE),
    error = function(e) {
        skip_to_next <<- TRUE
    })
# p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval = TRUE,risk.table =
# TRUE)

if (skip_to_next == TRUE) {
    rbinder <- cbind(as.character(TUCRsurv), NA, NA)
    ptable[i, ] <- rbinder
} else {
    grid.draw.ggsurvplot <- function(x) {
        survminer:::print.ggsurvplot(x, newpage = FALSE)
    }
    ggsave(file = paste(outputdir, "/Figure6_Supplementary/figure6h_supplementary.png",
        sep = ""), device = "png", plot = p)
}
```

Supplementary Figure 7

```
if (!dir.exists(paste(outputdir, "/Figure7_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure7_Supplementary/", sep = ""))
}
```

Supplementary Figure 7A

Generate FC Plots for each TUCR (GBM)

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 1

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)
```

```
}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))
}

if(!is.na(filterannot)){
  mergedcounts_trimmed <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random")
}

posdata <- mergedcounts_trimmed[,1:8] %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

mergedcounts_trimmed <- mergedcounts_trimmed[,9:length(colnames(mergedcounts_trimmed))]

match_colnames <- match(as.character(colnames(mergedcounts_trimmed)),as.character(metadata$survid))

mergedcounts_trimmed <- as.matrix(mergedcounts_trimmed)
rownames(mergedcounts_trimmed) <- posdata$alias

dds <-
  DESeqDataSetFromMatrix(countData = mergedcounts_trimmed,
                                colData = metadata,
                                design = ~dex)

dds <-
  DESeq(dds)

res <-
  results(dds, tidy=TRUE)

res <-
  as_tibble(res)

colnames(res)[1] <- "alias"

#variablename <- which(as.character(colnames(tablename) %in% as.character(row.names(table2name))))
```

```
n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x){
  cond <- factor(ifelse(seq(1,dim(x)[2],1) %in% t_index, 1,  0))
  d <- model.matrix(~1+cond)
  x <- t(apply(x,1,as.numeric))
  ex <- voom(x,d,plot=F)
  return(ex$E)
}

count_vm <- vm(mergedcounts_trimmed)

colnames(count_vm) <- colnames(mergedcounts_trimmed)

#if(is.sequential(match_colnames)){
#  print("metadata rows match countfile columns")
#  colnames(count_vm) <- metadata$id
#}else{
#  "metadata rows do not match countfile columns"
#}

scal <- function(x,y){
  median_n <- rowMedians(y)  # mean of normal
  sd_n <- apply(y,1,sd)  # SD of normal
  # z score as (value - mean normal)/SD normal
  res <- matrix(nrow=nrow(x), ncol=ncol(x))
  colnames(res) <- colnames(x)
  rownames(res) <- rownames(x)
  for(i in 1:dim(x)[1]){
    for(j in 1:dim(x)[2]){
      res[i,j] <- (x[i,j]-median_n[i])/sd_n[i]
    }
  }
  return(res)
}

z_rna <- scal(count_vm,count_vm[,n_index])

z_rna2 <- cbind(posdata,z_rna)

write.csv(as.data.frame(z_rna2),file=paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",diseas

fc_counts <- cbind(posdata,z_rna) %>%
  dplyr::select(-chrom,-start,-end,-strand)

fc.dotplot <- fc_counts %>%
  gather(key = "Sample", value = "ZScore",-id,-alias,-tag,-annot) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
```

```
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

fc.dotplot <- fc.dotplot %>%
  left_join(res,by="alias") %>%
  mutate(DESeq2 = ifelse(dex=="normal",0,log2FoldChange)) %>%
  mutate(fill = ifelse(DESeq2 >= 1,paper_red,
                ifelse(DESeq2 <= -1,paper_green,paper_gray))) %>%
  dplyr::select(-Sample,-Key,-id.y,-string,-log2FoldChange) %>%
  filter(!is.na(dex))

i <- 1

for(i in 1:length(unique(fc.dotplot$id.x))){

filterdotplot <- as.character(unique(fc.dotplot$id.x[i]))

print(i)

fc.dotplot2 <- fc.dotplot %>%
  filter(id.x == filterdotplot)


print(fc.dotplot2$alias[1])
TUCRname <- fc.dotplot2$alias[1]

if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}

if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_FC_

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
```

```
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_F

if(file.exists(paste(outputdir,"/Figure7_Supplementary/figure7a_supplementary.png",sep = ""))){}else{

fc.dotplot2 <- fc.dotplot %>%
  filter(str_detect(id.x,"uc.15_"))


print("uc.15")
TUCRname <- "uc.15"

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
```

```
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=paste(outputdir,"/Figure7_Supplementary/figure7a_supplementary.png",sep = ""),height=7,w
```

Supplementary Figure 7B

Generate FC Plots for each TUCR (LGG)

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 2

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
```

```
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,sep=""))
}

if(!is.na(filterannot)){
  mergedcounts_trimmed <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random")
}

posdata <- mergedcounts_trimmed[,1:8] %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

mergedcounts_trimmed <- mergedcounts_trimmed[,9:length(colnames(mergedcounts_trimmed))]

match_colnames <- match(as.character(colnames(mergedcounts_trimmed)),as.character(metadata$survid))

mergedcounts_trimmed <- as.matrix(mergedcounts_trimmed)
rownames(mergedcounts_trimmed) <- posdata$alias

dds <-
  DESeqDataSetFromMatrix(countData = mergedcounts_trimmed,
```

```
                                colData = metadata,
                                design = ~dex)

dds <-
  DESeq(dds)

res <-
  results(dds, tidy=TRUE)

res <-
  as_tibble(res)

colnames(res)[1] <- "alias"

#variablename <- which(as.character(colnames(tablename) %in% as.character(row.names(table2name))))

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x){
  cond <- factor(ifelse(seq(1,dim(x)[2],1) %in% t_index, 1,  0))
  d <- model.matrix(~1+cond)
  x <- t(apply(x,1,as.numeric))
  ex <- voom(x,d,plot=F)
  return(ex$E)
}

count_vm <- vm(mergedcounts_trimmed)

colnames(count_vm) <- colnames(mergedcounts_trimmed)

#if(is.sequential(match_colnames)){
#  print("metadata rows match countfile columns")
#  colnames(count_vm) <- metadata$id
#}else{
#  "metadata rows do not match countfile columns"
#}

scal <- function(x,y){
  median_n <- rowMedians(y)  # mean of normal
  sd_n <- apply(y,1,sd)  # SD of normal
  # z score as (value - mean normal)/SD normal
  res <- matrix(nrow=nrow(x), ncol=ncol(x))
  colnames(res) <- colnames(x)
  rownames(res) <- rownames(x)
  for(i in 1:dim(x)[1]){
    for(j in 1:dim(x)[2]){
      res[i,j] <- (x[i,j]-median_n[i])/sd_n[i]
    }
  }
  return(res)
}
```

```
z_rna <- scal(count_vm,count_vm[,n_index])

z_rna2 <- cbind(posdata,z_rna)

write.csv(as.data.frame(z_rna2),file=paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",diseas

fc_counts <- cbind(posdata,z_rna) %>%
  dplyr::select(-chrom,-start,-end,-strand)

fc.dotplot <- fc_counts %>%
  gather(key = "Sample", value = "ZScore",-id,-alias,-tag,-annot) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

fc.dotplot <- fc.dotplot %>%
  left_join(res,by="alias") %>%
  mutate(DESeq2 = ifelse(dex=="normal",0,log2FoldChange)) %>%
  mutate(fill = ifelse(DESeq2 >= 1,paper_red,
                ifelse(DESeq2 <= -1,paper_green,paper_gray))) %>%
  dplyr::select(-Sample,-Key,-id.y,-string,-log2FoldChange) %>%
  filter(!is.na(dex))

i <- 1

for(i in 1:length(unique(fc.dotplot$id.x))){

filterdotplot <- as.character(unique(fc.dotplot$id.x[i]))

print(i)

fc.dotplot2 <- fc.dotplot %>%
  filter(id.x == filterdotplot)


print(fc.dotplot2$alias[1])
TUCRname <- fc.dotplot2$alias[1]

if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}

if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_FC

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)
```

```
padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_F


if(file.exists(paste(outputdir,"/Figure7_Supplementary/figure7b_supplementary.png",sep = ""))){}else{

fc.dotplot2 <- fc.dotplot %>%
  filter(str_detect(id.x,"uc.15_"))


print("uc.15")
TUCRname <- "uc.15"

padj.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
  dplyr::select(padj)

padj.DESeq2 <- formatC(padj.DESeq2[[1]][1], format = "e", digits = 2)

foldchange.DESeq2 <- fc.dotplot2 %>%
  filter(dex == "tumor") %>%
```

```
  dplyr::select(DESeq2)

foldchange.DESeq2 <- round(foldchange.DESeq2[1,1],2)

fc.dotplot3 <- fc.dotplot2 %>%
  dplyr::select(dex,DESeq2,fill) %>%
  distinct()

p2 <- ggplot(data = fc.dotplot2,aes(x=dex, y=ZScore,fill=fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = fc.dotplot2, aes(x=dex, y=ZScore),binaxis='y', stackdir='center', stackratio=0.90,
  ggtitle(paste0(disease,"\n","FC = ",round(foldchange.DESeq2,2),"\n","FDR = ",padj.DESeq2)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_text(size = rel(1.6), hjust = 0.5, face="bold",margin=margin(0,0,0,0))) +
  labs(y="Z-Score", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p2,file=paste(outputdir,"/Figure7_Supplementary/figure7b_supplementary.png",sep = ""),height=7,w
```

Supplementart Figure 7C

Generate tpm Box Plots for each TUCR (GBM)

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 3

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",s

n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs
```

```
## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))
}

#if(makeRPKboxplots == TRUE){
```

```
if(!is.na(filterannot)){
  tpmcounts <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random") %>%
  mutate(length = end.x - start.x)
}else{
tpmcounts <- mergedcounts %>%
  mutate(length = end.x - start.x)}

tpmcounts.info <- tpmcounts %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

tpmcounts <- tpmcounts %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-id,-strand.x,-tag.x,-annot.x,-alias.x)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1000000

tpm <- function(counts,len,dep){
  #x <- tpmcounts/genelength
  #x2 <- t(t(x)/(seqdepth2))
  x <- counts/len
  return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts,genelength,seqdepth2)

tpm.df2 <- cbind(tpmcounts.info,tpm.df)

tpm.median <- as.data.frame(tpm.df)
tpm.median <- tpm.median %>%
  dplyr::summarize(median_tpm = rowMedians(tpm.df)) %>%
  dplyr::mutate(countif = ifelse(median_tpm >=1,1,0),proportion = sum(countif,na.rm = TRUE)/n())
tpm.median <- cbind(tpmcounts.info,tpm.median)
tpm.median <- tpm.median %>% dplyr::select(id,median_tpm,countif,proportion)

proportion.df <- round(as.numeric(tpm.median$proportion[1]),3)*100

write.csv(tpm.df2,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_tpms_allTUCRs.cs

tpm.median.write <- tpm.median[,-3]

tpm.median.write <- tpm.median.write[,-3]

tpm.median.write <- tpmcounts.info %>%
  left_join(tpm.median.write,by="id")

write.csv(tpm.median.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_median_

tpm.dotplot <- tpm.df2[,9:ncol(tpm.df2)]
TUCRids <- as.character(tpm.df2$id)
```

```
annot <- as.character(tpm.df2$annot)
length <- as.character(tpm.df2$length)
tpm.dotplot <- cbind(TUCRids,annot,length,tpm.dotplot)
tpm.dotplot <- tpm.dotplot %>%
  gather(key = "Sample", value = "TPM",-TUCRids,-annot,-length) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

tpm.dotplot <- tpm.dotplot[complete.cases(tpm.dotplot),]

median(tpm.dotplot$TPM,na.rm=TRUE)

i <- 1
for(i in 1:length(TUCRids)){
#TUCR <- "uc.110"
TUCR <- tpm.df2$id[i]
TUCRname <- as.character(tpm.df2$alias[i])
print(i)
print(TUCRname)
if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}
if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tp
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(TUCRids == TUCR)

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                      "Transcripts per kilobase million (TPM)",
                      "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
```

```
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tpm
}}

TUCR <- "uc.15"
TUCRname <- "uc.15"
print(i)
print(TUCRname)
if(file.exists(paste(outputdir,"/Figure7_Supplementary/figure7c_supplementary.png",sep = ""))){}else{
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(str_detect(TUCRids,"uc.15_"))

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                     "Transcripts per kilobase million (TPM)",
                     "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/figure7c_supplementary.png",sep = ""),height=7,wi

}
```

Supplementary Figure 7D

Generate tpm Box Plots for each TUCR

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 4

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_mergedcounts.txt",
```

```
n_countfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_mergedcounts.txt",sep

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_tcga_metadata.cs

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_gtex_metadata.csv

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",disease,"/",disease,"_seqdepth_counts

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/",normal,"/",normal,"_seqdepth_counts.cs

## Merge Data

if(!is.na(n_countfile)){
  mergedcounts <- read.table(t_countfile,header = TRUE)

  normalcounts <- read.table(n_countfile,header = TRUE)

  mergedcounts <- mergedcounts %>% left_join(normalcounts,by=c("id")) %>%
    dplyr::select(-chrom.y,-start.y,-end.y,-strand.y,-tag.y,-annot.y,-alias.y) %>%
    distinct()

  metadata <-
  read_csv(file = t_metadatafile)

  n_metadata <-
  read_csv(file = n_metadatafile)

  rm(normalcounts)

  metadata <- rbind(metadata,n_metadata)

  rm(n_metadata)

  seqdepth <- read.csv(file = t_seqdepthfile)

  n_seqdepth <- read.csv(file = n_seqdepthfile)

  seqdepth <- rbind(seqdepth,n_seqdepth)

  rm(n_seqdepth)

}else{
    mergedcounts <- read.table(t_countfile,header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",sep=""))
}

#if(makeRPKboxplots == TRUE){

if(!is.na(filterannot)){
  tpmcounts <- mergedcounts %>%
  filter(tag.x == filterannot & annot.x != "random") %>%
  mutate(length = end.x - start.x)
}else{
tpmcounts <- mergedcounts %>%
  mutate(length = end.x - start.x)}

tpmcounts.info <- tpmcounts %>%
  dplyr::select("chrom" = chrom.x,"start" = start.x,"end" = end.x,"strand"=strand.x,id,"alias"=alias.x,

tpmcounts <- tpmcounts %>%
  dplyr::select(-chrom.x,-start.x,-end.x,-id,-strand.x,-tag.x,-annot.x,-alias.x)

rownames(tpmcounts) <- as.character(tpmcounts.info$id)

genelength <- tpmcounts.info$length/1000

seqdepth2 <- as.vector(seqdepth$counts)/1000000

tpm <- function(counts,len,dep){
  #x <- tpmcounts/genelength
  #x2 <- t(t(x)/(seqdepth2))
  x <- counts/len
  return(t(t(x)/(dep)))
}

tpm.df <- tpm(tpmcounts,genelength,seqdepth2)

tpm.df2 <- cbind(tpmcounts.info,tpm.df)

tpm.median <- as.data.frame(tpm.df)
tpm.median <- tpm.median %>%
  dplyr::summarize(median_tpm = rowMedians(tpm.df)) %>%
  dplyr::mutate(countif = ifelse(median_tpm >=1,1,0),proportion = sum(countif,na.rm = TRUE)/n())
tpm.median <- cbind(tpmcounts.info,tpm.median)
tpm.median <- tpm.median %>% dplyr::select(id,median_tpm,countif,proportion)

proportion.df <- round(as.numeric(tpm.median$proportion[1]),3)*100

tpm.df.write <- cbind(tpmcounts.info,tpm.df2)

write.csv(tpm.df.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_tpms_allTU
```

