Environ Health Perspect

DOI: 10.1289/EHP3281

Supplemental Material

Signaling Events Downstream of AHR Activation That Contribute to Toxic Responses: The Functional Role of an AHR-Dependent Long Noncoding RNA (*slincR*) Using the Zebrafish Model

Gloria R. Garcia, Prarthana Shankar, Cheryl L. Dunham, Abraham Garcia, Jane K. La Du, Lisa Truong, Susan C. Tilton, and Robert L. Tanguay

Table of Contents

Table S1. List of all primers and probes.

Additional file - Excel document and Code and data zip Document

S1. CHART Sample Preparation

Zebrafish embryo preparation for CHART adapted from Bogdanovic et al. (2013) and CHART protocol and buffers are from Simon (2013).

To ensure that all embryos are collected at the same developmental stage, mate zebrafish females and males only for 15 min, collect the embryos in Petri dishes with embryo medium (EM).

Day 1 Embryo collection and exposure

Required Materials:

- 700-1200 embryos for screening
- 20-mL Amber vials (1 embryo/100 µL)
- 1 ng/mL TCDD and 0.1% DMSO solutions
- EM [15 mM NaCl, 0.5 mM KCl, 1 mM $MgSO_4$, 0.15 mM KH_2PO_4, 0.05 mM Na_2HPO_4 and 0.7 mM $NaHCO_3$ (Westerfield 2000)] for rinsing off TCDD
- Large Petri dishes for grow out (150-200 embryos/dish)

Method:

1. Sort and stage embryos in the morning.
2. Distribute embryos into amber vials and keep in incubator to develop.
3. At 6 hpf, remove EM and replace with 100 µL/embryo DMSO/TCDD solution.
4. Place vials on rocker for a 1 hour exposure time. Invert vials 4-6X every 15 min.
5. Wash 3 times with EM and move to large Petri dishes overnight in incubator.

****

Day 2 Dechorionation

Required Materials:

- 100-mm glass Petri dish for dechorionating
- Pronase aliquot (31.77 ug/µL)
- Clean large glass Petri dishes for dechorionated embryos to grow out to 48 hpf
- EM

Method:

1. Before 24 hpf, screen embryos, removing dead and malformed.
2. Carefully, add up to 1000 embryos into 100-mm glass Petri dish.
3. Add 83 µL (31.77 µg/µL) of pronase to 25 mL EM in 50 mL conical tube and mix by inverting.
4. Remove most of the solution from the glass Petri dish with the embryos and add the 25 mL pronase solution.
5. Observe the dish under a microscope while swirling the embryos (switching between counter clock wise and clock wise rotations).
6. As soon as the first embryo breaks out of the chorion, start washing the embryo with EM (The remaining chorions will come off during wash).
    a. Use a full bottle of EM (~1000 mL) to remove remaining chorions by continually filling and emptying the glass Petri dish.
7. Remove most (but not all) of the water from the glass Petri dish. Then gently dip the dish into a large plastic Petri dish filled with EM and try to evenly distribute the embryos into 4-5 Petri dishes.
    a. Overcrowding will increase the number of malformed embryos. Over-digested fish will look smaller and less pigmented.
8. Place embryos back into incubator to continue to develop.

****

Day 3

Protocol 1: Crosslinking and Preparation of Nuclei Pellets (Simon Protocol)

Required Materials:

- Liquid nitrogen in small dewar
- 15-mL Falcon tube(s) one for each exposure group
- Thawed 4% paraformaldehyde (PFA; 4 mL per falcon tube)
- EM (4.6 mL per falcon tube)
- 1.5 M Glycine (from 4 °C storage)
- 1X Ice cold PBS for washing (~15 mL per falcon tube)
- 1.5-mL Safelock microcentrifuge tubes
- Buffered tricaine methanesulfonate (MS-222)

Method:

1. Add buffered tricaine methanesulfonate to plates to anesthetize (MS-222, 100-200 mg/L) by prolonged immersion and monitored under a dissecting microscope until the fish stop moving, but the hearts still visibly beating (approximately 5 minutes). (DO NOT EUTHANIZE).
2. Screen the embryos, removing all malformed and dead embryos.
3. Count and place the embryos into new Petri dish (~475/biological replicate).
4. Transfer the embryos with a Pasteur pipette to a 15-mL Falcon tube and remove most of EM.
5. Add buffered tricaine methanesulfonate to plates to euthanize (MS-222, 200-300 mg/L) by prolonged immersion and monitored under a dissecting microscope until the hearts visibly stop beating (approximately 15 minutes). Rinse euthanized fish with ice-cold PBS 3 times.
6. On final PBS rinse, leave 7.5 mL in 15-mL falcon tube and add 6.5 mL 4% PFA (4% PFA (SIGMA P6148), phosphate buffer 200 mM, pH 7.4, NaOH 0.02 N) to the embryos and rotate them end over end at room temperature for 15 min. Final concentration of PFA is 1.86%.
7. Add glycine (Merck, 1.00590.1000) to a final concentration of 0.125 M (1.27 mL of 1.5 M glycine; stock can be kept filtered at 4 °C) to quench formaldehyde and rotate end over end for 5 min at room temperature.
8. Remove supernatant and rinse embryos three times in ice-cold 1X PBS**.** Remove PBS and proceed with cell lysis or freeze in liquid nitrogen and store pellets at −80 °C.

Important: Use fresh 4% PFA. It can be stored in aliquots at −20 °C. Subject PFA to a single freeze-thaw cycle (i.e. dispose of all unused PFA).

Warning: PFA is a toxic substance. Use gloves, mask and protective clothes during preparation and application.

STOPPING POINT!! ***Note: Moving forward with cell lysis instead of freezing may result in better chromatin shearing and total chromatin recovery.***

Warning: Wear protective clothes and gloves when using liquid nitrogen. Careless handling results in cold burns.

Troubleshooting: If embryos keep floating and supernatant cannot be properly removed, spin for 2 min at 1500 rpm to pellet the embryos. Remove PBS completely before freezing in liquid nitrogen.

****

Protocol 1: Crosslinking and Preparation of Nuclei Pellets (Simon Protocol) continued

Required Materials:

- Liquid nitrogen in small dewar
- Ice bucket
- Bioruptor Pico sonicator
- 15-mL Falcon tube(s)
- Labeled Bioruptor tubes (Diagenode C30010011) 0.65-mL sonication microtubes with maximum shearing volume of 100 µL/sample
- Cold 10-mL Dounce Homogenizer with tight pestle “B”
- RNAse Away
- RNAZap
- DTT
- Spermine
- Spermidine
- SUPERasIN
- Complete EDTA-Free Protease Inhibitor Solution

To do before starting:

- Make sure centrifuges are at 4 °C.
- Clean work area and pipettes and wipe down with RNAse Away.
- Turn on sonicator (Diagenode Bioruptor) (needs 1 hour to chill).
- Calculate all buffer volumes needed for the day.
- Clean 10-mL Dounce homogenizer with RNAZAP and rinse with water, place in ice bucket (needs minimum 15 minutes on ice).
- Thaw DTT, spermine, spermidine, SUPERasIN aliquots on ice.
- Prepare 100X Complete EDTA-free Protease Inhibitor solution by dissolving 2 tablets in 1 mL water.
- Prepare ice cold Complete Sucrose Buffer (8 mL per sample).
- Prepare ice cold Complete Glycerol Buffer (16 mL per sample).

Complete Sucrose buffer (21 mL)

- □ 51.3 g sucrose (0.3 M final)
- □ 50 mL of 10% (v/v) Triton X-100 (1% final)
- □ 5 mL of 1 M HEPES, pH 7.5 (10 mM final)
- □ 16.7 mL of 3 M potassium acetate (100 mM final)
- □ 50 µL of 1 M EGTA (0.1 mM)
- □ Distilled, deionized H_2O to 500 mL
- □ Store up to 6 months at 4 °C

Immediately before use, take **20 mL** of cold stored sucrose solution and add:

- □ 100 µL of 0.1 M spermidine (store stock in aliquots at -80 °C; 0.5 mM final)
- □ 30 µL of 0.1 M spermine (store stock in aliquots at -80 °C; 0.15 mM final)
- □ 200 µL of 100x Complete EDTA-free Protease Inhibitor (Roche)
- □ 20 µL of 1M DTT (1 mM final)
- □ 200 U SUPERasIN (10 µL of 20 U/µL)

Complete Glycerol buffer (41 mL)

- □ 125 mL glycerol (25% final)
- □ 5 mL of 1 M HEPES (10 mM final)
- □ 1 mL of 0.5 M EDTA (1 mM final)
- □ 50 µL of 1 M EGTA (0.1 mM final)
- □ 16.7 mL of 3 M potassium acetate (100 mM final)
- □ Distilled, deionized H_2O up to 500 mL
- □ Store up to 6 months at 4 °C

Immediately before use, take **40 mL** of cold stored Glycerol buffer solution and add:

- □ 200 µL of 0.1 M spermidine (store stock in aliquots at -80 °C; 0.5 mM final)
- □ 60 µL of 0.1 M spermine (store stock in aliquots at -80 °C; 0.15 mM final)
- □ 400 µL of 100x Complete EDTA-free Protease Inhibitor (Roche)
- □ 40 µL of 1 M DTT (1 mM final)
- □ 20 µL of 20 U/µL SUPERasIN (10 U/mL final)

