wormase (github.com/paabylab/wormase) code details

Avery Davis Bell
Original: August 2024 | Updated: April 2025

This document describes inputs and outputs (or other relevant aspects) of all scripts in this repository that are run on their own – *i.e.*, if a script is only called via an included workflow, that script may not be fully documented here.

Table of Contents

Generate diploid transcriptomes: data_generation_scripts/ getdiploidtranscriptomes/getdiploidtranscriptomes.nf

- Parameters

Category	Flag for script (in script as params.<this)	Default value	Description
General input	--strainlist	""	Path to one-column file with strains to process (one per line). These must be column headers in VCFs and will be used for output labelling as well.
General output	--outputdir	""	Parent output directory. Will be created if doesn't exist.
General input	--snpvcf		VCF containing all SNPs (only) for all strains of interest vs. reference genome of interest
General input	--indelvcf		VCF containing all INDELs (only) for all strains of interest vs. reference genome of interest
General input	--reffasta	c_elegans.PRJNA13758.WS276.genomic.fa	FASTA reference genome - for strain-specific genome creation; what variants were called against. Should be **unzipped**
General input	--refgtf		GTF file for reference genome. **Transcripts must NOT have underscores in their name if salmon is to be run through EMASE. Modify transcript names to exclude them if necessary.**
General input	--refname	N2ws276	name of reference strain/genome for output folders, files
General input	--idx	salmon	Which index to build? Possible values: 'bowtie2', 'salmon', 'all' (builds both), 'none' (builds none)
General input	--salmdecoy	no	'yes' or 'no' - add REFERENCE *genome* sequence as decoy sequence for salmon index building? Default 'yes'. *(true/false here not smooth - I think*

			gets interpreted as logical, which I didn't want to figure out how to deal with)
organizational	--g2gtoolsconda		path to python 2 conda environment where g2gtools properly set up *Note: using ~ here doesn't work when running on node*
organizational	--procgtfscriptdir		Directory containing Python GTF worker scripts gtfnonchroverlaps.py and gtftoemasegenemapping.py
organizational	--salmonenv		path to conda environment where salmon is installed

- Processes & outputs

Name	**Description**	**Any saved outputs in [subdirectories of outdir]**
refgenes2transcripts	get EMASE-style genes to transcripts mapping from reference genome [replaces earlier prepemaseref process]	params.outputdir + "/refgenomeinfo" ${refname}_emaseformat_genes2transcripts.tsv
reftranscriptomeandinfo	get reference transcriptome from genome; add ref strain name to transcript names; get reference transcript lengths Intermediate - for combining with strain-specific transcriptome/lengths to make diploid Chaining several things together here - they're just litle & short *Uses a custom python script; bioawk; gffread; seqkit*	N/A [all intermediates]
g2gvcf2chain	Chain indels onto reference	N/A [only want final genomes/transcriptomes used for bowtie & emase]
g2gpatch	Patch SNPs onto reference genome	N/A [only want final genomes/transcriptomes used for bowtie & emase]
g2gtransform	chain indels onto patched genome	N/A [only want final genomes/transcriptomes used for bowtie & emase]. *Would keep this if wanted the fasta long term*
g2gconvert	update GTF file based on new genome	N/A [only want final genomes/transcriptomes used for bowtie & emase]

	After this process, strain-specific genome generation is complete.	
getexclseqs	For alt strain, splits GTF into sequences that should be excluded vs. included from making transcriptome (those that end after chromosome ends are excluded) Runs gtfnonchroverlaps.py	N/A [all intermediates]
straintranscriptome	Generate strain-specific transcriptome, to later be combined into diploid transcriptome A couple processes together: gffread to extract, then some formatting (bioawk, seqkit)	N/A [all intermediates - strain-specific rather than diploid]
straintrnslengths	Get lengths of each transcript for one strain, adding in any that were excluded	N/A [all intermediates - strain-specific rather than diploid]
diptranscriptomelns	Combines reference & strain-specific transcriptome fastas and transcript length files to get pseudo-diploid transcriptome outputs	Ref-strain specific dir: $params.outputdir/$params.refname_$mystrain <ref>_<strain>_transcriptome.fa - pseudodiploid transcriptome (*equivalent to earlier* emase.pooled.transcripts.fa*)* <ref>_<strain>_transcriptlengths.txt - transcript lengths (including strain in transcript name) (*equivalent to earlier* emase.pooled.transcripts.info*)*
bowtie2idx	Builds bowtie2 single end index for diploid transcriptome. Bowtie2/2.3.5.1 is PACE version. **Only run if --idx is 'bowtie2' or 'all'**	Ref-strain specific dir: $params.outputdir/$params.refname_$mystrain *.bt* files - 6 total
salmondecoyprep	Creates fasta of diploid transcriptome + reference genome (as decoy), + chromosome names for use as salmon index's decoy **Only run if --idx is 'salmon' or 'all' AND --salmdecoy is 'yes'**	N/A
salmonidxdecoy	builds salmon index (using ref genome as decoy) **Only run if --idx is 'salmon' or 'all' AND --salmdecoy is 'yes'**	Ref-strain specific dir: $params.outputdir/$params.refname_$mystrain Directory *salmon_idx* in here has all salmon index info.

salmonidxnodecoy	builds salmon index without decoy **Only run if --idx is 'salmon' or 'all' AND --salmdecoy is 'no'**	Ref-strain specific dir: $params.outputdir/$params.refname_$mystrain Directory *salmon_idx* in here has all salmon index info.

Exon DNA coverage: data_generation_scripts/mosdepthmergedexons/mosdepthmergedexons.nf

- Parameters

Category	Flag for script (in script as params.<this>)	Default value (if highlighted, need to provide)	Description
General input	--gtf	""	path to GTF containing genes for which to determine coverage from all BAMs. Columns as ws276 GTF from Wormbase.
General input	--sampleinfo	""	Path to sample information file. Columns SampleID (name of sample for output), bam (path to BAM file to process for this sample), bai (path to BAM .bai index file for this sample)
General input	--outdir	"out"	Path to output directory
General input	--outname	"out"	Prefix for output files that contain all samples' mosdepth information
GTFtools	--refname	"ref"	Prefix for output file containing merged exons - i.e. reference genome name
GTFtools	--chrs	I,II,III,IV,V,X,MtDNA	Chromosomes to process exons/genes for - need to match the GTF. Default is for *C. elegans* ws276
GTFtools	--gtftoolsdir	/storage/coda1/p-apaaby3/0/shared/software/GTFtools_0.8.5	Directory containing GTF tools python script gtftools.py
mosdepth	--flag	1796	--flag (SAM flag bits to exclude) argument for mosdepth

			Default is mosdepth default; may very well want to change!
mosdepth	--mapq	0	-Q, mapq threshold argument for mosdepth, threshold below which read will be excluded **Default is mosdepth default; may very well want to change!**
Summary R script	--rscriptdir	~/rnaseqgitrepo/alignment	Directory containing exploregenecoverage_fromexons.R
Summary R script	--gff	""	Path to *genes only* gff3 file containing info on all genes from the GTF
Summary R script (subset)	--gsubset	''	OPTIONAL Path to no-header list of genes to run summary R script for - SUBSET of all genes. *It will also be run for all genes.*
Summary R script (subset)	--gsubsetname	""	OPTIONAL name of gene subset for output filenaming. Provide if provide --gsubset

- Processes

Name	**Description**	**Any saved outputs**
gtf2mergedexonbed	Get merged exons bed file from GTF using GTFtools	<refname>.mergedexons.bed.gz - bed file of merged exons made from GTF *[probably DON'T need to save this given same information is present in the coverage output beds, but keeping for now to be extra safe]*
mosdepth	Run mosdepth for genes. Also unzips bed output for downstream ease.	NA - combining together before saving
combinedpbeds	Combines *.regions.bed.gz mosdepth outputs into one file with all samples	<outname>.mergedexons.bed.gz - key file! One row per input gene; columns with gene info followed by one column per sample/strain containing mean coverage in that region (over that gene) for that strain

combinedpsums	Combines *.mosdepth.summary.txt mosdepth outputs into one file with all samples	<outname>.mosdepth.summary.txt. 2 rows per chromosome and total per sample. Columns: SampleID - which sample. Repeated for all rows with this sample's data Chrom - chromosome ID, or chromosome ID _region: data for entire chromosome or for all the provided regions (genes) on a given chromosome Length - length of chromosome/sum of length of regions Bases - read bases total aligned here at thresholds above Mean - mean coverage (# reads covering) across specified region Min - min coverage Max - max coverage
comboexonsexplore	Runs exploregenecoverage_fromexons.R for all genes *Added second*	<outname>_genecoveragefromexons_raw.txt.gz and <outname>_genecoveragefromexons_mednorm.txt.gz - per GENE coverages computed from merged exon bed, raw and normalized to across-gene median In /plots subdirectory, lots of plots of coverage - see description of outputs for full breakdown

EMASE: data_generation_scripts/emase/emasemultsamples.nf

- Parameters

Category	Flag for script (in script as params.<this)	Default value (if highlighted, need to provide)	Description
General input	--sampleinfo	""	Path to tab-delimited file containing sample information. Column names (descriptions): SampleID (sample ID as in input filenames, to be used in output filenames); RefDescrip (description of reference genome(s) to use in output file names); Bwt2BasePath (base path - before .#.bt2 - for bowtie2 index files - can just put xxx if not running bowtie2); SalmonIndexDir (path to directory generated by salmon index- can just put xxx if not running salmon); **PooledTranscriptLengths** (path to file containing emase.pooled.transcripts.info-style transcript length information, e.g. <ref>_<strain>__transcriptlengths.txt for the strain combinations in this sample); fldMean (mean library fragment length per sample, passed to --fldMean in salmon quant); fldSD (standard deviation library fragment length per sample, passed to --fldSD in salmon quant)
General output	--outputdir	""	Parent output directory. Will be created if doesn't exist.

General input	--refgenemappings	""	Path to prepare-emase reference output file transcript to gene mapping (emase.gene2transcripts.tsv). Tab delimited file where the first field is the gene ID, all other fields are the transcript IDs that belong to that gene
General input	--fastqdir	""	Directory containing all fastq.gz files to process. One or more per sample.
General	--rnamap	salmon	How to quantify RNA-seq for EMASE, bowtie2 mapping or salmon equivalence classes? Possible values: 'bowtie2', 'salmon', 'all' (runs both)
General	--libtype	SE	SE or PE - is libary single end or paired end? So processing can be done appropriately. (added 6/20/23)
General input	--plotinfo	""	Path to tab-delimited file containing plotting-related sample information, passed to eqclassalnmtsummary_multsamples.R. Must include columns SampleID (sample ID as everywhere else), those passed to --groupby, --splitby, --facetby (or subsets of those used)
Trimmomatic	--trimmodir		Path to trimmomatic v0.39 directory containing jar file and adapters directory (which itself contains the fasta files provided in next option, e.g. TruSeq3-SE.fa).
Trimmomatic	--trimmofa	"TruSeq3-SE.fa"	name of fasta file within trimmodir adapters/ directory matching those used in library preparation (added 6/20/23 so that can use this for PE sequencing)

Trimmomatic	--trimmoseedmism	1	Input to trimmomatic ILLUMINACLIP. How many of 16 bp can mismatch and still be counted as match.
Trimmomatic	--trimmoadapclipthresh	12	Input to trimmomatic ILLUMINACLIP. How accurate match between adapter sequence and read must be. Each correct base adds 0.6. They recommend 7-15 (12 bases needed for 7, 25 for 15).
salmon	--slibtype	SR	salmon --libtype option matching the library being aligned here
emase	--emasemodel	""	Normalization model for emase-zero (1-4)
emase	--emaset	0.0001	-t tolerance parameter for emase-zero
emase	-emasei	999	-i max # iterations parameter for emase-zero
eqclassalnmtsummary_multsamples.R	--groupby	Strain	--groupby input of eqclassalnmtsummary_multsamples.R. Column name (in --plotinfo file) to group samples by in output plots - samples will be split by this on the same plot
eqclassalnmtsummary_multsamples.R	--splitby	""	--splitby input of eqclassalnmtsummary_multsamples.R. Column name (in --plotinfo file) to split samples by, keeping all plots separate between these groups.
eqclassalnmtsummary_multsamples.R	--facetby	""	--facetby input of eqclassalnmtsummary_multsamples.R. Column name (in --plotinfo file) to facet samples by, keeping them on separate plots but in the same PDF in these groups.

eqclassalnmtsummary_multsamples.R	--genelists	""	--genelists input of eqclassalnmtsummary_multsamples.R. (# samples with unique alignments : # unique alignments per sample; can comma-separate to get more than 1 gene list per sample group. Sample grouping done based on by groupby, splitby, facetby inputs to this script)
organizational	--alntoolsenv		Path to alntools conda environment is installed in Anaconda3/2020.02 module
organizational	--salmonenv		path to conda environment where salmon is installed (Anaconda3)
organizationsl	--asescriptsdir		path to directory containing salmonalleleeqclasses.py and eqclassalnmtsummary_multsamples.R scripts

- Processes & outputs

Name	**Description**	**Any saved outputs in [subdirectories of outdir]**
mergeLaneFastqs	Merge files across lanes so that there's one fastq per sample	No
trimmoIlluminaAdapters 6/20/23 *and trimmoIlluminAdaptersPE*	Use trimmomatic to trim Illumina adapters from merged fastqs	Keeping trim logs. Don't need, but if hadn't done before might be interesting. In /triminfo 6/20/23 added - also keeping, if PE, unpaired reads (so reads whose mate was dropped thanks to trimming)
salmonquant	quantify RNA-seq data with salmon	Keeping *all* salmon outputs - quite a lot: might be useful longterm

6/20/23 *split into salmonquantSE and salmonquantPE*		In /salmonout/<sample>_<dip reference descrip> - one dir per
alntoolssalm2ec	Run alntools salmon2ec to get input for emase-zero Only runs when --rnamap is salmon or all **Note: this WILL NOT WORK if transcripts have underscores in them. Must use underscore-free GTF if so.**	No
emasezerosalm	Runs emase-zero on salmon output. Uses length correction. Only runs if --rnamap is salmon or all	4 files as above, but with 'salmon' in filename to specify inputs were salmon quantifications. In /emase
salmonalleleeqclasses	Runs salmonalleleeqclasses.py to get eq, allele-specific information per transcript and gene only runs if --rnamap is salmon or all	Per sample: _eqclasses_transcripts.txt.gz _eqclasses_genes.txt.gz In /chareqclasses/persample
eqclassalnmtsummary	Runs eqclassalnmtsummary_multsamples.R	All samples together; exact number depends on parameters - In chareqclasses/