```
tpm.median.write <- tpm.median[,-3]

tpm.median.write <- tpm.median.write[,-3]

tpm.median.write <- tpmcounts.info %>%
  left_join(tpm.median.write,by="id")

write.csv(tpm.median.write,paste(outputdir,"/TUCR_Database/SummaryTables/",disease,"/",disease,"_median

tpm.dotplot <- tpm.df2[,9:ncol(tpm.df2)]
TUCRids <- as.character(tpm.df2$id)
annot <- as.character(tpm.df2$annot)
length <- as.character(tpm.df2$length)
tpm.dotplot <- cbind(TUCRids,annot,length,tpm.dotplot)
tpm.dotplot <- tpm.dotplot %>%
  gather(key = "Sample", value = "TPM",-TUCRids,-annot,-length) %>%
  mutate(Key = gsub("[.]","-",Sample)) %>%
    mutate(string = ifelse(str_detect(Key,"TCGA"),
                           sub(".*TCGA", "", Key),
                           sub(".*GTEX", "", Key)),
           barcode = ifelse(str_detect(Key,"TCGA"),
                            tolower(paste("TCGA",substr(string, 1, 8),sep="")),
                            tolower(paste("GTEX",substr(string, 1, 20),sep="")))) %>%
  left_join(metadata,by="barcode")

tpm.dotplot <- tpm.dotplot[complete.cases(tpm.dotplot),]

median(tpm.dotplot$TPM,na.rm=TRUE)

i <- 1
for(i in 1:length(TUCRids)){
#TUCR <- "uc.110"
TUCR <- tpm.df2$id[i]
TUCRname <- as.character(tpm.df2$alias[i])
print(i)
print(TUCRname)
if(!dir.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",TUCRname,"/",sep=""))
}
if(file.exists(paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tp
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(TUCRids == TUCR)

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
```

```
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                     "Transcripts per kilobase million (TPM)",
                     "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/TUCR_Database/",TUCRname,"/",figureorder,"_",TUCRname,"_",disease,"_tpm
}}

TUCR <- "uc.15"
TUCRname <- "uc.15"
print(i)
print(TUCRname)
if(file.exists(paste(outputdir,"/Figure7_Supplementary/figure7d_supplementary.png",sep = ""))){}else{
tpm.dotplot2 <- tpm.dotplot %>% dplyr::filter(str_detect(TUCRids,"uc.15_"))

p<- ggplot(tpm.dotplot2, aes(x=dex, y=TPM)) +
  geom_boxplot(outlier.color="white") +
  #geom_violin(trim = FALSE)+
  #geom_point(pch=21, size=1.5) +
  geom_jitter(binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.30,fill="gray",width=0.2) +
  #ggtitle(paste(tpmethod," expression of ",TUCR)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.6), face="bold"),
        axis.text.y = element_text(size = rel(2.2)),
        axis.text.x = element_text(size = rel(2.2),angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
        labs(y=paste(ifelse((tpmethod=="tpm"|tpmethod=="TPM"),
                     "Transcripts per kilobase million (TPM)",
                     "Reads per kilobase million (tpm)")),
        x = "Sample",title=disease) +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/figure7d_supplementary.png",sep = ""),height=7,wi

}
```

Supplementary Figure 7E and 7F

```
# Load the WGCNA package
library(WGCNA)
# The following setting is important, do not omit.
options(stringsAsFactors = FALSE)
# Load the expression and trait data saved in the first part
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-01-dataInput.RData")
# The variable lnames contains the names of loaded variables.
lnames
# Load network data saved in the second part.
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-02-networkConstruction-auto.RData")
lnames
```

filter out modules that are particularly large.

```
# Define numbers of genes and samples
nGenes = ncol(datExpr)
nSamples = nrow(datExpr)
# Recalculate MEs with color labels
MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes

datExpr2 = as.data.frame(datExpr)
MEs = orderMEs(MEs0)
datTraits2 <- as.data.frame(datTraits2)
moduleTraitCor = cor(MEs, datTraits2, use = "p")
moduleTraitPvalue = corPvalueStudent(moduleTraitCor, nSamples)
```

```
geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))
MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

allmodules <- as.data.frame(t(MMPvalue))

allmodules2 <- allmodules %>%
    dplyr::mutate(Module = row.names(allmodules)) %>%
    gather(key = "Gene", value = "pvalue", -Module) %>%
    group_by(Gene) %>%
    mutate(maxmodule = min(pvalue, na.rm = TRUE)) %>%
    filter(pvalue == maxmodule) %>%
    group_by(Module) %>%
    dplyr::summarize(n = n())

allmodules3 <- allmodules2 %>%
    mutate(q1 = quantile(allmodules2$n, 0.25), q3 = quantile(allmodules2$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# repeat{

n_count <- length(allmodules3$n)

allmodules3 <- allmodules3 %>%
    mutate(q1 = quantile(allmodules3$n, 0.25), q3 = quantile(allmodules3$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
```

```
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# if(n_count == length(allmodules3$n)) break;

# }

allmodules_check <- as.data.frame(as.character(allmodules3$Module)) %>%
    dplyr::mutate(checker = TRUE)

colnames(allmodules_check) <- c("Module", "checker")

boxplot(allmodules3$n, ylab = "n")

moduleTraitCor2 <- as.data.frame(moduleTraitCor) %>%
    mutate(Module = row.names(moduleTraitCor)) %>%
    left_join(allmodules_check, by = "Module") %>%
    filter(checker == TRUE)

datTraits <- datTraits2

geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))

match_modules <- match(as.character(allmodules_check$Module), colnames(geneModuleMembership))

geneModuleMembership2 <- geneModuleMembership[, match_modules]

modNames = substring(names(geneModuleMembership2), 3)

MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

match_modules <- match(as.character(allmodules_check$Module), colnames(MMPvalue))

MMPvalue2 = MMPvalue[, match_modules]

names(geneModuleMembership2) = paste("MM", modNames, sep = "")
names(MMPvalue2) = paste("p.MM", modNames, sep = "")

# Save.image(file='./Inputs/general_files/wgcnafiles/allprelims.Rdata')
```

```
load(file="./Inputs/general_files/wgcnafiles/allprelims.Rdata")

datTraits2 <- datTraits2 %>%
  dplyr::select(-disease)

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
  # Filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

hc.Traits <- hclust(dist(t(datTraits2)))

clust_order_traits <- hc.Traits$order
```

```
clust_positions <- as.data.frame(cbind(as.character(colnames(datTraits)[clust_order_traits]),as.numeric

colnames(clust_positions) <- c("trait","clust_order")

geneTraitSignificance <- as.data.frame(cor(datExpr,datTraits2, use = "p")) %>%
  dplyr::select(-dex2,-p53status2)

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules,-Module,-checker,-p53status2,-dex2) %>%
  # Filter(is.numeric(cor)) %>%
  # Filter(trait == trait_id) %>%
  dplyr::group_by(trait) %>%
  arrange(desc(cor)) %>%
  ungroup() %>%
  dplyr::group_by(modules) %>%
  dplyr::mutate(sumnumber = sum(cor,na.rm=T),
         n = n(),
         weight = sumnumber/n) %>%
  ungroup() %>%
  arrange(desc(weight))

traitpositions <- moduleTraitCor_trait_id %>%
  dplyr::select(modules) %>%
  distinct() %>%
  dplyr::mutate(position = row_number())

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  left_join(traitpositions,by="modules") %>%
  left_join(clust_positions,by="trait")

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  mutate(clust_position_order = as.numeric(moduleTraitCor_trait_id$clust_order))

p <- ggplot(data = moduleTraitCor_trait_id,mapping = aes(x = reorder(modules,position),y = reorder(trai
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("TUCRs") +
  xlab("Modules") +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
```

```
            #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(moduleTraitCor_trait_id$modules)),
    x = moduleTraitCor_trait_id$position,
    y = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/Figure1/figure1h.png",sep=""), width = 7, height = 3, dpi = 600)
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))
}

#####GO-TERM ANALYSIS

if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")

if (!require('GO.db')) BiocManager::install('GO.db')
library(GO.db)

#if (!require('AnnotationDBI')) BiocManager::install('AnnotationDBI')
#library(AnnotationDBI)

if (!require('org.Hs.eg.db')) BiocManager::install('org.Hs.eg.db')
library(org.Hs.eg.db)

if (!require('limma')) BiocManager::install('limma')
library(limma)

i <- 1

genelist <- geneTraitSignificance %>%
  mutate(genenames = row.names(geneTraitSignificance))

for(i in 1:length(unique(moduleTraitCor_trait_id$modules))){

  print(i)
  print(unique(moduleTraitCor_trait_id$modules)[i])
```

```
module <- unique(moduleTraitCor_trait_id$modules)[i]
#module <- "yellowgreen"
column = match(module, modNames);
moduleGenes = moduleColors==module;

genelist2 <- as.data.frame(genelist[moduleGenes,])

genelist2 <- genelist2 %>%
  separate(genenames,into=c("Alias","kibble"),sep="___")

symbols <- as.character(genelist2$Alias)

EntrezIDs <- mapIds(org.Hs.eg.db, symbols, 'ENTREZID', 'SYMBOL')
allgenes <- cbind(symbols,EntrezIDs)
allgenes <- allgenes[complete.cases(allgenes),]

g <- goana(EntrezIDs)

g_bp <- g %>%
  filter(Ont == "BP")
topGO_bp <- topGO(g_bp) %>%
  mutate(log10_p = -log10(P.DE),module=module,color="red") %>%
  arrange(desc(log10_p))

g_mf <- g %>%
  filter(Ont == "MF")
topGO_mf <- topGO(g_mf) %>%
  mutate(log10_p = -log10(P.DE),module=module,color="blue") %>%
  arrange(desc(log10_p))

topGO_all <- rbind(topGO_bp,topGO_mf) %>%
                     arrange(as.numeric(log10_p)) %>%
                     dplyr::mutate(roworder = row_number()) %>%
                     mutate(newTerm = substr(Term,1,80))

p <- ggplot(topGO_all,aes(x=reorder(newTerm,roworder),y=as.numeric(log10_p),fill=module,color=color))
  geom_bar(stat="identity") + coord_flip() + scale_fill_identity() + scale_color_identity() + facet_w
  ggtitle(module) +
  labs(x="Go Term", y = "-log10(p-value)") +
theme(panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      panel.background = element_blank(),
      axis.line = element_line(colour = "black"),
      plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
      axis.title = element_text(size = rel(1.8), face="bold"),
      axis.text.y = element_text(size = rel(1.4), face="bold"),
      axis.text.x = element_text(size = rel(1.4), face="bold",angle=0,vjust = 0.5, hjust=1),
            #panel.grid = element_line(color = "lightgray",size = 0.75),
      legend.position="none",
      legend.title=element_blank(),
      legend.text=element_blank(),
      strip.text.x = element_text(size = rel(2.2)))
p
```

```
ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",module,"_all_bar.png",sep=

if(module == "#004C54"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8a.png",sep=""),heigh

if(module == "#FFC0CB"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8b.png",sep=""),heigh

if(module == "#0000FF"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9a.png",sep=""),heigh

if(module == "#008080"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9b.png",sep=""),heigh

}
```

```
####Trait Heatmaps

figureorder <- 7

#moduleTraitCor2 <- moduleTraitCor2 %>%
#  dplyr::select(-disease)

h <- 2

for(h in 2:482){
skip_to_next <<- FALSE
print(h)
trait_id <- colnames(moduleTraitCor2)[h]
#trait_id <-"uc.15"

print(trait_id)
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/",trait_id,sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",trait_id,sep=""))
}

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
  filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

geneTraitSignificance = as.data.frame(cor(datExpr,datTraits, use = "p"))

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  dplyr::mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules) %>%
  dplyr::filter(trait == trait_id) %>%
  dplyr::arrange(cor) %>%
  dplyr::mutate(position = row_number())

traitmodules <- geneModuleMembership2 %>%
  dplyr::mutate(rownamer = row.names(geneModuleMembership2)) %>%
  separate(rownamer,into=c("rownamer","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  dplyr::filter(rownamer==trait_id)

traitmodules <- t(traitmodules)

traitmodules <- as.data.frame(cbind(row.names(traitmodules),traitmodules))

colnames(traitmodules) <- c("Module","ModuleMembership")

traitpvalues <- MMPvalue2 %>%
  dplyr::mutate(rownamer = row.names(MMPvalue2)) %>%
  separate(rownamer,into=c("rownamer","kibble"),sep="___") %>%
  dplyr::select(-kibble) %>%
  dplyr::filter(rownamer==trait_id)

traitpvalues <- t(traitpvalues)

traitpvalues <- as.data.frame(cbind(paste("MM",str_remove(row.names(traitpvalues),"p.MM"),sep=""),traitp

colnames(traitpvalues) <- c("Module","MMpvalue")
traitmodules <- traitmodules %>%
  left_join(allmodules_check, by= "Module") %>%
  left_join(traitpvalues,"Module") %>%
  # Filter(checker == TRUE) %>%
  dplyr::mutate(moduleColors = str_remove(Module,"MM")) %>%
  dplyr::filter(moduleColors != "rownamer")
```

```
df_correlations <- data.frame(matrix(ncol=2,nrow=0, dimnames=list(NULL, c("module","cor"))))

i <- 1

for(i in 1:length(unique(traitmodules$moduleColors))){

module = as.character(unique(traitmodules$moduleColors)[i])
#module = "red"
column = match(module, modNames)
moduleGenes = traitmodules$moduleColors==module;
correlation <- cor(abs(geneModuleMembership[moduleGenes, column]),
                   abs(geneTraitSignificance[moduleGenes, 1]))

rbinder <- c(module,correlation)
df_correlations <- rbind(df_correlations,rbinder)


}

colnames(df_correlations) <- c("module","cor")

df_correlations <- as.data.frame(df_correlations) %>%
  mutate(Module = paste("MM",module,sep=""))

traitmodules2 <- traitmodules %>%
  left_join(df_correlations,by = "Module") %>%
  dplyr::select(module,cor,ModuleMembership,MMpvalue) %>%
  dplyr::mutate(RankModule = percent_rank(1-as.numeric(ModuleMembership)),Rankcor = percent_rank(cor),M
  dplyr::group_by(module) %>%
  dplyr::mutate(totalscore = sum(as.numeric(RankModule),as.numeric(Rankcor),na.rm=TRUE)) %>%
  dplyr::ungroup() %>%
  dplyr::arrange(as.numeric(cor)) %>%
  dplyr::mutate(position = row_number(),cor = ifelse(MMpvalue2 >= 0.05,NA,cor))


p <- ggplot(data = traitmodules2,mapping = aes(x = as.character(trait_id),y = reorder(module,position),
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("modules") +
  xlab(trait_id) +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
```

```
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(traitmodules2$module)),
    y = traitmodules2$position,
    x = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/TUCR_Database/",trait_id,"/",figureorder,"_",trait_id,"_wgcna_modulecor

if(!dir.exists(paste(outputdir,"/Figure6_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure6_Supplementary/",sep=""))
}

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure7_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure7_Supplementary/",sep=""))
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7e.png",sep=""), width = 3,
}

if(!dir.exists(paste(outputdir,"/Figure2/",sep=""))){
  dir.create(paste(outputdir,"/Figure2/",sep=""))
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2f.png",sep=""), width = 3, height = 7, dpi = 600)
}

