5'RNA-seq identifies Fhl1 as a modifier in Hypertrophic Cardiomyopathy
Supplemental Information (Methods, Computational Analyses, and Figures)

Danos C. Christodoulou[1-3], Hiroko Wakimoto[1,4], Kenji Onoue[1], Seda Eminaga[1,5], Joshua M. Gorham[1], Steve R. DePalma[1], Daniel S. Herman[1,2,6], Polakit Teekakirikul[1], David A. Conner[1], David M. McKean[1,7], Andrea A. Domenighetti[8], Anton Aboukhalil[9,10], Stephen Chang[1], Gyan Srivastava[11,12], Barbara McDonough[1], Philip L. De Jager[2,11,12], Ju Chen[8], Martha L. Bulyk[2,9,13,14], Jochen D. Muehlschlegel[15], Christine E. Seidman[1-3,7,16*] & J. G. Seidman[1-3*]

[1]Department of Genetics; [2]Ph.D. Programs in Biological and Biomedical Sciences; [3]Leder Human Biology and Translational Medicine; Harvard Medical School, Boston, Massachusetts 02115 and Harvard Integrated Life Sciences, Graduate School of Arts and Sciences, Cambridge, MA 02138, USA.
[4]Department of Cardiology, Boston Children's Hospital and Harvard Medical School, Boston, MA 02115, USA.
[5]Cardiovascular Division, King's College London and St Thomas' Hospital, London SE1 7EH, United Kingdom.
[6]The Harvard/M.I.T. MD-PhD program, Harvard Medical School, Boston, MA 02115, USA
[7]Division of Cardiovascular Medicine, Brigham and Women's Hospital, Boston, MA 02115, USA.
[8]Department of Medicine, University of California, San Diego, La Jolla, CA, USA.
[9]Division of Genetics, Department of Medicine, Brigham and Women's Hospital and Harvard Medical School, Boston, MA 02115, USA.
[10]Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA
[11]Program in Translational NeuroPsychiatric Genomics, Department of Neurology, Brigham and Women's Hospital and Harvard Medical School, Boston, MA 02115, USA.
[12]Broad Institute of Harvard University and MIT, Cambridge, MA 02139, USA.
[13]Department of Pathology, Brigham and Women's Hospital, Boston, MA 02115, USA.
[14]Harvard-MIT Division of Health Sciences and Technology, Cambridge, MA 02139, USA.
[15]Department of Anesthesiology, Perioperative and Pain Medicine, Brigham and Women's Hospital, Boston, MA 02115, USA.
[16]Howard Hughes Medical Institute, MD 20815, USA.
* Denotes equal contribution.

5’ RNA-Seq molecular biology steps and computational analyses

Below we describe the steps to construct RNA-seq libraries that yield increased proportions of 5’ end sequences and computational analysis steps including a description on identifying genes with altered start-site usage. We also provide an approach for obtaining and sorting data images from the UCSC browser (Supplemental software package).

Library preparation

The steps for construction of libraries from RNA with increased proportions of fragments from 5’ ends of genes (Supplemental Figure 1) are adapted from published methodologies (1, 2) and notably do not include normalization or fragmentation.

Step 1. Isolate RNA. Total RNA can be isolated with Trizol or column-based methods. Trizol has some recognized advantages such as it can isolate RNA molecules of a large size spectrum and results in a high yield of RNA amount.

Step 2. Assess RNA quality. Assay an aliquot of the transcriptome on a gel and assess quality of 18S and 28S using Bioanalyzer (Agilent) or the Tapestation (Agilent). Broken RNAs will have artificial 5’ ends as RNA hydrolysis may also occur at defined places. In addition, broken RNA will result in loss of the real 5’ ends since selection is done from the 3’ end (Step 3). Libraries constructed from broken RNA will have diminished enrichment of reads at 5’ ends.

Step 3. Perform two-rounds of selecting RNAs possessing poly-A tails by annealing them to poly-T oligos bound to beads (Invitrogen) so as to remove rRNA. After annealing the RNAs to the poly-T-beads, wash unbound material, and then elute the polyA species by melting the annealed structures at high temperature. It is crucial that temperatures do not exceed 72^0C to avoid extensive RNA hydrolysis. After 2 rounds of polyA selection, the rRNA should be ~5% of total RNA.

Step 4. Reverse transcribe (RT) mRNAs using random hexamers (Invitrogen). Be sure reaction is complete to reach 5’ ends. Random primers can be of any length. Large random primers (10-20bp) can exhibit higher specificities during annealing. This can result in preference to transcribing low expressing species, as the primer pool annealing to high expressing species would be depleted. While offering such an advantage, using long random primers may skew the measurement of the distribution of transcripts stronger than shorter random primers.

Step 5. Perform double-stranded DNA (dsDNA) synthesis by adding Pol I and RNA-se H to the reacted products from Step 4 and incubate overnight. It is vital that dsDNA synthesis be performed as completely as possible, as inefficiencies will result in loss of 5’ ends. Priming at the end of the fragment is needed to preserve the information at the 5’ end, from either hydrolyzed RNA fragments resulting from RNA-se H digestion or leftover random hexamer from Step 4. Incubating the reaction overnight may aid in completion of the reaction.The resulting dsDNA is a polar molecule with directional information. (Discussed further at Step 9.)

Step 6. Perform end-repair. This step typically uses addition of enzymes such as polymerases and exonucleases serving to obtain dsDNA fragments with no overhangs.

Step 7. Perform Adenosine addition at 3’ ends of both DNA strands and then ligate appropriate next-generation sequencing adapters. The Adenosine overhangs at 3’ ends to prevent self-ligation of dsDNA fragments during adapter ligation. An excess of adapter dimers may further reduce self-ligation events.

Step 8. Perform size selection by fractionating the DNA library in an agarose gel to excise a size of about ~150-400bp. This can be achieved by running an electrophoresis chamber or by using specialized equipment such as Pippin Prep (Sage Science).

Size selection allows for appropriate size molecules for next-generation sequencing and enriches fragments that resulted from annealing of random hexamers near the 5' ends of transcripts. Failure to optimally execute this step can result in substantial loss of the material. Increase in yield can be achieved by making thinner gels or with the Pippin Prep.

Step 9. Perform a uniform amplification of the library with PCR. The number of cycles of the PCR should be carefully selected so that the reaction does not reach saturation, but yields sufficient material for next-generation sequencers. To achieve uniform amplification of all fragments, stop the reaction at the stage of exponential amplification. A qPCR amplification reaction mirroring final amplification conditions can be used to map the amplification of the library. Cycle numbers at the upper range of the exponential phase can be selected for the final amplification. Multiple parallel reactions can be used to provide the necessary amount for sequencing. Using the majority of the library material in this process is important to maximize the complexity of the library. Note that lower than optimal cycles will lead to insufficient material and higher than optimal cycles will skew the data and introduce sequencing errors (such as when the abundance of dNTP becomes limiting).

A library constructed and sequenced with this approach contains directional information owing to the polarity of the cDNA molecule. Sequences will have one of two configurations: 1) when the fragment is sequenced from the end that corresponds to the upstream part of the reference transcript (5'end), the resulting sequence preserves the same strand and direction of the reference transcript. 2) Conversely, when the fragment is sequenced from the end that corresponds to the 3' end of the reference transcript, the resulting sequence is in reverse/complement relationship with the reference transcript. This information provides a signature of 5' end sites (Supplemental Figure 2).

Computational analysis

Gene/transcript definitions: Gene and transcript definitions are retrieved from the UCSC genome browser using the Table browser. Thus, UCSC transcript definitions that are assigned to RefSeq names are used and these definitions are further processed to obtain annotation tables. For example, to evaluate overall gene expression, all exons of isoforms for each gene are merged. A second annotation file is made that maintains the 5' end locations of isoforms of genes to process 5' end differences.

Using these UCSC transcript definitions, we estimate that 8,000 of the 30,000 annotated mouse genes have at least one additional annotated 5' end. The "count_5pr_isoforms.pl" is provided (Supplemental software package).

Expression analysis: Normalize libraries to control for differences in sequencing depth. For similar samples comparing total RNA housekeeping genes are good reference points. For dissimilar samples, a TATA box binding protein gene Tbp can provide the reference gene.

Reads at 5' end regions are expected to correspond to actual transcript levels (1 read:1 transcript relationship). The total reads derived from other regions of transcripts have a dependency on transcript length and possibly RNA structure.

A p-value is computed using Bayesian statistics (3) to compare the read numbers for a gene from two samples. This takes into account the number of reads as evidence and their relative proportion in the assessed

population. When a library is well constructed, this p-value is an accurate reflection of the comparison of the RNA species from the original sample.

Comparison of read-depth distributions at 5' end regions: Reads are first aligned to the genome and transcriptome. The read-depths at every base-pair position at 5' regions of genes are retrieved and compared between samples one gene at a time, after normalization to the respective total gene expression. This comparison calculates the fold change (ratio) of the normalized read-depths and a p-value using the Bayesian statistic. An assessment of a shift in read distribution is made when the calculated fold exceeds a parametrically defined fold-threshold (3-fold is used), and the p-value is lower compared to a p-value threshold (0.01 is used). Thus, changes in start-site usage is defined by quantifying the instances at 5' ends at which there is a change in distribution of reads on the gene locus. The sum of these instances is used as a score, it is tabulated for all genes, and it is interpreted as a positive signal that can recognize shifts in distribution in either direction. Genes with identified changes in distributions are queried visually on the UCSC browser or Integrative Genome Viewer (IGV) from highest to lowest score to visually identify the change in read-depth profiles and to assess whether these changes correspond to changes in 5' end usage. Evidence from UCSC browser sources (ESTs, mRNA annotations, conservation of sequence, and aligned locations of transcripts identified in other species) can be used to support the recognition of unannotated 5' ends.

The Supplemental software package provides a strategy to retrieve data images from the UCSC browser, and for sorting and organization of image data. Obtaining the coordinates of 5' regions using the UCSC browser allows retrieval of information about aligned reads in an automatic approach for multiple genes and for assessment of significance.

Data on 5' end usage is best when there is sufficient sequencing, when the 5' end uses a unique exon, and when the reads are distributed on the gene locus with a strong 5' end distribution. When applied genome-wide, this can provide a new dimension of information for transcriptional data.

Supplemental software package and image data

- This document is provided for the *review process* to display the contents of the Supplemental software package. The package itself can become part of the supplemental materials and/or can be hosted on the Harvard Medical School server.

The current location of the **Supplemental software package** including the manual (README.txt file in the package) is:

http://seidman.med.harvard.edu/fgs/software/RNAseq-5pr-package.tar.gz
(tar.gz is a UNIX file compression format and we can convert to .zip if helpful)

- Similarly, the **Supplemental image data** can become part of the manuscript's supplemental information or can be hosted on the Harvard Medical School server. The current location of the Supplemental image data is:

http://seidman.med.harvard.edu/fgs/software/Suppl-image-data.pptx

Supplemental software package contents

Manual

Version 1.0
January 2013

CONTENTS:

I. RELEASE NOTES

This software package is open-access, open-source, free to use. We ask that you cite associated published article.

*Assumes made Binary/Alignment/Map (BAM) files.

*Approach used here bundles and processes isoforms within a gene name.

*Potential future updates can be found at:
http://seidman.med.harvard.edu/fgs/software/RNAseq-5pr-package.tar.gz

*Includes 'wiggles' tool from Tophat (http://tophat.cbcb.umd.edu/) with a copy of the original license.

*Includes adapted perl module for calculating Bayesian p-value.
Standard version found at:
http://search.cpan.org/~scottzed/Bio-SAGE-Comparison-1.00.

*Includes adapted normwig.pl, countreads.pl, ann_mk.pl, expression_analyze.pl from our previous work described in:
Christodoulou DC, Gorham JM, Kawana M, DePalma SR, Herman DS, Wakimoto H. Quantification of gene transcripts with deep sequencing analysis of gene expression (DSAGE) using 1 to 2 µg total RNA. Curr Protoc Mol Biol. 2011 Jan;Chapter 25:Unit25B.9.

II. INSTALLATIONS

Computer cluster is recommended.

Requirements:
- Unix
- Samtools (http://samtools.sourceforge.net) and Bio-samtools (http://search.cpan.org/~lds/Bio-SamTools/)
- Image Magick (http://www.imagemagick.org/)
- wget (http://www.gnu.org/software/wget) (typically present in unix platforms)
- InsertPicture2011 (http://agentjim.com/MVP/PowerPoint/InsertPicturePowerPoint2011.html) requiring Microsoft Office 2011.
- Chi-square probability module from: http://search.cpan.org/~mikek/Statistics-Distributions-1.02

1. Unpacking will create the following directories:
<LOCATION>/RNAseq-5pr/bin
<LOCATION>/RNAseq-5pr/tables

where <LOCATION> is the directory of unpacked software.

Add to PATH: <LOCATION>/RNAseq-5pr/bin

Make Perl scripts executable:
'chmod +x <LOCATION>/RNAseq-5pr/bin/*'

Make environment variable pointing to the tables folder:
Add 'export RNASEQ5PR=<LOCATION>/RNAseq-5pr/tables'
in ~/.bash_profile or equivalent.

2. Install the required programs and modules.

Add Chi-square and Bayesian p-value modules to PERL5LIB environment.
Adapted bayesian p-value module directory for PERL5LIB is:
'<LOCATION>/RNAseq-5pr/bin/Bio-SAGE-Comparison-1.00/lib'

III. MAKE ANNOTATION FILES
---

1. Following fields are needed in tab-separated format:

- Isoform name
- Chromosome
- Strand
- Transcription start
- Transcription end
- Coding start or blank
- Coding end or blank
- Number of exons
- Exon starts
- Exon ends
- Gene symbol
- Description

*Step 2 will obtain a current annotation file from UCSC in that format. The exact steps may vary according to the organism.

2. Download current tables from UCSC:
- Go to UCSC browser (http://genome.ucsc.edu) and select "Tables" (Table Browser)
- Select the genome and assembly for "genome" and "assembly" fields
- Change "output format" to "selected fields from primary and related tables"
- Enter an "output file", e.g. "hg19"
- All other fields should display as follows:
    - "group": "Genes and Gene Prediction Tracks"
    - "track": "UCSC Genes"
    - "table": "knownGene"
- Click on "get output"

- This will bring the table browser to a new page. Then click on:
"name","chrom","strand","txStart","txEnd","cdsStart","cdsEnd","exonCount", "exonStarts","exonEnds" (from ".knownGene" table)
"geneSymbol", "description" (from ".kgXref" table)
- Click on "get output" to obtain the file.

2.2. For mouse (mm9) or human (hg19) replace all mitochondrial chromosome (chrM) transcripts with ENSEMBLE annotations.
- This will remove all UCSC chrM annotations:
'grep -v chrM hg19_date > hg19_date_2'

chrM ENSEMBLE annotations are included in the tables directory in files:
"mm9_chrM.ens" and "hg19_chrM.ens"

Insert ENSEMBLE chrM annotations as follows:
'cat hg19_chrM.ens >> hg19_date_2'

*Thus, gene definitions are not rigid. Experimenter has control to define appropriate transcript sets according to the organism/experiment. E.g. a

newly discovered/unannotated gene can be included at this stage.

*The new file is hg19_date_2 and can be renamed as needed.

3. Run: 'makeannotations5pr.pl <INPUT_TABLE>'
e.g. 'makeannotations5pr.pl hg19_date_2'

*This command is executing three perl programs sequentially and creates three annotation tables:
- hg19_date_2.ann
- hg19_date_2.5pr-ann
- hg19_date_2.upstinfo

*Included are pre-made tables for mouse (mm9 assembly) and human (hg19 assembly) with the date made in '/RNAseq-5pr/tables'. Note that UCSC updates the annotations periodically.

*"hg19_date_2" is the prefix for the example above. If renaming the prefix at this stage, all three tables must be renamed in parallel.

*Generated tables must be placed in the tables folder matching the RNASEQ5PR location.

*The table used for expression (".ann") consists of a merge of all the exons of the gene's isoforms.

*The 5' table (".5pr-ann") merges all the exons of isoforms preserving as different entries isoforms with different 5' ends.