ORNAMENTS: data_generation_scripts/ornaments/ornaments-elegans-dsl2.nf

- Parameters / inputs

Category	Flag for script (in script as params.<this)	Default value (if highlighted, need to provide)	Description
Data input	--strains	""	Path to no-header list of strains to process - should be all strains any sample comes from
Data input	--sampleinfo	""	Path to tab-delimited file containing sample information. Column names (descriptions): SampleID (sample ID as in input filenames, to be used in output filenames); Strain (strain whose strain-specific ornaments/kallisto index should be used); RefDescrip (description of reference genome(s) to use in output file names)
Data input	--out	""	Parent output directory. Will be created if doesn't exist.
Data input	--fastqdir	""	Directory containing all fastq.gz files to process. One or more per sample. **for paired end, this assumes fastqs are internal to sample-named directories here.
Data input	--vcf	WI.20220216.hard-filter.isotype.vcf.gz	path to VCF containing all samples of interest.
Data input	--gtf	c_elegans.PRJNA13758.WS283.canonical_geneset.gtf	Path to GTF annotation file **same build/version/format/etc as

			used in VCF creation and transcritome FASTA
Data input	--trfa	c_elegans.PRJNA13758.WS283.mRNA_transcripts.fa	Path to transcriptome fasta **same build/version/format/etc as used in VCF creation and GTF **__needs to be unzipped__
run kallisto?	--kallisto	true	if true, kallisto run on strain-specific indexes (after creating them), if not, it isn't
trimmomatic	--trimmodir		Path to trimmomatic v0.39 directory containing jar file and adapters directory (which itself contains the fasta files provided in next option, e.g. TruSeq3-SE.fa).
trimmomatic	--trimmofa	TruSeq3-PE-2.fa	name of fasta file within trimmodir adapters/ directory matching those used in library preparation
trimmomatic	--trimmoseedmism	1	Input to trimmomatic ILLUMINACLIP. How many of 16 bp can mismatch and still be counted as match.
trimmomatic	--trimmoadapclipthresh	12	Input to trimmomatic ILLUMINACLIP. How accurate match between adapter sequence and read must be. Each correct base adds 0.6. They recommend 7-15 (12 bases needed for 7, 25 for 15).
organizational	--orndir		Path to ornaments directory where scripts convert_genomic_vcf_to_transcriptomic_vcf.py and reate_personalized_transcriptome.py are

orgnanizational	--ornconda		path to conda environment set up to run ornaments & associated python scripts (as described in ornaments doc)
Post-ornaments R script	--ornamentsrscript	true	if true, runs ornaments_initplots_data2gene.R. Only need to fill in the next options if true.
Post-ornaments R script	--rscriptdir		// Directory containing ornaments_initplots_data2gene.R script
Post-ornaments R script	--plotsampinfo	""	path to plot script formatted sample info: Path to sample information file containing // information on parental and F1 samples. Columns // should include: SampleID, Generation (Parental or // F1), Allele1 (NA for parental, reference strain // for F1), Allele2 (NA for parental, non-reference // strain for F1 - used as Strain for F1 in // generation/strain model), Strain (NA for F1, // strain for parental. All strains/alleles should // be the way you want them to show up in outputs)
Post-ornaments R script	--plotbaseoutname	out	Base name for all plot output files

Post-ornaments R script	--plotstrains	""	Path to strains ordered as desired in plot. **Strains, not isotypes, probably
Post-ornaments R script	--tx2genef		// Path to file mapping transcripts to genes. Two columns (transcript ID, gene ID), no header.
Post-ornaments R script	--genegff		// Path to *genes only* gff3 file containing info on all gene_ids present in input counts file; includes gene location, name, biotype and other information.
Post-ornaments R script	--plotexclchrs	"mtDNA"	// Optional comma-separated, no space list of // ch romosomes to exclude entirely from plots/downstream analyses (named as in // ge negff). E.g. MtDNA cat be smart to exclude for // AS E analysis.
Post-ornaments R script	--plotinclbiotype	protein_coding	Comma-separated list of biotypes (as in genegff) to include in plot processing

- Processes & outputs

Name	**Description**	**Any saved outputs? And relevant notes**
makestrainvcf	Makes a VCF for one strain that artificially has only the variants where this strain has an alt genotype, puts 0|1 for the genotype field [so, this works when the strains in the input VCF are isogenic and are crossed with the reference for this experiment]	no, don't need/use after workflow
transcriptomicvcf	Runs ornaments' convert_genomic_vcf_to_transcriptomic_vcf	no, don't need/use after workflow

	.py, which "extracts transcriptome variants located in exonic regions and transforms the genomic coordinates of variants to transcriptomic coordinates."	**The alleles in the output VCF are in terms of transcript (**can see genome allele in the .coords output file - might be one to consider keeping/gzipping: has number of variants used in that transcriptome)
ornamentpersonalized	Runs ornaments' create_personalized_transcriptome.py, which "odifies the reference transcriptome with the variant information that was prepared above, and produces an ornament personalized transcriptome."	no, don't need/use after workflow could consider saving this…
ornamentsindex	Builds ornaments index	no, don't need/use after workflow
diptranscriptome	Runs ornaments' create_personalized_transcriptome.py to get diploid transcriptome *Only run if kallisto flag enabled*	no, don't need/use after workflow
straintranscriptome	Extracts the _R transcripts from diploid transcriptome to get strain-specific transcriptome *Only run if kallisto flag enabled*	could consider saving this…
kallistoindex	Runs kallisto index on personalized transcriptome from diploid (not ornaments one!) *Only run if kallisto flag enabled*	no
kallistodipindex	Runs kallisto index on diploid transcriptome *only run if kallisto flag enabled*	no
trimmoIlluminaAdapters PE	Run per sample. Use trimmomatic to trim Illumina adapters from merged fastqs **when they are PE** // **currently PE assumed to be named _1 and _2.fastq.gz	trim logs in triminfo/ dir (doesn't save unpaired reads currently)

ornamentsquant	Run ornaments quant on individual samples	All ornaments quantification output in ornaments/ directory (each sample has its own directory)
kallistoquant	Runs kallisto quant. **currently sets library as --rf-stranded, need to CHECK THIS! *Only run if kallisto flag enabled*	All kallisto on strain-specific genomes quantification in kallisto/ directory NB does for all samples, probably only want to use for parentals
kallistodipquant	Runs kallisto quant to diploid index. ** really only makes sense for F1 samples, but currently does for all for ease ***Only run if kallisto flag enabled*	All kallisto on diploid in kallisto/diploid directory NB does for all samples, probably only want to use for F1s
ornamentsinitplots	Runs ornaments_initplots_data2gene.R *Only run if ornamentsrscript flag true*	All plotting and ornaments per sample outputs of script in ornamentsinitanalysis/ output subdirectory

Implementing DE and ASE modeling: data_classification_scripts/ase_de_annotategenes_deseq2_fromemaseout.R

- Inputs

-s, --sampleinfo Path to sample information file containing information on parental and F1 samples. Columns should include: SampleID, Generation (Parental or F1), Allele1 (NA for parental, reference strain for F1), Allele2 (NA for parental, non-reference strain for F1 - used as Strain for F1 in generation/strain model), Strain (NA for F1, strain for parental. All strains/alleles should be the way you want them to show up in outputs), terms in --aseotherterms and --genstrainmodel should be column names if there are other terms provided not already accounted for.

-b, --baseoutname Base name for all output files [default: out]

-o, --outdir Outer output directory. Sub-directories will be created internally.

-e, --emasedir Path to directory with one EMASE output .gene.counts (or .gene.counts.gz) per sample. Filenames must start with sample ID (from sampinfof1, sampinfoparent)

-a, --allelerename Optional - allele mapping/renaming between files. Path to file with columns emase, sample: initial should be alleles as in EMASE outputs; final should be alleles as in sample information!

--aseotherterms (optional) Term(s) (columns of sampinfo) to include in ASE model before allele. ONLY TESTED WITH ONE PROVIDED. Comma-separated if multiple. If one provided, design will be ~<this> + <this>:Sample[grouped as needed] + Allele

-r, --refcategoryinfo Path to matrix describing the reference level for ALL factors in the ASE model, even if you

don't care (Allele, other terms). Columns 'colname', 'reflevel'. reflevel for allele should be as in sample info, not emase. **Important: ref level for Allele is assumed to be reference strain to be used as denominator for comparisons here and in generation/strain model

--alpha Alpha p-value threshold for FDR-like filtering (used for ASE and Generation/Strain group comparisons). [default: 0.1]

--alleleskewthresh proportion alleles from one haplotype OR the other for results to be considered significant. I.e., if 0.6, genes with >=60% of one allele (<=40% or >=60% alt. alleles) AND significant p-values are considered significant. Leave at default 0.5 to just threshold on p-value. [default: 0.5]

-g, --genstrainmodel Model design for model including all samples (parental and F1). Last term should be GenerationStrain - this will be made by combining generation and strain (or allele2 for F1s) to be able to compare among parental strains and between F1s and their parents. Include any batch or other covariates to regress out! [default: ~GenerationStrain].

--genstrainlfc Log2FoldChange threshold for genes to be called significant in GenerationStrain model (so, for parents vs. each other and F1 vs. each parent). Not used for testing, rather to categorize results. Consider matching this to alleleskewthresh (i.e. an alleleskewthresh of 0.6 is equivalent to a fold-change threshold of 1.5, LFC threshold of 0.5849625 [default: 0.5849625]

--genegff Path to *genes only* gff3 file containing info on all gene_ids present in input counts file; includes gene location, name, biotype and other information.

--exclchrs Optional comma-separated, no space list of chromosomes to exclude entirely (named as in genegff). E.g. MtDNA cat be smart to exclude for ASE analysis.

-i, --inclbiotype Comma-separated list of biotypes (as in genegff) to include in processing [default: protein_coding]

--exampuniqeqcts Example filepath to per-sample counts of eq classes, haplotype-specific eq classes, gene-specific eq classes for each _gene_ (*eqclasses_genes.txt.gz output of salmonalleleeqclasses.py). Where sample ID (as in --sampleinfo) goes in filename, should have string SAMP. One file should exist for each F1 sample. For any other differences among filenames (e.g. genome aligned to), include the glob '*' asterisk character

--strain2iso Path to strain-to-isotype mapping file for all non-reference strains included here (even if strain and isotype are identical!). Columns strain, isotype; one row per included strain.

-l, --leehypdivbed Filepath to hyperdivergent haplotypes BED file as downloaded from CeNDR 20210121 release (published as Lee et al 2020/2021 preprint/pub)

--genednacov Path to file containing information on DNA coverage per gene (e.g. *_genecoveragefromexons_mednorm.txt.gz output of exploregenecoverage_fromexons.R/mosdepthmergedexons.nf workflow). Columns must include gene_id, <named for all isotypes matching strains in RNA-seq data>. Each isotype-named column contains DNA coverage value to be added (e.g. gene median-normalized coverage) - no downstream processing/normalization done on these.

-d, --dnalowcovthresh Threshold below which DNA coverages (proved in --genednacov) will be flagged as low coverage

in column of output (lowDNACov) [default:
0.25]

--dnahighcovthresh Threshold above which DNA coverages (proved in --genednacov) will be flagged as high coverage in column of output (highDNACov) [default: 2]

- Outputs
  - DESeq2 objects
    - *_dds_ase.RData - DESeq2 object of ASE analysis, including results.
    - *_dds_genstrain.RData - DESeq2 object where all samples were included, generation/strain combination as explanatory.
  - Extraneous/side effect-related
    - PCA plots. In /pcaplots directory. [*added later]*
      - *_dds_genstrain_vst_pcaplots.pdf - Generation and Strain annotated
      - *_dds_genstrain_vst_pcaplots_with<first element from aseotherterms>.pdf - only made if --aseotherterms provided; differentiates the specified term based on size on the plot.
  - Main results tables - one per strain
    - File name: *_<Strain>_annotatedASEDEresults.txt.gz
    - One row per retained genes (# of these printed out)
    - Columns described in groups below if names not self explanatory. This is in their current order as well..
    - *Gene metadata from GFF*
      - gene_id, main gene ID (WB### format if C. elegans)
      - display_name, locus if there's a locus ID, otherwise sequence_name
      - locus, abc-1 format locus name (if C. elegans)
      - sequence_name, YXXX format sequence name (if C. elegans)
      - biotype
      - chr
      - start
      - end
      - strand
    - *Added gene metadata - hyperdivergence, gene coverage*
      - hypdiv, TRUE or FALSE: does gene overlap hyperdivergent haplotype (Lee et al 2020, as represented in CeNDR 2021.01.21 BED download) in isotype-representative strain matching this strain. Note: 0 vs. 1 indexing not changed or considered; any 1bp overlap counts.
      - dnacoverage, DNA coverage value for this gene *exactly as provided in* --genednacov file.
      - lowDNACov, TRUE or FALSE: is dnacoverage value below --dnalowcovthresh provided value
      - highDNACov, TRUE or FALSE: is dnacoverage value below --dnalowcovthresh provided value
    - *Gene informativeness/number of unique alignments*
      - unqalnmts.nOver0, number of F1 samples with more than 0 gene/haplotype unique alignments (from data in --exampuniqeqcts)

    - unqalnmts.allOver0, TRUE or FALSE: do all F1 samples have more than 0 gene/haplotype unique alignments (from data in --exampuniqeqcts)
    - unqalnmts.min, Minimum number of gene/haplotype unique alignments across F1 samples (from data in --exampuniqeqcts) (*useful for thresholding for informative genes!)*
    - unqalnmts.med, Median number of gene/haplotype unique alignments across F1 samples (from data in --exampuniqeqcts)
- *Statistical test result: ASE*
    - baseMean.ASE, DESeq2 gene expression mean value from F1 only ASE model (likely not super useful; size corrections etc. not used for this allele-vs-allele model)
    - log2FoldChange.ASE, DESeq2 calculated ashr-shrunken log2FC, non-reference-strain allele vs. reference-strain allele
    - lfcSE.ASE, standard error on above
    - pvalue.ASE, uncorrected p-value for ASE model fitting
    - padj.ASE, genome-wide adjusted p-value for ASE model fitting
    - altVref, non-reference-strain alleles / reference strain alleles ratio at this gene (from log2FoldChange.ASE)
    - altVtotal, proportion of alt (non-reference-strain) alleles at this gene (from log2FoldChange.ASE)
    - signifAtThresholds.ASE, TRUE or FALSE: does this gene exceed magnitude threshold (magnitude exceeds --alleleskewthresh: if this is 0.6, must have <40% or >60% alleles from one haplotype) and p value threshold (genome-wide adjusted p < --alpha)
- *Statistical test result: strain vs. strain (parental) & generation vs. generation. Note that non-reference is always numerator in strain vs. strain, F1 is always numerator in F1 vs. parental*
    - baseMean.GenerationStrain, DESeq2-calculated gene expression mean from all samples (F1 and parental)
    - log2FoldChange.ParentVsParent<ReferenceStrain>, DESeq2 calculated ashr-shrunken log2FC, non-reference parent vs. reference-strain parent
    - lfcSE.ParentVsParent<ReferenceStrain>, standard error on above
    - pvalue.ParentVsParent<ReferenceStrain>, uncorrected p-value for above
    - padj.ParentVsParent<ReferenceStrain>, genome-wide adjusted p-value for above
    - signifAtThresholds.ParentVsParent<ReferenceStrain>, TRUE or FALSE: does this gene exceed p value threshold (genome-wide adjusted p < --alpha) and magnitude threshold (absolute value log2FC > --genstrainlfc) in this test
    - log2FoldChange.F1VsParent<ReferenceStrain>, DESeq2 calculated ashr-shrunken log2FC, F1 vs. reference-strain parent
    - lfcSE.F1VsParent<ReferenceStrain>, standard error on above
    - pvalue.F1VsParent<ReferenceStrain>, uncorrected p-value for above
    - padj.F1VsParent<ReferenceStrain>, genome-wide adjusted p-value for above

- signifAtThresholds.F1VsParent<ReferenceStrain>, TRUE or FALSE: does this gene exceed p value threshold (genome-wide adjusted p < --alpha) and magnitude threshold (absolute value log2FC > --genstrainlfc) in this test
- log2FoldChange.F1VsParentNonRef, DESeq2 calculated ashr-shrunken log2FC, F1 vs. non-reference-strain parent
- lfcSE.F1VsParentNonRef, standard error on above
- pvalue.F1VsParentNonRef, uncorrected p-value for above
- padj.F1VsParentNonRef, genome-wide adjusted p-value for above
- signifAtThresholds.F1VsParentNonRef, TRUE or FALSE: does this gene exceed p value threshold (genome-wide adjusted p < --alpha) and magnitude threshold (absolute value log2FC > --genstrainlfc) in this test