####Trait Correlation Plots

traitmodules3 <- traitmodules2 %>%
  filter(as.numeric(MMpvalue) <= 0.05)



i <- 1

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
```

```
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_6_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_6.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_6.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_6.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- 2

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
```

```
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_5_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_5.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_5.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_5.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- 3

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
```

```
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_4_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_4.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_4.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_4.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
#module = "red"
```

```
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_1_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_1.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_1.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_1.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)-1

module = as.character(traitmodules3$module[i])
print(i)
print(paste(module))
```

```
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_2_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_2.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_2.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_2.png",sep=""), width = 5, height = 5, dpi = 600)
}

i <- length(traitmodules3$module)-2

module = as.character(traitmodules3$module[i])
print(i)
```

```
print(paste(module))
#module = "red"
correlation <- round(as.numeric(traitmodules3$cor[i]),3)
pvalue <- round(as.numeric(traitmodules3$MMpvalue[i]),6)
column = match(module, modNames);
moduleGenes = moduleColors==module;
sizeGrWindow(7, 7);
par(mfrow = c(1,1));

xvalues <- tryCatch(abs(geneModuleMembership[moduleGenes, column]),
                error = function(e) { skip_to_next <<- TRUE})
yvalues <- abs(geneTraitSignificance[moduleGenes, 1])
ngenes <- length(xvalues)

ggplot_df <- as.data.frame(cbind(xvalues,yvalues))

p <- ggplot(ggplot_df,aes(xvalues,yvalues,color=module)) +
  geom_point(size=5) +
  scale_color_identity() +
  xlab(paste("Module Membership in", module, "module")) +
                  ylab("Gene significance for trait") +
  ggtitle(paste(module,"(n = ",ngenes,", cor = ",correlation,", p = ",pvalue,")",sep="")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(1.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.4), face="bold"),
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold"),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_blank(),
        legend.text=element_blank(),
        plot.margin = unit(c(0,0,0,0), "cm"))

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/",trait_id,"/8_3_",trait_id,"_",module,".png",sep

if(trait_id == "uc.2"){
  ggsave(p,file=paste(outputdir,"/Figure6_Supplementary/supplementary_figure6f_3.png",sep=""), width = 5
}

if(trait_id == "uc.15"){
  ggsave(p,file=paste(outputdir,"/Figure7_Supplementary/supplementary_figure7f_3.png",sep=""), width = 5
}

if(trait_id == "uc.110"){
  ggsave(p,file=paste(outputdir,"/Figure2/figure2g_3.png",sep=""), width = 5, height = 5, dpi = 600)
}


}
```

Supplementary Figure 7G

Completing survival analysis for TUCRs

```
disease <- "GBM"

normal <- "cortex"

figureorder <- 6

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)
```

```
    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
```

```
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
```

```
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {

    km_countdata <- z_rna
    colnames(km_countdata) <- metadata$id[t_index]
    posdata$median <- rowMedians(survcounts)

    # km_countdata2 <- km_countdata[,complete.cases(clinical2)]

    probs <- c(0.25, 0.5, 0.75)
    q <- rowQuantiles(km_countdata, probs = probs)
    posdata$n25 <- q[, 1]
    posdata$n75 <- q[, 3]
    posdata <- posdata %>%
        dplyr::select(TUCR = alias.x, median, n25, n75)
    km_TUCRs <- cbind(posdata, km_countdata)

    km_TUCRs <- km_TUCRs %>%
        gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
        mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
        distinct %>%
        dplyr::select(TUCR, median, id, group) %>%
        left_join(clinical2, by = "id") %>%
        dplyr::filter(median != 0)

    TUCRids <- as.character(posdata$TUCR)
    i <- 1

    ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
        "pvalue", "method"))))

    for (i in 1:length(TUCRids)) {
        print(i)
        print(TUCRids[i])

        skip_to_next <- FALSE
        TUCRsurv <- as.character(TUCRids[i])
        if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
            dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
        }

        # TUCR <- 'uc.1'
        kmdata <- km_TUCRs %>%
            dplyr::filter(TUCR == TUCRsurv) %>%
            dplyr::select(TUCR, id, group, time, status)
        fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
            skip_to_next <<- TRUE
        })
```

```
        p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
            risk.table = TRUE), error = function(e) {
            skip_to_next <<- TRUE
        })
        # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
        # TRUE,risk.table = TRUE)

        if (skip_to_next == TRUE) {
            rbinder <- cbind(as.character(TUCRsurv), NA, NA)
            ptable[i, ] <- rbinder
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}

TUCRsurv <- "uc.15"
kmdata <- km_TUCRs %>%
    dplyr::filter(TUCR == TUCRsurv) %>%
    dplyr::select(TUCR, id, group, time, status)
fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
    skip_to_next <<- TRUE
})
p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE, risk.table = TRUE),
    error = function(e) {
        skip_to_next <<- TRUE
    })
# p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval = TRUE,risk.table =
# TRUE)

if (skip_to_next == TRUE) {
    rbinder <- cbind(as.character(TUCRsurv), NA, NA)
    ptable[i, ] <- rbinder
} else {
    grid.draw.ggsurvplot <- function(x) {
        survminer:::print.ggsurvplot(x, newpage = FALSE)
    }
    ggsave(file = paste(outputdir, "/Figure7_Supplementary/figure7g_supplementary.png",
        sep = ""), device = "png", plot = p)
}
```

Supplementary Figure 7H

```
disease <- "LGG"

normal <- "cortex"

figureorder <- 6

## read data

t_countfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/", disease,
    "_mergedcounts.txt", sep = "")

n_countfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_mergedcounts.txt", sep = "")

t_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_tcga_metadata.csv", sep = "")

n_metadatafile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_gtex_metadata.csv", sep = "")

t_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", disease, "/",
    disease, "_seqdepth_counts.csv", sep = "")

n_seqdepthfile <- paste("./Inputs/general_files/sequencingfiles/", normal, "/", normal,
    "_seqdepth_counts.csv", sep = "")

## Merge Data

if (!is.na(n_countfile)) {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    normalcounts <- read.table(n_countfile, header = TRUE)

    mergedcounts <- mergedcounts %>%
        left_join(normalcounts, by = c("id")) %>%
        dplyr::select(-chrom.y, -start.y, -end.y, -strand.y, -tag.y, -annot.y, -alias.y) %>%
        distinct()

    metadata <- read_csv(file = t_metadatafile)

    n_metadata <- read_csv(file = n_metadatafile)

    rm(normalcounts)

    metadata <- rbind(metadata, n_metadata)

    rm(n_metadata)

    seqdepth <- read.csv(file = t_seqdepthfile)

    n_seqdepth <- read.csv(file = n_seqdepthfile)
```

```
    seqdepth <- rbind(seqdepth, n_seqdepth)

    rm(n_seqdepth)

} else {
    mergedcounts <- read.table(t_countfile, header = TRUE)

    metadata <- read.csv(file = t_metadatafile)

    seqdepth <- read.csv(file = t_seqdepthfile)
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", sep = ""))
}

if (!dir.exists(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))) {
    dir.create(paste(outputdir, "/TUCR_Database/SummaryTables/", disease, sep = ""))
}

if (!is.na(filterannot)) {
    survcounts <- mergedcounts %>%
        filter(tag.x == filterannot & annot.x != "random")

} else {
    survcounts <- mergedcounts
}

posdata <- survcounts[, 1:8]
survcounts <- survcounts[, 9:length(colnames(survcounts))]

is.sequential <- function(x) {
    all(abs(diff(x)) == 1)
}

match_colnames <- match(as.character(colnames(survcounts)), as.character(metadata$survid))

survcounts <- as.matrix(survcounts)

n_index <- which(as.character(metadata$dex) %in% "normal")
t_index <- which(as.character(metadata$dex) %in% "tumor")

vm <- function(x) {
    # x <- mergedcounts
    cond <- factor(ifelse(seq(1, dim(x)[2], 1) %in% t_index, 1, 0))
    d <- model.matrix(~1 + cond)
    x <- t(apply(x, 1, as.numeric))
    ex <- voom(x, d, plot = F)
    return(ex$E)
```

```
}

count_vm <- vm(survcounts)

colnames(count_vm) <- metadata$id

scal <- function(x, y) {
    mean_n <- rowMeans(y)  # mean of normal
    sd_n <- apply(y, 1, sd)  # SD of normal
    # z score as (value - mean normal)/SD normal
    res <- matrix(nrow = nrow(x), ncol = ncol(x))
    colnames(res) <- colnames(x)
    rownames(res) <- rownames(x)
    for (i in 1:dim(x)[1]) {
        for (j in 1:dim(x)[2]) {
            res[i, j] <- (x[i, j] - mean_n[i])/sd_n[i]
        }
    }
    return(res)
}

z_rna <- scal(count_vm[, t_index], count_vm[, n_index])
rownames(z_rna) <- posdata[, 4]

# clinical <-
# read.table('./Inputs/general_files/survivalfiles/GBM.clin.merged.txt',header
# = TRUE)
clinical <- read.table("./Inputs/general_files/survivalfiles/glioma3.clin.merged.txt",
    header = TRUE)

clinical$time <- as.numeric(clinical$time)

clinical <- clinical %>%
    dplyr::select(barcode = patient, time, status)

clinical2 <- metadata %>%
    left_join(clinical, by = "barcode")

# Sum(clinical$patient %in% colnames(z_rna))

ind_tum <- which(unique(colnames(z_rna)) %in% clinical2$id)
ind_clin <- which(clinical2$id %in% colnames(z_rna))

out.tab <- c()
for (x in 1:nrow(count_vm)) {
    ind_gene <- x
    s <- Surv(clinical$time[ind_clin], clinical$status[ind_clin])
    cx <- coxph(formula = s ~ z_rna[ind_gene, ind_tum])
    cx <- tidy(cx)
    out.tab <- rbind(out.tab, cx)
}

surv_TUCR <- cbind(posdata, out.tab)
```

```
surv_TUCR <- surv_TUCR %>%
    dplyr::select(-term)

write_csv(surv_TUCR, paste(outputdir, "/TUCR_Database/SummaryTables/", disease, "/",
    disease, "_survival_coxph_allTUCRs.csv", sep = ""))

if (makekpmplots == TRUE) {

    km_countdata <- z_rna
    colnames(km_countdata) <- metadata$id[t_index]
    posdata$median <- rowMedians(survcounts)

    # km_countdata2 <- km_countdata[,complete.cases(clinical2)]

    probs <- c(0.25, 0.5, 0.75)
    q <- rowQuantiles(km_countdata, probs = probs)
    posdata$n25 <- q[, 1]
    posdata$n75 <- q[, 3]
    posdata <- posdata %>%
        dplyr::select(TUCR = alias.x, median, n25, n75)
    km_TUCRs <- cbind(posdata, km_countdata)

    km_TUCRs <- km_TUCRs %>%
        gather(key = "id", value = "count", -TUCR, -median, -n75, -n25) %>%
        mutate(group = ifelse(count >= n75, "high", ifelse(count <= n25, "low", NA))) %>%
        distinct %>%
        dplyr::select(TUCR, median, id, group) %>%
        left_join(clinical2, by = "id") %>%
        dplyr::filter(median != 0)

    TUCRids <- as.character(posdata$TUCR)
    i <- 1

    ptable <- data.frame(matrix(ncol = 3, nrow = 0, dimnames = list(NULL, c("TUCR",
        "pvalue", "method"))))

    for (i in 1:length(TUCRids)) {
        print(i)
        print(TUCRids[i])

        skip_to_next <- FALSE
        TUCRsurv <- as.character(TUCRids[i])
        if (!dir.exists(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))) {
            dir.create(paste(outputdir, "/TUCR_Database/", TUCRsurv, sep = ""))
        }

        # TUCR <- 'uc.1'
        kmdata <- km_TUCRs %>%
            dplyr::filter(TUCR == TUCRsurv) %>%
            dplyr::select(TUCR, id, group, time, status)
        fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
            skip_to_next <<- TRUE
        })
```

```
        p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE,
            risk.table = TRUE), error = function(e) {
            skip_to_next <<- TRUE
        })
        # p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval =
        # TRUE,risk.table = TRUE)

        if (skip_to_next == TRUE) {
            rbinder <- cbind(as.character(TUCRsurv), NA, NA)
            ptable[i, ] <- rbinder
        } else {
            grid.draw.ggsurvplot <- function(x) {
                survminer:::print.ggsurvplot(x, newpage = FALSE)
            }
            ggsave(file = paste(outputdir, "/TUCR_Database/", TUCRsurv, "/", figureorder,
                "_", TUCRsurv, "_", disease, "_kpmplot.png", sep = ""), device = "png",
                plot = p)
            p2value <- surv_pvalue(fit, data = kmdata, method = "survdiff") %>%
                dplyr::mutate(padj = p.adjust(pval, method = "bonferroni")) %>%
                dplyr::select(method, pval, padj)
            rbinder <- cbind(as.character(TUCRsurv), p2value)
            ptable[i, ] <- rbinder
        }
    }

    write_csv(ptable, paste(outputdir, "/TUCR_Database/SummaryTables/", disease,
        "/", disease, "_survival_kpm_allTUCRs.csv", sep = ""))
}

TUCRsurv <- "uc.15"
kmdata <- km_TUCRs %>%
    dplyr::filter(TUCR == TUCRsurv) %>%
    dplyr::select(TUCR, id, group, time, status)
fit <- tryCatch(survfit(Surv(time, status) ~ group, data = kmdata), error = function(e) {
    skip_to_next <<- TRUE
})
p <- tryCatch(ggsurvplot(fit, data = kmdata, conf.int = TRUE, pval = TRUE, risk.table = TRUE),
    error = function(e) {
        skip_to_next <<- TRUE
    })
# p <- ggsurvplot(fit,data=kmdata,conf.int = TRUE,pval = TRUE,risk.table =
# TRUE)

if (skip_to_next == TRUE) {
    rbinder <- cbind(as.character(TUCRsurv), NA, NA)
    ptable[i, ] <- rbinder
} else {
    grid.draw.ggsurvplot <- function(x) {
        survminer:::print.ggsurvplot(x, newpage = FALSE)
    }
    ggsave(file = paste(outputdir, "/Figure7_Supplementary/figure7h_supplementary.png",
        sep = ""), device = "png", plot = p)
}
```

Supplementary Figure 8

```
if (!dir.exists(paste(outputdir, "/Figure8_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure8_Supplementary/", sep = ""))
}
```

Supplementary Figure 8A and 8B

```
# Load the WGCNA package
library(WGCNA)
# The following setting is important, do not omit.
options(stringsAsFactors = FALSE)
# Load the expression and trait data saved in the first part
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-01-dataInput.RData")
# The variable lnames contains the names of loaded variables.
lnames
# Load network data saved in the second part.
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-02-networkConstruction-auto.RData")
lnames
```

filter out modules that are particularly large.