*Both the expression (".ann") and 5' tables (".5pr-ann") are linked to the original table by the original isoform names used to make them (last or second to last columns)

*.upstinfo file contains a measure of the distance to the nearest upstream gene. This is used to restrict the 5' comparison when another gene is nearby upstream.

*Annotations match complete genome assemblies downloaded from UCSC. If using genome from elsewhere, note that chromosomes with names such as "random", "chrUn" may not be included and isoforms matching those locations will create warning messages. If so, remove such isoforms. E.g., for hg19, this can be used: 'grep -v _hap hg19_date_2 | grep -v random | grep -v chrUn > hg19_date_simpl'.

*The ".5pr-ann" generated file can also be used to calculate the number of initiation starts of genes. Running "count_5pr_isoforms.pl" will display the genes that have at least two initiation starts.
Usage: e.g. 'count_5pr_isoforms.pl hg19_date_2.5pr-ann'

IV. RUN EXPRESSION AND 5' COMPARISONS
---

1. BAM files must be properly arranged in a sample directory and a BAM index must exist as follows:

- Make a directory matching the Sample's name (one word; unique name to differentiate when running multiple samples).
E.g. WT092212 (for wild-type sample run on 9/22/12)

- Place BAM file inside this folder. Give the BAM file a prefix the same as the sample's name:
e.g. "/WT092212/WT092212.bam"

*Having a folder for each sample helps organize all associated sample-specific analysis files in the sample's folder. Having the BAM file match the sample is needed for the analysis and is also useful when uploading the dataset either on UCSC browser or IGV as the name of the BAM names the uploaded data track.

- Make a BAM index file:
'samtools index WT092212.bam' (run inside sample's folder)

2. Compute gene expression profiles for each sample:
- 'countreads.pl <ANNOTATION-PREFIX> <BAM_prefix>'
e.g. 'countreads.pl hg19_date_2 WT092112' (run inside sample's folder)

*This generates gene expression profile in 'genes.counts' which is the total reads on gene loci and it becomes part of the sample's files.

3. Run comparison program. This can be run on at least two samples and will make all pair-wise comparisons for overall expression comparison and start site comparison.

This requires telling the program the annotation prefix and indicate the samples/directories for the comparisons.
Run 'compare5PR.pl <ANNOTATION_PREFIX> <SAMPLE1> <SAMPLE2> <SAMPLE3> etc'.
(run in the same directory where the SAMPLE1 and SAMPLE2 folders are located)

If the folders of these samples are in various locations, the following can be used:
'compare5PR.pl <ANNOTATION_PREFIX> <SAMPLE1> dir:</location-of-th
e-directory> <SAMPLE2> dir:</location-of-the-directory>' etc.

*This will create two files, one for expression and another for 5prime

comparison.

*'compare5PR.pl' can be edited to change the parameters used as needed.

V. OBTAIN AND ORGANIZE DATA IMAGES FROM THE UCSC BROWSER
---

* The start site comparison file will create a score per gene, based on the observed change in read distributions at start sites. Inspect genes with scores to allow elimination of false positives and to identify the region of start site change. Thus all genes with score greater than 20 can be inspected.

This can be done automatically, obtaining genes' read-depth and read alignment profiles as follows:

1. Make a UCSC Bedgraph and BAM custom tracks for every sample.

For a Bedgraph, make a normalized read-depth file:
- 'samtools view <BAM-FILE> >tmp.sam && wiggles tmp.sam cov.wig&& normwig.pl cov.wig <SAMPLE-NAME>' (run inside sample's folder)

Remove intermediate files: 'rm tmp.sam cov.wig'

The resulting zipped file as well as the BAM file can be uploaded on UCSC as custom tracks.

2. Create a session on UCSC Browser with desired settings. Save session as a file and make this file web-accessible. I.e. create a url pointing to this saved session file.
Verify it works using a web browser.

3. Identify the list of genes to be inspected. Sort the 5pr-comparison file descending by score. Then, select all genes with score above 20 or 30 and save these genes as a gene-list in the order of highest score to lowest score. Save in a text file, 1 gene per line. Use the same gene symbol as in the 5pr file as this will be used to access annotation information.

4. Run 'retrieveUCSCimsg.pl' from the software package as follows:
'retrieveUCSCimgs.pl <ANNOTATION-PREFIX> <URL> <GENELIST-input>'

*This will retrieve image data from UCSC and the final output image will be called:
<RANK-POSITION>_<GENE-NAME>.png
*The rank-position information on the file name will be used to prioritize

the gene profiles in Step 5.

5. Use InsertPicture2011 (or equivalent) to insert data images and file name in a Powerpoint file. InsertPicture2011 will insert these data images in the order determined by alphabetical sorting of the file-name.

*In this format, the images can be inspected quickly and notes can be indicated with arrows or text.

*Supplementary Image Data file includes data images retrieved from two UCSC sessions. First session displaying normalized read-depths; second session displaying the aligned reads (BAM file) put together with InsertPicture2011 as described.

VI. QUANTIFY CHANGES
---

1. For each identified gene changing start site, identify the chromosomal coordinates of the region with the start site change (call it start2) to be compared with another start region (call it start1). This can be achieved using the UCSC browser.

Make a text file formatted as follows and tab-separated:
- Header first line with: "Gene","start1","start2".
- Under "Gene" include the gene-symbol. Alternatively, the name of the file with the rank-position can be used, e.g. "Gapdh" or "10_Gapdh".
- Under "start1" place the coordinates of the second start region in the UCSC format such as "chrX:1,111-2,222".
- Under "start2" place the coordinates of the changing start region in the same format as above.

Save file as text.

*The gene-symbol must match the generated tables/annotations as it is used to retrieve strand information.

2. Run 'getreads-regions.pl' as below to compare two samples:
'getreads-regions.pl <ANNOTATION-PREFIX> <FILE-from-step1> <SAMPLE1> <SAMPLE2>'
(run in the same directory where the SAMPLE1 and SAMPLE2 folders are located)

3. Output file includes the number of reference transcript reads for the regions, the normalized fold (extent of start site change), the Chi-square, and the Chi-square p-value.

5primeanalysis.pl

```
#!/usr/bin/perl -w
use strict;
use lib "/groups/seidman/local/Bio-SAGE-Comparison-1.00/lib";
use Bio::SAGE::Comparison;
use Bio::DB::Sam;
use Bio::DB::Bam::Alignment;
use Bio::DB::Bam::Query;
use Bio::DB::Bam::Target;
use Bio::DB::Sam::Segment;
use Bio::DB::Bam::Pileup;
use Bio::DB::Sam::Constants;
use Bio::DB::Bam::AlignWrapper;
use Bio::DB::Bam::ReadIterator;
use Bio::DB::Bam::FetchIterator;

my $consUCSCtable=shift(@ARGV);
my $annfile=shift(@ARGV);
my $upstreamdistfile=shift(@ARGV);

my $minexpr=shift(@ARGV);# in reads, default is 20
my $fold=shift(@ARGV); #default is 3
my $pval=shift(@ARGV); #default 0.01
my $upst_dist=shift(@ARGV); #default is 200
my $downst_dist=shift(@ARGV); #default is 200


open (CONSTABLE, $consUCSCtable) or die "ERROR the UCSC table file $consUCSCtable could
``not be found \n";
open (ANNOTABLE, $annfile) or die "ERROR the UCSC table file $5primeannfile could not be found
``\n";


open(UPST,$upstreamdistfile) or die "the file $upstreamdistfile could not be found\n";
my %upstreaminfo=();
while(<UPST>){
        my $line=$_;chomp($line);my @elements=split(/\t/,$line);
        my $gene=$elements[0];my $txstart=$elements[1];my $dist=$elements[2];
        $upstreaminfo{$gene}{$txstart}{"dist"}=$dist;
}

my %datafolder=();my $outfile="";
while(@ARGV){
        my $sample=shift(@ARGV);my $folder="";
        if ((@ARGV) && ($ARGV[0] =~ /dir:/)){
```

```
		$folder=shift(@ARGV);
		$folder=~s/dir://;
	}
	else {
		$folder="./$sample";
	}
	chomp($folder);
	$datafolder{$sample}=$folder;
	$outfile.=$sample . "_";
}
chop($outfile);$outfile.=".cmp";
open (OUTFILE,">5pr_$outfile.$fold") or die "the outfile $outfile could not be created\n";

my %samplescounts=();
foreach my $sample (sort keys %datafolder){
	#print OUTSAMPLES join("\t",$sample,$datafolder{$sample} ."\n");
	my $genecountfile=$datafolder{$sample} . "/genes.counts";
	open(GENESCOUNTS,$genecountfile) or die "the file $genecountfile could not be opened\n";
	while(<GENESCOUNTS>){
		my $line=$_;chomp($line);
		my @elements=split(/\t/,$line);
		my $gene=$elements[0];my $value=$elements[1];
		$samplescounts{$gene}{$sample}=$value;
	}
	close GENESCOUNTS;
}
my %genenames=();
while (<CONSTABLE>){
	my $line=$_;chomp($line);
	my @elements=split(/\t/,$line);my $count2=-1;
	my $gene=$elements[0];
	my $lines=$elements[7];
	my @assoclines=split(/,/,$lines);
	foreach(@assoclines){
		$genenames{$_}{"gene"}=$gene;
	}
}
my %anngenes=();my $count=0;
while(<ANNOTABLE>){
	$count++;
	my $line=$_;chomp ($line);
	my @elements=split(/\t/,$line);
	my @assoclines=split(/,/,$elements[7]);
	my $gene=$genenames{$assoclines[0]}{"gene"};
	die "ERROR ANNTABLE to genecounts hash inconsistent\n" if (not $gene);
	$anngenes{$gene}{$count}{"strand"}=$elements[2];
```

```
		$anngenes{$gene}{$count}{"txstarts"}=$elements[3];
		$anngenes{$gene}{$count}{"txends"}=$elements[4];
		$anngenes{$gene}{$count}{"exonstarts"}=$elements[5];
		$anngenes{$gene}{$count}{"exonends"}=$elements[6];
		$anngenes{$gene}{$count}{"chr"}=$elements[1];
}

my @samplenames=();my $samplesnumber=0;
my %bamlink=();
foreach my $sample (sort keys %datafolder){
		$samplesnumber++;
		push(@samplenames,$sample);
		my $bamfile=$datafolder{$sample} . "/$sample.bam";
		$bamlink{$sample}=Bio::DB::Sam->new(-bam  =>$bamfile,);
}

print "opening bam files complete\n";

my $outfirstline="Gene\tChr\tStrand\tAnnotatedTxstarts\tExamined range\tTotal range";
my @comparisonelements=();
for(my $pos1=0;$pos1<$samplesnumber;$pos1++){
		my $sample1=$samplenames[$pos1];
		for (my $pos2=$pos1+1;$pos2<$samplesnumber;$pos2++){
			my $sample2=$samplenames[$pos2];
			$outfirstline.= "\t"."score($sample1/$sample2)";
			push(@comparisonelements,"$sample1/$sample2");
		}
}
print OUTFILE $outfirstline. "\n";

my $genecount=0;
foreach my $gene (sort keys %anngenes){
		print $gene . "\n";
		$genecount++;
		my %gene5prdiffs=();
		my $chr="";
		print $genecount . " starts analyzed\n" if (isint($genecount/1000));
		my %genespan1=();my %genespanrestr;my %txstarts=();
		my $genestart;my $geneend;
		my $examinedrange="";my $totalrange=0;my $strand="";
		for my $count (keys %{$anngenes{$gene}}){
			$chr=$anngenes{$gene}{$count}{"chr"};
			my $basecount=0;
			$strand=$anngenes{$gene}{$count}{"strand"};
			my @exonstarts=split(/,/,$anngenes{$gene}{$count}{"exonstarts"});
			my @exonends=split(/,/,$anngenes{$gene}{$count}{"exonends"});
```

```
		my $updist=0;
		if ($strand eq "+"){
			my $txstart=$exonstarts[0];my
``$updist1=0;$updist1=$upstreaminfo{$gene}{$txstart}{"dist"} if
``($upstreaminfo{$gene}{$txstart}{"dist"});
			$updist=$upst_dist if ($updist1>$upst_dist);
			$updist=$updist1/4 if ($updist1<=$upst_dist*4);
			$updist=0 if ($updist1<=$upst_dist);
			$updist=int($updist);
			$txstarts{$anngenes{$gene}{$count}{"txstarts"}}=1;
			my $firstexon=shift(@exonstarts);
			$firstexon=$firstexon-$updist;
			unshift(@exonstarts,$firstexon);
		}
		elsif ($strand eq "-"){
			my $txstart=$exonends[-1];my
``$updist1=0;$updist1=$upstreaminfo{$gene}{$txstart}{"dist"} if
``($upstreaminfo{$gene}{$txstart}{"dist"});
			$updist=$upst_dist if ($updist1>$upst_dist);
			$updist=$updist1/4 if ($updist1<=$upst_dist*4);
			$updist=0 if ($updist1<=$upst_dist);
			$updist=int($updist);
			$txstarts{$anngenes{$gene}{$count}{"txends"}}=1;
			my $lastexon=pop(@exonends);
			$lastexon=$lastexon+$updist;
			push(@exonends,$lastexon);
		}
		else {
			die "strand is not + or -\n";
		}

		foreach(@exonstarts){
			my $exonstart=$_;
			my $exonend=shift(@exonends);
			for (my $i=$exonstart;$i<=$exonend;$i++){
				$genespan1{$i}{"exists"}=1;
			}
		}

		if ($strand eq "+"){
			my $countpos=0;
			foreach my $i (sort { $a <=> $b } keys %genespan1){
				$countpos++;
				last if ($countpos >$downst_dist+$updist);
				$genespanrestr{$i}{"exists"}=1;
			}
```

```
	}
	elsif ($strand eq "-"){
		my $countpos=0;
		foreach my $i (sort { $b <=> $a } keys %genespan1){
			$countpos++;
			last if ($countpos >$downst_dist+$updist);
			$genespanrestr{$i}{"exists"}=1;
		}
	}
	%genespan1=();
}

my @rangestart=();my @rangeend=();

if ($strand eq "+"){
	my $countpos=0;my $prevpos=0;
	foreach my $i (sort { $a <=> $b } keys %genespanrestr){
		$totalrange++;
		$countpos++;
		$genestart=$i if $countpos==1;
		$geneend=$i;
		if ($countpos==1){
			$examinedrange=$i."-";
			push(@rangestart,$i);
		}
		elsif ($prevpos!=$i-1){
			$examinedrange=$examinedrange.$prevpos.",".$i."-";
			push(@rangeend,$prevpos);
			push(@rangestart,$i);
		}
		$prevpos=$i;
	}
	$examinedrange=$examinedrange.$prevpos;
	push(@rangeend,$prevpos);
}
elsif ($strand eq "-"){
	my $countpos=0;my $prevpos=0;
	foreach my $i (sort { $b <=> $a } keys %genespanrestr){
		$totalrange++;
		$countpos++;
		$geneend=$i if $countpos==1;
		$genestart=$i;
		if ($countpos==1){
			$examinedrange=$i."-";
			push(@rangeend,$i);
		}
```

```
                elsif ($prevpos!=$i+1){
                        $examinedrange=$examinedrange.$prevpos.",".$i."-";
                        push(@rangestart,$prevpos);
                        push(@rangeend,$i);
                }
                $prevpos=$i;
        }
        $examinedrange=$examinedrange.$prevpos;
        push(@rangestart,$prevpos);
}
else {
        print "$gene$count does not have either a + or - strand\n";
}
my %genecoverage=();
foreach my $sample (keys %bamlink){
        my $countrange=0;my @list=();
        foreach(@rangestart){
                my $val=$_;$val+=1 if ($val==0);
                my ($coverage) = $bamlink{$sample}->features(-type=>'coverage',-
``seq_id=>$chr,-start=>$val,-end=>$rangeend[$countrange]);
                my @data_points= $coverage->coverage;
                @list=(@list,@data_points);
                $countrange++;
        }
        $genecoverage{$sample}=\@list;
}
for(my $pos1=0;$pos1<$samplesnumber;$pos1++){
        my $sample1=$samplenames[$pos1];
        my @list1=@{$genecoverage{$sample1}};
        my $exprsample1=$samplescounts{$gene}{$sample1};
        for (my $pos2=$pos1+1;$pos2<$samplesnumber;$pos2++){
                my $sample2=$samplenames[$pos2];
                my $samples=$sample1."/".$sample2;
                my @list2=@{$genecoverage{$sample2}};
                my $exprsample2=$samplescounts{$gene}{$sample2};
                #print $gene . "\n";
                next if $exprsample1<$minexpr;#032510_ changed from 100(counts) to
``70(reads)
                next if $exprsample2<$minexpr;#032510_ changed from 100(counts) to
``70(reads)
                my $diffcount=0;my $listcount=0;
                foreach(@list1){
                        my $value1=$_;
                        my $value2=$list2[$listcount];
                        $listcount++;
                        #next if (($value1<=5) && ($value2<=5));#PARAMETER FROM 100
```

```
                                my ($p,$sign) = Bio::SAGE::Comparison::calculate_significance($value1,
``$value2, $exprsample1, $exprsample2, 1 );
                                next if $p>$pval;
                                my $ratio=1;
                                if ($value2 !=0){
                                        $ratio=$value1*$exprsample2/($value2*$exprsample1);
                                }
                                else {
                                        $ratio=15;
                                }
                                $diffcount++ if (($ratio >=$fold) || ($ratio <= 1/$fold));
                        }
                                $gene5prdiffs{$samples}{"count"}=$diffcount;
                }
        }
        my $txstarts="";
        if ($strand eq "+"){
                foreach my $txstart (sort { $a <=> $b } keys %txstarts){
                        $txstarts=$txstarts.$txstart.",";
                }
        }
        elsif ($strand eq "-"){
                foreach my $txstart (sort { $b <=> $a } keys %txstarts){
                        $txstarts=$txstarts.$txstart.",";
                }
        }
        my $printout=join("\t",$gene,$chr,$strand,$txstarts,$examinedrange,$totalrange);
        foreach(@comparisonelements){
                my $samples=$_;
                if (not $gene5prdiffs{$samples}{"count"}){
                        $printout.="\t0";
                }
                else{
                        $printout.="\t" . $gene5prdiffs{$samples}{"count"};
                }
        }
        print OUTFILE $printout . "\n";
}

print "output 5prime comparison file is: 5pr_$outfile.$fold\n";

sub isint{
  my $val = shift;
  return ($val =~ m/^\d+$/);
}
```

