DSG ASSIGNMENT - 2 (BIO 541)

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

NAME - PARAS DHIMAN

ROLL NO.- 2021482

1.

(a.)

Genomes | Genome Browser | Tools | Mirrors | Downloads | My Data | Projects | Help | About Us

mm8 STS Marker X84000

Chromosome: chr12
Start: 57628621
End: 57628751

UCSC STS Marker ID: 16942
UniSts Marker ID: 163526
MGI Marker ID: 97493
MGI Marker Name: PAX9

Left Primer: CACCAACACCAACACCTCTC
Right Primer: CTGAATACACACGACCTGGG
Distance: 131 bps

Map Position

Name	Chromosome	Position	
MGD	12	26.00	
Name	**Chromosome**	**Position**	**Score**
WHITEHEAD-MRC_RH	12	579.51	0.00

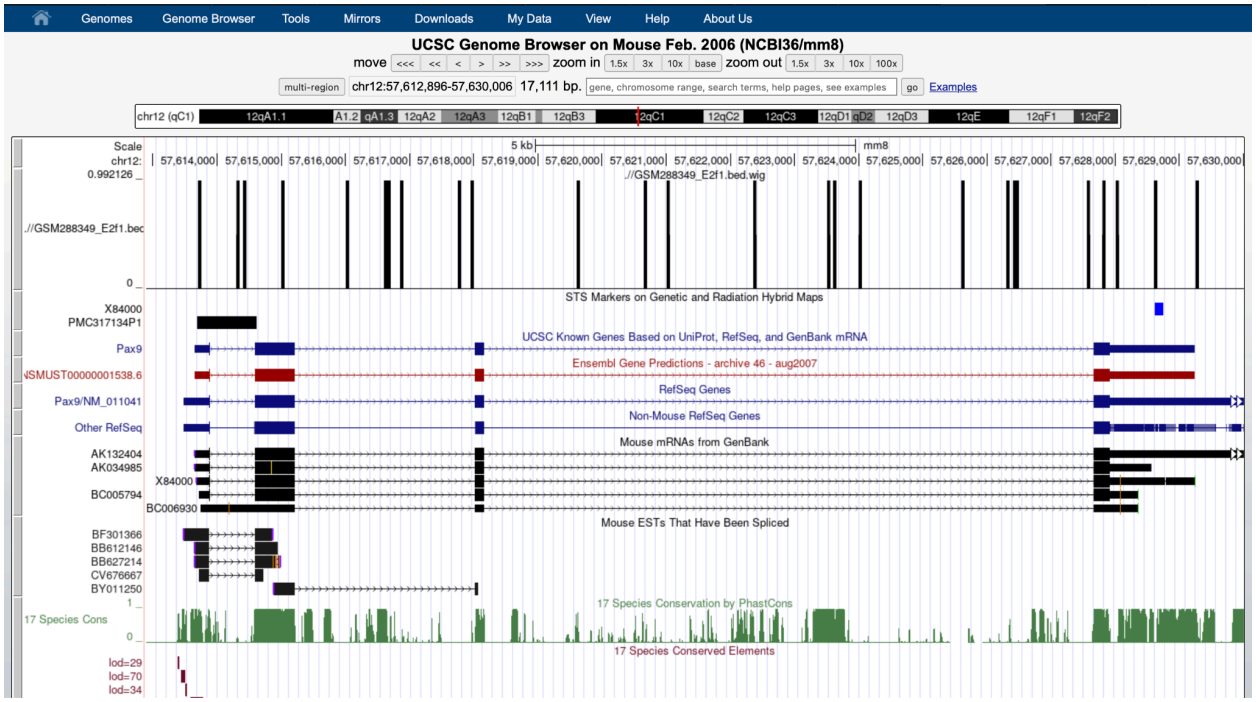
Genomic Alignments:

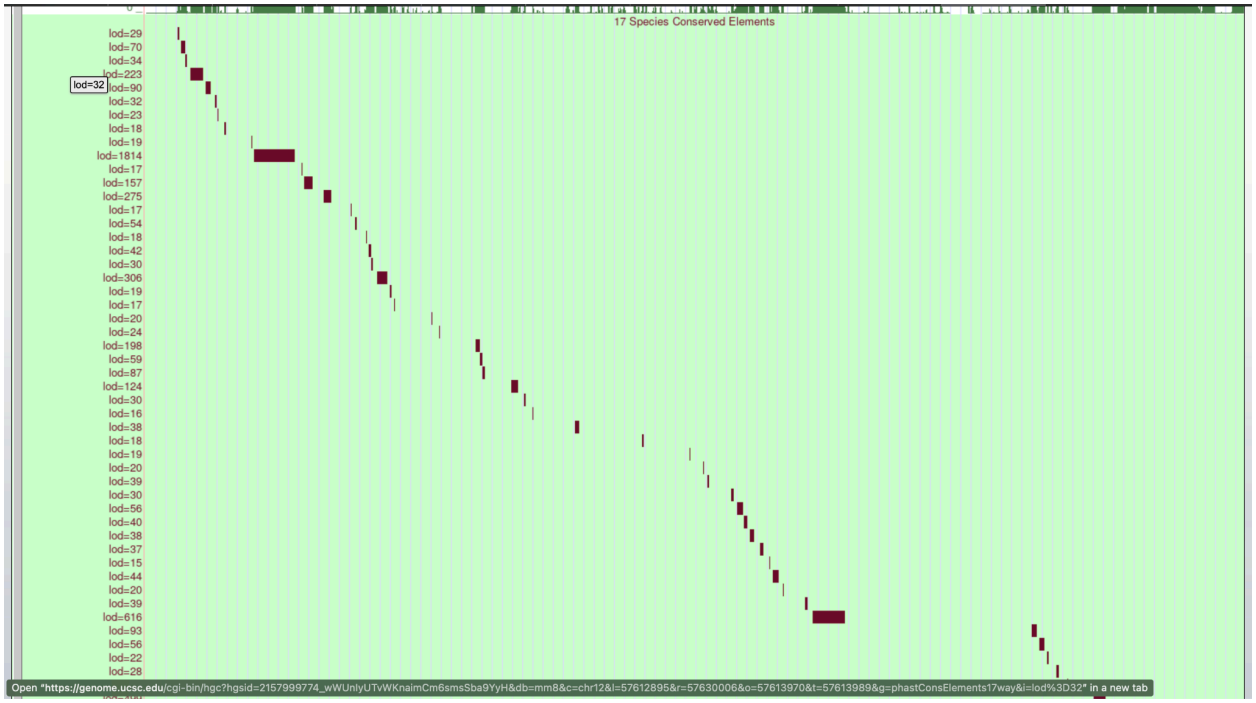
Primers:

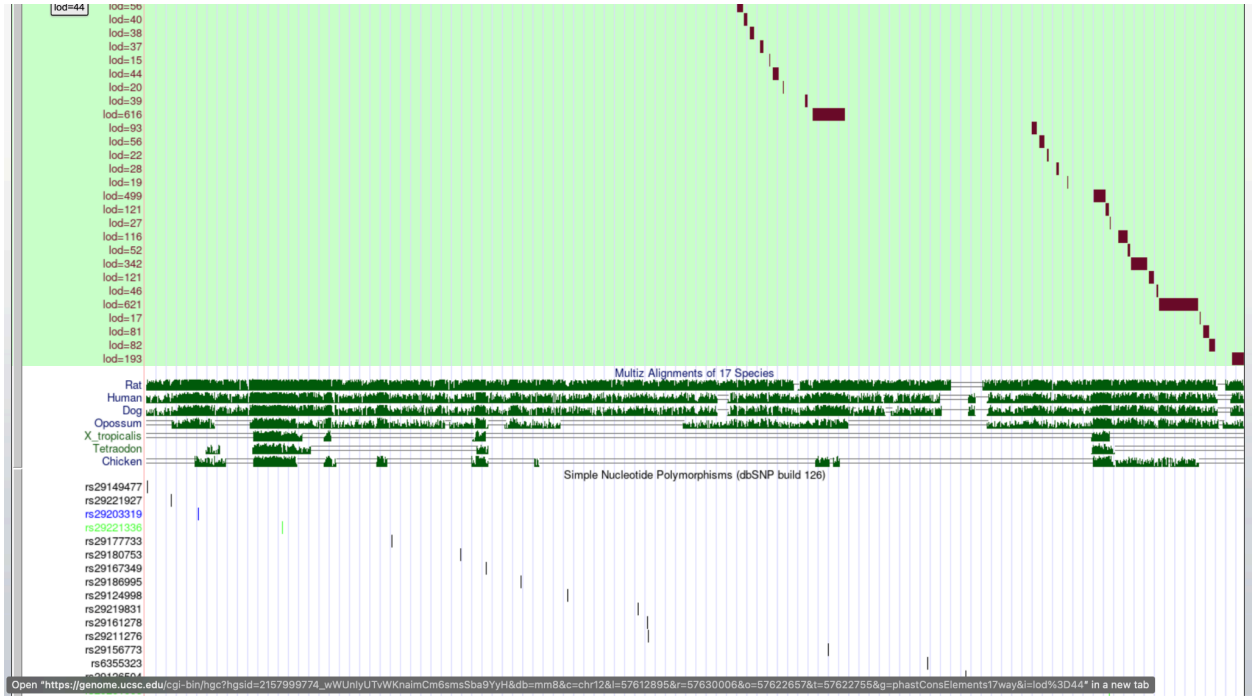
BROWSER	SIZE	IDENTITY	CHROMOSOME	STRAND	START	END	QUERY	START	END	TOTAL
browser	40	100.0%	12	+	57628621	57628751	16942_X84000	1	131	131

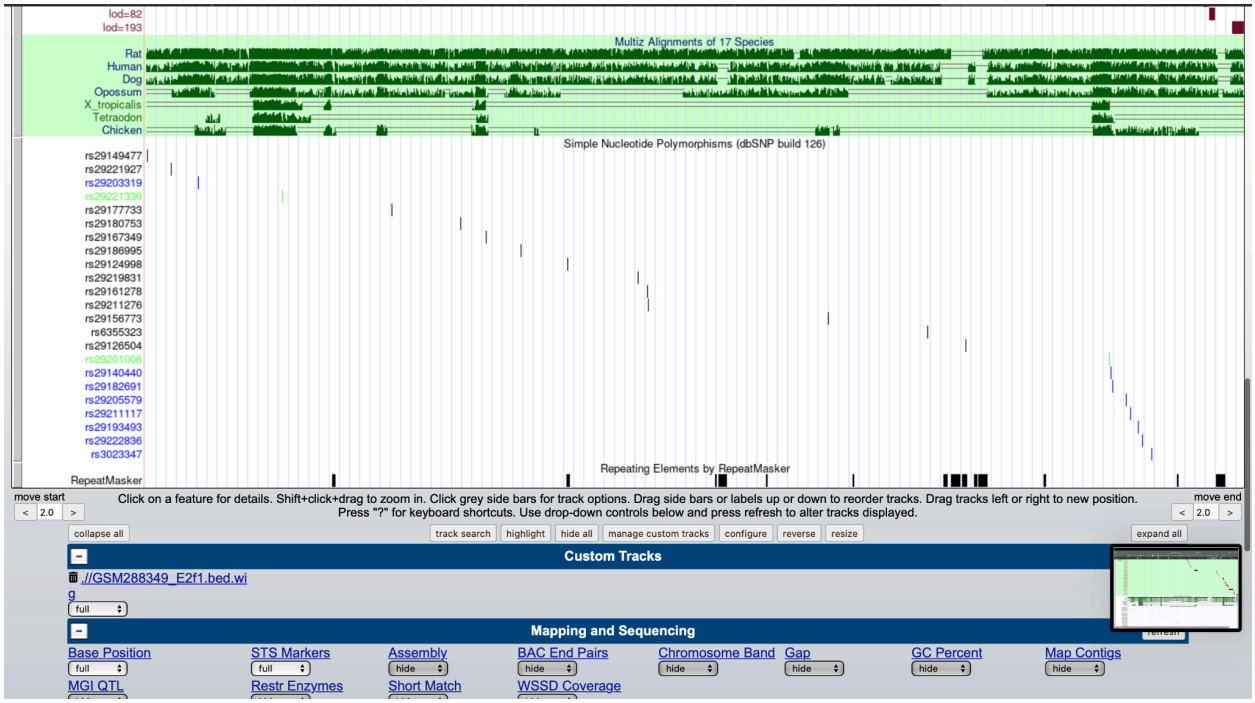
View details of parts of alignment within browser window.