```
# Define numbers of genes and samples
nGenes = ncol(datExpr)
nSamples = nrow(datExpr)
# Recalculate MEs with color labels
MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes

datExpr2 = as.data.frame(datExpr)
MEs = orderMEs(MEs0)
datTraits2 <- as.data.frame(datTraits2)
moduleTraitCor = cor(MEs, datTraits2, use = "p")
moduleTraitPvalue = corPvalueStudent(moduleTraitCor, nSamples)
```

```
geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))
MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

allmodules <- as.data.frame(t(MMPvalue))

allmodules2 <- allmodules %>%
    dplyr::mutate(Module = row.names(allmodules)) %>%
    gather(key = "Gene", value = "pvalue", -Module) %>%
    group_by(Gene) %>%
    mutate(maxmodule = min(pvalue, na.rm = TRUE)) %>%
    filter(pvalue == maxmodule) %>%
    group_by(Module) %>%
    dplyr::summarize(n = n())

allmodules3 <- allmodules2 %>%
    mutate(q1 = quantile(allmodules2$n, 0.25), q3 = quantile(allmodules2$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# repeat{
```

```
n_count <- length(allmodules3$n)

allmodules3 <- allmodules3 %>%
    mutate(q1 = quantile(allmodules3$n, 0.25), q3 = quantile(allmodules3$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# if(n_count == length(allmodules3$n)) break;

# }

allmodules_check <- as.data.frame(as.character(allmodules3$Module)) %>%
    dplyr::mutate(checker = TRUE)

colnames(allmodules_check) <- c("Module", "checker")

boxplot(allmodules3$n, ylab = "n")

moduleTraitCor2 <- as.data.frame(moduleTraitCor) %>%
    mutate(Module = row.names(moduleTraitCor)) %>%
    left_join(allmodules_check, by = "Module") %>%
    filter(checker == TRUE)

datTraits <- datTraits2

geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))

match_modules <- match(as.character(allmodules_check$Module), colnames(geneModuleMembership))

geneModuleMembership2 <- geneModuleMembership[, match_modules]

modNames = substring(names(geneModuleMembership2), 3)

MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

match_modules <- match(as.character(allmodules_check$Module), colnames(MMPvalue))

MMPvalue2 = MMPvalue[, match_modules]

names(geneModuleMembership2) = paste("MM", modNames, sep = "")
names(MMPvalue2) = paste("p.MM", modNames, sep = "")

# Save.image(file='./Inputs/general_files/wgcnafiles/allprelims.Rdata')
```

```
load(file="./Inputs/general_files/wgcnafiles/allprelims.Rdata")

datTraits2 <- datTraits2 %>%
  dplyr::select(-disease)

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
  # Filter(rownames == trait_id) %>%
```

```
  dplyr::select(-rownames) %>%
  t()

hc.Traits <- hclust(dist(t(datTraits2)))

clust_order_traits <- hc.Traits$order

clust_positions <- as.data.frame(cbind(as.character(colnames(datTraits)[clust_order_traits]),as.numeric

colnames(clust_positions) <- c("trait","clust_order")

geneTraitSignificance <- as.data.frame(cor(datExpr,datTraits2, use = "p")) %>%
  dplyr::select(-dex2,-p53status2)

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules,-Module,-checker,-p53status2,-dex2) %>%
  # Filter(is.numeric(cor)) %>%
  # Filter(trait == trait_id) %>%
  dplyr::group_by(trait) %>%
  arrange(desc(cor)) %>%
  ungroup() %>%
  dplyr::group_by(modules) %>%
  dplyr::mutate(sumnumber = sum(cor,na.rm=T),
         n = n(),
         weight = sumnumber/n) %>%
  ungroup() %>%
  arrange(desc(weight))

traitpositions <- moduleTraitCor_trait_id %>%
  dplyr::select(modules) %>%
  distinct() %>%
  dplyr::mutate(position = row_number())

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  left_join(traitpositions,by="modules") %>%
  left_join(clust_positions,by="trait")

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  mutate(clust_position_order = as.numeric(moduleTraitCor_trait_id$clust_order))

p <- ggplot(data = moduleTraitCor_trait_id,mapping = aes(x = reorder(modules,position),y = reorder(trai
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("TUCRs") +
  xlab("Modules") +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
```

```
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(moduleTraitCor_trait_id$modules)),
    x = moduleTraitCor_trait_id$position,
    y = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/Figure1/figure1h.png",sep=""), width = 7, height = 3, dpi = 600)
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))
}

#####GO-TERM ANALYSIS

if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")

if (!require('GO.db')) BiocManager::install('GO.db')
library(GO.db)

#if (!require('AnnotationDBI')) BiocManager::install('AnnotationDBI')
#library(AnnotationDBI)

if (!require('org.Hs.eg.db')) BiocManager::install('org.Hs.eg.db')
library(org.Hs.eg.db)

if (!require('limma')) BiocManager::install('limma')
library(limma)

i <- 1
```

```
genelist <- geneTraitSignificance %>%
  mutate(genenames = row.names(geneTraitSignificance))

for(i in 1:length(unique(moduleTraitCor_trait_id$modules))){

  print(i)
  print(unique(moduleTraitCor_trait_id$modules)[i])

  module <- unique(moduleTraitCor_trait_id$modules)[i]
  #module <- "yellowgreen"
  column = match(module, modNames);
  moduleGenes = moduleColors==module;

  genelist2 <- as.data.frame(genelist[moduleGenes,])

  genelist2 <- genelist2 %>%
    separate(genenames,into=c("Alias","kibble"),sep="___")

  symbols <- as.character(genelist2$Alias)

  EntrezIDs <- mapIds(org.Hs.eg.db, symbols, 'ENTREZID', 'SYMBOL')
  allgenes <- cbind(symbols,EntrezIDs)
  allgenes <- allgenes[complete.cases(allgenes),]

  g <- goana(EntrezIDs)

  g_bp <- g %>%
    filter(Ont == "BP")
  topGO_bp <- topGO(g_bp) %>%
    mutate(log10_p = -log10(P.DE),module=module,color="red") %>%
    arrange(desc(log10_p))

  g_mf <- g %>%
    filter(Ont == "MF")
  topGO_mf <- topGO(g_mf) %>%
    mutate(log10_p = -log10(P.DE),module=module,color="blue") %>%
    arrange(desc(log10_p))

  topGO_all <- rbind(topGO_bp,topGO_mf) %>%
                       arrange(as.numeric(log10_p)) %>%
                       dplyr::mutate(roworder = row_number()) %>%
                       mutate(newTerm = substr(Term,1,80))

  p <- ggplot(topGO_all,aes(x=reorder(newTerm,roworder),y=as.numeric(log10_p),fill=module,color=color))
    geom_bar(stat="identity") + coord_flip() + scale_fill_identity() + scale_color_identity() + facet_wr
    ggtitle(module) +
    labs(x="Go Term", y = "-log10(p-value)") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
```

```
        axis.text.y = element_text(size = rel(1.4), face="bold"),
        axis.text.x = element_text(size = rel(1.4), face="bold",angle=0,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank(),
        strip.text.x = element_text(size = rel(2.2)))
  p

ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",module,"_all_bar.png",sep=

if(module == "#004C54"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8a.png",sep=""),heigh

if(module == "#FFC0CB"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8b.png",sep=""),heigh

if(module == "#0000FF"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9a.png",sep=""),heigh

if(module == "#008080"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9b.png",sep=""),heigh

}
```

Supplementary Figure 9

```
if (!dir.exists(paste(outputdir, "/Figure9_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure9_Supplementary/", sep = ""))
}
```

Supplementary Figure 9A and 9B

```
# Load the WGCNA package
library(WGCNA)
# The following setting is important, do not omit.
options(stringsAsFactors = FALSE)
# Load the expression and trait data saved in the first part
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-01-dataInput.RData")
# The variable lnames contains the names of loaded variables.
lnames
# Load network data saved in the second part.
lnames = load(file = "./Inputs/general_files/wgcnafiles/TUCR-02-networkConstruction-auto.RData")
lnames
```

filter out modules that are particularly large.

```
# Define numbers of genes and samples
nGenes = ncol(datExpr)
nSamples = nrow(datExpr)
# Recalculate MEs with color labels
MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes

datExpr2 = as.data.frame(datExpr)
MEs = orderMEs(MEs0)
datTraits2 <- as.data.frame(datTraits2)
moduleTraitCor = cor(MEs, datTraits2, use = "p")
moduleTraitPvalue = corPvalueStudent(moduleTraitCor, nSamples)
```

```
geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))
MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

allmodules <- as.data.frame(t(MMPvalue))

allmodules2 <- allmodules %>%
    dplyr::mutate(Module = row.names(allmodules)) %>%
    gather(key = "Gene", value = "pvalue", -Module) %>%
    group_by(Gene) %>%
    mutate(maxmodule = min(pvalue, na.rm = TRUE)) %>%
    filter(pvalue == maxmodule) %>%
    group_by(Module) %>%
    dplyr::summarize(n = n())

allmodules3 <- allmodules2 %>%
    mutate(q1 = quantile(allmodules2$n, 0.25), q3 = quantile(allmodules2$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# repeat{

n_count <- length(allmodules3$n)

allmodules3 <- allmodules3 %>%
    mutate(q1 = quantile(allmodules3$n, 0.25), q3 = quantile(allmodules3$n, 0.75),
        q1minus = q1 - 1.5 * (q3 - q1), q3plus = q3 + 1.5 * (q3 - q1), outliercheck = ifelse(n >=
```

```
            q3plus | n <= q1minus, TRUE, FALSE))
# %>% filter(outliercheck == FALSE)

# if(n_count == length(allmodules3$n)) break;

# }

allmodules_check <- as.data.frame(as.character(allmodules3$Module)) %>%
    dplyr::mutate(checker = TRUE)

colnames(allmodules_check) <- c("Module", "checker")

boxplot(allmodules3$n, ylab = "n")

moduleTraitCor2 <- as.data.frame(moduleTraitCor) %>%
    mutate(Module = row.names(moduleTraitCor)) %>%
    left_join(allmodules_check, by = "Module") %>%
    filter(checker == TRUE)

datTraits <- datTraits2

geneModuleMembership = as.data.frame(cor(datExpr, MEs, use = "p"))

match_modules <- match(as.character(allmodules_check$Module), colnames(geneModuleMembership))

geneModuleMembership2 <- geneModuleMembership[, match_modules]

modNames = substring(names(geneModuleMembership2), 3)

MMPvalue = as.data.frame(corPvalueStudent(as.matrix(geneModuleMembership), nSamples))

match_modules <- match(as.character(allmodules_check$Module), colnames(MMPvalue))

MMPvalue2 = MMPvalue[, match_modules]

names(geneModuleMembership2) = paste("MM", modNames, sep = "")
names(MMPvalue2) = paste("p.MM", modNames, sep = "")

# Save.image(file='./Inputs/general_files/wgcnafiles/allprelims.Rdata')
```

```
load(file="./Inputs/general_files/wgcnafiles/allprelims.Rdata")

datTraits2 <- datTraits2 %>%
  dplyr::select(-disease)

datTraits <- as.data.frame(t(datTraits2)) %>%
  mutate(rownames = row.names(t(datTraits2))) %>%
  # Filter(rownames == trait_id) %>%
  dplyr::select(-rownames) %>%
  t()

hc.Traits <- hclust(dist(t(datTraits2)))

clust_order_traits <- hc.Traits$order
```

```
clust_positions <- as.data.frame(cbind(as.character(colnames(datTraits)[clust_order_traits]),as.numeric

colnames(clust_positions) <- c("trait","clust_order")

geneTraitSignificance <- as.data.frame(cor(datExpr,datTraits2, use = "p")) %>%
  dplyr::select(-dex2,-p53status2)

GSPvalue = as.data.frame(corPvalueStudent(as.matrix(geneTraitSignificance), nSamples))

#names(geneTraitSignificance) = paste("GS.", names(weight), sep="")
#names(GSPvalue) = paste("p.GS.", names(weight), sep="")

moduleTraitCor_trait_id <- as.data.frame(moduleTraitCor2) %>%
  mutate(modules = str_remove(Module,"ME")) %>%
  gather(key = "trait", value = "cor",-modules,-Module,-checker,-p53status2,-dex2) %>%
  # Filter(is.numeric(cor)) %>%
  # Filter(trait == trait_id) %>%
  dplyr::group_by(trait) %>%
  arrange(desc(cor)) %>%
  ungroup() %>%
  dplyr::group_by(modules) %>%
  dplyr::mutate(sumnumber = sum(cor,na.rm=T),
         n = n(),
         weight = sumnumber/n) %>%
  ungroup() %>%
  arrange(desc(weight))

traitpositions <- moduleTraitCor_trait_id %>%
  dplyr::select(modules) %>%
  distinct() %>%
  dplyr::mutate(position = row_number())

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  left_join(traitpositions,by="modules") %>%
  left_join(clust_positions,by="trait")

moduleTraitCor_trait_id <- moduleTraitCor_trait_id %>%
  mutate(clust_position_order = as.numeric(moduleTraitCor_trait_id$clust_order))

p <- ggplot(data = moduleTraitCor_trait_id,mapping = aes(x = reorder(modules,position),y = reorder(trai
  geom_tile() +
  scale_fill_gradient2(low = paper_blue,mid = "white",high = paper_red2,midpoint = 0) +
  ylab("TUCRs") +
  xlab("Modules") +
  labs(fill = "cor") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_blank(),
        axis.text.x = element_blank(),
```

```
            #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="right",
        legend.title=element_text(size = rel(2.5), face="bold"),
        legend.text=element_text(size = rel(2.0), face="bold"),
        plot.margin = unit(c(0,0,0,0), "cm")) +
        annotate(
    geom = "point",
    color = c(as.character(moduleTraitCor_trait_id$modules)),
    x = moduleTraitCor_trait_id$position,
    y = 0.5,
    shape = 15,
    size = 5)

p

ggsave(p,file=paste(outputdir,"/Figure1/figure1h.png",sep=""), width = 7, height = 3, dpi = 600)
```

```
if(!dir.exists(paste(outputdir,"/TUCR_Database/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/",sep=""))
}

if(!dir.exists(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))){
  dir.create(paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",sep=""))
}

#####GO-TERM ANALYSIS

if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")

if (!require('GO.db')) BiocManager::install('GO.db')
library(GO.db)

#if (!require('AnnotationDBI')) BiocManager::install('AnnotationDBI')
#library(AnnotationDBI)

if (!require('org.Hs.eg.db')) BiocManager::install('org.Hs.eg.db')
library(org.Hs.eg.db)

if (!require('limma')) BiocManager::install('limma')
library(limma)

i <- 1

genelist <- geneTraitSignificance %>%
  mutate(genenames = row.names(geneTraitSignificance))

for(i in 1:length(unique(moduleTraitCor_trait_id$modules))){

  print(i)
  print(unique(moduleTraitCor_trait_id$modules)[i])
```

```
module <- unique(moduleTraitCor_trait_id$modules)[i]
#module <- "yellowgreen"
column = match(module, modNames);
moduleGenes = moduleColors==module;

genelist2 <- as.data.frame(genelist[moduleGenes,])

genelist2 <- genelist2 %>%
  separate(genenames,into=c("Alias","kibble"),sep="___")

symbols <- as.character(genelist2$Alias)

EntrezIDs <- mapIds(org.Hs.eg.db, symbols, 'ENTREZID', 'SYMBOL')
allgenes <- cbind(symbols,EntrezIDs)
allgenes <- allgenes[complete.cases(allgenes),]

g <- goana(EntrezIDs)

g_bp <- g %>%
  filter(Ont == "BP")
topGO_bp <- topGO(g_bp) %>%
  mutate(log10_p = -log10(P.DE),module=module,color="red") %>%
  arrange(desc(log10_p))

g_mf <- g %>%
  filter(Ont == "MF")
topGO_mf <- topGO(g_mf) %>%
  mutate(log10_p = -log10(P.DE),module=module,color="blue") %>%
  arrange(desc(log10_p))

topGO_all <- rbind(topGO_bp,topGO_mf) %>%
                     arrange(as.numeric(log10_p)) %>%
                     dplyr::mutate(roworder = row_number()) %>%
                     mutate(newTerm = substr(Term,1,80))

p <- ggplot(topGO_all,aes(x=reorder(newTerm,roworder),y=as.numeric(log10_p),fill=module,color=color))
  geom_bar(stat="identity") + coord_flip() + scale_fill_identity() + scale_color_identity() + facet_w
  ggtitle(module) +
  labs(x="Go Term", y = "-log10(p-value)") +
theme(panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      panel.background = element_blank(),
      axis.line = element_line(colour = "black"),
      plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
      axis.title = element_text(size = rel(1.8), face="bold"),
      axis.text.y = element_text(size = rel(1.4), face="bold"),
      axis.text.x = element_text(size = rel(1.4), face="bold",angle=0,vjust = 0.5, hjust=1),
            #panel.grid = element_line(color = "lightgray",size = 0.75),
      legend.position="none",
      legend.title=element_blank(),
      legend.text=element_blank(),
      strip.text.x = element_text(size = rel(2.2)))
p
```