ann_mk.pl

```
#!/usr/bin/perl -w
use strict;
use Data::Dumper;
#die "input table needed and species needed (e.g. \'makeanntable.pl mm9UCSCtable.tb mm9\'\n" if
``not ($ARGV[1]);

##this is adapted from analyzing Deep Sequencing Analysis of Gene Expression tags for RNA-seq

my $inputUCSCtable = $ARGV[0];chomp($inputUCSCtable);
my $outputconsolidatedtable=$inputUCSCtable . ".ann";

open (UCSC, $inputUCSCtable) or die "ERROR the UCSC table file $inputUCSCtable could not be
``found \n";
open (NEWTABLE,">$outputconsolidatedtable") or die "error to make file\n";

my %hann1=();

print "reading gene annotation table..\n";
my $UCSCfirstline=<UCSC>;

my $count=1;
while(<UCSC>) {
        $count++;
        my $annline=$_;chomp($annline);
        my @annfileelements=();@annfileelements = split(/\t/, $annline);
        my $ucscname=$annfileelements[0];
        my $gene=$annfileelements[10];
        my $chromosomestrand=$annfileelements[1].$annfileelements[2];
        my $txstart=$annfileelements[3];
        if ($hann1{$gene}){
                my @otherstarts=();my @otherchromosomestrands=();
                for my $entries (keys %{$hann1{$gene}}){
                        my @othergeneelements=@{$hann1{$gene}{$entries}};
                        push (@otherstarts,$othergeneelements[3]);
                        my $otherchrstrand=$othergeneelements[1].$othergeneelements[2];
                        push (@otherchromosomestrands,$otherchrstrand);
                }
                my $varcount=0;
                my $differentann=0;my $dist=0;
                foreach (@otherstarts){
                        my $othertxstart=$_;
                        my $otherchromosomestrand=$otherchromosomestrands[$varcount];
                        $varcount++;
```

```
                        if (($chromosomestrand eq $otherchromosomestrand) && (abs($othertxstart-
``$txstart) > 5000000)){
                                $differentann=1;
                                $dist=abs($othertxstart-$txstart);
                        }
                }
                if ($differentann==1){
                        $gene=$gene."txstart$txstart";
                        print "inconsistency found and corrected: $dist $gene\n";
                }
        }
        $hann1{$gene}{$ucscname}=\@annfileelements;
}

print "reading gene annotation table complete with $count lines.\n";
print "processing entries..\n";


$count=0;
for my $gene (keys %hann1){
   my %genetemp=();
   for my $entries (keys %{$hann1{$gene}}){
                $count++;
                print "$count isoforms processed\n" if isint($count/1000);
                my @entry=@{$hann1{$gene}{$entries}};
                my $chromosomestrand=$entry[1].$entry[2];
                $genetemp{$chromosomestrand}{"starts"}{$entry[3]}=1;
                $genetemp{$chromosomestrand}{"ends"}{$entry[4]}=1;
                $genetemp{$chromosomestrand}{"strand"}=$entry[2];
                $genetemp{$chromosomestrand}{"exonstarts"}="" if (not
``$genetemp{$chromosomestrand}{"exonstarts"});
                $genetemp{$chromosomestrand}{"exonends"}="" if (not
``$genetemp{$chromosomestrand}{"exonends"});
                $genetemp{$chromosomestrand}{"exonstarts"}.=$entry[8];
                $genetemp{$chromosomestrand}{"exonends"}.=$entry[9];
                $genetemp{$chromosomestrand}{"chr"}=$entry[1];
                $genetemp{$chromosomestrand}{"description"}="" if (not
``$genetemp{$chromosomestrand}{"description"});
                my $description=$entry[11];($description)=$description=~m/(.*)isoform/ if
``($description=~/isoform/);$description=~s/\s+$//g;$description=~s/,$//;$description=~s/,\smRNA.$//;
                $genetemp{$chromosomestrand}{"description"}=$description;
                $genetemp{$chromosomestrand}{"line"}="" if (not
``$genetemp{$chromosomestrand}{"line"});
                $genetemp{$chromosomestrand}{"line"}.=$entries.",";
        }
        for my $chromosomestrand (keys %genetemp){
```

```
my @startsarray=();my @endsarray=();
for my $starts (keys %{$genetemp{$chromosomestrand}{"starts"}}){
	push (@startsarray,$starts);
}
$genetemp{$chromosomestrand}{"startarray"}=\@startsarray;
for my $ends (keys %{$genetemp{$chromosomestrand}{"ends"}}){
	push (@endsarray,$ends);
}
@startsarray = sort { $a <=> $b } @startsarray;
@endsarray = sort { $a <=> $b } @endsarray;
$genetemp{$chromosomestrand}{"endsarray"}=\@endsarray;
my $firststart=$startsarray[0];my $lastend=$endsarray[-1];
#print "\@startsarray is ", Dumper (\@startsarray);
#print "\@endsarray is ", Dumper (\@endsarray);
#print "firststart $firststart $lastend\n";
my @exonstarts=split(/,/,$genetemp{$chromosomestrand}{"exonstarts"});
my @exonends=split(/,/,$genetemp{$chromosomestrand}{"exonends"});
my %genespan=();my $exoncount=0;
foreach (@exonstarts){
	my $exonstart=$_;
	my $exonend=$exonends[$exoncount];
	$exoncount++;
	for (my $i=$exonstart;$i<=$exonend;$i++){
		$genespan{$i}=1;
	}
}
my $prev=0;
my @consstarts=();my @consends=();my $tempvalue=0;
for (my $i=$firststart;$i<=$lastend+1;$i++){
	if (not $genespan{$i}){
		$tempvalue=0;
	}
	else{
		$tempvalue=1;
	}
	push (@consstarts,$i) if (($prev==0) && ($tempvalue==1));
	push (@consends,$i-1) if (($prev==1) && ($tempvalue==0));
	$prev=0 if ($tempvalue==0);
	$prev=1 if ($tempvalue==1);
}
%genespan=();
my $txstarts="";my $txends="";my $exonstarts="";my $exonends="";

foreach (@{$genetemp{$chromosomestrand}{"startarray"}}){
	$txstarts=$txstarts.",".$_;
}
```

```
		$txstarts=substr($txstarts,1);

		foreach (@{$genetemp{$chromosomestrand}{"endsarray"}}){
			$txends=$txends.",".$_;
		}
		$txends=substr($txends,1);

		foreach (@consstarts){
			$exonstarts=$exonstarts.",".$_;
		}
		$exonstarts=substr($exonstarts,1);

		foreach (@consends){
			$exonends=$exonends.",".$_;
		}
		$exonends=substr($exonends,1);

		if (keys %genetemp == 1){
			#print join
``("\t",$gene,$genetemp{$chromosomestrand}{"chr"},$genetemp{$chromosomestrand}{"strand"},$txst
``arts,$txends,$exonstarts,$exonends."\n");
			print NEWTABLE join
``("\t",$gene,$genetemp{$chromosomestrand}{"chr"},$genetemp{$chromosomestrand}{"strand"},$txst
``arts,$txends,$exonstarts,$exonends,$genetemp{$chromosomestrand}{"line"},$genetemp{$chromos
``omestrand}{"description"}."\n");
		}
		else{
			my $newgene=$gene.$chromosomestrand;
			print NEWTABLE join
``("\t",$newgene,$genetemp{$chromosomestrand}{"chr"},$genetemp{$chromosomestrand}{"strand"},
``$txstarts,$txends,$exonstarts,$exonends,$genetemp{$chromosomestrand}{"line"},$genetemp{$chro
``mosomestrand}{"description"}."\n");
			#print  join
``("\t",$newgene,$genetemp{$chromosomestrand}{"chr"},$genetemp{$chromosomestrand}{"strand"},
``$txstarts,$txends,$exonstarts,$exonends."\n");

		}
	}
	delete $hann1{$gene};
	%genetemp=();
}

sub isint{
  my $val = shift;
  return ($val =~ m/^\d+$/);
}
```

```
print "$outputconsolidatedtable complete\n";
```

ann5prime_mk.pl

```
#!/usr/bin/perl -w
use strict;
use Data::Dumper;

my $inputUCSCtable = $ARGV[0];chomp($inputUCSCtable);
die "input table needed\n" if not ($inputUCSCtable);
my $outputconsolidatedtable=$inputUCSCtable . ".5pr-ann";

open (UCSC, $inputUCSCtable) or die "ERROR the UCSC table file $inputUCSCtable could not be
``found \n";
open (NEWTABLE,">$outputconsolidatedtable") or die "error to make file\n";

my %hann1=();

print "reading gene annotation table..\n";
my $UCSCfirstline=<UCSC>;

my $count=1;
while(<UCSC>) {
        $count++;
        my $annline=$_;chomp($annline);
        my @annfileelements=();@annfileelements = split(/\t/, $annline);
        my $gene=$annfileelements[10];
        my $chromosomestrand=$annfileelements[1].$annfileelements[2];
        my $txstart=$annfileelements[3];
        $hann1{$gene}{$annfileelements[0]}=\@annfileelements;
}
print "making $outputconsolidatedtable\n";
print "reading gene annotation table complete with $count lines.\n";
print "processing entries..\n";


$count=0;
for my $gene (keys %hann1){
        my %genetemp=();
   for my $entries (keys %{$hann1{$gene}}){
                $count++;
                print "$count isoforms processed\n" if isint($count/1000);
                my @entry=@{$hann1{$gene}{$entries}};
                my $chromosomestrand1st2ndexon=$entry[1].$entry[2];
                if ($entry[2] eq "+"){
                        my @exons=split(/,/,$entry[8]);
                        $chromosomestrand1st2ndexon=$chromosomestrand1st2ndexon.$exons[0];
                        if (exists $exons[1]){
```

```
	$chromosomestrand1st2ndexon=$chromosomestrand1st2ndexon.$exons[1];
			}
		}
		else {
			my @exons=split(/,/,$entry[9]);
			$chromosomestrand1st2ndexon=$chromosomestrand1st2ndexon.$exons[-1];
			if (exists $exons[-2]){

	$chromosomestrand1st2ndexon=$chromosomestrand1st2ndexon.$exons[-2];
			}
		}
		$genetemp{$chromosomestrand1st2ndexon}{"starts"}{$entry[3]}=1;
		$genetemp{$chromosomestrand1st2ndexon}{"ends"}{$entry[4]}=1;
		$genetemp{$chromosomestrand1st2ndexon}{"strand"}=$entry[2];
		$genetemp{$chromosomestrand1st2ndexon}{"exonstarts"}="" if (not
``$genetemp{$chromosomestrand1st2ndexon}{"exonstarts"});
		$genetemp{$chromosomestrand1st2ndexon}{"exonends"}="" if (not
``$genetemp{$chromosomestrand1st2ndexon}{"exonends"});

	$genetemp{$chromosomestrand1st2ndexon}{"exonstarts"}=$genetemp{$chromosomestrand1
``st2ndexon}{"exonstarts"}.$entry[8];

	$genetemp{$chromosomestrand1st2ndexon}{"exonends"}=$genetemp{$chromosomestrand1s
``t2ndexon}{"exonends"}.$entry[9];
		$genetemp{$chromosomestrand1st2ndexon}{"chr"}=$entry[1];
		$genetemp{$chromosomestrand1st2ndexon}{"line"}="" if (not
``$genetemp{$chromosomestrand1st2ndexon}{"line"});

	$genetemp{$chromosomestrand1st2ndexon}{"line"}=$genetemp{$chromosomestrand1st2ndex
``on}{"line"}.$entries.",";
	}
	for my $chromosomestrand (keys %genetemp){
		my @startsarray=();my @endsarray=();
		for my $starts (keys %{$genetemp{$chromosomestrand}{"starts"}}){
			push (@startsarray,$starts);
		}
		$genetemp{$chromosomestrand}{"startarray"}=\@startsarray;
		for my $ends (keys %{$genetemp{$chromosomestrand}{"ends"}}){
			push (@endsarray,$ends);
		}
		@startsarray = sort { $a <=> $b } @startsarray;
		@endsarray = sort { $a <=> $b } @endsarray;
		$genetemp{$chromosomestrand}{"endsarray"}=\@endsarray;
		my $firststart=$startsarray[0];my $lastend=$endsarray[-1];
		#print "\@startsarray is ", Dumper (\@startsarray);
```

```
#print "\@endsarray is ", Dumper (\@endsarray);
#print "firststart $firststart $lastend\n";
my @exonstarts=split(/,/,$genetemp{$chromosomestrand}{"exonstarts"});
my @exonends=split(/,/,$genetemp{$chromosomestrand}{"exonends"});
my %genespan=();my $exoncount=0;
foreach (@exonstarts){
	my $exonstart=$_;
	my $exonend=$exonends[$exoncount];
	$exoncount++;
	for (my $i=$exonstart;$i<=$exonend;$i++){
		$genespan{$i}=1;
	}
}
my $prev=0;
my @consstarts=();my @consends=();my $tempvalue=0;
for (my $i=$firststart;$i<=$lastend+1;$i++){
	if (not $genespan{$i}){
		$tempvalue=0;
	}
	else{
		$tempvalue=1;
	}
	push (@consstarts,$i) if (($prev==0) && ($tempvalue==1));
	push (@consends,$i-1) if (($prev==1) && ($tempvalue==0));
	$prev=0 if ($tempvalue==0);
	$prev=1 if ($tempvalue==1);
}
%genespan=();
my $txstarts="";my $txends="";my $exonstarts="";my $exonends="";

foreach (@{$genetemp{$chromosomestrand}{"startarray"}}){
	$txstarts=$txstarts.",".$_;
}
$txstarts=substr($txstarts,1);

foreach (@{$genetemp{$chromosomestrand}{"endsarray"}}){
	$txends=$txends.",".$_;
}
$txends=substr($txends,1);

foreach (@consstarts){
	$exonstarts=$exonstarts.",".$_;
}
$exonstarts=substr($exonstarts,1);

foreach (@consends){
```

```
                $exonends=$exonends.",".$_;
            }
            $exonends=substr($exonends,1);

            if (keys %genetemp == 1){
                #print join
``("\t",$gene,$genetemp{$chromosomestrand}{"chr"},$genetemp{$chromosomestrand}{"strand"},$txst
``arts,$txends,$exonstarts,$exonends."\n");
                print NEWTABLE join
``("\t",$gene,$genetemp{$chromosomestrand}{"chr"},$genetemp{$chromosomestrand}{"strand"},$txst
``arts,$txends,$exonstarts,$exonends,$genetemp{$chromosomestrand}{"line"}."\n");
            }
            else{
                my $newgene=$gene;#.$chromosomestrand;
                print NEWTABLE join
``("\t",$newgene,$genetemp{$chromosomestrand}{"chr"},$genetemp{$chromosomestrand}{"strand"},
``$txstarts,$txends,$exonstarts,$exonends,$genetemp{$chromosomestrand}{"line"}."\n");
                #print  join
``("\t",$newgene,$genetemp{$chromosomestrand}{"chr"},$genetemp{$chromosomestrand}{"strand"},
``$txstarts,$txends,$exonstarts,$exonends."\n");

            }
        }
        delete $hann1{$gene};
        %genetemp=();
}
print "$outputconsolidatedtable complete\n";

sub isint{
  my $val = shift;
  return ($val =~ m/^\d+$/);
}
```