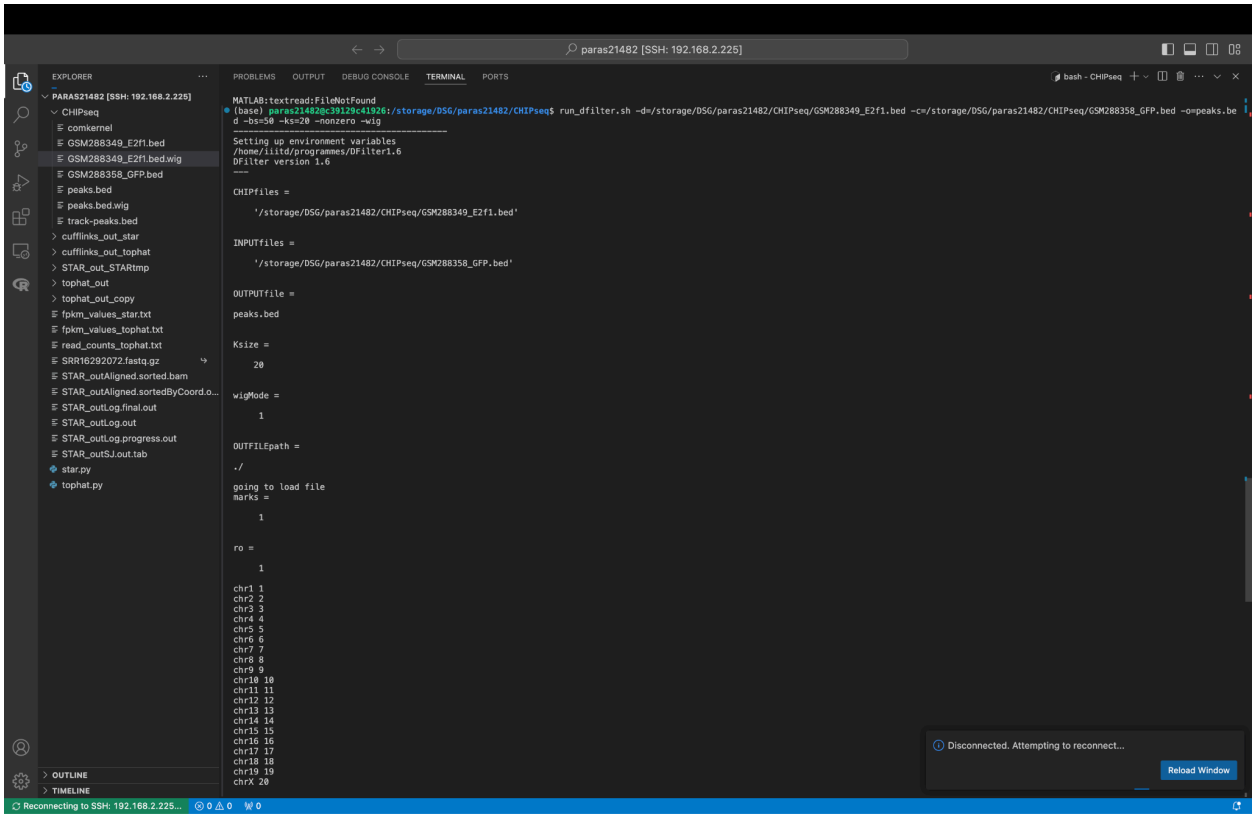
This is the only location found for X84000

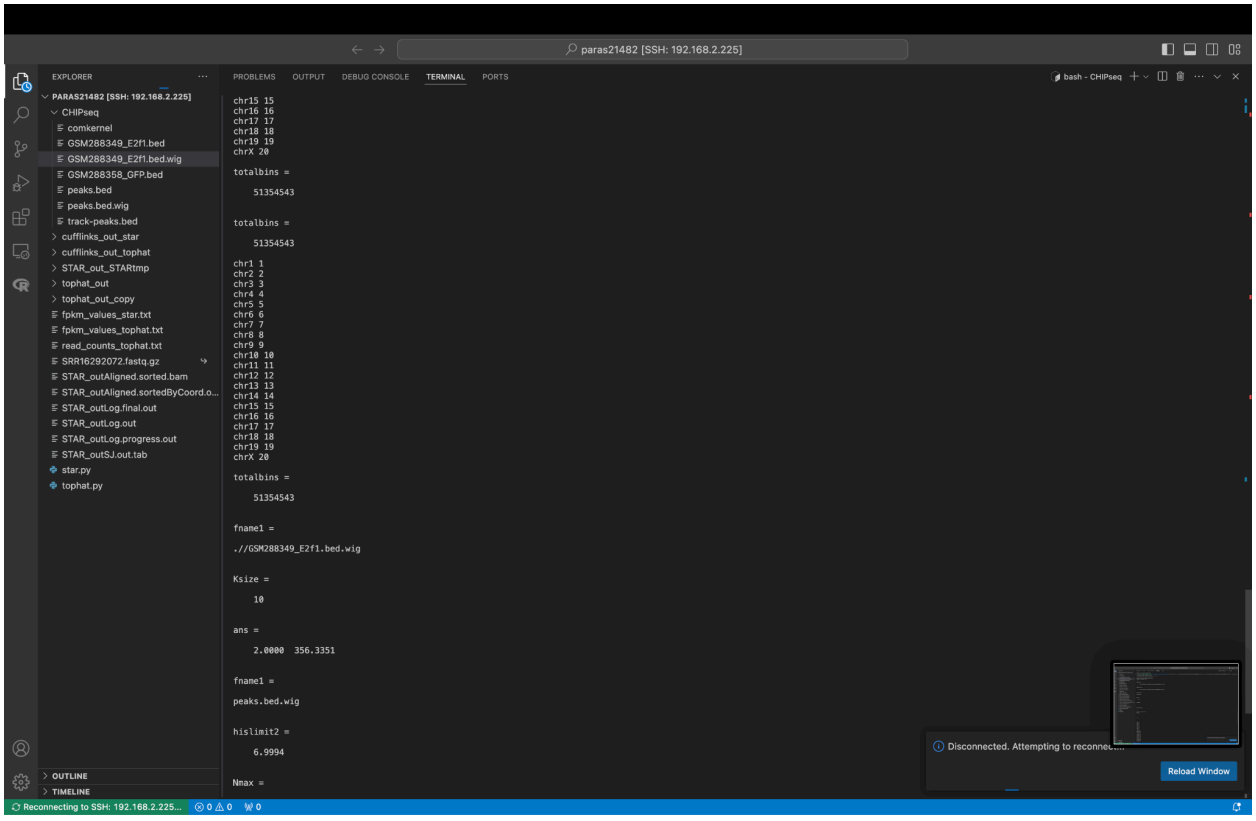

Genomes
Genome Browser
Tools
Mirrors
Downloads
My Data
View
Help
About Us
UCSC Genome Browser on Mouse Feb. 2006 (NCBI36/mm8)
move
zoom in
zoom out
multi-region
chr12:57,612,896-57,630,006
17,111 bp.
gene, chromosome range, search terms, help pages, see examples
go
Examples
chr12 (qC1)
Scale
chr12:
5 kb
mm8
.//GSM288349_E2f1.bed.wig
STS Markers on Genetic and Radiation Hybrid Maps
X84000
PMC317134P1
UCSC Known Genes Based on UniProt, RefSeq, and GenBank mRNA
Pax9
Ensembl Gene Predictions - archive 46 - aug2007
RefSeq Genes
Pax9/NM_011041
Non-Mouse RefSeq Genes
Other RefSeq
Mouse mRNAs from GenBank
AK132404
AK034985
BC005794
BC006930
Mouse ESTs That Have Been Spliced
BF301366
BB612146
BB627214
CV676667
BY011250
17 Species Conservation by PhastCons
17 Species Cons
17 Species Conserved Elements
lod=29
lod=70
lod=34

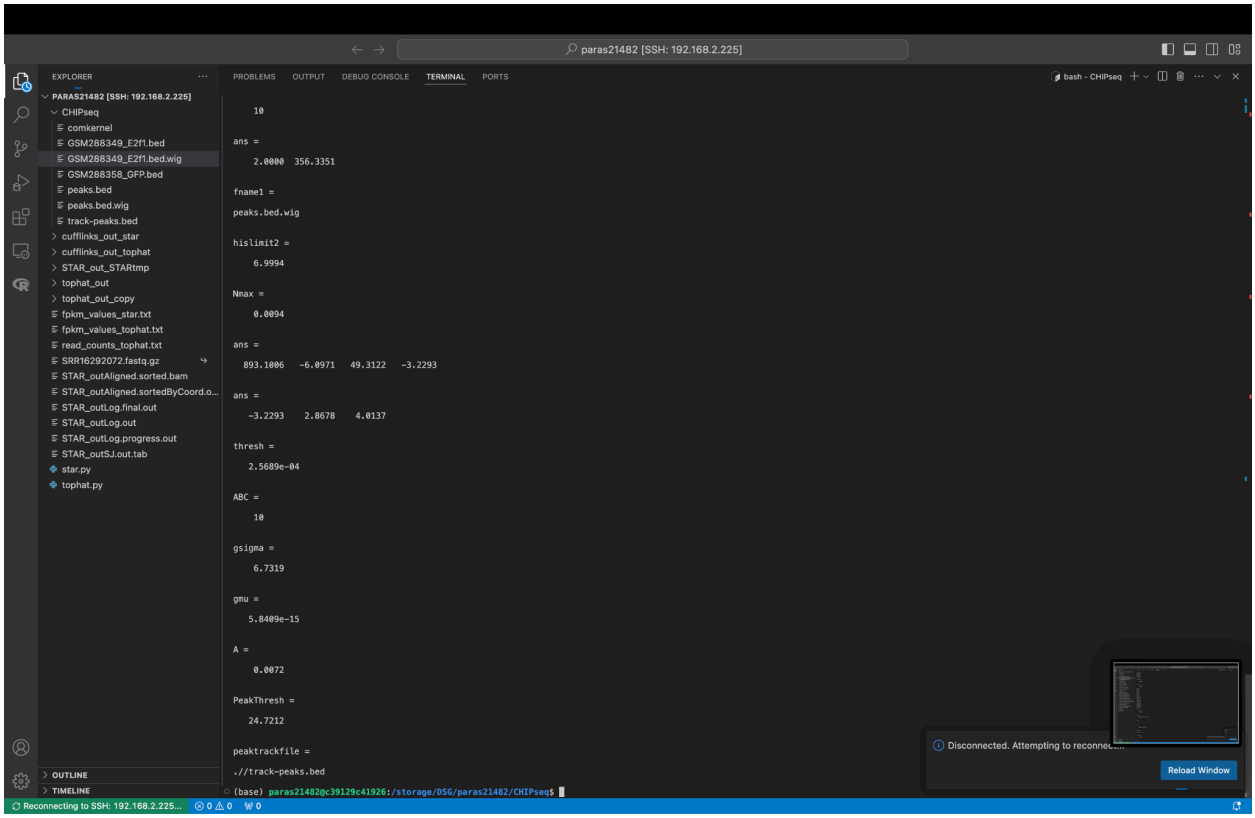

17 Species Conserved Elements
lod=29
lod=70
lod=34
lod=223
lod=32
lod=90
lod=32
lod=23
lod=18
lod=19
lod=1814
lod=17
lod=157
lod=275
lod=17
lod=54
lod=18
lod=42
lod=30
lod=306
lod=19
lod=17
lod=20
lod=24
lod=198
lod=59
lod=87
lod=124
lod=30
lod=16
lod=38
lod=18
lod=19
lod=20
lod=39
lod=30
lod=56
lod=40
lod=38
lod=37
lod=15
lod=44
lod=20
lod=39
lod=616
lod=93
lod=56
lod=22
lod=28
Open "https://genome.ucsc.edu/cgi-bin/hgc?hgsid=2157999774_wWUnlyUTvWKnaimCm6smsSba9YyH&db=mm8&c=chr12&l=57612895&r=57630006&o=57613970&t=57613989&g=phastConsElements17way&i=lod%3D32" in a new tab


lod=44
lod=40
lod=38
lod=37
lod=15
lod=44
lod=20
lod=39
lod=616
lod=93
lod=56
lod=22
lod=28
lod=19
lod=499
lod=121
lod=27
lod=116
lod=52
lod=342
lod=121
lod=46
lod=621
lod=17
lod=81
lod=82
lod=193
Multiz Alignments of 17 Species
Rat
Human
Dog
Opossum
X_tropicalis
Tetraodon
Chicken
Simple Nucleotide Polymorphisms (dbSNP build 126)
rs29149477
rs29221927
rs29203319
rs29221336
rs29177733
rs29180753
rs29167349
rs29186995
rs29124998
rs29219831
rs29161278
rs29211276
rs29156773
rs6355323
Open "https://genome.ucsc.edu/cgi-bin/hgc?hgsid=2157999774_wWUnlyUTvWKnaimCm6smsSba9YyH&db=mm8&c=chr12&l=57612895&r=57630006&o=57622657&t=57622755&g=phastConsElements17way&i=lod%3D44" in a new tab