Complete Sonication buffer (5 mL)

- □ 500 µL of 1 M HEPES pH 7.5 (50 mM final)
- □ 150 µL of 5 M NaCl (75 mM final)
- □ 2 µL of 0.5 M EGTA (0.1 mM final)
- □ 1 mL of 5% (w/v) *N*-lauroylsarcosine (0.5% final)
- □ 100 µL of 10% (w/v) sodium deoxycholate (0.1% final)
- □ Store up to 6 months at 4 °C

Complete Nuclei Wash buffer (20 mL)

- □ 5 mL of 1 M HEPES, pH 7.5 (50 mM final)
- □ 1.5 mL of 5 M NaCl (75 mM final)
- □ 20 µL of 0.5 M EGTA (0.1 mM final)
- □ Distilled, deionized H_2O to 100 mL
  Store up to 6 months at 4 °C

Immediately before use take **5 mL** of cold stored Sonication Buffer solution and add: □ 5 µL of 20 U/µL SUPERasIN (10 U/µL) □ 25 µL of 1 M DTT (5 mM final)	**Immediately before use**, add to **20 mL** of the above: □ 20 µL of 20 U/µL SUPERasIN (1 U/mL final) □ 20 µL of 1 M dithiothreitol (1 mM final) □ 200 µL of 100x Complete EDTA-free Protease Inhibitor (Roche)

Method: All steps should be performed on ice

10. Clean Dounce homogenizer by rinsing with RNaseZAP, washing multiple times with RNase-free water and a final rinse with 2 mL sucrose buffer. Store on ice at least 15 minutes before use to chill.
11. Resuspend each pellet in 4 mL of ice cold sucrose buffer and transfer to a chilled glass Dounce homogenizer (10-mL).
12. Homogenize with a tight pestle “B” 10 times and allow samples to chill on ice for 5 minutes, then homogenize 10 more times. Again, allow the samples to chill on ice, but for 10 minutes, then homogenize a final 10 times.
13. Prepare 4 mL of glycerol buffer in a new 15-mL conical tube. Add 4 mL of glycerol buffer to the sample in the Dounce homogenizer and pipette up and down several times. Gently layer the mixed solution on top of the new glycerol buffer in the 15-mL conical tube.
14. To pellet the nuclei, centrifuge the sample for 15 min at 1000 x g, 4 °C.
15. Carefully remove the supernatant using a pipette. You want to discard the top layer with as little mixing as possible.
16. **Repeat steps 9-13 one time.**
17. Can proceed with cross-linking and sonication or freeze in liquid nitrogen and store pellets at -80 °C.

STOPPING POINT!! ***Note: Moving forward with cross-linking and sonication instead of freezing may result in better chromatin shearing and total chromatin recovery.***

****

Protocol 3: CHART Enrichment with Designed Biotin Labeled Capture Oligos

Required Materials:

- Pellet from Protocol 1
- SUPERasIN 20 U/µL
- Complete EDTA-free protease inhibitor (1 tablet dissolved in 500 µL H2O for 100x working solution)
- Liquid nitrogen
- 1M DTT
- 0.4 mM Phenylmethylsulfonyl fluoride (PMSF)
- 20 µM CHART capture oligo cocktail
- MyOne Dynabeads C1
- RNase H 5 U/µL
- End over end rotator
- 15- and 50-mL Falcon tubes
- Dynal magnets and stand
- 0.65-mL tubes (Diagenode C30010011)
- Bioruptor Pico sonicator
- 1X PBS
- 32% PFA (methanol free)
- Wash Buffer 100 and 250 [WB 100 and WB 250; recipe provided in Simon (2013)]
- Denaturant Buffer [recipe provided in Simon (2013)]
- 2x Hybridization Buffer [recipe provided in Simon (2013)]
- HEB buffer (RNase H elution buffer)

Prepare all complete solutions prior to start [Recipes found in Simon (2013)]

Method:

1. Thaw a pellet of nuclei (Protocol 1) on ice.
2. Rinse the nuclei pellet with ice cold 10 mL PBS and to capture the nuclei, centrifuge for 10 min at 1000 x g, 4 °C. Repeat this step for a total of 2 rinses. During the second rinse, transfer the sample to a 50-mL conical tube and then spin down.
3. Resuspend pellet in 45 mL ice-cold PBS and then add 5 mL of 32% formaldehyde. Incubate the sample at room temperature for 30 min on an end-over-end rotator. Add 4.55 mL of 1.5

M glycine (0.125 M final concentration) to quench the reaction, incubate for an additional 5 min.

4. Centrifuge 10 min at 2000 x g, 4 °C, to collect the nuclei, remove the supernatant, and wash with 50 mL of ice-cold PBS. (Do not resuspend pellet with pipette, nuclei are very sticky and will adhere to pipette).
5. To collect the nuclei, centrifuge for 10 min at 1000 x g, 4 °C. Remove the supernatant and transfer the pellet to a 15-mL conical tube using two rinses of 5 mL ice-cold PBS (10 mL total).
6. To collect the nuclei, centrifuge 10 min at 1000 x g, 4 °C and rinse pellet with 5 mL of WB 100 (made fresh at room temp) and centrifuge. Repeat for a total of two rinses.
7. Resuspend the nuclei to 1.4 mL volume with WB 100 supplemented with SUPERasIN (20 U/mL final) and 1x protease inhibitors. (Add 80 µL Complete inhibitor and 8 µL SUPERasIN to 8.0 mL WB 100)
8. Aliquot 100 µL per sonication tube. Should get 12 tubes/sample. Solution is very viscous, cut tip of pipet tip to help draw up accurate volumes.
9. Sonicate for 10 min in Bioruptor Pico, with 30 second on/off cycles. This sonication will have to be optimized for fragment sizes of interest.
10. Combine sonicated chromatin into equal volumes in 1.5-mL tubes and clear the extract by centrifugation for 20 min at 16100 x g, 4 °C.
11. Aliquot the cleared extract (250 to 500 µL aliquots) and go directly to step 12 or flash freeze (N_2) and store at -80°C.
12. **12.** Add the following to 500 µL of cleared sonicated samples.
    - 10 µL of 20 U/µL SUPERasIN
    - 5 µL of 1 M DTT
    - 5 µL 100X protease inhibitor
        - Mix gently by inverting, quick spin.
13. **13.** Add 250 µL of Denaturant buffer.
    - **a.** Mix gently by inverting, quick spin.
14. **14.** Add 750 µL of 2X hybridization buffer.
    - **a.** Mix gently by inverting, quick spin
15. **15.** Transfer solution to new labeled tubes.
16. **16.** Add 108 pmol (5.4 µL of 20 µM CHART oligos) for every 100 µL of solution (27 µL primer/500 µL extract). Pipette up/down 10X gently to mix.
17. **17.** Hybridize overnight at room temperature on end-over-end rotator

****

Day 2:

18. Clear samples by centrifuging for 20 min, 16000 x g at room temperature and remove supernatant.

  a. Set the temperature controlled centrifuge to 22 °C, because centrifuge heats up as it runs.

19. While samples are spinning: Pre-rinse 150 µL of MyOne Dynabeads two times with 500 µL distilled deionized water.

20. Resuspend beads in 200 µL water and 100 µL denaturant buffer (300 µL total volume).

21. Transfer 100 µL bead suspension into 3 separate tubes and add 1/3 volume of cleared CHART reaction to each tube, and incubate overnight at room temperature on end-over-end rotator.

****

Day 3:

Bring following items out to thaw:

- 1 M DTT
- 20 U/µL SUPERasIN
- 0.4 mM PMSF

Make following buffers (always prepare fresh and use immediately):

Wash buffer 250 (WB 250)

- 2.5 mL of 5 M NaCl (250 mM final)
- 500 µL of 1 M HEPES, pH 7.5 (10 mM final)
- 200 µL of 0.5 M EDTA (2 mM final)
- 100 µL of 0.5 M E**G**TA (1 mM final)
- 1 mL of 10% (w/v) SDS (0.2% w/v final)
- 250 µL of 20% (w/v) *N*-lauroylsarcosine (0.10% final)
- Distilled, deionized water to 50 mL
  - Immediately before use, add 100 µL of 0.4 mM of phenylmethylulfonyl fluoride (PMSF) to solution, then filter with 0.22-µm filter

Rnase H eluction buffer (HEB) (4 mL)

- 3.626 mL H20

- 200 µL of 1 M HEPES pH 7.5 (50 mM final)
- 60 µL of 5 M NaCl (75 mM final)
- 50 µL of 20% (w/v) *N*-lauroylsarcosine (0.125% final)
- 8 µL of 10% (w/v) sodium deoxycholate (0.025%)
- 12 µL of 1 M $MgCl_2$ (3 mM final)
- 4 µL of 20 U/µL SUPERasIN (20 U/mL final)
- 40 µL of 1 M DTT (10 mM final)

Nucleic Acid XLR buffer (cross-link reversal)

- 125 µL of 1 M TrisCl, pH 7.5
- 125 µL of 10% (w/v) SDS
- 250 µL of 20 mg/mL proteinase K (Ambion, cat AM2548)
    - **Prepare immediately before use (During 37 °C incubation)**

22. Quick spin all samples, and then set No Oligo/Input control sample aside (will add HEB and process later)**.**

23. For samples that contain an oligo, pellet beads using magnet and resuspend in 750 µL of WB 250.

24. Wash beads a total of 4 times with 750 µL WB 250.

   a. Invert the tubes gently ~15X to mix.