```
ggsave(plot = print(p),paste(outputdir,"/TUCR_Database/SummaryTables/WGCNA/",module,"_all_bar.png",sep=

if(module == "#004C54"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8a.png",sep=""),heigh

if(module == "#FFC0CB"){

if(!dir.exists(paste(outputdir,"/Figure8_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure8_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure8_Supplementary/supplementary_figure8b.png",sep=""),heigh

if(module == "#0000FF"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9a.png",sep=""),heigh

if(module == "#008080"){

if(!dir.exists(paste(outputdir,"/Figure9_Supplementary/",sep=""))){
  dir.create(paste(outputdir,"/Figure9_Supplementary/",sep=""))
}

ggsave(plot = print(p),paste(outputdir,"/Figure9_Supplementary/supplementary_figure9b.png",sep=""),heigh

}
```

Supplementary Figure 10

Supplementary Figure 10A, 10B, 10C, and 10E

```
if (!dir.exists(paste(outputdir, "/Figure10_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure10_Supplementary/", sep = ""))
}

Figure_copy <- list.files("./Inputs/Figure10_Supplementary/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure10_Supplementary/", "supplementary_Figure10a.png",
    sep = ""))) {
    file.copy(paste0("./Inputs/Figure10_Supplementary/pregenerated_figures/", Figure_copy,
        sep = ""), paste("./", outputdir, "/Figure10_Supplementary/", sep = ""))
}
```

Supplementary Figure 10D

```
data <- read_csv("Inputs/Figure10_supplementary/supplementary_figure10d_input.csv")

p <- ggplot(data,aes(x=reorder(Sample,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label= pvalue,vjust=-1.5,hjust=0.5),color = "#000000",size=10,position = p
        scale_fill_manual(values = c(paper_purple,paper_red,paper_gold,paper_green,paper_blue)) +
        labs(y = "Relative GBX2 Expression",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  facet_wrap(~CellLine,scales="free")

p

ggsave(file=paste(outputdir,"/Figure10_supplementary/supplementary_figure10d.png",sep=""),
       plot=print(p),
       height = 9,
       width = 5,
       dpi = 600)
```

Supplementary Figure 11

Supplementary Figure 11A, 11B, and 11F (images)

```
if (!dir.exists(paste(outputdir, "/Figure11_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure11_Supplementary/", sep = ""))
}

Figure_copy <- list.files("./Inputs/Figure11_Supplementary/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure11_Supplementary/", "supplementary_Figure11ab.png",
    sep = ""))) {
    file.copy(paste0("./Inputs/Figure11_Supplementary/pregenerated_figures/", Figure_copy,
        sep = ""), paste("./", outputdir, "/Figure11_Supplementary/", sep = ""))
}
```

Supplementary Figure 11C

uc.110 expression in GBM Cell Lines

```
data <- read_csv("./Inputs/Figure11_Supplementary/supplementary_figure11c_input.csv")

p <- ggplot(data,aes(x=reorder(Sample,Order), y=as.numeric(FoldChange),fill=Sample)) +
```

```
        geom_bar(stat="identity"
                 #,color="black"
                   ) +
        geom_errorbar(aes(ymax=FoldChange+stderr, ymin=FoldChange-stderr, width=.2)) +
        geom_text(aes(y = FoldChange, label=pvalue,vjust=ifelse(FoldChange>0,0,1.75),hjust=0.5),color =
        scale_y_continuous(limits = c(-5,10),breaks= scales::pretty_breaks(n=15)) +
        #geom_hline(yintercept = log10(5),linetype = 2,size=1.5) +
        #geom_vline(xintercept = -0.65,linetype = 2,size=1.5) +
        #geom_vline(xintercept = 0.65,linetype = 2,size=1.5) +
        #scale_color_identity(guide = "legend", labels = cellfracdata$Fraction, breaks = cellfracdata$F
        labs(y = "log2(uc.110 FoldChange)",
        x = "GBM Cell Line") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.0), face="bold"),
              axis.text.x = element_text(size = rel(2.0), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="none",le
legend.text=element_blank()) +
  coord_cartesian(clip = "off")

p

ggsave(file=paste(outputdir,"/Figure11_Supplementary/supplementary_Figure11c.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
       dpi = 600)
```

Supplementary Figure 11D

uc.110 expression in GBM Adherent Cell Lines (summarized)

```
data <- read_csv("./Inputs/Figure11_Supplementary/supplementary_figure11d_input.csv")

p <- ggplot(data,aes(x=dex, y=FC,fill=Fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = data, aes(x=dex, y=FC),binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_text(size = rel(2.5), face="bold"),
        axis.text.x = element_text(size = rel(2.5), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
```

```
  labs(y="log2FoldChange", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

p

ggsave(file=paste(outputdir,"/Figure11_Supplementary/supplementary_Figure11d.png",sep=""),
       plot = print(p),
       height = 7,
       width = 4,
       dpi = 600)
```

Supplementary Figure 11E

uc.110 expression in GBM Stem Cell Lines (summarized)

```
data <- read_csv("./Inputs/Figure11_Supplementary/supplementary_figure11e_input.csv")

p <- ggplot(data,aes(x=dex, y=FC,fill=Fill)) +
  geom_boxplot(outlier.color="white") +
  #geom_bar(colour="black",stat="identity") +
  scale_fill_identity() +
  #scale_color_manual(values = c("black"="black","red"="red","green"="green")) +
  geom_jitter(data = data, aes(x=dex, y=FC),binaxis='y', stackdir='center', stackratio=0.90, dotsize=0.3
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)),
        axis.title = element_text(size = rel(1.8), face="bold"),
        axis.text.y = element_text(size = rel(2.5), face="bold"),
        axis.text.x = element_text(size = rel(2.5), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
        legend.position="none",
        legend.title=element_blank(),
        legend.text=element_blank()) +
  labs(y="log2FoldChange", x = "Tissue Type") +
  stat_summary(fun.y=mean, geom="point", shape=20, size=10, color=paper_blue, fill=paper_blue)

p

ggsave(file=paste(outputdir,"/Figure11_Supplementary/supplementary_Figure11e.png",sep=""),
       plot = print(p),
       height = 7,
       width = 4,
       dpi = 600)
```

Supplementary Figure 11F

```
data <- read_csv("./Inputs/Figure11_Supplementary/supplementary_figure11f_input.csv") %>%
  mutate(fillcolor = paste(Gene,Sample,sep=" "))

p <- ggplot(data,aes(x=reorder(Gene,order), y=FC,fill=reorder(Sample,order))) +
        geom_bar(stat="identity",position = "dodge") +
```

```
        geom_errorbar(aes(ymax=FC+stderr, ymin=FC-stderr, width=.2),position = position_dodge(0.9)) +
        geom_text(aes(y = FC, label=pvalue,vjust=-0.25,hjust=0.5),color = "#000000",size=10,position = p
        scale_fill_manual(values = c(paper_red,paper_green,paper_blue,paper_red,paper_green,paper_blue))
        labs(y = "Relative invading cells",
        x = "") +
        #ylim(-5, 10) +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(1.6), face="bold"),
              axis.text.y = element_text(size = rel(2.2), face="bold"),
              axis.text.x = element_text(size = rel(2.2), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=rel(2.0)),
strip.text.x = element_text(size = rel(2.2)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~CellLine)


p

ggsave(file=paste(outputdir,"/Figure11_Supplementary/supplementary_Figure11f.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
       dpi = 600)
```

Supplementary Figure 11G

Cell Fractionation Data

```
data <- read_csv("./Inputs/Figure11_Supplementary/supplementary_figure11g_input.csv")

p <- ggplot(data,aes(x=reorder(Sample,Order), y=Value,fill=Fraction)) +
        geom_bar(stat="identity"
                 #,color="black"
                 ) +
        geom_errorbar(aes(ymax=Ymax, ymin=Ymin, width=.2,vjust=0.5,hjust=1)) +
        geom_text(aes(y = 1.02, label=pvalue),color = "black",size=10) +
        #geom_hline(yintercept = log10(5),linetype = 2,size=1.5) +
        #geom_vline(xintercept = -0.65,linetype = 2,size=1.5) +
        #geom_vline(xintercept = 0.65,linetype = 2,size=1.5) +
        #scale_y_continuous(breaks = scales::pretty_breaks(n = 5)) +
        scale_fill_manual(values = c(paper_red2,paper_turq)) +
        #guides(colour = guide_legend(override.aes = list(size=10))) +
        labs(y = "fraction of total RNA expression",
        x = "Sample") +
        theme(plot.title = element_text(size = rel(2.2), hjust = 0.5, face="bold",margin=margin(0,0,0,0)
              axis.title = element_text(size = rel(2.0), face="bold"),
              axis.text.y = element_text(size = rel(2.0), face="bold"),
              axis.text.x = element_text(size = rel(2.0), face="bold",angle=90,vjust = 0.5, hjust=1),
              #panel.grid = element_line(color = "lightgray",size = 0.75),
panel.background = element_blank(), axis.line = element_line(colour = "black"),legend.position="top",leg
legend.text=element_text(size=20),
```

```
strip.text.x = element_text(size = rel(1.6)),
strip.background = element_blank()) +
  coord_cartesian(clip = "off") +
  facet_wrap(~Gene, ncol=5)

p

ggsave(file=paste(outputdir,"/Figure11_Supplementary/supplementary_Figure11g.png",sep=""),
       plot = print(p),
       height = 6,
       width = 6,
       dpi = 600)
```

Supplementary Figure 12

```
if (!dir.exists(paste(outputdir, "/Figure12_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure12_Supplementary/", sep = ""))
}

Figure_copy <- list.files("./Inputs/Figure12_Supplementary/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure12_Supplementary/", "supplementary_Figure12a.png",
    sep = ""))) {
    file.copy(paste0("./Inputs/Figure12_Supplementary/pregenerated_figures/", Figure_copy,
        sep = ""), paste("./", outputdir, "/Figure12_Supplementary/", sep = ""))
}
```

Supplementary Figure 13

```
if (!dir.exists(paste(outputdir, "/Figure13_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure13_Supplementary/", sep = ""))
}

Figure_copy <- list.files("./Inputs/Figure13_Supplementary/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure13_Supplementary/", "supplementary_Figure13a.png",
    sep = ""))) {
    file.copy(paste0("./Inputs/Figure13_Supplementary/pregenerated_figures/", Figure_copy,
        sep = ""), paste("./", outputdir, "/Figure13_Supplementary/", sep = ""))
}
```

Supplementary Figure 14

```
if (!dir.exists(paste(outputdir, "/Figure14_Supplementary/", sep = ""))) {
    dir.create(paste(outputdir, "/Figure14_Supplementary/", sep = ""))
}

Figure_copy <- list.files("./Inputs/Figure14_Supplementary/pregenerated_figures/")

if (!file.exists(paste(outputdir, "/Figure14_Supplementary/", "supplementary_figure14ac.png",
    sep = ""))) {
    file.copy(paste0("./Inputs/Figure14_Supplementary/pregenerated_figures/", Figure_copy,
```

```
        sep = ""), paste("./", outputdir, "/Figure14_Supplementary/", sep = ""))
}
```

ADDITIONAL MATERIALS AND METHODS

Data Availability Statement

RNA-Seq data for Figure 6A will be made available on the Gene Expression Omnibus (GEO) prior to publication. Detailed TUCR results can be found in supplementary materials, and also online at www.abounaderlab.org/tucr-database/. Please refer to the corresponding author for any data access questions.

TUCR Annotations [29, 30]

TUCR annotations were performed manually by overlaying consensus TUCR genomic annotation tracks to the hg38 human genome in the UCSC Genome Browser. In parallel, bedtools closest was used to identify genes that are intergenic or intragenic. These results were then cross referenced to identify a consensus genomic annotation for each TUCR. Detailed methods can be found as a codebook provided in the supplementary materials and online at: github.com/abounaderlab/tucr_project

TUCR Chromatin Landscaping

U87 H3K4me3, RNA Pol.II, and H3K27ac CHIP-Seq data and U87 ATAC-Seq data were acquired from the Gene Expression Omnibus. Randomized control TUCRs were generated using Quinlan Labs' bedtools [31, 32] and the shuffle command. [31, 32] Bedtools fisher and R/RStudio [53, 54] were used to perform chi-square tests to compare predicted overlaps of peaks to expected peaks. Detailed methods can be found as a codebook provided in the supplementary materials and online at: github.com/abounaderlab/tucr_project

```
#using SRA Toolkit version 2.10.5

#Transcriptional Amplification in Tumor Cells with Elevated c-Myc
#https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE36354

#CHIP-Seq U87-H3K4me3

prefetch SRR444442

#CHIP-Seq U87-RNAPolII

prefetch SRR444478

#CHIP-Seq U87-H3K27ac

prefetch SRR444436
```

Generate fastq files

```
for sra in SRR*
do
echo $sra
fastq-dump $sra
done
```

Build hg38 genome

```
wget https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/hg38.fa.gz
gunzip hg38.fa.gz

#Rivanna only.  Skip otherwise.
module load gcc/7.1.0
module load bowtie2/2.2.9

bowtie2-build hg38.fa hg38
```

Align fastq files to reference genome

```
for fq in *.fastq.gz
do
name=$(echo $fq | awk -F".fastq.gz" '{print $1}')
echo $name
bowtie2 -q -x hg38 -U $fq -S $name.sam
done


for fq in *.fastq
do
name=$(echo $fq | awk -F".fastq" '{print $1}')
echo $name
bowtie2 --no-unal -q -x hg38 -U $fq -S $name.sam
done
```

Convert sam to bam

```
for sam in *.sam
do
name=$(echo $sam | awk -F".sam" '{print $1}')
echo $name
samtools view -b $sam | samtools sort -o $name.bam
done
rm *sam
```

Index bam files

```
for bam in *bam
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools index $bam
done
```

Convert to bed

I got this methodology from here since my other methods were not working.