Analyzing reference bias: data_analyis_scripts/ ase_refbias_withinstrain.R

- Inputs (get by running script with --help)

```
-i, --input        Example filepath to ASE results input - output of
                   ase_de_annotategenes_deseq2_fromemaseout.R or
                   ase_de_annoategenes_deseq2_ornaments.R. (see that
                   script's documentation for format details).
                   **Where strain is identified in filepath, write
                   STRAIN e.g. STRAIN_annotatedASEDEresults.txt.gz
-a, --aseformat    'emase' or 'ornaments': how was ASE data in above
                   matrix generated (with
                   ase_de_annotategenes_deseq2_fromemaseout.R or
                   ase_de_annoategenes_deseq2_ornaments.R?)
                   [default: emase]
-s, --strains      Strains to read in results for and process
                   together. Either comma-separated (no spaces) list
                   or path to no-header file with one line per
                   strain.  Must match how strains are named in
                   input filenames.
-b, --baseoutname  Base name for all output files [default: out]
-o, --outdir       Outer output directory. Sub-directories will be
                   created internally. **NB: if you provide getwd()
                   here (quote wrapped), current directory will be
                   used
--alpha            Alpha p-value threshold USED for FDR-like
                   filtering in input signifAtThresholds.ASE column.
                   (Used here for drawing appropriate plot
                   thresholds) [default: 0.05]
--alleleskewthresh  proportion alleles from one haplotype OR the
                   other for results to be considered significant
                   USED in input signifAtThresholds.ASE column.
                   (Used here for drawing appropriate plot
                   thresholds).  I.e., if 0.6, genes with >=60% of
                   one allele (<=40% or >=60% alt. alleles) AND
                   significant p-values were considered significant.
                   [default: 0.6]
```

- Outputs
  - *_numbers_ase_genewisealleleskew.txt - Summary of number of genes in each set in each strain, number and proportion with ASE, as well as number with reference vs. alt allelic skew and some statistical tests on whether this looks globally reference biased. *All numbers come*

from proportion alt alleles determined by DESeq2; significance thresholds given when data was generated. Columns:

- **result**, describes strain
- **geneset**, description of gene set used (see 'current gene sets' above)
- **ngenes**, number of genes included in this gene set/strain combination
- **nASE**, number of genes with ASE called (significance thresholds given when data was generated)
- propASE, proportion of genes in this set with ASE called (significance thresholds given when data was generated)
- **nASE.ref,** number of genes with ref-biased ASE (signifAtThresholds.ASE was T and altVtotal < 0.5)
- **propASE.ref**, proportion genes with ref-biased ASE
- **nASE.alt**, number of genes with alt-biased ASE (signifAtThresholds.ASE was T and altVtotal > 0.5)
- **propgenesMoreRefAlleles**, proportion genes in this set with more reference alleles per DESeq2 log2FC
- **propASE.alt**, proportion genes with ref-biased ASE
- **propgenesMoreRefAlleles**, proportion genes in this set with more reference alleles per DESeq2 log2FC
- propASE.low95ci, 95% binomial confidence interval on ASE proportion (lower bound; mostly for plots)
- propASE.high95ci, 95% binomial confidence interval on ASE proportion (upper bound; mostly for plots)
- propASE.ref.low95ci, 95% binomial confidence interval on ref-biased ASE proportion (lower bound; mostly for plots)
- propASE.ref.high95ci, 95% binomial confidence interval on ref-biased ASE proportion (upper bound; mostly for plots)
- propASE.alt.low95ci, 95% binomial confidence interval on alt-biased ASE proportion (lower bound; mostly for plots)
- propASE.alt.high95ci, 95% binomial confidence interval on alt-biased ASE proportion (upper bound; mostly for plots)
- propASE.low95ci, 95% binomial confidence interval on ASE proportion (lower bound; mostly for plots)
- propASE.high95ci, 95% binomial confidence interval on ASE proportion (lower bound; mostly for plots)
- propgenesMoreRefAlleles.low95ci, 95% confidence interval on above proportion (lower bound)
- propgenesMoreRefAlleles.high95ci, 95% confidence interval on above proportion (upper bound)
- **binomtest.pval.2sided**, binomial test p-value on this proportion vs. 0.5 (two-sided)
- **binomtest.pval.refgreater**, binomial test p-value on this proportion vs. 0.5 - for the hypothesis that more are reference biased (one-sided)
- **wilcoxtest.diff50.pval.2sided**, wilcox rank test p-value for differences from 0.5 (even alleles) for genes split by whether they have more reference alleles called by DESeq2 or not - about MAGNITUDE of skew. Two sided (testing hypothesis that there is no median difference)
- **wilcoxtest.diff50.pval.refgreater**, wilcox rank test p-value for differences from 0.5 (even alleles) for genes split by whether they have more reference alleles called by

DESeq2 or not - about MAGNITUDE of skew. One sided (testing hypothesis that there is more REFERENCE skew)
    - **propRefAlleles.median**, median proportion reference alleles across genes
    - **wilcoxtest.medianV50.pval.2sided**, wilcox one sample test p-value for median prop ref alleles being different from 0.5
    - **wilcoxtest.medianV50.pval.refgreater**, wilcox one sample test p-value for median prop ref alleles being greater than 0.5
- Global allele skew/reference bias related: in /globalalleleskew directory
    - *_geneskewfrom50plots_<gene set short name as described in gene set information>.pdf: One file per gene set, Overlapping histograms showing the DESeq2-estimated distance from 50% reference alleles for all genes with any ref-bias vs. all genes with any alt-bias (so can compare overlap rather than left vs. right). 2 pages per result/sample group: first is basic/absolute plot, second has number of genes (y axis) on log10 scale
    - *_propgenesrefskew_forestplots.pdf - forest plot with one point per strain/treatment group showing global proportion of GENES with >50% reference alleles as inferred by DESeq2. One page per gene set.
    - *_propgenesrefskew_forestplots_onepage.pdf - same plot as above, but now all gene sets are plotted as facets in the same plot for quicker comparison across gene sets.
- Per-gene proportion reference alleles related: in /proprefpergene directory
    - *_proprefalleleshists_<gene set short name as described in gene set information>.pdf: One file per gene set, histogram showing the DESeq2 estimate of proportion reference alleles for each gene. 2 pages per result/sample group: first is basic/absolute plot, second has number of genes (y axis) on log10 scale
    - *_proprefalleles_violplots.pdf: All strains on one plot - each has one violin showing distribution of proportion reference alleles called for each gene. Each page is a different gene set. (So, like smushing the strains' histograms together)
- Volcano plots: in /volcanoplots
    - NOTE: these omit any with NA p-value (less of a concern given they're narrowed to informative)
    - *_volcanoplots_allgenesets_covsizehypdivoutline.pdf - faceted by strain; one point per gene; colored by significance; Upward-pointing triangle at 1e-10 means p<1e-10. Each point's size is scaled to DNA coverage (as in input); Red outline (not always of right size) is added if gene is in hyperdivergent haplotype. One page per gene set (so last pages have no hyperdivergent genes and therefore no outlined points).
    - *_volcanoplots_allgenesets_lowcovoutline.pdf - as above except no hyperdivergence information in plot and all points are the same size; now, points are outlined in red if flagged as low DNA coverage
- Called ASE related, in: /aseplots
    - *_propgenesASE_forestplots_onepage.pdf - forest plot with one point per strain showing proportion of genes called ASE in this gene set (numerator: number of ASE genes in a given gene set; denominator: total number of genes in that gene set). Each gene set is facet of the same plot.
    - *_propgenesrefbiasedOfASE_forestplots_onepage.pdf -forest plot with one point per strain showing proportion of ASE genes that are reference biased (numerator: reference-biased ASE genes; denominator: all ASE genes - so expect 50%; wide CIs

because both numerator and denominator are small). *Helpful for thinking about reference bias.*

- *_propgenesDiffDirsASE_forestplots_onepage.pdf - forest plot with multiple points per strain showing proportion genes ref-biased ASE, alt-biased ASE, either direction ASE. Shape delineates ASE direction. First page has total ASE as well as component ref- and alt-biased proportions; second page has ref and alt (no total)

- Hyperdivergent haplotype genes only, all in: /hypdivgenes
  - All are as described above, but only for hyperdivergent genes; have _hypdivgenes_ in the file names:
  - *_hypdivgenes_numbers_ase_genewisealleleskew.txt - as *_numbers_ase_genewisealleleskew.txt, but for ONLY hyperdivergent genes with different levels informativeness
  - *_geneskewfrom50plots_<hyp div gene set description>.pdf
  - *_hypdivgenes_propgenesrefskew_forestplots_onepage.pdf
  - *_hypdivgenes_propgenesASE_forestplots_onepage.pdf
  - *_hypdivgenes_propgenesrefbiasedOfASE_forestplots_onepage.pdf
  - *_hypdivgenes_propgenesDiffDirsASE_forestplots_onepage.pdf

Classify inheritance mode: data_classification/scripts/f1_parental_inhmode_withinstrain.R

- Classification details

classification	Parental (alt vs. N2)	F1 vs. N2	F1 vs. other parent	description
no_change	none (padj < 0.1 | FC > 1.5)	none (padj > 0.1 | FC < 1.5)	none (padj < 0.1 | FC < 1.5)	no DE - F1 expressed same as both parents as far as we can tell
overdominant	NA (doesn't matter)	+ (LFC > log2(1.5), padj < 0.1) - F1 higher expressed	+ (LFC > log2(1.5), padj < 0.1) - F1 higher expressed	F1 has higher expression than either parent (regardless of if parents have same expression)
underdominant	NA (doesn't matter)	- (LFC < -log2(1.5), padj < 0.1) - F1 lower expressed	- (LFC < -log2(1.5), padj < 0.1) - F1 lower expressed	F1 has lower expression than either parent (regardless of if parents have same expression)
N2_dominant	- (LFC > log2(1.5), padj < 0.1 - N2 lower expressed)	none (padj > 0.1 *only*)	- (LFC < **0**, padj < 0.1) - F1 lower expressed	F1 has same expression as N2, while N2 and F1 are significantly lower expressed than other parent. *Only one higher expression, here among parents, has to be higher than magnitude threshold*
	+ (LFC < -log2(1.5), padj < 0.1 - N2 higher expressed)	none (padj > 0.1 *only*)	+ (LFC > **0**, padj < 0.1) - F1 higher expressed	F1 has same expression as N2, while N2 and F1 are significantly higher expressed than other parent. *Only one lower expression, here among parents, has to exceed magnitude threshold*
	- (LFC > **0**), padj < 0.1 - N2 lower expressed)	none (padj > 0.1 *only*)	- (LFC < -log2(1.5), padj < 0.1) - F1 lower expressed	F1 has same expression as N2, while N2 and F1 are significantly lower expressed than other parent. *Only one higher expression, here between F1 and parent, has to be higher than magnitude threshold*
	+ (LFC < **0**, padj < 0.1 - N2 higher expressed)	none (padj > 0.1 *only*)	+ (LFC > log2(1.5), padj < 0.1) - F1 higher expressed	F1 has same expression as N2, while N2 and F1 are significantly higher expressed than other parent. *Only one lower expression, here between F1 and parent, has to exceed magnitude threshold*

alt_dominant	- (LFC < log2(1.5), padj < 0.1 - alt lower expressed)	- (LFC < **0**, padj < 0.1) - F1 lower expressed	none (padj > 0.1 *only*)	F1 has 'same expression as alt, while alt and F1 are significantly lower expressed than N2 parent. *Only one lower expression, here among parents, has to be lower than magnitude threshold*
	+ (LFC > -log2(1.5), padj < 0.1 - alt higher expressed)	+ (LFC > **0**), padj < 0.1) - F1 higher expressed	none (padj > 0.1 *only*)	F1 has same' expression as alt, while alt and F1 are significantly higher expressed than N2 parent. *Only one higher expression, here among parents, has to be higher than magnitude threshold*
	- (LFC < **0**, padj < 0.1 - alt lower expressed)	- (LFC < -log2(1.5), padj < 0.1) - F1 lower expressed	none (padj > 0.1 *only*)	F1 has 'same expression as alt, while alt and F1 are significantly lower expressed than N2 parent. Only one lower expression, here between F1 and parent, has to be lower than magnitude threshold
	+ (LFC > **0**), padj < 0.1 - alt higher expressed)	+ (LFC > log2(1.5), padj < 0.1) - F1 higher expressed	none (padj > 0.1 *only*)	F1 has same' expression as alt, while alt and F1 are significantly higher expressed than N2 parent. *Only one higher expression, here between F1 and parent, has to be higher than magnitude threshold*
additive	- (LFC < -log2(1.5), padj < 0.1 - alt lower expressed)	- (padj < 0.1, LFC < 0 *only*)	+ (padj < 0.1, LFC > 0 *only*)	alt < F1 < N2 (require 1.5-fold for alt vs. N2, significantly different LFC from 0 in the expected directions for F1 vs. parents). If DE from one parent, must be DE opposite-ly from the other parent!
	+ (LFC > log2(1.5), padj < 0.1 - alt higher expressed)	+ (padj < 0.1, LFC > 0 *only*)	- (padj < 0.1, LFC < 0 *only*)	alt > F1 > N2 (require 1.5-fold for alt vs. N2, significantly different LFC from 0 in the expected directions for F1 vs. parents). If DE from one parent, must be DE opposite-ly from the other parent!
ambiguous	*all other combinations,*	*e.g. parents aren't different but F1 is called different from only one of them*		

- Inputs

-i, --input Example filepath to ASE results input - output of ase_de_annotategenes_deseq2_fromemaseout.R (see

that script's documentation for format details).
**Where strain is identified in filepath, write
STRAIN e.g. STRAIN_annotatedASEDEresults.txt.gz

-b, --baseoutname Base name for all output files [default: out]

-o, --outdir Outer output directory. Sub-directories will be created internally.

-s, --strains Strains to read in results for and process together. Either comma-separated (no spaces) list or path to no-header file with one line per strain. Must match how strains are named in input filenames.