25. Pool together the three reactions from each CHART pull-down into a single 1.7-mL tube (make sure to rinse each tube with an additional 100 µL WB 250 to capture all beads).

26. Separate beads using a magnet, resuspend beads in 250 µL HEB, and transfer to a new 1.7 mL tube. Rinse the tube with another 250 µL HEB (in the same tube as above), so that no beads are left behind and transfer solution to the new tube (should have 1 tube for *slincR* and 1 tube for sense samples/biological replicate).

27. Separate beads using a magnet, remove the supernatant, and resuspend in 425 µL HEB. Aliquot 210 µL into two separate reactions for RNase H elution.

28. Add 5 µL RNase H to each tube and incubate at 37 °C for 30 min.

   a. Placed rocker in 37 °C incubator and gently rock.

29. While samples are incubating:

   a. Make the 1% no-oligo samples

      i. 1% = 4 µL supernatant + 396 µL HEB

  - **b.** Make nucleic acid XLR buffer (do not add proteinase K until right before adding to samples)
  - **c.** Label PCR tubes
    - **i.** Each sample will get 4 tubes with ~125 µL volume in each

30. Once oligo samples are done incubating, quick spin samples and then use magnet to isolate the beads. Store the supernatant into new 1.7 mL tube. Bring volume up to 400 µL with HEB if some sample is lost.

31. Add 100 µL sample of nucleic acid XLR buffer with proteinase K to all samples.
  - **a.** Invert tubes to mix. Quick spin.

32. Aliquot ~125 µL of sample into 4 pre-labeled PCR tubes

33. Incubate at 55 °C for 1 hr followed by 1 hr 40 min at 65 °C to reverse cross-link
  - **a.** Longer incubation times at 65 °C leads to improved efficiency of cross-link reversal, but increased RNA degradation. Determine best time experimentally (30-120 min).

Clean up DNA using ZYMO ZR-Duet DNA/RNA miniprep (Cat D7001)

1. Follow manufacture protocol and include in column DNase I digestion for RNA samples
  - **a.** RNA eluted in 25 µL H_2O
  - **b.** DNA eluted in 50 µL

Analyze Target enrichment via qPCR

1. Set up Vilo reverse transcription (RT) reactions (10 µL total) as follows:
  - 2 µL of 5X VILO master mix
  - 2 µL of RNA pull-down
  - 5 µL H_2O
  - 1 µL VILO RT enzyme mix
    - i. May include a larger volume of RNA, but too much sample may lead to inhibition of RT reaction. Determine experimentally.

2. Incubate as instructed (25 °C for 10 min, 42 °C for 60 min, 85 °C for 5 min, 4 °C hold)

3. Set up SYBR green qPCR (20 µL total) reaction
  - 10 µL 2X SYBR green master mix
  - 0.4 µL of 10 uM Forward and Reverse Primers
  - 4.2 µL H_2O

- 5 µL of diluted cDNA (cDNA diluted @ 1:20)

4. Run standard SYBR green program
    - 95 °C for 10 min
    - 95 °C for 15 sec, 60 °C (or optimal anneal for primers) for 1 min: 40 Cycles
    - Add melting stage if desired

References

Bogdanovic O, Fernandez-Minan A, Tena JJ, de la Calle-Mustienes E, Gomez-Skarmeta JL. 2013. The developmental epigenomics toolbox: ChIP-seq and MethylCap-seq profiling of early zebrafish embryos. Methods. 62: 207-215. DOI: 10.1016/j.ymeth.2013.04.011.

Simon MD. 2013. Capture hybridization analysis of RNA targets (CHART). Curr Protoc Mol Biol. Chapter 21: Unit 21 25. DOI: 10.1002/0471142727.mb2125s101.

Westerfield M. 2000. The zebrafish book. A guide for the laboratory use of zebrafish (Danio rerio). 4th ed. Eugene, OR: University of Oregon.