```
macs2 callpeak -t SRR444442.bam -n SRR444442 -g hs --nomodel
macs2 callpeak -t SRR444478.bam -n SRR444478 -g hs --nomodel
macs2 callpeak -t SRR444436.bam -n SRR444436 -g hs --nomodel


for SRR in $(find . -name 'SRR8723*')
```

```
do

name="${SRR##*/}"
echo $name
bam=$(echo $name/$name.bam)
echo $bam

macs2 callpeak -t $bam -n $name -g hs --nomodel

done

wc -l *Peak

cat SRR444442._peaks.narrowPeak | cut -f 1-4 | grep -v KI | grep -v MT | grep -v GL | grep -v JH | grep
cat SRR444478._peaks.narrowPeak | cut -f 1-4 | grep -v KI | grep -v MT | grep -v GL | grep -v JH | grep
cat SRR444436._peaks.narrowPeak | cut -f 1-4 | grep -v KI | grep -v MT | grep -v GL | grep -v JH | grep
```

```
wget ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM3669nnn/GSM3669993/suppl/GSM3669993_U87_ATAC.highPValue.

gunzip GSM3669993_U87_ATAC.highPValue.distal_intersect.H3K27Ac.bed.gz

mv GSM3669993_U87_ATAC.highPValue.distal_intersect.H3K27Ac.bed.gz > ATAC.bed
```

Download Genome Annotation file and sort

```
wget ftp://ftp.ensembl.org/pub/release-90/fasta/homo_sapiens/dna/Homo_sapiens.GRCh38.dna.primary_assembl

gunzip Homo_sapiens.GRCh38.dna.primary_assembly.fa.gz

samtools faidx Homo_sapiens.GRCh38.dna.primary_assembly.fa

awk -v OFS='\t' {'print $1,$2'} Homo_sapiens.GRCh38.dna.primary_assembly.fa.fai > hg38_genomeFile.txt

cat hg38_genomeFile.txt | grep -v KI | grep -v MT | grep -v GL | sed -e "s/^/chr/g" > temp

mv temp hg38_genomeFile.txt


bedtools sort -i pol2.bed -g hg38_genomeFile.txt > U87_pol2.sort.bed

bedtools sort -i h3k4me3.bed -g hg38_genomeFile.txt > U87_h3k4me3.sort.bed

bedtools sort -i h3k27ac.bed -g hg38_genomeFile.txt > U87_h3k27ac.sort.bed

bedtools sort -i h3k27ac.bed -g hg38_genomeFile.txt > U87_ATAC.sort.bed
```

Generate Random Intervals and sort all files

```
bedtools shuffle -i BEDFiles/hg38.ultraConserved.bed -g hg38_genomeFile.txt -seed 03072022 > random_hg3
Conserved.bed

for bed in $(find BEDFiles/*.bed -name '*.bed')
do
```

```
name=$(echo $bed | awk -F ".bed" '{print $1}')
echo $name

bedtools sort -g hg38_genomeFile.txt -i $bed > $name.sort.bed

done

rm *sort.sort.bed

rm BEDFiles/*sort.sort.bed
```

```
for bed in $(find * -name '*TUCR.bed')
do

name=$(echo "${bed##*/}" | awk -F ".bed" '{print $1}')
echo $name

bedtools slop -b 4000 -i $bed -g hg38_genomeFile.txt > temp

bedtools sort -i temp -g hg38_genomeFile.txt > $name.sort.bed

rm temp

done

for bed in $(find * -name '*CHESS.bed')
do

name=$(echo "${bed##*/}" | awk -F ".bed" '{print $1}')
echo $name

bedtools sort -g hg38_genomeFile.txt -i $bed > $name.sort.bed

done

for bed in $(find bedfiles/* -name '*.sort.bed')
do

name=$(echo "${bed##*/}" | awk -F ".sort.bed" '{print $1}')
echo $name

bedtools fisher -a $bed -b U87_pol2.sort.bed -g hg38_genomeFile.txt > Inputs/figure_specific_files/Figu

bedtools fisher -a $bed -b U87_h3k4me3.sort.bed -g hg38_genomeFile.txt > Inputs/figure_specific_files/F

bedtools fisher -a $bed -b U87_h3k27ac.sort.bed -g hg38_genomeFile.txt > Inputs/figure_specific_files/F

bedtools fisher -a $bed -b U87_ATAC.bed -g hg38_genomeFile.txt > Inputs/figure_specific_files/Figure2_S

done
```

TCGA AND GTEx RNA-Seq Data Download and Analysis [33, 34]

GBM (n = 161) and LGG (n = 505) RNA-Seq data were acquired from the Cancer Genome Atlas and were compared to normal brain cortex from the Genotype-Tissue Expression Database (GTEx, n = 260) using a workflow including bedtools, bowtie, the SRA toolkit, and R/RStudio. Detailed methods can be found as a codebook provided in the supplementary materials and online at: github.com/abounaderlab/tucr_project

Setup: RNA-Seq Analyses

The next part of this analysis involves using bash and R scripts to identify TUCRs that are deregulated and/or correlated with survival in GBM and LGG. The GBM analyses were stored in a directory named tcga_analysis, while the LGG analyses were stored in the lgg_tcga_analysis.

The following script generates these two directories.

```
## Create directory for all raw data from RNA-Seq analysis
mkdir RNASEQ-RAW

## Create directory for all processed data from RNA-Seq analysis
mkdir RNASEQ-PROCESSED

## Create directory for all TCGA analyses.
mkdir RNASEQ-PROCESSED/tcga_analysis/

## Create a subdirectory for each disease
mkdir RNASEQ-PROCESSED/tcga_analysis/GBM

mkdir RNASEQ-PROCESSED/tcga_analysis/LGG

## Create directory for all GTex analyses.
mkdir RNASEQ-PROCESSED/gtex_analysis/

## Create a subdirectory for cortex samples
mkdir RNASEQ-PROCESSED/gtex_analysis/cortex/reindex/
```

GTex Processing: Download RNA-Seq Data

Profile = mkgibertjr

Manifest = file-manifest-cortexonly.json

Download-Path = ./RNASEQ-PROCESSED/gtex_analysis/cortex/reindex/

```
GTexManifest <- fromJSON("file-manifest-fq.json")

NeedGTex <- read.table("GTex_Cortex.txt", header = TRUE)

# GTexManifest <- GTexManifest %>% mutate(SAMPID = str_remove(file_name,
#'.Aligned.sortedByCoord.out.patched.md.bam')) %>%
# left_join(NeedGTex, by='SAMPID')

GTexManifest <- GTexManifest %>%
    mutate(SAMPID2 = str_remove(file_name, ".m6A.1.fastq.gz"), SAMPID3 = str_remove(SAMPID2,
        ".m6A.2.fastq.gz"), SAMPID = str_remove(SAMPID3, ".fastq.gz")) %>%
    dplyr::select(-SAMPID2, -SAMPID3) %>%
    left_join(NeedGTex, by = "SAMPID")
```

```
GTexManifest <- GTexManifest[complete.cases(GTexManifest), ]

GTexManifest <- GTexManifest %>%
    select(-SAMPID, -SMTSD)

write_json(GTexManifest, "file-manifest-cortexonly.json", pretty = TRUE)
```

```
echo 'export PATH=$PATH:~/.gen3' >> ~/.bash_profile

source ~/.bash_profile

./gen3-client configure --profile=mkgibertjr --cred=credentials.json
--apiendpoint=https://gen3.theanvil.io

yes | ./gen3-client download-multiple --profile=mkgibertjr
--manifest=file-manifest-cortexonly.json
--download-path=./gtex_analysis/ --protocol=s3
```

GTex Processing: Reindex GTex BAM files

```
# Reindex the renamed files using samtools.
for bam in
$(find mkdir RNASEQ-PROCESSED/gtex_analysis/cortex/reindex -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools index $bam
done
```

We will complete the counts after processing the TCGA data in a similar fashion.

GTex Processing: Acquiring total reads from RNA-Seq BAM files

```
mkdir RNASEQ-PROCESSED/gtex_analysis/cortex/readcounts/

for bam in
$(find RNASEQ-PROCESSED/gtex_analysis/cortex/reindex/*.bam -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools view -c -F 260 $bam > $name.readcounts.txt
mv $name.readcounts.txt ./RNASEQ-PROCESSED/gtex_analysis/cortex/readcounts/
done

echo "id,counts" > RNASEQ-PROCESSED/gtex_analysis/cortex/readcounts/cortex_seqdepth_counts.csv

for txt in $(find RNASEQ-PROCESSED/gtex_analysis/cortex/readcounts/* -name '*.txt')
do
name=$(echo $txt | awk -F ".readcounts.txt" '{print $1}')
echo $name
reads=$(head $txt)
echo "$name,$reads" >> RNASEQ-PROCESSED/gtex_analysis/cortex/readcounts/cortex_seqdepth_counts.csv
```

```
done
```

TCGA Processing: Download Files

```
echo 'export PATH=$PATH:~/.gdc-client' >> ~/.bash_profile

source ~/.bash_profile

./gdc-client download -m ./gdc_GBM_miRNA_microarray_processed_manifest.2021-09-03.txt

gdc-client download -m gdc_manifest_e24fac38d3b19f67facb74d3efa746e08b0c82c2.txt -t gdc-user-token.2015
```

TCGA Processing: Extract headers from BAM files

The raw BAM files for GBM and LGG were stored in GBM-RNASEQ-RAW and LGG-RNASEQ-RAW, respectively. Since these files are labeled with unintuitive and encrypted names, I will generate clones of each BAM file using the TCGA ID as the name instead. This will make it easier to see which files are normal brain and which are tumors at a glance. TCGA files are not named this way by default because there is patient information contained within the BAM files that can be identified using the TCGA ID. Therefore, renaming the file in secure storage is preferrable. The first step to doing this is extracting the TCGA ID from the header present in each BAM file.

In the following series of scripts, a bash script was used to extract the header from each encrypted BAM file and generate a text file containing the full header.

```
## Extract header information from GBM BAM files
# Create directory for extracted headerfiles
mkdir RNASEQ-PROCESSED/tcga_analysis/GBM/head/

# Use a for loop to cycle through BAM files and extract header before creating a
#text file containing each BAM file's header.
for bam in $(find RNASEQ-RAW/GBM/* -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools view -H $bam > $name.headerids.txt
mv $name.headerids.txt ./RNASEQ-PROCESSED/tcga_analysis/GBM/head/
done

## Do the same thing for LGG
# Create directory for extracted headerfiles
mkdir RNASEQ-PROCESSED/tcga_analysis/LGG/head/

for bam in $(find RNASEQ-RAW/LGG/* -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools view -H $bam > $name.headerids.txt
mv $name.headerids.txt ./RNASEQ-PROCESSED/tcga_analysis/LGG/head/
done
```

TCGA Processing: Extract TCGA patient barcodes from BAM headers

Once the headers have been extracted from each BAM file, the TCGA ID is then extracted from each header. A master table is generated that contains the encrypted BAM file name and the new TCGA ID based named, which will be used to generate the renamed clone files in the next step.

```
## Get new file names for BAM files using TCGA IDs from extracted header
mkdir RNASEQ-PROCESSED/tcga_analysis/GBM/ids/

mkdir RNASEQ-PROCESSED/tcga_analysis/LGG/ids/

# Initialize summary file with TCGA IDs
echo "RAWID,TCGAID" > RNASEQ-PROCESSED/tcga_analysis/GBM/ids/tcgaIDs.csv

# Extract IDs from header files
for header in $(find RNASEQ-PROCESSED/tcga_analysis/GBM/head/* -name '*headerids.txt')
do
name=$(echo $header | awk -F ".headerids.txt" '{print $1}')
name2=$(echo $name | cut -c20-)
ids1=$(awk '$1=="@RG"{print $3}' $header)
ids2=$(echo $ids1 | cut -c4-)
echo "$name2.bam,$ids2.bam"
echo "$name2.bam,$ids2.bam" > $name.tcgaid.csv
echo "$name2.bam,$ids2.bam" >> RNASEQ-PROCESSED/tcga_analysis/GBM/ids/tcgaIDs.csv
mv $name.tcgaid.csv ./RNASEQ-PROCESSED/tcga_analysis/GBM/ids/
done

## Do the same thing for LGG

# Initialize summary file with TCGA IDs
echo "RAWID,TCGAID" > RNASEQ-PROCESSED/tcga_analysis/LGG/ids/tcgaIDs.csv

# Extract IDs from header files
for header in $(find RNASEQ-PROCESSED/tcga_analysis/LGG/head/* -name '*headerids.txt')
do
name=$(echo $header | awk -F ".headerids.txt" '{print $1}')
name2=$(echo $name | cut -c24-)
ids1=$(awk '$1=="@RG"{print $3}' $header)
ids2=$(echo $ids1 | cut -c4-)
echo "$name2.bam,$ids2.bam"
echo "$name2.bam,$ids2.bam" > $name.tcgaid.csv
echo "$name2.bam,$ids2.bam" >> RNASEQ-PROCESSED/tcga_analysis/LGG/ids/tcgaIDs.csv
mv $name.tcgaid.csv ./RNASEQ-PROCESSED/tcga_analysis/LGG/ids/
done
```

TCGA Processing: Reindex TCGA BAM files under new names

```
##Rename BAM files using TCGA ids

# Create folder for new files
mkdir RNASEQ-PROCESSED/tcga_analysis/GBM/reindex/

# Copy BAM files into new directory with updated names
while IFS=, read orig target; do
```

```
orig2=$(find RNASEQ-RAW/GBM/* -name $orig)
echo $orig2
cp $orig2 $target
mv $target RNASEQ-PROCESSED/tcga_analysis/GBM/reindex/
done < RNASEQ-PROCESSED/tcga_analysis/GBM/ids/tcgaIDs.csv

# Reindex the renamed files using samtools.
for bam in $(find RNASEQ-PROCESSED/tcga_analysis/GBM/reindex/ -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools index $bam
done

## Do the same thing for LGG
# Create folder for new files
mkdir RNASEQ-PROCESSED/tcga_analysis/LGG/reindex

# Copy BAM files into new directory with updated names
while IFS=, read orig target; do
orig2=$(find RNASEQ-RAW/LGG/* -name $orig)
echo $orig2
cp $orig2 $target
mv $target RNASEQ-PROCESSED/tcga_analysis/LGG/reindex/
done < RNASEQ-PROCESSED/tcga_analysis/LGG/ids/tcgaIDs.csv

# Reindex the renamed files using samtools.
for bam in $(find RNASEQ-PROCESSED/tcga_analysis/LGG/reindex/ -name '*.bam')
do
samtools view -H TCGA-CS-4938-01B-11R-1896-07.bam > header
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools reheader header $bam > $name.rehead.bam
samtools index $name.rehead.bam
done
```

TCGA Processing: Acquiring total reads from RNA-Seq BAM files

```
mkdir RNASEQ-PROCESSED/tcga_analysis/GBM/readcounts/

for bam in $(find RNASEQ-PROCESSED/tcga_analysis/GBM/reindex/*.bam -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools view -c -F 260 $bam > $name.readcounts.txt
mv $name.readcounts.txt ./RNASEQ-PROCESSED/tcga_analysis/GBM/readcounts/
done

echo "id,counts" > RNASEQ-PROCESSED/tcga_analysis/GBM/readcounts/GBM_seqdepth_counts.csv

for txt in $(find RNASEQ-PROCESSED/tcga_analysis/GBM/readcounts/* -name '*.txt')
do
name=$(echo $txt | awk -F ".readcounts.txt" '{print $1}')
```

```
echo $name
reads=$(head $txt)
echo "$name,$reads" >> RNASEQ-PROCESSED/tcga_analysis/GBM/readcounts/GBM_seqdepth_counts.csv
done

## Do the same for LGG

mkdir RNASEQ-PROCESSED/tcga_analysis/LGG/readcounts/

for bam in $(find RNASEQ-PROCESSED/tcga_analysis/LGG/reindex/*.bam -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools view -c -F 260 $bam > $name.readcounts.txt
mv $name.readcounts.txt ./RNASEQ-PROCESSED/tcga_analysis/LGG/readcounts/
done

echo "id,counts" > RNASEQ-PROCESSED/tcga_analysis/LGG/readcounts/LGG_seqdepth_counts.csv

for txt in $(find RNASEQ-PROCESSED/tcga_analysis/LGG/readcounts/* -name '*.txt')
do
name=$(echo $txt | awk -F ".readcounts.txt" '{print $1}')
echo $name
reads=$(head $txt)
echo "$name,$reads" >> RNASEQ-PROCESSED/tcga_analysis/LGG/readcounts/LGG_seqdepth_counts.csv
done
```

Example of parallelized slurm script to process multiple inputs

```
#!/bin/bash
# SBATCH --nodes=1  --ntasks-per-node=1 --cpus-per-task=1
# SBATCH --job-name array_cortex_process
# SBATCH --time=7-0:00:00       #amount of time for the whole job
# SBATCH --partition=standard   #the queue/partition to run on
# SBATCH --account=abounaderlab     #the account/allocation to use


echo "All jobs in this array have:"
echo "- SLURM_ARRAY_JOB_ID=${SLURM_ARRAY_JOB_ID}"
echo "- SLURM_ARRAY_TASK_COUNT=${SLURM_ARRAY_TASK_COUNT}"
echo "- SLURM_ARRAY_TASK_MIN=${SLURM_ARRAY_TASK_MIN}"
echo "- SLURM_ARRAY_TASK_MAX=${SLURM_ARRAY_TASK_MAX}"

echo "This job in the array has:"
echo "- SLURM_JOB_ID=${SLURM_JOB_ID}"
echo "- SLURM_ARRAY_TASK_ID=${SLURM_ARRAY_TASK_ID}"

# select our filename
N=${SLURM_ARRAY_TASK_ID}
# Comment one of the following two lines, depending on if the filenames have
# leading zeros
FILENAME=./BEDFiles/chr${N}_genes.bed # without leading zeros
CHROM=chr${N}
```

```
echo "My input file is ${FILENAME}"

module load gcc/9.2.0
module load bedtools/2.29.2
module load samtools/1.12

# Making directory
mkdir RNASEQ-PROCESSED/gtex_analysis/cortex/foldchange2

echo -e chrom"\t"start"\t"end"\t"strand"\t"id"\t"tag"\t"annot RNASEQ-PROCESSED/gtex_analysis/GBM/reinde

multiBamCov -bams RNASEQ-PROCESSED/gtex_analysis/cortex/reindex/*bam -bed ${FILENAME} -q 10 >> RNASEQ-P
```

Move BED Files to RNASEQ-PROCESSED Directory

```
mkdir RNASEQ-PROCESSED/BEDFiles
```

GTex Processing: Generating count tables for survival and differential expression.