-r, --refstrain Name of reference strain, matching name used in columns of input. Also used for plot labelling. [default: N2]

-a, --alpha P-value threshold for considering gene DE for *adjusted p-value*. Combined with magnitude threshold (--lfcthresh) for many inheritance mode classifications, used on its own in some cases where other DE used as a prior. [default: 0.1]

-l, --lfcthresh log2FoldChange threshold for considering gene DE when combined with --alpha. [default: 0.5849625]

-g, --genomebp Length of reference genome in bp. Used to get variants vs. reference per kb. Default is genome length from NCBI https://www.ncbi.nlm.nih.gov/assembly/GCF_000002985.6/ - for ws235 but unlikely to have changed much (and not enough to be meaningful for our purposes) [default: 100286401]

-v, --varsvsref Path to file containing columns strain (one row for each strain in main input), nvars (number variants vs. reference genome)

- Outputs
  - *Same as previous; will include ambiguous category where relevant*
  - _inheritancemode_pergenestrain.txt.gz - data with inheritance mode classification for each gene in each strain. Columns strain, gene_id, inhmode (defined as in table above)
    - *NB: went back and forth on whether to save all data again with this inheritance mode column added; eventually decided to just save this information and will merge it in with some data 'freeze' when have cis/trans & similar classifications - that way, can change how I call this if needed.*
  - *Number summaries*

- *_numbers_f1parentsdeinhmode.txt: Number of genes per strain, gene set with different DE AND as classified into each of the inheritance patterns. Columns (for inheritance mode columns, see classification table above for exactly how arrived at):
    - strain, strain this result is for
    - geneset, name of gene set (short format name)
    - ngenes, # genes in this gene set
    - ngenes_notconserved, # genes with non-conserved inheritance pattern [so can get breakdown of this]. *In theory this is same as DEvEither!*
    - DEv<refstrain>, # genes with F1 vs. refstrain DE called
    - DEvNonRef, # genes with F1 vs. alt strain DE called
    - DEvEither, # genes with F1 vs. either parent DE called
    - DEvBoth, # genes with F1 vs. both parent DE called
    - *<one column per inheritance mode!!>*
- e set each of the above numbers is. Long format for easy plotting. Columns:
    - strain, strain this result is for
    - geneset, name of gene set (short format name)
    - ngenes, # genes in this gene set
    - ngenes_notconserved, # genes with non-conserved inheritance pattern [so can get breakdown of this]
    - propName, which N is numerator of proportion (see descriptions above).
    - propDenom, ngenes or ngenes_notconserved: what is proportion OF? ***CAREFUL with ngenes_notconserved proportions - squishes out any real difference in total # not conserved genes, which is relevant to many things
    - prop, value of proportion
    - lowci, binomial 95% CI lower bound
    - highci, binomial 95% CI upper bound

○ *Bar plots - inheritance mode breakdown*

- *_inhmode_propofallgenes_allsets.pdf: Stacked bar plot, one bar per strain, showing inheritance mode breakdown of genes (proportion of total # genes in each set). Faceted by gene set (better for looking at all at once, though not critical here for one plot and 3 gene sets.)
- *_inhmode_propofallgenes_indivplots.pdf - as above, but one plot per page (no facets)
- *_inhmode_propofchangedgenes_allsets.pdf: Stacked bar plot, one bar per strain, showing inheritance mode breakdown of **non-conserved** genes as a proportion of number of non-conserved genes - to be able to more easily see the small minority of non-conserved genes. Faceted by gene set (better for looking at all at once, though not critical here for one plot and 3 gene sets.)
    - ****this is normalized to # not no_change so be careful as that may skew interpretation.** E.g., EG4348 has many more non-conserved genes total and this information is lost in these plots
- *_inhmode_propofchangedgenes_indivplots.pdf - - as above, but one plot per page (no facets)

○ *Scatter plots - log2FC vs. Ref against log2FRC vs. alt*

- In directory lfcVlfcplots
- *_lfcVlfcplots_allgenesets_plain.pdf: each page has all points for one gene set. 4 strains per page.

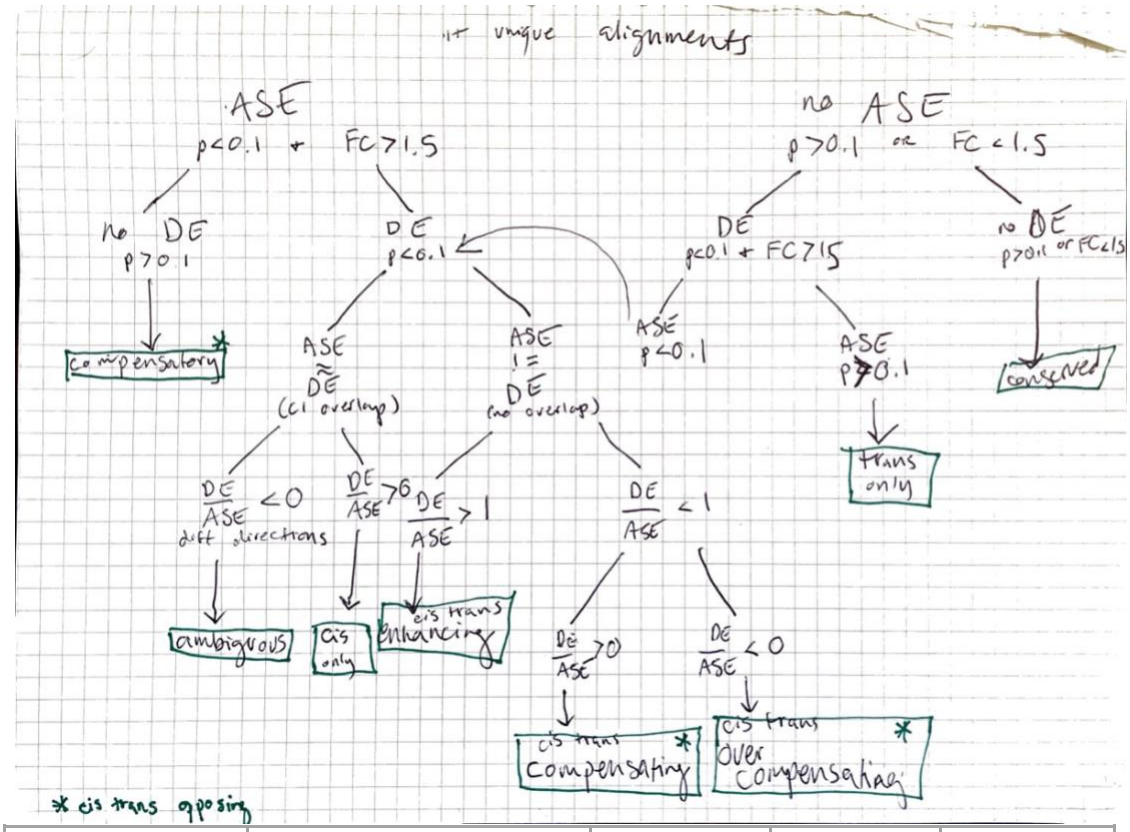
- *_lfcVlfcplots_allgenesets_hypdivoutline.pdf: as above, but hyperdivergent genes are outlined. Only relevant for the 'all' gene set (pre-excluded from others!)
- *_lfcVlfcplots_allgenesets_aseinformoutline.pdf: here, ASE-informative genes *as coded in script currently* (so, at beginning, 2+ unique alignments AND excluding hypdiv) are outlined. Not relevant for when genes in this gene set are the only ones plotted, of course.

○ *Proportion vs divergence* - helpful for comparing across strains even if divergence isn't of most interest
- in directory propvdivergenceplots
- *_propvdivergence_allinhplots_allsetsfacets.pdf - divergence from N2 (# variants) vs. proportion genes in various inheritance or related classifications. Whether proportion is of all genes or of non-conserved genes is noted in title, y axis label. *Values described in* _numbers_f1parentsdeinhmode.txt *description*. Each page has all 3 gene set facets(better for looking at all at once)
- _propvdivergence_allinhplots_<gene set >.pdf - as above but one page per proportion plot, within one gene set only. (no facets)

○ *Miscellaneous other analyses*
- In directory subanalyses/
- *Proportion genes with reference-dominant inheritance patterns vs. alt-dominant inheritance patterns* (to be able to quickly see which F1s favor ref or alt more heavily)
  - *_refdominantValtdominant_allsetsfacets.pdf - one page faceted by gene sets. X axis is proportion of genes in set that were classified as reference dominant strain inheritance pattern, Y axis is proportion of genes in set that were classified as non-reference dominant strain inheritance pattern
  - *_refdominantValtdominant_indivplots.pdf - as above but one page per gene set plot

Classify cis, trans regulation: data_classification_scripts/ ase_de_cistransclassifications.R

- Classifications



Classification	notes	ASE?	DE?	Additional
conserved	no regulatory divergence (different than no expression difference - larger category!)	no (p > 0.1 & FC < 1.5)	no (p > 0.1 & FC < 1.5)	
cis	Strain differences are underpinned solely (or at least largely) by cis changes (*x = y set of points in scatter plot)*	yes (p < 0.1 & FC > 1.5)	yes (p < 0.1)	ASE and DE in same direction (DE/ASE >0 log2FC space) ASE and DE LFCs' confidence intervals *overlap* (diff. CI

				thresholds tried for all)
trans	Strain differences are underpinned solely (or at least largely) by trans changes (*horizontal spread of points in scatter plot*)	no ($p > 0.1$)	yes ($p < 0.1$ & FC > 1.5)	
enhancing	There are both cis and trans regulatory changes, which act to change expression between the strains in the same direction	yes ($p < 0.1$ and FC > 1.5 OR $p < 0.1$ if there's DE with $p < 0.1$ & FC > 1.5)	yes ($p < 0.1$ if there's ASE; $p < 0.1$ & FC > 1.5 if ASE doesn't meet both thresholds)	ASE and DE LFCs' confidence intervals do not overlap DE/ASE > 1 (log2FC space)
compensating	There are both cis and trans regulatory changes, which act in opposite directions. The cis effect is 'larger' such that difference between strains is in the direction of the cis effect, but smaller than the cis effect.	yes ($p < 0.1$ and FC > 1.5 OR $p < 0.1$ if there's DE with $p < 0.1$ & FC > 1.5)	yes ($p < 0.1$ if there's ASE; $p < 0.1$ & FC > 1.5 if ASE doesn't meet both thresholds)	ASE and DE LFCs' confidence intervals do not overlap 0 > DE/ASE > 1 (log2FC space)
compensatory	There are both cis and trans regulatory changes, which act in opposite directions. The trans changes *fully offset* the cis changes such that there are no differences between strains (no DE). (*vertical spread of points in scatter plot)*	yes ($p < 0.1$ and FC > 1.5)	no ($p > 0.1$)	
overcompensating	There are both cis and trans regulatory changes, which act in opposite directions. The trans changes *more than offset* the cis changes such that the difference between strains is in the opposite direction of the cis effect.	yes ($p < 0.1$ and FC > 1.5 OR $p < 0.1$ if there's DE with $p < 0.1$ & FC > 1.5)	yes ($p < 0.1$ if there's ASE; $p < 0.1$ & FC > 1.5 if ASE doesn't meet both thresholds)	ASE and DE LFCs' confidence intervals do not overlap DE/ASE < 0 (log2FC space)

umbrella category: cis-trans opposing	All where it seems there's cis and trans regulatory divergence which oppose each other, may be better to use than sub-categories because relies slightly less on trusting actual estimated LFC numbers. sum of compensating, compensatory, overcompensating.	[see 3 sub-categories]	[see 3 sub-categories]	[see 3 sub-categories]
umbrella category: not conserved	Regulation has diverged (note, even if expression hasn't, i.e. no DE)	[see 6 sub-categories]	[see 6 sub-categories]	[see 6 sub-categories]
ambiguous	ASE and DE estimates uncertainties overlap, but ASE and DE are in opposite directions	yes ($p < 0.1$ and FC > 1.5 OR $p < 0.1$ if there's DE with $p < 0.1$ & FC > 1.5)	yes ($p < 0.1$ if there's ASE; $p < 0.1$ & FC > 1.5 if ASE doesn't meet both thresholds)	ASE and DE in opposite directions, but LFCs' confidence intervals overlap

- Inputs (run script with --help to get)

```
-i, --input       Example filepath to ASE results input - output of
                  ase_de_annotategenes_deseq2_fromemaseout.R (see
                  that script's documentation for format details).
                  **Where strain is identified in filepath, write
                  STRAIN e.g. STRAIN_annotatedASEDEresults.txt.gz
-b, --baseoutname  Base name for all output files [default: out]
-o, --outdir      Outer output directory. Sub-directories will be
                  created internally.
-s, --strains     Strains to read in results for and process
                  together. Either comma-separated (no spaces) list
                  or path to no-header file with one line per
                  strain.  Must match how strains are named in input
                  filenames.
-r, --refstrain   Name of reference strain, matching name used in
                  columns of input. Also used for plot labelling.
                  [default: N2]
-a, --alpha       P-value threshold for considering gene ASE or DE
                  for *adjusted p-value*. Combined with magnitude
```

threshold (--lfcthresh) for first identification,
used on its own if other expression difference
already established. [default: 0.1]

-l, --lfcthresh log2FoldChange threshold for considering gene ASE
or DE. Combined with significance threshold
(--alpha) to establish first expression diff.
[default: 0.5849625]

--asedecis Path to file specifying confidence intervals to
use to determine if ASE and DE magnitudes are
different. Must have one entry for each strain for
each desired threshold; thresholds can be
consistent or different across strains. Columns:
strain (name of strain), CI (confidence interval -
0.95 means 95%, etc), descrip (description of CI
for output files etc. Each strain should have a
row for each description. Order of descriptions in
this file is considered factor level order!)

-g, --genomebp Length of reference genome in bp. Used to get
variants vs. reference per kb. Default is genome
length from NCBI
https://www.ncbi.nlm.nih.gov/assembly/GCF_000002985.6/
- for ws235 but unlikely to have changed much (and
not enough to be meaningful for our purposes)
[default: 100286401]

-v, --varsvsref Path to file containing columns strain (one row
for each strain in main input), nvars (number
variants vs. reference genome)

- Outputs
  - *_regpattern_per1plusunqalnggenestrain_ <Description of confidence interval used to determine ASE, DE overlap - from --asedecis input>confdeaseoverlap.txt.gz: regulatory pattern classification for all genes with 1 or more minimum unique alignments across samples (all others have NA - not wasting disk space). Columns:
    - strain, strain
    - gene_id, gene ID (for matching back in)
    - regclass, assigned regulatory class, following the logic presented in flowchart above. Briefly:
      conserved - no DE or ASE. *Conserved in terms of REGULATION; MORE than this are not DE across strains!*
      ambiguous - ambiguous category, currently when ASE and DE have overlapping CIs but opposite directions of effect
      cis - ASE and DE of similar magnitudes (overlapping CIs)

trans - DE but no ASE
compensatory - ASE but no DE
compensating - ASE and DE; trans in opposite direction of cis (but cis winning out/ASE 'bigger')
overcompensating - ASE and DE; trans in opposite direction of cis so much so that DE is opposite dir from ASE
enhancing - cis and trans effects acting in same dir (DE and ASE unequal, DE/ASE > 1)

- Number summaries
  - *_numbers_regpatterns.txt: number of genes per strain, gene set (informative-ness) *and* confidence interval used to determine whether ASE and DE magnitudes differ classified as each regulatory pattern (see flowchart etc for details). Columns:
    - strain, strain result is for
    - CI, confidence interval used for determining ASE/DE overlap - as in 'descrip' column of --asedecis input
    - geneset, description of gene set (see table for full description)
    - ngenes, # genes in this geneset
    - <one column per regulatory pattern, containing # genes with this regulatory pattern>
    - cis-trans opposing, sum of genes with genes in the 3 regulatory patterns suggesting cis and trans have opposing/compensatory directions: compensatory, compensating, overcompensating
    - not conserved, sum of genes with non-conserved regulatory patterns
  - *_proportion_regpatterns.txt: proportion of gene set each regulatory pattern is. In long format for easy plotting (a bit ugly to sort through). *NB proportions certain classifications are of ASE, DE added later*. Columns:
    - strain, strain result is for
    - CI, confidence interval used for determining ASE/DE overlap - as in 'descrip' column of --asedecis input
    - geneset, description of gene set (see table for full description)
    - ngenes, # genes in this geneset
    - propName, what proportion is in this row? All values are regulatory pattern classifiers *except not conserved and cis-trans opposing, which are sums of other regulatory patterns as described above*
    - denom, (ADDED LATER),
      what is denominator for this row: 'totalngenes' means total genes in input,
      # 'ASEgenes' means genes with signifAtThresholds.ASE==T,
      # 'DEgenes' means genes with signifAtThresholds.ParentVsParent<refstrain>==T
- prop, proportion of genes in this regulatory class
- lowci, binomial 95% confidence interval lower bound on proportion
- highci, binomial upper 95% confidence interval lower bound on proportion
- Bar plots showing breakdown of all regulatory modes
  - *All in directory* stackedbarplots/
  - *_regpatterns_proportions_allsets.pdf - all plots together; rows are gene sets, columns are confidence intervals for ASE/DE overlap [useful for seeing all at once, notebook]