compare5PR.pl

```
#!/usr/bin/perl -w
use strict;

die "usage is compare5PR.pl <ANNOTATION_PREFIX> <SAMPLE1> <SAMPLE2> ... and at least 2
``samples are needed\n" if (not $ARGV[2]);

my $annot_prefix=shift(@ARGV);
my $input=join(" ",@ARGV);chomp($input);
my $annot_dir=$ENV{RNASEQ5PR} . "/";

my $ann_5pr=$annot_dir . $annot_prefix . ".5pr-ann";
my $ann_expr=$annot_dir . $annot_prefix . ".ann";
my $upst_file=$annot_dir . $annot_prefix . ".upstinfo";


#below are default parameters for the 5' comparison
my $minexpr=20; #in reads, default 20
my $fold=3;#default 3
my $pval=0.01;#default 0.01
my $upst_dist=200;#default 200
my $downst_dist=200; #default is 200

die "ERROR\n" if (system ("expression_analyze.pl $ann_expr $input && 5primeanalysis.pl $ann_expr
``$ann_5pr $upst_file $minexpr $fold $pval $upst_dist $downst_dist $input"));
```

count_5pr_isoforms.pl

```
#!/usr/bin/perl -w
use strict;

my $infile=$ARGV[0];chomp($infile);
open(IN,$infile) or die "$infile could not open\n";

my %genes=();
while(<IN>){
	my $line=$_;chomp($line);my @elements=split(/\t/,$line);
	my $gene=$elements[0];
	$genes{$gene}=0 if (not $genes{$gene});
	$genes{$gene}++;
}

my $count=0;
foreach my $gene (keys %genes){
	$count++ if $genes{$gene}>1;
	print $gene . "\t$genes{$gene}\n" if ($genes{$gene}>1);
}
print "The number of genes with at least two initiation starts is $count.\n";
```

countreads.pl

```
#!/usr/bin/perl -w
use strict;
use Bio::DB::Sam;

###061910spliced reads flanking but not aligning to an exon are not counted

die "input annotation prefix and bam prefix are needed\n" if (not $ARGV[1]);
my $bamprefix=$ARGV[1];
my $annfile=$ARGV[0];chomp($bamprefix);

my $tablesloc=$ENV{RNASEQ5PR};

$annfile="$tablesloc/" . $annfile . ".ann";

my $folder=$ARGV[2];
if (not $folder){
	$folder=".";
}
else{
	chomp($folder);
}
open (ANNOTABLE,$annfile) or die "the annotation file $annfile could not be found\n";

my %anngenes=();
while(<ANNOTABLE>){
	my $line=$_;chomp ($line);
	my @elements=split(/\t/,$line);
	my $gene=$elements[0];
	die "gene $gene exists in ann table\n" if exists $anngenes{$gene};
	$anngenes{$gene}{"chr"}=$elements[1];
	$anngenes{$gene}{"exonstarts"}=$elements[5];
	$anngenes{$gene}{"exonends"}=$elements[6];
	my $strand=$elements[2];
	$anngenes{$gene}{"strand"}=$strand;
	if ($strand eq "+"){
		$anngenes{$gene}{"txstart"}=$elements[3];
	}
	else {
		$anngenes{$gene}{"txstart"}=$elements[4];
	}
}

my $outfile=$folder . "/genes.counts";
my $outexonsfile=$folder . "/genes.exons.counts";
```

```
my $bamfile=$folder . "/". $bamprefix .".bam";
my $sam = Bio::DB::Sam->new(-bam  =>$bamfile,);
#my $outcoordinate_file= $folder . "/genes.exons.info";

open (OUT,">$outfile") or die "the $outfile could not be written\n";
open (OUTEXONREADS,">$outexonsfile") or die "the $outexonsfile could not be written\n";
#open (OUTEXONSINFO,">$outcoordinate_file") or die "the file $outcoordinate_file could not be
written\n";

foreach my $gene (sort keys %anngenes){
        my $chr=$anngenes{$gene}{"chr"};
        my $strand=$anngenes{$gene}{"strand"};
        my @exonstarts=split(/,/,$anngenes{$gene}{"exonstarts"});
        my @exonends=split(/,/,$anngenes{$gene}{"exonends"});
        my %qnames=();

        my $numberofexons=@exonstarts;

        my $exoncount=0;my $outexonline=$gene;my $outexoninfo=$gene;
        foreach(@exonstarts){
                my $exonnumber=0;
                if ($strand eq "+"){
                        $exonnumber=$exoncount+1;
                }
                elsif($strand eq "-"){
                        $exonnumber=$numberofexons - $exoncount;
                }
                my $exonreadcounts=0;
                my $exonstart=$_;
                my $exonend=$exonends[$exoncount];
                my $outexoninfo.="\t$chr" . ":$exonstart"."-$exonend";
                my @alignments = $sam->get_features_by_location(-seq_id => $chr,
                                        -start  => $exonstart,
                                        -end    => $exonend);
                foreach(@alignments){
                my $alignment = $_;
                my $start       = $alignment->start;
                my $end         = $alignment->end;
                next if (($start<$exonstart) && ($end>$exonend));
                my $qname= $alignment->qname;
                        $qnames{$qname}=1;
                        $exonreadcounts++;
                }
                $exoncount++;
                $outexonline.="\t". join
``(":",$anngenes{$gene}{"txstart"},$exonnumber,$chr,$exonstart,$exonend,$exonreadcounts);
```

```
        }
        my $genereadcounts=keys %qnames;
        print OUT $gene . "\t" . $genereadcounts . "\n";
        print OUTEXONREADS $outexonline . "\n";
        #print OUTEXONSINFO $outexonsinfo . "\n";
}
```

expression_analyze.pl

```
#!/usr/bin/perl -w
use strict;
#use lib "/groups/seidman/local/Bio-SAGE-Comparison-1.00/lib";

use Bio::SAGE::Comparison;

my $annfile=shift(@ARGV);

open(ANN,$annfile) or die "annotation file $annfile could not be found\n";
my %description=();
while(<ANN>){
	my $line=$_;chomp($line);my @elements=split(/\t/,$line);
	my $gene=$elements[0];
	my @starts=split(/,/,$elements[3]);my $start=$starts[0];
	my @ends=split(/,/,$elements[4]);my $end=$ends[-1];
	$elements[8]="" if (not $elements[8]);
	my $description=join("\t",$elements[1],$elements[2],$start,$end,$elements[8]);
	$description{$gene}=$description;
#	$description{$gene}{"chr"}=$elements[1];
#	$description{$gene}{"strand"}=$elements[2];
#	$description{$gene}{"start"}=$elements[3];
#	$description{$gene}{"end"}=$elements[4];
#	$description{$gene}{"description"}=$elements[8];
}
my $firstline_description="Chr\tStr\tStart\tEnd\tDescription";

die "no input samples/files provided" if (not @ARGV);
my %tophatfolder=();my $outfile="";
while(@ARGV){
	my $sample=shift(@ARGV);my $folder="";
	if ((@ARGV) && ($ARGV[0] =~ /dir:/)){
		$folder=shift(@ARGV);
		$folder=~s/dir://;
	}
	else {
		$folder="./$sample";
	}
	chomp($folder);
	$tophatfolder{$sample}=$folder;
	$outfile.=$sample . "_";
}
chop($outfile);$outfile.=".expr";

open (OUTFILE,">$outfile") or die "the outfile $outfile could not be created\n";
```

```
my %allcounts=();my %sampletotals=();my @samplenames=();my
``$outfirstline="Gene\t$firstline_description";my $outfirstlinepart2="";my @samplestotalreads=();
foreach my $sample (sort keys %tophatfolder){
	my $genecountfile=$tophatfolder{$sample} . "/genes.counts";
	open(GENESCOUNTS,$genecountfile) or die "the file $genecountfile could not be opened\n";
	my $genetotalreads=0;
	while(<GENESCOUNTS>){
		my $line=$_;chomp($line);
		my @elements=split(/\t/,$line);
		my $gene=$elements[0];my $value=$elements[1];
		my $description=$description{$gene};my @descrel=split(/\t/,$description);my
``$chr=$descrel[0];#next if ($chr ne "chrM");
		$allcounts{$gene}{$sample}=$value;
		$genetotalreads+=$value;
	}
	$sampletotals{$sample}=$genetotalreads;
	push(@samplestotalreads,$genetotalreads);
	push(@samplenames,$sample);
	$outfirstline=$outfirstline."\t$sample(norm)";
	$outfirstlinepart2=$outfirstlinepart2."\t$sample(reads)";
	close GENESCOUNTS;
}
$outfirstline.=$outfirstlinepart2;

my $totalsamples=@samplenames;
while (@samplenames){
	my $sample=shift(@samplenames);
	$totalsamples--;
	for (my $i=0;$i<$totalsamples;$i++){
		my $pair=$samplenames[$i];
		$outfirstline.="\tfold($sample/$pair)\tpvalue($sample/$pair)";
		#print $sample . $pair."\n";
	}
}
#print "expression comparison output is $outfile\n";
print OUTFILE $outfirstline."\n";

my %comparisons=();
foreach my $gene (sort keys %allcounts){
	#print $gene . "\n";
	my @dataarray=();
	for my $sample (sort keys %{$allcounts{$gene}}){
		push (@dataarray,$allcounts{$gene}{$sample});
	}
	my $datacount=@dataarray;
```

```
        die "data count in array is less than number of samples\n" if ($datacount != keys
``%tophatfolder);
        my $pos1=0;my @ps=();my @signs=();
        while (@dataarray){
                my $data1=shift(@dataarray);
                my @pair=();push(@pair,$data1);
                for (my $i=0;$i<$datacount-1;$i++){
                        my $data2=$dataarray[$i];
                        my ($p,$sign) = Bio::SAGE::Comparison::calculate_significance($data1, $data2,
``$samplestotalreads[$pos1], $samplestotalreads[$pos1+$i+1], 1 );
                        push(@ps,$p);push(@signs,$sign);
                }
                $pos1++;
                $datacount--;
        }
        $comparisons{$gene}{"pvalues"}=\@ps;
        $comparisons{$gene}{"signs"}=\@signs;
}
my $pvaluetest=1/(keys %allcounts);

foreach my $gene (sort keys %allcounts){
        #print $gene . "\n";
        my @normdataarray=();my @dataarray=();
        my $genefold="";
        my $printout=$gene."\t$description{$gene}";my $printout2="";
        for my $sample (sort keys %{$allcounts{$gene}}){
                my $normvalue=$allcounts{$gene}{$sample}*1000000/$sampletotals{$sample};
                $printout.="\t".$normvalue;
                $printout2.="\t".$allcounts{$gene}{$sample};
                push (@normdataarray,$normvalue);
                push (@dataarray,$allcounts{$gene}{$sample});
        }
        $printout.=$printout2;
        my $datacount=@dataarray;
        die "data count in array is less than number of samples\n" if ($datacount != keys
``%tophatfolder);

        my @folds=();
        while (@dataarray){
                my $normdata1=shift(@normdataarray);
                my $data1=shift(@dataarray);
                my @pair=();push(@pair,$normdata1);
                for (my $i=0;$i<$datacount-1;$i++){
                        my $normdata2=$normdataarray[$i];
                        my $data2=$dataarray[$i];
                        my $fold="";
```

```
                        if (($data2<10)&&($data1<10)){
                                $fold="nc";
                        }
                        elsif (($normdata2==0)&&($normdata1!=0)){
                                $fold=999;
                        }
                        else {
                                $fold=$normdata1/$normdata2;
                        }
                        push(@folds,$fold);
                }
                $datacount--;
        }
        my @ps=@{$comparisons{$gene}{"pvalues"}};
        my @signs=@{$comparisons{$gene}{"signs"}};
        my $arrayelement=0;my $calculations="";
        foreach(@folds){
                my $fold=$_;
                $fold="" if ($fold eq "nc");
                my $p=$ps[$arrayelement];
                my $sign=$signs[$arrayelement];
                $arrayelement++;
                $calculations=$calculations."\t$fold\t$p";
        }
        print OUTFILE $printout .$calculations."\n";
}
print "expression comparison output is $outfile\n";
```

findupstream.pl

```
#!/usr/bin/perl -w
use strict;


die "perl findupstreamdist.pl 5primeannfile GEannfile\n" if (not $ARGV[1]);
#open(ANNCONS,"/groups/seidman/dcc13/DSAGE/tables/mm9.ann") or die "mm9.ann could not be
``opened\n";
my $annfile=$ARGV[0];my $anncons=$ARGV[1];
chomp($anncons);my $nameprefix=$ARGV[2];chomp($nameprefix);

#my $annfile="/groups/seidman/dcc13/tables/hg19_table_082510_enschrM.tb_5primetable.tb";
open(ANN5PRIME,$annfile) or die "file $annfile could not be opened\n";
open(ANNCONS,$anncons) or die "$anncons could not be opened\n";
my $outfile=$nameprefix . ".upstinfo";
print "now making $outfile from processed tables\n";
open(OUTFILE,">$outfile") or die "file $outfile could not be created\n";
my %assoc_genes=();
while(<ANNCONS>){
        my $line=$_;chomp($line);
        my @elements=split(/\t/,$line);
        my $gene=$elements[0];
        my @alines=split(/,/,$elements[7]);
        foreach(@alines){
                $assoc_genes{$_}=$gene;
        }
}

my %genes=();
while(<ANN5PRIME>){
        my $line=$_;chomp($line);
        my @elements=split(/\t/,$line);
        my $chr=$elements[1];
        my $gene=$elements[0];
        my $strand=$elements[2];
        my $txstart=0;
        if ($strand eq "+"){
                $txstart=$elements[3];
        }
        else {
                $txstart=$elements[4];
        }
        my $key=$gene."_".$txstart;
        my @exonstarts=split(/,/,$elements[5]);
        my @exonends=split(/,/,$elements[6]);
```

```
        my @alines=split(/,/,$elements[7]);
        my $genename=$assoc_genes{$alines[0]};
        my @txstarts=split(/,/,$elements[3]);
        my @txends=split(/,/,$elements[4]);
        $genes{$chr}{$key}{"txstart"}=$txstarts[0];##For sorting purposes
        $genes{$chr}{$key}{"txend"}=$txends[-1];
        $genes{$chr}{$key}{"strand"}=$elements[2];
        $genes{$chr}{$key}{"elements"}=\@elements;
        $genes{$chr}{$key}{"exonstarts"}=\@exonstarts;
        $genes{$chr}{$key}{"exonends"}=\@exonends;
        $genes{$chr}{$key}{"genename"}=$genename;
        $genes{$chr}{$key}{"txstart_tr"}=$txstart;
}

my $count=0;
for my $chr (keys %genes){
        for my $gene (keys %{$genes{$chr}}){
                my $dist=50000;$dist=0 if ($chr eq "chrM");
                my $genename=$genes{$chr}{$gene}{"genename"};
                my $txstart=$genes{$chr}{$gene}{"txstart_tr"};
                my $strand=$genes{$chr}{$gene}{"strand"};
                my $txend=$genes{$chr}{$gene}{"txend"};########QUESTIONABLE
                my @elements=@{$genes{$chr}{$gene}{"elements"}};
                for my $genecmp (sort {$genes{$chr}{$a}{"txstart"} <=> $genes{$chr}{$b}{"txstart"}}
``keys %{$genes{$chr}}){
                        last if $dist==0;
                        next if ($genes{$chr}{$genecmp}{"txend"} < $txstart-5000);
                        my $genecmpname=$genes{$chr}{$genecmp}{"genename"};
                        next if ($genename eq $genecmpname);
                        last if ($genes{$chr}{$genecmp}{"txstart"} > ($txstart+5000));
                        my @exonstartscmp=@{$genes{$chr}{$genecmp}{"exonstarts"}};
                        my @exonendscmp=@{$genes{$chr}{$genecmp}{"exonends"}};
                        my $exoncountcmp=0;
                        foreach(@exonstartscmp){
                                my $exonstartcmp=$_;my
``$exonendcmp=$exonendscmp[$exoncountcmp];
                                $exoncountcmp++;
                                if ($strand eq "+"){
                                        last if $txstart < $exonstartcmp;
                                        if (($txstart <=$exonendcmp) && ($txstart >= $exonstartcmp)){
                                                $dist=0;last;
                                        }
                                        my $tempdist=$txstart-$exonendcmp;
                                        $dist=$tempdist if ($tempdist < $dist);
                                }
```

```
                        if ($strand eq "-"){
                                next if $txend > $exonendcmp;
                                last if $txend < ($exonstartcmp-5000);
                                if (($txend <=$exonendcmp) && ($txend >= $exonstartcmp)){
                                        $dist=0;last;
                                }
                                my $tempdist=$exonstartcmp - $txend;
                                $dist=$tempdist if ($tempdist < $dist);
                        }
                }
        }
        print OUTFILE join("\t",$genename,$txstart,$dist) . "\n";
        $count++;
        print $count . "\n" if isint($count/1000);
    }
}
print "complete\n";

sub isint{
  my $val = shift;
  return ($val =~ m/^\d+$/);
}
```