```
# Making directory
mkdir RNASEQ-PROCESSED/gtex_analysis/cortex/foldchange

for bed in $(find RNASEQ-PROCESSED/BEDFiles/ -name '*.bed')
do

name=$(echo $bed | awk -F ".bed" '{print $1}')
name2=$(basename -- $name)
echo $name2

if test -f $name2.counts.txt; then

echo "$name2.counts.txt exists."

else

echo -e chrom"\t"start"\t"end"\t"strand"\t"id"\t"tag"\t"annot RNASEQ-PROCESSED/gtex_analysis/cortex/rei

multiBamCov -bams RNASEQ-PROCESSED/gtex_analysis/cortex/reindex/*bam -bed $bed -q 10 >> RNASEQ-PROCESSE

fi
done

head -1 RNASEQ-PROCESSED/gtex_analysis/cortex/foldchange/chr10_genes.counts.txt > RNASEQ-PROCESSED/gtex
```

TCGA Processing: Generating count tables for survival and differential expression.

```
# Making directory
mkdir RNASEQ-PROCESSED/tcga_analysis/GBM/foldchange

for bed in $(find RNASEQ-PROCESSED/BEDFiles/ -name '*.bed')
do
```

```
name=$(echo $bed | awk -F ".bed" '{print $1}')
name2=$(basename -- $name)
echo $name2

if test -f $name2.counts.txt; then

echo "$name2.counts.txt exists."

else

echo -e chrom"\t"start"\t"end"\t"strand"\t"id"\t"tag"\t"annot RNASEQ-PROCESSED/tcga_analysis/GBM/reinde

multiBamCov -bams RNASEQ-PROCESSED/tcga_analysis/GBM/reindex/*bam -bed $bed -q 10 >> RNASEQ-PROCESSED/t

fi
done

head -1 RNASEQ-PROCESSED/tcga_analysis/GBM/foldchange/chr10_genes.counts.txt > RNASEQ-PROCESSED/tcga_ana

 ###NEEWWWWWWWW
head -1 RNASEQ-PROCESSED/gtex_analysis/cortex/foldchange2/cortex_chr10.counts.txt > RNASEQ-PROCESSED/gt

# Making directory
mkdir RNASEQ-PROCESSED/tcga_analysis/LGG/foldchange

for bed in $(find RNASEQ-PROCESSED/BEDFiles/ -name '*.bed')
do

name=$(echo $bed | awk -F ".bed" '{print $1}')
name2=$(basename -- $name)
echo $name2

if test -f $name2.counts.txt; then

echo "$name2.counts.txt exists."

else

echo -e chrom"\t"start"\t"end"\t"strand"\t"id"\t"tag"\t"annot RNASEQ-PROCESSED/tcga_analysis/LGG/reinde

multiBamCov -bams RNASEQ-PROCESSED/tcga_analysis/LGG/reindex/*bam -bed $bed -q 10 >> RNASEQ-PROCESSED/t

fi
done

head -1 RNASEQ-PROCESSED/tcga_analysis/LGG/foldchange/LGG_chr10_genes.counts.txt > RNASEQ-PROCESSED/tcg
```

Determining sequencing depth for TCGA files

```
mkdir tcga_analysis/readcounts/

for bam in $(find tcga_analysis/reindex/*.bam -name '*.bam')
do
```

```
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools view -c -F 260 $bam > $name.readcounts.txt
mv $name.readcounts.txt ./tcga_analysis/readcounts/
done

##Do the same for lgg

mkdir lgg_tcga_analysis/readcounts/

for bam in $(find lgg_tcga_analysis/reindex/*.bam -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
echo $name
samtools view -c -F 260 $bam > $name.readcounts.txt
mv $name.readcounts.txt ./lgg_tcga_analysis/readcounts/
done
```

De Novo Transcript Reassembly example using stringtie

```
module load gcc/7.1.0
module load stringtie/2.0.6

for bam in $(find LGG-RNASEQ-RAW/* -name '*.bam')
do
name=$(echo $bam | awk -F ".bam" '{print $1}')
name2=$(basename "$name")
if [ ! -f lgg_stringtie/$name2.gtf ]
then
    echo "$name2.gtf does not exist"
    stringtie $bam -o $name.gtf
    mv $name.gtf ./lgg_stringtie
else
    echo "$name2.gtf exists... skipping"
fi

if [ ! -f $name.gtf ]; then
  stringtie $bam -o $name.gtf
fi
mv $name.gtf ./lgg_stringtie
done

cd lgg_stringtie

stringtie --merge *rehead.gtf -G Homo_sapiens.GRCh38.99.gtf -o stringtie_merged.gtf

intersectBed -a hg38.ultraConserved.bed -b CHESSgenes.bed -wa -wb > intersectchess_TUCRs.bed

intersectBed -a hg38.ultraConserved.bed -b stringtie_merged.gtf -wa -wb > intersectstringtie_TUCRs.bed

cp stringtie_merged.gtf stringtie_merged.bed
```

TUCR Expression, Deregulation, and Survival Analyses [33, 34, 54, 55]

TUCR expression, deregulation, and survival analyses, were analyzed using processed TCGA and GTEx RNA-Seq data and a workflow using R/RStudio. Detailed methods can be found as a codebook provided in the supplementary materials and online at: github.com/abounaderlab/tucr_project.

```
wget http://gdac.broadinstitute.org/runs/stddata__2016_01_28/data/GBM/20160128/gdac.broadi
nstitute.org_GBM.Merge_Clinical.Level_1.2016012800.0.0.tar.gz

tar -xvzf gdac.broadinstitute.org_GBM.Merge_Clinical.Level_1.2016012800.0.0.tar.gz