S2. Differential expression custom R script

```
###########################################################################
##  Script and Data Analysis by Cheryl Dunham                            ##
##                                                                       ##
##  Budding Bioinformatician                                             ##
##  Tanguay Laboratory | Sinnhuber Aquatic Research Laboratory (SARL)    ##
##  28645 E Hwy 34                                                       ##
##  Corvallis, OR 97333                                                  ##
##  Oregon State University                                              ##
##                                                                       ##
##  (541) 737-6500                                                       ##
##  dunhamcg@gmail.com                                                   ##
###########################################################################


##----1) Install any missing packages----
#Use to install multiple CRAN packages
#install.packages(c('xlsx','dplyr'))
#Use to install a single CRAN package
#install.packages('xlsx')

#To install Bioconductor packages
#source("https://bioconductor.org/biocLite.R")
#biocLite('edgeR')
#Use to upgrade current Bioconductor packages
#biocLite()


##----2) Load packages and specify file paths----
#List all required packages to load in loop
all_packages <- c('dplyr',
                  'xlsx',   #requires updated version of Java (most likely 64
bit)
                  'rJava',  #required to help control Java memory problems
that arise from using xlsx package
                  'dichromat',  #color blind friendly color palettes
                  'scales',  #enables you to see what color codes look like
(makes a color strip image)
                  'ggplot2',
#Bioconductor packages:
                  'biomaRt',
                  'DESeq2',
                  'edgeR',
                  'ComplexHeatmap')
#Load all packages at once
sapply(all_packages, require, character.only = TRUE)

#I append this to the beginning of all files to ensure all file names go into
whatever folder you specify
output_filepath <- 'C:/Users/dunhamc/Dropbox/SlincR (2nd paper)/Reviewer
Comments/Resubmission/Data'
#File path of data to be analyzed
input_filepath <- 'C:/Users/dunhamc/Google Drive/RNA_Seq/Sox9blncRNA_RNA-
Seq/Data/Zv10_89/HTseq_count'
```

```
##----3) Import and combine sequence data----
#Set the working directory to the folder containing HTSeq-count output .tab
files
setwd(input_filepath)

HTseq_combo <- function(){
    #Combines data from multiple HTseq-count output files (sans the --
additional-attr option) into a single dataframe
    #be sure to be in the correct working directory before running the
function

      #List all file names from the current working folder that end in .tab
file extension
      filenames <- list.files(pattern = '.tab')
    #Read in the count table from the first file in the folder
    initial_count <- read.table(filenames[1], sep ='\t', stringsAsFactors =
FALSE, header = FALSE)
      #Creates the dataframe to hold information from all .tab files in the
current working folder
    counts_combo <- as.data.frame(matrix(ncol = length(filenames) + 1, nrow =
length(initial_count[,1])))
      #Sets the first column of the combined dataframe to the Ensembl gene ID
(the first column of HTSeq-count files)
    counts_combo[,1] <- initial_count[,1]
      #Sets the second column of the combined dataframe to the corresponding
gene count values for the first .tab file in the current folder
    counts_combo[,2] <- initial_count[,2]
      #Sets the column names of the first 2 columns in the combined dataframe
to 'Ensembl_ID' and the file name of the first .tab file in the current
folder
    names(counts_combo)[1:2] <- c('Ensembl_ID',filenames[1])

      #Create loop to add gene count information to the combined dataframe
from the rest of the .tab files in the current folder
    for (i in 2:length(filenames)){
        #Read in HTSeq-count data table
            counts <- read.table(filenames[i], sep = '\t', stringsAsFactors =
FALSE, header = FALSE)
            #Add gene count information to the combined dataframe
        counts_combo[,i+1] <- counts[,2]
            #Link the HTSeq-count file name to its corresponding gene counts
        names(counts_combo)[i+1] <- filenames[i]
    }

      #Remove non-count-data information from the combined data table (last 5
rows of the dataframe)
    counts_combo <- counts_combo[1:(length(counts_combo[,1])-5),]
      #Label rows of the combined data table with the corresponding Ensembl
gene ID
    row.names(counts_combo) <- counts_combo[,1]
      #Remove the first column of the combined dataframe, leaving only count
information in the data table
      counts_combo <- counts_combo[,-1]
    return(counts_combo)
}
```

```r
#Merge count data into one data table
merged_counts <- HTseq_combo()

#Set the working directory
setwd(output_filepath)


##----4) Add sample information and link sample info to count table----
sampleInfo <- read.xlsx(file = 'C:/Users/dunhamc/Google
Drive/RNA_Seq/Sox9blncRNA_RNA-Seq/Data/Zv10_89/Output/sampleInfo.xlsx',
sheetIndex = 'Sheet1', header = TRUE, stringsAsFactors = FALSE)

#Add experimental groups/factors for differential testing
sampleInfo$Morphant <- factor(sampleInfo$Morphant, levels = c('CM','SM'))
sampleInfo$Treatment <- factor(sampleInfo$Treatment, levels =
c('DMSO','TCDD'))
sampleInfo$Group <- factor(sampleInfo$Group)
#Ensure the 'Filename' column is of the character class and not a factor
sampleInfo$Filename <- as.character(sampleInfo$Filename)

#Verify that the file names associated with the merged count table matches
with the filenames provided in the sampleInfo spreadsheet
colnames(merged_counts) %in% sampleInfo$Filename
sampleInfo$Filename %in% colnames(merged_counts)

#Create gene count matrix arranged by experimental sample order from sample
information spreadsheet
experiment_counts <- merged_counts[,sampleInfo$Filename]
#Verify the column info from count data matches row info from sampleInfo
sheet
colnames(experiment_counts) == sampleInfo$Filename
sum(colnames(experiment_counts) != sampleInfo$Filename)
#Rename experimental count table with corresponding sample ID abbreviation
colnames(experiment_counts) <- sampleInfo$subID
#Write count matrix to file
write.table(experiment_counts, file =
paste0(output_filepath,'/Count_matrix.txt'), quote = FALSE, sep = "\t",
row.names = TRUE, col.names = TRUE)


##----5) Make the edgeR digital gene expression (DGE) data object----
#See the Bioconductor workflow, 'From reads to genes to pathways:
differential expression analysis of RNA-Seq experiments using Rsubread and
the edgeR quasi-likelihood pipeline'
<https://www.bioconductor.org/packages/release/workflows/vignettes/RnaSeqGene
EdgeRQL/inst/doc/edgeRQL.html>, for an up-to-date RNAseq differential
expression workflow
DGE <- DGEList(counts = experiment_counts,
                     genes = row.names(experiment_counts),
                     group = sampleInfo$Group)
#Add additional sample information to the DGE object
DGE$samples$Treatment <- sampleInfo$Treatment
DGE$samples$Morphant <- sampleInfo$Morphant
DGE$samples$ID <- sampleInfo$ID
DGE$samples$subID <- sampleInfo$subID
```

```
##----6) Add gene annotation information----
#Zv10 (GRCz10) Ensembl release 89 annotation information
ensemblDrer89 <- useMart(biomart = 'ENSEMBL_MART_ENSEMBL', dataset =
'drerio_gene_ensembl', host = 'may2017.archive.ensembl.org')

##--Biomart options--
#sets the data columns to be added to the new dataframe (attributes) and
filters the data based on user specifications and associated values (from the
main Ensembl Biomart site)
#Filters
#listFilters(ensemblMmus89)  #to see all possible filter names
#Filters for use in all annotation retrieval
filts <- 'ensembl_gene_id'
vals <- row.names(DGE) #values corresponding to filters (filts)

#Attributes
#listAttributes(ensemblMmus89)  #to see all possible attribute names
#Attributes for zebrafish annotation information
zebrafish_attributes <-
c('ensembl_gene_id','external_gene_name','gene_biotype')
#Attributes for zebrafish to human ortholog information
human_attributes <-
c('ensembl_gene_id','external_gene_name','hsapiens_homolog_ensembl_gene','hsa
piens_homolog_associated_gene_name')
#Attributes for zebrafish to mouse ortholog information
mouse_attributes <-
c('ensembl_gene_id','mmusculus_homolog_ensembl_gene','mmusculus_homolog_assoc
iated_gene_name')

##----Add zebrafish gene annotation information----
#Retrieve zebrafish Ensembl gene ID, gene name/symbol, and gene type for
those genes included in the DGE object
martTable1 <- getBM(attributes = zebrafish_attributes, filters = filts,
values = vals, mart = ensemblDrer89)

#Ensure the table is ordered by Ensembl gene ID
martTable1 <- martTable1[order(martTable1$ensembl_gene_id),]
#Check for duplicated Ensembl gene IDs
sum(duplicated(martTable1$ensembl_gene_id))

#Verify that the DGE gene ID annotation information corresponds with the
zebrafish BioMart annotation table before creating a master annotation table
sum(DGE$genes$genes != martTable1$ensembl_gene_id)

#Create master BioMart annotation table
martTable <- data.frame(genes = DGE$genes$genes)
#Add gene symbol information to the master BioMart annotation table
martTable$Symbol <- martTable1$external_gene_name

#Ensure that the Symbol column does not have NAs or blanks
sum(is.na(martTable$Symbol))
sum(martTable$Symbol == '')
if(sum(martTable$Symbol == '') > 0){
    for (i in seq_along((martTable$Symbol))){
        if (martTable$Symbol[i] == ''){
            martTable$Symbol[i] <- martTable$genes[i]
        }
    }
}

#Ensures that each value in the Symbol column is unique
sum(duplicated(martTable$Symbol))
#Add each Symbol to gene_symbols
gene_symbols <- martTable$Symbol[1]
for (i in 2:length(martTable$Symbol)){
    print(i)
    if (martTable$Symbol[i] %in% gene_symbols){
        #If the Symbol is not unique (i.e. already included in gene_symbols), 
then add a blank space to the end of its name, so that it appears the same 
but registers as unique for row names to be included in heatmaps
        while(martTable$Symbol[i] %in% gene_symbols){
            martTable$Symbol[i] <- paste0(martTable$Symbol[i], ' ')
        }
    }
    #Add each symbol to the cumulative gene_symgols list
    gene_symbols <- c(gene_symbols, martTable$Symbol[i])
}
##Verify that each value in the Symbol column is unique
sum(duplicated(martTable$Symbol))

#Add gene type to the master BioMart annotation table
martTable$Biotype <- martTable1$gene_biotype


##----Add human gene annotation information----
martTable2 <- getBM(attributes = human_attributes, filters = filts, values = 
vals, mart = ensemblDrer89)
#Order results by zebrafish Ensembl gene ID
martTable2 <- martTable2[order(martTable2$ensembl_gene_id),]

#Check for duplicated zebrafish Ensembl ID values