    - *_regpatterns_proportions_<Description of confidence interval used to determine ASE, DE overlap - from --asedecis input>confdeaseoverlap.pdf - separate plots per page; one PDF per confidence interval set [useful for presentations, zooming in]
- Scatter plots of DE vs. ASE with points colored by regulatory pattern classification
    - *All in directory* devsaseplots/
    - *_asevsdeplot_regpatterns_<Description of confidence interval used to determine ASE, DE overlap - from --asedecis input>confdeaseoverlap.pdf - each page is for one gene set, all genes from gene set (no outlining etc). Faceted by strain, strains ordered by divergence from reference.
    - *_asevsdeplot_regpatterns_<Description of confidence interval used to determine ASE, DE overlap - from --asedecis input>confdeaseoverlap_hypdivoutline.pdf - as above, but genes overlapping hyperdivergent haplotypes (as specified in input results file, no new classification done) outlined in bold. Irrelevant for gene sets excluding these genes, of course!
- proportions vs. divergence from reference
    - *All in directory* propvdivergenceplots/
    - *_regpatternsvsdivergence_allsets.pdf - one page per regulatory pattern (or sum of several, e.g. not conserved); gene sets X confidence intervals for ASE/DE overlap grid (so lots of plots per page!)
    - *_regpatternsvsdivergence_Description of confidence interval used to determine ASE, DE overlap - from --asedecis input>confdeaseoverlap.pdf - each PDF has different confidence interval for determining ASE, DE overlap; each page is one gene set. On each page, all regulatory classifications and 'umbrella' classifications (not conserved, all with opposing cis-trans) are facets so can see whole set at once.
    - _regpatternsvsdivergence_<Description of confidence interval used to determine ASE, DE overlap - from --asedecis input>confdeaseoverlap_PropOfASEGenes.pdf: cis and compensatory genes as proportion of genes with cis regulatory differences AKA those called ASE
    - _regpatternsvsdivergence_<Description of confidence interval used to determine ASE, DE overlap - from --asedecis input>confdeaseoverlap_PropOfDEGenes.pdf: cis only and trans only regulated genes as proportion of genes with between-strain differences, i.e. those called DE

Investigate cis-trans negative autocorrelation: data_analysis_scripts/ cistrans_estimates_explorations.R

- *Inputs (also get by running script with --help)*

-r, --resmatinput Example filepath to ASE results input - output of ase_de_annotategenes_deseq2_fromemaseout.R (see that script's documentation for format details). **Where strain is identified in filepath, write STRAIN e.g. STRAIN_annotatedASEDEresults.txt.gz

--regpats Regulatory pattern classifications for all genes, strains for which classification could be made - columns strain, gene_id, regclass. (*_regpattern_per1plusunqalnggenestrain_<conf int>confdeaseoverlap.txt.gz output of ase_de_cistransclassifications.R)

-a, --aseddsf path to R saved object containing dds.ase, the DESeq2 ASE testing object made by ase_de_annotategenes_deseq2_fromemaseout.R

-b, --baseoutname Base name for all output files. Recommend including the informative threshold here if that is ever a thing that could change. [default: out]

-o, --outdir Outer output directory. Sub-directories will be created internally.

-s, --strains Strains to read in results for and process together. Either comma-separated (no spaces) list or path to no-header file with one line per strain. Must match how strains are named in input filenames.

--refstrain Name of reference strain, matching name used in columns of input. Also used for plot labelling. [default: N2]

-i, --informthresh Gene must have this or more unique alignments in each sample to be considered informative for

ASE/cis-trans analyses [default: 5]

- *Outputs*
  - *_cistranscorrelations_inputLFCs.txt - Results for correlating *cis* estimates with *trans* estimates per gene (trans = parental - cis), from input log2FCs. Each strain has one row and the estimate and p values are included for Pearson's (r), Spearman (rho), and Kendall (tau) tests
  - *_cisVtrans_scatters_inputLFCs.pdf - Scatter plots showing cis estimates vs trans estimates for each gene (data underlying the correlations)
  - *_cistransestimates_replicatepairsetc.txt.gz - one row per strain per informative gene (different strain sets per gene). Columns:

    strain, strain row of results is for
    gene_id, gene row of results is for
    regclass, regulatory class from input
    log2FoldChange.ASE, from input
    log2FoldChange.ParentVsParentN2, from input
    transestimate_fromorigLFCs, parental difference minus ASE (two previous LFCs) - the 'trans estimate' that would come from my DESeq2 workflow for doing this
    samp_<1- # samples per strain>_aselfc, log2(ref allele counts/alt allele counts) for each of the samples within the strain. Number 1 is the first sample when all are sorted by name, etc
    samp_<1- # samples per strain>_transParentLFC, trans estimate: parental difference (**global** log2 parent vs parent - from all samples) minus the cis estimate *for the given sample specified.* So, same number minus the numbers in the preceding columns.
- *_cistranscorrelations_allreplicatepairs.txt - Results for correlating *cis* estimates with *trans* estimates per gene (trans = parental - cis), from generating these estimates from every pair of samples (including self pairs, as 'bias' control)
- *_cisVtrans_scatters_replicatepairs.pdf - cis vs trans estimates scatter plots for all strains for all ways of estimating cis and trans: normal way (my DESeq2 way), for all possible combinations of using individual samples to calculate cis (so, same sample used for cis and trans and different samples used for cis and trans estimates)
- *_cistranscorrdotplots_allmethods.pdf - dot plots where rows are strains, columns are different correlation estimates (r, rho, tau). To show differences between DESeq2 method & different among-pair correlation metrics

Regulatory pattern x inheritance mode: data_analysis_scripts/regpatterninhmodeoverlap.R

- Inputs (also run script with --help to see)

-r, --resmatinput Example filepath to ASE results input - output of ase_de_annotategenes_deseq2_fromemaseout.R (see that script's documentation for format details). **Where strain is identified in filepath, write STRAIN e.g. STRAIN_annotatedASEDEresults.txt.gz

--regpats Regulatory pattern classifications for all genes, strains for which classification could be made - columns strain, gene_id, regclass. (*_regpattern_per1plusunqalnggenestrain_<conf int>confdeaseoverlap.txt.gz output of ase_de_cistransclassifications.R)

-i, --inhmode Inheritance mode classifications for all genes and strains. Columns strain, gene_id, inhmode (*_inheritancemode_pergenestrain.txt.gz output from f1_parental_comparisons_withinstrain.R)

--informthresh Gene must have this or more unique alignments (EMASE; or this or more Ornaments counts) in each sample to be considered informative for ASE/cis-trans analyses [default: 2]

-b, --baseoutname Base name for all output files [default: out]

-o, --outdir Outer output directory. Sub-directories will be created internally.

-s, --strains Strains to read in results for and process together. Either comma-separated (no spaces) list or path to no-header file with one line per strain. Must match how strains are named in input filenames. Strains will be plotted/leveled in this order. [default: JU1088,EG4348,CB4856,QX1211]

--refstrain Name of reference strain, matching name used in

columns of resmatinput. Also used for plot

labelling. [default: N2]

- Outputs
  - *Numbers, proportions of genes falling into regulatory pattern-inheritance mode combinations*
    - Each of these has all strains and gene sets in SAME file - would want to narrow to one of each to see a basic matrix of regulatory patterns x inheritance modes
    - *_regxinh_counts.txt, how many genes have each combination of regulatory pattern (rows) and inheritance mode (columns). Columns:
      - strain, which strain is this row's numbers from
      - geneset, which gene set (unique alignments, hypdiv-wise) is this row's numbers from
      - regpat, what regulatory pattern (e.g. cis, trans, compensatory, etc - specified in inputs and script)
      - <one column for each inheritance mode - values show number genes with this inheritance mode, regulatory pattern specified in regpat>
    - *_regxinh_counts_bychr.txt, as above, but broken down by chromosome additionally: added 'chr' column.
    - *_regxinh_props_ofall.txt, formatted as counts, but instead of numbers each value is the proportion of all genes in the described set/strain that have that combination of regulatory pattern & inheritance mode
    - *_regxinh_props_ofall_bychr.txt, as above, but broken down by chromosome additionally: added 'chr' column. Proportion is **within-chromosome** - so all categories in a chromosome add to 1
    - *_regxinh_props_ofeachregpat.txt, formatted as global proportions file, but instead of numbers each value is the proportion of *genes with the specified regulatory pattern* that have that inheritance mode (e.g. proportion of 'cis' genes with each inheritance mode)
    - *_regxinh_props_ofeachinhmode.txt, this has inheritance modes specified in a column (3rd column is 'inhmode' rather than 'regpat'), regulatory patterns as column headers after descriptive columns. Each value is the proportion of *genes with the specified inheritance mode* that have the given regulatory pattern (e.g. portion of alt_dominant genes with each regulatory pattern)
  - *Barplots showing reg pattern - inheritance mode combination breakdowns. Should be reasonably self explanatory from legends, labels*
    - *All are in /barplots subdirectory*
    - *_strainsxgenesets_globalnsps_barplots.pdf: each page has strains (rows) x gene sets (columns) plots. Bars are either inheritance mode or regulatory pattern, subdivided by the other classification. This PDF has global numbers, proportions (proportion of all genes in strain/gene set, rather than within bars); some pages exclude genes classified as both no change and conserved
      - 8 pages: 2 x-axis bar categories x 2 y axes (proportions, numbers) x 2 category sets (including all, excluding those classified as no change & conserved)
    - *_strainsxgenesets_propswithincategs_barplots.pdf: each page has strains (rows) x gene sets (columns) plots. Bars are either inheritance mode or regulatory pattern,

subdivided by the other classification. This PDF has **within-bar** proportions, e.g. what % of 'conserved' genes are each of the inheritance modes for that geneset, strain.
        - 2 pages: 2 x-axis bar categories; proportion done within those
    - _strainbars_globalnsps_barplots_<geneset>.pdf - one per geneset. Each page has STRAINS as bars, colored by either regulatory pattern or inheritance mode and split into little plots by the other categorization (so can, e.g., compare JU1088's distribution of reg patterns within alt_dominant genes to another strain's). This file has global numbers, proportions (proportion of all genes in strain/gene set, rather than within bars)
        - 4 pages: 2 x-axis bar categories x 2 y axes (proportions, numbers)
    - _strainbars_propswithincategs_barplots_<geneset>.pdf - one per geneset. Each page has STRAINS as bars, colored by either regulatory pattern or inheritance mode and split into little plots by the other categorization (so can, e.g., compare JU1088's distribution of reg patterns within alt_dominant genes to another strain's). This PDF has **within-bar** proportions, e.g. what % of 'conserved' genes are each of the inheritance modes for that geneset, strain.
        - 2 pages: 2 x-axis bar categories; proportion done within those
- *Heat plots showing reg pattern-inheritance mode breakdowns: x axis is inheritance mode, y axis is reg pattern; each square is one combination*
    - *All are in /heatplots subdirectory*
    - *These follow Fig 4 Sanchez-Ramirez nigoni x briggsae paper*
    - *_strainsxgenesets_globalns_heatplots.pdf - Color is total N in each category overlap. *Different per strain* - strain with most informative genes will have highest N, brightest color or whatever. Faceted by strain and informative gene set. Pages have all included genes, then *excluding genes classified as *both* no_change and conserved (if exclude either - as next page - also lose, e.g., the compensatory genes that show up as no_change between F1 and parent), then excluding no_change/conserved categorized genes (helps dynamic range)
    - *_strainsxgenesets_globalps_heatplots.pdf - as above, but color is proportion of all genes that each category combination is *within strain* - so strains have same theoretical range here, which isn't true of absolute numbers
    - *_strainsxgenesets_globalns_bychr_heatplots.pdf - Color is total N in each category (diff per strain); faceted by CHROMOSOME and then strain. Pages are different gene sets: informative ness *and* whether no-change/conserved overlap is included or excluded
    - *_strainsxgenesets_globalps_bychr_heatplots.pdf - Color proportion of total analyzed genes in each category (same 0-1 theoretical boundaries in each strain *on each chromosome* - proportion done within chromosome); faceted by CHROMOSOME and then strain. Pages are different gene sets: informative ness *and* whether no-change/conserved overlap is included or excluded

Analyze excess of compensation: data_analysis_scripts/compensationamounttesting.R

- *Inputs (also get by running script with --help)*

-r, --regpatns Counts of regulatory patterns within strain and geneset - *numbers_regpatterns.txt output of ase_de_cistransclassifications.R. **MUST be for only one 'CI' column value**

-b, --baseoutname Base name for all output files; should likely include strain, for example [default: out]

-v, --varsvsref Path to file containing columns strain (one row for each strain in main input), nvars (number variants vs. reference genome) [other columns optional]; for ordering strains etc

-o, --outdir [[output-related]] Output directory. **NB: if you provide getwd() here (quote wrapped), current directory will be used [default: out]

- *Outputs*
  - *_overlappingcistransprops_longformat.txt: Long-form results and proportions of testing the surprisingness of number of genes with both cis and trans effects (including enhancing or just leaving them out). Columns:
    - strain - strain test is in
    - geneset - gene set test is in (informative or informative excluding hypdiv/bad coverage)
    - test, description of what test done here (enhancing included vs not) - same for multiple rows
    - propshown, in proportion columns, which proportions shown (long format!). Options:
      - Overlapping of trans", "Overlapping of cis", "Overlapping of all in analysis", "expected overlap (null) [calculated from prop cis * prop trans]
    - n.trans.and.cis, Number as described - same multiple rows [enhancing included/excluded as specified]
    - n.trans.no.cis, ""
    - n.cis.no.trans, ""
    - n.no.cis.trans, ""
    - or.fet, FET odds ratio for this test - same for multiple rows
    - pvalue.fet, FET odds ratio for this test - same for multiple rows
    - p, proportion as described in 'propshown'