mv gdac.broadinstitute.org_GBM.Merge_Clinical.Level_1.2016012800.0.0 GBM.Merge_Clinical
```

Acquiring and parsing clinical survival data

TUCR weighted gene correlation network analysis (WGCNA) [36]

TUCR WGCNA was performed using processed TCGA and GTEx RNA-Seq data using a modified version of the R/RStudio workflow designed by Drs. Peter Langfelder and Steve Horvath at UC Los Angeles. Detailed methods can be found as a codebook provided in the supplementary materials and online at: github.com/abounaderlab/tucr_project

De novo transcript reassembly and validation [35]

De novo transcript assembly was performed on TCGA GBM and LGG RNA-Seq data using standard protocols and the stringtie bioinformatics package. Results were validated using PCR:
10 min at 95oC, followed by 40 cycles of 10 seconds at 95oC and 1 minute at 60oC. Detailed methods can be found as a codebook provided in the supplementary materials and online at: github.com/abounaderlab/tucr_project

Patient Samples

GBM Tumor samples were acquired from the UVA Tumor Bank. Detailed patient information can be found as a supplement (Supplementary Table 2).

Cell Lines and stem cells

U87, U251, A172, and T98G glioblastoma cell lines were used in in vitro experiments and were acquired from ATCC. U87 cells were cultured in 500 mL minimum essential media (MEM) Earles (Gibco, #.11095-080) containing 5 mL penicillin/streptomycin (pen/strep, Gibco, Cat #.15140-133), 5 mL MEM non-essential amino acids (NEAA, Gibco, #.11140-050), 5 mL sodium pyruvate (Gibco, 100 nM, #.11360-070), 10 mL sodium bicarbonate (Gibco, 7.5%, #.25080-094), and 50 mL fetal bovine serum (FBS). T98G cells were cultured in 500 mL MEM Earles media containing 5 mL pen/strep, 5 mL NEAA, 5 mL sodium pyruvate, and 50 mL FBS. A172 cells were cultured in 500 mL Dulbecco's modified eagle media (DMEM, Gibco, #.11965-092) containing 5 mL pen/strep, and 50 mL FBS. U251 cells were cultured in 500 mL RPMI L-Glutamine media (Gibco, #.11875093) containing 5 mL pen/strep and 25 mL FBS. GSC-34 and GSC-28 glioblastoma stem cells were cultured in neurobasal (L-glutamine negative) media (Gibco, #.21103-049) containing 5 mL pen/strep, 5 mL B-27 (without Vit-A, Gibco, #.12587-010), 2.5 mL N-2 (Gibco, #.17502-048), 1 mL EGF, 1 mL FGF, and 1.25 mL L-Glutamine. All cell media contained in 5 µL Plasmocure reagent to prevent mycoplasma contamination.

Primer and Oligo Design

Primers and siRNAs were designed using the Primer3 and Thermofisher design portals, respectively. uc.110 forward primer sequence is 5'-CAGCCAAAGGGGAAGTGTAT-3', and the reverse sequence is 5'-CCGTCCTCCCTGCACTAAAT-3'.
MFRP forward primer sequence is 5'- GCATCTATTCATGTGGCAGGC-3', and the reverse sequence is 5'-TACTCCGGACCCTCCAGTTG-3'.
The miR-544 precursor was ordered from Invitrogen (#.AM17100). Negative control oligos were ordered from Ambion (#.AM4635).

uc.110 stable overexpression

The full uc.110 transcript from "de novo transcript reassembly and validation" was cloned into the pCDH-EF1-MCS-BGH-PGK-GFP-T2A-Puro vector (# CD550A-1) using stbl3 competent e.coli cells and ampicillin selection. Amplified vector was extracted using the miniprep kit (Qiagen, #.27106). 0.75 µg of this vector, 0.75 µg of psPAX2 lentiviral gag-pol packaging vector, and 0.5 µg of pMD.2G VSV-G enveloping protein was transfected in 6 µL X-tremeGENE transfection reagent (#.06366236001) into 293T cells per manufacturer instructions to generate a lentivirus that was transduced to U87, U251, and A172 cells in media without antibiotics. These cells were subjected to antibody (puromycin) selection for uc.110-positive cells at D3.

uc.110 quantitative (q)PCR

Total RNA was isolated using the RNEasy+ kit (Qiagen, #.74134) according to manufacturer instructions. RNA concentration and purity was measured via nanodrop. 800 ng of cDNA was synthesized (BIORAD T100 Thermal Cycler) using the iScript (BIORAD, #. 1708890) synthesis kit per manufacturer instructions. A 20 µL reaction mixture was then created for each condition with the following concentrations: 1 µL of combined forward/reverse primers (5 µM), 10 µL of iQ SYBR Green master mix (#1798880), 4 µL of nuclease free water, and 5 µL of synthesized cDNA. These reactions were cycled (BIORAD CFX Real Time System) in 96-well plates: 10 min at 95oC, followed by 40 cycles of 10 seconds at 95oC and 1 minute at 60oC.

Cell Counting (Accumulation) Assay [37-39, 44]

Cells were seeded in 6-well culture plates with full serum media at 30,000/well density at D-1. At D0, each well was transfected via master mix 3 µL of siRNAs (20 µM) via 9 µL Lipofectamine 2000 (Invitrogen, #.11668-019) in 300 µL OPTI-MEM (Gibco, #.31985-070) and 700 µL antibiotic and empty media for 6 hours. At 6 hours, media were replaced with fresh media containing antibiotics and FBS. Cells were then counted via haemocytometer at Days 1, 3, 5, and 7 for each cell line.

Transwell Cell Invasion/Migration Assay [42-44]

Cells were seeded in 6-well culture plates with full serum media at 300k/well density at D-1. At D0, each well was transfected via master mix 3 µL of siRNAs (20 µM) via 9 µL Lipofectamine 2000 in 300 µL OPTI-MEM and 700 µL antibiotic and empty media for 6 hours. At 6 hours, the media were replaced with fresh media containing antibiotics and FBS. The cells were then seeded in empty media at 200k/chamber into Transwell Invasion Chambers coated with Collagen IV.
After 8 hours, non-invading cells were cleared and invading cells were stained with Crystal Violet.

AlamarBlue Cell Viability Assay [40-41]

Cells were seeded in 96-well culture plates with full serum media at 10k/well density at D-1. Border wells were filled with media to account for edge effects. At D0, each well was transfected via master mix 1 µL of siRNAs (20 µM) via 3 µL Lipofectamine 2000 in 30 µL OPTI-MEM and 70 µL antibiotic and empty media for 6 hours. At 6 hours, media were replaced with fresh media containing antibiotics and FBS. Functional assays were performed using the AlamarBlue kit (Life Technologies #. A50100) per manufacturer instructions. Reactions were allowed to proceed for 1 hour.

Ex vivo knockdown of uc.110

Cells were seeded in 6-well culture plates with full serum media at 300k/well density at D-1. At D0, each well was transfected with 3 µL of siRNAs (20 µM) via 9 µL Lipofectamine 2000 in 300 µL OPTI-MEM and 700 µL antibiotic and empty media for 6 hours. At 6 hours, the media were replaced with fresh media containing antibiotics and FBS. Mouse experiments were performed using xenograft models and intracranial injections of U251 cells post transfection with siRNA oligonucleotides. Cells were injected at D2 and were imaged at two-week intervals via MRI.
Survival was assessed sured daily and tumor volume was measured at the end of life.

Characterization of transcriptome post-uc.110 knockdown

Cells were seeded in 6-well culture plates with full serum media at 300k/well density at D1. At D0, each well was transfected with 3 µL of siRNAs (20 µM) using 9 µL Lipofectamine 2000 in 300 µL OPTI-MEM and 700 µL antibiotic and empty media for 6 hours. At 6 hours, the media were replaced with fresh media containing antibiotics and FBS. RNA Libraries were collected and sequenced via RNA-Seq on Day 2 (post transfection).

Luciferase Reporter Vector Construction

The Luciferase reporter vector were constructed via insertion of uc.110 conserved region and 3'UTR of MFRP downstream of Renilla luciferase stop codon in psi-CHECK2 dual luciferase vectors (Promega, Madison, WI, USA). The insertions were validated by sequencing. Uc.110 and MFRP primer pairs with XholI and NotI sequence at 5' and 3' respectively, uc.110-FW: 5'- ATATATctcgagCGAGGTGAGAACCAGAGTGT-3', uc.110-RW: 5'- AATAATgcggccgcTTGGCTGCCTAATGAGTCACA-3', MFRP-FW: 5'- ATATATctcgagAAATGGGGTCTGGTCCT
3' and MFRP-RW: 5'- AATAATgcggccgcTCGCCTTTCTCTCCCGGA-3' were used for PCR amplification. Site-directed mutagenesis of predicted miR-544 target sites for both uc.110 and MFRP were performed to generate mutant vectors.

3'UTR Reporter Assays

To determine whether miR-544 directly binds to the MFRP 3'UTR and uc.110, cells were transfected with miR-544 or miR-scr (control) for 24 hour. The cells were then transfected with luciferase reporter control or 3'UTR-MFRP or uc.110 as well as corresponsive mutant vectors for 24 hours. Luciferase assays wered performed using the Luciferase System Kit (Promega) and luminescence was measured. Renilla luciferase activity was double normalized by dividing each well first by firefly activity and then by average luciferase/firefly value in a parallel set done with constitutive luciferase plasmid.

TCF/LEF reporter Assays

Cells were seeded in 6-well culture plates with full serum media at 300k/well density at D-1. At D0, each well was transfected with 3 µL of siRNA/miRNA (20 µM) using 9 µL Lipofectamine 2000 in 300 µL OPTI-MEM and 700 µL antibiotic and empty media for 6 hours. At 6 hours, the media were replaced with fresh media containing antibiotics and FBS. MFRP and uc.110 sequences were cloned into the PROMEGA pmirGLO Luciferase vector (E1330). BPS Dual reporter luciferase assays were ordered for TCF/LEF (#.60500) and uc.110/MFRP (#.60683) experiments.

In Vivo Tumor Formation

Adult male and female Nude: Hsd:Athymic Nude-Foxn1 mice were purchased from Harlan. All the animal work was conducted at the Animal Research Core Facility at the University of Virginia School of Medicine in accordance with the institutional guidelines. Mice used for this study were anesthetized with ketamine (17.4 mg/20g), xylazine (2.6 mg/20g) and placed on a sterotactic frame. Tumor xenografts were generated by implantation of U251 cells transfected with si-uc.110-1, si-cu.110-2 or si-Scr. U251 cells (3x105 cells; n=5) were stereotactically implanted into mice in their right striata at the coordinates from the bregma 1mm

anterior, 1.5 mm lateral and 2.5 mm intraparenchymal. Three weeks after tumor implantation, the animals were subjected to brain MRI. To measure the tumor size, 20 ul of gadopentetate dimeglumine (Magnevist, Bayer Healthcare) was intraperitoneally injected 15 minutes before scanning. Tumor volumes were measured using MicroDicom.

Statistical Analyses

Comparisons between means of samples were performed using Student's t-test and one-way ANOVAs. Comparisons between categorical variables were performed using chi-squared and Fisher's exact test. Comparisons were considered statistically significant if the p-value was less than 0.05. Multiple hypothesis correction using the Bonferroni method was performed, converting raw p-values to FDR, unless otherwise stated. Molecular experiment tests were performed in SigmaPlot 14.0, while computational experiment tests were performed using bedtools and/or RStudio. Detailed methods can be found as a codebook provided in the supplementary materials and online at: github.com/abounaderlab/tucr_project.

AUTHOR CONTRIBUTIONS

Author contributions are defined using Elsevier's CRediT format: Myron Gibert Jr: Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Writing, Visualization, Supervision, Project Administration, Funding Acquisition Ying Zhang: Methodology, Investigation, Resources, Formal Analysis, Data Curation, Review and Editing Shekhar Saha: Methodology, Investigation, Resources, Formal Analysis, Data Curation, Review and Editing Pawel Marcinkiewicz: Investigation, Resources, Formal Analysis, Data Curation, Review and Editing Sylwia Bednarek: Methodology, Validation, Formal Analysis, Investigation, Review and Editing, Visualization Collin Dube: Methodology, Writing, Visualization, Review and Editing Kadie Hudson: Methodology, Writing, Visualization, Review and Editing Yunan Sun: Methodology, Writing, Visualization, Review and Editing Bilhan Chagari: Formal Analysis, Investigation, Writing, Data Curation Aditya Sarkar: Formal Analysis, Investigation, Writing, Data Curation Christian Roig-Laboy: Formal Analysis, Investigation, Writing, Data Curation Natalie Neace: Formal Analysis, Investigation, Writing, Data Curation Karim Saoud: Formal Analysis, Investigation, Writing, Data Curation Initha Setiady: Formal Analysis, Investigation, Writing, Data Curation Farina Hanif: Methodology, Investigation, Review and Editing David Schiff: Tumor tissue contribution, Review and Editing Pankaj Kumar: Methodology, Software, Validation, Formal Analysis, Resources, Data Curation Benjamin Kefas: Methodology, Investigation, Review and Editing Markus Hafner: Conceptualization and editing Roger Abounader: Conceptualization, Methodology, Resources, Review and Editing, Supervision, Project Administration, Funding Acquisition

ACKNOWLEDGMENTS

This article was supported by NIH UO1 CA220841, NINDS 1R21NS122136-01, NINDS RO1 NS122222, NCI Cancer Center Support Grant 5P30CA044579, and a Schiff Foundation grant (all to R.A.). The work was also supported by the UVA Cancer Center Bioinformatics Core and the Molecular Imaging Core. We thank the dbGAP and the TCGA data management teams for providing access to raw RNAseq data. We would also like to express our gratitude to the patients for their participation in TCGA. Special thanks goes to Drs. Stephen Turner and Alex Koeppel for their work with the Bioinformatics Core and early contributions to some of the computational studies.

REFERENCES

1. Bejerano, G., et al., Ultraconserved elements in the human genome. Science, 2004. 304(5675): p. 1321-5.
2. Gibert, M.K., Jr., et al., Transcribed Ultraconserved Regions in Cancer. Cells, 2022. 11(10).
3. Calin, G.A. and C.M. Croce, Chronic lymphocytic leukemia: interplay between noncoding RNAs and protein-coding genes. Blood, 2009. 114(23): p. 4761-70.

4. Calin, G.A., et al., Ultraconserved regions encoding ncRNAs are altered in human leukemias and carcinomas. Cancer Cell, 2007. 12(3): p. 215-29.
5. Colwell, M., et al., Evolutionary conservation of DNA methylation in CpG sites within ultraconserved noncoding elements. Epigenetics, 2018. 13(1): p. 49-60.
6. Edwards, J.K., et al., MicroRNAs and ultraconserved genes as diagnostic markers and therapeutic targets in cancer and cardiovascular diseases. J Cardiovasc Transl Res, 2010. 3(3): p. 271-9.
7. Fabris, L. and G.A. Calin, Understanding the Genomic Ultraconservations: T-UCRs and Cancer. Int Rev Cell Mol Biol, 2017. 333: p. 159-172.
8. Lujambio, A., et al., CpG island hypermethylation-associated silencing of non-coding RNAs transcribed from ultraconserved regions in human cancer. Oncogene, 2010. 29(48): p. 6390-401.
9. McCole, R.B., et al., Abnormal dosage of ultraconserved elements is highly disfavored in healthy cells but not cancer cells. PLoS Genet, 2014. 10(10): p. e1004646.
10. Scaruffi, P., The transcribed-ultraconserved regions: a novel class of long noncoding RNAs involved in cancer susceptibility. ScientificWorldJournal, 2011. 11: p. 340-52.
11. Zambalde, E.P., et al., Transcribed Ultraconserved Regions Are Associated with Clinicopathological Features in Breast Cancer. Biomolecules, 2022. 12(2).
12. Braicu, C., et al., The Function of Non-Coding RNAs in Lung Cancer Tumorigenesis. Cancers (Basel), 2019. 11(5).
13. Catana, C.S., et al., Non-coding RNAs, the Trojan horse in two-way communication between tumor and stroma in colorectal and hepatocellular carcinoma. Oncotarget, 2017. 8(17): p. 29519-29534.
14. Irimie, A.I., et al., The Unforeseen Non-Coding RNAs in Head and Neck Cancer. Genes (Basel), 2018. 9(3).
15. Ling, H., et al., Junk DNA and the long non-coding RNA twist in cancer genetics. Oncogene, 2015. 34(39): p. 5003-11.
16. Liz, J., et al., Regulation of pri-miRNA processing by a long noncoding RNA transcribed from an ultraconserved region. Mol Cell, 2014. 55(1): p. 138-47.
17. Sakamoto, N., et al., Non-coding RNAs are promising targets for stem cell-based cancer therapy. Noncoding RNA Res, 2017. 2(2): p. 83-87.
18. Kiran, M., et al., A Prognostic Signature for Lower Grade Gliomas Based on Expression of Long Non-Coding RNAs. Mol Neurobiol, 2019. 56(7): p. 4786-4798.
19. Dey, B.K., A.C. Mueller, and A. Dutta, Long non-coding RNAs as emerging regulators of differentiation, development, and disease. Transcription, 2014. 5(4): p. e944014.
20. Cruickshanks, N., et al., Role and Therapeutic Targeting of the HGF/MET Pathway in Glioblastoma. Cancers (Basel), 2017. 9(7).
21. Davis, M.E., Glioblastoma: Overview of Disease and Treatment. Clin J Oncol Nurs, 2016. 20(5 Suppl): p. S2-8.
22. Lathia, J.D., et al., Cancer stem cells in glioblastoma. Genes Dev, 2015. 29(12): p. 1203-
23.
24. Kalkan, R., Glioblastoma Stem Cells as a New Therapeutic Target for Glioblastoma. Clin Med Insights Oncol, 2015. 9: p. 95-103.
25. Thakkar, J.P., et al., Epidemiologic and molecular prognostic review of glioblastoma. Cancer Epidemiol Biomarkers Prev, 2014. 23(10): p. 1985-96.
26. Mao, H., et al., Deregulated signaling pathways in glioblastoma multiforme: molecular mechanisms and therapeutic targets. Cancer Invest, 2012. 30(1): p. 48-56.
27. Altaner, C., Glioblastoma and stem cells. Neoplasma, 2008. 55(5): p. 369-74.
28. Mourad, P.D., et al., Why are systemic glioblastoma metastases rare? Systemic and cerebral growth of mouse glioblastoma. Surg Neurol, 2005. 63(6): p. 511-9; discussion 519.
29. Zhang, Y., et al., The p53 Pathway in Glioblastoma. Cancers (Basel), 2018. 10(9).
30. Raney, B.J., et al., Track data hubs enable visualization of user-defined genome-wide annotations on the UCSC Genome Browser. Bioinformatics, 2014. 30(7): p. 1003-5.
31. Kent, W.J., et al., The human genome browser at UCSC. Genome Res, 2002. 12(6): p. 996-1006.
32. Quinlan, A.R., BEDTools: The Swiss-Army Tool for Genome Feature Analysis. Curr Protoc Bioinformatics, 2014. 47: p. 11 12 1-34.
33. Quinlan, A.R. and I.M. Hall, BEDTools: a flexible suite of utilities for comparing genomic features.

Bioinformatics, 2010. 26(6): p. 841-2.
34. Cancer Genome Atlas Research, N., et al., The Cancer Genome Atlas Pan-Cancer analysis project. Nat Genet, 2013. 45(10): p. 1113-20.
35. Stanfill, A.G. and X. Cao, Enhancing Research Through the Use of the Genotype-Tissue Expression (GTEx) Database. Biol Res Nurs, 2021. 23(3): p. 533-540.
36. Pertea, M., et al., CHESS: a new human gene catalog curated from thousands of largescale RNA sequencing experiments reveals extensive transcriptional noise. Genome Biol, 2018. 19(1): p. 208.
37. Langfelder, P. and S. Horvath, WGCNA: an R package for weighted correlation network analysis. BMC Bioinformatics, 2008. 9: p. 559.
38. Hanif, F., et al., miR-3174 Is a New Tumor Suppressor MicroRNA That Inhibits Several Tumor-Promoting Genes in Glioblastoma. Int J Mol Sci, 2023. 24(11).
39. Mulcahy, E.Q.X., et al., MicroRNA 3928 Suppresses Glioblastoma through Downregulation of Several Oncogenes and Upregulation of p53. Int J Mol Sci, 2022. 23(7).
40. Cruickshanks, N., et al., Discovery and Therapeutic Exploitation of Mechanisms of Resistance to MET Inhibitors in Glioblastoma. Clin Cancer Res, 2019. 25(2): p. 663-673.
41. Kumar, P., A. Nagarajan, and P.D. Uchil, Analysis of Cell Viability by the alamarBlue Assay. Cold Spring Harb Protoc, 2018. 2018(6).
42. Voytik-Harbin, S.L., et al., Application and evaluation of the alamarBlue assay for cell growth and survival of fibroblasts. In Vitro Cell Dev Biol Anim, 1998. 34(3): p. 239-46.
43. Stoellinger, H.M. and A.R. Alexanian, Modifications to the Transwell Migration/Invasion Assay Method That Eases Assay Performance and Improves the Accuracy. Assay Drug Dev Technol, 2022. 20(2): p. 75-82.
44. Liu, X. and X. Wu, Utilizing Matrigel Transwell Invasion Assay to Detect and Enumerate Circulating Tumor Cells. Methods Mol Biol, 2017. 1634: p. 277-282.
45. Justus, C.R., et al., In vitro cell migration and invasion assays. J Vis Exp, 2014(88).
46. Guessous, F., et al., Cooperation between c-Met and Focal Adhesion Kinase Family Members in Medulloblastoma and Implications for Therapy. Mol Cancer Ther, 2012. 11(2): p. 288-97.
47. Sonoda, Y., et al., Akt pathway activation converts anaplastic astrocytoma to glioblastoma multiforme in a human astrocyte model of glioma. Cancer Res, 2001. 61(18): p. 6674-8.
48. Won, J., et al., Membrane frizzled-related protein is necessary for the normal development and maintenance of photoreceptor outer segments. Vis Neurosci, 2008. 25(4): p. 563-74.
49. Katoh, M., Molecular cloning and characterization of MFRP, a novel gene encoding a membrane-type Frizzled-related protein. Biochem Biophys Res Commun, 2001. 282(1): p. 116-23.
50. Slaby, O., R. Laga, and O. Sedlacek, Therapeutic targeting of non-coding RNAs in cancer. Biochem J, 2017. 474(24): p. 4219-4251.
51. Saleembhasha, A. and S. Mishra, Long non-coding RNAs as pan-cancer master gene regulators of associated protein-coding genes: a systems biology approach. PeerJ, 2019. 7: p. e6388.
52. Li, L., et al., HOX cluster-embedded antisense long non-coding RNAs in lung cancer. Cancer Lett, 2019. 450: p. 14-21.
53. Jiang, D., et al., Long Chain Non-Coding RNA (lncRNA) HOTAIR Knockdown Increases miR-454-3p to Suppress Gastric Cancer Growth by Targeting STAT3/Cyclin D1. Med Sci Monit, 2019. 25: p. 1537-1548.
54. Soler, M., et al., The transcribed ultraconserved region uc.160+ enhances processing and A-to-I editing of the miR-376 cluster: hypermethylation improves glioma prognosis. Mol Oncol, 2022. 16(3): p. 648-664.
55. Team, R., RStudio: Integrated Development for R. 2020.
56. Team., R.D.C. R: A Language and Environment for Statistical Computing. 2012; Available from: http://www.r-project.org/.