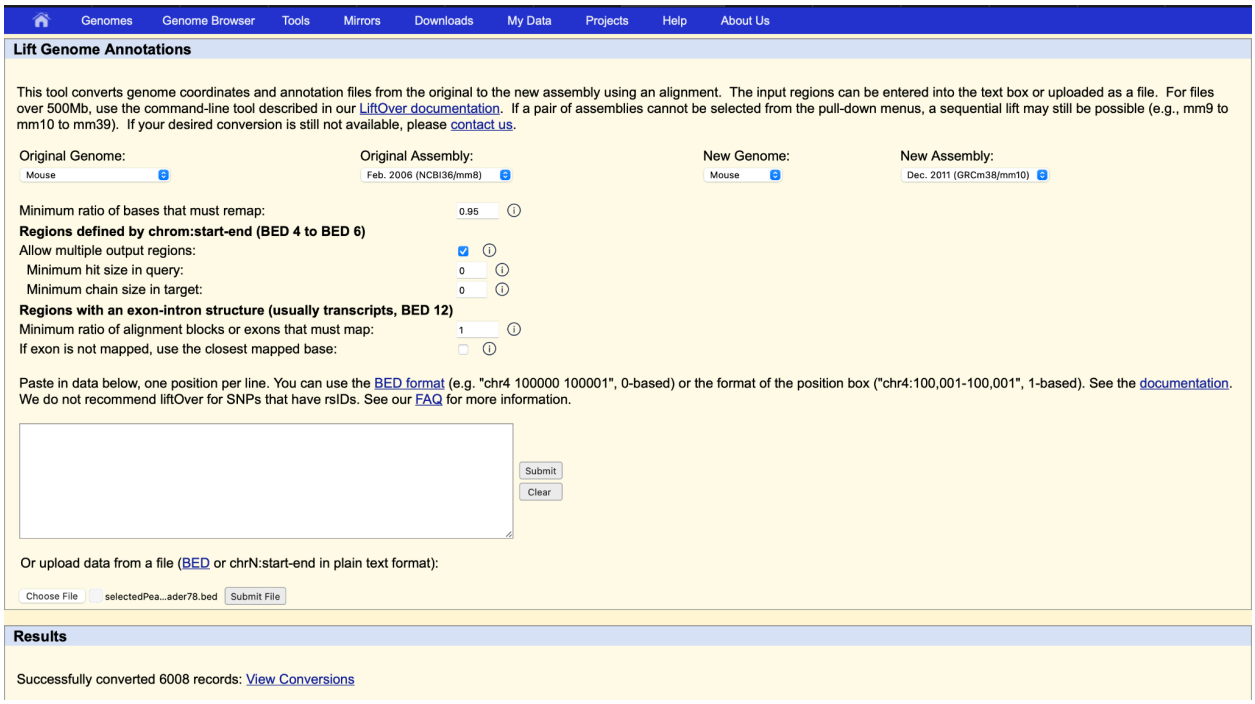

lod=82
lod=193
Multiz Alignments of 17 Species
Rat
Human
Dog
Opossum
X_tropicalis
Tetraodon
Chicken
Simple Nucleotide Polymorphisms (dbSNP build 126)
rs29149477
rs29221927
rs29203319
rs29221336
rs29177733
rs29180753
rs29167349
rs29186995
rs29124998
rs29219831
rs29161278
rs29211276
rs29156773
rs6355323
rs29126504
rs29201006
rs29140440
rs29182691
rs29205579
rs29211117
rs29193493
rs29222836
rs3023347
Repeating Elements by RepeatMasker
RepeatMasker
move start
move end
Click on a feature for details. Shift+click+drag to zoom in. Click grey side bars for track options. Drag side bars or labels up or down to reorder tracks. Drag tracks left or right to new position.
Press "?" for keyboard shortcuts. Use drop-down controls below and press refresh to alter tracks displayed.
collapse all
track search
highlight
hide all
manage custom tracks
configure
reverse
resize
expand all
Custom Tracks
//GSM288349_E2f1.bed.wig
full
Mapping and Sequencing
Base Position
STS Markers
Assembly
BAC End Pairs
Chromosome Band
Gap
GC Percent
Map Contigs
MGI QTL
Restr Enzymes
Short Match
WSSD Coverage


paras21482 [SSH: 192.168.2.225]
EXPLORER
PARAS21482 [SSH: 192.168.2.225]
CHIPseq
comkernel
GSM288349_E2f1.bed
GSM288349_E2f1.bed.wig
GSM288358_GFP.bed
peaks.bed
peaks.bed.wig
track-peaks.bed
cufflinks_out_star
cufflinks_out_tophat
STAR_out_STARtmp
tophat_out
tophat_out_copy
fpkm_values_star.txt
fpkm_values_tophat.txt
read_counts_tophat.txt
SRR16292072.fastq.gz
STAR_outAligned.sorted.bam
STAR_outAligned.sortedByCoord.o...
STAR_outLog.final.out
STAR_outLog.out
STAR_outLog.progress.out
STAR_outSJ.out.tab
star.py
tophat.py
OUTLINE
TIMELINE
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
bash - CHIPseq
MATLAB:textread:FileNotFound
(base) paras21482@c39129c41926:/storage/DSG/paras21482/CHIPseq$ run_dfilter.sh -d=/storage/DSG/paras21482/CHIPseq/GSM288349_E2f1.bed -c=/storage/DSG/paras21482/CHIPseq/GSM288358_GFP.bed -o=peaks.bed -bs=50 -ks=20 -nonzero -wig
Setting up environment variables
/home/iiitd/programmes/DFilter1.6
DFilter version 1.6
CHIPfiles =
'/storage/DSG/paras21482/CHIPseq/GSM288349_E2f1.bed'
INPUTfiles =
'/storage/DSG/paras21482/CHIPseq/GSM288358_GFP.bed'
OUTPUTfile =
peaks.bed
Ksize =
20
wigMode =
1
OUTFILEpath =
./
going to load file
marks =
1
ro =
1
chr1 1
chr2 2
chr3 3
chr4 4
chr5 5
chr6 6
chr7 7
chr8 8
chr9 9
chr10 10
chr11 11
chr12 12
chr13 13
chr14 14
chr15 15
chr16 16
chr17 17
chr18 18
chr19 19
chrX 20
Disconnected. Attempting to reconnect...
Reload Window
Reconnecting to SSH: 192.168.2.225...


paras21482 [SSH: 192.168.2.225]
EXPLORER
PARAS21482 [SSH: 192.168.2.225]
CHIPseq
comkernel
GSM288349_E2f1.bed
GSM288349_E2f1.bed.wig
GSM288358_GFP.bed
peaks.bed
peaks.bed.wig
track-peaks.bed
cufflinks_out_star
cufflinks_out_tophat
STAR_out_STARtmp
tophat_out
tophat_out_copy
fpkm_values_star.txt
fpkm_values_tophat.txt
read_counts_tophat.txt
SRR16292072.fastq.gz
STAR_outAligned.sorted.bam
STAR_outAligned.sortedByCoord.o...
STAR_outLog.final.out
STAR_outLog.out
STAR_outLog.progress.out
STAR_outSJ.out.tab
star.py
tophat.py
OUTLINE
TIMELINE
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
bash - CHIPseq
chr15 15
chr16 16
chr17 17
chr18 18
chr19 19
chrX 20
totalbins =
51354543
totalbins =
51354543
chr1 1
chr2 2
chr3 3
chr4 4
chr5 5
chr6 6
chr7 7
chr8 8
chr9 9
chr10 10
chr11 11
chr12 12
chr13 13
chr14 14
chr15 15
chr16 16
chr17 17
chr18 18
chr19 19
chrX 20
totalbins =
51354543
fname1 =
.//GSM288349_E2f1.bed.wig
Ksize =
10
ans =
2.0000 356.3351
fname1 =
peaks.bed.wig
hislimit2 =
6.9994
Nmax =
Disconnected. Attempting to reconnect...
Reload Window
Reconnecting to SSH: 192.168.2.225...


paras21482 [SSH: 192.168.2.225]
EXPLORER
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
bash - CHIPseq
PARAS21482 [SSH: 192.168.2.225]
CHIPseq
comkernel
GSM288349_E2f1.bed
GSM288349_E2f1.bed.wig
GSM288358_GFP.bed
peaks.bed
peaks.bed.wig
track-peaks.bed
cufflinks_out_star
cufflinks_out_tophat
STAR_out_STARtmp
tophat_out
tophat_out_copy
fpkm_values_star.txt
fpkm_values_tophat.txt
read_counts_tophat.txt
SRR16292072.fastq.gz
STAR_outAligned.sorted.bam
STAR_outAligned.sortedByCoord.o...
STAR_outLog.final.out
STAR_outLog.out
STAR_outLog.progress.out
STAR_outSJ.out.tab
star.py
tophat.py
OUTLINE
TIMELINE
10
ans =
2.0000 356.3351
fname1 =
peaks.bed.wig
hislimit2 =
6.9994
Nmax =
0.0094
ans =
893.1006 -6.0971 49.3122 -3.2293
ans =
-3.2293 2.8678 4.0137
thresh =
2.5689e-04
ABC =
10
gsigma =
6.7319
gmu =
5.8409e-15
A =
0.0072
PeakThresh =
24.7212
peaktrackfile =
.//track-peaks.bed
(base) paras21482@c39129c41926:/storage/DSG/paras21482/CHIPseq$
Disconnected. Attempting to reconnect...
Reload Window
Reconnecting to SSH: 192.168.2.225...

(b.)


Genomes
Genome Browser
Tools
Mirrors
Downloads
My Data
Projects
Help
About Us
Lift Genome Annotations
This tool converts genome coordinates and annotation files from the original to the new assembly using an alignment. The input regions can be entered into the text box or uploaded as a file. For files over 500Mb, use the command-line tool described in our LiftOver documentation. If a pair of assemblies cannot be selected from the pull-down menus, a sequential lift may still be possible (e.g., mm9 to mm10 to mm39). If your desired conversion is still not available, please contact us.
Original Genome:
Mouse
Original Assembly:
Feb. 2006 (NCBI36/mm8)
New Genome:
Mouse
New Assembly:
Dec. 2011 (GRCm38/mm10)
Minimum ratio of bases that must remap:
0.95
Regions defined by chrom:start-end (BED 4 to BED 6)
Allow multiple output regions:
Minimum hit size in query:
0
Minimum chain size in target:
0
Regions with an exon-intron structure (usually transcripts, BED 12)
Minimum ratio of alignment blocks or exons that must map:
1
If exon is not mapped, use the closest mapped base:
Paste in data below, one position per line. You can use the BED format (e.g. "chr4 100000 100001", 0-based) or the format of the position box ("chr4:100,001-100,001", 1-based). See the documentation. We do not recommend liftOver for SNPs that have rsIDs. See our FAQ for more information.
Submit
Clear
Or upload data from a file (BED or chrN:start-end in plain text format):
Choose File
selectedPea...ader78.bed
Submit File
Results
Successfully converted 6008 records: View Conversions

Genomes | Genome Browser | Tools | Mirrors | Downloads | My Data | Projects | Help | About Us

Lift Genome Annotations

This tool converts genome coordinates and annotation files from the original to the new assembly using an alignment. The input regions can be entered into the text box or uploaded as a file. For files over 500Mb, use the command-line tool described in our LiftOver documentation. If a pair of assemblies cannot be selected from the pull-down menus, a sequential lift may still be possible (e.g., mm9 to mm10 to mm39). If your desired conversion is still not available, please contact us.

Original Genome:	Original Assembly:	New Genome:	New Assembly:
Mouse	Feb. 2006 (NCBI36/mm8)	Mouse	Dec. 2011 (GRCm38/mm10)

Minimum ratio of bases that must remap: 0.95

Regions defined by chrom:start-end (BED 4 to BED 6)

Allow multiple output regions: ☑

Minimum hit size in query: 0

Minimum chain size in target: 0

Regions with an exon-intron structure (usually transcripts, BED 12)

Minimum ratio of alignment blocks or exons that must map: 1

If exon is not mapped, use the closest mapped base: ☐

Paste in data below, one position per line. You can use the BED format (e.g. "chr4 100000 100001", 0-based) or the format of the position box ("chr4:100,001-100,001", 1-based). See the documentation. We do not recommend liftOver for SNPs that have rsIDs. See our FAQ for more information.

Submit

Clear

Or upload data from a file (BED or chrN:start-end in plain text format):

Choose File no file selected Submit File

Results

Successfully converted 6008 records: View Conversions

Submit

Clear

Or upload data from a file (BED or chrN:start-end in plain text format):

Choose File no file selected Submit File

Results

Successfully converted 6008 records: View Conversions

Conversion failed on 9 records. Display failure file Explain failure messages

Failed input regions:

Parameters Used

Minimum ratio of bases that must map: 0.95

BED 4 to BED 6 Options

Allow multiple output regions: on

Minimum hit size in query: 0

Minimum chain size in target: 0

BED 12 Options

Min ratio of alignment blocks or exons that must map: 1.00

If thickStart/thickEnd is not mapped, use the closest mapped base: off

Command Line Tool

To lift genome annotations locally on Linux systems, download the *LiftOver* executable and the appropriate chain file. Run *liftOver* with no arguments to see the usage message. See the LiftOver documentation.

```
#Partially deleted in new
chr4    129901258       129902258       652.14  129901758       8       7       0
#Partially deleted in new
chr9    100886151       100887901       1165.14 100886901       2       9       0
#Partially deleted in new
chr12   4241110 4243260 1327.72 4241960 2       11      0
#Partially deleted in new
chr12   4865460 4867010 529.02  4866060 5       6       0
#Partially deleted in new
chr14   17030261        17031511        618.91  17030911        7       7       0
#Partially deleted in new
chr14   17031911        17033261        657.93  17032561        8       7       0
#Partially deleted in new
chr17   32013098        32015198        778.10  32014598        1       7       0
#Partially deleted in new
chr17   33413048        33414948        637.00  33414348        7       6       0
#Partially deleted in new
chr17   33512648        33514148        953.29  33513548        1       9       0
```

```
Deleted in new:
    Sequence intersects no chains
Partially deleted in new:
    Sequence insufficiently intersects one chain
Split in new:
    Sequence insufficiently intersects multiple chains
Duplicated in new:
    Sequence sufficiently intersects multiple chains
Boundary problem:
    Missing start or end base in an exon
```

(c.)