sum(duplicated(martTable2$ensembl_gene_id))
#Create column in zebrafish to human BioMart annotation table for combining 
multiple human orthologs (gene ID) into one line
martTable2$hsapiens_homolog_ensembl_gene_all <- 
martTable2$hsapiens_homolog_ensembl_gene
#Create column in zebrafish to human BioMart annotation table for combining 
multiple human orthologs (gene name) into one line
martTable2$hsapiens_homolog_associated_gene_name_all <- 
martTable2$hsapiens_homolog_associated_gene_name


##--Combine multiple human gene ID and symbol information into one line
for (i in 2:length(DGE$genes$genes)){
    print(i)
    if (martTable2$ensembl_gene_id[(i-1)] == martTable2$ensembl_gene_id[i]){
        while(martTable2$ensembl_gene_id[(i-1)] == 
martTable2$ensembl_gene_id[i]){
            martTable2$hsapiens_homolog_ensembl_gene[i] <- 
martTable2$hsapiens_homolog_ensembl_gene[i-1]
            martTable2$hsapiens_homolog_ensembl_gene_all[i] <- 
paste(martTable2$hsapiens_homolog_ensembl_gene_all[i-1], 
martTable2$hsapiens_homolog_ensembl_gene_all[i], sep = ', ')
```

```
            martTable2$hsapiens_homolog_associated_gene_name[i] <-
martTable2$hsapiens_homolog_associated_gene_name[i-1]
            martTable2$hsapiens_homolog_associated_gene_name_all[i] <-
paste(martTable2$hsapiens_homolog_associated_gene_name_all[i-1],
martTable2$hsapiens_homolog_associated_gene_name_all[i], sep = ', ')
            martTable2 <- martTable2[-(i-1),]
        }
    }
}

#Check for duplicated zebrafish Ensembl ID values
sum(duplicated(martTable2$ensembl_gene_id))
#Verify that the master BioMart annotation table information corresponds with
the zebrafish to human ortholog BioMart annotation table before adding its
columns to the master table
sum(martTable$genes != martTable2$ensembl_gene_id)

#Add human Ensembl gene ID ortholog information to the master BioMart
annotation table
martTable$Hsap_ID <- martTable2$hsapiens_homolog_ensembl_gene
#Add human Ensembl gene ID multiple ortholog information to the master
BioMart annotation table
martTable$Hsap_IDs <- martTable2$hsapiens_homolog_ensembl_gene_all
#Add human gene symbol ortholog information to the master BioMart annotation
table
martTable$Hsap_symbol <- martTable2$hsapiens_homolog_associated_gene_name
#Add human gene symbol multiple ortholog information to the master BioMart
annotation table
martTable$Hsap_symbols <-
martTable2$hsapiens_homolog_associated_gene_name_all

##----Add mouse gene annotation information----
martTable3 <- getBM(attributes = mouse_attributes, filters = filts, values =
vals, mart = ensemblDrer89)
#Order results by zebrafish Ensembl gene ID
martTable3 <- martTable3[order(martTable3$ensembl_gene_id),]

#Check for duplicated zebrafish Ensembl ID values
sum(duplicated(martTable3$ensembl_gene_id))
#Create column in zebrafish to mouse BioMart annotation table for combining
multiple mouse orthologs (gene ID) into one line
martTable3$mmusculus_homolog_ensembl_gene_all <-
martTable3$mmusculus_homolog_ensembl_gene
#Create column in zebrafish to mouse BioMart annotation table for combining
multiple mouse orthologs (gene name) into one line
martTable3$mmusculus_homolog_associated_gene_name_all <-
martTable3$mmusculus_homolog_associated_gene_name

##--Combine multiple mouse gene ID and symbol information into one line
for (i in 2:length(DGE$genes$genes)){
    print(i)
    if (martTable3$ensembl_gene_id[(i-1)] == martTable3$ensembl_gene_id[i]){
        while(martTable3$ensembl_gene_id[(i-1)] ==
martTable3$ensembl_gene_id[i]){
            martTable3$mmusculus_homolog_ensembl_gene[i] <-
martTable3$mmusculus_homolog_ensembl_gene[i-1]
```

```
            martTable3$mmusculus_homolog_ensembl_gene_all[i] <-
paste(martTable3$mmusculus_homolog_ensembl_gene_all[i-1],
martTable3$mmusculus_homolog_ensembl_gene_all[i], sep = ', ')
            martTable3$mmusculus_homolog_associated_gene_name[i] <-
martTable3$mmusculus_homolog_associated_gene_name[i-1]
            martTable3$mmusculus_homolog_associated_gene_name_all[i] <-
paste(martTable3$mmusculus_homolog_associated_gene_name_all[i-1],
martTable3$mmusculus_homolog_associated_gene_name_all[i], sep = ', ')
            martTable3 <- martTable3[-(i-1),]
        }
    }
}

#Check for duplicated zebrafish Ensembl ID values
sum(duplicated(martTable3$ensembl_gene_id))
#Verify that the master BioMart annotation table information corresponds with
the zebrafish to mouse ortholog BioMart annotation table before adding its
columns to the master table
sum(martTable$genes != martTable3$ensembl_gene_id)

#Add mouse Ensembl gene ID ortholog information to the master BioMart
annotation table
martTable$Mmus_ID <- martTable3$mmusculus_homolog_ensembl_gene
#Add mouse Ensembl gene ID multiple ortholog information to the master
BioMart annotation table
martTable$Mmus_IDs <- martTable3$mmusculus_homolog_ensembl_gene_all
#Add mouse gene symbol ortholog information to the master BioMart annotation
table
martTable$Mmus_symbol <- martTable3$mmusculus_homolog_associated_gene_name
#Add mouse gene symbol multiple ortholog information to the master BioMart
annotation table
martTable$Mmus_symbols <-
martTable3$mmusculus_homolog_associated_gene_name_all

#Ensure that the BioMart annotation table does not contain NA values
martTable[is.na(martTable)] <- ''

##----OR: Add annotation information from file----
#martTable <- read.delim(file = file.choose(), header = TRUE,
stringsAsFactors = FALSE)

#Verify that the master BioMart annotation table corresponds with the order
of genes in the DGE object
sum(DGE$genes$genes != martTable$genes)
#Add BioMart annotation information to the DGE object
DGE$genes$Symbol <- martTable$Symbol
DGE$genes$Biotype <- martTable$Biotype
DGE$genes$Hsap_ID <- martTable$Hsap_ID
DGE$genes$Hsap_IDs <- martTable$Hsap_IDs
DGE$genes$Hsap_symbol <- martTable$Hsap_symbol
DGE$genes$Hsap_symbols <- martTable$Hsap_symbols
DGE$genes$Mmus_ID <- martTable$Mmus_ID
DGE$genes$Mmus_IDs <- martTable$Mmus_IDs
DGE$genes$Mmus_symbol <- martTable$Mmus_symbol
DGE$genes$Mmus_symbols <- martTable$Mmus_symbols

#Write DGE annotation table to file
```

```
write.table(DGE$genes, file =
paste0(output_filepath,'/Zfish_martTable_Zv10_89.txt'), quote = FALSE, sep =
"\t", row.names = FALSE, col.names = TRUE)


##----7) Additional filtering based on edgeR vignette and (Chen 2016)----
#Determine counts per million corresponding to a minimum of 10-20 read counts
mean_cpm_gene_count_of_15 <- as.numeric(cpm(15, mean(DGE$samples$lib.size)))
#Determine the number of samples in the smallest group
num_samples_smallest_group <- min(table(sampleInfo$Group))  #4 is the minimum
of samples in each group

#Only keep genes that have a minimum of 10-20 counts in at least 4 samples to
be included in analysis
keep.exprs <- rowSums(cpm(DGE)> mean_cpm_gene_count_of_15) >=
num_samples_smallest_group
DGE <- DGE[keep.exprs, , keep.lib.sizes=FALSE]

#Write background genes to file
write.table(DGE$genes$genes, file =
paste0(output_filepath,'/Zfish_background_genes.txt'), quote = FALSE, sep =
"\t", row.names = FALSE, col.names = FALSE)
#Write human ortholog background genes to file
write.table(DGE$genes$Hsap_ID[DGE$genes$Hsap_ID != ''], file =
paste0(output_filepath,'/Metacore_background_genes.txt'), quote = FALSE, sep
= "\t", row.names = FALSE, col.names = FALSE)
#Write annotation table to file
write.table(DGE$genes, file = paste0(output_filepath,'/DGE_annotation.txt'),
quote = FALSE, sep = "\t", row.names = FALSE, col.names = TRUE)


##----8) Normalize libraries----
DGE <- calcNormFactors(DGE)
DGE$samples
write.table(DGE$samples, file =
paste0(output_filepath,'/DGE_libsize_normfactors.txt'), quote = FALSE, sep =
"\t", row.names = TRUE, col.names = TRUE)

##----Create normalized count data for use with heatmaps and PCA plots----
#Calculate cpm to factor in TMM normalization
rltdata <- cpm(y = DGE, normalized.lib.sizes = TRUE) #get normalized count
data in counts per million units
#Multiply normalized cpm numbers by library size per million to get count
values normalized by library size
for (bio_sample in colnames(rltdata)){
    rltdata[,bio_sample] <-
(rltdata[,bio_sample])*((DGE$samples[bio_sample,'lib.size']/1000000))
}
#Assign TMM-normalized counts to the 'normalized_counts' slot of the DGE
object
DGE$normalized_counts <- rltdata
#Regular log normalization of count data (corrects for heteroskedasticy)
rltdata <- rlogTransformation(round(rltdata))
#Add gene labels to TMM-normalized, rlog transformed data
row.names(rltdata) <- row.names(DGE$counts)
#Assign TMM-normalized, rlog transformed counts to the 'rlog_normalized' slot
of the DGE object
```

```
DGE$rlog_normalized <- rltdata
#Write TMM-normalized, rlog transformed count table to file
write.table(DGE$rlog_normalized, file =
paste0(output_filepath,'/DGE_rlog_normalized_counts.txt'), quote = FALSE, sep
= "\t", row.names = TRUE, col.names = TRUE)


##----9) Make the design matrix----
design <- model.matrix(~0 + DGE$samples$group) #the zero means that there is
no intercept in the model, this helps with contrasts in later steps
#Assign sample names to the rows of the design matrix
rownames(design) <- colnames(DGE)
#Assign group names to the columns of the design matrix
colnames(design) <- levels(DGE$samples$group)
design


##----10) Estimate dispersions----
DGE <- estimateDisp(DGE, design, robust = TRUE)
DGE$common.dispersion
sqrt(DGE$common.dispersion)
plotBCV(DGE, main = 'Dispersion estimation')


##----11) Fit data to the design model----
fit <- glmQLFit(DGE, design, robust = TRUE)
plotQLDisp(glmfit = fit, main = 'Quasi-liklihood Dispersion')


##----12) Design contrast matrix to extract comparisons of interest----
my_contrasts <- makeContrasts(`CM_TCDD-CM_DMSO` = CM_TCDD - CM_DMSO,
                              `SM_TCDD-SM_DMSO` = SM_TCDD - SM_DMSO,
                              `SM_DMSO-CM_DMSO` = SM_DMSO - CM_DMSO,
                              `SM_TCDD-CM_TCDD` = SM_TCDD - CM_TCDD,
                              levels = design)


##----13) Set up ComplexHeatmap parameters----
#Set heatmap global text options
ht_global_opt(heatmap_row_names_gp = gpar(fontfamily = 'sans'),
              heatmap_row_title_gp = gpar(fontfamily = 'sans'),
              heatmap_legend_title_gp = gpar(fontfamily = 'sans', fontsize =
8),
              heatmap_legend_labels_gp = gpar(fontfamily = 'sans', fontsize =
8),
              heatmap_column_names_gp = gpar(fontfamily = 'sans'),
              heatmap_column_title_gp = gpar(fontfamily = 'sans'),
              annotation_legend_title_gp = gpar(fontfamily = 'sans', fontsize
= 8),
              annotation_legend_labels_gp = gpar(fontfamily = 'sans',
fontsize = 8))

#Set up color schemes for row annotation information
biotypes <- data.frame(biotypes = c('antisense', 'lincRNA', 'misc_RNA',
'Mt_rRNA', 'Mt_tRNA', 'processed_transcript', 'protein_coding',
'rRNA','sense_overlapping'))
Biotype_colScale <- data.frame(Biotype = biotypes$biotypes,