getreads-regions.pl

```
#!/usr/bin/perl -w
use strict;
use lib "/groups/seidman/local/Bio-SAGE-Comparison-1.00/lib";
use Bio::SAGE::Comparison;
use Bio::DB::Sam;
use lib "/groups/seidman/dcc13/Statistics-Distributions-1.02";
use Distributions;


die "usage is getreads-regions.pl ANN-PREFIX COORD-LIST SAMPLE1 SAMPLE\n" if (not
``$ARGV[3]);

my $annot_prefix=shift(@ARGV);
my $annot_dir=$ENV{RNASEQ5PR} . "/";
my $annfile=$annot_dir . $annot_prefix . ".ann";

my $inregionfile=shift(@ARGV);
my $sample1=shift(@ARGV);
my $sample2=shift(@ARGV);chomp($sample2);
my @samples=($sample1,$sample2);

my $outfile="$sample1.$sample2.regionreads.txt";
open(OUT,">$outfile");

open(ANN,$annfile) or die "could not open $annfile\n";
my %strands=();
while(<ANN>){
        my $line=$_;chomp($line);my @el=split(/\t/,$line);
        $strands{$el[0]}=$el[2];
}

open(REGIONS,$inregionfile) or die "$inregionfile could not open\n";
my $firstline_rg=<REGIONS>;chomp($firstline_rg);my @spl_names=split(/\t/,$firstline_rg);my
$reg1n=$spl_names[1];my $reg2n=$spl_names[2];

print OUT join("\t","Gene",$sample1."-".$reg1n,$sample1."-".$reg2n,$sample2."-".$reg1n,$sample2."-
``".$reg2n,"Fold","Chi-sq","Chi-sq-prob")."\n";
while(<REGIONS>){
        my $line=$_;chomp($line);my @elements=split(/\t/,$line);
        my $gene=$elements[0];my $gene_symb=$gene;
        my @convtmp=split(/_/,$gene);shift(@convtmp);##NEEDS SILENCING
        $gene_symb=join("_",@convtmp) if @convtmp>1;$gene_symb=$convtmp[0] if
``(@convtmp==1);
```

```
	my $strand;if ($strands{$gene_symb} eq "+"){$strand=1}elsif($strands{$gene_symb} eq "-
``"){$strand=-1;}

	my $region1=$elements[1];my $region2=$elements[2];
	$region1=~s/,//g;$region2=~s/,//g;
	my @regions=($region1,$region2);
	my %reads=();
	foreach(@samples){
		my $sample=$_;
		my $bamfile="./$sample/$sample.bam";
		my $sam = Bio::DB::Sam->new(-bam  =>$bamfile,);
		foreach(@regions){
			my $region=$_;
			my %qnames=();
			my @spl1=split(/:/,$region);
			my $chr=$spl1[0];
			my @spl2=split(/-/,$spl1[1]);
			my $rstart=$spl2[0];my $rend=$spl2[1];
			my @alignments = $sam->get_features_by_location(-seq_id => $chr,
					-start  => $rstart,
					-end    => $rend);
	 foreach(@alignments){
		my $alignment = $_;
		my $start      = $alignment->start;
		my $end        = $alignment->end;
		my $astrand=$alignment->strand;
		next if ($strand != $astrand);
		if ($strand==1){
			next if (($start<$rstart) || ($start > $rend));
		}
		elsif ($strand==-1){
			next if (($end < $rstart) || ($end > $rend));
		  }
	    my $qname= $alignment->qname;
	    $qnames{$qname}=1;
	 }
			my $val=keys %qnames;
			$reads{$sample}{$region}=$val+1;#######NOTE +1 to be substracted later
		}
	}


	my $chisprob="und";my $chisq="und";
	my $sam1reg1=$reads{$sample1}{$region1}-1;
	my $sam2reg1=$reads{$sample2}{$region1}-1;
	my $sam1reg2=$reads{$sample1}{$region2}-1;
```

```
        my $sam2reg2=$reads{$sample2}{$region2}-1;
        my $fold="undef";
        $fold=($sam1reg2/$sam1reg1)/($sam2reg2/$sam2reg1) if ($sam1reg1 && $sam2reg2);
        if
``(((($sam1reg1+$sam2reg1)*($sam1reg2+$sam2reg2)*($sam2reg1+$sam2reg2)*($sam1reg1+$sam
``1reg2))>0){
                $chisq=((((($sam1reg1*$sam2reg2)-
``($sam2reg1*$sam1reg2))*(($sam1reg1*$sam2reg2)-
``($sam2reg1*$sam1reg2)))*($sam1reg1+$sam2reg1+$sam1reg2+$sam2reg2))/(($sam1reg1+$sam
``2reg1)*($sam1reg2+$sam2reg2)*($sam2reg1+$sam2reg2)*($sam1reg1+$sam1reg2));
                $chisprob=Statistics::Distributions::chisqrprob (1,$chisq);
        }
        print OUT
``join("\t",$gene,$sam1reg1,$sam1reg2,$sam2reg1,$sam2reg2,$fold,$chisq,$chisprob) . "\n";
}
print "output is $outfile\n";
```

makeannotations5pr.pl

```
#!/usr/bin/perl -w
use strict;

die "input table needed\n" if (not $ARGV[0]);
my $inputtable=$ARGV[0];chomp($inputtable);
die "ERROR\n" if (system ("ann_mk.pl $inputtable && ann5prime_mk.pl $inputtable &&
findupstream.pl $inputtable.5pr-ann $inputtable.ann $inputtable"));
```

normwig.pl

```
#!/usr/bin/perl -w
use strict;
my $cycles=21;
die "input wiggle file and sample name needed (e.g.\'normwig.pl coverage.wig Sample1\')\n" if (not
``$ARGV[1]);
my $sample=$ARGV[1];chomp($sample);
my $folder=$ARGV[2];
if (not $folder){
        $folder=".";
}
else{
        chomp($folder);
}
my $wigin=$folder . "/" . $ARGV[0];
my $wigout=$folder ."/$sample"."_norm.wig";
open (WIGIN,$wigin) or die "the wig file $wigin could not be located\n";
open (WIGOUT,">$wigout") or die "the wig file $wigout could not be opened\n";
my $firstline=<WIGIN>;
print WIGOUT "track type=bedGraph name=\"$sample\"\n";

my $totalcounts=0;
while (<WIGIN>){
        my $line=$_;chomp($line);
        my @elements=split(/\t/,$line);
        next if ($elements[2]==0);
        $totalcounts+=($elements[2]-$elements[1])*$elements[3];
}
close(WIGIN);open (WIGIN,$wigin);
$firstline=<WIGIN>;
while(<WIGIN>){
        my $line=$_;chomp($line);
        my @elements=split(/\t/,$line);
        next if ($elements[2]==0);
        my $normvalue=($elements[3]*$cycles/$totalcounts) *1000000;
        print WIGOUT join("\t",$elements[0],$elements[1],$elements[2],$normvalue."\n");
}
close(WIGIN);close(WIGOUT);
system "gzip $wigout\n";
print "Wiggle/Bedgraph normalized for sample $sample\n";
```

retrieveUCSCimgs.pl

```
#!/usr/bin/perl -w
use strict;

die "usage is retrieveUCSCimgs.pl ANNOTATION-PREFIX URL GENE-LIST\n" if (not $ARGV[2]);

my $annot_prefix=shift(@ARGV);
my $annot_dir=$ENV{RNASEQ5PR} . "/";
my $ann_expr=$annot_dir . $annot_prefix . ".ann";

my $settingslink=shift(@ARGV);
my $genelist=shift(@ARGV);
chomp($genelist);

open(CONS,$ann_expr) or die "$ann_expr could not be opened\n";
open(LIST,$genelist) or die "$genelist could not be opened\n";

my %genes=();
while(<CONS>){
	my $line=$_;chomp($line);
	my @elements=split(/\t/,$line);
	my $gene=$elements[0];
	my $chr=$elements[1];
	my $strand=$elements[2];
	my @txstarts=split(/,/,$elements[3]);
	my $txstart=$txstarts[0];
	my @txends=split(/,/,$elements[4]);
	my $txend=$txends[-1];
	$txstart-=1000 if ($strand eq "+");
	$txend+=1000 if ($strand eq "-");
	my $coord=$chr . ":" . $txstart . "-" . $txend;
	$genes{$gene}=$coord;
}
my $totalnum=0;
while(<LIST>){
	$totalnum++;
}
my $lentotal=length($totalnum);
close LIST;
open(LIST,$genelist) or die "$genelist could not be opened\n";

my $count=0;
while (<LIST>){
	$count++;my $ct2=$count;
	my $lenct2=length($ct2);
```

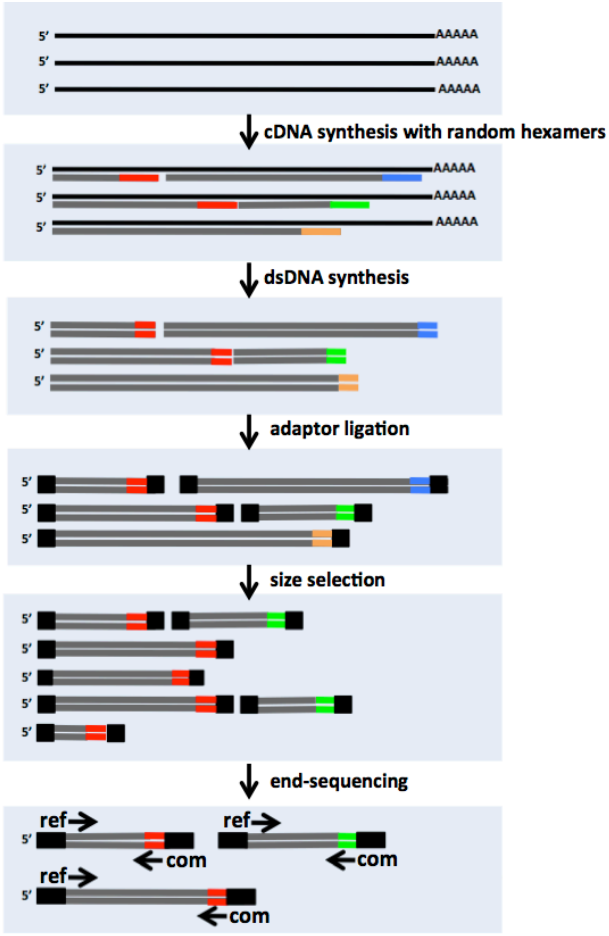
```
        if($lenct2<$lentotal){
                until($lenct2==$lentotal){
                        $ct2="0$ct2";
                        $lenct2=length($ct2);
                }
        }

        my $line=$_;chomp($line);
        my @elements=split(/\t/,$line);
        my $gene=$elements[0];
        my $coord=$genes{$gene};

        my $url="http://genome.ucsc.edu/cgi-
``bin/hgTracks?hgS_doLoadUrl=submit&hgS_loadUrlName=$settingslink&position=$coord";
        print $url . "\n";

        system "wget -O output-file.html \"$url\"\n";
        open (OUTPUTHTML,"output-file.html") or die "file output-file.html could not be found\n";
        my $png_main="";my $png_side="";my $png_bluelines="";
        while (<OUTPUTHTML>) {
                ($png_main) = m/(hgt_genome\S+\.png)/ if (m/(hgt_genome\S+\.png)/);
                ($png_bluelines)=m/(blueLines\S+\.png)/ if (m/(blueLines\S+\.png)/);
                ($png_side)=m/(side_genome\S+\.png)/ if (m/(side_genome\S+\.png)/);
                last if (($png_main) && ($png_bluelines) && ($png_side));
        }

        my $pngout=$ct2 . "_" . $gene . ".png";
        sleep 1;
        system("wget -O png_main.png  http://genome.cse.ucsc.edu/trash/hgt/$png_main");
        sleep 1;
        system("wget -O png_side.png  http://genome.cse.ucsc.edu/trash/hgtSide/$png_side");
        sleep 1;
        system("wget -O png_bluelines.png  http://genome.cse.ucsc.edu/trash/hgt/$png_bluelines");
        #system("convert png_main.png png_side.png png_bluelines.png -gravity center -composite -
``format png out1.png");
        system("convert png_bluelines.png png_main.png png_side.png  -gravity center -composite -
``format png out1.png");
        system("convert out1.png png_side.png -gravity center -composite -format png out2.png");
        system("convert out2.png -background white -flatten $pngout");
        close OUTPUTHTML;

        system "rm output-file.html png_main.png png_side.png png_bluelines.png out1.png
``out2.png";
        print "saved as $pngout\n";
        sleep 3;
}
```

Supplemental Table Legends

Supplemental Table 1. Quantified extent of 5' changes for identified genes and other information. This gene-list is sorted by the extent of 5' change quantified in column H. (A) Gene, (B) Description, (C) Annotated transcriptional starts, (D) Reference transcript reads corresponding to the baseline start 'start1' for the indicated sample, (E) Reference transcript reads corresponding to the regulated start 'start2' for the same sample, (F-G) Reference transcript reads corresponding to the baseline and regulated starts for the second sample, (H) The extent of start-site change quantified by $\frac{MHC403start2/MHC403start1}{WTstart2/WTstart1}$. (I) P-value assessment of the start-site change using Chi-square, (J) 5' score between MHC403/+ and WT samples (from Supplemental table 1), (K) 5' score comparison for the same genes between two wild-type samples, (L) Position of the gene as prioritized by the 5'RNA-seq scoring method, (M) Extent of start-site change quantified from 5'RACE (Supplemental Figure 6) using ImageJ (http://rsbweb.nih.gov/ij/). (N-Y)Expression levels and comparison between MHC403/+ (with HCM) and Wild-type for whole tissue Left-Ventricle (N-Q; same experiment as in Supplemental table 1), isolated myocytes (R-U), and isolated fibroblasts (V-Y). Expression values are read counts normalized to 1 million total sample reads.