Genomes | Genome Browser | Tools | Mirrors | Downloads | My Data | Projects | Help | About Us

Lift Genome Annotations

This tool converts genome coordinates and annotation files from the original to the new assembly using an alignment. The input regions can be entered into the text box or uploaded as a file. For files over 500Mb, use the command-line tool described in our LiftOver documentation. If a pair of assemblies cannot be selected from the pull-down menus, a sequential lift may still be possible (e.g., mm9 to mm10 to mm39). If your desired conversion is still not available, please contact us.

Original Genome:	Original Assembly:	New Genome:	New Assembly:
Mouse	Feb. 2006 (NCBI36/mm8)	Mouse	Dec. 2011 (GRCm38/mm10)

Minimum ratio of bases that must remap: 0.95

Regions defined by chrom:start-end (BED 4 to BED 6)

Allow multiple output regions: ☑

Minimum hit size in query: 0

Minimum chain size in target: 0

Regions with an exon-intron structure (usually transcripts, BED 12)

Minimum ratio of alignment blocks or exons that must map: 1

If exon is not mapped, use the closest mapped base: ☐

Paste in data below, one position per line. You can use the BED format (e.g. "chr4 100000 100001", 0-based) or the format of the position box ("chr4:100,001-100,001", 1-based). See the documentation. We do not recommend liftOver for SNPs that have rsIDs. See our FAQ for more information.

Submit Clear

Or upload data from a file (BED or chrN:start-end in plain text format):

Choose File no file selected Submit File

Results

Successfully converted 999 records: View Conversions

Paste in data below, one position per line. You can use the BED format (e.g. "chr4 100000 100001", 0-based) or the format of the position box ("chr4:100,001-100,001", 1-based). See the documentation. We do not recommend liftOver for SNPs that have rsIDs. See our FAQ for more information.

Submit Clear

Or upload data from a file (BED or chrN:start-end in plain text format):

Choose File no file selected Submit File

Results

Successfully converted 999 records: View Conversions

Parameters Used

Minimum ratio of bases that must map:	0.95
BED 4 to BED 6 Options	
Allow multiple output regions:	on
Minimum hit size in query:	0
Minimum chain size in target:	0
BED 12 Options	
Min ratio of alignment blocks or exons that must map:	1.00
If thickStart/thickEnd is not mapped, use the closest mapped base:	off

Command Line Tool

To lift genome annotations locally on Linux systems, download the *LiftOver* executable and the appropriate chain file. Run *liftOver* with no arguments to see the usage message. See the LiftOver documentation.

- Apr. 3, 2012: GREAT version 2 adds new annotations to human and mouse ontologies and visualization tools for data exploration.
- Feb. 18, 2012: The GREAT user help forums are opened.
- May 2, 2010: GREAT version 1 is launched, concurrent to Nature Biotechnology publication (reprint, Faculty of 1000 "Must Read"). How to Cite GREAT?

More news items...

Species Assembly

- Human: GRCh38 (UCSC hg38, Dec. 2013)
- Human: GRCh37 (UCSC hg19, Feb. 2009)
- Mouse: GRCm38 (UCSC mm10, Dec. 2011)
- Mouse: NCBI build 37 (UCSC mm9, Jul. 2007)

Can I use a different species or assembly?

Test regions

- BED file: Choose File selecte...78.bed
- BED data:

What should my test regions file contain?
How can I create a test set from a UCSC Genome Browser annotation track?

Background regions

- Whole genome
- BED file: Choose File no file selected
- BED data:

When should I use a background set?
What should my background regions file contain?

Association rule settings

Show settings »

Submit Reset

GREAT version 4.0.4 current (08/19/2019 to now)

Job Description

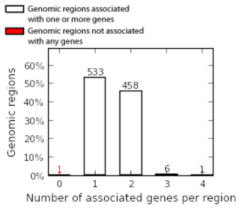
Loading...

Region-Gene Association Graphs

What do these graphs illustrate?

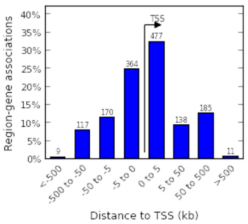
Number of associated genes per region

Download as PDF.



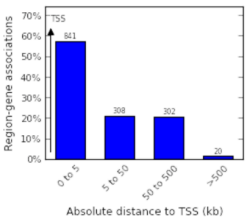
Binned by orientation and distance to TSS

Download as PDF.



Binned by absolute distance to TSS

Download as PDF.



Global Controls

Global Export *Which data is exported by each option?*

Ensembl Genes (no terms)

Global controls

Table controls: Export Shown top rows in this table: 20 Set Term annotation count: Min: 1 Max: Inf Set Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	*Binom FDR Q-Val*	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	*Hyper FDR Q-Val*	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage

No results meet your chosen criteria.

Significance

Significant P-Values are **bold** in the tables below.
Terms are **bold** when significant by both binomial and hypergeometric tests.
Which statistics does GREAT provide?

View

(•) Significant By Both ( ) Significant By Region-based Binomial ( ) Ignore statistical significance
Which view should I use?

Reset view

Ensembl Genes (no terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	*Binom FDR Q-Val*	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	*Hyper FDR Q-Val*	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage

No results meet your chosen criteria.

The test set of 999 genomic regions picked 1,110 (5%) of all 21,395 genes.
Ensembl Genes has 21,395 terms covering 21,395 (100%) of all 21,395 genes, and 21,395 term - gene associations.
21,395 ontology terms (100%) were tested using an annotation count range of [1, Inf].

GO Biological Process (20+ terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	*Binom FDR Q-Val*	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	*Hyper FDR Q-Val*	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage
ncRNA metabolic process	7	**4.5405e-15**	**8.4907e-12**	3.0929	65	6.51%	32	**2.9393e-6**	2.3695	52	423	4.68%
ncRNA processing	13	**2.8497e-13**	**2.8694e-10**	3.3332	51	5.11%	43	**3.2476e-5**	2.4542	41	322	3.69%
RNA processing	16	**4.0558e-12**	**3.3181e-9**	2.1334	98	9.81%	42	**3.1458e-5**	1.8917	74	754	6.67%
ribonucleoprotein complex biogenesis	19	**1.2755e-9**	**8.7875e-7**	2.4131	59	5.91%	44	**3.8358e-5**	2.2762	47	398	4.23%
tRNA metabolic process	24	**3.8723e-9**	**2.1120e-6**	3.6103	30	3.00%	71	**1.1835e-3**	2.7535	24	168	2.16%
posttranscriptional regulation of gene	28	**2.1917e-8**	**1.0246e-5**	2.2436	58	5.81%	82	**3.0006e-3**	2.0058	41	394	3.69%

GREAT Overview News Use GREAT Demo Video How to Cite Help Forum Bejerano Lab, Stanford University

GO Biological Process (20+ terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	*Binom FDR Q-Val*	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	*Hyper FDR Q-Val*	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage
ncRNA metabolic process	7	**4.5405e-15**	**8.4907e-12**	3.0929	65	6.51%	32	**2.9393e-6**	2.3695	52	423	4.68%
ncRNA processing	13	**2.8497e-13**	**2.8694e-10**	3.3332	51	5.11%	43	**3.2476e-5**	2.4542	41	322	3.69%
RNA processing	16	**4.0558e-12**	**3.3181e-9**	2.1334	98	9.81%	42	**3.1458e-5**	1.8917	74	754	6.67%
ribonucleoprotein complex biogenesis	19	**1.2755e-9**	**8.7875e-7**	2.4131	59	5.91%	44	**3.8358e-5**	2.2762	47	398	4.23%
tRNA metabolic process	24	**3.8723e-9**	**2.1120e-6**	3.6103	30	3.00%	71	**1.1835e-3**	2.7535	24	168	2.16%
posttranscriptional regulation of gene expression	28	**2.1917e-8**	**1.0246e-5**	2.2436	58	5.81%	82	**3.0006e-3**	2.0058	41	394	3.69%
RNA modification	29	**3.4338e-8**	**1.5500e-5**	3.8962	24	2.40%	90	**3.4649e-3**	2.9298	19	125	1.71%
multi-organism metabolic process	31	**5.3369e-8**	**2.2535e-5**	6.6254	14	1.40%	135	**1.7099e-2**	4.8187	8	32	0.72%
RNA localization	34	**1.2824e-7**	**4.9371e-5**	3.2217	28	2.80%	91	**3.6787e-3**	2.6669	22	159	1.98%
rRNA processing	35	**1.8111e-7**	**6.7734e-5**	3.0894	29	2.90%	98	**4.8218e-3**	2.4871	24	186	2.16%
viral gene expression	37	**2.3101e-7**	**8.1728e-5**	10.8804	9	0.90%	139	**1.9612e-2**	6.4249	6	18	0.54%
organelle fission	41	**2.8488e-7**	**9.0954e-5**	2.3103	46	4.60%	60	**2.5433e-4**	2.2988	39	327	3.51%
rRNA metabolic process	45	**5.0827e-7**	**1.4785e-4**	2.9340	29	2.90%	108	**7.3315e-3**	2.4093	24	192	2.16%
translation	47	**6.6821e-7**	**1.8610e-4**	2.1941	48	4.80%	145	**2.4124e-2**	1.8603	36	373	3.24%
histone modification	49	**1.0794e-6**	**2.8834e-4**	2.0474	54	5.41%	146	**2.5247e-2**	1.8739	35	360	3.15%
nuclear division	50	**1.2173e-6**	**3.1870e-4**	2.3128	41	4.10%	79	**2.3914e-3**	2.1991	34	298	3.06%
RNA phosphodiester bond hydrolysis	51	**1.2504e-6**	**3.2094e-4**	3.3867	22	2.20%	165	**4.5828e-2**	2.4453	17	134	1.53%
cellular response to leukemia inhibitory factor	60	**2.3047e-6**	**5.0281e-4**	2.4401	35	3.50%	121	**1.3098e-2**	2.6895	18	129	1.62%
RNA export from nucleus	62	**2.8203e-6**	**5.9545e-4**	4.7211	14	1.40%	129	**1.5460e-2**	3.4522	12	67	1.08%
macromolecule methylation	66	**3.3209e-6**	**6.5864e-4**	2.3317	37	3.70%	174	**4.8040e-2**	2.1010	23	211	2.07%

The test set of 999 genomic regions picked 1,110 (5%) of all 21,395 genes.
GO Biological Process has 13,090 terms covering 17,925 (84%) of all 21,395 genes, and 1,163,819 term - gene associations.

GO Biological Process (20+ terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	*Binom FDR Q-Val*	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	*Hyper FDR Q-Val*	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage
ncRNA metabolic process	7	**4.5405e-15**	**8.4907e-12**	3.0929	65	6.51%	32	**2.9393e-6**	2.3695	52	423	4.68%
ncRNA processing	13	**2.8497e-13**	**2.8694e-10**	3.3332	51	5.11%	43	**3.2476e-5**	2.4542	41	322	3.69%
RNA processing	16	**4.0558e-12**	**3.3181e-9**	2.1334	98	9.81%	42	**3.1458e-5**	1.8917	74	754	6.67%
ribonucleoprotein complex biogenesis	19	**1.2755e-9**	**8.7875e-7**	2.4131	59	5.91%	44	**3.8358e-5**	2.2762	47	398	4.23%
tRNA metabolic process	24	**3.8723e-9**	**2.1120e-6**	3.6103	30	3.00%	71	**1.1835e-3**	2.7535	24	168	2.16%
posttranscriptional regulation of gene expression	28	**2.1917e-8**	**1.0246e-5**	2.2436	58	5.81%	82	**3.0006e-3**	2.0058	41	394	3.69%
RNA modification	29	**3.4338e-8**	**1.5500e-5**	3.8962	24	2.40%	90	**3.4649e-3**	2.9298	19	125	1.71%
multi-organism metabolic process	31	**5.3369e-8**	**2.2535e-5**	6.6254	14	1.40%	135	**1.7099e-2**	4.8187	8	32	0.72%
RNA localization	34	**1.2824e-7**	**4.9371e-5**	3.2217	28	2.80%	91	**3.6787e-3**	2.6669	22	159	1.98%
rRNA processing	35	**1.8111e-7**	**6.7734e-5**	3.0894	29	2.90%	98	**4.8218e-3**	2.4871	24	186	2.16%
viral gene expression	37	**2.3101e-7**	**8.1728e-5**	10.8804	9	0.90%	139	**1.9612e-2**	6.4249	6	18	0.54%
organelle fission	41	**2.8488e-7**	**9.0954e-5**	2.3103	46	4.60%	60	**2.5433e-4**	2.2988	39	327	3.51%
rRNA metabolic process	45	**5.0827e-7**	**1.4785e-4**	2.9340	29	2.90%	108	**7.3315e-3**	2.4093	24	192	2.16%
translation	47	**6.6821e-7**	**1.8610e-4**	2.1941	48	4.80%	145	**2.4124e-2**	1.8603	36	373	3.24%
histone modification	49	**1.0794e-6**	**2.8834e-4**	2.0474	54	5.41%	146	**2.5247e-2**	1.8739	35	360	3.15%
nuclear division	50	**1.2173e-6**	**3.1870e-4**	2.3128	41	4.10%	79	**2.3914e-3**	2.1991	34	298	3.06%
RNA phosphodiester bond hydrolysis	51	**1.2504e-6**	**3.2094e-4**	3.3867	22	2.20%	165	**4.5828e-2**	2.4453	17	134	1.53%
cellular response to leukemia inhibitory factor	60	**2.3047e-6**	**5.0281e-4**	2.4401	35	3.50%	121	**1.3098e-2**	2.6895	18	129	1.62%
RNA export from nucleus	62	**2.8203e-6**	**5.9545e-4**	4.7211	14	1.40%	129	**1.5460e-2**	3.4522	12	67	1.08%
macromolecule methylation	66	**3.3209e-6**	**6.5864e-4**	2.3317	37	3.70%	174	**4.8040e-2**	2.1010	23	211	2.07%

The test set of 999 genomic regions picked 1,110 (5%) of all 21,395 genes.
GO Biological Process has 13,090 terms covering 17,925 (84%) of all 21,395 genes, and 1,163,819 term - gene associations.
13,090 ontology terms (100%) were tested using an annotation count range of [1, Inf].