                               Color =
I(colorschemes$Categorical.12[c(12,5,3,7,8,9,10,11,6)]))
row.names(Biotype_colScale) <- Biotype_colScale$Biotype
show_col(Biotype_colScale$Color)

#Set up column annotation information
col_anno_info <- data.frame(Treatment = sampleInfo$Treatment, Morphant =
sampleInfo$Morphant)

col_anno <- HeatmapAnnotation(
    df = col_anno_info,
    col = list(
        Treatment = c('DMSO' = I(colorschemes$Categorical.12[7]),
                      'TCDD' = I(colorschemes$Categorical.12[12])),
        Morphant = c('CM' = I(colorschemes$Categorical.12[4]),
                     'SM'= I(colorschemes$Categorical.12[2]))),
    show_legend = TRUE,
    annotation_legend_param = list(
        Treatment = list(nrow = 1,
                         title = "Treatment",
                         title_position = "topcenter",
                         legend_direction = 'horizontal',
                         fontfamily = 'sans'),
        Morphant = list(nrow = 1,
                        title = "Morphant",
                        title_position = "topcenter",
                        legend_direction = 'horizontal',
                        fontfamily = 'sans')),
    show_annotation_name = TRUE,
    annotation_name_side = 'left',
    annotation_name_gp = gpar(fontsize = 8),
    height = unit(5, "mm")
)#END col_anno

#Creates 'padding' object so column annotation names are not cut off
padding <- unit.c(unit(1, "mm"),
                  grobWidth(textGrob("  Treatment  ")) - unit(22.5, "mm"),
                  unit(1, "mm"),
                  unit(1, "mm"))


##----14) Find significant genes for each comparison without a fold change
cutoff----
fold_change = 1

for(comparison in colnames(my_contrasts)){
    Sys.sleep(.01)
    print(comparison)
      #Make a label for each contrast
    cutoff_contrast <- paste0(comparison, '_log2(',fold_change,')')

      #Test for differential expression using quasi-likelihood F-tests
    comparisons <- glmQLFTest(glmfit = fit,
                              contrast = my_contrasts[,comparison])
    #Extract the significant differentially expressed genes that pass the
Benjamini-Hochberg FDR cutoff of 0.05
    sig_table <- data.frame(topTags(comparisons,
                                    n = Inf, adjust.method = "BH",
                                    p.value = 0.05))

    ##----create output folder for each contrast----
    if(!dir.exists(paste0(output_filepath,'/',
cutoff_contrast))){dir.create(paste0(output_filepath,'/', cutoff_contrast))}

    if(ncol(sig_table) > 0){
          #Create data objects for each DGELRT object created in the loop
        assign(cutoff_contrast, comparisons)
            #Create data table for each table of significant differentially
expressed genes
        assign(paste(cutoff_contrast, 'sigGenes', sep = '_'), sig_table)
        #Write table of significant differentially expressed genes to text
file
            write.table(x = sig_table, file =
paste0(output_filepath,'/',cutoff_contrast,'/', cutoff_contrast,
'_sig_log_ratios.txt'), sep = '\t', row.names = FALSE, quote = FALSE)
        #Write table of significant differentially expressed genes to Excel
file
        xlsx::write.xlsx2(x = sig_table, file =
paste0(output_filepath,'/',cutoff_contrast,'/', cutoff_contrast,
'_sig_log_ratios.xlsx'), row.names = FALSE)
        .jcall("java/lang/System", method = "gc") #helps with java memory
problems

            #Create table for use with MetaCore
        metacore_table <- sig_table[sig_table$Hsap_ID != '',
c('Hsap_ID','logFC','FDR')]
            #Write significanly differentially expressed upregulated human
orthologs to text file
        write.table(x = metacore_table[(metacore_table$logFC > 0),],file =
paste0(output_filepath,'/',cutoff_contrast,'/', cutoff_contrast,
'UR_Metacore_sigGenes.txt'), sep = '\t', row.names = FALSE, quote = FALSE)
            #Write significanly differentially expressed downregulated human
orthologs to text file
        write.table(x = metacore_table[(metacore_table$logFC < 0),],file =
paste0(output_filepath,'/',cutoff_contrast,'/', cutoff_contrast,
'DR_Metacore_sigGenes.txt'), sep = '\t', row.names = FALSE, quote = FALSE)
            #Write significanly differentially expressed human orthologs to
text file
        write.table(x = metacore_table,file =
paste0(output_filepath,'/',cutoff_contrast,'/', cutoff_contrast,
'_Metacore_sigGenes.txt'), sep = '\t', row.names = FALSE, quote = FALSE)
        #Write list of significant differentially expressed genes to text
file
        write.table(x = sig_table$genes, file =
paste0(output_filepath,'/',cutoff_contrast,'/', cutoff_contrast,
'_sigGene_list.txt'), sep = '\t', row.names = FALSE, col.names = FALSE, quote
= FALSE)
        #Write list of significant differentially expressed upregulated genes
to text file
        write.table(x = sig_table$genes[sig_table$logFC > 0], file =
paste0(output_filepath,'/',cutoff_contrast,'/', cutoff_contrast,
'UR_sigGene_list.txt'), sep = '\t', row.names = FALSE, col.names = FALSE,
quote = FALSE)
```

```
        #Write list of significant differentially expressed downregulated
genes to text file
        write.table(x = sig_table$genes[sig_table$logFC < 0], file =
paste0(output_filepath,'/',cutoff_contrast,'/', cutoff_contrast,
'DR_sigGene_list.txt'), sep = '\t', row.names = FALSE, col.names = FALSE,
quote = FALSE)

        ##----Heatmap of top genes sorted by fold change----
            #Order genes based on their absolute log2(fold change)
        heatmap_selector_FC <- sig_table[order(abs(sig_table$logFC),
decreasing = TRUE),]
            #Select the top 30 genes
        heatmap_selector_FC <- heatmap_selector_FC[1:30,'genes']
            #Determine the data from the TMM-normalized, rlog transformed
count table that matches the top 30 genes
        heatmap_selector_FC <- row.names(rltdata) %in% heatmap_selector_FC
            #Select data from the TMM-normalized, rlog transformed count
table matching the top 30 genes
        rlogdataFC <- data.frame(rltdata[heatmap_selector_FC,])
            #Select data from the annotation table matching the top 30 genes
        phenodataFC <- DGE$genes[heatmap_selector_FC, 'Biotype',drop = FALSE]
            #Label count data with gene symbols
        row.names(rlogdataFC) <- DGE$genes$Symbol[heatmap_selector_FC]
            #Label phenotype data with gene symbols
        row.names(phenodataFC) <- DGE$genes$Symbol[heatmap_selector_FC]
            #Scale the transformed count data
        rlogdataFC <- t(scale(t(rlogdataFC)))

        ##Set up row annotation information
        row_anno_info <- phenodataFC
        row_key <- names(table(as.character(row_anno_info$Biotype)))
        row_cols <- character()
        for (i in seq.int(1:length(row_key))){
            row_col <- Biotype_colScale[row_key[i],'Color']
            row_cols <- c(row_cols,row_col)
        }
        row_col_list <- list(Biotype = row_cols)
        names(row_col_list[[1]]) <- row_key

        ##Create main heatmap layer
        hmap <- Heatmap(matrix = rlogdataFC,
                        name = 'main_mat',
                        heatmap_legend_param =
                            list(title = 'Z-score',
                                 color_bar = 'continuous',
                                 legend_direction = 'horizontal',
                                 title_position = 'topcenter',
                                 legend_side = 'bottom',
                                 fontfamily = 'sans'),
                        column_title = cutoff_contrast,
                        column_title_side = 'top',
                        column_title_gp = gpar(fontsize = 10),
                        column_names_gp = gpar(fontsize = 8),
                        column_dend_height = unit(8,'mm'),
                        top_annotation = col_anno,
                        show_row_names = FALSE,
                        show_row_dend = TRUE,
                        row_dend_width = unit(13, "mm"),
                        show_heatmap_legend = TRUE
        )#END hmap

        ##Create row annotations layer
        rmap <- Heatmap(matrix = row_anno_info,
                        name = 'Biotype',
                        col = row_col_list$Biotype,
                        column_title_gp = gpar(fontsize = 8),
                        column_names_gp = gpar(fontsize = 8),
                        show_row_names = TRUE,
                        row_title_gp = gpar(fontsize = 8),
                        row_names_side = 'right',
                        row_names_gp = gpar(fontsize = 8),
                        heatmap_legend_param = (
                            Biotype = list(nrow = if(length(row_key) >
5){2}else{1},
                                           title = "Biotype",
                                           title_position = "topcenter",
                                           legend_direction = 'horizontal',
                                           fontfamily = 'sans')),
                        width = unit(2.5,'mm')
        )#END rmap

        #Draw the combined heatmap and write to file
        pdf(file = paste0(output_filepath,'/',cutoff_contrast,'/',
cutoff_contrast,'_heatmap_top30FC.pdf'), width = 3.4, height = 6)
        draw(hmap + rmap, padding = padding, gap = unit(1, "mm"),
heatmap_legend_side = 'bottom', annotation_legend_side = 'bottom')
        dev.off()
    }
}


##----15) Write session information----
save.image(paste0(output_filepath,'/RNAseq_RData_',Sys.Date(),'.RData'))
writeLines(capture.output({

cat('########################################################################
###\n##  Script and Data Analysis by Cheryl Dunham
##\n##
##\n##  Budding Bioinformatician
##\n##  Tanguay Laboratory | Sinnhuber Aquatic Research Laboratory (SARL)
##\n##  28645 E Hwy 34
##\n##  Corvallis, OR 97333
##\n##  Oregon State University
##\n##
##\n##  (541) 737-6500
##\n##  dunhamcg@gmail.com
##\n#########################################################################
##\n\n')
    cat('Input filepath:', input_filepath, '\n\n')    cat('Output filepath:',
output_filepath,
'\n\n________________________________________________________________________
_____________\n______________________________________________________________
_______________________\n\n\n')
    print(sessionInfo())
```

```
cat('\n______________________________________________________________________
_______________\n____________________________________________________________
_________________________\n\n\nRStudio Version Information\n\n')
    print(RStudio.Version())

cat('________________________________________________________________________
_____________\n______________________________________________________________
_______________________\n\n\nAttached Package Citation Information\n\n')
    print(lapply(all_packages, citation))

}), paste0(output_filepath,'/RNAseq_sessionInfo_',Sys.Date(),'.txt'))
```