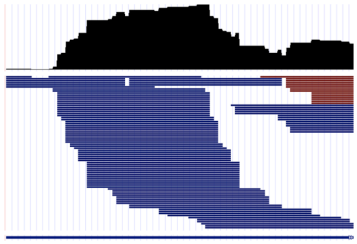
Supplemental Table 2. Gene ontology analysis of genes with start site changes in LV tissue from HCM mice. Abbreviations are MF, molecular function; CC, cellular compartment; BP, biological process; Count, number of genes with start site changes; FDR, false discovery rate.

Supplemental Table 3. Expression comparison between pooled MHC403/+ (n=5) and MHC403/+ Fhl1(-) (n=4) samples for fibrotic and hypertrophic genes, indicating increased disease pathology in animals lacking Fhl1. For baseline, expression information from a pooled wild-type (n=5) and a pooled Fhl1 null (n=3) samples are also included. This table includes the top 8 highest expressing collagen genes (genes with the highest normalized-reads values). (A) Gene, (B) Description, (C) Chromosome, (D) Genome start coordinate, (E) Genome end coordinate, (F) Strand, (G-J) Read counts normalized to 1 million total sample reads for the indicated samples, (K-N) Gene read counts for the indicated samples, (O) Fold between the two normalized read counts of the two samples as indicated, (P) Bayesian p-value comparison between the two read values.



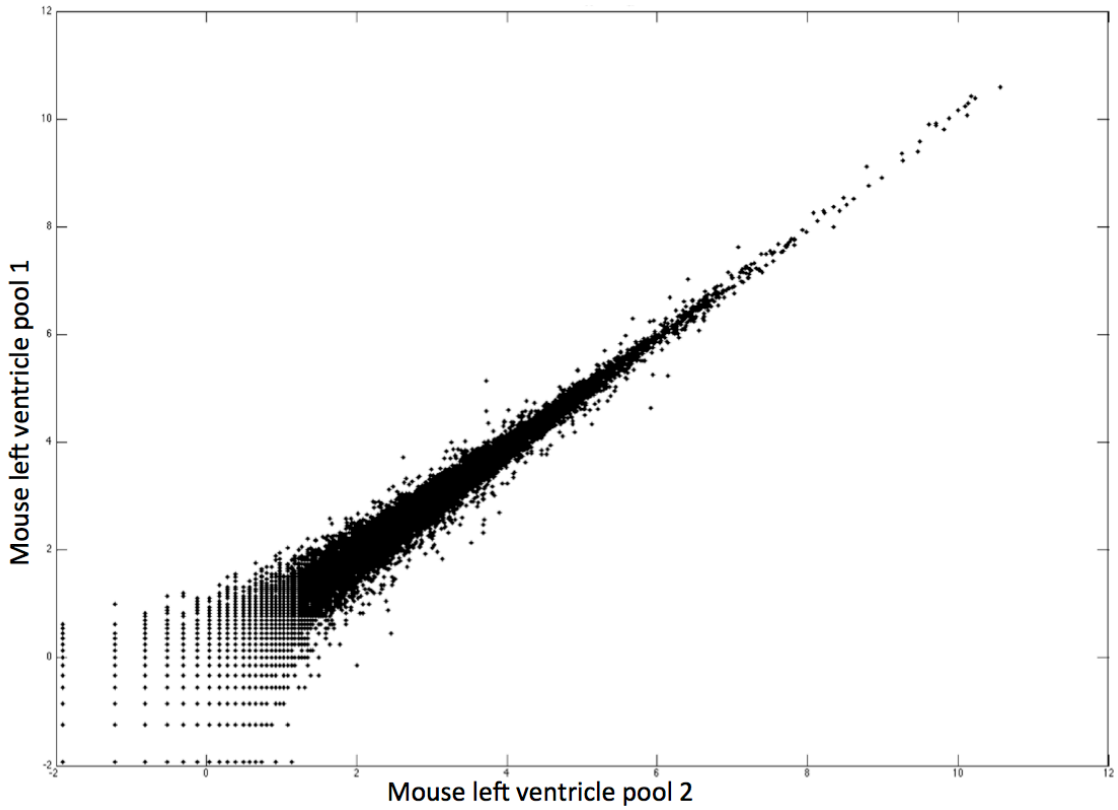
Supplemental Figure 1 cDNA library construction steps to enhance 5’ sequencing. Random priming near 5' ends of transcripts (red) results in the synthesis of smaller fragments that are selected for sequencing.

a.

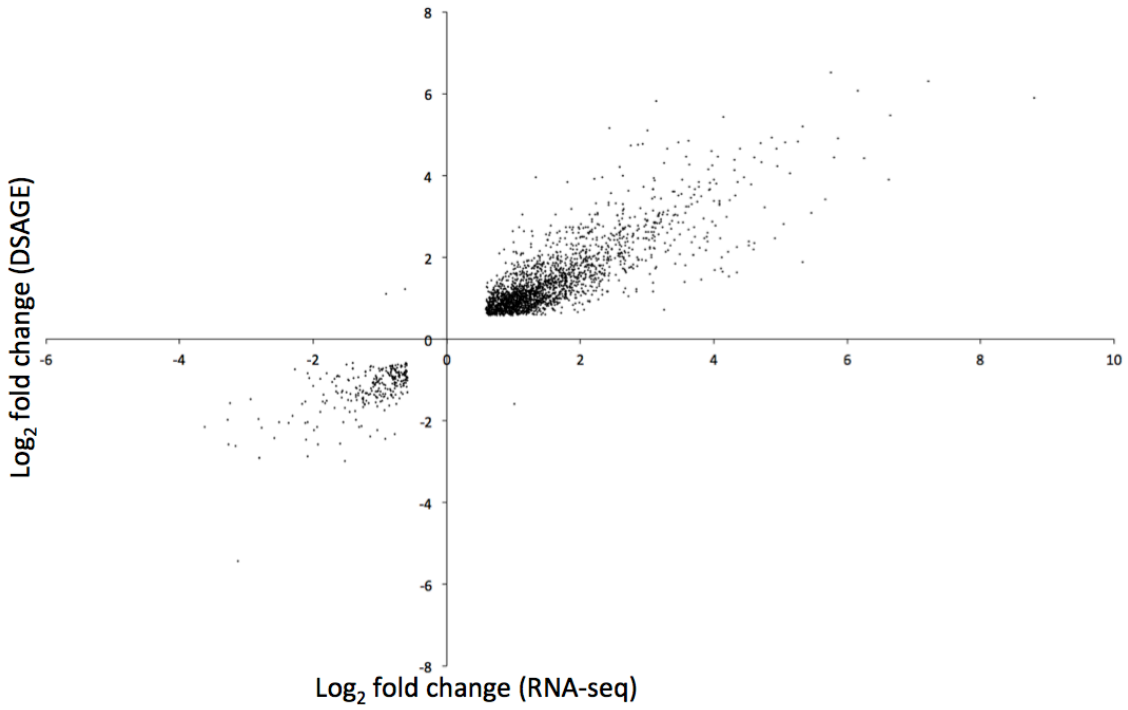


b.

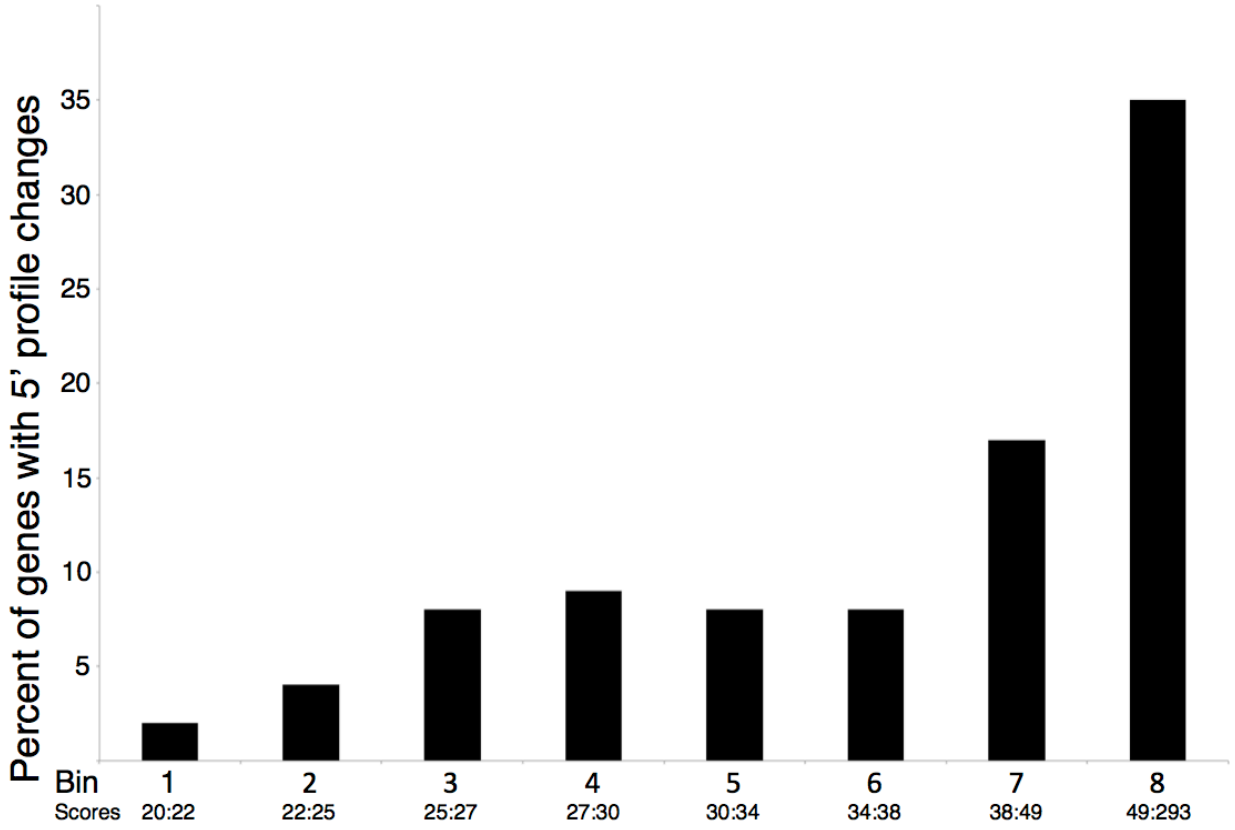
Supplemental Figure 2 Read depth and reads of first exons of *Fhl1* of (a) *bFhl1* and (b) *iFhl1* isoforms. Reads from the transcript (+, blue) strand precede reads on complement (-, red) strand, and illustrate directional information at 5' regions.



Supplemental Figure 3 Technical replication of RNAseq libraries. Log values (base 10) of normalized expression values from two biological replicate samples of wild-type heart tissue are shown (R=0.994).



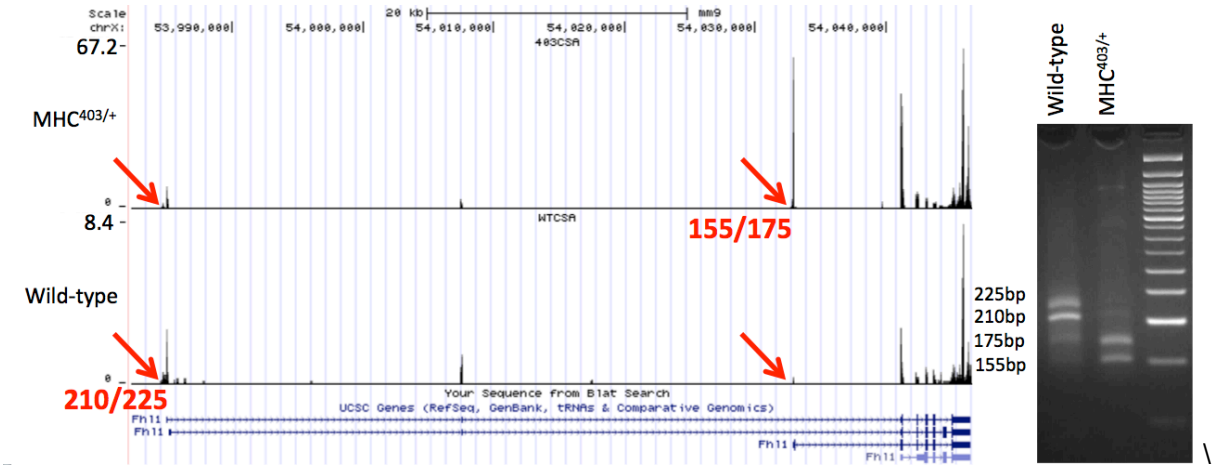
Supplemental Figure 4 Comparison of RNA-seq with deep sequencing analysis of gene expression (DSAGE [4]). The fold change (base 2) of gene expression that shows agreement (first and third quadrants; first quadrant is upper right, second is upper left and increasing counter-clockwise) or disagreement (second and fourth quadrants) between the two methods is plotted for all genes (a minimum of 10 reads was required). Agreement was defined when both methods concurred that expression was increased or decreased by 1.5 fold, p-value<0.001. Disagreement was defined when gene expression was discordant by 1.5 fold, p-value<0.001. There is strong correlation between these methods R=0.9.



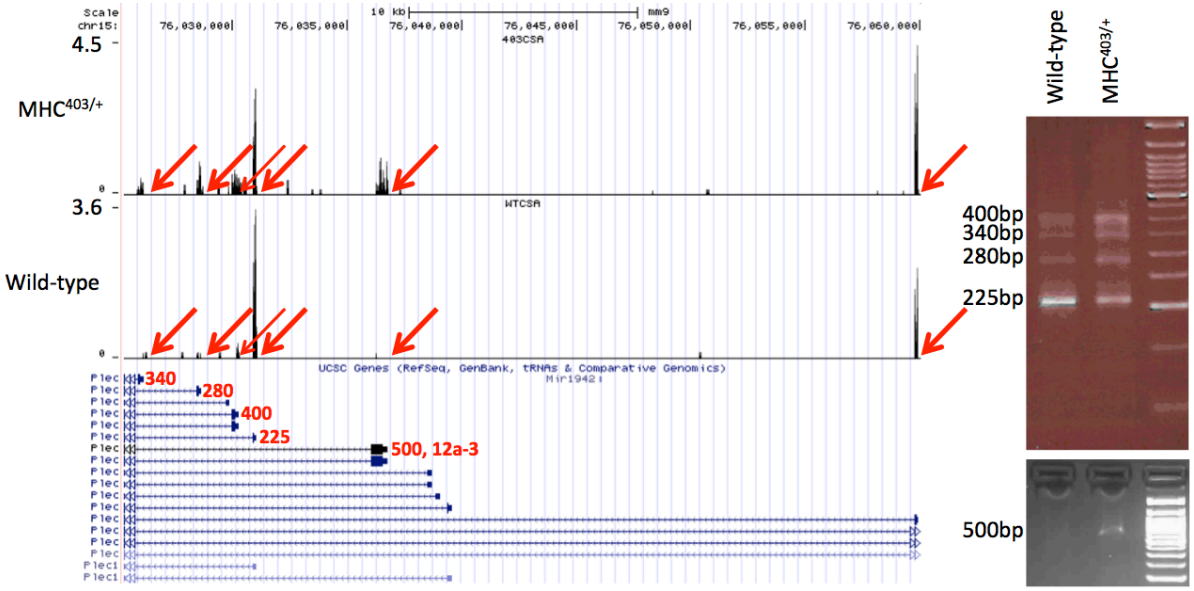
Supplemental Figure 5 Percentage of genes with 5' profile changes follows increasing scores for 800 evaluated genes. Genes with 5'RNA-seq scores ≥ 20 (Supplemental table 1) were selected. The gene profiles (Supplemental Image data) were evaluated for 5' changes. Then, genes were binned based on the score, from lowest to highest scoring (100 genes/bin). The number of genes in each bin with 5'profile changes is shown (y-axis). The bin number, and the score range corresponding to the binned genes are shown (x-axis).

a.