GO Cellular Component (20+ terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

GREAT Overview News Use GREAT Demo Video How to Cite Help Forum Bejerano Lab, Stanford University

13,090 ontology terms (100%) were tested using an annotation count range of [1, Inf].

GO Cellular Component (20+ terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	*Binom FDR Q-Val*	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	*Hyper FDR Q-Val*	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage
intracellular ribonucleoprotein complex	15	**1.4692e-17**	**1.6592e-15**	2.3662	118	11.81%	23	**6.3196e-8**	1.9775	87	848	7.84%
ribonucleoprotein complex	16	**1.5828e-17**	**1.6758e-15**	2.3637	118	11.81%	24	**6.8021e-8**	1.9728	87	850	7.84%
chromosomal region	30	**5.4157e-9**	**3.0581e-7**	2.4504	53	5.31%	31	**7.3508e-6**	2.3748	43	349	3.87%
ribonucleoprotein granule	31	**1.8553e-8**	**1.0139e-6**	3.0125	35	3.50%	46	**6.4926e-3**	2.2970	23	193	2.07%
nuclear envelope	34	**1.5578e-7**	**7.7613e-6**	2.0132	65	6.51%	32	**3.3180e-5**	2.1367	48	433	4.32%
mitochondrial matrix	36	**4.1680e-7**	**1.9613e-5**	2.4807	39	3.90%	43	**4.9247e-3**	2.0002	33	318	2.97%
organelle inner membrane	38	**1.0999e-6**	**4.9030e-5**	2.0793	52	5.21%	62	**3.7022e-2**	1.6509	40	467	3.60%
telomerase holoenzyme complex	40	**2.0802e-6**	**8.8096e-5**	9.9491	8	0.80%	53	**1.7017e-2**	5.5071	6	21	0.54%
cytoplasmic ribonucleoprotein granule	41	**2.1491e-6**	**8.8796e-5**	2.6695	30	3.00%	61	**3.5153e-2**	2.1181	20	182	1.80%
MIS12/MIND type complex	43	**4.7649e-6**	**1.8772e-4**	37.8186	4	0.40%	54	**1.6800e-2**	14.4561	3	4	0.27%
ribosomal subunit	44	**6.0285e-6**	**2.3210e-4**	2.4849	31	3.10%	48	**9.9484e-3**	2.1820	24	212	2.16%
spindle microtubule	48	**2.5994e-5**	**9.1738e-4**	3.6445	15	1.50%	40	**2.5073e-3**	4.0774	11	52	0.99%
kinetochore	51	**5.9272e-5**	**1.9688e-3**	2.5597	23	2.30%	36	**8.1214e-4**	2.8985	20	133	1.80%
ribosome	52	**6.3400e-5**	**2.0654e-3**	2.1009	34	3.40%	52	**1.5042e-2**	2.0093	27	259	2.43%
mitochondrial ribosome	54	**7.7051e-5**	**2.4171e-3**	3.6931	13	1.30%	63	**4.6840e-2**	2.6586	12	87	1.08%
myelin sheath	56	**1.0724e-4**	**3.2440e-3**	2.1565	30	3.00%	60	**3.0674e-2**	2.0585	22	206	1.98%
histone acetyltransferase complex	61	**1.4169e-4**	**3.9349e-3**	3.1219	15	1.50%	57	**2.7431e-2**	2.8555	12	81	1.08%
protein acetyltransferase complex	63	**1.7219e-4**	**4.6301e-3**	2.9351	16	1.60%	59	**2.8859e-2**	2.6943	13	93	1.17%
mitotic spindle	64	**2.1219e-4**	**5.6165e-3**	2.7719	17	1.70%	50	**1.4202e-2**	2.9479	13	85	1.17%
condensed chromosome, centromeric region	94	**1.9212e-3**	**3.4622e-2**	2.2009	18	1.80%	42	**4.5088e-3**	2.8006	17	117	1.53%

The test set of 999 genomic regions picked 1,110 (5%) of all 21,395 genes.
GO Cellular Component has 1,694 terms covering 19,074 (89%) of all 21,395 genes, and 371,380 term - gene associations.
1,694 ontology terms (100%) were tested using an annotation count range of [1, Inf].

GO Molecular Function (10 terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

mitotic spindle	64	2.1219e-4	5.6165e-3	2.7719	17	1.70%	50	1.4202e-2	2.9479	13	85	1.17%
condensed chromosome, centromeric region	94	1.9212e-3	3.4622e-2	2.2009	18	1.80%	42	4.5088e-3	2.8006	17	117	1.53%

The test set of 999 genomic regions picked 1,110 (5%) of all 21,395 genes.
GO Cellular Component has 1,694 terms covering 19,074 (89%) of all 21,395 genes, and 371,380 term - gene associations.
1,694 ontology terms (100%) were tested using an annotation count range of [1, Inf].

GO Molecular Function (10 terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	Binom FDR Q-Val	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	Hyper FDR Q-Val	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage
RNA binding	1	8.6672e-25	3.5804e-21	2.0465	220	22.02%	1	5.1692e-13	1.9148	166	1,671	14.95%
ribonucleoprotein complex binding	3	3.1325e-14	4.3135e-11	5.1350	34	3.40%	6	1.7422e-6	3.8550	26	130	2.34%
single-stranded DNA binding	6	1.6962e-9	1.1678e-6	4.1008	27	2.70%	11	1.1263e-3	3.5045	18	99	1.62%
histone acetyltransferase activity	10	1.9017e-7	7.8560e-5	5.1323	16	1.60%	33	1.7852e-2	3.7197	11	57	0.99%
N-acyltransferase activity	11	2.9290e-7	1.1000e-4	3.8495	21	2.10%	45	4.6262e-2	2.6525	15	109	1.35%
peptide-lysine-N-acetyltransferase activity	12	3.0414e-7	1.0470e-4	4.9514	16	1.60%	35	2.3222e-2	3.5936	11	59	0.99%
peptide N-acetyltransferase activity	16	6.5540e-7	1.6922e-4	4.4008	17	1.70%	37	2.2729e-2	3.3521	12	69	1.08%
microtubule plus-end binding	17	1.2410e-6	3.0157e-4	8.8607	9	0.90%	42	3.3230e-2	7.4134	5	13	0.45%
ribosome binding	33	1.1685e-4	1.4627e-2	3.3430	14	1.40%	35	2.3222e-2	3.5936	11	59	0.99%
helicase activity	50	3.5894e-4	2.9655e-2	2.2540	23	2.30%	31	7.4070e-3	2.6771	20	144	1.80%

The test set of 999 genomic regions picked 1,110 (5%) of all 21,395 genes.
GO Molecular Function has 4,131 terms covering 17,189 (80%) of all 21,395 genes, and 227,341 term - gene associations.
4,131 ontology terms (100%) were tested using an annotation count range of [1, Inf].

Human Phenotype (no terms)

Global controls

Mouse Phenotype Single KO (13 terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	Binom FDR Q-Val	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	Hyper FDR Q-Val	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage
embryonic lethality prior to organogenesis	1	4.0047e-13	3.6723e-9	2.1701	103	10.31%	6	1.7414e-10	2.3810	84	680	7.57%
embryonic lethality between implantation and	4	2.1328e-8	4.8895e-5	2.4470	49	4.90%	14	9.5482e-5	2.6456	35	255	3.15%

GREAT Overview News Use GREAT Demo Video How to Cite Help Forum Bejerano Lab, Stanford University

4,131 ontology terms (100%) were tested using an annotation count range of [1, Inf].

Human Phenotype (no terms)

Global controls

Mouse Phenotype Single KO (13 terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	Binom FDR Q-Val	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	Hyper FDR Q-Val	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage
embryonic lethality prior to organogenesis	1	4.0047e-13	3.6723e-9	2.1701	103	10.31%	6	1.7414e-10	2.3810	84	680	7.57%
embryonic lethality between implantation and somite formation, complete penetrance	4	2.1328e-8	4.8895e-5	2.4470	49	4.90%	14	9.5482e-5	2.6456	35	255	3.15%
embryonic lethality between implantation and placentation	6	8.3265e-8	1.2726e-4	2.0633	64	6.41%	12	1.5407e-5	2.3671	49	399	4.41%
embryonic lethality between implantation and somite formation	8	1.0241e-7	1.1739e-4	2.2977	50	5.01%	16	2.3890e-4	2.4960	36	278	3.24%
abnormal blastocyst morphology	17	1.8747e-6	1.0112e-3	2.4268	36	3.60%	18	3.7475e-4	2.8109	28	192	2.52%
abnormal mitotic spindle assembly checkpoint	18	5.3236e-6	2.7121e-3	10.8016	7	0.70%	23	7.2931e-3	12.0467	5	8	0.45%
abnormal fibroblast physiology	21	6.9635e-6	3.0407e-3	2.1733	40	4.00%	28	3.1334e-2	2.2946	25	210	2.25%
absent costovertebral joint	24	1.2208e-5	4.6645e-3	70.8777	3	0.30%	31	4.1203e-2	19.2748	3	3	0.27%
abnormal inner cell mass morphology	39	4.6998e-5	1.1050e-2	3.0294	18	1.80%	27	2.5832e-2	3.2908	14	82	1.26%
absent inner cell mass proliferation	45	6.0828e-5	1.2395e-2	4.3900	11	1.10%	33	4.6609e-2	4.3368	9	40	0.81%
abnormal cell cycle	49	7.0851e-5	1.3259e-2	2.0157	37	3.70%	22	5.7866e-3	2.3786	29	235	2.61%
abnormal inner cell mass proliferation	62	1.6336e-4	2.4161e-2	3.0804	15	1.50%	30	3.4413e-2	3.5584	12	65	1.08%
embryonic lethality before implantation, complete penetrance	73	3.4500e-4	4.3337e-2	2.3648	21	2.10%	24	2.1894e-2	2.7535	19	133	1.71%

The test set of 999 genomic regions picked 1,110 (5%) of all 21,395 genes.
Mouse Phenotype Single KO has 9,170 terms covering 9,466 (44%) of all 21,395 genes, and 551,620 term - gene associations.
9,170 ontology terms (100%) were tested using an annotation count range of [1, Inf].

Mouse Phenotype (15 terms)

Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	Binom FDR Q-Val	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	Hyper FDR Q-Val	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage

GREAT Overview News Use GREAT Demo Video How to Cite Help Forum Bejerano Lab, Stanford University

Mouse Phenotype Single KO has 9,170 terms covering 9,466 (44%) of all 21,395 genes, and 551,620 term - gene associations.
9,170 ontology terms (100%) were tested using an annotation count range of [1, Inf].