S3. Customized Set Origin Tool

To use this Macro in ImageJ, copy into a text editor, such as Notepad++, save text as a .ijm file (e.g. Customized_Set_Origin_Tool.imj), and place in your ImageJ Program Files path (e.g. C:\Program Files (x86)\ImageJ\plugins\Scripts)

(see https://imagej.net/Scripting#Adding_scripts_to_the_Plugins_menu, last accessed: 10/10/2018)

```
//----------------------- Set Origin Tool (customized) ----------------------
//
// Based on the 'Set Origin Tool'
// from the following ImageJ thread:
// http://imagej.1557.x6.nabble.com/How-to-SET-a-point-as-the-new-ZERO-AXIS-POINT-td4651547.html last accessed 10/10/2018
//
// Customized by Cheryl L. Dunham
// Budding Bioinformatician
// Tanguay Lab | Sinnhuber Aquatic Research Laboratory (SARL)
// Oregon State University
// Corvallis, Oregon
// phone: (541) 737-3608
// email: dunhamcg@gmail.com
//
// Modified to work with the modified "Click Coordinates Tool".
// This macro will set the origin of an image on click. The point will
// be set as (0,0) on an (x,y) pixel-based coordinate system.

  macro "Set Origin Tool - C00fL808fL08f8" {;
    // added below line to ensure scale is set in pixels
     run("Set Scale...", "distance=0 known=0 pixel=1 unit=pixel");
     getCursorLoc(x, y, z, flags);
     run("Properties...", "origin="+ x+","+y);
     showStatus("Origin set to "+x+","+y);
   }
```

S4. Customized Click Coordinates Tool

To use this Macro in ImageJ, copy into a text editor, such as Notepad++, save text as a .ijm file (e.g. Customized_Click_Coordinates_Tool.imj), and place in your ImageJ Program Files path (e.g. C:\Program Files (x86)\ImageJ\plugins\Scripts)

(see https://imagej.net/Scripting#Adding_scripts_to_the_Plugins_menu, last accessed: 10/10/2018)

```
//------------------- Click Coordinates Tool (customized) -------------------
//
// Based on the 'Click Coordinates Tool'
// from the ImageJ Developer Resources Macros page:
// https://imagej.nih.gov/ij/macros/tools/ClickCoordinatesTool.txt, last
accessed 10/10/2018
//
// Customized by Cheryl L. Dunham
// Budding Bioinformatician
// Tanguay Lab | Sinnhuber Aquatic Research Laboratory (SARL)
// Oregon State University
// Corvallis, Oregon
// phone: (541) 737-3608
// email: dunhamcg@gmail.com
//
// On each click into an image, the x, y coordinates of the point are
// written into the "Results" window. The point can be also marked
// in the image (destructively). This tool can handle scaled images
// (also with nontrivial pixel aspect ratio). Double click on the tool
// icon to display the options dialog box. The "Invert Y" option in
// Analyze>Set Measurements is supported.
//
// Modified to work with the modified "Set Origin Tool".
// This macro will return (x,y) pixel coordinates with respect to the
// user-defined origin (selected using the modified "Set Origin Tool").


  // removed  var outputScaled = 1;     report raw coordinates (pixels) if
false
  var drawPoints = 0;      // draw cross at position of click
  var drawNumbers = 0;     // draw line number for each click

  macro 'Click Coordinates Tool -
C000P515335150P5a595775950D46D64P88ab0D8bDa8Pe8cc0Pc8c90D9fDbfDdf' {
     requires("1.37e");
     getCursorLoc(x, y, z, flags); // retrieve pixel coordinates on click
     if (drawPoints || drawNumbers) setupUndo();
     if (drawPoints) {
        setLineWidth(1);
        tickLength = 3; // the "radius" of the crosses marking the points
        drawLine(maxOf(x-tickLength,0),y, minOf(x+tickLength,getWidth()-1),
y);
        drawLine(x,maxOf(y-tickLength,0), x, minOf(y+tickLength,getHeight()-
1));
     }
     if (drawNumbers) {
        setFont("SansSerif",9);
        if (drawPoints) {
           setJustification("left");
           xText = x + tickLength + 1;
        } else {
           setJustification("center");
           xText = x + 1;
        }
        drawString(nResults+1, xText, y+6);
     }
     invertY =
parseInt(call("ij.plugin.filter.Analyzer.getMeasurements"))&4096!=0;
     if (invertY) y = getHeight() - y - 1;
     xScale = 1;
     yScale = 1;

       // removed outputScaled if else statement
       // replaced with below three lines of code
       toScaled(scaledx, scaledy);  // gets "Set Origin Tool"-defined origin
offset
       newx = x + scaledx; // sets the x coordinate in relation to the user-
defined origin
       newy = -(y + scaledy); // sets the y coordinate in relation to the
user-defined origin

       // below 2 lines of code replace the original setResult() lines
       setResult("X", nResults, newx);
     setResult("Y", nResults-1, newy);
       updateResults();

  }

  macro 'Click Coordinates Tool Options...' {
     requires("1.37e");
     Dialog.create("Click Coordinates Tool Options");
     Dialog.addCheckbox("Draw Cross at Each Clicked Point", drawPoints);
     Dialog.addCheckbox("Write Point Number at Each Clicked Point",
drawNumbers);
     Dialog.show();
       // removed outputScaled = Dialog.getCheckbox();
     drawPoints = Dialog.getCheckbox();
     drawNumbers = Dialog.getCheckbox();
  }
```