        - low95ci, lower 95% binomial CI bound on proportion estimate (as described in 'prop'shown)
        - high95ci, as above but upper bound
- *_overlappingcistrans_statres.txt: FET p-values from same tests in above file (same column names, doesn't have proportion stuff), but cleaned up so they're not repeated multiply. For ease of examination.
- *_enhancingVsopposing_longformat.txt. Results of testing the proportion genes that are cis-trans opposing of all genes with both *cis* and *trans* effects (opposing + enhancing) vs 50% - showing this is more than enhancing. Columns:
    - strain - strain test is in
    - geneset - gene set test is in (informative or informative excluding hypdiv/bad coverage)
    - test, same description for all - "Cis-trans opposing of all cis and trans (this + enhancing)"
    - p, proportion opposing/(opposing + enhancing)
    - low95ci, binomial 95% bounds on this
    - high95ci,
    - pvalue, p-value from binomial test of this vs. 50%
- *_propOppOfCisTrans_facets.pdf: Plot prop of trans and cis that are opposed: doesn't show how surprising this is but does display data. NB the 'compensated' genes are the same for both columns. With cis-trans as facets, strains on x axis
- *_propOppOfCisTrans_together.pdf: Plot prop of trans and cis that are opposed: doesn't show how surprising this is but does display data. NB the 'compensated' genes are the same for both columns
- *_propExpObsOverlap.pdf: plot proportion genes expected to have both cis and trans vs observed that do
- *_propOppOfCisandTrans.pdf: Plot proportion of genes with cis and trans effects that have opposing effects (probably best in combination with another type?)
- Related to simulating to see how much data would need to be wrong to make these not a big deal
    - *_simulatermcompgenes_recomputecistransoverlap.txt.gz: Results of simulations assuming compensatory genes are 'wrong', moving them to other categories. total number genes held constant always. Take gene away From compensatory genes, add that gene to cis only to simulate DE was missed at that gene; add to conserved to simulate that ASE was spuriously called at that gene . Columns:
        - strain - strain test is in
        - geneset - gene set test is in (informative or informative excluding hypdiv/bad coverage)
        - test, whether numbers were moved to simuate DE missed or ASE spurious
        - percentCompMoved, what % compensatory genes were moved in this simulation
        - nCompMoved, what # compensatory genes were moved in this simulation
        - propWrongDenom, For the propWrong number, what's denominator - are we looking at % DE that were missed or what

    - propWrong, number genes moved here divided by total number of genes of category of interest (see propWrong)
    - or.fet, FET odds ratio for this simulation
    - pvalue.fet, FET p-value for this simulation
- *_simulatermcompgenes_plots.pdf: Plots of results of this simulation - proportion of genes excluded (separate pages for of compensatory, DE missed, ASE spurious) vs. -log10 p value of FET
- *_propCisTransAtNotCisTrans.pdf - Compare prop of genes with cis effects OVERALL vs. at genes with trans influences [opposing & all]

Analyze surprisingness of proportion of compensated genes: data_analysis_scripts/ transpropcompensationtesting.R

- *Inputs (also get by running script with --help)*

```
-r, --regpatns     Counts of regulatory patterns within strain and
                   geneset - *numbers_regpatterns.txt output of
                   ase_de_cistransclassifications.R.  **MUST be for
                   only one 'CI' column value**
-b, --baseoutname  Base name for all output files; should likely
                   include strain, for example [default: out]
-v, --varsvsref    Path to file containing columns strain (one row
                   for each strain in main input), nvars (number
                   variants vs. reference genome) [other columns
                   optional]; for ordering strains etc
-o, --outdir       [[output-related]] Output directory. **NB: if you
                   provide getwd() here (quote wrapped), current
                   directory will be used [default: out]
```

- *Outputs*
  - *_transprops_statres_ofNotASEandASE.txt - key/only data. Numbers and proportions of the main comparisons: trans genes without ASE of all genes without ASE; trans-reg (compensatory) genes of ASE of all genes with ASE. Specifically, columns:
    - strain
    - geneset - from input; here, 5plusUnqAlns (informative) and 5plusUnqAlns_exclhypdivbadcov (informative excluding hypdiv & bad coverage)
    - # n.trans.nonase, # trans-only regulated genes, obligatorily don't have ASE
    - # n.non.ase, # genes that don't have ASE but simplified to ones we feel strongly about (trans regulated or not): conserved + trans-labeled genes
    - # n.cistransopposing, # cis-trans opposing (umbrella category is already in input)
    - # n.ase, # with any cis effect (those with ASE simplified to ones we feel strongly about, trans regulated or not): cis + enhancing + cis-trans opposing
    - # p.trans.ofnonase, n.trans.nonase/n.non.ase
    - # p.trans.ofnonase_low95, lower binomial 95% CI
    - # p.trans.ofnonase_hi95, upper binomial 95% CI
    - # p.comp.ofase, n.cistransopposing/n.ase
    - # p.comp.ofase_low95, lower binomial 95% CI
    - # p.com.pofase_hi95, upper binomial 95% CI
    - # or.fet, odds ratio of Fisher's Exact Test comparing these proportions
    - # pvalue.fet, p-value (two-sided) of Fisher's Exact Test comparing these proportions

- *_transprops_statres_longformat_ofNotASEandASE.txt - above data in LONG format
- *_transprops_ofNotASEandASE_barplot.pdf: barplot showing proportion genes with, without trans effects/compensation of non-ASE and ASE genes
- *_transprops_ofNotASEandASE_dotplot.pdf - same data as above, but proportion shown as a point and all strains plotted together, with lines connecting same strain's points. *Not quite right*
- Related to removing ASE genes to see how many I have to remove for the compensated proportion to not be >> than DE of non-ASE
    - all in subdirectory reduceaseretes/
    - *_downase_data.txt: data underlying this; shows number, proportion ASE genes 'removed' and results of re-FET-ing them
    - *_downase_premovedVsFETpval.pdf: line plots showing proportion ASE calls removed vs. FET p-value, plus point shape shows whether pattern is same as or changed from original (wherein there is way higher proportion compensated than of non-ASE genes with *trans* effect)
        - different pages have different facets
- *demissedInASE_ofNDE.txt - quick calculations of what proportion of DE calls those 'missed' if ASE compensation is 'false' would be. Columns:

    strain
    geneset
    DECalled, # genes with DE called - generous interpretation: cis+enhancing+trans + compensating + overcompensating
    DEMissed, # genes where DE would be missed if ASE is real and compensation is not (# genes classified 'compensatory')
    p.ofcalled, DEMissed / DECalled
    p.ofcalledPlusMissed, DEMissed/(DEMissed + DECalled)

WormCat GSEA with custom background gene set: data_analysis_scripts/wormcat_givebackgroundset.R

- *Inputs (run script with –help to get_*

```
-t, --targetlist      Gene list to test for enrichment (e.g. your
                      upregulated genes) (wormbase IDs). Column title
                      should be 'Wormbase.ID'
-b, --backgroundlist  Background gene list against which to test
                      enrichment (e.g. expressed genes) (wormbase
                      IDs). Column title should be 'Wormbase.ID'
-a, --annots          Genome-wide annotations from Wormcat, usually
                      titled whole_genome_v2_nov-11-2021.csv
-o, --outdir          Output directory path (relative or absolute);
                      should describe this specific comparison if
                      running multiple. [default: wormcatout]
--titleplot           Title for Wormcat-generated plots [default: ]
-w, --wormcatcodedir  Path to directory containing WormCat five
                      helper function R scripts (from
                      https://github.com/dphiggs01/Wormcat/tree/master/R)
                      [default: Wormcat/R]
```

- *Outputs*
  - Makes the wormcat outputs for your input lists! .csvs, .svgs, etc - see wormcat website for descriptions (www.wormcat.com)

WormCat GSEA – analyze multiple runs: data_analysis_scripts/combinewormcatout_aseetc.R

- Inputs (available via running script with --help)

```
-n, --nftorun          Path describing comparisons that were run
                       through wormcat (with nextflow workflow):
                       --torun parameter.  Key column is testname
                       which describes output directory names in
                       **ASSUMED** format <tested
                       genes>_vs_<background
                       genes><_exclhypdivbadcov if those
                       excluded>_<STRAIN>
-w, --wormcatparentdir  Directory containing all wormcat output
                       directories to process together
-b, --baseoutname      Base name for all output files [default: out]
-o, --outdir           Output directory path [default: out]
-s, --strains          Strains to summarize across (i.e., all
                       processed strains excluding any tests done in
                       metacategory.  So number of strains in which
                       comparison is significant will be of these.)
                       [default: JU1088,EG4348,CB4856,QX1211]
```

- Outputs
  - *Meta info columns common across outputs - these were hardcoded:*
    testname, long name that was original wormcat output directory name
    Strain, worm strain
    testset, tested genes descriptor
    backgroundset, background geneset descriptor
    exclhypdivbadcov, T or F, were hypdiv bad cov genes excluded (as flagged in file name)

- *_combinedwormcatapv_<cat1-3>.txt: All wormcat's significant categories (from rgs_fisher_cat#_apv.csv wormcat output) combined for all gene set tests, plus some extra info. Each gene set test has one row per enriched category - if no enriched categories, one row populated with NAs. Each category level (1-3) has its own file, exactly like wormcat. Columns:
  - *columns containing meta info on test: as above*
  - *columns directly from wormcat:*
    - Category, enriched category name (from wormcat)
    - RGS, number of genes in test set that were in this category (*pretty sure,* from wormcat)
    - AC, number of genes in background set that were in this category (*pretty sure,* from wormcat)
    - PValue, Fisher's test p-value for this category (from wormcat)
    - Bonferroni, Bonferroni-corrected p-value for this category, what was used as inclusion criteria (from wormcat)

        - *Additional columns for interpretation:*
            - nAnnotGenesTestset, # genes in the test set total that were annotated and used in wormcat for this level of category test (sum across all categories, pulled from all category wormcat file)
            - nAnnotGenesBackground, # genes in the background set total that were annotated and used in wormcat for this level of category test (sum across all categories, pulled from all category wormcat file)
            - pTestInCategofAllTest, proportion of test set genes that were in this category (RGS/nAnnotGenesTestset)
            - pBackgroundInCategofAllBg, proportion of background set genes that were in this category (AC/nAnnotGenesBackground)
            - pTestInCategofCateg, proportion of genes in category that were in the test set (RGS/AC)
            - foldEnrichmentTestinCateg, fold enrichment of category in test set vs. background set (pTestInCategofAllTest/pBackgroundInCategofAllBg)
- *_nwormcatsigcats_<cat1-3>.txt: Number of categories wormcat called significant for each test (test set, background set, STRAINI). One row for each set of genes tested. Each category level (1-3) has its own file, exactly like wormcat. Columns:
    - *columns containing meta info on test*
    - nEnrichedCats, number of categories enriched in this strain for this test
- *_percatnstrainsenriched_<cat1-3>.txt: Number of strains with significantly enriched categories for each gene set tested (uses input strain list - any tests done with combinations across strains pre-excluded). (this does NOT say whether categories were the same or not - just count of strains that had any significantly enriched category for this test). One row for each set of genes tested combined across strains. Each category level (1-3) has its own file, exactly like wormcat. Columns:
    - *columns containing meta info on test, excluding strain & long format test name*
    - nEnrichedCats, number of categories enriched in this strain for this test
- *Dot plots*
    - For all, any dot for any enrichment category means that category was included in wormcat's significant outputs (significant at Bonferroni). (with obvious exception of if NA category included as it is on some plots)
    - *_dotplotstrainxaxis_<excl or incl for hypdiv genes>.pdf: 6 page PDF, one page per wormcat category level. First set of plot includes all tests - even if no enrichments detected - and second includes only tests where enrichments were detected. Here, strains x enriched category is plotted; each different test has its own facet (easy to compare strains within test)
    - *_dotplottestxaxis_<excl or incl for hypdiv genes>.pdf: 6 page PDF, one page per wormcat category level. First set of plot includes all tests - even if no enrichments detected - and second includes only tests where enrichments were detected. Here, tests x enriched category is plotted; each different strain has its own facet (easy to compare tests within strain)

Chromosome location vs various ASE-related expression phenotypes: data_analysis/scripts/ chrlocenrichment_asederpim.R

- Notes
  - Uses chromosome arm vs. center designations from Rockman 2009, filled in to be complete
  - genes are assigned to the region in which their midpoint falls (so, possibly imperfect, but unlikely to make much of a difference)
  - For density plots, each category has a fake gene added at the beginning and end of each chromosome (to get span right) - it is plotted/taken into account in density
- Inputs (run script with –help to get these printed out)

-r, --resmatinput Example filepath to ASE results input - output of ase_de_annotategenes_deseq2_fromemaseout.R (see that script's documentation for format details). **Where strain is identified in filepath, write STRAIN e.g. STRAIN_annotatedASEDEresults.txt.gz

--regpats Regulatory pattern classifications for all genes, strains for which classification could be made - columns strain, gene_id, regclass. (*_regpattern_per1plusunqalnggenestrain_<conf int>confdeaseoverlap.txt.gz output of ase_de_cistransclassifications.R)

-i, --inhmode Inheritance mode classifications for all genes and strains. Columns strain, gene_id, inhmode (*_inheritancemode_pergenestrain.txt.gz output from f1_parental_comparisons_withinstrain.R)

-b, --baseoutname Base name for all output files [default: out]

-o, --outdir Outer output directory. Sub-directories will be created internally.

-s, --strains Strains to read in results for and process together. Either comma-separated (no spaces) list or path to no-header file with one line per strain. Must match how strains are named in input filenames. Strains will be plotted/leveled in this

order. [default: JU1088,EG4348,CB4856,QX1211]

--refstrain Name of reference strain, matching name used in columns of resmatinput. Also used for plot labelling. [default: N2]

-c, --chrfile File with columns chr, start, end. Used for plotting coordinates

--chrregions File containing chromosome tip, arm, center start/end positions; columns chr, domain [tip, arm, center], subdomain [left or right where relevant], start [bp position, included], end [bp position, included]

--informthresh Gene must have this or more unique alignments in each sample to be considered informative for ASE/cis-trans analyses [default: 2]

-g, --genomebp Length of reference genome in bp. Used to get variants vs. reference per kb. Default is genome length from NCBI https://www.ncbi.nlm.nih.gov/assembly/GCF_000002985.6/ - for ws235 but unlikely to have changed much (and not enough to be meaningful for our purposes) [default: 100286401]

-v, --varsvsref Path to file containing columns strain (one row for each strain in main input), nvars (number variants vs. reference genome)

- Outputs
  - *_<how are genes split - aseVnot, deVnot, regpattern, or inhmode>_genedensityplot.pdf - density plots of the bp location of genes (from gene midpoint). Each density line is one class of genes (e.g. ASE or not ASE). Tips, centers, arms of chromosomes highlighted with contrasting gray/white stripes.
    - At least 2 pages per PDF - including and excluding hyperdivergent/low-coverage genes
    - DEvNot and inhmode also have plots for both all genes (***including background genes that aren't expressed very much, presumably)*** and ASE informative genes (more useful for comparing to ASE, of course). (ASE vs not and regulatory pattern are obligately restricted to ASE informative genes)

- *_chrdomainVmulticategorychisqresults.txt - ChiSq test results for different gene categories loading into chromosome domains (arm, center, tip). Each category has a row - long row-wise data. Details:
  -

column name	notes	possible values (where relevant/reasonable to include)
strain		
testedCategory	Which category is this row for - ASE vs not, etc	ASE (informative genes), DE (all genes), DE (ASE informative genes), Regulatory pattern (ASE informative genes), Inheritance mode (all genes), Inheritance mode (ASE informative genes)
hypdivgenes	whether hypdiv (and low coverage genes) are excluded or included for this row's results	inclhypdiv, nonhypdiv
category	Value of tested category this row gives information for	T or F for ASE, DE each regulatory pattern for reg patterns each inheritance mode for inheritance modes
arm	# genes in this category on chromosome arms genome wide	
center	# genes in this category on chromosome	

	centers genome wide	
tip	# genes in this category on chromosome tips genome wide	
arm_pOfTotal	proportion of TOTAL gene observations for genes included in analysis in arms - same for each row of category	
center_pOfTotal	proportion of TOTAL gene observations for genes included in analysis in chr centers - same for each row of category	
tip_pOfTotal	proportion of TOTAL gene observations for genes included in analysis in chr tips - same for each row of category	
arm_pOfThisRowCategory	proportion of this row category (e.g., ASE genes) in chr arms	
center_pOfThisRowCategory	proportion of this row category (e.g.,	