Mouse Phenotype (15 terms) Global controls

Table controls: Export | Shown top rows in this table: 20 Set | Term annotation count: Min: 1 Max: Inf Set | Visualize this table: [select one]

Term Name	Binom Rank	Binom Raw P-Value	*Binom FDR Q-Val*	Binom Fold Enrichment	Binom Observed Region Hits	Binom Region Set Coverage	Hyper Rank	*Hyper FDR Q-Val*	Hyper Fold Enrichment	Hyper Observed Gene Hits	Hyper Total Genes	Hyper Gene Set Coverage
embryonic lethality prior to organogenesis	1	**3.1752e-12**	**3.0403e-8**	2.0505	108	10.81%	7	**8.2610e-11**	2.3371	89	734	8.02%
embryonic lethality between implantation and somite formation, complete penetrance	6	**5.6559e-8**	**9.0259e-5**	2.3440	50	5.01%	20	**9.6714e-5**	2.5700	36	270	3.24%
abnormal fibroblast physiology	9	**9.8299e-8**	**1.0458e-4**	2.3243	49	4.90%	29	**9.9304e-4**	2.4379	32	253	2.88%
embryonic lethality between implantation and somite formation	11	**3.5631e-7**	**3.1015e-4**	2.1812	51	5.11%	25	**3.4232e-4**	2.3852	37	299	3.33%
abnormal mitotic spindle assembly checkpoint	17	**2.2093e-6**	**1.2444e-3**	9.8673	8	0.80%	30	**1.0770e-3**	11.5649	6	10	0.54%
decreased erythrocyte cell number	19	**4.2710e-6**	**2.1524e-3**	2.1704	42	4.20%	47	**1.5438e-2**	2.2432	27	232	2.43%
abnormal blastocyst morphology	20	**5.0029e-6**	**2.3951e-3**	2.2592	38	3.80%	23	**2.1697e-4**	2.7405	30	211	2.70%
absent costovertebral joint	27	**1.2208e-5**	**4.3294e-3**	70.8777	3	0.30%	53	**2.5164e-2**	19.2748	3	3	0.27%
abnormal inner cell mass morphology	36	**2.6018e-5**	**6.9202e-3**	3.0568	19	1.90%	38	**6.2050e-3**	3.4419	15	84	1.35%
absent inner cell mass proliferation	47	**6.9529e-5**	**1.4165e-2**	4.3234	11	1.10%	59	**4.0433e-2**	4.1303	9	42	0.81%
embryonic lethality before implantation, complete penetrance	58	**1.1173e-4**	**1.8446e-2**	2.3961	24	2.40%	32	**1.9289e-3**	2.9044	22	146	1.98%
embryonic lethality before implantation	61	**1.3668e-4**	**2.1454e-2**	2.2713	26	2.60%	41	**6.8819e-3**	2.5774	23	172	2.07%
abnormal inner cell mass proliferation	65	**1.7757e-4**	**2.6157e-2**	3.0562	15	1.50%	55	**2.6525e-2**	3.4522	12	67	1.08%
abnormal embryonic epiblast morphology	72	**2.5968e-4**	**3.4533e-2**	2.3599	22	2.20%	50	**1.6659e-2**	3.4325	13	73	1.17%
abnormal bilaminar embryonic disc	88	**4.5352e-4**	**4.9346e-2**	2.2153	23	2.30%	44	**1.0826e-2**	3.4158	14	79	1.26%

The test set of 999 genomic regions picked 1,110 (5%) of all 21,395 genes.
Mouse Phenotype has 9,575 terms covering 9,654 (45%) of all 21,395 genes, and 705,265 term - gene associations.
9,575 ontology terms (100%) were tested using an annotation count range of [1, Inf].

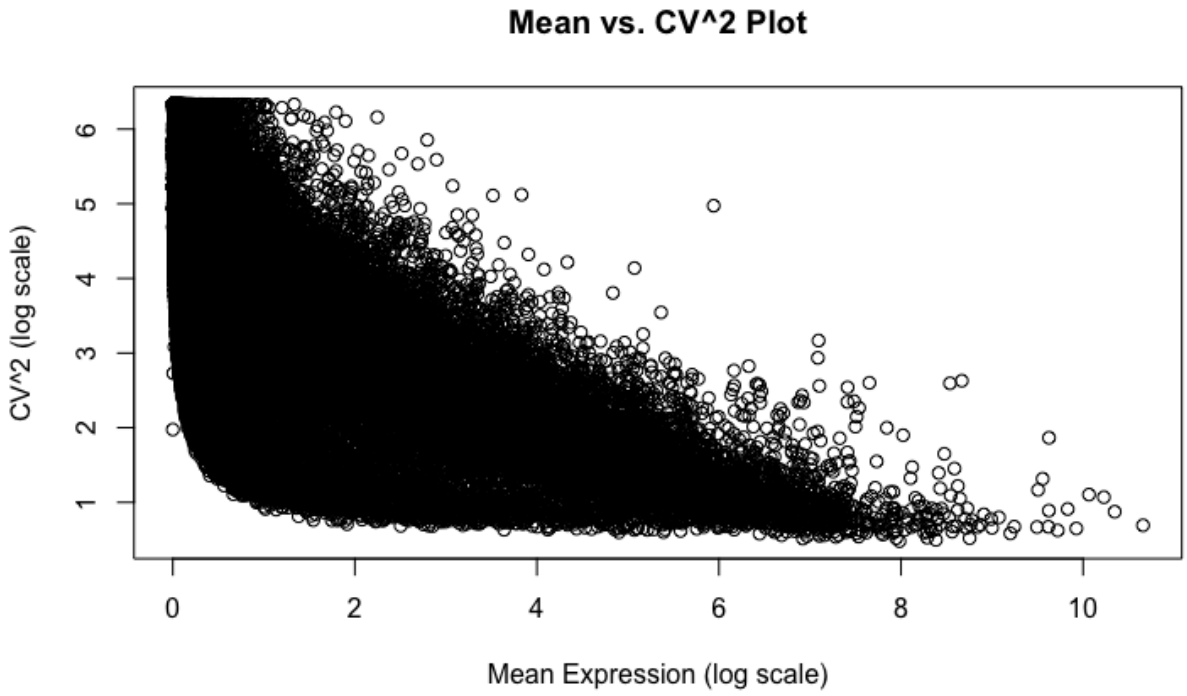
STANFORD SCHOOL OF MEDICINE Bejerano Lab STANFORD ENGINEERING

Terms of Use About Us Contact Us

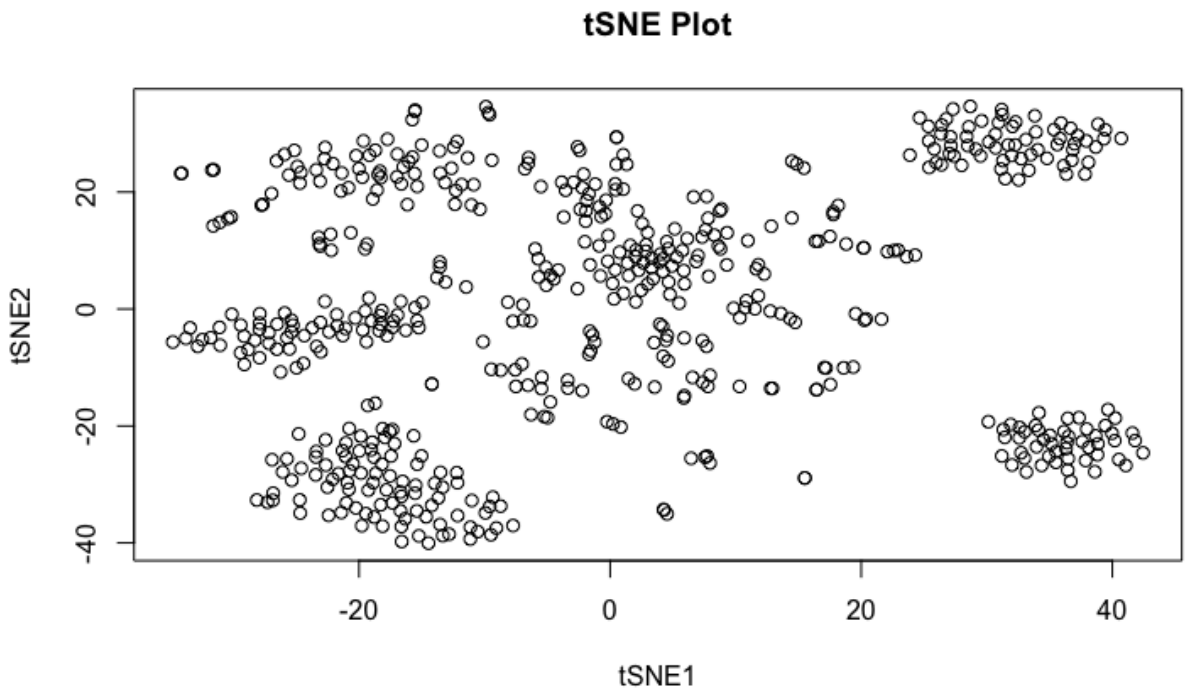
Submit

2.

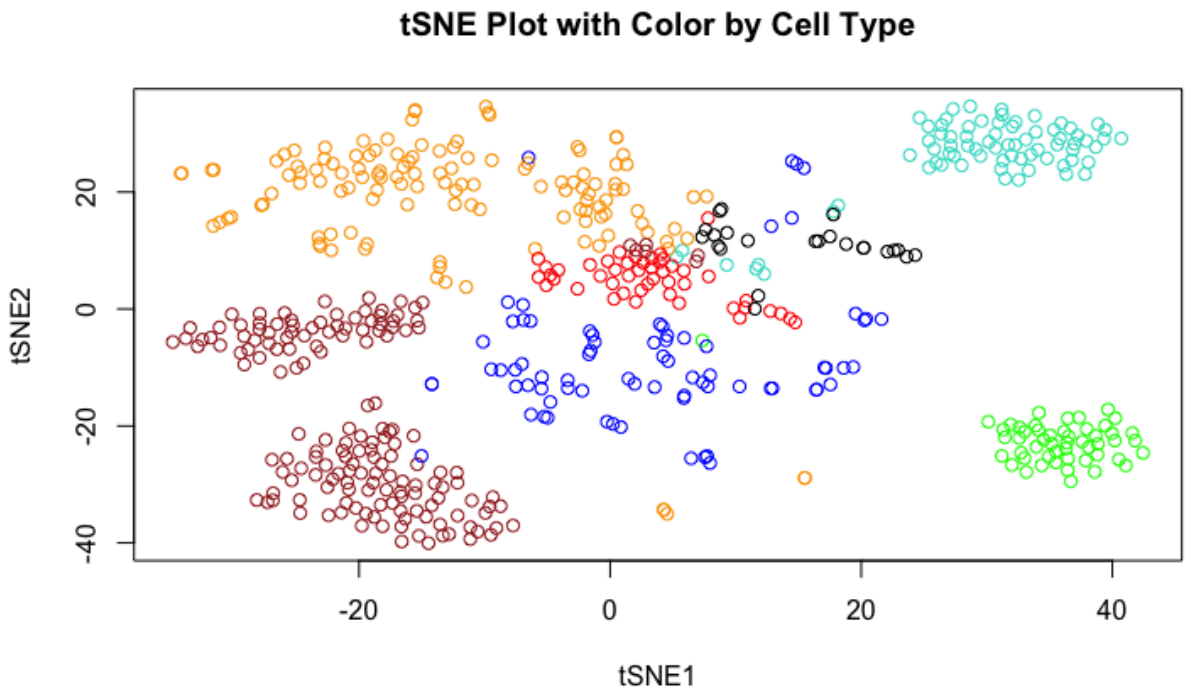
(a.)


Mean vs. CV^2 Plot
CV^2 (log scale)
1
2
3
4
5
6
0
2
4
6
8
10
Mean Expression (log scale)

(b.)



(c.)



The process of feature selection to pick highly variant genes involves iterating through the dataset and identifying genes that exhibit significant changes in expression levels across different cell types. In the initial stepP, a general approach to feature selection may have been applied, which could include methods like selecting genes based on differential expression or statistical significance without specifically considering variability across expression levels.

However, in the repeated feature selection process, we focus on identifying genes with high variability in expression levels within each bin of mean expression. This approach allows us to capture genes that may not have been selected in the initial stepP but are considered relevant due to their dynamic expression patterns.

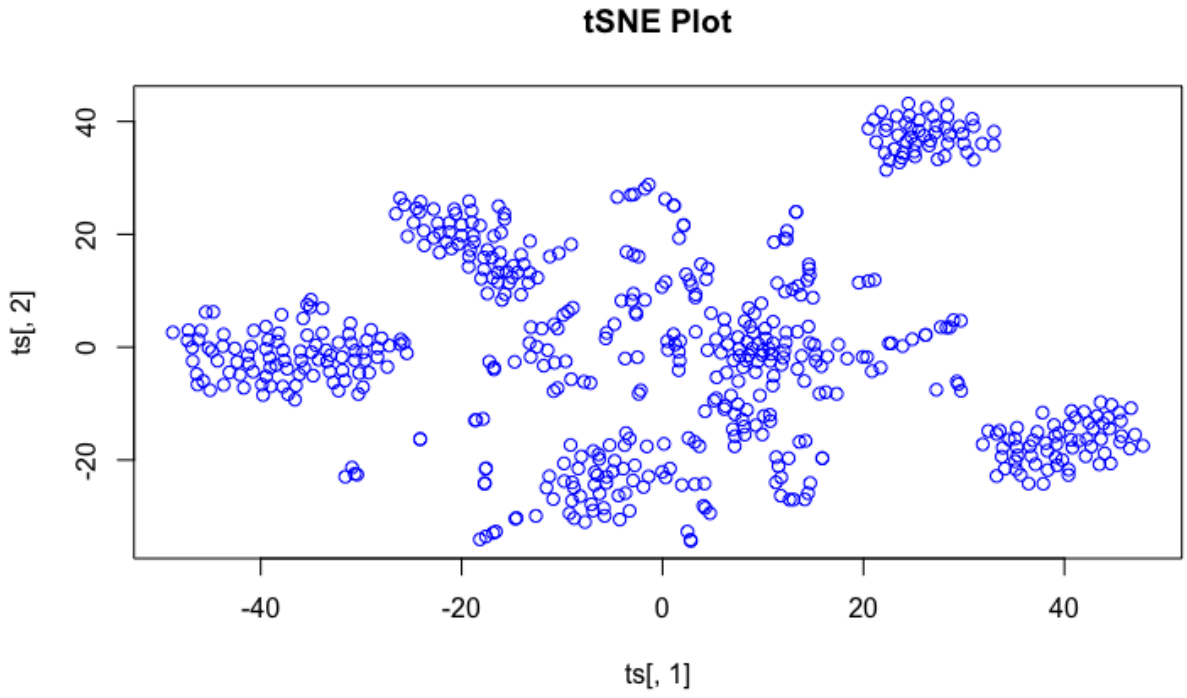
The coefficient of variation squared (CV2) is used as a metric to quantify the variability of gene expression within each bin. CV2 measures the relative dispersion of expression values, taking into account both the mean and standard deviation of expression levels. By calculating CV2 for genes within each bin of mean expression, we can identify genes that show pronounced changes in expression levels across different cell types or biological conditions.