S5. Cartilage morphometrics R script

```
###########################################################################
##  Script by Cheryl Dunham                                              ##
##                                                                       ##
##  Budding Bioinformatician                                             ##
##  Tanguay Laboratory | Sinnhuber Aquatic Research Laboratory (SARL)    ##
##  28645 E Hwy 34                                                       ##
##  Corvallis, OR 97333                                                  ##
##  Oregon State University                                              ##
##                                                                       ##
##  (541) 737-6500                                                       ##
##  dunhamcg@gmail.com                                                   ##
###########################################################################


##----1) Install any missing packages----
#Use to install multiple CRAN packages
#install.packages(c('xlsx','dplyr'))
#Use to install a single CRAN package
#install.packages('xlsx')


##----2) Load packages and specify file paths----
#List all required packages to load in loop

#List all required packages to load in loop
all_packages <- c('dplyr',
                  'xlsx',
                  'ggplot2')
#Load all packages at once, ignore the screen print out
sapply(all_packages, require, character.only = TRUE)


#I append this to the beginning of all files to ensure all file names go into
whatever folder you specify
output_filepath <- 'C:/Users/dunhamc/Google
Drive/R_scripts/Manuscripts/SlincR (2nd paper)'
input_filepath <- 'C:/Users/dunhamc/Google
Drive/Tanguay_Lab/Phenotype_Scrn/Cart_stain/08-11-
2017/Cart_data_8_11_17_b.xlsx'

##----3) Read in table----
#read in file, specify path to file, tab of Excel workbook, and column header
is names in first row
analysis_table <- read.xlsx(file = input_filepath, sheetIndex = 'R_input',
header = TRUE, stringsAsFactors = FALSE)

#turn morphant status, treatment, and location into factors and specify
control with levels
analysis_table$Morphant <- factor(analysis_table$Morphant, levels =
c('ConMO','SlincR_MO'))
analysis_table$Treatment <- factor(analysis_table$Treatment, levels =
c('DMSO','TCDD'))
analysis_table$Location <- factor(analysis_table$Location)

#make new column defining groups required for statistical testing below
```

```
analysis_table$Group <- factor(paste(analysis_table$Morphant,
analysis_table$Treatment, sep = '_'), levels =
c('ConMO_DMSO','ConMO_TCDD','SlincR_MO_DMSO','SlincR_MO_TCDD'))


##----4) Scatterplots w/ x and y SE bars----
cart_summary <- analysis_table %>%
    group_by(.dots= c('Morphant','Treatment','Location')) %>%
    summarize(N = n(),
              mean_x = signif(mean(X_coord),digits = 3),
              sd_x = signif(sd(X_coord), digits = 3),
              SE_x = signif(sd(X_coord)/sqrt(n()), digits = 3),
              mean_y = signif(mean(Y_coord), digits =3),
              sd_y = signif(sd(Y_coord), digits = 3),
              SE_y = signif(sd(Y_coord)/sqrt(n()), digits = 3))


##----Position A----
pdf(paste0(output_filepath,'/Cart_data_A.pdf'), width = 10, height = 8)
print(ggplot(data = subset(cart_summary, Location == 'A'), aes(x = mean_x, y
= mean_y)) +
          geom_point(aes(fill = Treatment, shape = Morphant, size =
Morphant), color = 'black', stroke = 0.05) +
          geom_hline(yintercept = 0) +
          geom_vline(xintercept = 0) +
          scale_y_continuous(name = 'y-axis', expand = c(0,0), limits =
c(0,165), position = 'right') +
          scale_x_continuous(name = 'x-axis', expand = c(0,0), limits = c(-
255,0), position = 'bottom') +
          scale_shape_manual(values = c(22,23)) +
          scale_size_manual(values = c(9,9)) +
          scale_fill_manual(values = c('blue','red')) +
          geom_errorbar(aes(ymin = mean_y-SE_y, ymax = mean_y + SE_y),
                        width = 5, size = 0.75) +
          geom_errorbarh(aes(xmin = mean_x-SE_x, xmax = mean_x + SE_x),
                         height = 5, size = 0.75) +
          guides(fill = guide_legend(override.aes = list(size = 8, color =
c('blue','red'), order = 1),
                                     shape = guide_legend(override.aes =
list(size = c(8,8),fill ='black')), order = 2)) +
          ggtitle('Position A') +
          theme(text = element_text(family = 'sans'),
                plot.title = element_text(size = 24, hjust = 0.5),
                legend.title = element_text(size = 14),
                legend.text = element_text(size = 12),
                legend.key = element_rect(fill = 'white'),
                axis.title = element_text(size = 18),
                axis.text = element_text(size = 14),
                axis.ticks = element_line(color = 'black'),
                panel.background = element_rect(fill = 'white', color =
'black'),
                panel.border = element_blank(),
                panel.grid = element_blank()))
dev.off()

##----Position B----
pdf(paste0(output_filepath,'/Cart_data_B.pdf'), width = 10, height = 8)
```

```
print(ggplot(data = subset(cart_summary, Location == 'B'), aes(x = mean_x, y
= mean_y)) +
          geom_point(aes(fill = Treatment, shape = Morphant, size =
Morphant), color = 'black', stroke = 0.05) +
          geom_hline(yintercept = 0) +
          geom_vline(xintercept = 0) +
          scale_y_continuous(name = 'y-axis', expand = c(0,0), limits =
c(0,165), position = 'left') +
          scale_x_continuous(name = 'x-axis', expand = c(0,0), limits =
c(0,255), position = 'bottom') +
          scale_shape_manual(values = c(22,23)) +
          scale_size_manual(values = c(9,9)) +
          scale_fill_manual(values = c('blue','red')) +
          geom_errorbar(aes(ymin = mean_y-SE_y, ymax = mean_y + SE_y),
                        width = 5, size = 0.75) +
          geom_errorbarh(aes(xmin = mean_x-SE_x, xmax = mean_x + SE_x),
                         height = 5, size = 0.75) +
          guides(fill = guide_legend(override.aes = list(size = 8, color =
c('blue','red'), order = 1),
                                     shape = guide_legend(override.aes =
list(size = c(8,8),fill ='black')), order = 2)) +
          ggtitle('Position B') +
          theme(text = element_text(family = 'sans'),
                plot.title = element_text(size = 24, hjust = 0.5),
                legend.title = element_text(size = 14),
                legend.text = element_text(size = 12),
                legend.key = element_rect(fill = 'white'),
                axis.title = element_text(size = 18),
                axis.text = element_text(size = 14),
                axis.ticks = element_line(color = 'black'),
                panel.background = element_rect(fill = 'white', color =
'black'),
                panel.border = element_blank(),
                panel.grid = element_blank()))
dev.off()

##----Position C----
pdf(paste0(output_filepath,'/Cart_data_C.pdf'), width = 10, height = 8)
print(ggplot(data = subset(cart_summary, Location == 'C'), aes(x = mean_x, y
= mean_y)) +
          geom_point(aes(fill = Treatment, shape = Morphant, size =
Morphant), color = 'black', stroke = 0.05) +
          geom_hline(yintercept = 0) +
          geom_vline(xintercept = 0) +
          scale_y_continuous(name = 'y-axis', expand = c(0,0), limits = c(-
165,0), position = 'right') +
          scale_x_continuous(name = 'x-axis', expand = c(0,0), limits = c(-
255,0), position = 'top') +
          scale_shape_manual(values = c(22,23)) +
          scale_size_manual(values = c(9,9)) +
          scale_fill_manual(values = c('blue','red')) +
          geom_errorbar(aes(ymin = mean_y-SE_y, ymax = mean_y + SE_y),
                        width = 5, size = 0.75) +
          geom_errorbarh(aes(xmin = mean_x-SE_x, xmax = mean_x + SE_x),
                         height = 5, size = 0.75) +
          guides(fill = guide_legend(override.aes = list(size = 8, color =
c('blue','red'), order = 1),
                                     shape = guide_legend(override.aes =
list(size = c(8,8),fill ='black')), order = 2)) +
          ggtitle('Position C') +
          theme(text = element_text(family = 'sans'),
                plot.title = element_text(size = 24, hjust = 0.5),
                legend.title = element_text(size = 14),
                legend.text = element_text(size = 12),
                legend.key = element_rect(fill = 'white'),
                axis.title = element_text(size = 18),
                axis.text = element_text(size = 14),
                axis.ticks = element_line(color = 'black'),
                panel.background = element_rect(fill = 'white', color =
'black'),
                panel.border = element_blank(),
                panel.grid = element_blank()))
dev.off()


##----Position D----
pdf(paste0(output_filepath,'/Cart_data_D.pdf'), width = 10, height = 8)
print(ggplot(data = subset(cart_summary, Location == 'D'), aes(x = mean_x, y
= mean_y)) +
          geom_point(aes(fill = Treatment, shape = Morphant, size =
Morphant), color = 'black', stroke = 0.05) +
          geom_hline(yintercept = 0) +
          geom_vline(xintercept = 0) +
          scale_y_continuous(name = 'y-axis', expand = c(0,0), limits = c(-
165,0), position = 'top') +
          scale_x_continuous(name = 'x-axis', expand = c(0,0), limits =
c(0,255), position = 'left') +
          scale_shape_manual(values = c(22,23)) +
          scale_size_manual(values = c(9,9)) +
          scale_fill_manual(values = c('blue','red')) +
          geom_errorbar(aes(ymin = mean_y-SE_y, ymax = mean_y + SE_y),
                        width = 5, size = 0.75) +
          geom_errorbarh(aes(xmin = mean_x-SE_x, xmax = mean_x + SE_x),
                         height = 5, size = 0.75) +
          guides(fill = guide_legend(override.aes = list(size = 8, color =
c('blue','red'), order = 1),
                                     shape = guide_legend(override.aes =
list(size = c(8,8),fill ='black')), order = 2)) +
          ggtitle('Position D') +
          theme(text = element_text(family = 'sans'),
                plot.title = element_text(size = 24, hjust = 0.5),
                legend.title = element_text(size = 14),
                legend.text = element_text(size = 12),
                legend.key = element_rect(fill = 'white'),
                axis.title = element_text(size = 18),
                axis.text = element_text(size = 14),
                axis.ticks = element_line(color = 'black'),
                panel.background = element_rect(fill = 'white', color =
'black'),
                panel.border = element_blank(),
                panel.grid = element_blank()))
dev.off()
```

```
##----5) Write session information----
writeLines(capture.output({

cat('########################################################################
###\n##  Script and Data Analysis by Cheryl Dunham
##\n##
##\n##  Budding Bioinformatician
##\n##  Tanguay Laboratory | Sinnhuber Aquatic Research Laboratory (SARL)
##\n##  28645 E Hwy 34
##\n##  Corvallis, OR 97333
##\n##  Oregon State University
##\n##
##\n##  (541) 737-6500
##\n##  dunhamcg@gmail.com
##\n#########################################################################
##\n\n')
    cat('Input filepath:', input_filepath, '\n\n')
    cat('Output filepath:', output_filepath,
'\n\n_________________________________________________________________________
_____________\n______________________________________________________________
_______________________\n\n')
    print(sessionInfo())

cat('\n______________________________________________________________________
_______________\n____________________________________________________________
_________________________\n\n\nR Citation Information\n')
    print(citation(package = 'base'))

cat('\n______________________________________________________________________
_______________\n____________________________________________________________
_________________________\n\n\nRStudio Citation Information\n\n')
    print(RStudio.Version())

cat('________________________________________________________________________
_____________\n______________________________________________________________
_______________________\n\n\nAttached Package Citation Information\n\n')
    print(lapply(all_packages, citation))

}),
paste0(output_filepath,'/Morphometric_analysis_sessionInfo_',Sys.Date(),'.txt
'))
```

Table S1 List of all primers and probes

Application	Gene	Forward (5'-3')	Reverse primer (5'-3')
qRT-PCR	β-actin	AAGCAGGAGTACGATGAGTC	TGGAGTCCTCAGATGCATTG
	sox9b	AGACGCAGATCTCCACCAAT	CAGATCCGCTTTACTGCACA
	slincR	GATTACACTCACCCCGCAGC	CAATCCTCAGTGTCCGTCAGT
	cyp1a	TGCCGATTTCATCCCTTTCC	AGAGCCGTGCTGATAGTGTC
qRT-PCR CHART	sense probe	GCGGGTATCCAGATGGATATTATGGGATACACGTGCTTCAGCACTCTCTGGATATCACA/iSp18/3BioTEG/	
	slincR probe	TGTGATATCCAGAGAGTGCTGAAGCACGTGTATCCCATAATATCCATCTGGATACCCGC/iSp18/3BioTEG/	
qRT-PCR	sox9b -2042 promoter	GCTGTGTGCAGAACAGCATT	GCTGGCTTCCACACTCATCT
	sox9b -963 promoter	AGCAGCAGCGGTGTTTATCT	GTGCACTTGCTGACGGATAG
	sox9b -502 promoter	GCAGCTGGGGTGAAGATAGA	CAGCATTGCTCGACAGTCAT
	sox9b 5' UTR +45	TTTTCGGAGTGCTCACACAC	AGCTCCGGACACACTCATCT