	ASE genes) in chr center	
tip_pOfThisRowCategory	proportion of this row category (e.g., ASE genes) in chr tips	
ChiSq	ChiSq test statistic for chisq test of this category - - same for each row of category	
Df	Degrees of freedom for chisq test of this category - same for each row of category	
pvalue	p-value for chisq test of this category - same for each row of category	

- *_geneswithdomain.txt.gz: Genes annotated with their chromosomal domain information, for ease of downstream use. Columns:
  gene_id, from GTF (was already in input)
  display_name, from GTF (was already in input)
  chr, from GTF (was already in input)
  start, from GTF (was already in input)
  end, from GTF (was already in input)
  genepos, gene midpoint - this is what is used to overlap domain
  domain, arm center or tip
  subdomain, left or right for arm/tip, empty string for center
- *Barplots & underlying data*
  - In sub-directory /barplots
  - *_longformproportions_chrdomainXmulticategories.txt: Number of genes in each combination of domain and other category of interest (ASE, reg pattern, etc), and proportion this is overall and of domain or other category value. Columns:
    - testedCategory, nice format description of which category tested against domain here

        - hypdivgenes, included or not information for hyperdivergent etc genes
        - categ1name, catrowcol input - what is name of the category 1 column
        - category1, Actual category of categ1 that this row describes (e.g. TRUE for ASE)
        - categ2name, catcol input - what is name of the category 2 column
        - category2, Actual category of categ2 that this row describes (e.g. center for domain)
        - total.n, total # genes - input number
        - total.thiscateg1, total # genes with categ1name (e.g., TRUE for ASE)
        - total.thiscateg2, total # genes with categ2name at category2 (e.g., center for domain)
        - n.thiscombo, number of genes that have categ1name at category 1 and categ2name at category 2 (e.g., TRUE for ASE and in center domain)
        - prop.<total, thiscateg1, thiscateg2>, proportion this number comprises: of all genes, of genes with category 1 having categ1 name (e.g., of ASE = TRUE genes),
        - of genes with category 2 having categ2 name (e.g., of domain = center genes)
        - low95.<total, thiscateg1, thiscateg2>, lower binomial 95% confidence interval bound for the proportion specified
        - high95.<total, thiscateg1, thiscateg2>, upper binomial 95% confidence interval bound for the proportion specified
    - *_domainVs <ase, de, reg patterns, etc>_barplots_domainx.pdf - Domain on X axis, bar split by other category. Many pages per PDF, one PDF per other category domain is tested against. Each page is for one gene set (e.g. including hypdiv), metric plotted (number, global proportion, proportion within bar)
    - *_domainVs <ase, de, reg patterns, etc>_barplots.pdf - as above, but non-domain category is on x axis and bars are split into 3 (arm, center, tip domain)
- *Various proportions plotted vs. divergence from N2*
    - *_propindomainvsdivergencefromN2_<what is plotted vs proportion, in domain>.pdf: one page per gene set. Plotted is the proportion of genes in <domain> that are <category, e.g. ASE T or F, or each individual regulatory pattern> - domains are facets, so left column is the proportion of arm genes that are <category, e.g. ASE true>

GENERATING NUCDIV STATS WITH POPGENOME: data_generation_scripts/nucdivcendr_geneswindows_allandasestrains.R

- Inputs (get by running with --help)

```
-v, --vcf       Path to VCF to process. Must have matching tabix
                index in same directory.  Note: ***popgenome/called
                packages for VCF processing do NOT do well with the ~
                shortcut in filenames - need to write it all out!
-g, --gtf       Path to *unzipped* GTF matching input VCF.
-o, --outdir    Output directory. Currently tmp files written here
                too. [default: out]
--out           Output file prefix. Data, plots will be written to
                this _<descriptive suffixes> [default: out]
-p, --pops      File containing sub-populations for which to compute
                statistics. **ALL samples will be added to a
                population called 'all' automatically.  Format: 2
                columns titled PopName and ID. PopName contains the
                name for the population (same for all IDs for that
                population), ID the sample ID matching VCF. As many
                rows per population as samples in that population.
-c, --chrfile   File with columns chr, start, end. One line per
                chromosome to process; start and end should match
                GTF/GFF for these data
-r, --reffa     Example filepath to reference genome fasta for one
                chromosome matching GTF, VCF build! Chromosome in
                name must be as provided in --chrfile.  Where
                chromosome ID is in filename, replace with -CHR-,
                e.g. c_elegans.PRJNA13758.WS276.genomic.-CHR-.fa.
                NOTE: need to be unzipped for PopGenome!
-w, --windowbp  Size of window to bin genome into for sliding window
                analyses. Non-overlapping windows of this size will
                be considered. [default: 10000]
```

- Outputs - current
  - *_nucdivpergene.txt.gz: gene-level statistics in *long format:* One row per population (e.g. all strains, just RNA seq strains, etc) - site class (all sites in region; sites in coding regions [CDS] only, synonymous sites only, nonsynonymous variants only) pair. Columns:

Column name	details
chr	Chromosome
gene_id	WB gene ID
pos.start	Start position of gene
pos.end	End position of gene
pos.mid	mid position of gene
genelength	gene length
cdslength	coding (CDS) length of gene (summed across all CDS exons, with overlapping exons merged - done in popgenome) for protein-coding genes
population	Which samples were included for the calculation/summary with results in this row. 'all' means all samples; other names are as provided in --pops input to script
sites	Which sites were included for the calculation/summary with results in this row. Options: all - all sites in the region. ***Included for all genes.*** coding - SNPs in the region that are in coding regions, i.e. in CDS. ***Only included for coding genes.*** syn - SNPs in the region that result in synonymous changes. ***Only included for coding genes.*** nonsyn - SNPs in the region that result in nonsynonymous changes. ***Only included for coding genes.***
nSegregatingSites	number of polymorphic/segregating sites in region in this population, site class
pSegSites	proportion of total *appropriately matched* sites in the region that are segregating: for 'all' sites, this is gene length; for coding, synonymous, and nonysynonymous, this is CDS length
pi_raw	raw Nei's pi (not per site)
pi_persite	Raw pi divided by *total appropriately matched* sites. for 'all' sites, this is gene length; for coding,

	synonymous, and nonysynonymous, this is CDS length
tajimasD	Tajima's D for this class of sites in this population

- *_nucdivper<size of window in bp as provided in --windowbp>window.txt.gz
  - As with per gene (described above), but for equally sized non-overlapping windows instead. Omits 'gene_id' and 'genelength', 'cdslength', 'sites' columns (*only done for ALL sites in window)*
- Plots: genomic position vs. nucleotide diversity statistics
  - All include loess-smoothed trend/mean line.
  - *_nucdivplots_<whether per-gene or per-window stats are displayed>_stackedpops.pdf: each page is one statistic (column of main output) for one set of sites (all, coding, synonymous, nonsynonymous for pergene). Populations are rows. So, you can see the same statistic's value across whatever populations you've used
    - Lots of pages (stats * subsites) for per-gene; only one page per stat for nucdiv plots
  - *_nucdivplots_pergene_stackedsubsites.pdf: each page is one statistic (column of main output) for one population. Subsite sets (all, coding, synonymous, nonsynonymous) are rows. So, you can see the same statistic across different types of sites.
    - LOTS of pages if many populations (stats * populations)

Arbitrary gene-level categories/values/phenotypes vs. ASE-related expression classifications: data_analysis_scripts/aseetc_vs_general.R

- Notes
  - max amount of genes processed is those in first strain's ASE input: if no gene narrowing is specified, this is the 'kept' list specified
  - NEW is cis change compensated or not test:
    - genes that have a cis change are included
    - not compensated = called cis or called enhancing
    - compensated = cis trans opposing (compensating, compensatory, overcompensating)
  - everything is done for 'all' meta strain as well as each strain individually. 'all' includes *all genes present in each strain;* if informative, has to be called informative in all!
    - update 5/9/24 : for all strain, gene must be not informative in ALL strains to be not informative in 'all' metastrain (else, it's NA)
  - For categorical, some (binomial) modeling is done that only works for T/F (binomial) 'test' data - NOT RUN FOR MULTIPLE CATEGORIES as outcomes (probably would need to do this multinomial…)
- Inputs (also get by running with --help)

-r, --resmatinput [[ASE-related data input]] Example filepath to ASE results input - output of ase_de_annotategenes_deseq2_fromemaseout.R (see that script's documentation for format details). **Where strain is identified in filepath, write STRAIN e.g. STRAIN_annotatedASEDEresults.txt.gz

--regpats [[ASE-related data input]] Regulatory pattern classifications for all genes, strains for which classification could be made - columns strain, gene_id, regclass. (*_regpattern_per1plusunqalnggenestrain_<conf int>confdeaseoverlap.txt.gz output of ase_de_cistransclassifications.R)

-i, --inhmode [[ASE-related data input]] Inheritance mode classifications for all genes and strains. Columns strain, gene_id, inhmode (*_inheritancemode_pergenestrain.txt.gz output

from f1_parental_comparisons_withinstrain.R)

-a, --aseinfo [[ASE-related data input]] File containing information on ASE strains to use to generate ASE-relevant outputs. One row per WILD strain crossed with reference. Columns: isotype, strain [wild strain - as in resmatinput naming], refparent [ref strain], nvars [number of variants differentiating wild strain and reference strain]. Wild strains ordered how you want to plot them!!

--informthresh [[ASE-related data input]] Gene must have this or more unique alignments in each sample to be considered informative for ASE/cis-trans analyses [default: 5]

-g, --genelist [[gene-related data input]] OPTIONAL Filepath to list of genes to narrow to if desired [if omitted, all genes in --resmatinput will be used]. Can be no-header list of gene_ids (same genes retained for all strains) OR two-column file with columns strain, gene_id (genes will be retained in strain-specific manner) [default:]

--genesdescrip [[gene-related data input]] OPTIONAL description of the genes in genelist to pass through to outputs, plots (descriptive). Default is 'all genes' if no gene list provided [default: all genes]

-t, --totest [[gene-related data input]] Path to file containing the data to test against ASE etc! Must have columns gene_id, --testcolumn value; if it has column strain, data will be matched in strain-specific manner; if not, data will be assigned multiply to each strain

--testcolumn [[gene-related data input]] Column name in --totest file that contains the data of interest

to test ASE etc against. E.g., Pi if you're
looking at nucleotide diversity via a column
tilted Pi in --totest file

--testdatatype [[gene-related data input]] Type of data in totest file, --testcolumn value: quantitative or categorical (binary T/F!). Acceptable values: [quantitative, categorical] [default: quantitative]

--testdatalabel [[gene-related data input]] Description of test data to be used for plot labels and the like (presumably you should also have some sort of nod to this in output names/directories)

-o, --outdir [[output-related]] Output directory **NB: if you provide getwd() here (quote wrapped), current directory will be used[default: out]

-b, --baseoutname [[output-related]] Base name for all output files [default: out]

--genomebp [[misc/plotting related]] Length of reference genome in bp. Used to get variants vs. reference per kb. Default is genome length from NCBI https://www.ncbi.nlm.nih.gov/assembly/GCF_000002985.6/ - for ws235 but unlikely to have changed much (and not enough to be meaningful for our purposes) [default: 100286401]

- Outputs
  - If data is continuous/quantitative:
    - *Two way tests/data*
      - *_twowaytests_mannwhitneyresults.txt: Mann-Whitney/Wilcox test results, and underlying values and ns, for each strain for each two-way test (ASE informative vs uninformative; ASE vs not; DE vs not; DE vs not (ASE informative genes); Cis-regulatory change: compensated in trans or not). Columns:

        strain, strain this rows test results are for
        test, test results in this row
        genes, genes descriptor (from input)
        med.T, median value of genes TRUE in this comparison
        med.F, median value (of input test data) of genes F in this comparison

T.greater, T genes median greater?
ngenes.T, number genes with T values for this test (nas removed)
ngenes.F, number genes with F values for this test (nas removed)
W, Mann-Whitney/Wilcox test statistic
wilcox.p.value, Mann-Whitney/Wilcox p value

- *_twowaytests_sinaplots.pdf: ALL sina plots for T/F tests (as above: ASE vs not, etc). Each page is one test.
- *Multi-category tests/data*
  - *_sinaplots_<Name of category in this sina plot: regpatall, regpatsimple, inhmode, inhmodeinformonly>.pdf: One PDF file per classification with more than two categories - i.e., regulatory pattern and inheritance mode. Category (e.g. regulatory pattern) is on X axis, test data is on y axis. 4 pages per PDF/classification: ANOVA p-value annotated; ANOVA p-value annotated and data on log10 scale; Tukey contrasts *signif different from base factor* labeled; Tukey contrasts *signif different from base factor* labeled on log10 scale
    - NOTE log10 done automatically - may not be appropriate; definitely gets rid of 0s!
  - *_multiwaytests_nspercategory.txt: record of number of genes in each category (with non-NA phenotypes). Columns:
    - strain - strain this row has gene # for
    - category - overall category tested, e.g. regulatory pattern
    - genes - geneset included description (from input; informativeness done based on what makes sense here/internally)
    - categorylabel - e.g., cis, trans, etc for regulatory pattern
    - n - number of genes NON-NA for test data in this category
  - *_multiwaytests_ANOVAresults.txt: ANOVA test results for all multi-way tests (Regulatory pattern (all), Regulatory pattern (simplified), Inheritance mode, Inheritance mode (ASE informative genes)). One row per category/strain tested. Columns:

strain
category: what category is result for (e.g., regulatory pattern)
genes: geneset included description (from input; informativeness done based on what makes sense here/internally)
Df_resid, DF residuals from ANOVA out
SumSq_resid, sum squares residuals from ANOVA out
MeanSq_resid, mean squares residuals from ANOVA out
Df_category, DF for test category from ANOVA out
SumSq_category, sum squares for test category from ANOVA out
MeanSq_category, mean squares for test category from ANOVA out
Fvalue, ANOVA F value

pvalue, ANOVA p value

- *_multiwaytests_ANOVAresults_significantonly.txt: as above, but ONLY for tests where ANOVA p < 0.05 (for quick glancing; obviously these data are contained in above file)
- *_multiwaytests_TukeyHSDresults.txt:. Tukey's HSD results for ALL comparisons for ALL categories (one row per 2 way category comparison per classification tested *per strain*)….columns:

  strain
  category: what category is result for (e.g., regulatory pattern)
  genes: geneset included description (from input; informativeness done based on what makes sense here/internally)
  comparison, cat1-cat2 detail of comparison : default way tukey output displays [so can make sure this is represented by next columns - shouldn't be needed, had a bug in here initially so I'm cautions]
  # cat1, category this test is in (vs cat 2)
  # cat2, category this test is in (vs cat1)
  # diff, Tukey output
  # lwr, Tukey output
  # upr, Tukey output
  # tukey.padj, Tukey output (adjusted p)
- *_multiwaytests_TukeyHSDresults_significantonly.txt: as above, but ONLY for tests where ANOVA p < 0.05 *and* Tukey adjusted p < 0.05 for this comparison for quick glancing; obviously these data are contained in above file)
- *_multiwaytests_TukeyHSDresults_countsigsummary.txt: for each among-category comparison, how many strains (*including 'all' meta strain*) had significant Tukey comparison. Columns:

  category, what is classification being tested, e.g. regulatory pattern
  genes, description of gene set (provided in input)
  cat1, for this row, which categories compared (cat1 vs cat2)
  cat2, for this row, which categories compared (cat1 vs cat2)
  whichgreateravg, which category greater in magnitude (average across strains)
  nStrainsSigTuk, # strains *including meta strain all* with significant Tukey contrasts between these categories
  whichstrains, strains *including meta strain all* that have significant Tukey contrasts at these categories