The difference is observed in the repeated feature selection process compared to stepP which arised from the emphasis on variability in gene expression. While stepP may have focused on other criteria for feature selection, such as overall expression levels or statistical significance, the repeated process specifically targets genes with high CV2 values, indicating significant variability in expression.

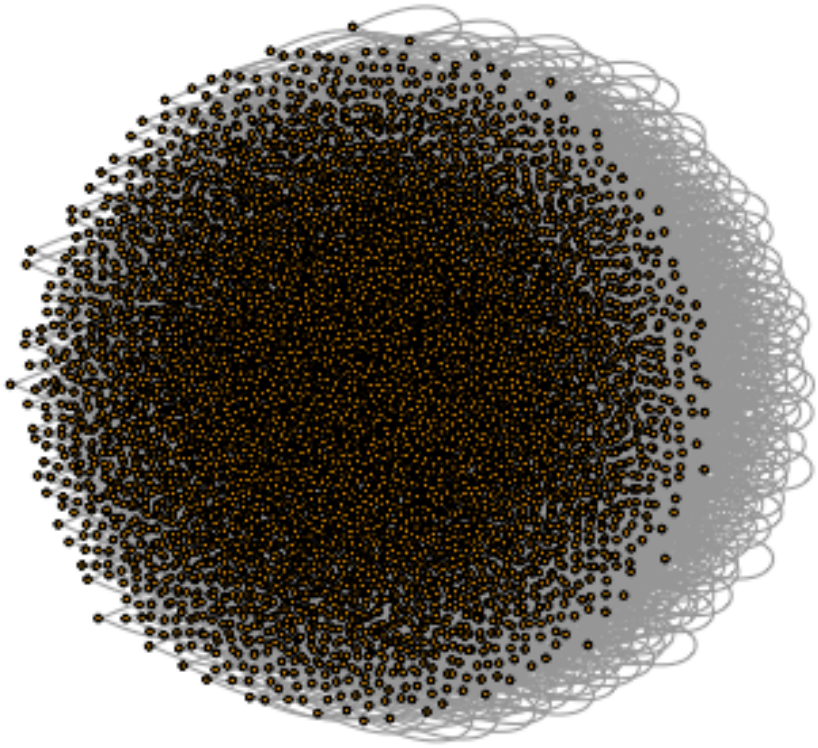
Therefore, the repeated feature selection process provides a more focused and targeted approach to identifying genes that exhibit dynamic expression patterns across different cell types. These highly variant genes represent potential candidates for further investigation into their functional roles and significance in cellular processes or disease mechanisms.

3.

(a.)



(b.)



Predicting a gene network and identifying the gene with the highest centrality involves interpreting the significance of these findings:

1. **Gene Network Prediction:**
    - Predicting a gene network involves inferring relationships or interactions between genes based on their expression patterns across cells.
    - Genes in the network are represented as nodes, and edges between nodes indicate inferred relationships, such as co-expression or regulatory interactions.
2. **Identification of Top Centrality Gene:**
    - The gene with the highest centrality score in the network is identified as the most central or influential gene.
    - Centrality measures, such as degree centrality, betweenness centrality, or closeness centrality, quantify the importance or influence of a gene within the network based on its connectivity to other genes.
3. **Meaning of Highest Centrality:**

- A gene with the highest centrality score signifies its prominence within the gene network.
- It indicates that the gene is highly connected to other genes and plays a crucial role in mediating interactions or regulating biological processes.
- In practical terms, this gene may act as a hub or key regulator in the network, influencing the expression of other genes or coordinating cellular functions.

4. **Biological Implications:**
    - Identifying the gene with the highest centrality provides valuable insights into gene regulatory networks and cellular processes.
    - This gene may represent a potential target for further experimental validation or functional studies to elucidate its role in cellular function, disease mechanisms, or response to stimuli.
    - Understanding the centrality of this gene can shed light on its importance in biological systems and help prioritize candidate genes for further investigation in research or clinical contexts.

In summary, identifying the gene with the highest centrality in a predicted gene network highlights its significance as a central player in regulating gene expression and coordinating cellular processes, offering valuable insights into gene function and regulatory mechanisms.

4.

Linear Discriminant Analysis (LDA) and the ROC-AUC maximizing detection filter (Hotelling Observer) are both techniques used in data analysis, but they serve different purposes and operate in different contexts.

Linear Discriminant Analysis (LDA):

- **Purpose:** LDA is a supervised dimensionality reduction technique used for classification. It finds the linear combination of features that best separates two or more classes.

- **Operation:** LDA works by projecting the data onto a lower-dimensional space while maximizing the separation between classes.
- **Application in Genomics:** LDA can be applied in genomics for tasks such as classifying different subtypes of diseases based on gene expression profiles. For example, in cancer genomics, LDA can help distinguish between different cancer types based on the expression levels of relevant genes.

ROC-AUC Maximizing Detection Filter (Hotelling Observer):

- **Purpose:** The Hotelling Observer is a method used in signal detection theory, particularly in medical imaging, to optimize detection performance by maximizing the area under the Receiver Operating Characteristic (ROC) curve.
- **Operation:** It involves designing a detection filter that maximizes the ability to discriminate between signal and noise.
- **Application in Genomics:** In genomics, the Hotelling Observer could be applied to tasks such as detecting genomic abnormalities or mutations against a background of normal genetic variation. For example, in analyzing copy number variation data, the Hotelling Observer could be used to identify regions of the genome that are significantly amplified or deleted compared to the normal baseline.

Differences:

Purpose:

- LDA is primarily used for dimensionality reduction and classification tasks, aiming to find a linear combination of features that best separates different classes.
- The Hotelling Observer, on the other hand, is specifically designed for optimizing detection performance by maximizing the area under the ROC curve, focusing on distinguishing signal from noise.

Nature of Input:

- LDA typically operates on labeled data, where the classes of observations are known a priori, and its goal is to find discriminant functions to separate these classes.
- The Hotelling Observer often deals with unlabeled data, focusing on maximizing the detection of signals against background noise without necessarily knowing the specific classes of observations.

Algorithmic Approach:

- LDA involves computing class means and covariance matrices to derive discriminant functions or axes that maximize separation between classes, often assuming Gaussian distributions.
- The Hotelling Observer employs statistical methods tailored for signal detection, aiming to design an optimal detection filter by considering the statistical properties of the signal and noise distributions.

Output Interpretation:

- LDA provides insight into the discriminant axes or components that best explain the variance between classes, allowing for visualization and interpretation of the class separation.
- The Hotelling Observer typically provides a detection score or statistic that quantifies the likelihood of a signal being present, which may not directly translate into interpretable features or components.

Application Contexts:

- LDA finds applications in various fields such as pattern recognition, machine learning, and genomics, where classification or dimensionality reduction tasks are prevalent.
- The Hotelling Observer is commonly used in signal processing, medical imaging, and detection theory contexts, where the emphasis is on maximizing detection sensitivity and specificity, especially in scenarios with low signal-to-noise ratios.

Similarities:

Discriminant Analysis Basis:

- Both LDA and the Hotelling Observer are rooted in discriminant analysis principles, aiming to discern patterns or signals from data.

Statistical Foundation:

- Both techniques rely on statistical methods and assumptions to model the distribution of data and make decisions based on discriminant functions or detection filters.

Optimization Objective:

- Although in different contexts, both LDA and the Hotelling Observer aim to optimize certain criteria: LDA seeks to maximize class separation, while the Hotelling Observer aims to maximize detection performance as measured by the ROC-AUC.

Decision Making:

- Both techniques involve making decisions based on analysis results, whether it's classifying observations into predefined classes (LDA) or detecting signals against noise (Hotelling Observer).

Application Flexibility:

- While they have distinct primary applications, both LDA and the Hotelling Observer can find utility across various domains and disciplines, depending on the nature of the data and the analysis goals.

Linear Discriminant Analysis (LDA) in Genomics:

Cancer Subtype Classification: LDA can classify cancer subtypes based on gene expression or DNA methylation data.
Drug Response Prediction: LDA can predict drug response in cancer patients using genomic profiles.
Genetic Disease Diagnosis: LDA assists in diagnosing genetic disorders by analyzing genomic variants or gene expression patterns.

ROC-AUC Maximizing Detection Filter (Hotelling Observer) in Genomics:

Copy Number Variation (CNV) Detection: Hotelling Observer detects CNVs by distinguishing signal from background noise.
Rare Variant Detection: It enhances the sensitivity of identifying disease-associated variants in rare variant analysis.
ChIP-seq Peak Detection: Hotelling Observer improves peak detection accuracy in ChIP-seq data by filtering out noise, aiding in identifying protein binding sites.

5.

ChIP-seq (Chromatin Immunoprecipitation followed by sequencing) is a powerful technique used to study protein-DNA interactions in the genome. However, several sources of bias can affect ChIP-seq data, potentially leading to inaccurate peak calls. Here are three common sources of bias and methods to mitigate them during peak calling:

Sequencing Bias:

- **Bias Description:** Sequencing bias arises from variations in the efficiency of sequencing different regions of the genome. Certain regions may be overrepresented or underrepresented due to factors such as GC content, mappability, or PCR amplification efficiency.
- **Mitigation Strategies:**

- Normalize read counts: Apply normalization methods such as reads per million (RPM) or reads per kilobase per million (RPKM) to adjust for differences in sequencing depth across genomic regions.
- Downsample reads: Randomly subsample reads to ensure uniform coverage across the genome and reduce the impact of sequencing bias.
- Use control samples: Employ control samples (e.g., input DNA or mock immunoprecipitation) to distinguish true ChIP-seq signals from sequencing artifacts.

GC Bias:

- **Bias Description:** GC bias refers to the uneven distribution of sequencing reads across genomic regions with varying GC content. High GC content regions may exhibit reduced sequence coverage due to PCR amplification biases or difficulties in sequencing.
- **Mitigation Strategies:**
  - GC content normalization: Normalize read counts based on the GC content of genomic regions to mitigate biases introduced by GC-rich or GC-poor regions.
  - GC-matched controls: Use control samples with GC content similar to the ChIP sample to account for GC bias during peak calling.
  - GC content weighting: Apply weights or correction factors to read counts based on the GC content of genomic regions to compensate for bias during peak calling.

Mapping Bias:

- **Bias Description:** Mapping bias arises from discrepancies in the efficiency of aligning sequencing reads to the reference genome across genomic regions. Regions with repetitive sequences, structural variations, or mapping ambiguities may exhibit lower mapping efficiency.
- **Mitigation Strategies:**
  - Use alternative mapping algorithms: Employ specialized mapping algorithms or aligners optimized for handling repetitive regions or complex genomic features to improve mapping accuracy.
  - Filter low-quality reads: Remove low-quality or ambiguously mapped reads from the analysis to reduce the impact of mapping bias on peak calling.
  - Adjust mapping parameters: Fine-tune mapping parameters (e.g., mismatch allowance, seed length) to optimize alignment accuracy and minimize mapping bias.

6.

The key differences between the binomial distribution and the negative binomial distribution lie in their underlying assumptions and the scenarios in which they are applied:

Binomial Distribution:

- The binomial distribution models the number of successes in a fixed number of independent Bernoulli trials.
- It is used when there are only two possible outcomes for each trial (success or failure), and the probability of success remains constant across all trials.
- Example: Flipping a fair coin multiple times and counting the number of heads obtained.

Negative Binomial Distribution:

- The negative binomial distribution models the number of successes until a specified number of failures occur in a sequence of independent Bernoulli trials.
- It is used when the number of trials required to achieve a certain number of successes is variable, and the probability of success remains constant across trials.
- Example: Counting the number of tails obtained before getting the third head when flipping a fair coin.

Why Negative Binomial Distribution is More Accurate for RNA-seq Read-Counts:

- **Overdispersion:** RNA-seq read-counts often exhibit overdispersion, where the variance exceeds the mean. This occurs due to biological variability, technical noise, and the Poisson sampling process inherent in RNA-seq data.
- **Flexibility:** The negative binomial distribution allows for greater flexibility in modeling overdispersed count data compared to the Poisson distribution, which assumes equal mean and variance.
- **Model Fit:** Negative binomial distribution provides a better fit to RNA-seq read-count data by allowing the variance to be larger than the mean, capturing the inherent variability more accurately.
- **Expression Heterogeneity:** RNA-seq data often exhibit varying levels of gene expression across samples, leading to different levels of dispersion. The negative

binomial distribution can account for this heterogeneity more effectively than the Gaussian or Poisson distributions.

Applicability to ChIP-seq Read-Counts:

- Similar to RNA-seq data, ChIP-seq read-counts can also exhibit overdispersion due to biological variability and technical noise.
- Therefore, the negative binomial distribution can also be more appropriate for modeling ChIP-seq read-counts, especially when analyzing enrichment of genomic sites for specific proteins or histone modifications.
- By using the negative binomial distribution, researchers can accurately capture the variability in ChIP-seq read-count data and make more reliable inferences about the presence and abundance of protein-DNA interactions across genomic sites.