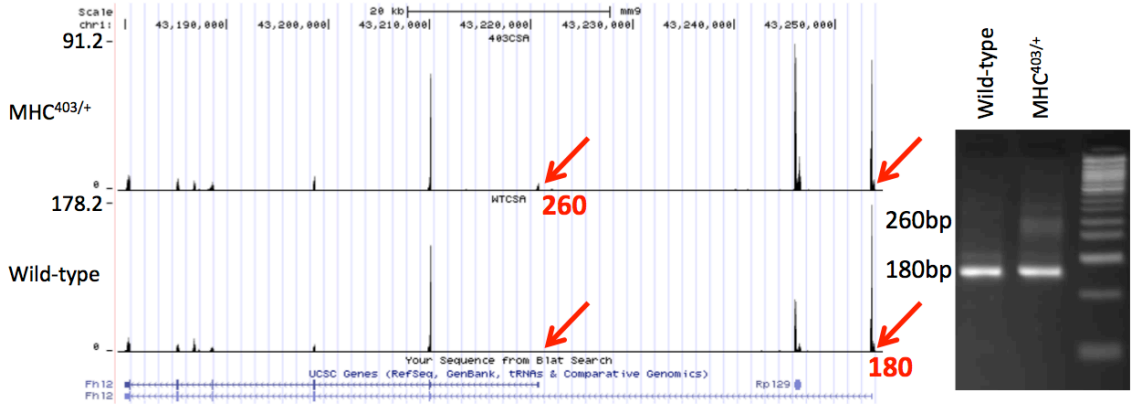
b. 1. Fhl1



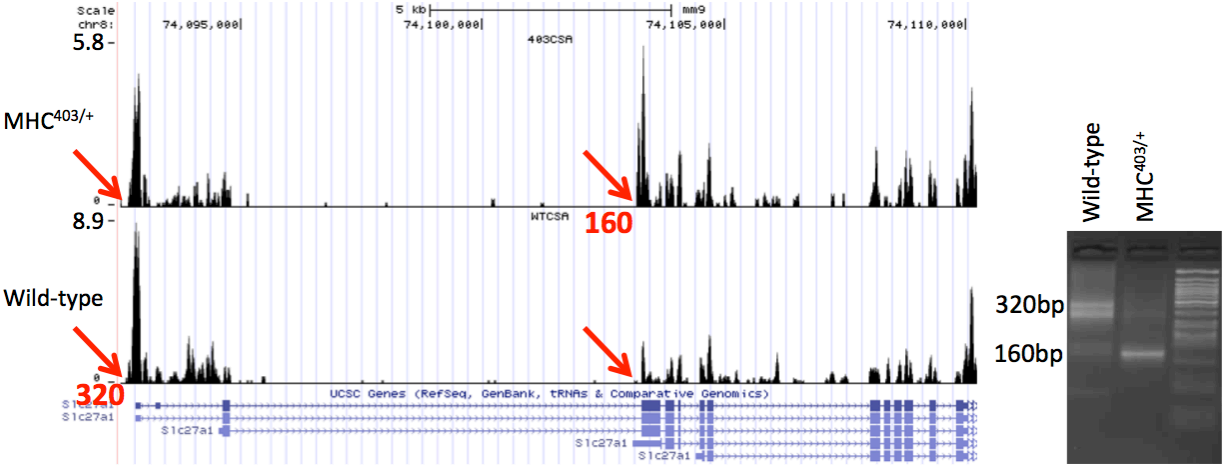
2. Plec1

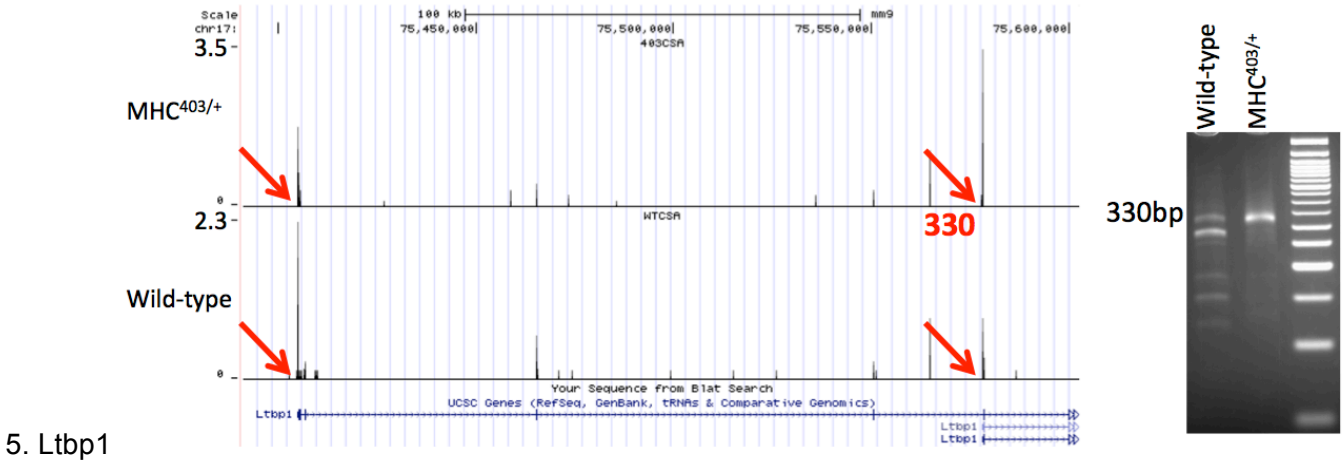


3. Fhl2

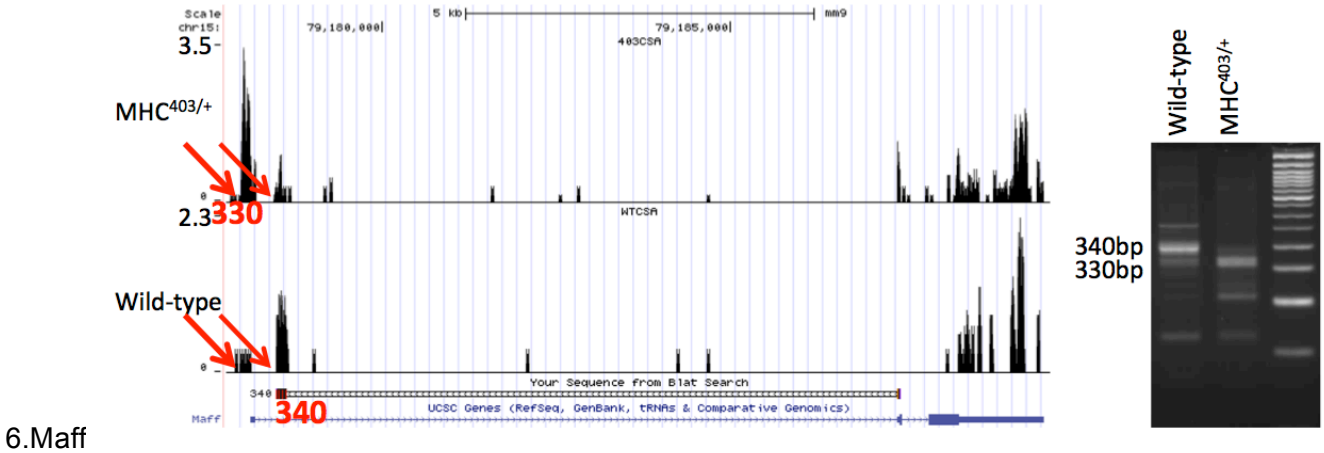


4. Slc27a1





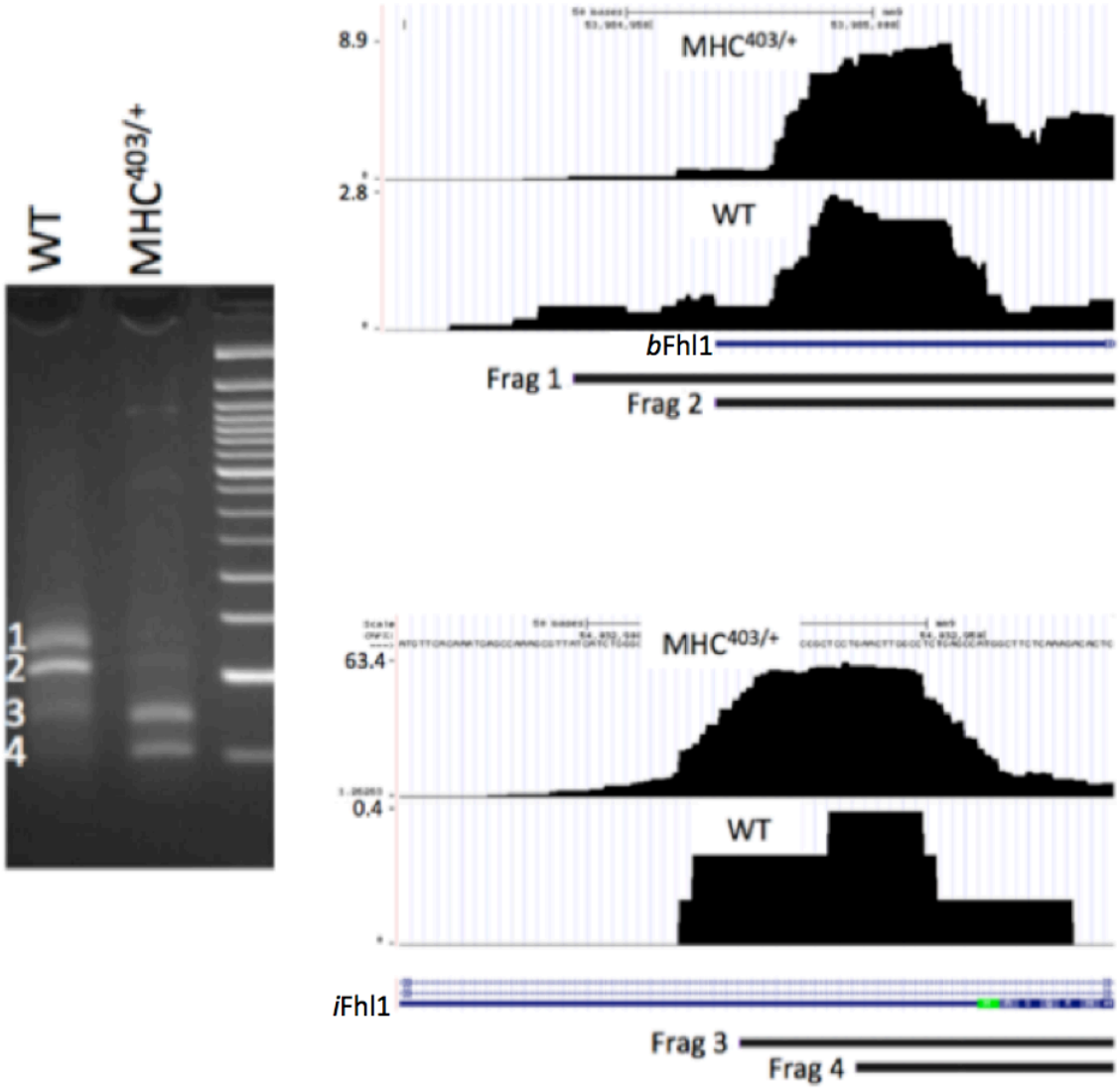
5. Ltbp1



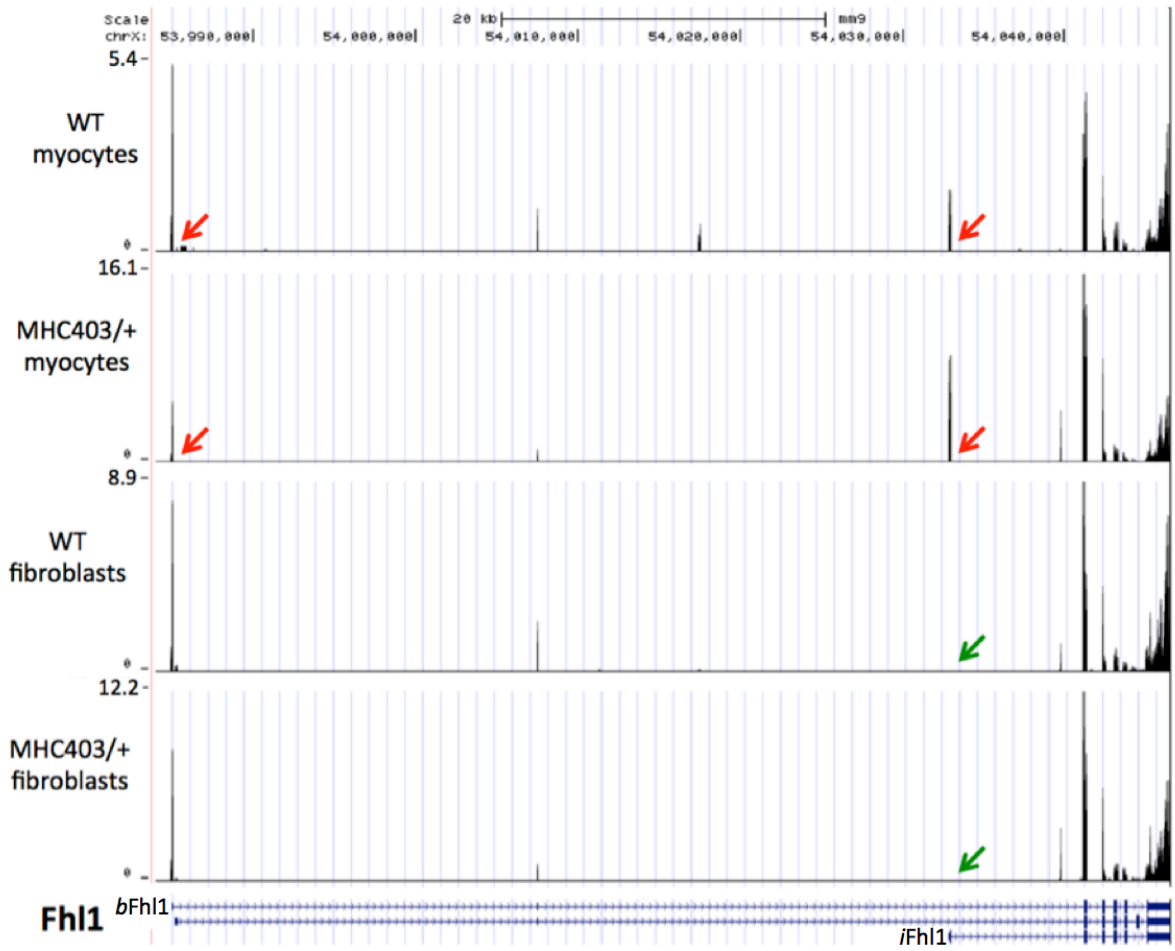
6.Maff

Supplemental Figure 6 a. Primer design scheme for semi-quantitative 5'RACE. In the same reaction, the relative concentration of queried isoforms can be assessed. Red arrows indicate primers that correspond to the ligated adapter at the 5' end (supplied by company). Black arrows indicate primers designed to correspond to shared sequences in different isoforms.