- If data is categorical
  - *all ns are for total non-NA observations; Nas fully excluded*
  - *_alltests_longformproportions.txt: Number of genes in each combination of test data and other category of interest (ASE, reg pattern, etc), and proportion this is overall and of test data value or other category value. Columns:
    - strain, strain this result/n is for
      - *NB, includes 'all' metastrain which counts each genes multiple times*
    - testedCategory, nice format description of which category tested against test data here

        - genes, gene set description from input
        - categ1name, name of category 1 (e.g., regclass, informative gene, etc)
        - category1, Actual category of categ1 that this row describes (e.g. TRUE for ASE)
        - categ2name, what is name of the category 2 column (test data column - input value)
        - category2, Actual category of categ2 that this row describes (e.g. TRUE for essential gene T/F)
        - total.n, total # **genes not NA in either cat1 or cat2 here** - input number
        - total.thiscateg1, total # genes with categ1name (e.g., TRUE for ASE)
        - total.thiscateg2, total # genes with categ2name at category2 (e.g., center for domain)
        - n.thiscombo, number of genes that have categ1name at category 1 and categ2name at category 2 (e.g., TRUE for ASE and in center domain)
        - prop.<total, thiscateg1, thiscateg2>, proportion this number comprises: of all genes, of genes with category 1 having categ1 name (e.g., of ASE = TRUE genes),
        - of genes with category 2 having categ2 name (e.g., of domain = center genes)
        - low95.<total, thiscateg1, thiscateg2>, lower binomial 95% confidence interval bound for the proportion specified
        - high95.<total, thiscateg1, thiscateg2>, upper binomial 95% confidence interval bound for the proportion specified
    - *_alltests_chisqtestresults.txt: Chi-square test results for each strain for test data vs. <other classification of interest>. One row per classification - doesn't give information on breakdown amongst categories (see next output for that). Columns:
        strain
        testedCategory -Which category is this row for - ASE vs not, etc
        genes - gene set description (given in input)
        ChiSq - test statistic
        Df - degrees of freedom this test
        pvalue - Chisq test p-value (*uncorrected for multiple hypothesis tests*)
- *_alltests_chisqtestresults_percategory.txt: Chi-square results for each strain x category test vs categorical testdata; one row per strain x category (e.g. regulatory pattern) x specific category (e.g. 'cis' regulated genes). Columns:
    - *NB, chisq test removes any all-0 category so can do test for rest; if a category (e.g. ambiguous genes) is missing, it was all 0s*
    -

column name	notes	possible values (where relevant/reasonable to include)
strain		

testedCategory	Which category is this row for - ASE vs not, etc	ASE informative vs uninformative, ASE vs not, DE vs not, DE vs not (ASE informative genes), Cis-regulatory change: compensated in trans or not, Regulatory pattern (all), Regulatory pattern (simplified), Inheritance mode, Inheritance mode (ASE informative genes)
genes	gene set description (given in input)	--genesdescrip input
category	Value of tested category this row gives information for	T or F for ASE, DE each regulatory pattern for reg patterns each inheritance mode for inheritance modes *Categories omitteed if they had no observations in input data*
<test data column>.<test data value 1>	# genes in this category with test data of the provided test data category	
<test data column>.<test data value..n>	""	
<test data column>.<test data value 1>_pOfTotal	proportion of TOTAL gene observations for genes included in this test data category - same for each row of category	
< same as above for all testData categories)	""	

<test data column>.<test data value 1>_pOfThisRowCategory	proportion of this row category (e.g., ASE genes) in this test data category (e.g., essential gene TRUE)	
< same as above for all testData categories)	""	
ChiSq	ChiSq test statistic for chisq test of this category -- same for each row of category	
Df	Degrees of freedom for chisq test of this category - same for each row of category	
pvalue	p-value for chisq test of this category - same for each row of category	

- IF categorical data is T/F, binomial models run for all multi-way tests and betas extracted for each category vs. all reference categories of interest:
- Barplots - lots of varieties - in sub-directory barplots/
  - *_barplots_<classification plotted against test data>_testdatax.pdf: 3 barplots where test data categories are on x axis *for each classification tested, e.g. ASE vs not; regulatory pattern*
    - page 1: each bar shows the proportion *of that x axis category* with the shaded value. E.g., proportion of TRUE test data that are each regulatory pattern. Annotated with chi-sq test p-value. (each bar sums to 1)
    - page 2: Each bar shows proportion of *all analyzed genes* that are in the bar, with the given classification (all bars together sum to 1)
    - page 3: each bar shows number (of analyzed genes) in that bar, shaded by classification (*all bars sum to total analyzed genes)*
  - *_barplots_<classification plotted against test data>_<classification plotted against test data>x.pdf: 3 barplots where non-test data classification categories are on x axis *for each classification tested, e.g. ASE vs not; regulatory pattern (e.g., cis, trans, etc on x axis).*
    - page 1: each bar shows the proportion *of that x axis category* with the shaded value. E.g., proportion of cis-regulated genes that are each test data value. Annotated with chi-sq test p-value. (each bar sums to 1)
    - page 2: Each bar shows proportion of *all analyzed genes* that are in the bar, with the given classification (all bars together sum to 1)
    - page 3: each bar shows number (of analyzed genes) in that bar, shaded by classification (*all bars sum to total analyzed genes)*

- Proportions vs. divergence from N2 (probably way more plots than are useful) - in subdirectory divergence/
    - *_propintestvsdivergencefromN2_<classification plotted against test data>.pdf - **these are confusing.** One for each flavor of ASE data tested. For both pages, column facets are values in test data and row facets are values of other classification (e.g., ASE vs not, regulatory pattern, etc). Points plotted on first page are *proportion of testdat value genes (e.g., essential = F genes) that have the given other classification value (e.g., conserved - so proportion of essential genes that are conserved).* Points plotted on second page are *proportion of other classification* (e.g., conserved genes) *that have the given testdat value (e.g., non-essential genes with expression conservation.).* Goal is to see if this changes or not as get more diverged from N2

Estimate transcriptomic age with RAPToR: data_classification_scripts/RAPToR.R

- Interactive script. Run and choose input DESeq2 objects for age analysis.

Analyze Wolf et al. 2023 human gene expression level, variability data: data_analysis_scripts/ wolf2023humexpanalyses.R

- Inputs (get by running with --help)

-w, --wolfsupp4 Filepath to S4 Data from Wolf et al 2023 (https://doi.org/10.1371/journal.pgen.1010833.s013), across study rank xlsx data tab as tab-delimited txt file

-o, --outdir Output directory [default: out]

-b, --baseoutname Base name for all output files [default: humangeneexp_wolf2023]

- Outputs
  - Quantitative/data exactly as input-related
    - *_cortestresults_meanrankvssdrank.txt: correlation test results for testing mean vs variance (ranks from all the studies)
    - *_scatterplot_meanrankvssdrank.pdf - scatter plot showing all genes as points, mean rank on X axis sd rank on y axis
  - Converting genes variation (sd rank) to deciles: least to most variable
    - *_sinaplot_sddecileVsmeanrank.pdf - sina plot with variation deciles on X axis, expression level (rank, mean) on Y axis
    - *_ANOVAresults_sddecileVsmeanrank.txt - ANOVA overall results for deciles vs mean expression
    - *_ANOVAresults_tukey_sddecileVsmeanrank.txt - Tukey's HSD test results for all comparisons (so can look at specific ones as desired)

Analyze Glaser-Schmitt et al. 2024 ASE, gene expression level data: data_analysis_scripts/ glaserschmitt_drosase_meanexprvsaseetc.R

- Inputs (also get by running script with --help)

```
-t, --transcriptinfo  Path to file with transcript name, comment,
                      length, GC% - from using bioawk on transcript
                      fasta [default:
                      FlyBase_transcript_LengthGC_20220115.txt.gz]
-g, --gcountsMG       Path to midgut gene counts for each sample
                      (GSE263264_MG_gcounts_allgenotypes_allreads.tsv.gz
                      from GEO) [default:
                      GSE263264_MG_gcounts_allgenotypes_allreads.tsv.gz]
--gcountsHG           Path to hindgut gene counts for each sample
                      (GSE263264_HG_gcounts_allgenotypes_allreads.tsv.gz
                      from GEO) [default:
                      GSE263264_HG_gcounts_allgenotypes_allreads.tsv.gz]
-a, --aseMG           EXAMPLE Path to midgut ASE (subsampled) output,
                      from Data S3. Where specific strain cross is,
                      should have STRAIN [default:
                      s3data_ASESTRAIN.txt]
--aseHG               EXAMPLE Path to hindgut ASE (subsampled)
                      output, from Data S4. Where specific strain
                      cross is, should have STRAIN [default:
                      s4data_ASESTRAIN.txt]
-o, --outdir          Output directory [default: out]
-b, --baseoutname     Base name for all output files [default:
                      drosase_glaserschmitt2024]
```

- Outputs
  - *Related to my RNA normalization QC (in directory quality checks)*
    - *_parentalprepostVST_MAPlots.pdf - hex mean (rank) vs. SD plots on vst data (like in DESeq2 QC analyses), as well as non-vst data (see titles)
  - *Misc*
    - qualitychecks/*_regclass_scatters.pdf - DE vs. ASE scatter plots, for my own quick reference, colored by original regulatory class designations (from paper)

- qualitychecks/*_regclassNEW_scatters.pdf - as above, but with new regulatory classes that I've fixed (assigned genes without DE or ASE as conserved rather than any ambiguous; fixed cis x trans/cis + trans where directionality wasn't taken into account; labeled with my labels)

- *Gross mean vs variance of expression data (before looking at ASE data - just among the parentals within tissue) [12 parents - 3 x 4 strains]*
  - *All in Directory: parentmeanvar/*
  - *_overallparentalmeanvarexp_multiplenormalizations.txt.gz: data in long format. Mean expression (and SD, var, etc) across all parents when expression was normalized two different ways (see "expnormmethod") column.
  - *_overallparentalmeanvarexp_multiplenormalizations_plots.pdf: mean and SD per gene for the two normalizations plotted against one another (for QC; one page per plot)
  - *_overallparentalmeanvarexp_multiplenormalizations_corrs.txt: Mean vs. SD correlations for each tissue, normalization scheme
  - Stat results for comparing means across SD deciles:
    - *_overallparentalmeanvarexp_multiplenormalizations_SDDeciles_ANOVA.txt
    - *_overallparentalmeanvarexp_multiplenormalizations_SDDeciles_Tukey.txt
    - *_overallparentalmeanvarexp_multiplenormalizations_SDDeciles_Ns.txt
- *data_inclcombostrain_ExprAndASE.txt.gz - **the data for all the downstream analyses. Columns:**

Column name	Description if not obvious
strain	strain crossed OR 'all' - if all, it's within-tissue 'meta-strain': genes that are present in ASE analyses in *all 4 crosses (so each gene present 4x)*
tissue	
FBgn	
log2FoldChange.ParentVsParent	DE value from supp data input
padj.ParentVsParent	DE p-value from supp data input
log2FoldChange.ASE	ASE value from supp data input
padj.ASE	ASE p-value from supp data input
padj.CHM	CHM test adjusted p-value (comparing ASE & DE LFCs) from supp data input
signifAtThresholds.ASE	T or F, ASE signif at p<0.05 (paper threshold)
signifAtThresholds.ParentVsParent	T or F, DE signif at p<0.05 (paper threshold)
signifAtThresholds.ASE.thresh	T or F, ASE signif at p <0 .05 and FC > 1.5 (added FC threshold). (if p < 0.05 and FC not this level, NA)

signifAtThresholds.ParentVsParent.thresh	T or F, DE signif at p <0 .05 and FC > 1.5 (added FC threshold). (if p < 0.05 and FC not this level, NA)
regclass.orig	regulatory class from paper supp data (*some mistakes with this I think)*
inform	T or F - was gene included in ASE/DE analysis (vs just having enough expression)
regclass.update	regulatory class as I've called it - uses same data as input, but uses my terms, calls any without ASE and DE as conserved, fixes a mixup with cis+trans/cisxtrans not taking into account directionality
ciscompensated	for genes with cis effects (ASE), T or F for if compensated. F: cis and enhancing classes; T: compensatory and overcompensating classesd ('compensating' means DE lower magnitude than ASE, excluding this from this column entirely)
regclass.update.comb	regclass.update with cis-trans opposing combined together: compensatory, overcompensating, and compensating are termed 'cis-trans opposing'
CGnum	gene info
Chrom	gene info
Length	gene info - from input from paper
ntranscripts	gene info - from flybase transcript fasta processing
length.median	gene info - from flybase transcript fasta processing - median length of transcript
GC.median	median GC proportion - from flybase transcript fasta processing
meanexp.vst	Expression: mean of the two parental strains' here, after length normalizing (cqn) and vst normalization (DESeq2)
meanexp.shlog	Expression: mean of the two parental strains here, after length normalizing (cqn) and log2 + 1 normalization
NTestedCrosses	# crosses this gene had ASE test data in

testedInAll	T or F, did this have results in all tested crosses

- Two-way comparisons (groups' expression compared)
  - **For vst norm data and log2(+1) norm data - see file name for which**
  - Comparisons done:
    -

myname	shortname	myxname	columnname	mynarrow
Used for ASE/DE testing vs not	inform	Used for ASE/DE testing	inform	inform==T | inform==F
ASE vs not	ASE	ASE called	signifAtThresholds.ASE	inform==T
DE vs not	DE	DE called	signifAtThresholds.ParentVsParent	inform==T
ASE vs not - FC > 1.5	ASE_FC	ASE called (FC > 1.5)	signifAtThresholds.ASE.thresh	inform==T&!is.na(signifAtThresholds.ASE.thresh)
DE vs not - FC > 1.5	DE_FC	DE called (FC > 1.5)	signifAtThresholds.ParentVsParent.thresh	inform==T&!is.na(signifAtThresholds.ParentVsParent.thresh)
Cis-regulatory change: compensated in trans or not	ciscompensated	Cis change is compensated	ciscompensated	inform==T & !is.na(ciscompensated)

  - *_twowaytests_mannwhitneyresults.txt - stat test results for these comparisons. (Same format as ase vs general script output)
  - *_twowaytests_sinaplots.pdf - sina plots of above
- Multi-way comparisosn (groups' expression compared)
  - **For vst norm data and log2(+1) norm data - see file name for which**
  - Comparisons done:
    -

myname	shortname	myxname	columnname	mynarrow
Regulatory pattern (original, issues)	regpat_orig		regclass.orig	!is.na(regclass.orig)
Regulatory pattern (updated, all)	regpat_update		regclass.update	!is.na(regclass.update)
Regulatory pattern (updated, simplified)	regpat_update_simple		regclass.update.comb	!is.na(regclass.update.comb)
Regulatory pattern (updated, changed only)	regpat_update_changed		regclass.update	!is.na(regclass.update) & !regclass%in%c('ambiguous', 'conserved')

Regulatory pattern (updated, simplified, changed only)	regpat_update_simplechanged		regclass.update.comb	!is.na(regclass.update) & !regclass%in%c('ambiguous', 'conserved')

- Stat results (same formats as my ase vs general script output):
  - *_multiwaytests_ANOVAresults.txt
  - *_multiwaytests_TukeyHSDresults.txt
  - *_multiwaytests_nspercategory.txt
- *_multiwaytests_sinaplots.pdf: sina plots, one per page; annoted with overall ANOVA p