7.

Hierarchical clustering is a popular method used in data analysis for grouping similar objects into clusters based on their features. There are two main types of hierarchical clustering: agglomerative and divisive. Here, I'll explain the agglomerative hierarchical clustering algorithm, which is the more commonly used approach:

Agglomerative Hierarchical Clustering Algorithm:

Initialization:

- Begin with each data point as its own cluster.
- Compute the distance matrix, which represents the pairwise distances between all data points.

Merge Closest Clusters:

- Identify the two closest clusters based on a chosen distance metric (e.g., Euclidean distance, Manhattan distance).
- Merge these two clusters into a single cluster, creating a hierarchical structure.

Update Distance Matrix:

- Recompute the distance matrix to reflect the distances between the newly formed cluster and all other clusters.
- Update the distance matrix based on a chosen linkage criterion (e.g., single linkage, complete linkage, average linkage).

Repeat:

- Repeat steps 2 and 3 iteratively until only a single cluster remains, or until a predefined number of clusters is obtained.

Hierarchical Structure:

- As clusters are merged, a hierarchical dendrogram is constructed, representing the merging process.
- The dendrogram visually illustrates the relationships between data points and clusters at different levels of granularity.

Linkage Criteria:

- **Single Linkage:** Compute the distance between the closest points in two clusters.
- **Complete Linkage:** Compute the distance between the farthest points in two clusters.
- **Average Linkage:** Compute the average distance between all pairs of points in two clusters.
- **Ward's Linkage:** Minimize the increase in variance when merging clusters.

Stopping Criteria:

- Hierarchical clustering can be stopped based on various criteria, such as a predetermined number of clusters, a threshold distance value, or a specific level of the dendrogram where clusters are cut.

Visualization:

- Hierarchical clustering results can be visualized using dendrograms, which display the hierarchical structure of the clusters.
- Dendrograms provide insight into the relationships between data points and clusters, allowing users to interpret the clustering results.

Complexity:

- The time complexity of agglomerative hierarchical clustering is
- $O(n^3)$ for n data points, primarily due to the computation of the distance matrix and merging of clusters.

- However, optimizations and heuristics can be employed to reduce the computational complexity in practice.

Hierarchical clustering is versatile and intuitive, making it a valuable tool for exploring and understanding the structure of complex datasets in various domains, including biology, social sciences, and computer science.

Pseudo code is as follows:

```
def hierarchical_clustering(data, distance_metric, linkage_criteria, num_clusters):
    # Initialize clusters: each data point is its own cluster
    clusters = [[point] for point in data]

    # Compute pairwise distance matrix
    distance_matrix = compute_distance_matrix(data, distance_metric)

     # Perform agglomerative clustering until desired number of clusters is reached
    while len(clusters) > num_clusters:
        # Find closest pair of clusters based on chosen linkage criterion
                    cluster1, cluster2 = find_closest_clusters(clusters, distance_matrix, linkage_criteria)

        # Merge closest clusters
        merged_cluster = merge_clusters(cluster1, cluster2)

        # Update clusters list
        clusters.remove(cluster1)
        clusters.remove(cluster2)
        clusters.append(merged_cluster)

        # Update distance matrix
         distance_matrix = update_distance_matrix(clusters, distance_matrix, linkage_criteria)

    return clusters

def compute_distance_matrix(data, distance_metric):
    # Compute pairwise distances between data points
    distance_matrix = []
    for i in range(len(data)):
```

```
        distances = []
        for j in range(len(data)):
            distances.append(distance_metric(data[i], data[j]))
        distance_matrix.append(distances)
    return distance_matrix

def find_closest_clusters(clusters, distance_matrix, linkage_criteria):
    # Find the pair of clusters with the smallest distance based on linkage
criterion
    min_distance = float('inf')
    closest_clusters = None
    for i in range(len(clusters)):
        for j in range(i + 1, len(clusters)):
            distance = calculate_cluster_distance(clusters[i], clusters[j],
distance_matrix, linkage_criteria)
            if distance < min_distance:
                min_distance = distance
                closest_clusters = (clusters[i], clusters[j])
    return closest_clusters

def calculate_cluster_distance(cluster1, cluster2, distance_matrix,
linkage_criteria):
    # Calculate distance between two clusters based on chosen linkage
criterion
    if linkage_criteria == 'single':
        # Single linkage: distance between closest points in two clusters
            return min(distance_matrix[i][j] for i in cluster1 for j in
cluster2)
    elif linkage_criteria == 'complete':
        # Complete linkage: distance between farthest points in two clusters
            return max(distance_matrix[i][j] for i in cluster1 for j in
cluster2)
    elif linkage_criteria == 'average':
        # Average linkage: average distance between all pairs of points in
two clusters
            return sum(distance_matrix[i][j] for i in cluster1 for j in
cluster2) / (len(cluster1) * len(cluster2))

def merge_clusters(cluster1, cluster2):
    # Merge two clusters into a single cluster
    return cluster1 + cluster2

def update_distance_matrix(clusters, distance_matrix, linkage_criteria):
```

```
    # Update distance matrix after merging clusters
      # Here, we can recompute the entire distance matrix, but optimized
implementations exist
    return compute_distance_matrix(clusters, distance_metric)

# Example usage:
data = [[1, 2], [3, 4], [5, 6], [7, 8]]
clusters = hierarchical_clustering(data, euclidean_distance, 'single', 2)
print(clusters)
```

8.

One approach for biclustering is the iterative biclustering algorithm, which aims to identify subsets of rows and columns in a data matrix that exhibit similar patterns. Here's an explanation of the iterative biclustering approach:

Iterative Biclustering Algorithm:

Initialization:

- Begin with the entire data matrix.
- Define initial row and column clusters based on some criteria (e.g., random assignment, hierarchical clustering).

Iterative Refinement:

- Alternate between optimizing row clusters and column clusters iteratively until convergence.
- In each iteration:
  - Fix row clusters and optimize column clusters: For each row cluster, find the subset of columns that best represent the cluster's pattern using techniques such as k-means clustering or non-negative matrix factorization (NMF).
  - Fix column clusters and optimize row clusters: For each column cluster, find the subset of rows that best represent the cluster's pattern using similar techniques.
- Update row and column clusters based on the optimized subsets.

Convergence Criteria:

- Define convergence criteria based on changes in row and column clusters between iterations (e.g., no significant change in cluster assignments).

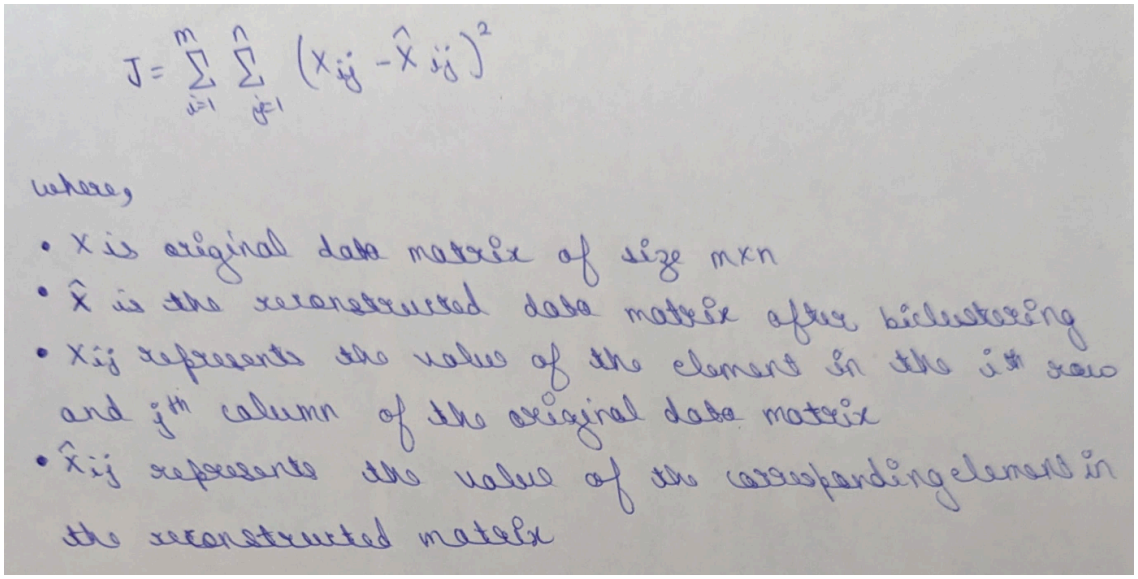
Output:

- The final output consists of the identified biclusters, which are subsets of rows and columns exhibiting coherent patterns.

Cost Function:

In biclustering, the cost function typically measures the quality of the biclusters and guides the optimization process. The specific cost function depends on the objectives of the biclustering algorithm and the characteristics of the data. One commonly used cost function in biclustering is based on minimizing the residual sum of squares (RSS) or the Frobenius norm between the original data matrix and the reconstructed matrix after biclustering.

Mathematically, the cost function J can be defined as:



Minimizing this cost function ensures that the biclustering algorithm identifies biclusters that capture the essential patterns in the data while minimizing reconstruction error.

By minimizing the cost function, the iterative biclustering algorithm iteratively refines row and column clusters to identify biclusters that exhibit coherent patterns in both rows and columns of the data matrix.

Cost Functions in Biclustering:

The goal of biclustering algorithms is to identify biclusters that best capture the underlying co-expression patterns within the data. Here are some common cost functions used for minimization:

- **Sum of Squared Errors (SSE):** Measures the total squared difference between data points within a bicluster and the bicluster's mean value. Lower SSE indicates a tighter co-expression pattern.
- **Cluster Variance:** Similar to SSE, but calculates the variance within each bicluster instead of the total squared error. Minimizing variance promotes biclusters with low variability.
- **Correlation-based Measures:** These functions utilize correlation coefficients (e.g., Pearson, Spearman) to assess the strength and direction of linear relationships between rows and columns within a bicluster. Maximizing correlation promotes biclusters with strong co-expression patterns.
- **Information Gain:** This metric measures the information gained about one variable (gene) by knowing the value of another (condition). It helps identify biclusters where rows (genes) share similar expression patterns across specific columns (conditions).

Pseudo Code:

```
def iterative_biclustering(data_matrix, num_row_clusters, num_col_clusters, max_iterations, convergence_threshold):
    # Initialize row and column clusters
    row_clusters = initialize_row_clusters(data_matrix, num_row_clusters)
    col_clusters = initialize_col_clusters(data_matrix, num_col_clusters)

    # Initialize iteration counter
    iteration = 0

    # Iterate until convergence or maximum iterations reached
    while iteration < max_iterations:
        # Update column clusters given fixed row clusters
                updated_col_clusters  =  update_col_clusters(data_matrix, row_clusters)

        # Update row clusters given fixed column clusters
                updated_row_clusters  =  update_row_clusters(data_matrix, updated_col_clusters)

        # Check convergence based on changes in cluster assignments
```

```
                if  check_convergence(row_clusters,  updated_row_clusters,
col_clusters, updated_col_clusters, convergence_threshold):
            break  # Convergence achieved, exit loop

        # Update clusters for next iteration
        row_clusters = updated_row_clusters
        col_clusters = updated_col_clusters

        # Increment iteration counter
        iteration += 1

    # Output final biclusters
            biclusters   =   extract_biclusters(data_matrix,   row_clusters,
col_clusters)
    return biclusters

# Function to initialize row clusters
def initialize_row_clusters(data_matrix, num_row_clusters):
    # Use k-means or random initialization to create initial row clusters
    # Return initial row cluster assignments
    return initial_row_clusters

# Function to initialize column clusters
def initialize_col_clusters(data_matrix, num_col_clusters):
       # Use k-means or random initialization to create initial column
clusters
    # Return initial column cluster assignments
    return initial_col_clusters

# Function to update column clusters given fixed row clusters
def update_col_clusters(data_matrix, row_clusters):
     # For each row cluster, find the subset of columns that best represent
the cluster's pattern
      # Use k-means or non-negative matrix factorization (NMF) to update
column clusters
    # Return updated column cluster assignments
    return updated_col_clusters

# Function to update row clusters given fixed column clusters
def update_row_clusters(data_matrix, col_clusters):
     # For each column cluster, find the subset of rows that best represent
the cluster's pattern
     # Use k-means or non-negative matrix factorization (NMF) to update row
```

```
clusters
    # Return updated row cluster assignments
    return updated_row_clusters

# Function to check convergence based on changes in cluster assignments
def check_convergence(old_row_clusters, new_row_clusters, old_col_clusters,
new_col_clusters, threshold):
    # Compute changes in cluster assignments for rows and columns
    row_changes = compute_changes(old_row_clusters, new_row_clusters)
    col_changes = compute_changes(old_col_clusters, new_col_clusters)

    # Check if changes are below convergence threshold
    if row_changes < threshold and col_changes < threshold:
        return True  # Convergence achieved
    else:
        return False  # Not converged

# Function to extract biclusters from row and column clusters
def extract_biclusters(data_matrix, row_clusters, col_clusters):
    # Extract subsets of rows and columns corresponding to each bicluster
    # Return biclusters as a list of tuples or data structures
    return biclusters

# Function to compute changes in cluster assignments
def compute_changes(old_clusters, new_clusters):
    # Compute changes in cluster assignments between old and new clusters
    # Return a measure of change (e.g., sum of squared differences)
    return changes
```