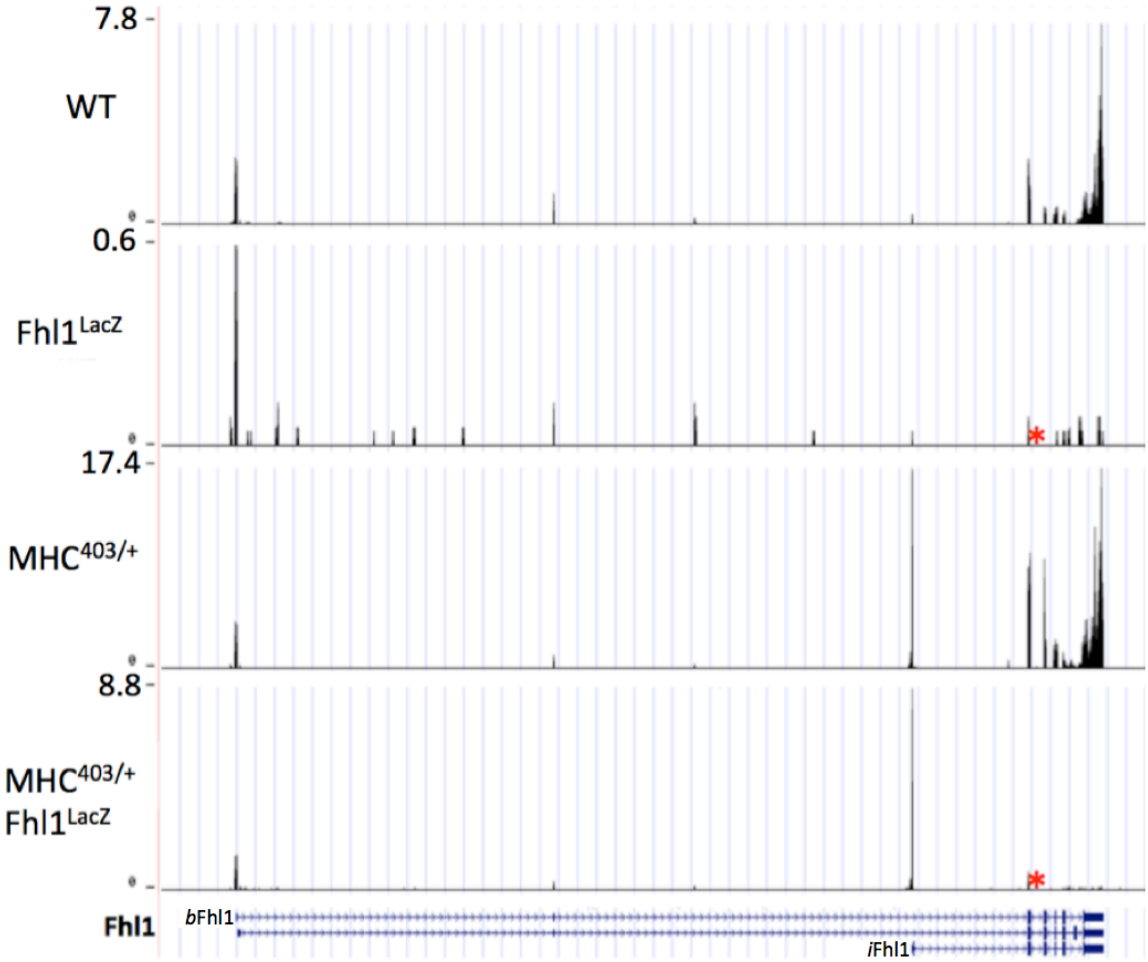
b. 5'RACE for 6 genes showing concordant results with 5'RNA-seq. Sizes displayed are approximations and they are used to identify the fragment. All fragments indicated were sequenced by di-deoxy sequencing and mapped to the shown locations with BLAT from UCSC browser.
b1-5: genes scored by the start-site algorithm. b2: Plec has multiple unique 5' ends and about 6 are detected by 5'RNA-seq. 5'RACE with a common primer identifies 4 of these isoforms in proportions consistent with levels from 5'RNA-seq (upper gel). When a proximal primer is used specific to an isoform not initially detected by 5'RACE (12a-3), it then also detected in consistent levels with 5'RNA-seq (lower gel). This shows that while 5'RACE can verify products, it can lead to false negatives and that 5'RNA-seq can be used in a more unbiased manner. b5: the isoform predominant in wild-type could not be amplified with the common primers (the resulting fragment would be too long to amplify). The same reaction conditions (with the same amount of template) was used in these reaction (as with the other reaction b1-4 and b6) indicating that the fragment shown is in much higher proportion in the HCM sample, consistent with 5'RNA-seq data. Thus, 5'RNA-seq can provide semi-quantitative information in instances when amplification when using 5'RACE is not possible. b6: a gene with an unannotated start (its corresponding fragment from 5'RACE is shown using BLAT). 5'RNA-seq can discern this change (even though the algorithm did not score this instance).



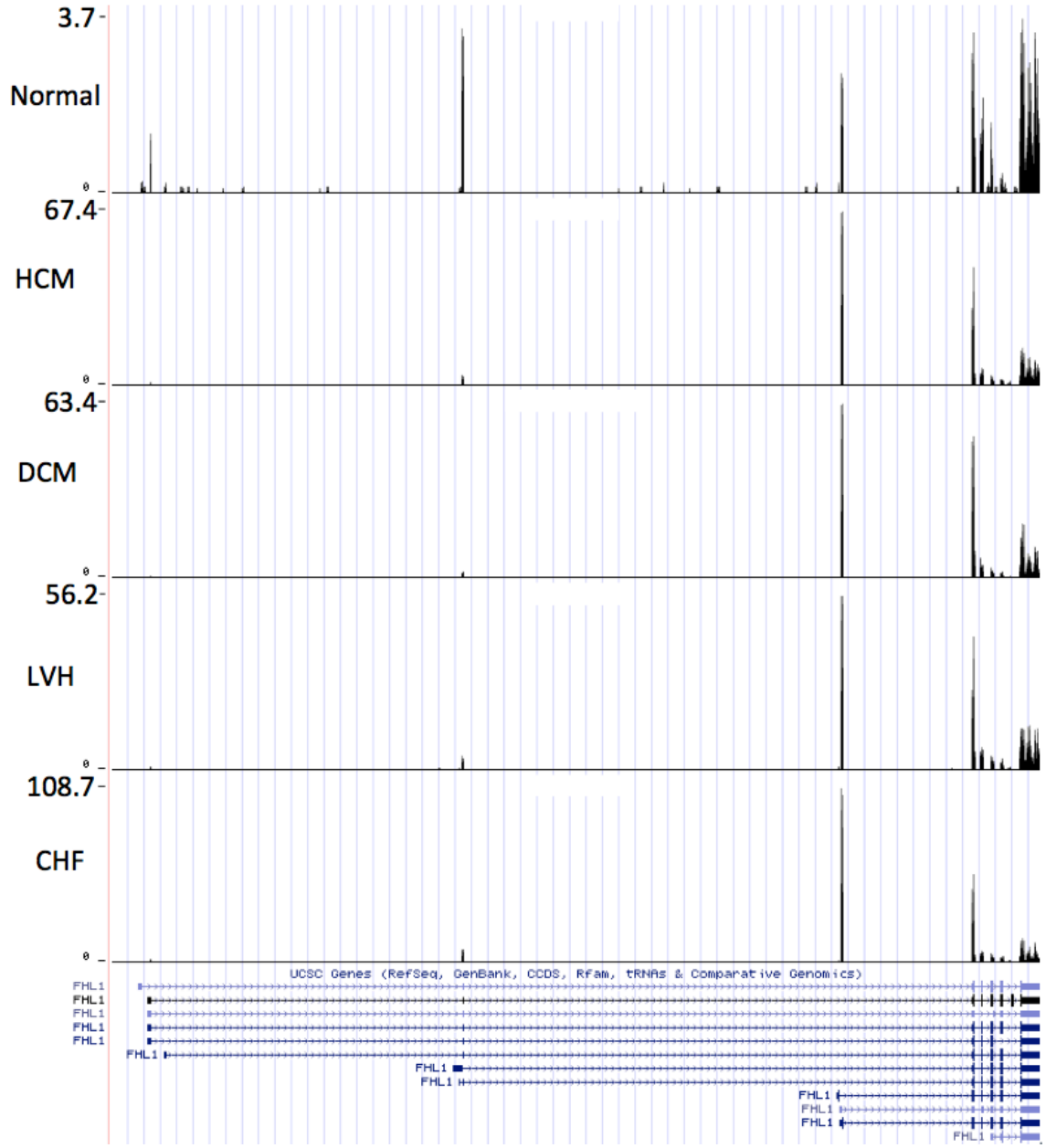
Supplemental Figure 7 5'RACE validates Fhl1 5' end changes using primers that correspond to a common downstream exon of the isoforms (Methods). Fragments (numbered) were sequenced and aligned to the genome with Blat (UCSC browser) and shown with RNA-seq data. Alignment of these products corresponds to RNA-seq reads. The same total RNA was used for 5'RACE and RNA-seq.



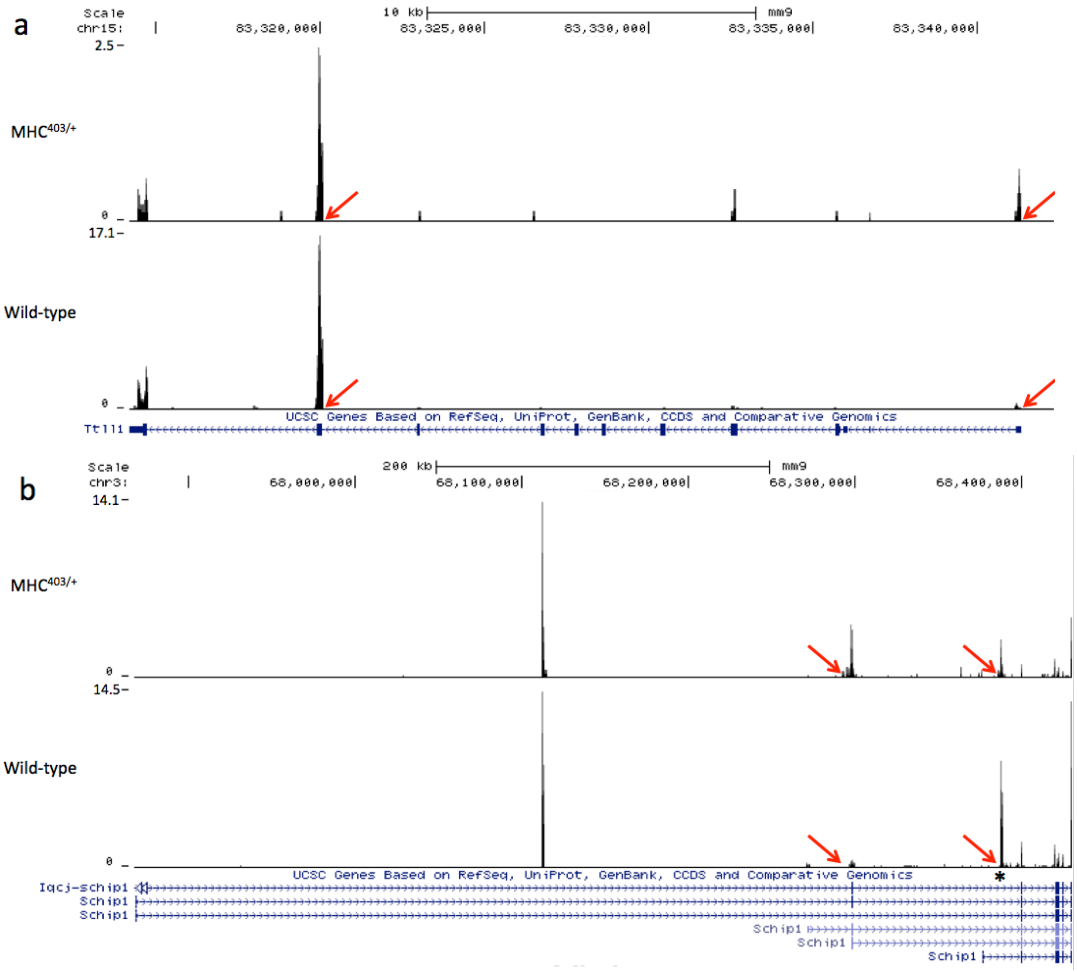
Supplemental Figure 8 Transcriptional profiles of isolated myocytes and non-myocytes/fibroblasts were assessed with RNA-seq. Transcriptional switch of Fhl1 occurs in myocytes (red arrows) while isoform 2 is not expressed in fibroblasts (green arrows).



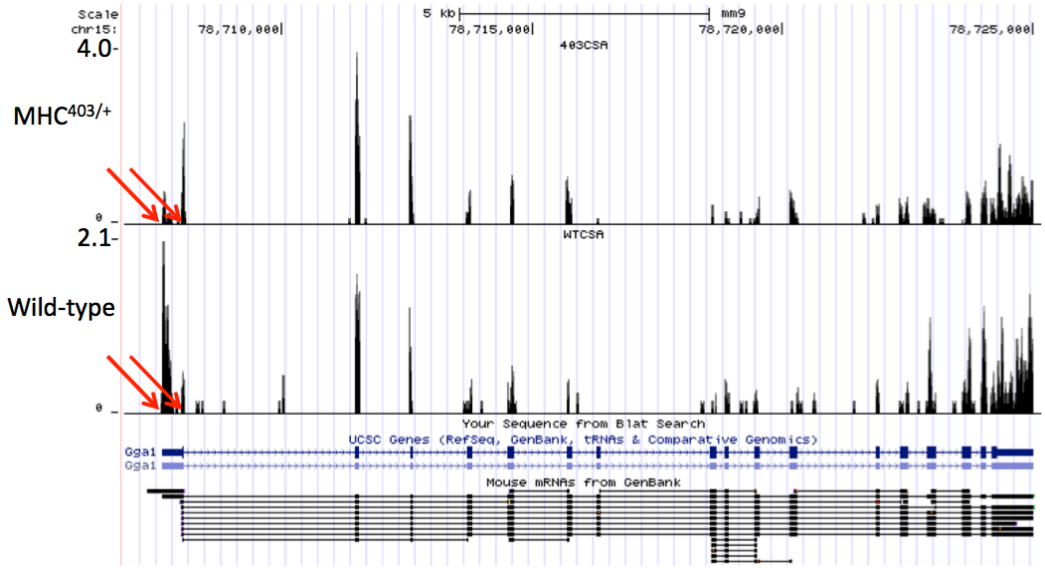
Supplemental Figure 9 Fhl1 expression in left ventricle tissues derived from male wild-type, Fhl1^{LacZ} (Fhl1-null), MHC$^{403/+}$, MHC$^{403/+}$/ Fhl1 ^{LacZ}. The location of the inserted LacZ sequences is indicated (*). Note that this insertion excises Fhl1 protein-encoding sequences but does not alter 5' regulatory elements that drive LacZ expression. The LacZ sequences were detected by manual sequence assembly (not shown) and encode the protein in-frame.



Supplemental Figure 10 Read depth for representative samples for normal, HCM, DCM, LVH and CHF (quantified changes at 5' ends are shown in Figure 3B).


a
Scale
chr15:
10 kb
mm9
83,320,000
83,325,000
83,330,000
83,335,000
83,340,000
2.5
MHC403/+
0
17.1
Wild-type
0
UCSC Genes Based on RefSeq, UniProt, GenBank, CCDS and Comparative Genomics
Ttll1
b
Scale
chr3:
200 kb
mm9
68,000,000
68,100,000
68,200,000
68,300,000
68,400,000
14.1
MHC403/+
0
14.5
Wild-type
0
UCSC Genes Based on RefSeq, UniProt, GenBank, CCDS and Comparative Genomics
Iqcj-Schip1
Schip1
Schip1
Schip1
Schip1
Schip1

c



Supplemental Figure 11 Three read-depth gene profiles of Ttll1(a), Schip1(b), Gga1(c) which are high scoring by 5' analysis. We verified the unannotated start-site of Schip1 (*) with 5'RACE. Arrows indicate predicted start-site changes from 5'RNA-seq (the direction of the arrow corresponds to the strand of the gene).

Supplemental References

1. Wang, Z., Gerstein, M., and Snyder, M. 2009. RNA-Seq: a revolutionary tool for transcriptomics. *Nat Rev Genet* 10:57-63.

2. Christodoulou, D.C., Gorham, J.M., Herman, D.S., and Seidman, J.G. 2011. Construction of normalized RNA-seq libraries for next-generation sequencing using the crab duplex-specific nuclease. *Curr Protoc Mol Biol* Chapter 4:Unit4 12.

3. Audic, S., and Claverie, J.M. 1997. The significance of digital gene expression profiles. *Genome Res* 7:986-995.

4. Christodoulou, D.C., Gorham, J.M., Kawana, M., DePalma, S.R., Herman, D.S., and Wakimoto, H. 2011. Quantification of gene transcripts with deep sequencing analysis of gene expression (DSAGE) using 1 to 2 microg total RNA. *Curr Protoc Mol Biol* Chapter 25:Unit25B 29.